A novel network flow feature scaling method based on cloud-edge collaboration

Zeyi Li
School of Computer Science
Nanjing University of Posts and Telecommunications
Nanjing, China
0009-0006-5990-1825

Ze Zhang
School of Internet of Things
Nanjing University of Posts and Telecommunications
Nanjing, China
1022072003@njupt.edu.cn

Mengyi Fu
School of Internet of Things
Nanjing University of Posts and Telecommunications
Nanjing, China
1022041101@njupt.edu.cn

Pan Wang
School of Modern Posts
Nanjing University of Posts and Telecommunications
Nanjing, China
wangpan@njupt.edu.cn

***Abstract*—The AI-based network traffic classification technology has played a vital role in Zero-touch network and Service Management (ZSM). Network Traffic Features are the critical input for AI-based network traffic classification. However, unlike images and natural language traffic, features have high dispersion and weak stability. On the one hand, part of the key numerical flow features have a wide range of values. By scaling the features, the model can be better trained, which allows the classifier to make more accurate predictions. On the other hand, edge networks cannot scale features as well as the training set in the cloud. This situation affects the model's performance on edge devices. This paper proposes a novel processing method of flow features for the AI-based network traffic classification model using cloud-edge collaboration. The method leverages the strong computational power of cloud computing to scale the flow features and transmit parameters derived from feature scaling to edge networks. The edge network processes the real-time flow through parameters, significantly enhancing the devices' accuracy. To validate its effectiveness, we conducted four experiments, comparing models trained with and without feature scaling. The experimental results demonstrate that the model trained with feature scaling outperforms the model trained without scaling.**

***Index Terms*—Feature Engineering, Feature Scaling, Traffic Classification, Deep Learning, Zero-touch network**

I. Introduction

The current highly flexible architectures of Beyond 5G (B5G) and 6G have given rise to network-slicing technologies, such as SDN. While enabling cross-domain collaboration and agile implementation of networks, these technologies also bring unprecedented complexity to network management. Closed-loop automated management is inevitable, and traditional management models are challenging to meet. It is fashionable for Artificial Intelligence (AI) to make network services gradually move towards intelligence. Against this backdrop, the intelligent network management system becomes an inevitable technology servicing Next-Generation Networks (NGN). Since European Telecommunications Standards Institute (ETSI) defined "Zero-touch network and Service Management" (ZSM), NGNs request network services to be able to make autonomous decisions without human intervention [4]. The network traffic classification technology, which provides relevant information and decision suggestions for information flow control, becomes a hub part of the automated decision.

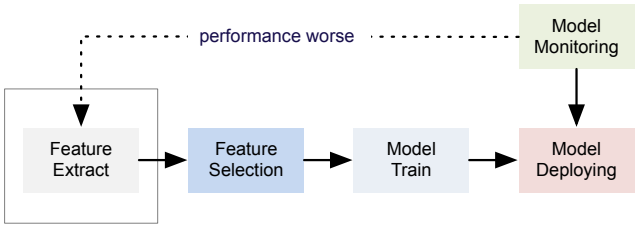


Fig. 1: The process of network traffic classification based on AI.

With the emergence of deep learning and high-performance computing technologies, the ability of Deep Learning (DL) to automatically extract features brings a new dimension of enhancement to Traffic Classification(TC). As shown in Fig.1, DL-TC is usually divided into the following steps [10]. (1) Feature Extraction (FE); (2)Feature Selection; (3) Model Train; (4) Model Deploy; (5) Model Monitor. In our context, FE plays a prerequisite role in the five steps of the DL-TC. At present, the development of FE on the DL-TC is divided into two main phases: The first phase is the packet-based FE technique, where a raw packet is converted to a matrix in a fixed length as input to the model. This FE approach ignores information about time delays, which is sensitive for different services [11]. Thus, this FE method still has shortcomings. The subsequent second stage is the flow-based FE technique, combined with deep learning techniques to find the intrinsic patterns of the traffic characteristics of different services. The flow-based FE method takes the time sensitivity into account while significantly reducing the complexity of the operation.

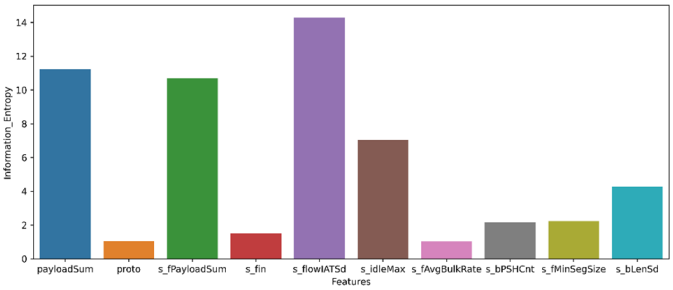


Fig. 2: Information entropy of features.

TABLE I: Standard deviation of features.

