ENHANCING THE EFFECTIVENESS OF ENCRYPTED TRAFFIC CLASSIFICATION THROUGH DATA PRESERVATION AND INPUT ALIGNMENT WITH DEEP NEURAL NETWORKS

Nguyen Hong Son, Nguyen Trung Hieu

Faculty of Information Technology, Posts and Telecommunications Institute of Technology, Ho Chi Minh City, Vietnam

ABSTRACT

Network traffic classification plays a crucial role in network management and security. Most of the network traffic today is in encrypted form, making traffic identification more difficult. In this context, machine learning and deep learning have emerged as the foundational technologies to solve the problem. To date, numerous encrypted network traffic classifiers based on machine learning and deep learning have been proposed and extensively evaluated in experiments. However, the instability in the performance of these models when deployed on real networks has posed a challenge that has not been satisfactorily addressed so far. In this study, we propose a feasible method to build a more sustainable encrypted network traffic classifier. The classifieris builtbased on innovative input data generation techniques that preserve important latent features and facilitate the CNN deep learning network to maximise its inference ability. The proposed method aims to improve the model's performance and adapt well to the variability and resource constraints of real-world networks. Experimental results show that our model achieves classification performance comparable to state-of-the-art methods. While handling full information of the data samples to avoid missing potential variability factors, the model still maintains simplicity to minimise the limited computational cost of real networks.

KEYWORDS

Encrypted traffic classification, machine learning, deep learning, CNN, data transformation, VPN-nonVPN dataset

1. INTRODUCTION

Network traffic classification has long been a fundamental problem for network service providers. Based on network traffic classification, ISPs develop capabilities to identify network applications, identify tunnelling patterns, and detect malicious traffic. Traditional solutions rely on a deep analysis of packets, particularly the packet overhead, to identify traffic types based on assigned ports and other control information. However, most of today's traffic is in different encrypted forms through tunnel channels such as VPN, Tor. Along with the policy of dynamic port switching, this has prevented most of the deep inspection activities from the data packet, resulting in such traditional methods no longer being feasible. While there is no clear service information in the packet data for classification, what can be done is to use the encrypted data for classification, and that is why machine learning and deep learning (ML/DL)have been applied. Up to now, there have been many studies applying ML/DL to the problem of encrypted traffic classification, in which supervised learning methods are commonly used. The basic work is to convert what is seen from the encrypted packets into features, label them and use ML/DL algorithms to build a classifier. The results from the proposed methods have demonstrated the potential of ML/DL solutions for the problem of encrypted traffic classification. However, ML/DL models that perform well in the lab do not always achieve similar results in real network environments. There are many reasons for this problem. The fundamental reason is that classifiers are not able to adapt to many objective real-world factors because these factors are omitted or have not been included in the learning process. A typical case is that feature selection in the process of processing raw data into input data of the classifier has accidentally lost some important real-world factors. Normally, feature selection methods will calculate the importance of each feature based on the Euclidean distance measure and retain the features with high importance. However, the nature of some important factors in network data is not represented by the Euclidean distance measure; ignoring them will lack of information for classification. Another case is that current classifiers are trained with old data sets that have not been updated with the traffic to reflect the new encryption method. Cryptographic protocols such as QUIC or SSL/TLS 1.3 are increasingly complex[1], reducing the effectiveness of existing models when faced with unknown traffic patterns. More flexible models with continuous learning capabilities are needed to adapt to the rapid changes in cryptographic technology.

Another reality is that the network environment is constantly changing [2], with frequent changes and varying degrees of instability. The dynamic effects of the system are unpredictable, such as jitter, turnaround time, congestion control, and error control [3,4], creating new samples/observations that challenge the generalization ability of the model. The challenges posed by the network environment also stem from the high data transmission rate as well as the scale of network traffic. In fact, it is very difficult for classifiers to respond to high traffic rates and real-time. To have a good encrypted network traffic classifier, these dynamic factors must be taken into account when building it.

In addition, the working speed of the classifier depends on the complexity of the software structure and algorithms used to build it. From a software perspective, spatial and temporal complexity will lead to increased demand for computational resources and negatively affect the model's ability to respond in practice. When the demand for computational resources increases while the infrastructure's capacity is limited, the model cannot keep up with the large and high-speed traffic on the core network. Therefore, it is necessary to minimize the complexity of the classifiers and fully consider including network dynamics in the design and construction of the model. This is a challenging task, but not without solutions.

In this study, we carefully analyse the objective practical factors that lead to the limitations of current classification models. On that basis, we find the necessary techniques to respond well to each of those objective, practical factors in building a classifier. To achieve this, we propose building a classifier based on a reasonably simple deep learning model that processes data in a way that accurately reflects the characteristics of network traffic and aligns well with neural networks.

Focusing on specific work, we analyse the strengths and weaknesses of the deep neural network used to build the encrypted network traffic classifier. Thereby, we know clearly what the input data must be like for the deep neural network to work best. At the same time, we also consider the computational complexity of neural network architectures so that the computational resource demand does not exceed the supply capacity of the underlying infrastructure, particularly in edge computing environments. In addition, we also study the changes in the original network data during the conversion process into the final input data of the ML/DL model according to current popular processing methods. Determine the causes of loss of objective reality in the data and the coverage level of the transformed data. From there, we propose a data preprocessing method that includes all necessary network elements and possesses many favourable properties, leveraging the advantages of deep learning models. This advantage helps to fully exploit the classification ability of deep learning models. We prioritise the design options that can withstand the challenges and create an advanced pipeline to build a high-performance encrypted network traffic classifier that adapts well to the real-world network environment. Experimental results with the ISCX VPN-non VPN Dataset [5] show promising results, achieving state-of-the-art and robustness.

Our research has some contributions as follows:

- Propose a data preprocessing method that fully preserves the properties of real network traffic.
- Propose the concept of data harmonisation with deep learning neural networks: data harmonisation involves data preprocessing techniques that are designed to optimise its compatibility with the AI model's learning dynamics.
- Propose a preprocessing method to generate data that is well-matched with the CNN deep learning neural network.
- Build an encrypted network traffic classifier that is not only highly efficient but also well-adaptable to the real network environment.

The rest of the paper is organised as follows: Section 2 provides a realistic view of the appeal of the encrypted traffic classification problem and the development through research to build a high-quality classifier. Section 3 presents the proposed method from the main idea to the data transformation method. The construction and testing process of the proposed method is presented in Section 4. Section 5 provides evaluation and analysis of the results and new findings, and provides an objective comparison with the novel research results. The paper ends with conclusions in Section 6 on the contributions of the research and future directions.

