Electronic Research Archive

http://www.aimspress.com/journal/era

ERA, 31(8): 5056–5082.
DOI: 10.3934/era.2023259
Received: 31 May 2023
Revised: 11 July 2023
Accepted: 11 July 2023
Published: 14 July 2023

Research article

ODTC: An online darknet traffic classification model based on multimodal self-attention chaotic mapping features

Jiangtao Zhai, Haoxiang Sun, Chengcheng Xu* and Wenqian Sun

School of Electronics and Information Engineering, Nanjing University of Information Science and Technology, Nanjing 210044, China

* **Correspondence:** Email: xuchengcheng9@163.com.

Abstract: Darknet traffic classification is significantly important to network management and security. To achieve fast and accurate classification performance, this paper proposes an online classification model based on multimodal self-attention chaotic mapping features. On the one hand, the payload content of the packet is input into the network integrating CNN and BiGRU to extract local space-time features. On the other hand, the flow level abstract features processed by the MLP are introduced. To make up for the lack of the indistinct feature learning, a feature amplification module that uses logistic chaotic mapping to amplify fuzzy features is introduced. In addition, a multi-head attention mechanism is used to excavate the hidden relationships between different features. Besides, to better support new traffic classes, a class incremental learning model is developed with the weighted loss function to achieve continuous learning with reduced network parameters. The experimental results on the public CICDarketSec2020 dataset show that the accuracy of the proposed model is improved in multiple categories; however, the time and memory consumption is reduced by about 50%. Compared with the existing state-of-the-art traffic classification models, the proposed model has better classification performance.

Keywords: darknet traffic; deep learning; logistic chaos map; multi-head self-attention mechanism; online parameter update

1. Introduction

The darknet can meet the needs of Internet users for identity concealment, and users can only access it through specific anonymity tools. Nowadays, the commonly used anonymous tools include Tor, I2P, VPN, JAP, and others [1]. These anonymous tools can be used to hide the user's identity information and counter traffic analysis technology. While protecting user privacy, the darknet also brings challenges to the network order. Accurately and effectively classifying darknet traffic has positive and

important significance in the field of network security [2].

Currently, common methods for traffic classification includes port number-based, machine learning-based and deep learning-based [3]. The port number-based identification method depends on the corresponding relationship between the port number and the application. With the gradual maturity of port obfuscation technology, it is difficult to identify the corresponding relationship between ports and applications. This will lead to a serious decline in the method's effectiveness. The identification method based on deep packet inspection (DPI) [4] implements traffic classification by defining regular expressions for different categories. The DPI method is effective for plaintext traffic and is useless for ciphertext traffic. With the increasing proportion of encrypted traffic, this method gradually fails [5].

At present, many researchers adopt traditional machine learning algorithms based on artificial features to solve the darknet traffic classification problem. For instance, several conventional machine learning methods, such as the Support Vector Machine (SVM), Naive Bayes (NB), Random Forest (RF), Decision Tree (DT), and others (Hu et al. [6], 2020; Montieri et al. [1], 2019), have already been applied to it. However, these methods either rely heavily on hand-crafted features or resort to a time-consuming feature selection process [7]. Thus, they may lead to unstable performance when dealing with different network environments.

As a further improvement, deep learning methods, such as Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), have also been introduced to automatically extract high-level feature representations from network traffic (Bayat et al. [8], 2021; Hu et al. [9], 2021; Lashkari et al. [10], 2020; Lin et al. [11], 2021; Kim et al. [12], 2022). These methods significantly reduce the reliance on expertise and improve the generalizability of different network environments. The main purpose of these methods is to extract new, more expressive, meaningful and distinctive features from the original packet through automated methods.

Although the above methods can achieve pretty classification performance, there are still many deficiencies. They often ignore the relationship between different features and perform poorly when darknet user behavior is similar [13]. The nature of the used features is relatively simple, and the models cannot accurately fit the nonlinear relationship between the input and output. The models need to be retrained when faced with new traffic types [14]. Some user behaviors in the darknet are too similar, making it necessary to extract detailed features to enhance their representation.

To solve the above problems, we propose an online identification method of darknet traffic based on multi-modal self-attention chaotic features. This method combines the statistical features of darknet traffic and load features to reconstruct the feature mode of traffic to obtain richer feature, and introduces incremental learning, which can continuously learn new traffic categories without retraining the model. The proposed method accurately classifies user behavior in the darknet and meets the online parameter update requirement of new traffic classes. The main contributions of this paper can be summarized as follows.

1) We propose an end-to-end traffic classification model based on CNN-BiGRU and MLP. The model utilizes the feature of automatic feature extraction in-depth learning and combines multimodal inputs of traffic images and text. On one hand, the traffic payload is encoded and input into CNN-BiGRU to obtain spatio-temporal features. On the other hand, MLP is used to process traffic text features. This method combines different modal features to enhance the model's representation ability and improve the classification accuracy.

2) We introduce a local feature amplification module that is composed of a multi-head self-attention

mechanism and a Logistic chaotic map. The multi-head self-attention mechanism can look for the correlation between local and global features, while the Logistic chaotic map amplifies fuzzy features. 3) We use an incremental learning model and weighted loss function to continuously learn new traffic categories without retraining the model. The model is trained with partial samples of old and new classes to overcome the parameter forgetting problem of deep learning models. Moreover, we introduce a deviation correction layer to deal with the imbalance between old and new sample classes. This reduces network parameters and saves training time and resource consumption, significantly enhancing the model's online update capability. The experimental results on public datasets CICDarknet2020 and Darknet2020 demonstrate that the proposed method outperforms existing methods in the classification of darknet encrypted traffic.

The rest of the paper is organized as follows: Section 2 summarizes the current research achievements in the analysis of darknet encrypted traffic and identifies the shortcomings of existing studies. Section 3 provides a detailed description of the proposed method for analyzing darknet encrypted traffic. Section 4 presents the dataset used in this study, along with the specific experimental parameters and analysis of the experimental results. Finally, we conclude this paper in Section 5.

2. Related works

In recent years, researchers have carried out a lot of fruitful work in darknet encrypted traffic analysis. Shahbar and Zincir-Heywood [15] proposed to use the C4.5 decision tree based on the statistical characteristics of network flow to analyze the user behavior traffic on the I2P network. Rao et al. [16] proposed an unsupervised method based on gravitational clustering to identify Tor anonymous traffic from normal network traffic. Iliadis and Kaifas [17] conducted a comparative study on feature selection and classification models, and the important flow statistical features are selected to accurately classify the CICDarknet2020 darknet traffic dataset.

Researchers tend to use deep learning methods for darknet traffic classification because of their powerful nonlinear learning capabilities which can automatically extract deep features. Sarwar et al. [18] adopted the improved convolutional recurrent neural network CNN-LSTM and CNN-GRU deep learning methods for darknet traffic classification to complete the application identification task. In addition, they compared the performance between traditional machine learning and deep learning methods. The results show that deep learning is more accurate than traditional machine learning methods in classifying darknet traffic. Shapira and Shavitt [19] intercepted the network flow over a period of time, normalized it into a two-dimensional histogram of arrival time and packet size, and uses 2DCNN to classify it. Lashkari et al. [10] proposed the DeepImage method, which used a tree classifier to select important statistical features for one-hot encoding and converted them to grayscale images, and finally 2DCNN was used to classify darknet traffic.

To improve the representation ability of darknet traffic features, spatiotemporal feature fusion methods and attention mechanisms are gradually applied to classification models. Yao et al. [20] proposed an LSTM network with a hierarchical attention mechanism. Xie et al. [21] proposed the HSTF-Model to extract spatial and temporal features at the packet level and flow level. Hassan et al. [22] proposed a deep learning model based on CNN and an improved LSTM network, which used CNN to extract spatial features from network traffic data, and used LSTM to extract temporal features from the above features extracted by CNN. Kanna and Santhi [23] proposed a hybrid model of improved CNN and

multi-scale LSTM networks for extracting spatiotemporal features from input network streams. Liu et al. [24] proposed a Bi-GRU network with an attention mechanism as a lightweight model to classify traffic. Angelo and Palmieri [25] adopted the method of spatiotemporal feature fusion to enhance the representativeness of traffic features, and at the same time introduced an attention mechanism to increase the weight of the features that contribute the most to traffic classification. However, these methods do not extract the intrinsic connection between local and global features at different locations, but only use the attention mechanism to assign weights to feature vectors. The traffic generated by some different user behaviors is very similar, and it is difficult to extract fuzzy features that affect the classification performance. Lopez-Martin et al. [26] proposed a novel MSIF method in response to the counterintuitive results that may arise from highly conflicting evidence in multisource information fusion. This method is based on a newly defined generalized evidential divergence measure among multiple sources of evidence and provides a new approach for extracting fuzzy features.

The types of dark network applications are complex, constantly changing and growing, so the capability update online is crucial to the model. However, most existing models can only update parameters offline. Therefore, researchers began to reduce the parameter update time of the model to make it has certain adaptability. Xiao [27] proposed to extract the port number, packet payload length, packet interval time, window size and other attributes of the first 20 packets of the data stream form a 20×6 matrix, and input it into the CNN-LSTM model. Vu et al. [28] selected only 22 features to characterize the flow and used a generative adversarial network CGAN to address the class imbalance problem. Wang et al. [29] truncated the first 784 bytes of the flow and constructed an end-to-end model based on CNN for online traffic classification. Liu et al. [30] characterized the network flow with the long sequence of the first 128 packets in the flow and input it into the LSTM to identify the flow online. Although the classification efficiency is improved by the methods above, it is difficult to realize the online update of the model parameters. After the traffic class changes, the model needs to learn from scratch, which consumes a lot of time and memory. In addition, due to memory constraints, it is not possible to retain a large number of samples from the old class, resulting in an imbalance between the new and old classes. During the fine-tuning process, the model expands its knowledge by learning from samples of the new class while trying to preserve its understanding of the old class as much as possible. However, limited memory capacity means that only a limited number of samples from the old class can be retained. This may lead to difficulties in handling the old class by the model, as it does not have enough samples to accurately represent and distinguish these classes. This imbalance will directly affect the performance of the model.

