Fine-grained Traffic Classification Based on Improved Residual Convolutional Network in Software Defined Networks

Chang Su, Yanqing Liu, and Xianzhong Xie

***Abstract*—As Internet traffic becomes increasingly complex, a growing number of applications are carrying different services. To provides service-aware traffic classification, allowing network operators to better analyze network composition as well as manage and schedule efficiently network resources to facilitate sustainable development network, fine-grained traffic classification has become a crucial and challenging problem. Software defined networks has become the most promising new network architecture in the future with two major advantages of centralized control and programmability. Deep learning has also become a mainstream technology by its excellent feature extraction ability for large datasets. Thus the combination of software defined networks and deep learning can effectively solve the challenge of data set collection and feature extraction in the field of traffic classification. Inspired by research in computer vision, in this paper we propose an improved residual convolutional network approach to network traffic classification, which addresses the network degradation problem that occurs with increasing network depth in traditional deep learning methods for fine-grained network traffic identification, and can effectively learn deeper network features to achieve fine-grained network traffic classification. Experimental results show that the overall accuracy of the proposed method can reach 99.93%, which is about 4%, 13%, 7% and 2% higher than other state-of-the-art models such as classical residual networks ResNet-18, One-dimensional Convolutional Neural Networks (1D-CNN), Long Short-Term Memory (LSTM) and combined model CNN-LSTM respectively, thus verify the effectiveness of our method.**

***Index Terms*—Deep Learning (DL), Fine-grained Traffic Classification, Long Short-Term Memory (LSTM), Residual Convolutional Networks, Software Defined Networks (SDN).**

I. Introduction

Software Defined Network is an emerging and promising network model that can greatly simplify network management, improve network resource utilization, optimize network performance and promote innovation and growth [1]. SDN undoubtedly offers new opportunities to achieve intelligence within the network. However, there are many challenges in the development and application of software defined networks, one of which is fine-grained traffic classification.

With the development of network technology and various new applications emerging, network traffic is growing exponentially, making the network overwhelmed and prone to problems such as wasted resources and network congestion [2]. Therefore, classifying traffic into multiple priorities through traffic classification is an extremely important network function. The results of classification algorithms can be classified into two categories: coarse-grained and fine-grained [3]. Coarse-grained classification identifies traffic into several categories based on coarse traffic types such as protocols or applications. Nevertheless, fine-grained classification aims to distinguish individual applications or service types instead of classifying them into coarse traffic categories, which facilitates operators to further analyze traffic composition and efficiently allocate network resources to improve Quality of Service (QoS) and Quality of Experience (QoE) [4], as well as to further promote sustainable network development.

In addition, the results of traditional machine learning methods for traffic classification are very dependent on the feature engineering of network traffic, which requires a lot of time and manpower, and is not applicable to some core or edge devices. Deep learning is an end-to-end learning approach, and its output is not affected by the input data features and does not require human extraction of features, which gets rid of the heavy feature learning and feature selection work, so the deep learning-based methods can perform more accurate, faster, and finer-grained traffic classification problems. Contributions of this paper are summarized as following three folds:

1) We reconstruct a fine-grained traffic dataset named CIC_TF based on the public traffic and then present a complete preprocessing algorithm to convert it into image data for further training.
2) A new residual convolutional network is proposed for the traffic classification task, which solves the gradient disappearance problem of traditional deep learning methods when performing fine-grained classification.
3) Extensive experiments conducted on the other datasets and state-of-the-art methods show that the method proposed in this paper significantly outperforms other in terms of the accuracy and other metrics, demonstrating its superiority to effectively classify different network traffic and identify its underlying application types.

To the best of our knowledge, this interesting attempt is the first application of residual convolutional network approach to fine-grained traffic classification domain. In this paper, a new method based on representation learning and residual convolutional networks is proposed to achieve more efficient and fine-grained traffic classification in SDN.

College of Computer Science and Technology, Chongqing University of Posts and Telecommunications, Chongqing, China

II. Related Work

With the iteration of technology, artificial intelligence has gradually applied to various areas of the network such as network resource scheduling [5], multimedia data processing [6], traffic classification [7], anomaly detection [8], etc.

Software Defined Networks have been widely discussed and studied as very promising network architecture. Xie et al. [3], [9] summarize the various challenges and problems of applying deep learning (DL) methods to SDN and discusses the application of machine learning (ML) algorithms in detail. It is a good overview for the study of traffic classification tasks. Yan et al. [7] present several representative works on traffic classification in SDN, discusses the research challenges and future directions of traffic classification, and provide inspiration for the research direction of this paper. Wang et al. [4], [10] provides a comprehensive survey of state-of-the-art traffic classification methods, focusing on the deep learning-based classification of encrypted traffic for mobile services, and discusses some noteworthy issues and challenges.

Network traffic classification techniques have evolved through roughly three stages so far:

A. Port-based And Payload-based Traffic Classification

Port-based and payload-based [11] traffic classification is the earliest classification technology, but with the emergence of dynamic ports and encrypted traffic, the identification accuracy of this method has significantly decreased.

B. Machine Learning-based Traffic Classification

Currently popular machine learning-based traffic classification can be divided into three categories: supervised learning [12], unsupervised learning [13], and semi-supervised learning [14]. These methods are mainly based on some external statistical characteristics of the traffic, but requires complex feature engineering, which increases the complexity of the methods.