2. Related Works

Encrypted network traffic classification is of great significance in the management, operation and security of data transmission networks. Therefore, it has attracted many researchers very early. Classification methods have evolved significantly in recent years, in the trend of maintaining network visibility without violating privacy through packet decryption. Machine learning has emerged as a potential method for this trend. It is easy to see that solutions to the problem of encrypted network traffic classification have also kept pace with the development in the field of machine learning. Early studies solved the problem of encrypted traffic classification by using classical machine learning models based on flow features and statistics. For example, in [6] models such as Random Forest (RF), Support Vector Machines (SVM), K-Nearest Neighbours (KNN), and Gradient Boosting (GB) were used, using statistical metadata such as packet length, inter-arrival time, and flow duration. Among them, the random forest classifier achieved the highest result of 97.45%, showing the effectiveness of statistical features even in the absence of payload data. Similarly, [7] also suggested that flow-level statistical features are sufficient to classify encrypted traffic classes. Their evaluations on multiple models, such as RF, SVM, KNN, Naive Bayes, and Logistic Regression, also confirmed that RF and SVM are the most effective. More sophisticated, in [8] ÉCLAIR was introduced, a system capable of real-time classification according to the transmission rate using Decision Trees and Random Forests deployed in P4-programmable switches. This study also showed that machine learning classifiers can work directly on the data plane to achieve high throughput and minimise latency. Along with the development of artificial neural networks, there have also been studies applying deep learning to classify encrypted network traffic. By applying deep learning, the studies have avoided manual feature engineering and improved the generalisation of the model. One such work is [9], in which the authors introduced a framework using CNN (Convolutional Neural Network) and SAE

(Stacked Autoencoders) on raw packet data and published quite impressive results with multi-class classification accuracy reaching 98% and traffic level reaching 94%. However, while this result needs to be further verified, their framework's packet-by-packet preprocessing approach has the disadvantage of incurring a significant load on the processing power of real-world network infrastructures. Another solution that avoids packet-by-packet processing is [10], which presents a flow-based deep learning framework for classifying encrypted network traffic. The authors transform the stream data into images to allow CNNs to efficiently extract spatial and temporal features. This solution outperforms traditional and early deep learning solutions, especially in encrypted and masked traffic conditions. The classification accuracy for five traffic types and three cases of VPN, non-VPN, and Tor is 91.3% with merged datasets. Another similar study is [11], which also takes advantage of CNN to classify flows without using any feature engineering. The authors transform the traffic data into image-like matrices and obtain an average accuracy of more than 93.8%. This also demonstrates the superiority of CNN over traditional machine learning models. Another study is [12], which also approaches the traffic in the form of flows and transforms the flow traffic into gray scale images and takes advantage of an autoencoder to reconstruct small traffic samples to improve the classification quality. The authors conduct traffic classification based on GCN and obtain an accuracy of more than 94.3% in multi-class classification on the merged ISCX VPN-nonVPN dataset. Other advanced deep learning applications have also been developed for traffic classification in SDN networks, such as in [13,14,15].The authors in [13] combined CNN and LSTM to capture spatial and sequential features in a Software-Defined Networking (SDN) environment. The hybrid model gives promising results while also supporting real-time analysis. Another hybrid form comes from [16], where the authors propose to combine CNN and BERT(Bidirectional Encoder Representations from Transformers). The purpose of the hybrid architecture is to capture both local and contextual features of the encoded packet sequences. The results from this study also show superiority over other advanced studies, such as Deep Packet [9].

Another frontier of deep learning is also explored by [17] for the problem of encrypted traffic classification. In this work, the authors applied a contrastive learning technique, which allows the model to learn from unlabeled data and achieve high accuracy even with only a small number of labelled data, which is promising for low-supervision environments. Concerned with adaptability to real-world conditions like ours, [18] introduced a lightweight XGBoost model, which still achieves the same level of performance as state-of-the-art solutions but with significantly reduced computational costs. Another more practical work is [19]. In this work, the authors proposed a low-delay framework that performs classification on the data at the beginning of a communication session. It uses an adaptive learning mechanism with neural network encoders to achieve accurate and fast results, suitable for real-time environments. In addition, to improve generalizability and robustness, [4] proposed two domain-specific data augmentation strategies: Average Augmentation and MTU Augmentation. This technique significantly improves performance on small and unbalanced datasets while minimising the loss of accuracy when changing the size of the MTU (Maximum Transmission Unit). With the impressive success of transformers in large language models, some studies have also applied transformers to encrypted traffic classification models, as in [20,21]. There is also research applying machine learning models to classify network traffic with a recently published dataset [22]. In general, the problem of encrypted network traffic classification is not simply a classification task on a dataset with all features known in advance. Many practical factors greatly affect the feasibility of a classification model. First of all, the specific nature of the applications, in a specific domain, generates network traffic and then the requirements for latency or speed and the moderate resource needs that can be met. All of this poses challenges to the current task of classifying encrypted network traffic.

3. THE PROPOSED METHOD

Our method is inspired by real-world challenges observed in encrypted network traffic classification. It focuses on identifying key stages in the model development pipeline and applying appropriate processing techniques to enhance overall effectiveness.

3.1. The Main Idea

Our main idea is that the encrypted network traffic classifier will be more efficient and sustainable if the following three criteria are achieved during the development process:

(1)Preserve the nature of the data
(2)Maximise the degree of data harmonisation with the deep learning network
(3)Minimise the computational scale to suit the actual conditions

All steps in our deep learning-based classification model construction pipeline adhere to the three criteria above, as illustrated in Figure 1. In general, the work begins with the input data preparation steps, followed by the model selection and learning method selection. Data preparation for the input of a deep learning model is nothing more than performing the necessary preprocessing steps to transform raw data into input data for a deep learning neural network. Specifically, after the data cleaning step, the structure and local relationships of the data are enhanced through flow-based mapping: grouping packets into flows. Converting traffic from packets to flows is to avoid the costly processing of each packet, as in the method of [9]. This is to ensure criterion (3).

During the survey, we discovered that when there is a very large difference in values in the same attribute of the data set, there are very large values and also very small values in the same characteristic column. If using conventional scaling methods, small values will be lost, and important potential objective factors may be lost. This violates the principle of fully preserving the nature of the data, criterion (1). Therefore, we perform scaling in our way through the method of transforming the number system. The method of transforming the number system aims at two goals simultaneously: One is to expand the attributes of the data to make the image size large enough during the data visualisation stage, and the other is to scale the value to preserve small values in the attribute. The details of the method are presented in Section 3.2.

In this study, we use the method of converting each observation data flow into an image, as in studies [10, 11]. The reason for choosing this transformation method is that the networking dynamics factors from network services, implicitly present in the network data, affect the quality of the classifier. If selecting features using traditional filtering methods, important network factors may be lost. Identifying the hidden factors in the data and deciding which features to retain during filtering is not a straightforward task. Therefore, to preserve the key characteristics of the network data, we convert each flow in the observation data into an image, retaining most of the relevant features to satisfy criterion (1). Converting data into an image is also a step aimed at satisfying the requirement of harmony between data and deep learning networks. In this study, we chose CNN as the neural network for the classification model. Therefore, converting network flow data into an image with a spatial architecture will help exploit the strengths of the CNN network, according to criterion (2).

The above data transformation processing steps aim to ensure criterion (1) to preserve the nature of the data and also to contribute to the harmony between the data and the CNN deep learning network, criterion (2). However, to maximise the harmony with CNN, it is possible to continue processing the image data using Sobel and Laplacian techniques to highlight areas with sudden changes, creating important features for CNN to learn easily. At the same time, applying the histogram balancing technique helps to spread the information in the image evenly, increasing the contrast so that CNN can detect details better. However, if considering criterion (3), it is possible to skip the complicated image processing steps.