Feature	s_fTcpWindown	s_bLenMin	...	s_flowIATSd
Standard Deviation	28337.9425	91.0067	...	464488621.8755

These advantages are precisely what the packet-based FE techniques lack. Hence, the flow-based FE methods have seen a new wave and derived a series of TC methods achieving better performance.

However, flow features generated by flow-based FE differ from traditional images and languages. The image's upper and lower bound, consisting of values between 0 and 255, is defined. Natural Language Processing (NLP) directly feeds vectors generated by Word2vec into the model. To the degree that network traffic is not similar to images, text, or speech. From Fig.2, the information entropy of different features of network traffic varies greatly. 'payloadSum' is more unstable than proto and 'fin'. From Table.I, the degree of numerical dispersion varies between the features. The features that FE extracts do not have fixed boundaries and have outliers that make model convergence difficult. In summary, performing feature scaling for flow features is necessary before training the model. Typically, feature scaling consists of four types: min-max, z-score, max-abs, and robust. Presently, few papers discuss flow feature processing methods in real scenarios.

Edge computing, which transfers heavy computing loads to the cloud, allows the cloud to perform complex computations and securely compute the results at the edge. Inspired by cloud-edge collaboration, this paper proposes a feature scaling method for an AI-based network traffic classification model that collaborates with the cloud edge. The method scales the flow features in the cloud. The parameters are stored and sent to the edge. The edge scales flow features according to parameters in real-time and sends them to the classifier for results. The main contributions of this paper are as follows:

1) This paper proposes a cloud-edge collaborative approach at the edge to scale real-time flow features and obtain classification results.
2) This paper conducts five experiments with and without feature scaling and concludes that feature scaling is essential in real network scenarios.

II. Related Works

A. Deep Learning based Traffic Classification

DL, also known as deep structured learning or hierarchical learning, is achieved by learning data representation. Compared to traditional machine learning algorithms, DL can automatically extract features without human intervention, which makes it ideal for traffic classification.

There have been many scholars who have studied how to apply the DL to the TC. Dicks et al. [6] explore the computational efficiency and accuracy of Long Short-Term Memory (LSTM) and Multi-Layer Perceptron (MLP) deep learning models for packet-based classification of traffic in a community network. Aceto et al. [2] study distiller: encrypted traffic classification via multimodal multitask deep learning. To this end, a novel multimodal multitask deep learning approach for traffic classification is proposed, leading to the Distiller classifier. Guan et al. [7] propose a traffic classification method based on deep transfer learning for 5G IoT scenarios with scarce labeled data and limited computing capability and train the classification model by weight transferring and neural network fine-tuning. Pang et al. [13] present a Multimodal Classification method named MTCM to exploit the context for the classification task systematically. However, the following limitations remain: (1) the traffic representation is generated from raw packet bytes, resulting in the absence of critical information; (2) the model structure of directly applying deep learning algorithms does not take traffic features into account. The preprocessing phase of the proposed end-to-end encrypted traffic classification method by Wang et al. [17] uses the USTC-TL2016 tool to generate byte data of raw traffic in IDX3 files for input data of CNN model, which is different from a traditional divide-and-conquer method.

However, many articles [1] [15] focus on something other than feature scaling in the TC domain.

B. Feature Scaling based Traffic Classification

Since the data generated by the flow-based FE method has many outliers, many articles have also used feature scaling prior to experimentation.

Wang [16] uses the Min-Max Normalization method for data preprocessing to deal with anomalous data. The article discusses the challenges of capturing large labeled datasets and proposes a GAN-based approach that can perform well with a small number of labeled traffic samples. The experimental results show that the proposed method outperforms supervised learning methods like CNN. Normalization is a common and essential preprocessing step in traffic classification. The paper [14] implements five data normalization methods: Roubstand Scaler, Min-Max, Standard Scaler, and L2 Normalization. These methods apply to numerical and nominal features of NSL-KDD and UNSW NB-15 datasets. The paper [8] proposes a data preprocessing method with undersampling and embedded feature selection to mitigate the flow samples' imbalance and extract the input stream's dominant features, where the normalization is done using global min-max normalization. Chen et al. [5] developed the SMC (sequential

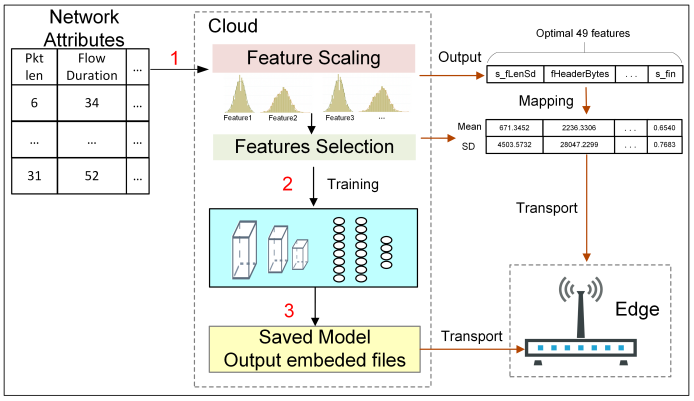


Fig. 3: Workflow of TC Methodology based Cloud-Edge Collaboration.

