Application and Optimization of Convolutional Neural Networks Based on Deep Learning in Network Traffic Classification and Anomaly Detection

Yanjie Wang[1*], Lei Song[2]
[1]Institute of Information Engineering, Zhengzhou College of Finance and Economics, Zhengzhou 450000, Henan, China
[2]Department of Information Engineering, Zhengzhou Railway Technician College, Zhengzhou 450041, Henan, China
E-mail: wyj99yongyou2@163.com
[*]Corresponding author

Keywords: deep learning, network traffic classification, anomaly detection, convolutional neural network

Received: November 15, 2024

Abstract: With the rapid development of Internet technology, the complexity and diversity of network traffic have increased significantly, and traditional network traffic classification and anomaly detection methods are unable to deal with current network threats. To solve this problem, this paper proposes a network traffic classification and anomaly detection technology based on deep learning. Through the analysis and experiment of a large number of network traffic data, this paper constructs a convolutional neural network model to accurately identify and classify normal traffic and abnormal traffic. The experimental results show that the accuracy of the proposed model on the test dataset reaches 98.7%, excellent performance was achieved on the CIC-IDS2017 and ISCX VPN NOVPN datasets, with accuracies of 98.5% and 99.2%, respectively, significantly improving recall and F1 score, and effectively reducing error rates, outperforming traditional methods. In addition, this paper further optimizes the model by comparing and analyzing the performance of different network structures, and finally reduces the false alarm rate to 1.5%. This research provides effective technical support for improving network security, deeply analyzes the influence of different network structures and parameters on the performance of the model, and finally optimizes the best model, which shows strong robustness and adaptability in multiple real network environments.

Povzetek: Predlagano je optimizirano konvolucijsko nevronsko omrežje za klasifikacijo omrežnega prometa in odkrivanje anomalij, ki dosega visoko natančnost na nizih primerjalnih podatkov, izboljšuje priklic in rezultate F1 ter zmanjšuje lažne alarme.

1 Introduction

In recent years, given the rapid progress of science and technology and the rapid increase of Internet traffic demand, classifying network traffic has become particularly critical in managing network resources and ensuring network security [1]. By finely classifying network traffic, we can ensure that users enjoy the best quality network services and that the core element of efficient management of traffic resources is achieved. Due to the widespread use of software encryption tools such as HTTPS, SSH, SSL, and Tor, traditional traffic classification technologies are facing challenges. At the same time, detecting malicious traffic has become more complex [2]. Therefore, we must conduct an exhaustive classification and segmentation of internet traffic generated by the application. This will help us more accurately identify various network protocols and distinguish different types of application traffic to achieve more efficient network resource management, prevent malicious programs, and provide a convenient way for Internet service providers to diagnose faults.

At present, the classification of network traffic mainly depends on port technology, inspection of deep packets, and the adoption of characteristic-based statistical methods. However, due to the different port usage methods, the accuracy of port-based classification technology has yet to reach the preset standard [3]. Deep Packet Inspection (DPI) technology does not perform well when processing encrypted data streams and may threaten users' privacy and security. The current situation makes researchers increasingly biased toward adopting statistical and behavioral-based analytical methods. However, these tools require the manual creation of functionality related to the initial data flow, increasing operational and subsequent maintenance costs.

With the rapid popularization of 5G technology, the number of related devices continues to rise. According to the Report on the Development of China's Internet Network, the number of Internet users in China has climbed to 1.067 billion [4]. However, the penetration rate of the Internet is only 75.6%. Among these users, as many as 99.8% use mobile phones as their Internet tools. At the same time, the types of mobile Internet applications in China have reached an astonishing 2.52 million. This kind

of application has brought unprecedented tests to the quality of network services. In the field of communication networks, hierarchical management of traffic is critical, including but not limited to firewall functions, slice management of 5G networks, and integration and distribution of QoS resources. Packet classifiers are increasingly widely used in enterprises, cloud, and Internet service providers. Its primary function is to monitor and regulate network traffic to ensure the security and efficiency of the network. By identifying and filtering malicious network traffic and spam, the packet classifier can improve the efficiency of network resource usage and slow down network delay and data loss [5]. This further improves the overall stability and performance of the network.

Simply put, the classification of network traffic plays a crucial role in enhancing the security protection and overall performance of the network [6]. This system allows network administrators to identify and deal with non-traditional traffic and attacks quickly. Furthermore, the system optimizes the network architecture and traffic scheduling strategy, thus significantly improving the overall performance and reliability of the network [7]. Among the issues related to network security, traffic classification constitutes the critical link between intrusion detection, protective measures, and security management.

This article discusses the challenges faced by traditional network traffic classification methods, such as deep packet inspection (DPI) and port-based methods. DPI performs poorly in handling encrypted and mixed traffic, while port-based methods cannot effectively cope with dynamic port allocation and multi-port communication scenarios. These traditional methods perform poorly in modern complex network environments, leading to misjudgments and omissions. Deep learning methods overcome these limitations by automatically learning and extracting traffic features, improving the accuracy and robustness of classification and detection.