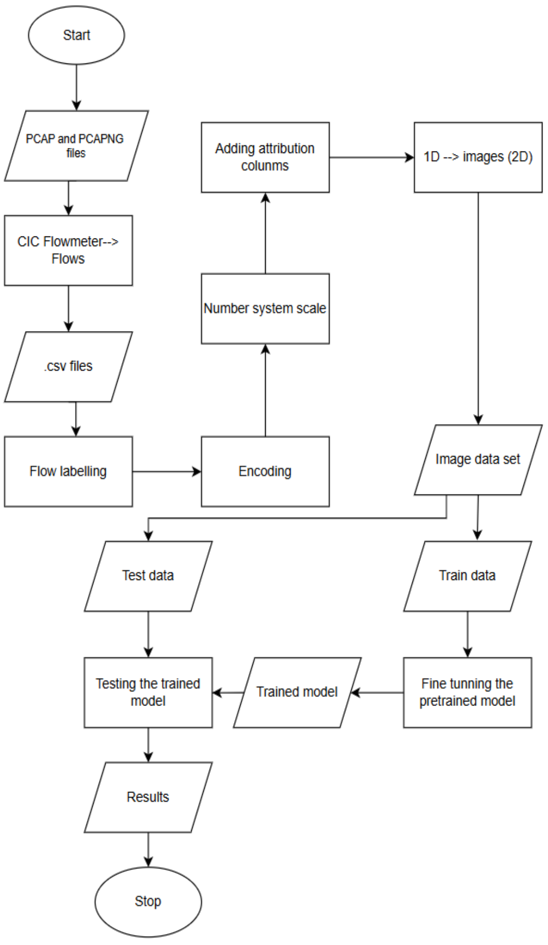


Figure 1. The pipeline of the proposed classification model construction process

In addition, in our design, we use the transfer learning method to take advantage of the high-quality pre-trained CNN models available. Therefore, another important factor to enhance the compatibility of the input data with the CNN deep learning network is the compatibility in the context of applying the transfer learning method. In transfer learning, a high-performance pre-trained CNN model for image classification is selected and additionally trained on the traffic image dataset. However, if the image format of the transformed image is different from the image format of the dataset that was previously used to train the pre-trained CNN, it will not be compatible. Therefore, we also propose a method to transform the traffic image data to the same format as the images in the image dataset that was used to train the selected pre-trained CNN model. The key point of this transformation method is to expand the properties of the original data to make the transformed image size similar to the image size in the dataset used to train the pre-trained CNN model. Image format conversion does not use the usual image resizing method because it will reduce image quality. The conversion method explained in detail below will contribute to expanding the number of features for this purpose, ensuring criterion (2).

3.2. Data Transformation Method

This section will explain in detail the method to achieve two goals at the same time: scaling the numeric values and expanding the number of data attributes.

As introduced above, in the attribute columns, there is a very different between the values. If applying the usual scaling methods, small attribute values will disappear, violating the criterion of preserving the nature of the data. Therefore, in such cases, our method converts the decimal values of the attributes to the 256 number system corresponding to the digits from 0 to 255. This value range is also the range of commonly used pixel values. Next, each digit will be separated into an added attribute.

For example: Convert the decimal value 2211115095 to the 256 number system
221111509510 = 131 202 240 87256

This splits into four attributes with corresponding values of 131, 202, 240, and 87.

We see that when the attribute values are less than 256, we get the converted value itself, which means that the small values have been preserved.

The number of attribute columns extended from the original attribute column will depend on the largest attribute value.

In our study, the addition of attributes is necessary, and the reason is explained as follows:

The pretrained CNN models are trained on a square image dataset of size 32x32. According to the principle of converting a data vector (1D) to an image (2D) of size 32x32, the data vector must have 1024 elements or attributes. However, the network data has a much smaller number of attributes, so when converted to an image, it will have a smaller size and will not be compatible with the pretrained CNN. Therefore, it is necessary to add an appropriate number of attributes to transform the image size tobe compatible with the pretrained CNN model. The above-mentioned scaling method provides a way to add this attribute.

The 256-number scaling method is a two-in-one solution. However, it is easy to see that this scaling method may not provide enough additional attributes. Another way to add attributes is needed. We propose another way, through three steps as follows:

-Step 1: Calculate the importance of all attributes in the scaled dataset
-Step 2: Select the attributes with the highest importance; the number of selected attributes is exactly equal to the number needed to be added, that is, the number missing after adding according to the above scaling method.
-Step 3: Double the selected attributes in place.

The entire process of expanding the number of attributes is summarised in formulas (1) and (2) as follows:

Let X be a data vector on the original data set with p attributes X=[x1,x2,…,xp], and S(X) is the scale function, we have:

$$X' = S(X) = \bigcup_{i=1}^{p} T(x_i) \qquad (1)$$ Type equation here.

T(xi)=[a_1, a_2,…,a_m], with m: number of numeric characters in the 256 number system of the max value of the feature.

X' is a data vector with number of attributes q>p, $X' = [x'_1, x'_2, \dots, x'_q]$

Let the required image size be nxn=d, and q is the length of vector X'

$$X'' = D(X') = \bigcup_{i=1}^{q} x'_i[copy(x'_i)] \qquad (2)$$

where:

- $copy(x'_i)$ is to add a feature of the same x'_i in the next x'_i
- $[copy(x'_i)] = copy(x'_i)\ \ if\ x'_i \in \text{The (d} - \text{q) highest important features}$
- $X''$is the final resulting data vector with d elements before being converted to a 2D image.
  $X'' = [x''_1, x''_2, \dots \dots, x''_d]$

4. EXPERIMENT AND RESULTS

The proposed method is validated through multiple experimental scenarios and compared against results from several recent representative studies. All experiments in this study were performed on a local computer with 2xCPU INTEL XEON 3.7 GHz-16 CORE/32 THREAD, 64GB DDR4, GPU Geforce RTX 3060 12GB.

4.1. Datasets

In the process of developing machine learning models, one of the important factors is choosing the appropriate dataset based on the purpose of the application. To date, the ISCX VPN-nonVPN Dataset (2016) [5] is still the dataset commonly used for the problem of classifying encrypted traffic. VPN-nonVPN Dataset is a dataset that stores network traffic divided into two types: Encrypted traffic (VPN) and Non-encrypted traffic (nonVPN). This dataset was created by the University of New Brunswick, Canada and published in 2016. This dataset includes 100,000 network traffic flows, of which 50,000 flows are Encrypted traffic (VPN) and 50,000 flows are Non-encrypted traffic (non-VPN). All were collected from two computers, one connected to VPN via an external VPN service provider, connected to that provider using OpenVPN (UDP mode), the other not connected to VPN. To generate SFTP and FTPS traffic, the authors also used the external service provider and FileZilla as a client. Network traffic was collected using Wireshark and TCPdump software, powerful tools for computer network engineers. The dataset was collected from 14 types of network traffic from many different applications, such as YouTube, Skype, Hangouts, etc. The entire dataset is 28GB in size and saved as .pcap and .pcapng files, a popular file format for storing packets. It includes 102 .pcap files and 38 .pcapng files; each file is identified by its file name as to which application it was collected. This is to support labelling for classification purposes.

The CIC-Darknet 2020 Dataset was developed by researchers from the CIC at the University of New Brunswick as a means to develop classification models for the early detection of malware and malicious activities. The dataset is a combination of two datasets published in 2016, the ISCX VPN-nonVPN Dataset (ISCXVPN2016), introduced above, and the Tor-nonTor dataset (ISCXTor2016). VPN and Tor traffic are combined into their respective Darknet categories. The goal is to detect darknet traffic and classify malicious activities using Tor. The dataset includes encrypted Tor traffic, regular non-Tor traffic, and applications over Tor such as the Tor browser, OnionShare, Ricochet, Skype, and BitTorrent. It also provides .pcap and .csv files. A lot of traffic uses TLS/Tor encryption.

From the survey and analysis of the above dataset, the VPN-nonVPN dataset was selected to build a classifier and evaluate the proposed solution. We use .pcap files and use CICFlowMeter to convert a series of packet transmission and reception operations into each network traffic flow based on the time axis represented in the .pcap file. After going through the conversion process into .csv format, the received dataset is represented with 83 attributes, and the "label" column is not labelled. Depending on the classification purpose, labels will be assigned appropriately to the data samples.