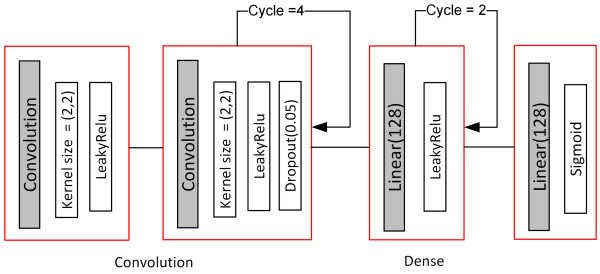


Fig. 4: the Parameters of the convolutional layer.

message Characterization) system, which allows online traffic classification using sequential size information of several message segments, with segmented linear normalization of the message sizes during data preprocessing. Similarly, the paper [3] nonlinearly normalizes the message size into several scale ranges, e.g., 0-100, 100-500, to narrow down the dynamic range of sample values. The paper [9] proposes a framework for preprocessing and analyzing log files for periodic or real-time intrusion detection. The framework first uses n-gram analysis to convert HTTP queries into numerical matrices, which are log-normalized, and the feature dimensions are reduced by principal component analysis.

However, these articles have not discussed feature scaling for real network scenarios. The next chapter proposes how feature scaling in TC methods can be used in real scenarios.

III. The Proposed Method

A. The Workflow of TC

This subsection describes the flow of AI-based TC models from the cloud to the edge. From Fig.3, the feature processing module extracts the network flow attribute features. The feature extraction module first processes raw packets to obtain various attributes of flows, including Forward Inter Arrival Time, Flow Duration, and so on.

As stated in Chapter 1, outliers in the feature attributes affect the model's accuracy. Therefore, the Feature Scaling module in the cloud first learns the distribution of features and passes the distribution parameters to edge devices. The feature scaling module generates new flow features. The AI model trains new flow features and automates the conversion to c-files.

At the edge router, flow features are extracted from real-time packets. New flow features are reconstructed from flow features based on parameters transmitted from the cloud. Finally, the flow features are fed into the classifier to obtain classification results.

B. Cloud-Edge Collaboration-based TC methodology design and training

To make it easier to describe these methods, the following network traffic terms are defined in this chapter:

1) Packet: A packet is a basic unit in flow. Set $P = \{p_1, p_2, \ldots, p_{i-1}, p_i\}, 0 \leq i \leq M, M$ is the number of packets.
2) Flow: Source address, destination address, source port, destination port, and TCP/UDP protocol five-tuple consists of flow, which is unidirectional to differs from Session. $T = \{t_1, t_2, \cdots, t_{m+n}\}$, represents a set of flows.
3) Flow Feature: It includes packet-level features, flow-level features, and statistical features, formally defined as $F = \{f_1, f_2, \ldots, f_{i-1}, f_i\}, 0 \leq i \leq K, K$ is the number of flows. F is the flow feature vector, composed of a total of 91 feature sub-items.
4) New Flow Feature: Features are scaled to become new features, defined as FF_w.
5) The vector of FF_w is the basic unit of FAM. FAM is composed of FF_ws: $FAM = \{FF_{w1}, FF_{w2}, \ldots, FF_{wi}\}$, i is the number of flows.
6) Parameters for feature scaling: TC selects different feature scaling methods before training. The parameters in feature scaling are output to the edge router. For example, the parameters of Standardization method are the mean and standard deviation of each feature. $\Theta = \{\theta_1, \quad \theta_2, \ldots, \theta_n\}$. Θ is the set of all parameters. θ_i is the parameter of the i-th feature. n is the number of features.

The pseudocode is shown in Algorithm 1. This model adaptively adjusts the number of convolutional layers (CovL) according to the number of features. In this experiment, we choose 49 features for training. 49 features is a quadruple convolution. After input to the model, f is reshaped. Multiple groups of FF_w makes up the FAM. FAM is input to the first convolutional layer to obtain v_1. This process is known as $v_{i+1} = Cov(v_i)$, $0 \leq i < j$, j is the number of convolutional layers. v_{j-1} flattens and inputs to the fully connected layer to obtain w. The process is defined as $w = Dense(flatten(v_{j-1}))$. w is classified by the function $sigmoid$. The parameters of the model are shown in Fig.4.