C. Deep Learning-based Traffic Classification

The feature extraction and selection of deep learning-based methods are done automatically by model training, compared with traditional machine learning methods, deep learning methods have higher learning ability and can learn the features of the original traffic directly without additional manually designed unique or private traffic information, these characteristics make DL-based approach is a highly desirable traffic classification method [1], [15]–[18]. Wang [8] proposes an end-to-end encrypted traffic classification method with a one-dimensional convolutional neural network (1D-CNN) that integrates feature extraction and classifier into a unified end-to-end framework. Hohammad [19] proposes a Deep Packet framework, which implements Stacked Auto-Encoder (SAE) and 1D-CNN, respectively, for automatic extraction of network traffic to accurately classify traffic. Zeng [20] proposes a lightweight traffic classification and intrusion detection framework based on deep learning, which implements 1D-CNN, Long Short Term Memory (LSTM), and SAE. In [21], a semi-supervised learning encrypted traffic classification method based on Generative Adversarial Networks (GAN), is proposed to be embedded in SDN edge gateways to achieve the goal of traffic classification and further improve network resource utilization. Lin [2] proposes a new encrypted traffic classification scheme TSCRNN based on stream spatio-temporal features for efficient management of industrial Internet of Things (IoT). Lan [22] proposes a novel self-attentive deep learning method for dark network traffic classification and application identification. Reference [23] proposed an online multimedia traffic classification framework based on convolutional neural network (CNN), which is capable of fast early classification and class incremental learning.

In summary, there are several major problems with the current related traffic classification literature as follows. First, most researchers have selected traffic datasets with high consistency in each category and clear differentiation between different categories of traffic, and thus are able to achieve high accuracy. Secondly, there is no unified traffic pre-processing scheme, and various methods of feature processing or traffic extraction substantially increase the time consumption of pre-processing. Third, in terms of model selection, most studies have chosen CNN and LSTM, and only a few articles discuss the performance of other deep learning approaches. To tackle these issues, this paper firstly selects datasets with both discrimination and similarity between categories from public dataset websites, and then processes the original network traffic dataset into image data using the unified traffic preprocessing method and further transforms it into the model-recognizable form, and finally proposes an improved model framework based on residual convolutional networks to classify the preprocessed data and conducts several comparative experiments to verify the reliability of the model.

III. Problem Definition

The segmentation granularity and classification granularity of the original traffic is a prerequisite that needs to be clarified before performing the traffic classification task.

A. Segmentation Granularity

Network traffic splitting granularity includes TCP connections, flows, sessions, services, and hosts [24]. Different split granularity results in different traffic units. Flows are defined as all packets with the same 5-tuple (timestamp, source IP, destination IP, protocol, and packet length). Sessions are bidirectional flows, including traffic in both directions. In addition, the inherent characteristics of the flow are intuitively reflected in the application layer, and the data of other layers should also contain some traffic characteristic information. Therefore, this paper uses the segmentation granularity of all layers and sessions to retain the data characteristics of the original traffic to the maximum extent.

Suppose network traffic T consists of a series of consecutive packets $P = \{p_1, p_2, \ldots, p_i, \ldots, p_n\}$, where p_i represents the i-th packet. By using the traffic slicing tool SplitCap [25] to extract the five tuples from the packet headers, each

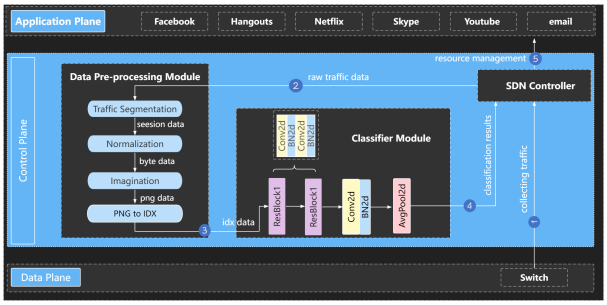


Fig. 1. Basic architecture of fine-grained traffic classification in SDN.

packet can be defined as $p_i = (t_i, \mathrm{src}_i, dst_i, \mathrm{pro}_i, \mathrm{len}_i), i \in \{1, 2, \ldots, |P|\}$, where t_i represents the timestamp, src_i represents the source address, dst_i represents the destination ip address, $proc_i$ represents the protocol type, and len_i represents the packet length.

B. Classification Granularity

The first step in building a traffic classifier is to specify the classification goals. Typical objectives include Quality of Service (QoS), resource allocation, and intrusion detection. To achieve these goals, traffic classes can be classified according to the following classification granularity.

- protocols (e.g. Ftp, Http)
- applications (e.g. Youtube, QQ)
- service types (e.g. Video, Audio, Chat)

Therefore, the goal of traffic classification is to label each flow with the corresponding traffic category. This paper focuses on solving the more complex traffic classification problem based on service type.

C. Basic Architecture

The traffic classification task in SDN can be summarized in the following steps: (1) The switch collects network traffic and sends it to the controller via OpenFlow protocol. (2) The controller extracts the traffic data and then sends the features to the preprocessing module to process the raw data set. (3) The processing results are sent to the classification module to classify the flows. (4) The classification results are then sent to the SDN controller. (5) The controller analyzes the traffic composition and makes efficient management and scheduling decisions based on the classification results [7]. The specific process is shown in Fig. 1.

IV. Data

A. Data Preparation

Due to the limitations of the experimental conditions, we cannot collect data from the actual SDN environment, so we obtained the original datasets of 16 real-world environments from the Canadian Cyber Security Institut [26], and named them CIC_TF, this dataset involves several common classes of audio, video, and session traffic datasets, including both coarse-grained datasets of different applications and fine-grained datasets of different sessions of the same application.