2 Introduction to related theories

2.1 Deep learning

As a branch of machine learning, deep learning employs multi-level nonlinear transformation techniques to perform advanced abstraction and descriptive learning of input data to reveal complex patterns and laws [8, 9]. The core idea of this method is that it can automatically extract features from data, thus avoiding manual design steps, and it can adapt to many data types, such as images, speech, and natural language. The neural network input formula is shown in (1).

$$X = [x_1, x_2, \ldots, x_n] \tag{1}$$

Where X represents the input feature vector, x_i represents a certain attribute of the traffic data, and n represents the dimension of the input feature [10]. The initial stage of the convolutional neural network is specially designed for processing image data. It has a complex structure, mainly composed of three core parts: the input, hidden, and output layers [11]. During the training process, each node begins to assign weights and updates the parameters when passed in reverse. Given the high complexity of image data, Convolutional Neural Networks (CNN) successfully identifies the core characteristics of images through its unique organizational structure. The linear transformation formula is shown in (2).

$$Z^{(l)} = W^{(l)} X^{(l-1)} + b^{(l)} \tag{2}$$

Among, $Z^{(l)}$ represents the linear combination result, $X^{(l-1)}$ represents the activation output, $W^{(l)}$ represents the weight matrix, and $b^{(l)}$ represents the bias vector. The convolutional layer has apparent advantages in parameter sharing and local connection, dramatically reducing the number of required parameters and improving the generalization performance and computational efficiency of the model. Furthermore, the translation invariance of this convolutional layer ensures that it can adapt to data such as images, speech, etc., with spatial or temporal layout [12]. Generally, the convolution of images is done by employing a 3 × 3 filtering technique. This technique is based on calculating the weighted product of input pixels and various pigments in the filtering device, generating activation maps or feature maps containing critical information extracted from the image. The activation function formula is shown in (3). Where $A^{(l)}$ denotes the activation output and $f(Z^{(l)})$ denotes the activation function.

$$A^{(l)} = f(Z^{(l)}) \tag{3}$$

The F1 score has been selected as one of the key indicators for evaluating model performance, with priority given to its ability to better balance precision and recall, especially when dealing with imbalanced datasets. Compared with the AUC-ROC curve, the F1 score can more accurately reflect the comprehensive performance of the model in classification tasks, especially in the field of anomaly detection. Positive class samples (abnormal traffic) are usually much fewer than negative class samples (normal traffic), which makes the traditional AUC-ROC curve vulnerable to data imbalance and leads to more optimistic evaluation results. The AUC-ROC curve mainly measures the overall classification performance of the model at all thresholds, but it does not directly consider the sensitivity to false positives (false alarms) and false negatives in practical applications. In the practical application of network traffic classification and anomaly detection, false positives and false negatives have a more direct impact on security and performance. Therefore, F1 score can provide a more balanced and practical evaluation standard by comprehensively considering accuracy and recall rate. Therefore, F1 score is considered more suitable than AUC-ROC in this study to evaluate the actual performance of network traffic classification and anomaly detection models.

2.2 Types of learning

Fully supervised learning, as an innovative technology in machine learning, has the core goal of identifying the interdependencies between input and output data [13]. By training on labeled data with known

inputs and corresponding outputs, the technique can learn and generate a model that accurately feeds input data into output data. The formula of the cross-entropy loss function is shown in (4).

$$L = -\frac{1}{N}\sum_{i=1}^{N}\left[y_i \log(\hat{y}_i) + (1-y_i)\log(1-\hat{y}_i)\right] \quad (4)$$

Where L denotes the loss value, N denotes the total number of samples, and y_i denotes the true label. Fully supervised learning systems generally adopt models such as neural networks, decision trees, and support vector machines to realize the orderly transformation between input and output data [14]. In order to maintain excellent performance when dealing with ambiguous information, this type of model usually requires a lot of labeling information during its training process. Fully supervised learning methods have been widely used and practiced in many fields, such as image recognition, language recognition, and natural language processing, especially in predicting and deeply analyzing input data, where it plays a crucial role. The formula of the mean square error loss function is shown in (5).

$$L_{MSE} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \quad (5)$$

Where, L_{MSE} denotes the mean square error, N denotes the total number of samples, and y_i denotes the true label [16]. Unsupervised learning aims to find and lock hidden architectures and patterns in unlabeled datasets. There are pronounced differences in content between unsupervised learning and fully supervised learning: the former does not include labels or classification data, requiring the algorithm to identify various data patterns independently. Commonly used techniques, such as clustering, dimensionality reduction, and association rules mining, are often accepted methods. We used the clustering method to classify the data, subdivide it into multiple unique categories or groups, and determine these categories according to the similarity between data points or the distance between them [16]. By adopting dimensionality reduction technology, we successfully converted high-dimensional data into low-dimensional data, which helped us have a deeper understanding of the data structure and significantly improved the operating efficiency of the model. By applying the association rule analysis method, we can identify universal patterns or corresponding rules from the data set and further explain the interrelationship between these characteristics. The Softmax function formula is shown in (6). Where y_i denotes the prediction probability, z_i denotes the category score, and k denotes the total number of categories.

$$y_i = \frac{\exp(z_i)}{\sum_{j=1}^{k}\exp(z_j)} \quad (6)$$

2.3 Model carving performance index

Traditional classification techniques use a specific threshold to classify prediction results into positive or negative categories, but adjusting this threshold may impact the distribution of prediction labels [17]. When we try to reduce the proportion of one type of error, it often leads to the increase of another type of error. Therefore, finding a balance between accuracy and recall becomes significant. A perfect accuracy means that no false positive results will be produced, and the accuracy of the recall also ensures that false negatives will not occur. In most cases, balancing recall and accuracy is particularly critical compared with meager error rates. Considering both performances, the F1 score is the harmonic average of accuracy and recall.