At the edge router, we classify the real-time network traffic in the following steps: 1. This method extracts f from the P; 2. This method scales the network flow features based on the parameters passed from the cloud. The process is defined as $FF_w = scale(f)$; 3. The classifier performs real-time

Algorithm 1 supervised Learning Algorithm for TC based Cloud-Edge Collaboration

Cloud Input:
Training data: F;
Training parameters: $batch\_size$, num_epochs, $learning\_rate$, $optimizer$, j is the number of CovL;
Cloud Output: TC-classifier, Parameters for feature scaling: Θ.

Initial training dataset;
Feature Scaling, $FF_w = scale(f)$;
Outputs Θ and transmit Θ to the edge router;
$FAM = batch\_size * FF_w$;
for epoch in num_epochs **do**
  Calculate the first layer of convolution, $v_1 = Cov(FAM)$;
  for i in j **do**
    Calculate the i-th convolution, $v_{i+1} = Cov(v_i)$;
  end for
  Flatten the convolutional data: $z = flatten(v_{j-1})$;
  Calculate the fully connected layer: $w = Dense(z)$;
  Classification results: $sigmoid(w)$;
  Update the weights and bias;
end for

Edge Input: p_i, Feature Extract, TC-classifier, Parameters for feature scaling:Θ
Edge Output: Classification results.

Extract features from $p_1, p_2, \ldots, p_{i-1}, p_i$ to get f. ;
Feature Scaling $FF_w = scale(f)$;
Results: classifier(FF_w)

classification based on FF_w.

IV. EXPERIMENTS

A. Data Set Description

The dataset comprises nine popular applications and background traffic from terminals collected in the campus network scenario, including Bilibili, QQ music, Honor of Kings, Teamfight Tactics, Tencent Meeting, and Game for peace. These apps cover the five popular app categories of video, music, Moba games, First Person Shooting (FPS), and Role-playing game (RPG) games. To collect the data, we used a semi-automatic web traffic generation tool. We leveraged an automated traffic generator to collect traffic from Bilibili and QQ music, with 20,014 flows from Bilibili and 37,668 flows from QQ music. For the other interactive games, we chose to collect them manually. We used pcapdroid to mark the network traffic as it occurred at the endpoint and the router, where the two were compared and filtered. The feature calculation tool calculated the flow features for each application's pcap. In addition, this experiment also statistics background traffic information, which contains information about network location services, security components, and unreachable vendors. The exact number of flows is shown in Table.II.

TABLE II: Numbers of flows.

Appname	Number
QQMusic	37668
Iqiyi	27252
Game for peace	11978
Tencent Meeting	8050
MOOC	13542
TikTok	14941
Bilibili	20014
Teamfight Tactics	9960
Honor of Kings	19834
Background	26770

B. Data Set Analysis

This self-collected dataset contains 91 features. Fig.5 shows the statistical distribution of features and feature outliers. The distribution of features was selected as the 29 most typical features. 'fTcpwindown' dispersion, considered a typical feature, is prominent. The number of samples in the part close to 0 on the left is large. The vast region on the right also has a large number of samples. Looking at the left side of Fig.5, there are also many outliers in the dataset's features. The researcher can easily see that these tremendous values affect the model's accuracy. In the next chapter, we use experiments to show that outliers affect accuracy. Thus, feature scaling becomes inevitable from the point of view of feature distribution.

C. Feature Scaling

Feature scaling improves the performance and convergence speed of machine learning algorithms by unifying or normalizing the range of values of different features to ensure a consistent scale between features. There are commonly used feature scaling methods:

1) Standardization: also known as Z-score standardization, which is done by removing the mean and scaling the variance of features so that the mean of the feature is 0 and the variance is 1.
2) Normalization: also known as Min-Max scaling, scales the range of values of a feature to a fixed interval, usually [0, 1] or [-1, 1].
3) Offset Scaling: linear scaling of features, mapping the value range of features to a non-zero interval, usually [-1, 1].
4) Robust Scaling (RS) resists the effects of outliers by using the median and interquartile range of the statistic. For each feature, the median is calculated for all samples; then the upper quartile (Q3) and lower quartile (Q1) are calculated for all samples, and then the IQR is calculated, i.e. $IQR = Q3 - Q1$. finally, the original value for each sample is subtracted from the median and divided by the IQR.

Next we will scale the data using each of these four feature scaling methods and verify the real network traffic in a real scenario.

D. Performance Evaluation

We scaled the dataset's features using Min-Max scaling, Z-score normalization, Max-abs scaling, and Robust scaling.