In order to solve the above problems, the traffic is input into the model in a multimode way [31]. Based on CNN-Bi-GRU network and MLP network, spatiotemporal features and high-level abstract features are extracted. We integrate the multi-head self-attention mechanism [32] into the feature extraction process, to make full use of the internal relationship of spatiotemporal features extracted at different locations. Logistic chaotic mapping amplifies ambiguous features, so it is helpful to obtain highly discriminative features for fine-grained darknet traffic classification. In this paper, convolution cores with different sizes are used in the network, and multiple granularity features are fused to classify the dark network traffic more accurately. Furthermore, in order to facilitate the online update of the model, we adopt the incremental learning [33] method. First, inheritance learning is performed using a weighted loss function. Then, a deviation correction layer [34] is introduced to deal with the imbalance problem between old and new class samples. This approach significantly improves the online update

efficiency of the proposed model.

3. The proposed method

The overall framework of the proposed model is shown in Figure 1, which mainly includes three modules: a multimodal reconstruction module, a feature learning module and an online update module. The detailed descriptions of each module are shown as follows.

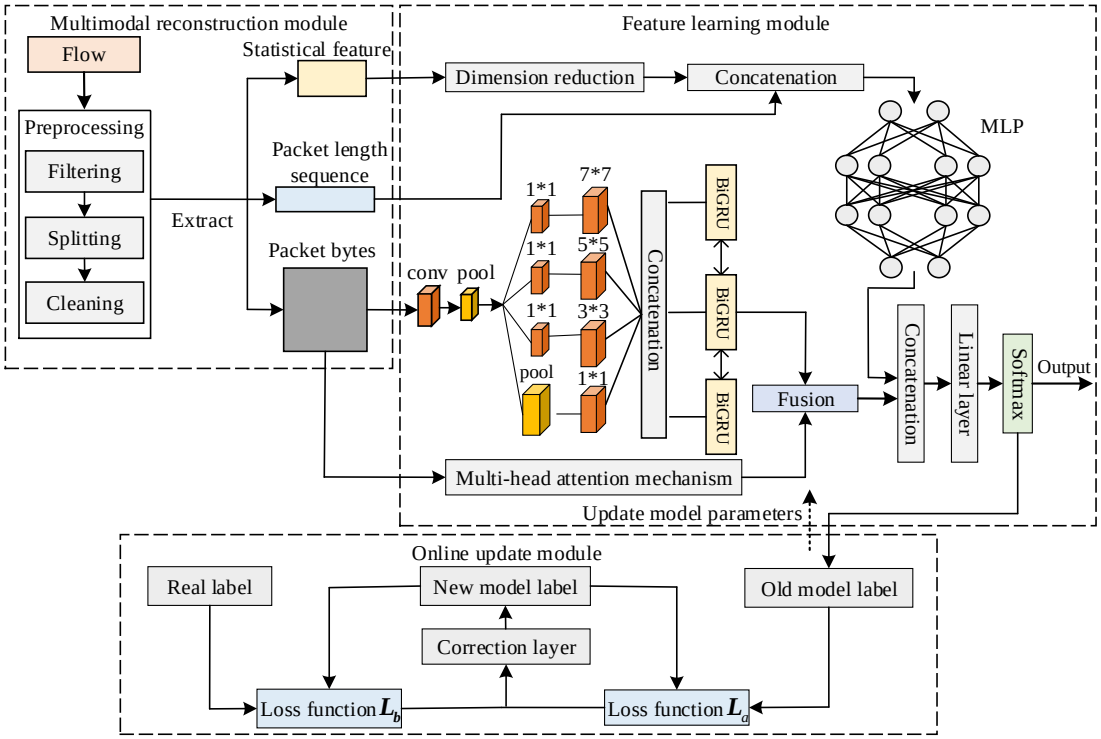


Figure 1. The framework of the proposed scheme.

3.1. Model overview

The multimodal reconstruction process primarily consists of three steps. 1) The packet is divided into flows based on the quintuple information. 2) The obtained flows are reconstructed into two feature modalities: one containing the original payload content and the other containing statistical information. 3) Both feature modalities are jointly inputted into the flow feature learning module.

The feature learning module consists of a Multilayer Perceptron (MLP), CNN-BiGRU network, multi-head self-attention mechanism, and Logistic chaos mapping. It mainly involves four steps. 1) The statistical features are fed into the MLP network to extract abstract high-level features. 2) The payload content is inputted into the CNN-BiGRU network to extract spatial feature vectors from the convolutional layer, which are then fused with the output of the BiGRU layer. 3) The multi-head self-attention mechanism is employed to capture the intrinsic relationships between temporal and spatial features. 4) Logistic chaos mapping is used to clarify and disentangle the fuzzy features, resulting in chaotic feature vectors.

The online classification and update module consists of three steps. 1) Utilize the parameters of the feature learning module to predict unknown samples. 2) Employ a weighted loss function to handle new classes. 3) Introduce a deviation correction layer to address the imbalance between new and old samples. When a new class emerges, the model performs small-scale incremental training to update its parameters.

Table 1. The symbol meaning table.

Symbol	Remark
B	The bytes of network flow with only payload content
N	Number of packets in network flow for text features extraction
X_c	Content features
X_t	Text features
H	Number of heads in the multi-head self-attention module
K	Key matrix of the self-attention mechanism
Q	Query matrix of the self-attention mechanism
V	Value matrix of the self-attention mechanism
F_c	Output of the content feature learning module
F_t	Output of the text feature learning module
F_m	Output of the multi-head self-attention mechanism
Y	Ture label of ground truth presented in a one-hot encoding vector
Y_p	Predicted probability vector of being a specific application type

3.2. Traffic multimodal reconstruction module

The traffic multimodal reconstruction starts with traffic pre-processing, which includes three steps: traffic filtering, traffic segmentation, and traffic cleaning. The purpose of traffic filtering is to discard irrelevant packets such as damaged packets, repeated packets and protocol packets that are not related to classification. Traffic segmentation combines packets with the same five-tuple information (source and destination IP and port can be interchangeable) into flow samples. Traffic cleaning process randomizes the MAC and IP addresses of the samples to avoid overfitting. The anonymous network flows processed by the above steps are used as traffic samples for the experiments in this paper.

The deep learning models can handle multiple data modalities, each contains different information attributes. As shown in Figure 2, multimodal inputs can be processed to obtain more comprehensive features, resulting in stronger traffic representation. The multi-modal input module includes three types of traffic input forms: traffic statistical features, intra-flow packet length sequences, and raw bytes of data packets.

1) Traffic statistical characteristics

Traffic statistical features are extracted from bidirectional network traffic samples. When establishing a communication, the transmission direction of the first data packet is considered as the upstream traffic, and the vice versa is considered as the downstream. As shown in Table 2, these statistical features have better capabilities to characterize the pre-filtered traffic, such as Flow IAT (inter-arrival time between packets in a flow), and Idle time (time interval between flows). For different types of traffic, such as email, chat and VoIP, the interaction frequency varies, resulting in different Flow IAT values.

The idle time also varies for different types of traffic; for example, browsing behavior generates traffic where the browser typically requests multiple resources from servers and needs to frequently send and receive packets, resulting in shorter transmission intervals. While, for P2P traffic, data transmission between nodes usually waits for responses from other nodes, resulting in longer transmission intervals. These statistical features can reflect the characteristics of different types of traffic and can be used as the basis for traffic classification.

2) Characteristics of packet length sequence within a flow

Python and its Dpkt tool library are used to parse the samples to get the packet length sequence in network flows. To explore the impact of different packet sequence lengths on traffic classification results, different length of packet sequence within flows are selected for processing by the feature learning module. The packet length sequence of different types of traffic has certain regularities, which often reflect the characteristics of information interaction during communication. The P2P traffic typically involves large file transfers, which need to be split into smaller data blocks, and the packet length sequence typically follows a long-tailed distribution. The browsing and chatting traffic are relatively low, and the packet length sequence is usually smooth and continuous. The packet length sequence of file transfer traffic follows a bimodal distribution, with a relatively large number of both large and small packets. Although some traffic features are hidden after obfuscation and encryption, this processing usually only adjusts the data packets to some extent and does not completely change the packet length sequence features. Therefore, packet length sequences can still be used for traffic classification.

Table 2. Statistical characteristics.

Name	Description
(Max/Min/Mean/Std) Idle	Flow Idle time (max, min, mean, std)
Flow Bytes/s	Bytes transferred per second
(Max/Min/Mean/Std)Total/Fwd/Bwd Packet Length	Packet length (max, min, mean, std)
Flow Duration	Flow duration
PSH/ACK Flag Count	The number of ACK, PSH packets in TCP flow
Flow Packets/s	The number of packets transmitted per second
(Total/Max/Min/Mean)Fwd/Bwd Flow IAT	Packet interarrival time (max, min, average and total sum)
Total Fwd/Bwd Packets	Total amount of forward and backward data packets
Fwd/Bwd Init Win Bytes	Forward and backward initial window size

3) Original byte characteristics of packets

In the real network environment, the length of each flow is different, however, the CNN-BiGRU model requires a consistent size for the input data. We intercept a fixed number of bytes for this. The maximum length of a packet is 1,500 bytes, as defined by the maximum transmission unit in the network, thus, we choose to intercept the header of 1,024 bytes. The first 1024 bytes of the header are not only convenient for model input, but also can reflect the traffic characteristics more accurately. The previous bytes contain more valuable information than the later bytes, which are used to establish connections and exchange state information between the two communicating parties.