In many studies, authors have used nonVPN dataset and VPN dataset separately for application classification, which has shown positive results. However, the use of merged datasets for multi-class classification is still limited. Therefore, in this study, we focus on classifying encrypted traffic on the merged dataset to be more practical. The two classifiers that we are interested in building are a multi-class traffic classifier that distinguishes between encrypted and unencrypted traffic and a multi-class traffic classifier that does not care whether it is encrypted or not. To do this, we label the data with nonVPN and VPN distinctions in turn, creating a dataset with 11 traffic types as shown in Table 1. Next, we label the data without distinguishing between encrypted and unencrypted data, creating a dataset with 5 traffic types as shown in Table 2. In both cases, the dataset is in the form of mixed data of different types.

Table 1.Labled traffic categorizations in dataset of 11 classes.

Traffic type	**Content**
NonVPN_Streaming	youtube, vimeo, spotify, facebook_video, youtubehtml, skype_video
VPN_VoIP	vpn_facebook_audio, vpn_hangouts_audio, vpn_skype_audio, vpn_voipbuster
VPN_chat	vpn_aim_chat, vpn_facebook_chat, vpn_hangouts_chat, vpn_icq_chat, vpn_skype_chat
VPN_email	vpn_email
VPN_file transfer	vpn_ftps, vpn_sftp, vpn_skype_files
VPN_p2p	vpn_bittorrent
VPN_streaming	vpn_netflix, vpn_spotify, vpn_vimeo, vpn_youtube
NonVPN_VoIP	facebook_audio, skype_audio, voipbuster
NonVPN_chat	aim_chat, facebook_chat, skype_chat
NonVPN_Email	email,
NonVPN_File transfer	skype_files, scp, sftp

Table 2. Labeled traffic categorizations in dataset of 5 classes.

Traffic type	**Content**
Browse	Firefox, Chrome
Chat	aim_chat, facebook_chat, skype_chat,vpn_aim_chat, vpn_facebook_chat, vpn_hangouts_chat, vpn_icq_chat, vpn_skype_chat
File Transfer	skype_files, scp, sftp, vpn_ftps, vpn_sftp, vpn_skype_files
Video	youtube, vimeo, spotify, facebook_video, youtubehtml, skype_video, vpn_netflix, vpn_spotify, vpn_vimeo, vpn_youtube
VoIP	facebook_audio, skype_audio, voipbuster, vpn_facebook_audio, vpn_hangouts_audio, vpn_skype_audio, vpn_voipbuster

4.2. Selected Pretrained CNN Model

As presented above, the deep neural network CNN is used for the classification model, and the transfer learning method will be exploited in this study. However, the choice of a pretrained CNN model is also in the direction of the lowest possible computational demand to adapt well to practical implementation. While searching, we found that the pretrainedResNet18 model provided in [23] meets the purpose of verifying our method. According to [23], this model has been trained on the CIFAR10 dataset with image size 32x32 and achieved a published testing accuracy of nearly 95%. In fact, in the verification, we also received the accuracy result of 94%-95%. Therefore, we chose this pretrained model for the experiments here.

4.3. Evaluation Metrics

The proposed method is evaluated through the performance of the classification model. We also use standard metrics commonly used to evaluate a classification model, including accuracy, precision, recall, and F1-score. These standard parameters are calculated based on the measurements of the confusion matrix, including:

True Negatives (TN): The number of predictions in the case of an actual FALSE, which was predicted as FALSE
False Positives (FP): The number of predictions in the case of an actual FALSE, which was predicted as TRUE
False Negatives (FN): The number of predictions in the case of an actual TRUE, which was predicted as FALSE
True Positives (TP): The number of predictions in the case of an actual TRUE, which was predicted as TRUE

The formulas for calculating standard performance parameters are as follows:

$$Accuracy = \frac{TN + TP}{TN + FP + FN + TP} \quad (3)$$

$$Precision = \frac{TP}{TP + FP} \quad (4)$$

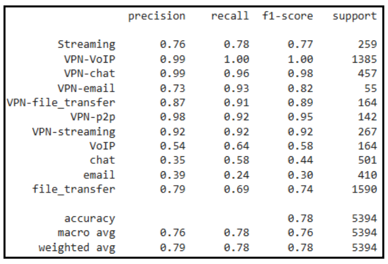
$$Recall = \frac{TP}{TP + FN} \quad (5)$$

$$F1 - score = \frac{TP}{TP + \frac{FN+FP}{2}} \quad (6)$$

4.4. Building a Multi-Class Classifier for 11 Traffic Types

To evaluate the proposed method, this section focuses on building a multi-class traffic classifier from a mixed dataset of 11 traffic types with or without encryption. In the first step, the .pcap files of the ISCX VPN-nonVPN traffic dataset are converted to flows and stored in .csv files, labelled according to 11 traffic classes as described in Table 1 and forming a mixed dataset. The next processing steps are as described in Figure 1 in Section 3. Most of the features of a data sample are retained, and through the initial transformation steps, the dataset has 83 features. String features such as IP addresses are also transformed. However, to have comparison results, we conduct experiments on the case of building a classifier using a dataset that has not been appliedto the proposed transformation method. Accordingly, the data will go through the initial transformation steps as shown in Figure 1, but do not apply (ignore) "number system scale" and "adding attribution columns". The data is converted to image format, and the final image dataset is used to train the pretrained model selected in section 4.2. The dataset is divided into train and test data in the ratio of 80:20. Applying transfer learning, the pretrainedResNet18 model retains most of the initial weight layers and fine-tunes the final layer.

The results of fine-tuning and model testing are shown in Figure 2. The classifier was built without the proposed data transformation step, resulting in very low results, with an accuracy of only about 78%.

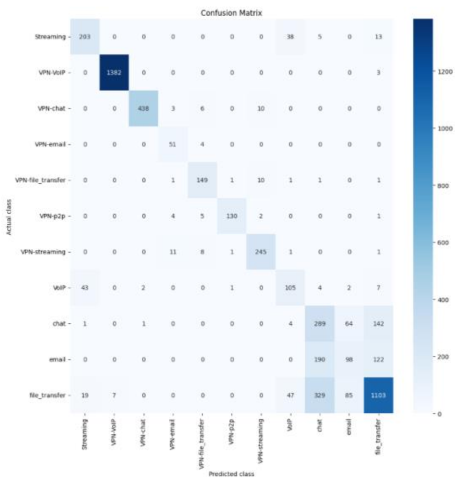


```
                   precision    recall  f1-score   support

        Streaming       0.76      0.78      0.77       259
         VPN-VoIP       0.99      1.00      1.00      1385
         VPN-chat       0.99      0.96      0.98       457
        VPN-email       0.73      0.93      0.82        55
VPN-file_transfer       0.87      0.91      0.89       164
          VPN-p2p       0.98      0.92      0.95       142
    VPN-streaming       0.92      0.92      0.92       267
             VoIP       0.54      0.64      0.58       164
             chat       0.35      0.58      0.44       501
            email       0.39      0.24      0.30       410
    file_transfer       0.79      0.69      0.74      1590

         accuracy                           0.78      5394
        macro avg       0.76      0.78      0.76      5394
     weighted avg       0.79      0.78      0.78      5394
```

(a)



(b)

Figure 2. The results of the 11 traffic type classification with the merged VPN-nonVPN dataset and the proposed method were not applied: (a) the values of evaluation parameters, (b) the confusion matrix.

Next, build a classifier that applies the data transformation method proposed in section 3.2, and each data sample is increased to 256 features. The entire data set with 256 features is converted into a 32x32 image set and is ready for model training. Some traffic images are depicted in Figure 3.