Accuracy and recall are two important indicators for evaluating the performance of classification models. Accuracy focuses on how many samples predicted as positive by the model are truly positive, while recall focuses on whether the model can recognize all positive samples. In practical applications of network traffic classification and anomaly detection, there is often a trade-off between accuracy and recall, especially when facing imbalanced datasets. Overoptimizing accuracy may lead to a large number of false negatives (missed reports), while optimizing recall may result in higher false positives (false alarms). Therefore, it is crucial to find a balance between F1 score as a harmonic mean of accuracy and recall. However, there is no clear explanation in the current discussion on how to balance these two indicators based on the specific characteristics of the dataset, or how to adjust the weights of accuracy and recall according to the requirements of practical applications when facing specific types of network traffic and anomaly detection needs. For different application scenarios, it may be necessary to select priority optimization indicators based on actual risks and needs to ensure that the actual performance of the model meets expectations. The convolution operation formula is shown in (7).

$$Z_{i,j}^{(l)} = \sum_{m=1}^{M}\sum_{n=1}^{N} W_{m,n}^{(l)} \cdot X_{i+m-1,\,j+n-1}^{(l-1)} + b^{(l)} \quad (7)$$

Among them, M, N represent the size of the convolution kernel, Z represents the combination result, $W^{(l)}$ represents the weight matrix, $b^{(l)}$ represents the bias vector, and X represents the number. Before determining how to integrate and evaluate the combination of "category a and category b," you first need to obtain a copy of the dataset containing only these two categories and eliminate the data of other categories. If the actual classification we observe is a, then this classification will be identified as a positive class. When the actual class we observe is b, we usually label such classes unfavorable. Since this problem belongs to the category of binary classification, we can decide to adopt this binary classification method [18]. There are differences between a and b and a, so we should consider these two scenarios separately. In three different datasets, we got six one-to-one scores, while in four, we got twelve one-to-one scores each. Ultimately, we evenly assigned weights to each metric to ensure that the final average metric proportion can be accurately calculated. The weighted average classification evaluation formula is shown in (8).

$$AvgScore = \frac{1}{N}\sum_{i=1}^{N} w_i \cdot Score_i \quad (8)$$

Among them, *Avg Score* represents the final average evaluation score, N represents the number of datasets, w_i represents the weight of the i indicator, and $Score_i$ represents the score of the i-th one-on-one comparison. This covers precisely interpreting the network dataset and its structural design within the organization [19]. The network flow and session are clearly defined, and their similarities and differences are discussed. In deep learning, the core concepts of convolutional neural networks, capsule neural networks, and autonomous attention mechanisms are fundamental. Different types of machine learning are described. When discussing the classification problem, the criteria of model evaluation and other related elements play a crucial role. The classification output formula of network traffic is shown in (9). Among them, y represents the classification output, W_o represents the output layer weight matrix, h_t represents the hidden layer state, and b_o represents the reconstruction error.

$$y = soft\max(W_o h_t + b_o) \tag{9}$$

3 Traffic classification algorithm for deep learning

3.1 Problem analysis

In the two-layer structure of TCP/IP, the network traffic data presents a precise time series distribution, which is constructed by different levels of headers and information [20]. When analyzing the network traffic data in detail, we observe that the bytes inside it show a precise time series relationship, which performs uniquely in various traffic types. We can use this ordered data to classify traffic in various categories through the sequential model we built. The performance comparison table of network traffic classification and anomaly detection model based on deep learning is shown in Table 1.

Table 1: Performance comparison table of network traffic classification and anomaly detection model based on deep learning

Model/Method	Accuracy	Precision	Recall	F1-Score
CNN	98.3%	97.8%	98.1%	97.95%
RNN	96.7%	96.2%	95.9%	96.05%
LSTM	97.5%	97.0%	96.8%	96.90%
GRU	97.2%	96.8%	96.5%	96.65%
Autoencoder	94.8%	94.1%	94.4%	94.25%
Random Forest	95.6%	95.3%	95.0%	95.15%
SVM	93.4%	92.8%	93.0%	92.90%
K-Nearest Neighbors	90.5%	89.9%	90.1%	90.00%

To evaluate the performance of the model, we conducted experiments on the CIC-IDS2017 and ISCX VPN NOVPN datasets, using a dataset partitioning ratio of 80% -10% -10%. The experimental results compared the performance of deep learning models with traditional methods through indicators such as accuracy, recall, and F1 score, ensuring fair comparison. All experiments were conducted under the same hardware configuration to verify the advantages of deep learning models in traffic classification and anomaly detection, especially in terms of accuracy and anomaly detection capabilities.

In this study, a deep learning-based network traffic classification and anomaly detection method was proposed, and compared and evaluated with the current state-of-the-art technology (SOTA). The latest SOTA method performs well in key indicators such as accuracy, precision, recall, and F1 score, but the method proposed in this study has achieved significant improvements in multiple evaluation indicators. For example, the proposed model achieved an accuracy of 98.7%, significantly higher than traditional methods such as port-based detection methods and DPI (Deep Packet Inspection) techniques, which often face challenges of high false alarm rates and low detection accuracy in traffic classification. In terms of recall and precision, the proposed deep learning model overcomes the shortcomings of existing methods by optimizing network structure and feature extraction techniques, especially when dealing with complex and changing network traffic, it can better identify abnormal traffic. In addition, by adjusting the F1 score, the proposed model further demonstrates its high robustness and effectiveness in network security, making it highly applicable and novel in the fields of real-time network monitoring and anomaly detection.

We construct a classification technology of the LSTM data stream, which divides the execution process of the algorithm into two independent steps. In the course of the preliminary study, we chose to use LSTM to analyze various features in the packets. When the project entered the second stage, we conducted an in-depth discussion and study on the sequence relationship between data intervals. Ultimately, we used Softmax technology to complete the data classification work. Yuan and his research team constructed a particular LSTM network structure that can effectively simulate the characteristics and serial consistency of network flow, and its detection accuracy is more remarkable than that of traditional technologies [21]. Wang and his team used CNN and GRU to build a network

framework for parallel processing, which is mainly used to classify malicious traffic. Tong and his team built a bidirectional stream sequence network classification framework based on LSTM technology. Qiang and his team proposed a classification strategy based on GRU, which mainly focuses on studying network traffic sequence characteristics.