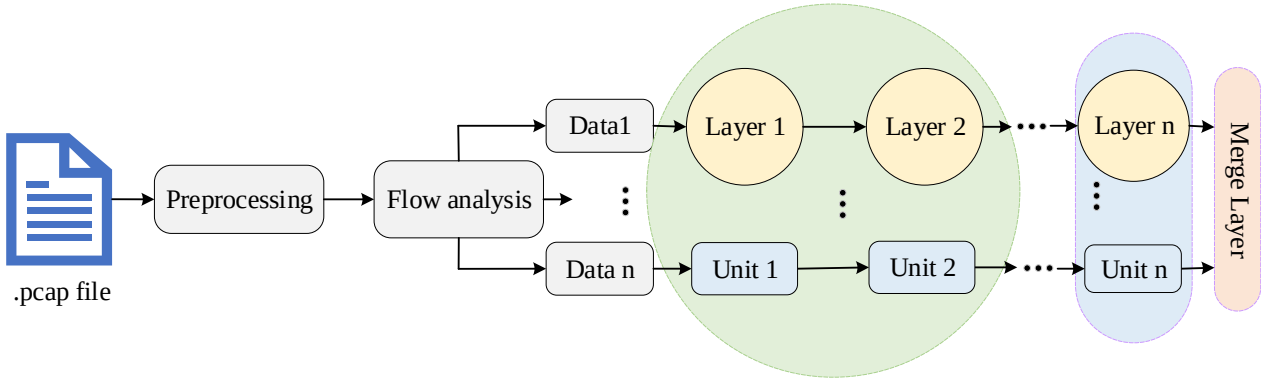


Figure 2. Multimodal input module.

Network traffic has a hierarchical structure, including packet-level and traffic-level, which contains rich information. The traffic statistics, length sequences of packets in the flows, and raw bytes of packets represent different information properties, which are the three traffic modes. Fully mining and integrating the information in these three modes for classification tasks can make the features more comprehensive and representative. Moreover, single traffic information may be affected by network environment fluctuations, and multimodal traffic information can enhance the robustness of the model.

3.3. Feature learning module

3.3.1. Feature engineering

The XGBoost algorithm is always used to evaluate the importance of flow features in order to reduce the feature dimension [35]. The input vector $X = \{X_1, X_2, X_3, \cdots X_n\}$ uses different features and values to split the dataset. The algorithm structure is similar to a tree model. When looking for a certain feature and its corresponding size to divide the dataset, the goal is to maximize the gain of the objective function. The features corresponding to the nodes with the largest gain should be sorted in order to obtain the important feature vector $X = \{X_1, X_2, X_3, \cdots X_n\}, m < n$ after dimensionality reduction. The gain function can be formalized as

$$Gain = \frac{1}{2}[\frac{G_L^2}{H_L + \lambda} + \frac{G_R^2}{H_R + \lambda} - \frac{(G_L + G_R)^2}{H_L + H_R + \lambda}] - \gamma \tag{3.1}$$

$G_L^2/(H_L + \lambda)$ and $G_R^2/(H_R + \lambda)$ are the gain scores for the left and right subtrees. $(G_L + G_R)^2/(H_L + H_R + \lambda)$ is the undivided gain fraction. The symbol γ represents the cost of complexity and *gain* function is obtained by the difference between the objective functions before and after the split, which can be denoted as

$$Gain = Object_{old} - Object_{new} \tag{3.2}$$

The *Object* function can be denoted as

$$Object = \sum_{i=1}^{n} l(y_i, \hat{y}^{(k)}) + \sum_{k=1}^{k} \Omega(f_k) \tag{3.3}$$

In Eq (3.3), $\sum_{i=1}^{n} l(y_i, \hat{y}^{(k)})$ is the loss function, which indicates the difference between the actual value and the predicted one. $\sum_{k=1}^{k} \Omega(f_k)$ is the regularization function that controls the complexity as a penalty term. y_i is the true value, $\hat{y}^{(k)}$ is the predicted value of the k-th tree. The objective function of training the k-th tree be $Object_k$. The parameters of the $K-1$th tree are regarded as known constants, and the objective function can be formalized as

$$Object_k = \sum_{i=1}^{n} l(y_i, \hat{y}^{(k-1)} + f_k(x_i)) + \sum_{j=1}^{k-1} \Omega(f_i) + \Omega(f_k) \tag{3.4}$$

In Eq (3.4), $f_k(x_i)$ is the prediction result of the i-th sample in the k-th tree. The Taylor series is used to approximately expand $Object_k$, and the calculation process can be represented as

$$Object_k = \sum_{i=1}^{n} [g_i f_k(x_i) + \frac{1}{2} h_i f_k^2(x_i)] + \Omega(f_k) \tag{3.5}$$

In Eq (3.5), g_i is the first derivative of $l(y_i, \hat{y}^{(k-1)})$ with respect to $\hat{y}^{(k-1)}$, and h_i is the second derivative of $l(y_i, \hat{y}^{(k-1)})$ with respect to $\hat{y}^{(k-1)}$. The calculation process of the i-th sample of the k-th tree is $f_k(x_i) = \omega q(x_i)$. ω is a vector that stores the values of the child nodes. $q(x_i)$ indicates that the sample x belongs to the child node i. The set I_j is defined as the samples that fall on the j node, and the penalty term can be formalized as

$$\Omega(f_k) = \gamma T + \frac{1}{2} \lambda \sum_{j=1}^{T} \omega_j^2 \tag{3.6}$$

In Eq (3.6), T is the number of nodes, γ and λ are the control weights. The objective function can be expressed as

$$Object_k = \sum_{i=1}^{T} [G_j \omega_j + \frac{1}{2} (H_j + \lambda) \omega_j^2] + \gamma T \tag{3.7}$$

where $G_j = \sum_{i \in I_i} g_i$, $H_i = \sum_{i \in I_i} h_i$ are known constants, the objective function is essentially a one-dimensional quadratic function. When $\omega_j = -G_j/(H_j + \lambda)$, the objective function takes the maximum value, which can be formalized as

$$Object_k = -\frac{1}{2} \sum_{j=1}^{T} \frac{G_j^2}{H_j + \lambda} + \gamma T \tag{3.8}$$

As shown in Table 3, part of the high correlation features is given, and the importance of the traffic statistical features is ranked according to the node gain value.

Table 3. Feature importance.

No.1	Name	No.1	Name
1	Bwd Packet Length Min	11	Fwd Packets/s
2	Flow IAT Max	12	Fwd IAT Mean
3	Flow IAT Min	13	Packet Length Max
4	Bwd Init Win Bytes	14	Bwd Packets/s
5	Flow Duration	15	Fwd Packet Length Std
6	Fwd Header Length	16	Fwd Init Win Bytes
7	Bwd Packet Length Max	17	Packet Length Mean
8	Idle Max	18	Packet Length Min
9	Bwd Packet Length Mean	19	Fwd Packet Length Max
10	Fwd Packet Length Mean	20	Fwd Packet Length Min

3.3.2. Statistical features and packet length sequence learning

MLP [36] is a typical machine learning model with strong nonlinear representation ability, which is widely used in classification tasks. Each layer in an MLP is a linear mapping representation whose output is equal to subtract offset from the product of input and weight. Multiple layers of such linear mapping representations are combined to solve nonlinear problems. This is equivalent to combining the features with multiple parametric weights. It can be found that the features and feature tuples are most suitable for prediction results, thus, it is used for processing advanced features and the calculation process can be formalized as

$$f_i = g_r(w_i(g_r(w_l x_t + b_i))), i > 1 \tag{3.9}$$

where W_i, b_i, and f_i are respectively the weight matrix, bias vector and output vector of the i-th layer. $g(\cdot)$ is the nonlinear activation function. As shown in Figure 3, the packet sequence size and time-related features X_t are combined into text features, which are jointly input into the MLP to obtain high-level abstract features.

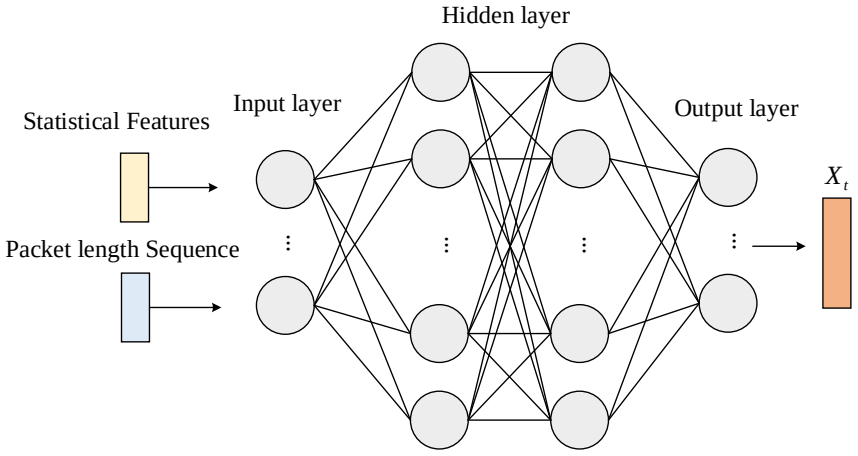


Figure 3. Packet length sequence and statistical feature learning process.