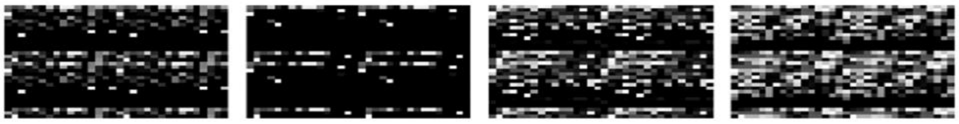
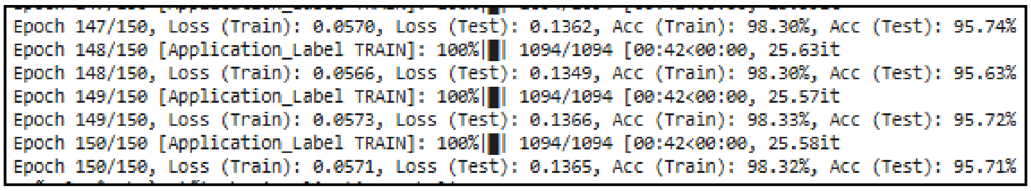


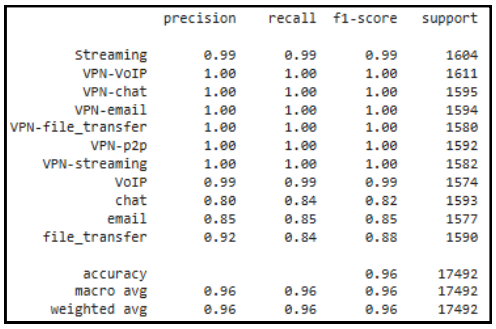
Figure 3. Some images in the image dataset were obtained after transformation, applying the proposed processing method.

The training data set and the test data set are divided in a ratio of 80:20. Here, we continue to apply the transfer learning technique using the pretrained ResNet18 model selected in section 4.2. At this time, the classifier is built with the proposed data transformation method, giving very promising results. As shown in Figure 4, the best result achieved during the fine-tuning process over 150 epochs has a test accuracy of 95.74%. Compared to the case without applying the transformation method, which only reached 78%, this result is much higher. The change in test accuracy over epochsis depicted in Figure 5, and the change in loss over epochs is depicted in Figure 6. Both show that the parameters reach stability after 100 epoches.



```
Epoch 147/150, Loss (Train): 0.0570, Loss (Test): 0.1362, Acc (Train): 98.30%, Acc (Test): 95.74%
Epoch 148/150 [Application_Label TRAIN]: 100%|█| 1094/1094 [00:42<00:00, 25.63it
Epoch 148/150, Loss (Train): 0.0566, Loss (Test): 0.1349, Acc (Train): 98.30%, Acc (Test): 95.63%
Epoch 149/150 [Application_Label TRAIN]: 100%|█| 1094/1094 [00:42<00:00, 25.57it
Epoch 149/150, Loss (Train): 0.0573, Loss (Test): 0.1366, Acc (Train): 98.33%, Acc (Test): 95.72%
Epoch 150/150 [Application_Label TRAIN]: 100%|█| 1094/1094 [00:42<00:00, 25.58it
Epoch 150/150, Loss (Train): 0.0571, Loss (Test): 0.1365, Acc (Train): 98.32%, Acc (Test): 95.71%
```

(a)

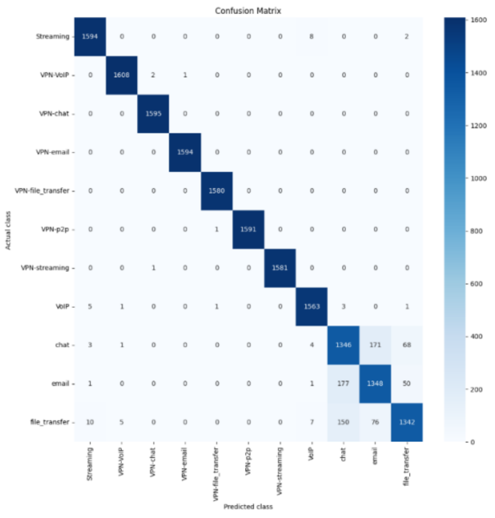


```
                   precision    recall  f1-score   support

        Streaming       0.99      0.99      0.99      1604
         VPN-VoIP       1.00      1.00      1.00      1611
         VPN-chat       1.00      1.00      1.00      1595
        VPN-email       1.00      1.00      1.00      1594
VPN-file_transfer       1.00      1.00      1.00      1580
          VPN-p2p       1.00      1.00      1.00      1592
    VPN-streaming       1.00      1.00      1.00      1582
             VoIP       0.99      0.99      0.99      1574
             chat       0.80      0.84      0.82      1593
            email       0.85      0.85      0.85      1577
    file_transfer       0.92      0.84      0.88      1590

         accuracy                           0.96     17492
        macro avg       0.96      0.96      0.96     17492
     weighted avg       0.96      0.96      0.96     17492
```

(b)



	Streaming	VPN-VoIP	VPN-chat	VPN-email	VPN-file_transfer	VPN-p2p	VPN-streaming	VoIP	chat	email	file_transfer
Streaming	1594	0	0	0	0	0	0	8	0	0	2
VPN-VoIP	0	1608	2	1	0	0	0	0	0	0	0
VPN-chat	0	0	1595	0	0	0	0	0	0	0	0
VPN-email	0	0	0	1594	0	0	0	0	0	0	0
VPN-file_transfer	0	0	0	0	1580	0	0	0	0	0	0
VPN-p2p	0	0	0	0	1	1591	0	0	0	0	0
VPN-streaming	0	0	1	0	0	0	1581	0	0	0	0
VoIP	5	1	0	0	1	0	0	1563	3	0	1
chat	3	1	0	0	0	0	0	4	1346	171	68
email	1	0	0	0	0	0	0	1	177	1348	50
file_transfer	10	5	0	0	0	0	0	7	150	76	1342

(c)

Figure 4. The results of the 11 traffic type classification with the merged VPN-nonVPN dataset and the proposed method wereapplied : (a) the best accuracy during test, (b) the values of evaluation parameters,and(c) the confusion matrix.

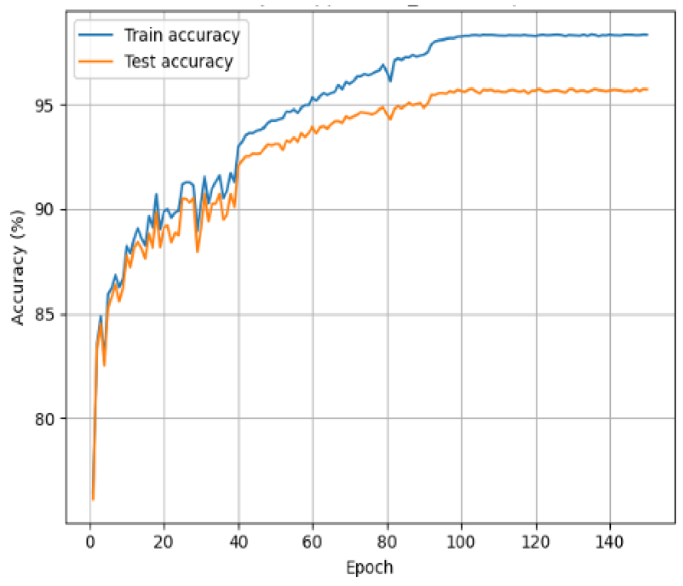


Figure 5. Accuracy over epochs when training and testing in the case of the proposed method was applied in the 11 traffic type classification with the merged VPN-nonVPN dataset.