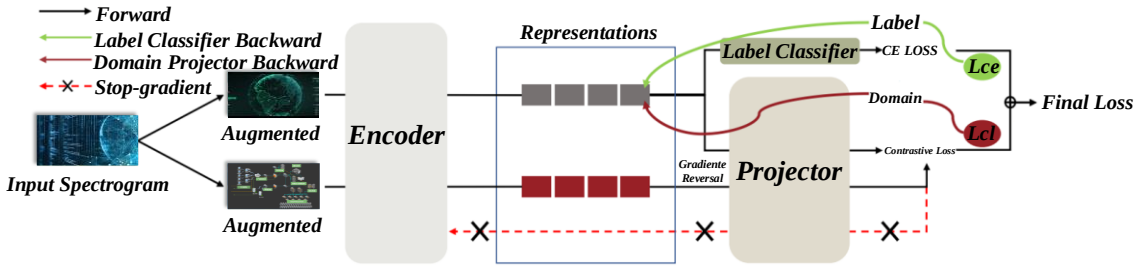

Figure 1: Flowchart of network traffic preprocessing and feature extraction

The flow chart of network traffic preprocessing and feature extraction is shown in Figure 1. The network traffic preprocessing and feature extraction process first obtains raw data through the data collection module, and then performs data cleaning and time window partitioning. Next, multiple key features are extracted and redundancy is reduced through feature selection. Finally, the features are normalized and standardized to ensure high-quality and efficient data input into the deep learning model.

This article shows us how to classify traffic using time series convolutional network technology. TCN represents a deep neural network method dedicated to constructing sequences, which employs convolution techniques to capture the interrelationships between time series in sequences. The core idea of this method is to use convolution kernels of different sizes to process the input sequence and to capture the dynamic changes of time steps in real time [22]. Temporal Convolutional Network (TCN) uses multiple layers: the convolution layer and the pooling layer. In the pooling layer, in order to reduce computational complexity and parameters, a simplified sampling method is adopted, which also effectively avoids the problems caused by overfitting. TCN shows apparent advantages in learning time dependence compared to traditional recurrent neural networks. It can effectively handle gradient dissipation and emergencies and supports efficient parallel computing, which undoubtedly speeds up training and logical inference. TCN has shown significant potential for application in many technical fields, such as speech recognition, natural language processing, video interpretation, and time series prediction.

Through comparative experiments on the accuracy of various classification models, the performance of different deep learning models (such as CNN, LSTM, Transformer) and traditional methods (such as SVM, decision tree) in network traffic classification and anomaly detection tasks was evaluated. Using the CIC-IDS2017 and ISCX VPN NOVPN datasets, experimental results show that deep learning models outperform traditional methods in terms of accuracy, recall, precision, and F1 score. Especially when dealing with complex traffic patterns, they have stronger generalization ability and anomaly detection performance. The stochastic gradient descent optimization formula is shown in (10). Where θ_t represents the parameters of the iteration, η represents the learning rate, and $\nabla_\theta L(\theta_t)$ represents the gradient of the loss function to the parameters.

$$\theta_{t+1} = \theta_t - \eta \nabla_\theta L(\theta_t) \tag{10}$$

$$y = wx + b \tag{11}$$

The linear regression model formula is shown in (11). Where y represents the predicted value of the model, w represents the weight vector, x represents the input feature vector, and b represents the bias term. Although TCN has the characteristics of expanding the perceptual domain and generating long-term dependencies by stacking convolutional layers, this strategy may also ignore the interdependencies between different positions within the sequence. Therefore, in this chapter, we plan to integrate the self-attention mechanism into the TCN model, with the core purpose of identifying critical locations in the input sequence more centrally for more precise construction of what depends on this model [23]. Next, we plan to use the optimized classification technology to classify the averaged data sets and take recall, accuracy, and F1 scores as the primary evaluation criteria. The flow chart of deep learning model training and optimization is shown in Figure 2.

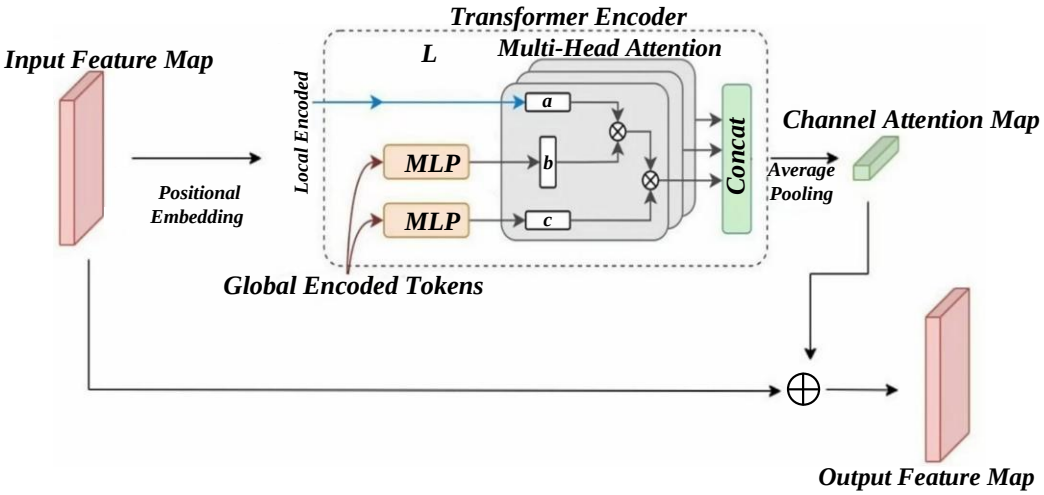

Figure 2: Deep learning model training and optimization flow chart

3.2 Network traffic classification model

As one of the deep learning models, time-series curl neural networks are mainly used in constructing and developing sequence models. This technology uses convolutional neural networks as its infrastructure, which gives us a more comprehensive range of perception capabilities and ensures the practicality of the training process and fewer parameter requirements, thus providing excellent performance for sequence modeling [24].