TABLE I
Composition of the Dataset.

Type	Quantity	ratio (%)	Type	Quantity	ratio (%)
email	342	2.41	hangout_chat	34	0.24
facebook_chat	31	0.22	Outlook	1413	9.96
facebook_video	16	0.11	skype_audio	9	0.06
FTP	6000	42.31	skype_chat	428	3.02
ftps_down	7	0.05	vimeo	12	0.08
ftps_up	13	0.09	spotify	14	0.10
Gmail	1638	11.55	weibo	4113	29.00
hangouts_audio	88	0.62	youtube	24	0.17

The number of packets contained in each class is shown in Table I, where the proportion of various traffic data varies and the video traffic data accounts for a larger proportion, which is in line with the traffic distribution in real traffic distribution.

B. Pre-processing Process

The purpose of this process is to convert the original traffic data in PCAP format into data in IDX format that can be recognized by the residual convolutional network and to solve problems such as data inconsistency and redundant data. It consists of four steps: traffic segmentation, normalization, imagination, and PNG to IDX.

- **Traffic segmentation.** We split the original continuous PCAP traffic data into smaller discrete PCAP traffic data in the form of session and all layers as granularity and then remove the empty files and duplicate files to eliminate interference.
- **Normalization.** This step trims all files to a uniform length. We unify the files to 784 bytes (28*28), if the file size is less than 784 bytes, add 0x00 at the end to make up to 784 bytes. Finally, all files are divided into test and training sets in a ratio of 1:9.
- **Imagination.** This step converts the files with the same size processed in the previous step into grayscale images. Each byte of the original file represents one pixel, 0x00 means black, and 0xff means white.
- **PNG to IDX.** This step requires further conversion of the image data in PNG format to a data format recognizable by the network model. We use the IDX padding generator to convert the image data of the same size into a two-dimensional format IDX file.

Algorithm 1 describe the detailed steps of the whole pre-processing process in detail.

C. Visualization Analysis

In this section, some of the images generated in step 3 of the data preprocessing process are visualized as shown in Fig. 2, where each grayscale image has a size of 28 * 28 bytes. As can be seen from Fig. 2, there is no obvious degree of distinction between the traffic of different service types of the same application, such as audio, chat, and video traffic in facebook. There is also no high degree of consistency between uniform service type traffic from different applications, such as facebook_video, hangots_video, and youtube. Therefore, it is necessary to learn as many features as possible to maximize the differentiation of these smaller granularity service traffic.

Algorithm 1 Data Preprocessing

Input: $T_1, T_2, \ldots, T_n$
Output: Processed Dataset $G(G, L)$
for each p in $(1, n)$ **do**
  Split the T_p into $T_p^1, T_p^2, \ldots, T_p^i, \ldots$
  for each i **do**
    Purify T_p^i
  end for
  Remove duplicated packet from $T_p^1, T_p^2, \ldots, T_p^i, \ldots$
  for each i **do**
    Divide the test set and training set into 1:9
    Normalize the length of T_p^i to 784 bytes
  end for
  for each i **do**
    Generate PNG traffic image $G_p^1, G_p^2, \ldots, G_p^i, \ldots$
  end for
  for each i **do**
    Changing MSB for image data (G_p^j, L_p^j)
  end for
end for

Fig. 2. Data visualization.

V. Research Methodology

A. Overview of Residual Convolutional Networks

The main idea of the residual network is identity mapping to ensure that the deep layer effect is not weaker than the shallow one [27]. The basic structure of the residual blocks is shown in Fig. 3.

As shown in Fig. 3, the residual block is divided into two parts: identity mapping and residual mapping. Suppose the input of the neural network is X, and the expected output is $\mathrm{H(X)}$, the input X is directly passed to the output by means of direct mapping (i.e., the right part of the curve in the figure), and if the channels of $\mathrm{F(X)}$ and X are the same, the output result is $\mathrm{H(X)} = \mathrm{F(X)} + \mathrm{X}$, and if the channels of $\mathrm{F(X)}$ and X are not the same, the number of channels is adjusted by using the special one-dimensional convolutional kernel to make it the same as the channels of X, and the output result is $\mathrm{H(X)} = \mathrm{F(X)} + \mathrm{WX}$, when $\mathrm{F(X)} = 0$, then the output result in both cases are $H(X) = X$, that is identity mapping.

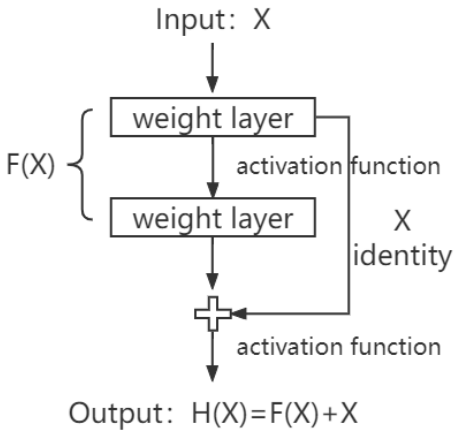


Fig. 3. Basic structure of the residual block.