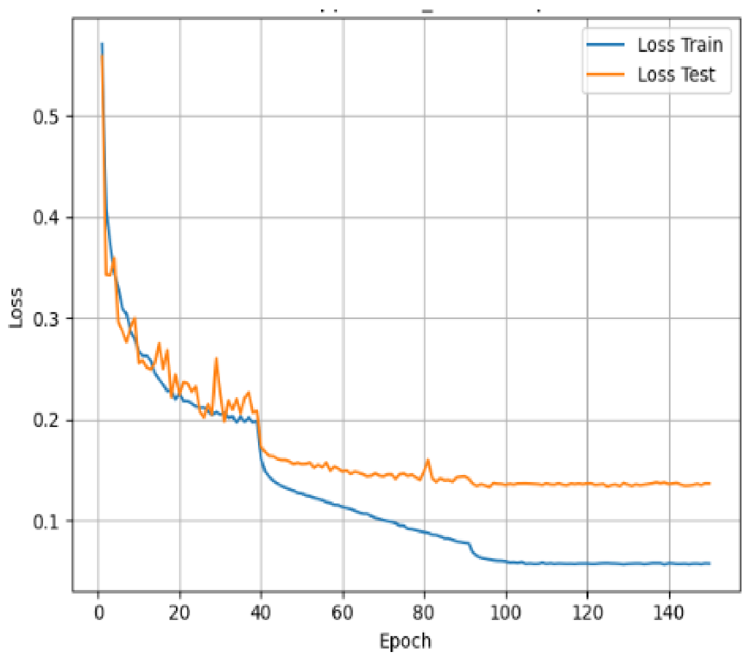


Figure 6. Loss over epochs during training and testing in the case of the proposed method was applied in the 11 traffic type classification with the merged VPN-nonVPN dataset.

4.5. Building a Multi-Class Classifier for 5 Traffic Types

In this section, we continue to apply the proposed method to build a multi-class traffic classifier from a mixed dataset consisting of 5 types of traffic, regardless of whether it is encrypted or not. The .pcap files of the ISCX VPN-nonVPN traffic dataset are converted into flows and saved as .csv files, labelled according to 5 traffic classes as shown in Table 2 and form a merged dataset.

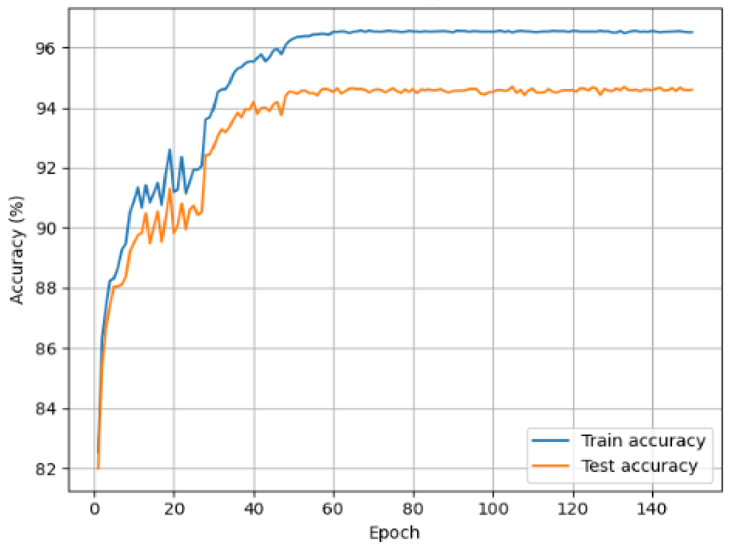


Figure 7. Accuracy over epochs when training and testing in the case of the proposed method was applied in the 5 traffic type classification with the merged VPN-nonVPN dataset

The further processing steps are described in Figure 1 in Section 3. Most of the features of a data sample are also retained, and through the initial transformation steps, the dataset has 83 features. The string features, such as IP addresses, are also transformed as done on the dataset with 11 types of traffic above. The dataset after the encoding step, with 83 features, is transformed according to the proposed method in Section 3.2 into a dataset with 256 features. Each data sample in the dataset is further converted into a 2D image to obtain the final image dataset ready for model training. Applying transfer learning, once again, the pretrained ResNet18 model selected in section 4.2 is used to build the classifier.

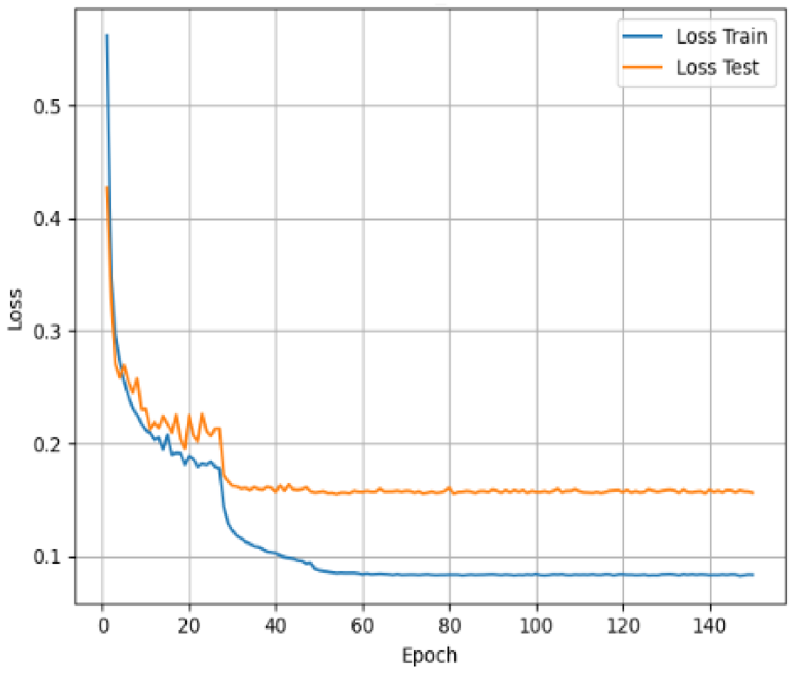
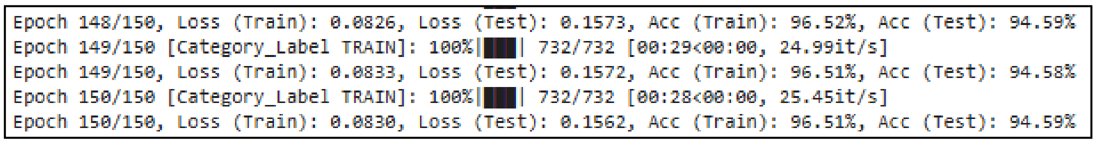


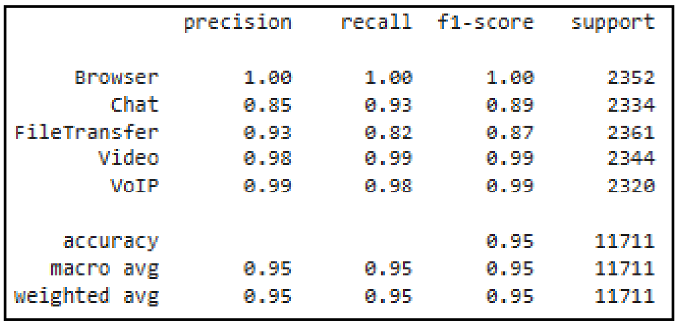
Figure 8. Loss over epochs during training and testing in the case of the proposed method was applied in the 5 traffic type classification with the merged VPN-nonVPN dataset.