3.3.3. The spatiotemporal feature learning

The spatiotemporal feature learning part consists of CNN, self-attention mechanism and BiGRU [37]. It uses the *CONV* layer to extract the spatial relationship features of the original byte X_c. Each *CONV* layer consists of a convolutional layer, a pooling layer and a Relu activation function with different convolution kernel sizes. The calculation process of the convolutional layer can be denoted as

$$F_c(x_c) = g_r(g_{pool}(f_{convi}(x_c))) \tag{3.10}$$

where f_{convi} is the convolutional layer with kernel size i, g_{pool} is the pooling layer, and g_r is the nonlinear activation function. A self-attention mechanism layer is added after the *CONV* layer to obtain the intrinsic relationship of the features extracted by the *CONV* layer and improve the feature richness. After the self-attention mechanism layer, in order to obtain the time series features in the network flow, the BiGRU network is used to extract the features of X_c. The calculation process of the BiGRU layer can be formalized as

$$X_{ci} = concat[h_{t-1}, x_t] \tag{3.11}$$

where h_{t-1} is the memory state of the previous time $t-1$, $X_c$is the feature vector of the current time t, and X_{ci} is the input vector of the current time t.

Each element of the output vector of the reset gate neuron r_t and the input neuron z_t is between 0 and 1. It is used to control the amount of information. The activation function of the memory gate neuron is tanh, and the elements of the output vector are all between -1 and 1.The reset gate neuron r_t, the memory gate neuron $\tilde{h}_t$, and the input neuron state z_t update computation can be formalized as

$$r_t = \sigma(X_{ci}W_r + b_r) \tag{3.12}$$

$$\tilde{h}_t = \tanh([r_t \odot h_{t-1}, X_{ci}]W_h + b_h) \tag{3.13}$$

$$z_t = \sigma(X_{ci}W_z + b_z) \tag{3.14}$$

where σ represents the nonlinear activation function *sigmoid*, $W_r, b_r, W_z, b_z, W_h, b_h$ are the weight matrix and bias vector parameters of the reset gate, input gate and memory gate neurons, respectively.

After the state value of each gate is achieved, the output value h_t of this unit can be formalized as

$$h_t = (1-z) \odot h_{t-1} + z_t \odot \tilde{h}_t \tag{3.15}$$

As shown in Figure 4, the vector X_c is input into this module. After multiple convolution layers with different convolution kernel sizes to extract spatial features, BiGRU layers to extract time series features, and finally the spatiotemporal feature F_c is obtained.

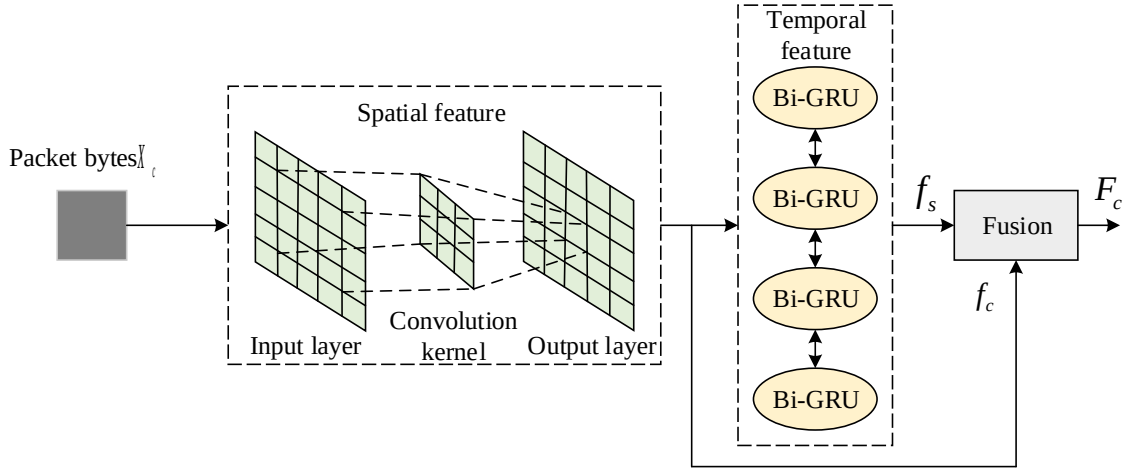


Figure 4. Spatio-temporal feature learning.

3.3.4. The Multi-head self-attention mechanism

The self-attention mechanism calculates the similarity between each feature vector and learns the relationship between each feature. The multi-head self-attention mechanism [32] is an extension method of the self-attention mechanism. The difference between them is that the latter performs one linear transformation on Q, K and V, while the former performs multiple linear transformations on Q, K and V. The attention value is calculated separately for each linear transformation, and the results are concatenated. This can effectively improve the fitting ability of the model.

As shown in Figure 5, the $attention_i$ value between features in parallel are computed, and the final attention matrix $F_m(Q, K, V)$ can be achieved. In order to achieve the purpose of detection and classification. $F_m(Q, K, V)$ can be denoted as

$$F_m(Q, K, V) = \{Attention_1, Attention_2, \cdots\cdots, Attention_3\} \tag{3.16}$$

where Q, K and V represent the query matrix, key-value matrix, and value matrix, respectively. $Attention_i$ is the attention value calculated by the i-th linear transformation of Q, K and V. The calculation process of $Attention_i$ can be expressed as

$$Attention_i = Attention\{W_{qi}X_c, W_{ki}X_c, W_{vi}X_c\} \tag{3.17}$$

where W_{qi}, W_{ki}, W_{vi} are the weight matrices after the i-th linear transformation. In our model, the number of heads H is 7 and the output of the multi-head attention mechanism can be denoted as F_m. It will be used in the subsequent feature fusion module.

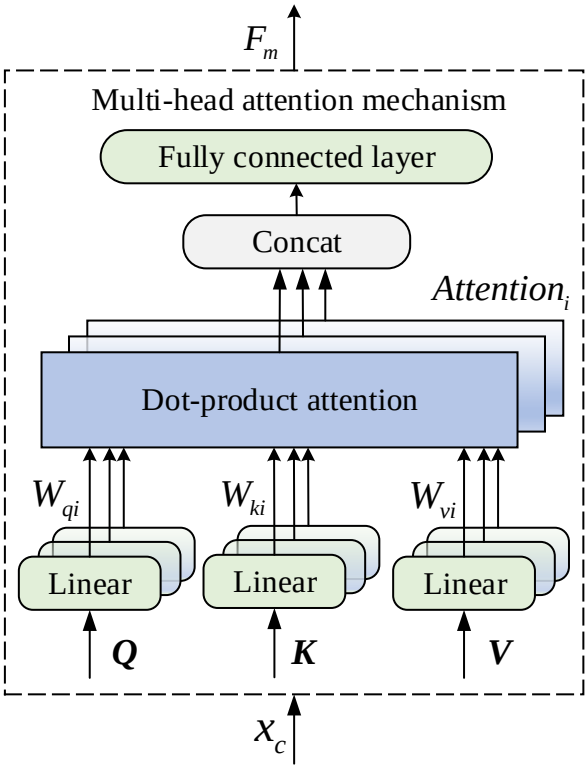


Figure 5. Multi-head attention mechanism.

3.3.5. The logistic chaos map

As it is known chaos arises from nonlinear dynamical systems. A dynamical system describes a time-varying process that has a very sensitive dependence on the initial value. The process is deterministic, quasi-random, aperiodic, and convergent. The mathematical expression of a one-dimensional logistic chaotic map is a difference equation that was first used to describe changes in quantity. Later, as its characteristics met the requirements of a serial cipher, it became widely used in the field of secure communication. The calculation process can be formalized as

$$x_{p+1} = x_p\mu(1 - x_p), \mu \in [0, 4], x \in (0, 1) \tag{3.18}$$

As shown in Figure 6, when the mapping parameter μ is set to 4, the value of the feature vector after chaotic mapping is locally amplified [38]. The output values will tend to a certain defined class, depending on the difference in the initial values. The value of the multidimensional feature vector indicates the degree of difference. This makes the mapped feature vectors easier to distinguish when there are small differences in the value of feature vectors with multiple dimensions. We only focus on the features with blurred edges and do not consider deterministic features after chaotic mapping. The partially magnified fuzzy features are magnified in different degrees according to their initial values. The chaotic map feature vector and the original multi-dimensional feature vector are combined as the classification basis.

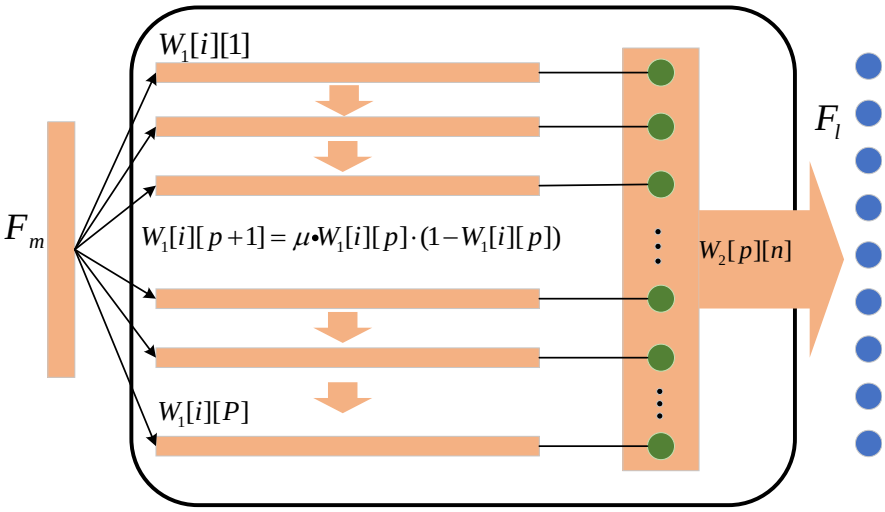


Figure 6. Logistic chaotic map feature map.