B. Convolutional Layer

The biggest advantages of convolutional networks over traditional neural networks are parameter sharing and sparse connections, which greatly reduce the complexity of neural network model training and have been widely used in various fields such as image recognition and target detection. Suppose the input shape is $n_h * n_w$, the shape of the convolutional kernel is $k_h * k_w$, the padding size is p, and the step size is s, then the output shape is calculated as follows [28]

$$\lfloor \frac{n_h + 2p - kn}{s} + 1 \rfloor * \lfloor \frac{n_w + 2p - k_w}{s} + 1 \rfloor \tag{1}$$

The complete feature map is obtained by multiple convolution kernels, and the output of the k-th feature map in the l-th layer at position (i, j) is represented as follows.

$$z_{i,j,k}^l = w_k^l x_{i,j}^l + b_k^l \tag{2}$$

where w_k^l and b_k^l are the weight vector and bias term of the k-th convolution kernel in the l-th layer, respectively, and $x_{i,j}^l$ is the perceptual field centered at position (i, j) in the l-th layer.

This paper uses the batchnorm (BN) method [29] to normalize the data while avoiding the problem of overfitting the network. For each Mini-Batch, the training process contains m training instances, and the specific BN operation is to transform the activation value of each neuron in the hidden layer as follows.

$$BN(z_{i,j,k}^l) = \frac{z_{i,j,k}^l - E[z_{i,j,k}^l]}{\sqrt{Var[z_{i,j,k}^l]}} \tag{3}$$

where E is the mean and var is the variance. The output $a_{i,j,k}^l$ after the activation function processing can be expressed as

$$a_{i,j,k}^l = a(BN(z_{i,j,k}^l)) \tag{4}$$

C. Adaptive Averaging Pooling Layer

The addition of the pooling layer can downsample the sample features collected by the convolutional layer to further reduce the number of parameters. The adaptive averaging pooling layer can dynamically calculate the size of the kernel

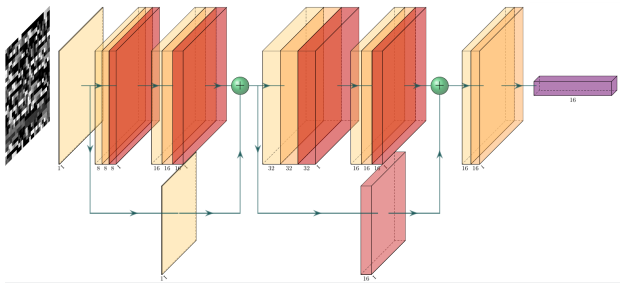


Fig. 4. Structure of the residual convolutional network model.

and the step size of each move adaptively by inputting the original size and the target size and using the average method for each operation [30]. Assuming that the padding is zero, the size of the pooling kernel can be simplified and expressed as

$$kernel\_size = \lceil \frac{output\_size}{input\_size} \rceil \tag{5}$$

D. Proposed Method

The residual network structure was first proposed by Kaiming He et al [27]. The method won the championship of image classification in the visual recognition competition. However, to obtain higher accuracy, the number of stacked residual blocks usually reaches tens to hundreds, which also increases the time complexity of training. To balance time and efficiency, the network structure is improved in this paper. The specific architecture of the residual convolutional network structure proposed in this paper is shown in Fig. 4. The model first reads $batch\_size = n$ grayscale traffic images of size $I = 28*28*1$ from the IDX file. Then it goes through two residual layers, each residual layer includes a convolutional layer, a batch normalization layer and an activation function layer. To make the network training easier and the network generalization stronger, this paper uses batch norm and relu functions to optimize the network results and then transforms them into n_classes of feature maps through a convolutional layer and a normalization layer, and finally outputs each class of features through an adaptive average pooling layer AdaptiveAvgPool2d outputs the feature values of each class. The specific network structure and parameters settings are shown in Table II.

TABLE II
PARAMETER SETTINGS OF THE CLASSIFICATION MODEL.

Layer	Parameter Setting	Output Shape	Param
Conv2d	in=1, out=8, kernel=3, stride=2, padding=15	[-1, 8, 28, 28]	80
BatchNorm2d	features=8, eps=1e-05, momentum=0.1	[-1, 8, 28, 28]	16
Conv2d	in=8,out=16, kernel=3, stride=2, padding=15	[-1, 16, 28, 28]	1,168
BatchNorm2d	features=16, eps=1e-05, momentum=0.1	[-1, 16, 28, 28]	32
ResBlock	/	[-1, 16, 28, 28]	0
Conv2d	in=16,out=16, kernel=3, stride=2, padding=15	[-1, 32, 28, 28]	4,640
BatchNorm2d	features=32, eps=1e-05, momentum=0.1	[-1, 32, 28, 28]	64
Conv2d	in=32, out=16, kernel=3, stride=2, padding=15	[-1, 16, 28, 28]	4,624
BatchNorm2d	features=16, eps=1e-05, momentum=0.1	[-1, 16, 28, 28]	32
ResBlock	/	[-1, 16, 28, 28]	0
Conv2d	in=16,out=11, kernel=3, stride=2, padding=15	[-1, 11, 28, 28]	1,595
BatchNorm2d	features=11, eps=1e-05, momentum=0.1	[-1, 11, 28, 28]	22
AdaptiveAvgPool2d	out_size=1	[-1, 11, 1, 1]	0

VI. EXPERIMENTAL SETTING AND RESULTS

A. Experimental Setting and Evaluation Metrics

The experimental environment for this paper is as follows. NVIDIA-SMI 460.32.03 GPU as the accelerator, Python 3.7.13 and Pytorch 1.12.0 as the experimental framework, experimental data randomly selected one-tenth of the data as test data, and the rest as training data. The minimum batch size read during training is 128, the loss function is selected as the cross-entropy loss function, Pytorch's built-in SGD is used as the optimizer and and setting the parameter momentum to further avoid overfitting, the learning rate is 0.01, the momentum parameter is set to 0.9, and the training time is 150 epochs. The pseudo-code of the training process is shown in algorithm 2.