The final image dataset is split into train and test sets in the ratio of 80:20. The train dataset is used to fine-tune the pretrainedResNet 18 model in 150 epochs. The change in testing accuracy over epochsis depicted in Figure 7, and the change in loss over epochs is depicted in Figure 8. Both show that stability is achieved after 60 epochs. As depicted in Figure 9(a), the best testing accuracy achieved in this case is 94.59%. The resulting parameters depicted in Figures 9(b) and 9(c) both show that the quality of the model is satisfactory according to the initial target. Some comparisons with new results are presented in the following section.



```
Epoch 148/150, Loss (Train): 0.0826, Loss (Test): 0.1573, Acc (Train): 96.52%, Acc (Test): 94.59%
Epoch 149/150 [Category_Label TRAIN]: 100%|███| 732/732 [00:29<00:00, 24.99it/s]
Epoch 149/150, Loss (Train): 0.0833, Loss (Test): 0.1572, Acc (Train): 96.51%, Acc (Test): 94.58%
Epoch 150/150 [Category_Label TRAIN]: 100%|███| 732/732 [00:28<00:00, 25.45it/s]
Epoch 150/150, Loss (Train): 0.0830, Loss (Test): 0.1562, Acc (Train): 96.51%, Acc (Test): 94.59%
```

(a)

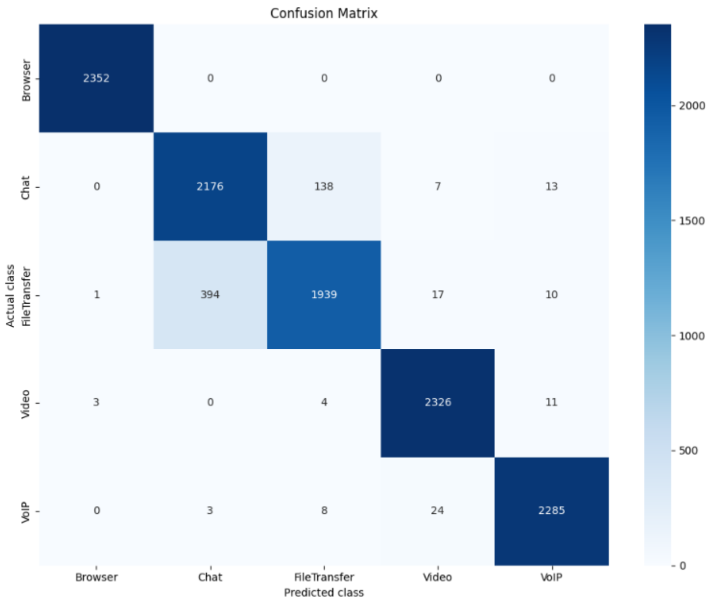


```
              precision    recall  f1-score   support

     Browser       1.00      1.00      1.00      2352
        Chat       0.85      0.93      0.89      2334
FileTransfer       0.93      0.82      0.87      2361
       Video       0.98      0.99      0.99      2344
        VoIP       0.99      0.98      0.99      2320

    accuracy                           0.95     11711
   macro avg       0.95      0.95      0.95     11711
weighted avg       0.95      0.95      0.95     11711
```

(b)



(c)

Figure 9. The results of the 5 traffic type classification with the merged VPN-nonVPN dataset and the proposed method were applied: (a) the best accuracy during test, (b) the values of evaluation parameters, (c) the confusion matrix.

5. Discussion

The results in Section 4.4 show that if we omit the data transformation procedure proposed in Section 3.2, the 11-class traffic classifier has much lower performance parameters than when this transformation procedure is applied. Specifically, the accuracy measurement when applying the proposed method is 95.74% compared to the accuracy of 78% when not transforming the data by this method, as depicted in Figure 2 and Figure 4. This proves that the proposed method has achieved superior performance when fine-tuning the pretrained model with preserved data and similar to the data used to train the previous model. The proposed method also achieved satisfactory results in the case of classifying 5 traffic types, regardless of whether they are encrypted or not. The accuracy achieved, as depicted in Figure 9, is 94.59%, which is completely comparable to current advanced methods. Table 3 compares with some recent typical studies that have similarities in using the same dataset and have simple configurations that are easy to adapt to real conditions. Some other studies have been published recently with quite high results. However, these studies do not address the conservation of traffic flow dynamics and have complex configurations that are difficult to adapt to real conditions and are not suitable for comparison.

Table 3.The comparison of results from several similar studies using the ISCX VPN-nonVPN dataset.

	Method	**Number of classes**	**Merged VPN-nonVPN dataset**	**Accuracy (%)**	**Remark**
1	ENTC using self-supervised learning [17]	3	Yes	86.56	Flow-based, Self-supervised, resnet34
2	ENTC using self-supervised learning [17]	3	No	-	
3	CNN-based ENTC[11]	10	Yes	-	
4	CNN-based ENTC[11]	≤10	No	93.86	Flow-based, Flowpic, VPN dataset and non-VPN dataset separately
5	FlowPic ENTC [10]	10	Yes	-	
6	FlowPic ENTC [10]	10	No	94.22	Flow-based, Flowpic, non-VPN dataset only
7	FlowPic ENTC[10]	3	Yes	83	Flow-based, Flowpic
8	FlowPic ENTC[10]	5	**Yes**	**91.3**	Flow-based, Flowpic
9	Few-shot traffic classification [12]	12	No	-	
10	Few-shot traffic classification [12]	12	**Yes**	**94.33**	Flow-based
11	Few-shot traffic classification [12]	6	No	97.67	Flow-based, VPN dataset only
12	Few-shot traffic classification [12]	6	Yes	-	
13	**Our method**	11	**Yes**	**95.62**	Flow-based, Flowpic
14	**Our method**	5	**Yes**	**94.71**	Flow-based, Flowpic

Table 3 only compares with the studies on the same ISCX dataset, which has two subsets, VPN dataset and nonVPN dataset. The studies also have in common the use of CNN and all group raw data in flows (flow-based). Among them, two studies [10-11] both convert flow data into images to exploit the strengths of CNN, like our study. The cases presented in the table are distinguished by the number of classes in the corresponding multi-class traffic classification problem. At the same time, they are also distinguished by the classification method on separate subset datasets or classification on merged datasets that are mixed with both.

From Table 3, it can be observedthat when the classifier is built and tested on each separate dataset, the VPN dataset or nonVPN dataset often achieves a fairly high classification accuracy of over 93% as shown in rows 4, 6, and 11 of Table 3. On row 11 is the result from the latest study [12] with an accuracy of 97.67%. However, [12] classifies 6 traffic classes on the VPN dataset only. From Table 3, it is also shown that the multi-class classification problem presents a significant challenge when combining the VPN dataset and the nonVPN dataset into a merged dataset, called the merged VPN-nonVPN dataset. Most of them have an accuracy below 91.3%, except for the recently published study [12], which achieved 94.33% when classifying 12 classes on the merged dataset, on row 10 of the table. In this study, we choose the more difficult case, focusing on classifying 11 applications and 5 traffic types on the merged VPN-nonVPN dataset. Our results when classifying 11 applications achieved an accuracy of 95.74%. Although the accuracy is not much higher, about 1.41%, compared to [12], our method includes the preservation and adaptation of reality as the original goal. In particular, on row 14 of the table, our classification results of 5 traffic types achieved an accuracy of 94.71%. This result is a breakthrough compared to the result of 91.3% of [10] on row 8 of the table, which also classified the same 5 traffic types. During the evaluation of the method, we tried to find the most common evaluation scenarios from previous studies to make the evaluation more objective. However, in some cases, as in the table, we did not have data for comparison, including rows 2, 3, 5, 9, and 12.