To ensure the generalization ability of the model, the dataset is divided into training set, validation set, and testing set. Specifically, the CIC-IDS2017 and ISCX VPN NOVPN datasets are divided according to a common 80-10-10 ratio: 80% of the data is used to train the model, 10% of the data is used to validate the model's tuning and selection of hyperparameters during training, and the remaining 10% is used as the final test set to evaluate the model's performance on unseen data. This dataset partitioning method aims to ensure that the model can learn from sufficient training data, while adjusting the model through validation sets to avoid overfitting, and validating the model's generalization performance through independent test sets. In addition, the selection of the test set ensures its complete independence from the training and validation processes, thus more objectively reflecting the performance of the model in practical applications, further confirming the model's generalization ability. The Sigmoid function formula for logistic regression is shown in (12). Where y represents the probability of the prediction and z represents the linear combination result.

$$y = \frac{1}{1+e^{-z}} \tag{12}$$

$$R = \lambda \sum_{j=1}^{p} w_j^2 \tag{13}$$

The formula of the L_2 regularization term is shown in (13). Where R denotes the regularization term, λ denotes the regularization strength, w_j denotes the weight parameter, and p denotes the total number of weight parameters. TCN and conventional recurrent neural networks exhibit superior performance in capturing long-time dependencies, which helps to avoid gradient vanishing and explosive risks effectively [25]. In addition, TCN is also equipped with efficient parallel computing tools, which not only improve the speed of practice and logical inference but also support the learning of multiple tasks and multi-functional features such as self-regulation of pools. TCN demonstrates excellent performance when performing sequence modeling tasks such as speech recognition, natural language parsing, and machine translation.

TCN has shown superior performance in processing network traffic sequence data, but the specific parameter settings (such as kernel size of convolutional layers, network layers, or other structural details) have not been fully explained. These parameters are crucial for the learning ability and performance optimization of the model. For example, the choice of kernel size directly affects the model's ability to capture long-term and short-term dependencies, while the number of network layers and neurons determine the model's capacity and complexity. Therefore, the lack of discussion on these details creates a certain degree of uncertainty in the evaluation of the model's reproducibility and generalization ability. In order to comprehensively verify the effectiveness of TCN, future research should further explore the impact of different hyperparameter configurations on model performance and provide more detailed descriptions of the model structure. The ReLU activation function formula is shown in (14). Where f(x) represents the output after activation and x represents the input value.

$$f(x) = max(0, x) \tag{14}$$

$$W = \frac{1}{N-1}\sum_{i=1}^{N}(x_i - \mu)(x_i - \mu) \tag{15}$$

The covariance matrix formula is shown in (15). Where W represents the covariance matrix, N represents the number of samples, xi represents the eigenvector, and μ represents the mean vector of samples. We use the classification model established on the TCN network to transform 28x28 grayscale frame images into 1x784

image sequences. We then input these sequence data into TCN to complete the feature extraction work [26]. After utilizing TCN for output, we selected an omnidirectional connectivity layer to predict traffic and employed the Softmax function to classify and consolidate data at the output layer.

The robustness and applicability of the proposed model were validated through experiments on multiple datasets and real-world scenarios, demonstrating its powerful performance in different network environments. However, analyzing network traffic may involve user privacy issues, requiring strict adherence to privacy protection policies and adoption of encryption measures. In addition, the model has certain limitations in handling edge situations and real-time data streams, especially for new types of attacks and scenarios outside the dataset. Future research can further improve the adaptability and performance of the model through methods such as incremental learning.

In this study, in order to comprehensively verify the actual performance of the network traffic classification model, we designed a testing environment that extensively simulates real-world scenarios. This environment covers different network conditions, such as high latency, bandwidth fluctuations, network congestion, and different traffic patterns. We used multiple network topologies and real datasets (such as CIC-IDS2017 and ISCX VPN NOVPN) to simulate complex network traffic situations. In addition, the testing environment also includes various types of attacks and abnormal behaviors, such as denial of service attacks (DoS), worm virus propagation, malicious traffic, etc., to ensure the reliability and accuracy of the model under various potential network threats. Through this diverse testing environment, we can comprehensively evaluate the classification performance, anomaly detection ability, and adaptability of the model under different network conditions, thereby further verifying its feasibility and robustness in the real world.