3.3.6. Feature fusion and classification

The feature fusion process is divided into two parts. The first part is self-attention feature fusion. In order to more comprehensively learn the relationship between different location features of the network flow, a multi-head self-attention mechanism is used in the overall model. The query matrix Q and key matrix K are used to generate the distribution of attention weights, and the value matrix V is used to obtain the selected information. The computational process of the attention feature fusion can be denoted as

$$F_o = W_o(F_c \cdot Softmax(\frac{\mathrm{F_m^T F_m}}{\sqrt{dim(F_m)}})) + b_o \tag{3.19}$$

where W_o and b_o are the weight and bias values of the feature fusion part, respectively. F_m is the output feature vector of the multi-head self-attention mechanism.

The other part is multimodal feature fusion. We concatenate high-level statistical features extracted by MLP and features fused by self-attention [39]. The fused features serve as the final basis for classification. The fused features are fed into the $Softmax$ layer. The difference between the true class label and the predicted result is taken as the loss value during backpropagation. For the class imbalance problem in darknet traffic, we adopt a focal loss [40] function to deal with it. The calculation process of the classification layer can be formalized as

$$\hat{y} = softmax\mathrm{W}_{\hat{y}} \cdot \mathrm{Concat}(\mathrm{F_t}, \mathrm{F_o}) + \mathrm{b}_{\hat{y}} \tag{3.20}$$

3.4. Online classification and update module

The module continuously optimizes the parameters through the true label Y and the predicted label Yp. Online update module is divided into two cases. We use the parameters fitted by the feature learning module to predict the old class samples. After a new category sample appears, the small-scale inheritance training should be performed on the original model. When learning new types of data

incrementally, most machine learning technologies will encounter catastrophic forgetting problems, which will lead to a sharp decline in the performance of old classes. Therefore, a weighted loss function L is used to retain the training memory, and the calculation process can be formalized as

$$L = \lambda \cdot L_d + (1 - \lambda) \cdot L_c \tag{3.21}$$

L_d and L_c represent two types of loss functions, which measure the degree of difference from the original model and the true label, respectively. The calculation process of L_d can be denoted as

$$L_d = \sum_{x \in \hat{X}^n \cup X^m} \sum_{k=1}^{n} \frac{e^{\hat{o}_k(x)/H}}{\sum_{j=1}^{n} e^{\hat{o}_j(x)/H}} \log(\frac{e^{o_k(x)/H}}{\sum_{j=1}^{n} e^{o_j(x)/H}}) \tag{3.22}$$

where H represents the ability to learn parameters from the old class model. We set the appropriate H value to preserve the model parameters learned for the old class as much as possible. $\hat{X}^n \cup X^m$ represents a collection of old and new class samples. $\hat{o_k}(x)$ is the prediction output vector. $o_k(x)$ is the output vector of the original model.

L_c measures the length of the difference between the predicted vector and the true label and it is used in all categories. Its calculation process can be formalized as

$$L_c = \sum_{(x,y) \in \hat{X}^n \cup X^m} \sum_{k=1}^{m+n} p_k(x) \cdot log(\hat{p_k}(x)) + (1 - p_k(x)) \cdot \log(1 - \hat{p_k}(x)) \tag{3.23}$$

where $p_k(x)$ is the true label of the sample, and $\hat{p_k}(x)$ is the predicted output result of the k-th class sample through the $Softmax$ layer.

The model trained by combining the inheritance of the loss function will be more inclined to classify the samples into new categories. The bias correction layer can make the data more evenly distributed in the network, thereby reducing the dependence on the traffic characteristics of the new category and improving the generalization ability of the model. The calculation process of the bias correction layer is shown in Eq (3.24):

$$p_k = w \cdot o_k(x) + v \tag{3.24}$$

where w and v are the training parameters of the positive deviation layer, and o_k is the output feature vector of the k-th class. The output feature vector of the old class is retained, and the feature vector of the new class is corrected by the bias correction layer to obtain the final classification result.

The new model is initialized with the trained parameters of the old model, and the new class dataset and a small part of the old dataset are used for inheritance training [41]. The last pre-trained layer is truncated and replaced with the classification layer with the new class output. In order to ensure the weight parameters learned by the old model on the previous categories are not easily forgotten, a learning rate that is ten times smaller than the original is used to update the parameters during the inheritance training process.

4. Experimental results and analysis

The public CICDarknet2020 [42] and Darknet2020 [6] dataset is used in this paper to evaluate the model performance. The CICDarknet2020 dataset consists of VPN and Tor network traffic, where

VPN traffic comes from ISCXVPN2016 [43] dataset and Tor traffic comes from dataset ISCXTor2016 [44] dateset. The network flow samples include the normal and darknet network traffic, in which the number of normal network flows is 10,248 and the number of darknet network flows is 21,041. The user behavior traffic in CICDarknet2020 can be divided into 8 categories in total (called dataset 1 in this paper). As shown in Table 4, the dataset includes Audio, Browsing, Chat, Email, P2P, File Transfer, Video and VoIP.

Table 4. The detailed description of dataset 1.

	Dataset 1	
Type	Detailed source	Amount
Browsing	Firefox and Chrome	263
Chat	ICQ, AIM, Skype, Facebook and Hangouts	4541
Email	SMTPs, POP3s and IMAPs	582
Video	Vimeo and YouTube	1346
File Transfer	Skype, FTP over SSH (SFTP) and (FTP) over SSL (FTPS) using FileZilla and an external service	2610
P2P	uTorrent and Transmission (BitTorrent)	220
Audio	Vimeo and YouTube	13,284
VoIP	Facebook, Skype and Hangouts voice calls	1465

4.1. Experimental description

As shown in Table 5, the experimental environment used in these experiments: Windows10, system processor: Intel(R) Core (TM) i5-9300HF CPU @ 2.40GHz quad-core, RAM: 16GB, system type: 64-bit operating system x64-based processor, graphics card: Nvidia GeForce GTX1650. The PyTorch deep learning library, as well as Wireshark, Anaconda and PyCharm is used in the experiments.

Table 5. Experimental platform parameters.

Experimental environment	Windows10
System processor	Intel(R) Core (TM)i5-9300HFCPU 2.40GHz quad-core
RAM	16GB
System type	64-bit operating system x64-based processor
Graphics card	Nvidia GeForce GTX1650
Others	Pytorch Wireshark Anaconda Pycharm etc.

Table 6. Training setup.

	Hyperparameters	Parameter
	Learning rate	0.001
	Batch size	32
	Dropout	0.01
	Loss function	Cross-entropy and KL divergence
	optimizer	Adam

4.2. Evaluation indicators

In this paper, the Accuracy, Precision, Recall, F1-Score and Confusion Matrix are used to evaluate the classification models [45]. The accuracy rate describes the overall performance of the classifier, and the precision rate evaluates the classification effect of each category in the classification problem. The F1-Score is used to evaluate the performance of the classifier. The confusion matrix can be used to observe the classification of each category in detail. TP means that a positive sample is correctly identified as a positive sample. TN means negative samples are correctly identified as negative samples. FP means that negative samples are misidentified as positive samples. FN means that positive samples are misidentified as negative samples.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{4.1}$$

$$Precision = \frac{TP}{TP + FP} \tag{4.2}$$

$$Recall = \frac{TP}{TP + FN} \tag{4.3}$$

$$F1 - score = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{4.4}$$

4.3. Parameter optimization of the model

In this section, we will explore the impact of three parameters (N, M, L) on the classification performance of the model. (N: number of flow statistical features, M: packet sequence length, L: standard bytes)

4.3.1. Number of flow statistical features

We used feature engineering to rank the importance of network flow temporal statistical features and input features sequentially based on feature importance. The goal was to use a smaller number of features to achieve the best accuracy. As shown in Figure 7, the model achieved high accuracy and tended to stabilize with the top 15 important features. Therefore, we chose N = 15 for the experiment.

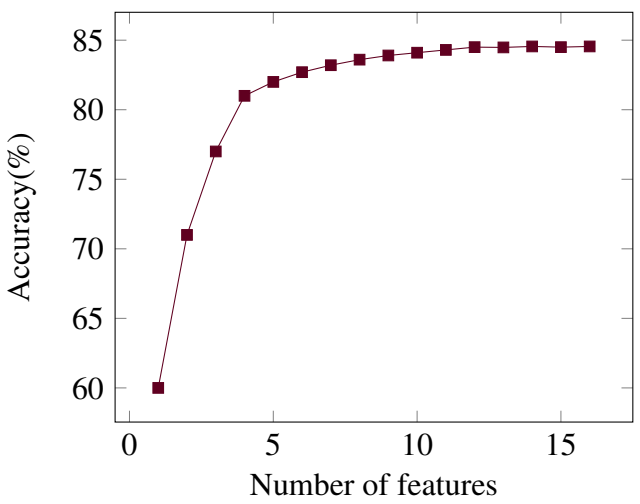
4.3.2. Packet sequence length

In order to make full use of the sequence characteristics of packets in the stream, the length of the packet sequence is gradually increased. As shown in Figure 7, the model is more accurate and tends to be stable when the packet size sequence length M reaches 80. Therefore, we chose M = 80 for the follow-up experiments.