6. CONCLUSIONS

A high-performance and adaptable encrypted network traffic classifier has been developed. The goal has been achieved based on the design and construction principles revolving around three criteria: preserving data information, aligning data with deep learning models, and minimising the computational scale. Qualitatively compared with current advanced methods, our method ensures these three criteria at the same time. Quantitatively, the proposed method has been tested on a multi-class classification problem with a typical merged dataset that is more difficult than the separate datasets. The experimental results show that our method has achieved higher accuracy than state-of-the-art methods under comparable test scenarios. To the best of our knowledge, our paper introduces two novel concepts: (1) data preprocessing techniques that fully preserve the intrinsic characteristics of real network traffic, and (2) the harmonisation of input data formats with deep learning neural networks. This can be seen as a starting point for the efficiency and sustainability of the model. Our method still has many aspects forperformance improvement. For example, when the computational infrastructure allows, we can increase the level of processing to better harmonize the input data with the deep learning model or upgrade the deep learning model. Another research aspect is to quantify the computational scale to compare and balance the demand, and to estimate and control the network dynamics that affect the efficiency of the model. These aspects pave the way for a new phase of our research.

CONFLICT OF INTERESTS

The authors declare no conflict of interest.

REFERENCES

[1] Jiuxing Zhou, et al, Challenges and Advances in Analyzing TLS 1.3-Encrypted Traffic: A Comprehensive Survey, Electronics 2024, 13, 4000. https://doi.org/ 10.3390/electronics13204000

[2] Aleisa, M.A., Traffic classification in SDN-based IoT network using two-level fused network with self-adaptive manta ray foraging. Sci Rep 15, 881 (2025). https://doi.org/10.1038/s41598-024-84775-5

[3] M. Jiang et al., "FA-Net: More Accurate Encrypted Network Traffic Classification Based on Burst with Self-Attention," 2023 International Joint Conference on Neural Networks (IJCNN), Gold Coast, Australia, 2023, pp. 1-10, doi: 10.1109/IJCNN54540.2023.10191615.

[4] Yehonatan Zion, et al, Enhancing Encrypted Internet Traffic Classification Through Advanced Data Augmentation Techniques, arXiv:2407.16539v1 [cs.LG], (2024), https://doi.org/10.48550/arXiv.2407.16539

[5] G. Draper-Gil, A. H. Lashkari,M. S. I.Mamun, and A. A. Ghorbani, “Characterization of encrypted and VPN traffic using time-related features,” in Proc. 2nd Int. Conf. Inf. Syst. Secur. Privacy, 2016, pp. 407–414.

[6] Reham T. Elmaghraby, et al, Encrypted network traffic classification based on machine learning, Ain Shams Engineering Journal, Volume 15, Issue 2, 2024, 102361, ISSN 2090-4479, https://doi.org/10.1016/j.asej.2023.102361.(2023)

[7] Alwhbi, I.A.; Zou, C.C.;Alharbi, R.N. Encrypted Network Traffic Analysis and Classification Utilizing Machine Learning. Sensors 2024, 24, 3509. https://doi.org/10.3390/s24113509

[8] A. T. -J. Akem, G. Fraysse and M. Fiore, "Encrypted Traffic Classification at Line Rate in Programmable Switches with Machine Learning," NOMS 2024-2024 IEEE Network Operations and Management Symposium, Seoul, Korea, Republic of, 2024, pp. 1-9, doi: 10.1109/NOMS59830.2024.10575394.

[9] Lotfollahi, M.; et al, Deep packet: A novel approach for encrypted traffic classification using deep learning. Soft Comput. 2020, 24, 1999–2012

[10] T. Shapira and Y. Shavitt, "FlowPic: A Generic Representation for Encrypted Traffic Classification and Applications Identification," in IEEE Transactions on Network and Service Management. 2021, doi: 10.1109/TNSM.2021.3071441

[11] Zulu Okonkwo, Ernest Foo, Qinyi Li, and ZheHou. 2022. A CNN Based Encrypted Network Traffic Classifier. In Proceedings of the 2022 Australasian Computer Science Week (ACSW '22). Association for Computing Machinery, New York, NY, USA, 74–83. https://doi.org/10.1145/3511616.3513101

[12] Xu, S., Han, J., Liu, Y. et al. Few-shot traffic classification based on autoencoder and deep graph convolutional networks. Sci Rep 15, 8995 (2025). https://doi.org/10.1038/s41598-025-94240-6

[13] Nuñez-Agurto, D.; et al, Novel Traffic Classification Approach by Employing Deep Learning on Software-Defined Networking. Future Internet 2024, 16, 153. https://doi.org/10.3390/fi16050153

[14] S. Salmi, M. Bagaa, M. A. Ouameur, O. Bekkouche and A. Ksentini, "SDN-based Network Traffic Classification using Deep Reinforcement Learning," GLOBECOM 2024 - 2024 IEEE Global Communications Conference, Cape Town, South Africa, 2024, pp. 3715-3720, doi: 10.1109/GLOBECOM52923.2024.10901070.

[15] Salau, A.O., Beyene, M.M. Software defined networking based network traffic classification using machine learning techniques. Sci Rep 14, 20060 (2024). https://doi.org/10.1038/s41598-024-70983-6

[16] Huang, H.; Zhou, Y.; Jiang, F. CLA-BERT: A Hybrid Model for Accurate Encrypted Traffic Classification by Combining Packet and Byte-Level Features. Mathematics 2025, 13, 973. https://doi.org/10.3390/math13060973

[17] M. S. Towhid and N. Shahriar, "Encrypted Network Traffic Classification using Self-supervised Learning," 2022 IEEE 8th International Conference on Network Softwarization (NetSoft), Milan, Italy, 2022, pp. 366-374, doi: 10.1109/NetSoft54395.2022.9844044.

[18] NimeshaWickramasinghe, ArashShaghaghi, Elena Ferrari, Sanjay Jha, Less is More: Simplifying Network Traffic Classification Leveraging RFCs, arXiv:2502.00586v2 [cs.CR], 2025, https://doi.org/10.48550/arXiv.2502.00586

[19] Xi, T.; et al. SwiftSession: A Novel Incremental and Adaptive Approach to Rapid Traffic Classification by Leveraging Local Features. Future Internet 2025, 17, 114. https://doi.org/10.3390/fi17030114

[20] Xinjie Lin, et all. ET-BERT: A Contextualized Datagram Representation with Pre-training Transformers for Encrypted Traffic Classification. In Proceedings of the ACM Web Conference 2022 (WWW '22). Association for Computing Machinery, New York, NY, USA, 633–642. https://doi.org/10.1145/3485447.3512217

[21] Ruijie Zhao, et al. Yet another traffic classifier: a masked autoencoder based traffic transformer with multi-level flow representation. In Proceedings of the Thirty-Seventh AAAI Conference on Artificial Intelligence (AAAI'23/IAAI'23/EAAI'23), Vol. 37. AAAI Press, Article 605, 5420–5427. https://doi.org/10.1609/aaai.v37i4.25674

[22] MezatiMessaoud, Classification of Network Traffic using Machine Learning Models on the NetML Dataset, International Journal of Computer Networks and Communications, May 2025, Volume 17, Number 3.

[23] https://huggingface.co/edadaltocg/resnet18_cifar10

AUTHORS

Nguyen Hong Son received his B.Sc. in Computer Engineering from HCMC University of Technology in HoChiMinh City, and his M.Sc. and PhD in Communication Engineering from the Posts and Telecommunications Institute of Technology in Hanoi. His current research interests include communication engineering, information systems, AI/ML/DL, network security, and cloud computing.

Nguyen Trung Hieu received his B.Sc. in Information Technology from HCMC Open University in HoChiMinh City and his M.Sc. in Information Systems from the Posts and Telecommunications Institute of Technology in Hanoi. His research areas are communication engineering, information systems, and AI/ML/DL