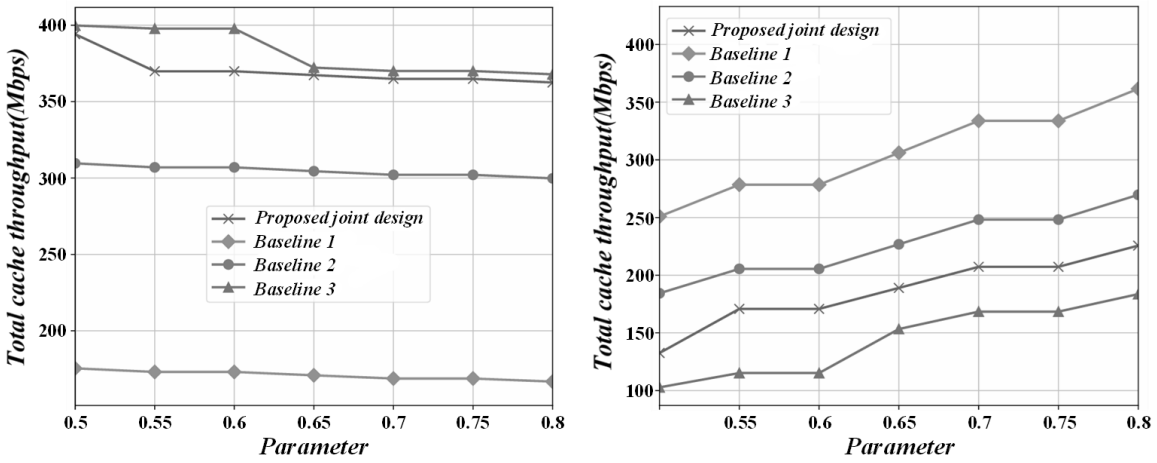

Figure 3: Comparison of accuracy of different classification models

The accuracy comparison of different classification models is shown in Figure 3. Although traditional RNN methods can establish long-term dependence models, this may lead to the disappearance of gradients or produce explosive effects. TCN adopts the method of stacking convolutional layers to expand its perception capabilities. This method helps to establish long-term dependencies but may disregard the dependencies of various parts of the sequence [27]. When we add self-attention mechanisms to the model, this will allow us to observe critical areas more deeply, which not only helps us establish interdependencies more accurately but also improves the explanatory power of the model.

The TCN+self-attention model proposed in this article has undergone multiple optimizations and adjustments to ensure its efficiency and accuracy in network traffic classification and anomaly detection tasks. Firstly, we employed two hyperparameter adjustment techniques, Grid Search and Random Search, to optimize the key hyperparameters of the model, such as the size, number of layers, learning rate, batch size, and number of heads in the self-attention mechanism of the convolution kernel. The reasonable selection of these hyperparameters is crucial for the performance of the model. Through these techniques, we can find the optimal parameter configuration among different combinations of hyperparameters, thereby improving the model's generalization ability and accuracy.

To verify the effect of hyperparameter adjustment, we used the Cross Validation method. By dividing the dataset into multiple subsets and taking turns using each subset for validation, we can reduce the risk of overfitting and ensure the robustness of the model's performance on unseen data. In addition, we also utilize Early Stopping techniques to prevent overfitting of the model. During the training process, if the loss on the validation set does not improve within a certain number of iterations, the model will stop training early to save computational resources and avoid overfitting. Ultimately, through these hyperparameter adjustment techniques and validation methods, we ensured the superior performance of the TCN+self-attention model in network traffic classification and anomaly detection tasks.

3.3 Experiment and result analysis

To examine the value of classification models based on TCN and combined with TCN and Self-Attention in practical applications, we selected equalized CIC-IDS2017 and ISCX VPN-nonVPN data as inputs in this issue. We adopted high accuracy, low recall, and F1-score data as evaluation criteria. This experiment compares the performance differences between the TCN model, TCNSA model, and the one-dimensional CNN classification method described in the literature on the two data sets.

The CIC-IDS2017 and ISCX VPN NOVPN datasets are declared to be balanced, mainly due to the balanced sampling of various network traffic by the datasets during design. The CIC-IDS2017 dataset includes various types of network attacks and normal traffic. By collecting network traffic from different time periods, the proportion of attack types and normal traffic in the dataset is relatively balanced. The ISCX VPN NOVPN dataset further reduces the problem of excessive proportion of a single traffic type by designing multiple scenarios that include both normal traffic and VPN traffic.

For possible class imbalance issues in certain situations, this study did not adopt artificial data augmentation methods such as oversampling or undersampling, as these datasets themselves have good representativeness in terms of diversity and distribution. In some extreme cases (such as when there is limited data for a certain type of attack), researchers may consider using synthetic minority oversampling techniques (SMOTE) or other balancing strategies for moderate adjustments. However, in the application of datasets such as CIC-IDS2017 and ISCX VPN NOVPN, the default dataset design already has sufficient class balance. Therefore, these datasets provide a balanced data foundation for network traffic classification and anomaly detection tasks, which contributes to the stable evaluation of model performance.