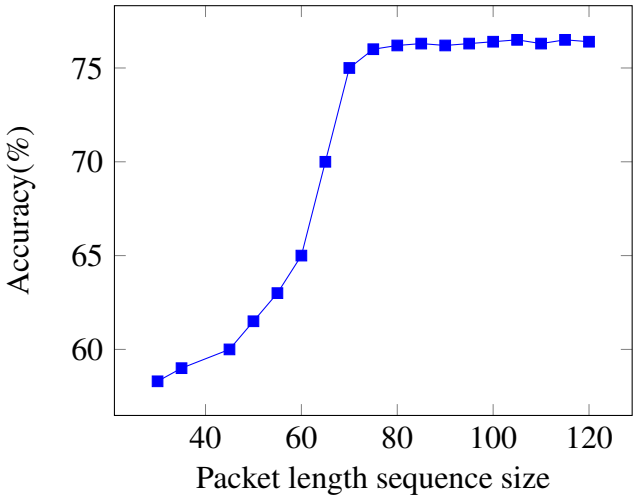
4.3.3. Standard bytes

To explore the impact of standard bytes on the classification results, we intercepted different flow bytes for experiments. As shown in Figure 7, the classification accuracy rates under different standard

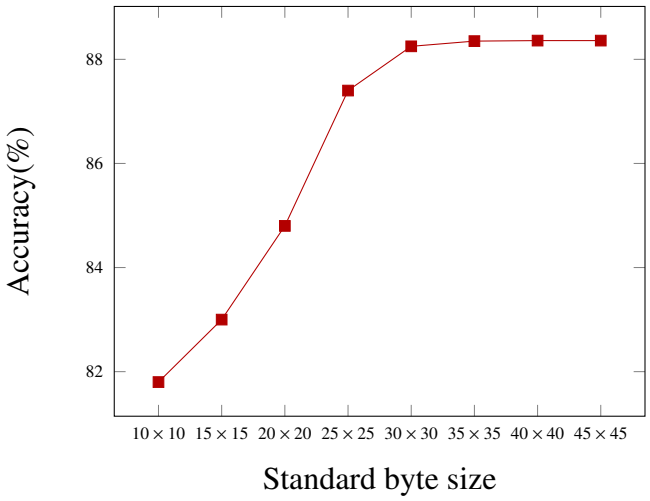
byte lengths were given. As the number of network stream bytes increased from 100 to 700, the accuracy increased significantly. It is noted that the addition of standard bytes would provide more characteristic information. The curve change showed that the provided characteristic information tended to saturate when the standard byte reached a certain length. Therefore, we chose L = 784 for subsequent experiments. It can provide enough information and improve the efficiency of data processing.



(a) Number of statistical features N



(b) Packet sequence length M



(c) Standard byte L

Figure 7. The influence of model parameter values on classification performance.

4.4. Experimental results and analysis

1) Ablation experiments

The proposed method contains multiple classification performance gain parts. In order to evaluate the contribution of each part to the final classification performance of the model, an ablation experiment is performed.

Table 7. The contribution of different modules to model classification performance.

Classification model	Accuracy	Macro-Precision	Macro-Recall	Macro-F1-score
MLP	86.46%	87.32%	85.70%	86.51%
CNN-BiGRU	88.27%	89.22%	86.78%	87.98%
CNN-BiGRU+ Self-attention fusion module	90.38%	91.56%	88.42%	89.96%
CNN-BiGRU+ Self-attention fusion module +MLP	**91.41%**	**92.67%**	**89.54%**	**91.08%**

The accuracy and F1-score are 86.46 and 86.51%, respectively, when MLP is used to deal with statistical features and in-flow packet length sequence features. Due to the limited amount of information carried by packet length sequence and statistical features, the lack of representation of features leads to the method using only these two features is not accurate enough in classifying user behavior traffic.

When the original bytes of the packet are input into the CNN-BiGRU cascade network to extract the spatio-temporal fusion features, the model accuracy and F1-score are 88.27 and 87.98% respectively, which are 1.81 and 1.47% higher than those using statistical features and in-flow packet length sequence features. This shows that in the process of user behavior traffic identification, the information carried by the original bytes of the packet is more abundant, and the traffic category is more representative after extracting the spatial and temporal features.

After introducing the self-attention feature fusion module, the accuracy and F1-score of the model are increased by 2.11 and 1.98%, reaching 90.38 and 89.96%, respectively. This shows that the module can analyze the relationship between features from multiple perspectives, extract feature correlation and assign more weights to important features. Features with high correlation can provide more valuable information, so as to improve the classification ability of the model for traffic.

After adding statistical features and packet length sequence features, the classification accuracy and F1-score of the model reach 91.41 and 91.08%, respectively. It shows that multi-modal data input can provide feature information of different nature, enhance the characterization ability of features for traffic and contribute to the fine-grained classification of anonymous network traffic.

2) Comparison with the state-of-the-art methods

To reflect the advantages of the proposed method in terms of classification performance, this paper compares it with state-of-the art methods. The comparison results of each category of applications are shown in Table 8. The experimental results are obtained by conducting traffic classification experiments on the CICDarket2020 dataset. Figure 8 shows the accuracy of different classification models on different applications of the darknet traffic dataset. The proposed model achieves the best performance in most categories and is slightly lower than the DarknetSec [19] method in the Browsing and Video-Stream categories. Overall, our method achieves the best classification performance.

It can be observed that the proposed method achieves high accuracy, exceeding 95%, in classifying categories such as Audio, Chat and File-Transfer. However, the classification performance is relatively lower, reaching only above 80% for categories like Browsing and Email. It is attributed to the larger number of samples available for categories like Audio compared to the smaller sample size for categories like Browsing. As a result, the model struggles to obtain sufficient learning for the latter. Therefore, the future work section of this paper also suggests addressing and resolving the issue of small-sample class classification. This will help improve the overall performance and generalization capability of the model and deserves further attention and exploration in future research.

Table 8. The accuracy of different methods on darknet user behavior traffic classification.

Method	Audio	Browsing	Chat	File-Transfer	Email	P2P	Video-Stream	VoIP	Average Accuracy
Our Model	**98.46%**	81.27%	**98.63%**	**95.63%**	**81.07%**	**93.52%**	92.26%	**91.56%**	**96.02%**
RF [43]	95.74%	79.43%	91.58%	91.16%	76.53%	91.19%	88.27%	89.12%	92.98%
DIDarket [10]	97.25%	80.16%	93.21%	91.76%	79.48%	92.16%	90.64%	89.24%	94.40%
FlowPic [19]	98.21%	78.15%	94.23%	90.54%	79.35%	90.64%	89.22%	89.68%	94.89%
Deep Packet	95.45%	79.74%	91.58%	92.69%	77.68%	91.69%	89.51%	90.35%	93.16%
DarknetSec	98.36%	**82.74%**	95.58%	93.69%	80.68%	93.29%	**92.56%**	91.35%	95.96%
CNN+LSTM [46]	96.31%	80.26%	95.17%	93.38%	79.21%	93.28%	91.52%	90.72%	94.57%

In addition, the proposed method has an online update function. When the traffic category increases, the combination loss function is used for inheritance small-scale training and the deviation correction layer is used to correct the parameters. The browsing and email categories were used as new traffic categories, and compared with the retraining method, the fine-tuning method and the combined loss function method respectively. The retraining method retrains the model according to all the old and new data samples to adapt the model parameters to all the old and new traffic categories. The fine-tuning method is to freeze the upper-layer parameters of the model and update the parameters to adapt to the new category using only the fully connected layer. The combined loss function method is weighted by two loss functions. One loss function measures the difference between the predicted label and the true label, and the other loss function measures the difference between the predicted label of the new model and the old model.

In order to better compare the classification performance with different methods, all indicators are normalized. Figure 9 shows that although the re-training method performs best in the old and new categories and the overall accuracy, it consumes a lot of time and memory space, which is not conducive to the timely update of model parameters. Fine-tuning methods tend to identify predicted samples as new categories, resulting in severe parameter forgetting on old categories and poor overall classification performance. The combined loss function method has a good classification performance on the new class samples, but the accuracy is low on the old class samples. The model in this chapter uses the inheritance training of the combined loss function, and corrects the parameters through the deviation correction layer. This method has less time consumption, occupies less memory space, and is easy to update the model parameters in time. It has achieved good classification results in both new and old categories. Although the accuracy of the relative proportion training method is slightly reduced on the old categories, the parameter update efficiency is significantly improved, and the overall classification efficiency is greatly improved. Therefore, this method can add the new category traffic to the classification model faster while ensuring the accuracy.

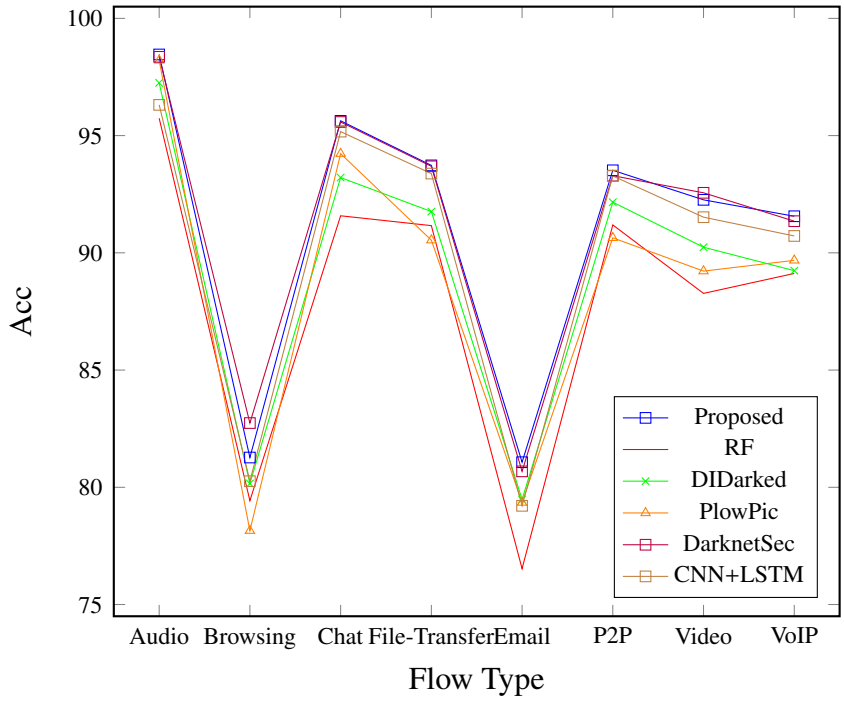


Figure 8. Classification accuracy of different types of applications.