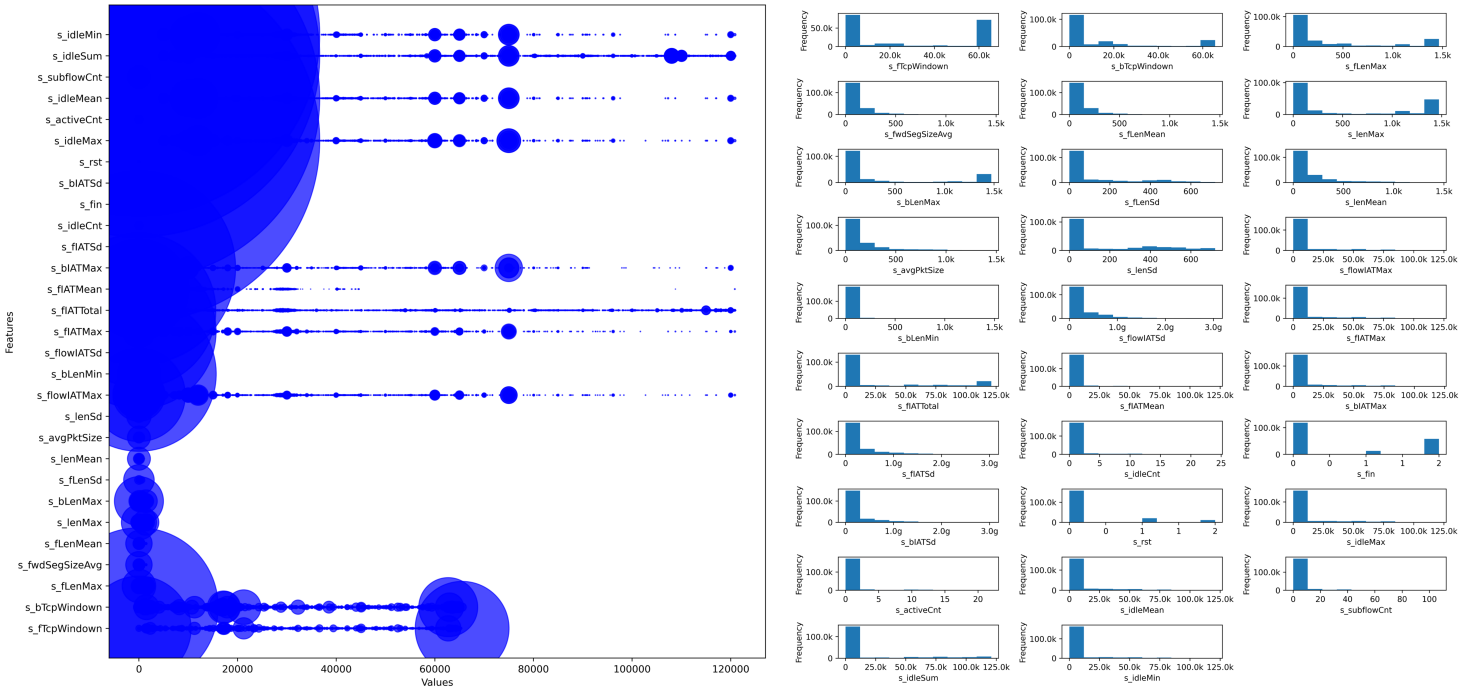


Fig. 5: Feature Values.

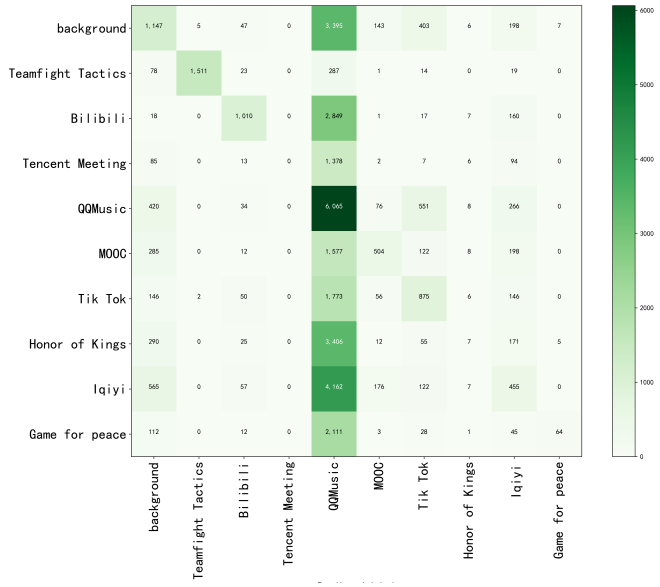


Fig. 6: Results of no scaling.