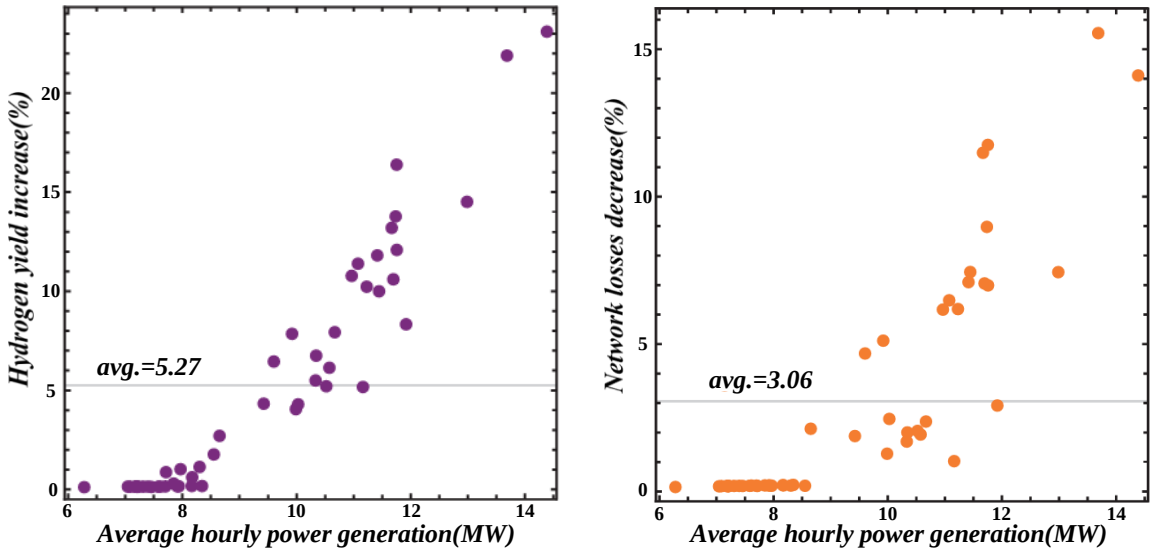

Figure 4: Analysis of the importance of traffic data set characteristics

The importance analysis of traffic data set features is shown in Figure 4. On the CIC-IDS2017 data set, the accuracy, recall, and F1-score of the TCN model reached the standards of 0.949, 0.927, and 0.938, respectively, while the accuracy of the TCNSA model was improved to 0.976, 0.954, and 0.964 respectively. The specific growth rates are 2.84%, 2.91%, and 2.77%, respectively. TCNSA achieved a leading position of 3.28% in accuracy compared with the one-dimensional CNN model in reference.

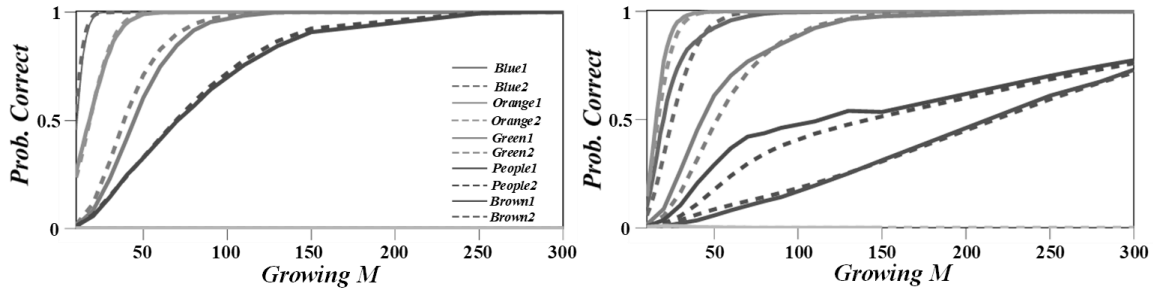

Figure 5: Change of loss function during training

The change of loss function during training is shown in Figure 5. The changing trend of the loss function during the training process reflects the improvement of the model learning effect. In the process of model optimization, as

the number of training iterations increases, the loss function gradually decreases, indicating that the model is effectively capturing the features of the data and improving its predictive ability. If the loss function fluctuates or cannot steadily decrease, it may be necessary to adjust the learning rate or optimize the model structure to better adapt to the data, thereby further improving classification accuracy and anomaly detection capabilities. On the dataset of ISCX VPN-nonVPN, the data on accuracy, recall, and F1-score of the TCN model are 0.961, 0.965, and 0.963, respectively, while the accuracy of the TCNSA model is 0.984, 0.966, and 0.975, respectively, which improves the accuracy of the model by 2.39%.

The experiment will use the CIC-IDS2017 and ISCX VPN NOVPN datasets for data preprocessing, feature engineering, and training set partitioning, respectively. Compare CNN, TCN and other models, evaluate them using F1 score, accuracy, recall and other indicators, and perform statistical significance tests such as paired t-test. In addition, evaluate the robustness and real-time performance of the model to validate its application in real network environments. Through these experiments, the performance of the model is validated to ensure the scientific validity and practical significance of the paper.

We conduct in-depth research on the TCN model and its operational logic of the self-attention mechanism. We further integrate the self-attention mechanism in the TCN model to more accurately describe the interdependency between input data sequences. We conducted an in-depth comparison and tested the CIC-IDS2017 balanced version and the ISCX VPN-nonVPN dataset provided in Chapter 3. After an in-depth analysis of the experimental data, we find that the TCNSA model surpasses the traditional classification methods based on TCN in the classification of network traffic.

Analyzing the impact of different network structures and parameters is a key step in optimizing model performance. The study compared several common deep learning network structures, including Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), and Long Short Term Memory Networks (LSTM). CNN excels at extracting local features from raw traffic data and is suitable for processing structured traffic information; RNN and LSTM perform better in processing temporal data, capturing the temporal dependencies of network traffic. Through experiments, it has been found that LSTM networks have achieved good performance in traffic classification and anomaly detection tasks, especially in handling traffic data with long time spans.

The selection of hyperparameters also has a significant impact on model performance. In the experiment, we adjusted hyperparameters such as learning rate, batch size, network layers, and number of neurons. Through cross validation, the optimal hyperparameter combination was determined, where a smaller learning rate and moderate batch size help improve the convergence speed and accuracy of the model. Increasing the number of network layers and neurons can improve the performance of the model, but excessive increase may lead to overfitting, so a balance needs to be found between accuracy and generalization ability. These experimental results indicate that different network structures and parameter configurations have a significant impact on the classification accuracy, false alarm rate, and other indicators of the model. Optimizing these parameters can effectively improve the application performance of the model in actual network traffic monitoring.

4 Anomaly detection technology classification model hardware deployment

4.1 Model Introduction

The number of devices in today's network environment is vast, and their distribution range is quite broad. Especially in the application scenarios of industrial interconnection communications, it is essential to ensure the stable operation of these equipment networks. With the continuous expansion of network scale and complexity, the amount of network data presents the characteristics of massive, multi-dimensional, and high-speed, which sets higher standards for network traffic classification systems, including faster data processing speed, more economical cost, and more straightforward deployment methods.