The models are evaluated in terms of accuracy, recall, precision, and F1-score commonly used in classification tasks.

Algorithm 2 Residual Convolutional Network Training Process (RCNTraining)

Input: $\{(x_i, y_i)\}_{i=1}^{N_{data}}$, a dataset of sequence pairs.
Hyperparameters: $N_{epochs}, N_{batch\_size}$
Output: z_i, the training result of x_i.
for $n = 1$ to N_{epochs} **do**
  for $i = 1$ to N_{data} **do**
    $(x_i, y_i) \leftarrow$ the i-th training sample
    $z_i \leftarrow$ RCNTraining(x_i)
    $loss \leftarrow$ CrossEntropyLoss(z_i, y_i)
    Back propagation calculate and update the gradient
    $i \leftarrow i + N_{batch\_size}$
  end for
end for

B. Experimental Results

Table III shows the precision, recall, and F1 score of each category in the dataset CIC_TF on the residual convolutional network model proposed in this paper, and finally, the results of macro-average and weighted average are calculated. The experimental results show that the overall classification accuracy of the model can reach 99.93%, the macro-average accuracy, recall, and F1 score can reach 99.98%, 100% and 100%, respectively, and the weighted average accuracy, recall and F1 score can reach 100%, 100% and 100%, respectively. In a word, no matter the small sample size of ftps_down, skype_audio traffic, or the large sample size of FTP traffic, it can achieve high accuracy. In addition, the model can also accurately classify different service flows of the same application such as session and video traffic in facebook, and similar service flows of different applications, such as facebook's video flow, hangouts' video flow, and skype video flow, the model can also effectively extract features and classified. In conclusion, the residual convolutional network model proposed in this paper has a good performance in classifying similar fine-grained traffic data.

To better observe the specific types that each type of traffic is classified by the model when it is classified, we also calculate the confusion matrix of the model classification results, as shown in Fig. 5, where the right side indicates the heat value of the classification accuracy, the higher the accuracy, the higher the classification accuracy and the darker

TABLE III
EVALUATION RESULTS OF THE RESIDUAL CONVOLUTIONAL NETWORK MODEL ON THE CIC_TF DATASET.

Classes	Precision	Recall	F1-score	Samples
email	1.00	1.00	1.00	342
facebook_chat	0.97	1.00	0.98	31
facebook_video	1.00	1.00	1.00	16
FTP	1.00	1.00	1.00	6000
ftps_down	1.00	1.00	1.00	7
ftps_up	1.00	1.00	1.00	13
Gmail	1.00	1.00	1.00	1638
hangouts_audio	1.00	1.00	1.00	88
hangout_chat	1.00	0.97	0.99	34
Outlook	1.00	1.00	1.00	1413
skype_audio	1.00	1.00	1.00	9
skype_chat	1.00	1.00	1.00	428
vimeo	1.00	1.00	1.00	12
spotify	1.00	1.00	1.00	14
weibo	1.00	1.00	1.00	4113
youtube	1.00	1.00	1.00	24
accuracy	/	/	**0.99**	14182
macro avg	0.99	1.00	1.00	14182
weighted avg	1.00	1.00	1.00	14182

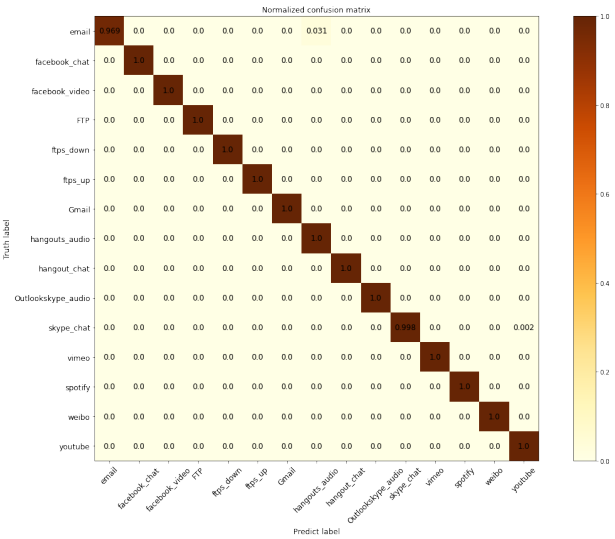


Fig. 5. Confusion matrix of the residual convolutional network model on the CIC_TF dataset.

the color, the abscissa represents the model prediction result, and the ordinate indicates the real label. It is obvious from the figure that the value on the diagonal line has the darkest color, which also shows that the model can distinguish various types of traffic very well, and it can also be seen to which categories various types of traffic will be classified by the model in the classification, for example, 0.031% of samples in email traffic are classified as hangouts_audio traffic.

C. Comparative Experiments

1) Dataset: To verify the effectiveness of the model on different traffic datasets, we conducted experiments on the USTC-TK traffic dataset [8] using the same hyperparameters as the experiments above, which contains ten types of traffic, including eight common different application traffic, part of this traffic is malware traffic collected from real network environments, and the other part is normal traffic collected using IXIA BPS traffic simulation devices. The experimental results are shown in Table IV. Since the ten classes of traffic in the dataset belong to different applications and are clearly distinguishable, our model can achieve almost 100% classification accuracy.