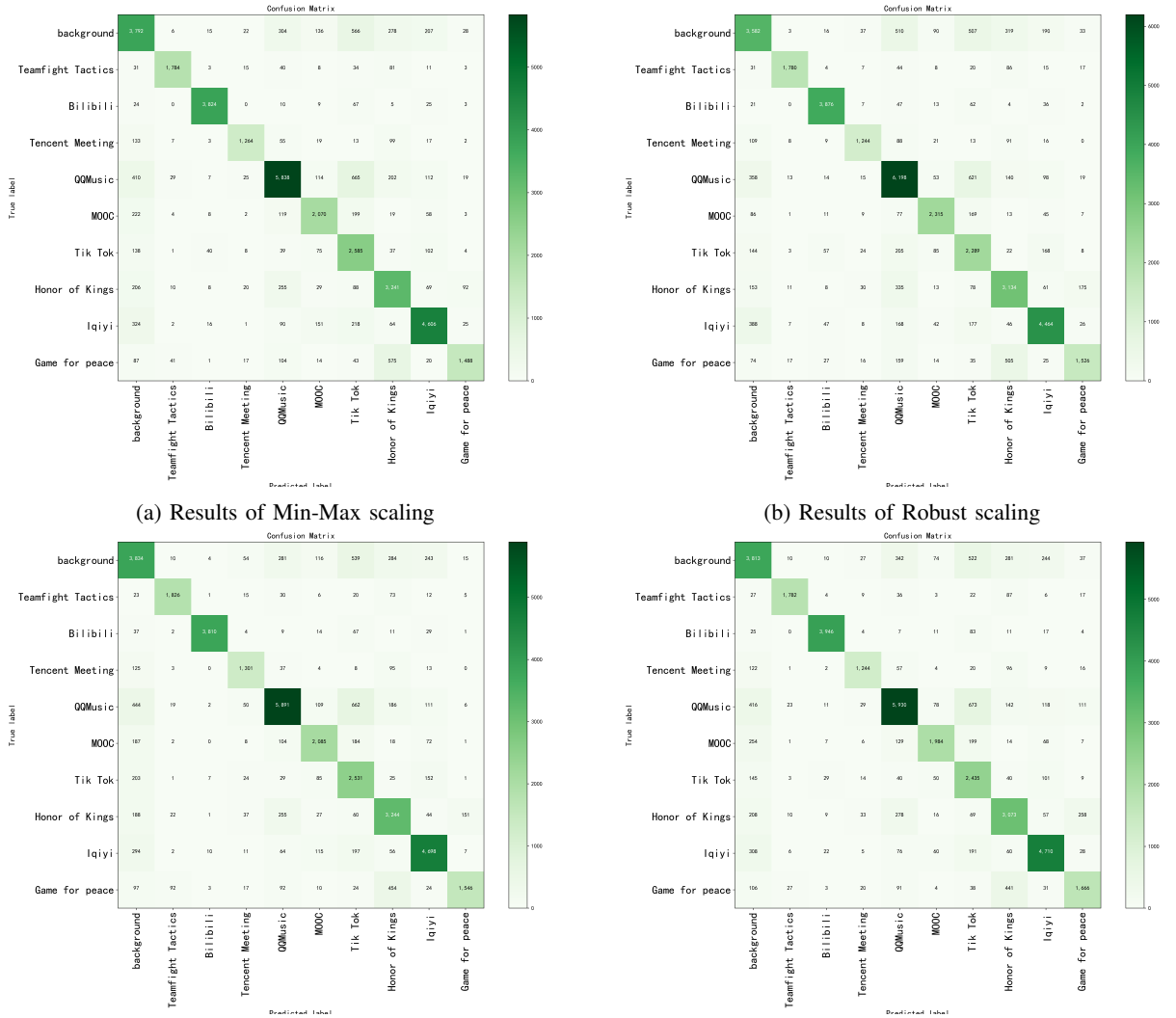
This experiment predicts the test set using trained classifiers. Experimental comparisons are performed on the original dataset without feature scaling.

The confusion matrix shows the number of correct and incorrect identifications for each type of application. From Fig.7 and Fig.6 shown in the experimental results, we can see that the prediction of the data after feature scaling is generally better than that of the data without feature scaling. When training the dataset without feature scaling, the model mostly learns the relevant features of QQ music due to the relatively large amount of QQ music in the dataset. Therefore, the model tends to predict mostly QQ Music, which leads to inaccurate classifying results. When features are scaled, the values of the features in the dataset are standardized in terms of magnitude, which reduces the impact of outliers on the model. The model is capable of learning the individual features of each application. As can be seen from Table.III, the predictions are also largely accurate.

Table.III shows that the prediction result of the data after feature scaling is around 0.8. For many vital applications, such as Bilibili and Tencent Meeting, the accuracy reaches over 90%. The prediction accuracy for each application is also between 0.7 and 0.9. The accuracy and recall of background traffic are not high. One of the reasons is that the background traffic contains logs from different systems, which makes it difficult to learn the features. Herefore, it is easy to misclassify and reduce the overall performance. The prediction score of the data without feature scaling is only around 0.3, and it is predicted to be QQ Music. This also leads to a relatively high recall of 0.79 for QQ Music, but the precision is only 0.22. Tencent Meeting and QQ Music are from the same provider, so a lot of public service traffic leads to a high false positive rate.