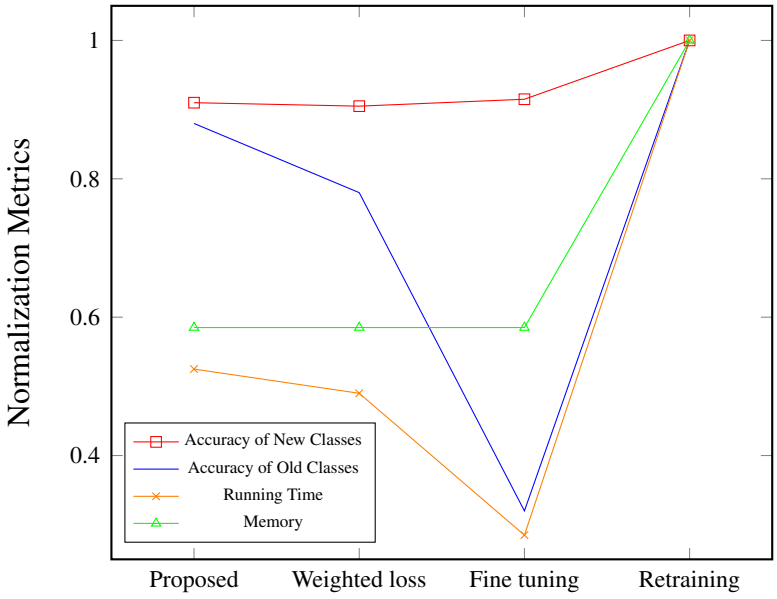


Figure 9. Comparison of classification efficiency of different methods.

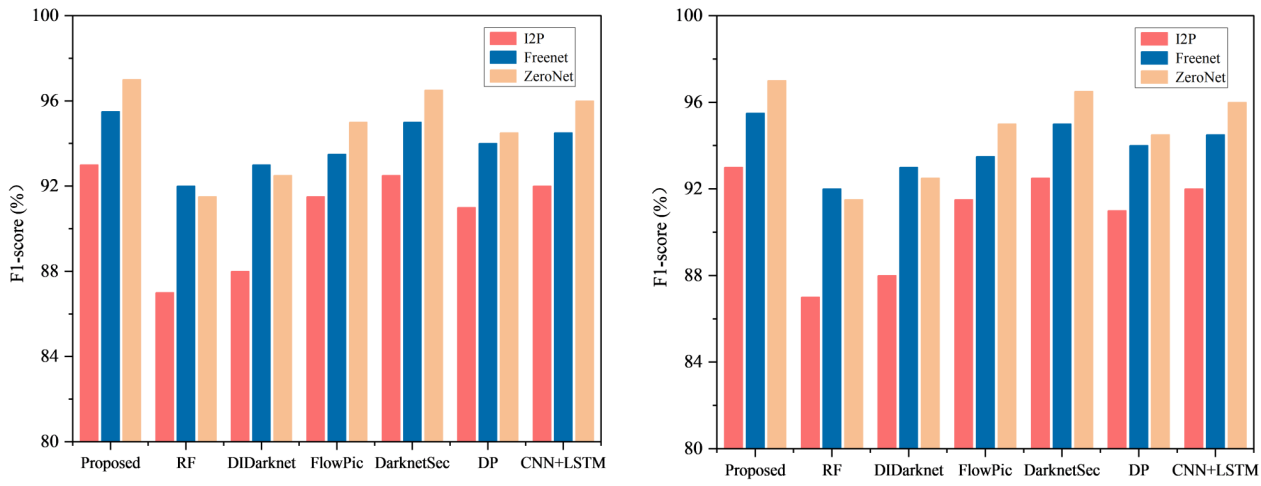
3) Classification performance and comparison experiments

The model proposed in this paper can effectively complete the classification task on the VPN and Tor traffic datasets. However, there are many types of darknet traffic, in order to explore the classification performance of the proposed model in the face of different types of darknet user behavior traffic,

we conduct comparative experiments on the public dataset Darknet2020 to classify the user behavior traffic of three anonymous network tools (I2P, Freenet [47] and ZeroNet [48]). As shown in Table 9, the Darknet2020 dataset consists of four common darknet traffic, Tor, I2P, Freenet, and ZeroNet, and contains 25 darknet user behaviors (called dataset 2 in this paper). The comparison experimental results are shown in Figure 10. It can be observed that among all types of anonymous network traffic, the proposed method achieves higher accuracy and F1-score compared to other anonymous traffic classification methods. It is easily achieved that the proposed method has the best performance.

Table 9. The detailed description of dataset 2.

	Dataset 2			
-	Tor	I2P	ZeroNet	Freenet
Browsing	1281	1921	7972	4990
Chat	841	442	1531	1123
Email	553	1084	352	2980
File Transfer	1077	1791	2157	4897
P2P	1018	2910	1394	-
Audio	1567	-	820	-
Vedio	1703	-	1251	2397
VoIP	592	-	-	-
Total	8632	8148	15477	16387



(a) Acc

(b) F1-Score

Figure 10. The comparison results of different methods on the dataset 2.

5. Conclusions and future works

This paper proposed a general multimodal multitask DL architecture ODTC for multipurpose classification. It is aimed to provide an effective design basis for sophisticated darknet traffic management.

The MLP and CNN-BiGRU are adopted to process the features of the two modalities respectively. Additionally, the multi-modal information, including flow statistical features and payloads, is used to reconstruct network traffic. The flow forms of different modes have different characteristics, which greatly enriches the diversity of characteristics. Afterward, we perform chaotic mapping on the features amplify the ambiguous features, which can enhance the representation ability of the features. The multi-head attention mechanism is able to extract the intrinsic relationship between features and improves classification performance at the same time. In addition, an incremental learning model is adopted, which consists of two advantages. On the one hand, inheritance learning uses a weighted loss function to improve training efficiency. On the other hand, a Deviation correction layer is introduced to address the imbalance between old and new class samples.

By comparative experiments, it is verified that the proposed method outperforms existing methods. In addition, the effectiveness of the proposed model for other various other types of darknet traffic datasets is also verified. In the future, we will investigate the impact of class incremental learning on model performance.

Category incremental learning can help us adapt to ever-changing environments and data, enabling the model to quickly learn and adapt to new categories without retraining the entire model. With the development of technology and the accumulation of data, we need to be able to handle large-scale and constantly changing datasets. Understanding category incremental learning can help us improve existing incremental learning algorithms and develop more efficient and robust models.

Use of AI tools declaration

The authors declare they have not used Artificial Intelligence (AI) tools in the creation of this article.

Acknowledgements

This work was supported by the National Natural Science Foundation of China (Grants No. 61931004, 62072250), the National Key Research and Development Program of China (Grants No. 2021QY0700).

Conflict of interest

The authors declare there is no conflict of interest.

References

1. A. Montieri, D. Ciuonzo, G. Bovenzi, V. Persico, A. Pescapé, A dive into the dark web: hierarchical traffic classification of anonymity tools, *IEEE Trans. Network Sci. Eng.*, **7** (2019), 1043–1054. https://doi.org/10.1109/TNSE.2019.2901994
2. G. Aceto, A. Pescapé, Internet censorship detection: a survey, *Comput. Networks*, **83** (2015), 381–421. https://doi.org/10.1016/j.comnet.2015.03.008
3. Y. D. Goli, R. Ambika, Network traffic classification techniques-a review, in *2018 International Conference on Computational Techniques, Electronics and Mechanical Systems (CTEMS)*, (2018), 219–222. https://doi.org/10.1109/CTEMS.2018.8769309

4. T. Bujlow, V. Carela-Español, P. Barlet-Ros Independent comparison of popular DPI tools for traffic classification, *Comput. Networks*, **76** (2015), 75–89. https://doi.org/10.1016/j.comnet.2014.11.001
5. S. Rezaei, X. Liu, Deep learning for encrypted traffic classification: an overview, *IEEE Commun. Mag.*, **57** (2019), 76–81. https://doi.org/10.1109/MCOM.2019.1800819
6. Y. Hu, F. Zou, L. Li, P. Yi, Traffic classification of user behaviors in Tor, I2P, ZeroNet, Freenet, in *2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, (2020), 418–424. https://doi.org/10.1109/TrustCom50675.2020.00064
7. Z. Fan, R. Liu, Investigation of machine learning based network traffic classification, in *2017 International Symposium on Wireless Communication Systems (ISWCS)*, (2017), 1–6. https://doi.org/10.1109/ISWCS.2017.8108090
8. N. Bayat, W. Jackson, D. Liu, Deep learning for network traffic classification, preprint, arXiv: 2106.12693.
9. X. Hu, C. Gu, F. Wei, CLD-Net: a network combining CNN and LSTM for internet encrypted traffic classification, *Secur. Commun. Netw.*, **2021** (2021), 1–15. https://doi.org/10.1155/2021/5518460
10. A. H. Lashkari, G. Kaur, A. Rahali, Didarknet: a contemporary approach to detect and characterize the darknet traffic using deep image learning, in *2020 the 10th International Conference on Communication and Network Security*, (2020), 1–13. https://doi.org/10.1145/3442520.3442521
11. K. Lin, X. Xu, H. Gao, TSCRNN: a novel classification scheme of encrypted traffic based on flow spatiotemporal features for efficient management of IIoT, *Comput. Networks*, **190** (2021), 107974. https://doi.org/10.1016/j.comnet.2021.107974
12. K. Kim, J. H. Lee, H. K. Lim, S. W. Oh, Y. H. Han, Deep RNN-based network traffic classification scheme in edge computing system, *Comput. Sci. Inf. Syst.*, **19** (2022), 165–184. https://doi.org/10.2298/CSIS200424038K
13. J. Lan, X. Liu, B. Li, Y. Li, T. Geng, DarknetSec: a novel self-attentive deep learning method for darknet traffic classification and application identification, *Comput. Secur.*, **116** (2022), 102663. https://doi.org/10.1016/j.cose.2022.102663
14. Z. Wu, Y. Dong, X. Qiu, J. Jin, Online multimedia traffic classification from the QoS perspective using deep learning, *Comput. Networks*, **204** (2022), 108716. https://doi.org/10.1016/j.comnet.2021.108716
15. K. Shahbar, A. N. Zincir-Heywood, Effects of shared bandwidth on anonymity of the I2P network users, in *2017 IEEE Security and Privacy Workshops (SPW)*, (2017), 235–240. https://doi.org/10.1109/SPW.2017.19
16. Z. Rao, W. Niu, X. S. Zhang, H. Li, Tor anonymous traffic identification based on gravitational clustering, *Peer-to-Peer Networking Appl.*, **11** (2018), 592–601. https://doi.org/10.1007/s12083-017-0566-4
17. L. A. Iliadis, T. Kaifas, Darknet traffic classification using machine learning techniques, in *2021 10th International Conference on Modern Circuits and Systems Technologies (MOCAST)*, (2021), 1–4. https://doi.org/10.1109/MOCAST52088.2021.9493386