TABLE IV
EVALUATION RESULTS OF THE RESIDUAL CONVOLUTIONAL NETWORK MODEL ON THE USTC-TK DATASET.

Classes	Precision	Recall	F1-score	Samples
BitTorrent	1.00	1.00	1.00	752
Facetime	1.00	1.00	1.00	600
FTP	1.00	1.00	1.00	300
Gmail	1.00	1.00	1.00	863
MySQL	1.00	1.00	1.00	200
Outlook	1.00	1.00	1.00	752
Skype	1.00	1.00	1.00	632
SMB	1.00	1.00	1.00	266
Weibo	1.00	1.00	1.00	495
WorldOfWarcraft	1.00	1.00	1.00	788
accuracy	/	/	**1.00**	22148
macro avg	1.00	1.00	1.00	22148
weighted avg	1.00	1.00	1.00	22148

TABLE V
PARAMETER SETTINGS FOR COMPARISON MODELS.

Methods	Parameters
Proposed	epochs=150, batch_size=128, block_num=2, lr=1e-2, momentum=0.9
ResNet-18	epochs=200, batch_size=128, block_num=4, lr=5e-3
1D-CNN	epochs=100, batch_size=128, padding_size = 15, lr=1e-2
LSTM	epoch = 50, batch_size=128, hidden_size=128, lr=1e-2
CNN-LSTM	epochs=50,batch_size=128, hidden_size=128,seq_length=28, num_directions=1,lr=1e-2

2) Other Advanced Methods: To verify the advancedness of the model proposed in this paper for fine-grained traffic classification tasks, we compared the performance of the classical residual network model ResNet-18 [27] on the CIC_TF dataset horizontally, while comparing other deep learning methods such as 1D-CNN [8], LSTM [20] and CNN-LSTM [31] vertically, with the specific parameter settings of the model shown in Table V (subject to the best results during the experiments). The accuracy, precision, recall, and F1 score results of the experiments are shown in Table VI and Fig. 6. From Table VI, it can be seen that the current advanced deep learning methods can achieve more than 80% classification accuracy, specifically, the overall classification accuracy of the ordinary ResNet-18,1D-CNN, LSTM and CNN-LSTM can reach 95%, 86%, 92% and 94%, respectively, The overall accuracy of the improved residual convolution model proposed in this paper can reach about 99.93%, which can be improved by 4%, 13%, 7% and 2% respectively compared with the other methods. From Fig. 6, it can be more intuitively seen that the model proposed in this paper significantly outperforms other types of deep learning methods.

3) Hyperparameters: We conducted a series of experiments on the dataset CIC_TF to explore the effect of hyperparameter settings on model accuracy. The main hyperparameters include training period $epoch$, number of residual blocks $block\_num$, learning rate lr, and batch size $batch\_size$. For each exper-

TABLE VI
EXPERIMENTAL RESULTS OF VARIOUS COMPARISON METHODS.

Methods	Accuracy	Precision	Recall	F1-score
Proposed	**0.99**	**0.99**	**1.00**	**1.00**
ResNet-18	0.95	0.89	0.87	0.90
1D-CNN	0.86	0.85	0.83	0.84
LSTM	0.92	0.91	0.88	0.90
CNN-LSTM	0.94	0.93	0.92	0.96

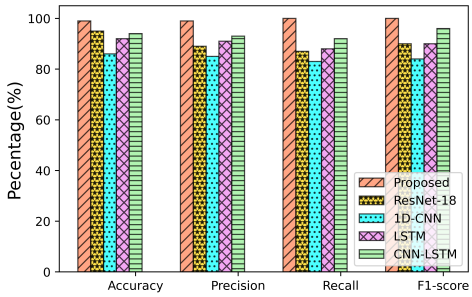
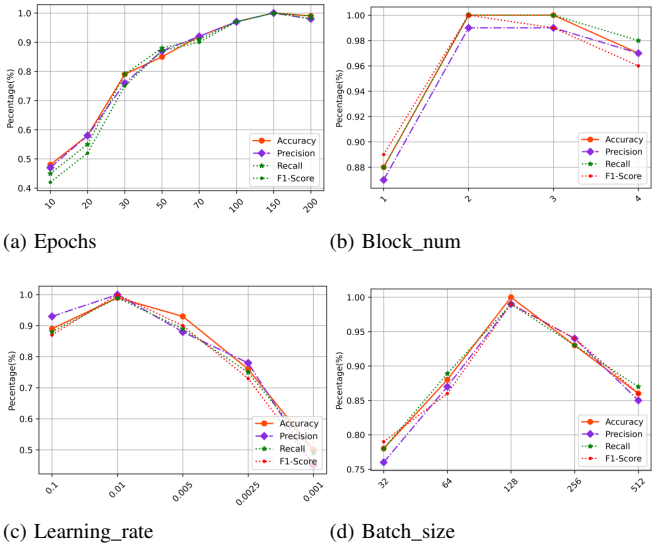


Fig. 6. Bar chart of experimental results for various comparison methods.