E. Experiment Discussion

This experiment explains the model with SHAP [12]. Our focus was on the analysis of the features of the network traffic that identify the correct Honor of Kings. As can be seen from Fig.8, the activity level of the flow is an important factor in the model. An active flow is defined as the difference in arrival



(a) Results of Min-Max scaling

(b) Results of Robust scaling

(c) Results of Z-score scaling

(d) Results of Max-abs scaling

Fig. 7: Results of different feature scaling methods.

time of two neighboring packets in any direction. This is the time interval formula: $interval = Arrival\_time\_p_{i+1} - Arrival\_time\_p_i$. We identify the flow as active if interval<0.5. Honor of Kings, as a representation of Moba games, requires frequent human-machine interaction. Therefore, the active flows of Moba games will be more frequent. Therefore, we think it is plausible that the model predicts the feature of active flows.

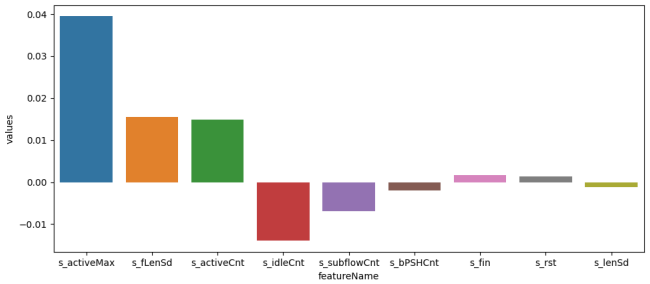


Fig. 8: Honor of Kings App Features Contribution.

V. Conclusion and Future Work

In AI-based TC, feature scaling plays a critical role. This paper proposes a network traffic features processing method for the AI-based network traffic classification model using cloud-edge collaboration. The technique leverages the mighty computing power of the cloud to perform feature extraction and feature selection on collected traffic. It calculates the distribution of the selected features in a flow. The edge, ensuring accuracy, processes the real-time flow through the operators transmitted by the cloud, which solves the problem that edge devices cannot infer accurately. In addition, we conducted four experiments with and without feature scaling. The experimental results show that the model trained with feature scaling works better than the model trained without

TABLE III: Evaluation results (including Max-abs, Min-Max, Robust, Z-score, and no feature scaling)

Feature Scaling	app name	precision	recall	f1-score
Max-abs Scaling accuracy:0.8056	Background	0.7402	0.6724	0.7047
	Teamfight Tactics	0.9586	0.9000	0.9284
	Bilibili	0.9695	0.9650	0.9672
	Tencent meeting	0.9161	0.8012	0.8548
	QQ Music	0.8444	0.8009	0.8221
	MOOC	0.8286	0.7784	0.8027
	Tik Tok	0.5703	0.8391	0.6790
	Honor of Kings	0.7338	0.7665	0.7498
	Iqiyi	0.8141	0.8762	0.8440
	Game for peace	0.8741	0.6634	0.7543
Min-Max scaling accuracy:0.8034	Background	0.6999	0.6939	0.6969
	Teamfight Tactics	0.9787	0.8748	0.9238
	Bilibili	0.9873	0.9534	0.9701
	Tencent meeting	0.9103	0.7699	0.8342
	QQ Music	0.8613	0.7858	0.8218
	MOOC	0.8714	0.7741	0.8199
	Tik Tok	0.5672	0.8441	0.6785
	Honor of Kings	0.6965	0.8077	0.7480
	Iqiyi	0.8321	0.8673	0.8493
	Game for peace	0.9023	0.6460	0.7529
Robust scaling accuracy:0.8027	Background	0.7496	0.6695	0.7073
	Teamfight Tactics	0.9741	0.8884	0.9293
	Bilibili	0.9845	0.9445	0.9641
	Tencent meeting	0.9019	0.8078	0.8523
	QQ Music	0.8327	0.8106	0.8215
	MOOC	0.7567	0.8736	0.8109
	Tik Tok	0.5477	0.8079	0.6528
	Honor of Kings	0.7217	0.7777	0.7487
	Iqiyi	0.8730	0.8397	0.8560
	Game for peace	0.8594	0.6310	0.7277
Z-score accuracy:0.8051	Background	0.7107	0.6823	0.6962
	Teamfight Tactics	0.9286	0.8968	0.9124
	Bilibili	0.9814	0.9646	0.9729
	Tencent meeting	0.9099	0.7854	0.8431
	QQ Music	0.8378	0.8071	0.8222
	MOOC	0.7673	0.7715	0.7694
	Tik Tok	0.5872	0.8424	0.6920
	Honor of Kings	0.7571	0.7620	0.7596
	Iqiyi	0.8514	0.8479	0.8497
	Game for peace	0.8583	0.6993	0.7707
No feature scaling accuracy:0.3037	Background	0.3637	0.2069	0.2637
	Teamfight Tactics	0.9147	0.8101	0.8592
	Bilibili	0.8452	0.2432	0.3777
	Tencent meeting	0.0000	0.0000	0.0000
	QQ Music	0.2289	0.7913	0.3550
	MOOC	0.5366	0.1490	0.2332
	Tik Tok	0.3547	0.2548	0.2966
	Honor of Kings	0.1074	0.0104	0.0190
	Iqiyi	0.2401	0.1212	0.1611
	Game for peace	0.8548	0.0227	0.0443

scaling.