18. M. B. Sarwar, M. K. Hanif, R. Talib, M. Younas, M. U. Sarwar, DarkDetect: darknet traffic detection and categorization using modified convolution-long short-term memory, *IEEE Access*, **9** (2021), 113705–113713. https://doi.org/10.1109/ACCESS.2021.3105000

19. T. Shapira, Y. Shavitt, FlowPic: a generic representation for encrypted traffic classification and applications identification, *IEEE Trans. Netw. Serv. Manage.*, **18** (2021), 1218–1232. https://doi.org/10.1109/TNSM.2021.3071441

20. H. Yao, C. Liu, P. Zhang, S. Wu, C. Jiang, S. Yu, Identification of encrypted traffic through attention mechanism based long short-term memory, *IEEE Trans. Big Data*, **8** (2022), 241–252. https://doi.org/10.1109/TBDATA.2019.2940675

21. J. Xie, S. Li, X. Yun, Y. Zhang, P. Chang, Hstf-model: an http-based trojan detection model via the hierarchical spatio-temporal features of traffics, *Comput. Secur.*, **96** (2020), 101923. https://doi.org/10.1016/j.cose.2020.101923

22. M. M. Hassan, A. Gumaei, A. Alsanad, M. Alrubaian, G. Fortino, A hybrid deep learning model for efficient intrusion detection in big data environment, *Inf. Sci.*, **513** (2020), 386–396. https://doi.org/10.1016/j.ins.2019.10.069

23. P. R. Kanna, P. Santhi, Unified deep learning approach for efficient intrusion detection system using integrated spatial–temporal features, *Knowledge-Based Syst.*, **226** (2021), 107132. https://doi.org/10.1016/j.knosys.2021.107132

24. L. Liu, J. Zhen, G. Li, G. Zhan, Z. He, B. Du, et al., Dynamic spatial-temporal representation learning for traffic flow prediction, *IEEE Trans. Intell. Transp. Syst.*, **22** (2021), 7169–7183. https://doi.org/10.1109/TITS.2020.3002718

25. G. D'Angelo, F. Palmieri, Network traffic classification using deep convolutional recurrent autoencoder neural networks for spatial–temporal features extraction, *J. Network Comput. Appl.*, **173** (2021), 102890. https://doi.org/10.1016/j.jnca.2020.102890

26. M. Lopez-Martin, B. Carro, A. Sanchez-Esguevillas, J. Lloret, Network traffic classifier with convolutional and recurrent neural networks for internet of things, *IEEE Access*, **5** (2017), 18042–18050. https://doi.org/10.1109/ACCESS.2017.2747560

27. F. Xiao, GEJS: a generalized evidential divergence measure for multisource information fusion, *IEEE Trans. Syst. Man Cybern.: Syst.*, **53** (2023), 2246–2258. https://doi.org/10.1109/TSMC.2022.3211498

28. L. Vu, C. T. Bui, Q. U. Nguyen, A deep learning based method for handling imbalanced problem in network traffic classification, in *Proceedings of the 8th International Symposium on Information and Communication Technology*, (2017), 333–339. https://doi.org/10.1145/3155133.3155175

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

30. C. Liu, L. He, G. Xiong, Z. Cao, Z. Li, FS-Net: a flow sequence network for encrypted traffic classification, in *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications*, (2019), 1171–1179. https://doi.org/10.1109/INFOCOM.2019.8737507

31. G. Aceto, D. Ciuonzo, A. Montieri, A. Pescapé, DISTILLER: encrypted traffic classification via multimodal multitask deep learning, *J. Network Comput. Appl.*, **183–184** (2021), 102985. https://doi.org/10.1016/j.jnca.2021.102985
32. G. Xie, Q. Li, Y. Jiang, Self-attentive deep learning method for online traffic classification and its interpretability, *Comput. Networks*, **196** (2021), 108267. https://doi.org/10.1016/j.comnet.2021.108267
33. G. Bovenzi, L. Yang, A. Finamore, G. Aceto, D. Ciuonzo, A. Pescapè, et al., A first look at class incremental learning in deep learning mobile traffic classification, preprint, arXiv: 2107.04464.
34. F. Hu, S. Zhang, X. Lin, L. Wu, N. Liao, Y. Song, Network traffic classification model based on attention mechanism and spatiotemporal features, *EURASIP J. Inf. Secur.*, **2023** (2023), 1–25. https://doi.org/10.1186/s13635-023-00141-4
35. Z. Wang, B. Ma, Y. Zeng, X. Lin, K. Shi, Z. Wang, Differential preserving in XGBoost model for encrypted traffic classification, in *2022 International Conference on Networking and Network Applications (NaNA)*, (2022), 220–225. https://doi.org/10.1109/NaNA56854.2022.00044
36. Q. Lyu, X. Lu, Effective media traffic classification using deep learning, in *Proceedings of the 2019 3rd International Conference on Compute and Data Analysis*, (2019), 139–146. https://doi.org/10.1145/3314545.3316278
37. C. Y. Lin, B. H. Chen, W. Y. Lan, An efficient approach for encrypted traffic classification using CNN and bidirectional GRU, in *2022 2nd International Conference on Consumer Electronics and Computer Engineering (ICCECE)*, (2022), 368–373. https://doi.org/10.1109/ICCECE54139.2022.9712708
38. A. Velichko, Neural network for low-memory IoT devices and MNIST image recognition using kernels based on logistic map, *Electronics*, **9** (2020), 1432. https://doi.org/10.3390/electronics9091432
39. V. Tong, H. A. Tran, S. Souihi, A. Mellouk, A novel QUIC traffic classifier based on convolutional neural networks, in *2018 IEEE Global Communications Conference (GLOBECOM)*, (2018), 1–6. https://doi.org/10.1109/GLOCOM.2018.8647128
40. Y. Guo, Z. Li, Z. Li, G. Xiong, M. Jiang, G. Gou, FLAGB: focal loss based adaptive gradient boosting for imbalanced traffic classification, in *2020 International Joint Conference on Neural Networks (IJCNN)*, (2020), 1–8. https://doi.org/10.1109/IJCNN48605.2020.9207336
41. Z. Bu, B. Zhou, P. Cheng, K. Zhang, Z. Ling, Encrypted network traffic classification using deep and parallel network-in-network models, *IEEE Access*, **8** (2020), 132950–132959. https://doi.org/10.1109/ACCESS.2020.3010637
42. L. A. Iliadis, T. Kaifas, Darknet traffic classification using machine learning techniques, in *2021 10th International Conference on Modern Circuits and Systems Technologies (MOCAST)*, (2021), 1–4. https://doi.org/10.1109/MOCAST52088.2021.9493386
43. G. Draper-Gil, A. H. Lashkari, M. S. I. Mamun, A. A. Ghorbani, Characterization of encrypted and vpn traffic using time-related, in *Proceedings of the 2nd International Conference on Information Systems Security and Privacy (ICISSP)*, (2016), 407–414. https://doi.org/10.5220/0005740704070414

44. A. H. Lashkari, G. Draper-Gil, M. S. I. Mamun, A. A. Ghorbani, Characterization of tor traffic using time based features, in *Proceedings of the 3rd International Conference on Information Systems Security and Privacy (ICISSP 2017)*, (2017), 253–262. https://doi.org/10.5220/0006105602530262

45. Q. A. Al-Haija, M. Krichen, W. A. Elhaija, Machine-learning-based darknet traffic detection system for IoT applications, *Electronics*, **11** (2022), 556. https://doi.org/10.3390/electronics11040556

46. N. Rust-Nguyen, M. Stamp, Darknet traffic classification and adversarial attacks, preprint, arXiv: 2206.06371.

47. R. Wang, Y. Zhao, A survey on anonymous communication systems traffic identification and classification, in *2021 3rd International Conference on Advanced Information Science and System (AISS 2021)*, **36** (2021), 1–5. https://doi.org/10.1145/3503047.3503087

48. N. Rust-Nguyen, M. Stamp, Darknet traffic classification and adversarial attacks, preprint, arXiv:2206.06371.

AIMS Press