(a) Epochs (b) Block_num

(c) Learning_rate (d) Batch_size

Fig. 7. Graph of experimental results for each hyperparameter.

iment, only one parameter value is changed, and the default values are used for the other parameters. As shown in Fig. 7a, the model accuracy increases as the epoch increases and reaches the optimal accuracy about 100% at the epoch of 150, but when the epoch continues to increase, the learning ability of the model begins to decline. Fig. 7b shows the effect of the number of residual blocks on the model effect. When the number of residual blocks is only one, the model does not learn all the features well, and when the number of residual blocks increases to 2, the model effect has a significant improvement, but the model may start to degrade as the number of residual blocks increases to 3, 4 or even more. Fig. 7c shows the relationship between the learning rate and the performance of the model. The training process updates the size of the learning rate by 0.5 times, and it is evident that for our model in this paper, the learning rate of 0.01 works best, and as the learning rate is further reduced, the model performance drops catastrophically. Fig. 7d explores the effect of batch size on the classification effect of the model. The relevant practice has proved that the batch size is usually and the effect is the best. After further experiments, it is found that as the batch continues to increase, the model effect is significantly improved, and at 128 reaching the optimum, but when the batch continues to increase, the model cannot learn more subtle features so that the accuracy of the model begins to decline.

In summary, the setting of the model hyperparameters plays a pivotal role in the classification results, and the values of the parameters are not linearly correlated with the results, and we need to find the optimal values in the experiments to achieve the optimal results.

VII. CONCLUSIONES

Based on the systematic summary of related papers on traffic classification methods, this paper proposes a fine-grained traffic classification model based on residual convolutional networks in SDN. We design a fine-grained traffic dataset and a preprocessing method based on representation learning, and then propose a classification method based on an improved residual convolutional network to achieve high accuracy classification effect. What's more, the method proposed in this paper improves the traditional convolutional neural networks and effectively solves the gradient disappearance problem that may occur when traditional deep learning methods identify fine-grained traffic.

However, there are still some limitations in this paper, the main goal of this paper is high precision, so we didn't analyze the time complexity. In practical applications, both time and accuracy are important metrics. So in future work, we will further consider traffic classification from real-time or online, identification of new types of traffic and optimization of model parameters.

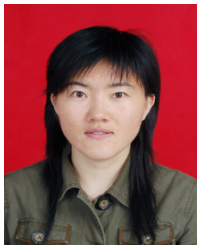
ACKNOWLEDGMENTS

This work was supported by Funds for high-level scientific research and innovation platform of Chongqing Municipal Education Commission (No. D61992019004).

REFERENCES

[1] P. Wang, F. Ye, X. Chen, and Y. Qian, "Datanet: Deep learning based encrypted network traffic classification in SDN home gateway," *IEEE Access*, vol. 6, pp. 55380–55391, 2018.
[2] K. Lin, X. Xu, and H. Gao, "TSCRNN: A novel classification scheme of encrypted traffic based on flow spatiotemporal features for efficient management of iiot," *Comput. Networks*, vol. 190, p. 107974, 2021.
[3] J. Xie, F. R. Yu, T. Huang, R. Xie, J. Liu, C. Wang, and Y. Liu, "A survey of machine learning techniques applied to software defined networking (SDN): research issues and challenges," *IEEE Commun. Surv. Tutorials*, vol. 21, no. 1, pp. 393–430, 2019.
[4] P. Wang, X. Chen, F. Ye, and Z. Sun, "A survey of techniques for mobile service encrypted traffic classification using deep learning," *IEEE Access*, vol. 7, pp. 54024–54033, 2019.
[5] S. Chang, M. Tsai, M. Lee, and J. Ho, "Optimal qoe scheduling in MPEG-DASH video streaming," *Int. J. Interact. Multim. Artif. Intell.*, vol. 6, no. 7, p. 71, 2021.