However, the accuracy of the traffic classification method proposed in this paper for the recognition of applications still needs to be improved. Also, the experiments do not measure the recognition efficiency of the classifier on edge devices. In the future, our work will continue to test this TC method in real scenarios and further improve the accuracy of the classifier.

ACKNOWLEDGMENT

The paper is sponsored by National Natural Science Fundation (General Program) Grant No.61972211, China and University Industry Academy Research Innovation Fund No.2021FNA02006, China.

REFERENCES

[1] Giuseppe Aceto, Domenico Ciuonzo, Antonio Montieri, and Antonio Pescapé. Mobile encrypted traffic classification using deep learning: Experimental evaluation, lessons learned, and challenges. *IEEE Transactions on Network and Service Management*, 16(2):445–458, 2019.

[2] Giuseppe Aceto, Domenico Ciuonzo, Antonio Montieri, and Antonio Pescapé. Distiller: Encrypted traffic classification via multimodal multitask deep learning. *Journal of Network and Computer Applications*, 183-184:102985, 2021.

[3] Afeez Ajani Afuwape, Ying Xu, Joseph Henry Anajemba, and Gautam Srivastava. Performance evaluation of secured network traffic classification using a machine learning approach. *Computer Standards & Interfaces*, 78:103545, 2021.

[4] Chafika Benzaid and Tarik Taleb. Ai-driven zero touch network and service management in 5g and beyond: Challenges and research directions. *IEEE Network*, 34(2):186–194, 2020.

[5] Wenxiong Chen, Feng Lyu, Fan Wu, Peng Yang, Guangtao Xue, and Minglu Li. Sequential message characterization for early classification of encrypted internet traffic. *IEEE Transactions on Vehicular Technology*, 70(4):3746–3760, 2021.

[6] Matthew Dicks and Josiah Chavula. Deep learning traffic classification in resource-constrained community networks. In *2021 IEEE AFRICON*, pages 1–7, 2021.

[7] Jianfeng Guan, Junxian Cai, Haozhe Bai, and Ilsun You. Deep transfer learning-based network traffic classification for scarce dataset in 5g iot systems. *International Journal of Machine Learning and Cybernetics*, 12(11):3351–3365, 2021.

[8] Yanpei Hua. An efficient traffic classification scheme using embedded feature selection and lightgbm. In *2020 Information Communication Technologies Conference (ICTC)*, pages 125–130, 2020.

[9] Antti Juvonen and Tuomo Sipola. Adaptive framework for network traffic classification using dimensionality reduction and clustering. In *2012 IV International Congress on Ultra Modern Telecommunications and Control Systems*, pages 274–279, 2012.

[10] Zeyi LI, Pan WANG, and Zixuan WANG. Flowgananomaly: Flow-based anomaly network intrusion detection with adversarial learning. *Chinese Journal of Electronics*, 33:1–15, 2022.

[11] Manuel Lopez-Martin, Belen Carro, Antonio Sanchez-Esguevillas, and Jaime Lloret. Network traffic classifier with convolutional and recurrent neural networks for internet of things. *IEEE access*, 5:18042–18050, 2017.

[12] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. *Advances in neural information processing systems*, 30, 2017.

[13] Bo Pang, Yongquan Fu, Siyuan Ren, Siqi Shen, Ye Wang, Qing Liao, and Yan Jia. A multi-modal approach for context-aware network traffic classification. In *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5, 2023.

[14] Murtaza Ahmed Siddiqi and Wooguil Pak. An agile approach to identify single and hybrid normalization for enhancing machine learning-based network intrusion detection. *IEEE Access*, 9:137494–137513, 2021.

[15] Meng Wang, Yiqin Lu, and Jiancheng Qin. A dynamic mlp-based ddos attack detection method using feature selection and feedback. *Computers & Security*, 88:101645, 2020.

[16] Pan Wang, Zixuan Wang, Feng Ye, and Xuejiao Chen. Bytesgan: A semi-supervised generative adversarial network for encrypted traffic classification in sdn edge gateway. *Computer Networks*, 200:108535, 2021.

[17] Wei Wang, Ming Zhu, Jinlin Wang, Xuewen Zeng, and Zhongzhen Yang. End-to-end encrypted traffic classification with one-dimensional convolution neural networks. In *2017 IEEE international conference on intelligence and security informatics (ISI)*, pages 43–48. IEEE, 2017.