[6] R. F. Mansour, C. Soto, R. Soto-Díaz, J. Escorcia-Gutierrez, D. Gupta, and A. Khanna, "Design of integrated artificial intelligence techniques for video surveillance on iot enabled wireless multimedia sensor networks," *Int. J. Interact. Multim. Artif. Intell.*, vol. 7, no. 5, p. 14, 2022.
[7] J. Yan and Y. Jing, "A survey of traffic classification in software defined networks," in *2018 1st IEEE International Conference on Hot Information-Centric Networking (HotICN)*, 2018.
[8] W. Wang, M. Zhu, J. Wang, X. Zeng, and Z. Yang, "End-to-end encrypted traffic classification with one-dimensional convolution neural networks," in *2017 IEEE International Conference on Intelligence and Security Informatics, ISI 2017, Beijing, China, July 22-24, 2017*, pp. 43–48, IEEE, 2017.
[9] F. Pacheco, E. Exposito, M. Gineste, C. Baudoin, and J. Aguilar, "Towards the deployment of machine learning solutions in network traffic classification: A systematic survey," *IEEE Commun. Surv. Tutorials*, vol. 21, no. 2, pp. 1988–2014, 2019.
[10] S. Rezaei and X. Liu, "Deep learning for encrypted traffic classification: An overview," *IEEE Commun. Mag.*, vol. 57, no. 5, pp. 76–81, 2019.
[11] M. Finsterbusch, C. Richter, E. Rocha, J. Muller, and K. Hanssgen, "A survey of payload-based traffic classification approaches," *IEEE Commun. Surv. Tutorials*, vol. 16, no. 2, pp. 1135–1156, 2014.
[12] F. Tang, Z. M. Fadlullah, B. Mao, and N. Kato, "An intelligent traffic load prediction-based adaptive channel assignment algorithm in sdn-iot: A deep learning approach," *IEEE Internet Things J.*, vol. 5, no. 6, pp. 5141–5154, 2018.
[13] S. Zhao, Y. Xiao, Y. Ning, Y. Zhou, and D. Zhang, "An optimized k-means clustering for improving accuracy in traffic classification," *Wirel. Pers. Commun.*, vol. 120, no. 1, pp. 81–93, 2021.
[14] O. Aouedi, K. Piamrat, and D. Bagadthey, "A semi-supervised stacked autoencoder approach for network traffic classification," in *28th IEEE International Conference on Network Protocols, ICNP 2020, Madrid, Spain, October 13-16, 2020*, pp. 1–6, IEEE, 2020.
[15] W. Wang, M. Zhu, X. Zeng, X. Ye, and Y. Sheng, "Malware traffic classification using convolutional neural network for representation learning," in *2017 International Conference on Information Networking, ICOIN 2017, Da Nang, Vietnam, January 11-13, 2017*, pp. 712–717, IEEE, 2017.
[16] S. Rezaei and X. Liu, "Multitask learning for network traffic classification," in *29th International Conference on Computer Communications and Networks, ICCCN 2020, Honolulu, HI, USA, August 3-6, 2020*, pp. 1–9, IEEE, 2020.
[17] X. Xie and J. Wu, "Real-time flow identification based on neural network and openflow over sdn," in *2018 10th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC)*, vol. 02, pp. 12–16, 2018.
[18] T. Yang, S. Vural, P. Qian, Y. Rahulan, N. Wang, and R. Tafazolli, "Achieving robust performance for traffic classification using ensemble learning in SDN networks," in *ICC 2021 - IEEE International Conference on Communications, Montreal, QC, Canada, June 14-23, 2021*, pp. 1–6, IEEE, 2021.
[19] M. Lotfollahi, M. J. Siavoshani, R. S. H. Zade, and M. Saberian, "Deep packet: a novel approach for encrypted traffic classification using deep learning," *Soft Comput.*, vol. 24, no. 3, pp. 1999–2012, 2020.
[20] Y. Zeng, H. Gu, W. Wei, and Y. Guo, "$deep-full-range$: A deep learning based network encrypted traffic classification and intrusion detection framework," *IEEE Access*, vol. 7, pp. 45182–45190, 2019.
[21] P. Wang, Z. Wang, F. Ye, and X. Chen, "Bytesgan: A semi-supervised generative adversarial network for encrypted traffic classification in SDN edge gateway," *Comput. Networks*, vol. 200, p. 108535, 2021.
[22] J. Lan, X. Liu, B. Li, Y. Li, and T. Geng, "Darknetsec: A novel self-attentive deep learning method for darknet traffic classification and application identification," *Comput. Secur.*, vol. 116, p. 102663, 2022.
[23] Z. Wu, Y. Dong, X. Qiu, and J. Jin, "Online multimedia traffic classification from the qos perspective using deep learning," *Comput. Networks*, vol. 204, p. 108716, 2022.
[24] J. Cheng, Y. Wu, Y. E, J. You, T. Li, H. Li, and J. Ge, "MATEC: A lightweight neural network for online encrypted traffic classification," *Comput. Networks*, vol. 199, p. 108472, 2021.
[25] "Experts in network security monitoring and network forensics." https://www.netresec.com/index.ashx?page=SplitCap. Accessed 2022.
[26] "Canadian institute for cybersecurity, datasets." https://www.unb.ca/cic/datasets/index.html. [online], Accessed 2022.
[27] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pp. 770–778, IEEE Computer Society, 2016.
[28] J. Gu, Z. Wang, J. Kuen, L. Ma, A. Shahroudy, B. Shuai, T. Liu, X. Wang, G. Wang, J. Cai, and T. Chen, "Recent advances in convolutional neural networks," *Pattern Recognition*, vol. 77, pp. 354–377, 2018.
[29] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," in *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part IV* (B. Leibe, J. Matas, N. Sebe, and M. Welling, eds.), vol. 9908 of *Lecture Notes in Computer Science*, pp. 630–645, Springer, 2016.
[30] J. Chen, X. Wang, Z. Guo, X. Zhang, and J. Sun, "Dynamic region-aware convolution," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, pp. 8064–8073, Computer Vision Foundation / IEEE, 2021.
[31] V. Bijalwan, V. B. Semwal, G. Singh, and T. K. Mandal, "Hdl-psr: Modelling spatio-temporal features using hybrid deep learning approach for post-stroke rehabilitation," *Neural Processing Letters*, pp. 1–20, 2022.



Chang Su received PH.D. degree from the University of Liverpool, U.K. She is currently a Full Professor with the School of Computer Science and Technology, Chongqing University of Posts and Telecommunications, China. She is a member of the China standardization committee and an expert of ISO/IEC. From 2011 to 2013, she was a postdoctoral fellow in the Computer Department of Cornell University. Her research interests include wireless network communications, satellite communications, artificial intelligence and deep learning.

Yanqing Liu received bachelor's degree from Sichuan Ethnic College. She is currently a graduate student in the School of Computer Science at Chongqing University of Posts and Telecommunications, China. Her main research directions are computer networks and artificial intelligence. Her research interests include software defined networks, deep learning and image recognition.

Xianzhong Xie received PH.D. degree from the Xidian University, China, in 2000. He is currently a Full Professor with Chongqing Key Lab of Computer Network and Communication Technology, Chongqing University of Posts and Telecommunications, China. He has published two books and over 100 scientific papers in international journals and conferences. His research interests include cognitive radio network, interference cancellation and MIMO technology, green communication network.