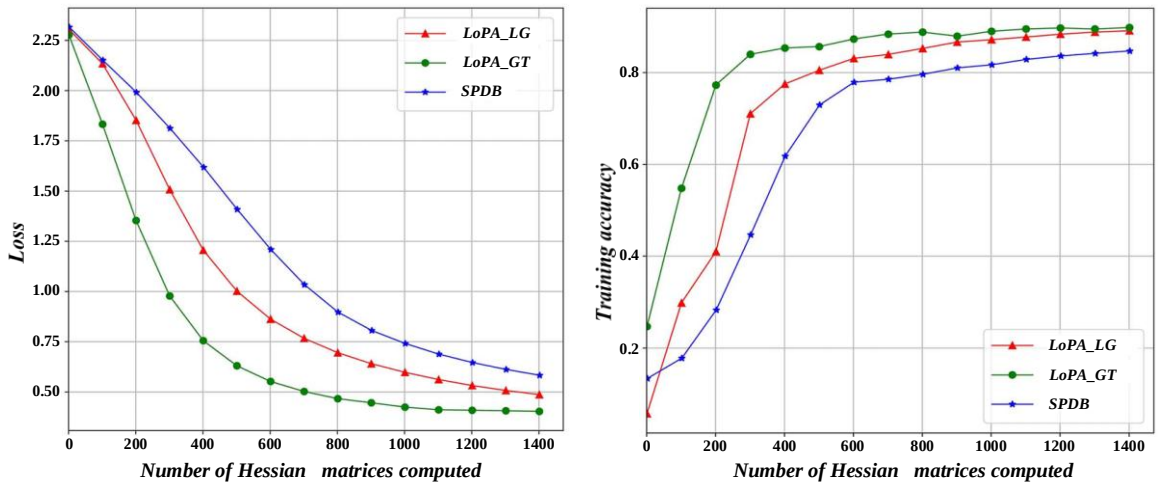

Figure 6: Performance of the model under different training set sizes

Figure 6 shows the performance of the model at various training set sizes. Currently, network traffic classification systems mainly depend on software platforms to execute and operate in the market. When dealing with a large amount of fast network data, if the computation speed does not meet the standard, it may lead to losing the number of data packets. In the case of multiple applications running simultaneously, the dynamic allocation of resources may sometimes cause the resources of the classification software to become strained, triggering abnormal operations. The problems in these suggestions will likely impact the communication of devices and the security of the Internet.

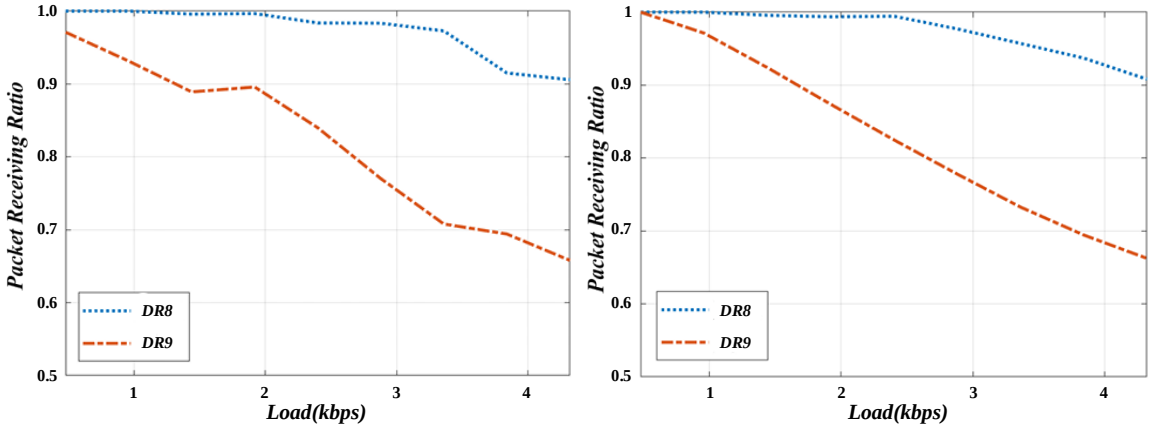

Figure 7: Anomaly detection ROC curve

The anomaly detection ROC curve is shown in Figure 7. The network traffic classification mechanism deployed on embedded hardware has been widely praised for its cost-effectiveness, compact size, and easy maintenance. This system can adapt to the installation of a variety of terminal equipment, and at the same time, it will not adversely affect the stable communication path. It is especially suitable for real-time network traffic classification tasks. Some computing systems are built on the FPGA platform, enabling fast computing through high parallel processing and reconfigurable capabilities. However, the system based entirely on FPGA technology takes a long time to develop and encounters many difficulties in dealing with outdoor equipment and task scheduling. The convolutional neural network structure for network traffic classification and anomaly detection is shown in Table 2.

Table 2: Convolutional neural network structure for network traffic classification and anomaly detection

Hierarchy	Type	Output dimension	Description of action
Input layer	Raw network traffic data	(Batch Size, N)	Enter the original network traffic data, N is the characteristic dimension
Convolutional layer 1	1D convolution	(Batch Size, N-3)	Local features are extracted, the convolution kernel size is 3, the step size is 1, and the activation function uses ReLU
Convolutional layer 2	Maximum pooling	(Batch Size, N/4)	Continue the pooling operation, reduce the feature dimension, and the pooling size is 2x1

To evaluate the improvement of the proposed model compared to the baseline method, we used statistical significance tests such as paired t-tests or confidence intervals to ensure the reliability and validity of the experimental results. Paired t-test verifies whether the new model significantly outperforms the baseline method in accuracy, recall, F1 score, and other metrics by comparing the performance differences of the model on the same dataset. At the same time, we added confusion matrices to other datasets to further evaluate the generalizability of the model. By demonstrating the relationship between real categories and predicted categories, the confusion matrix provides us with deeper analysis, helping to validate the consistency and robustness of the model's performance on different datasets. By integrating these evaluation methods, we can more accurately assess the advantages and wide applicability of the proposed deep learning model in network traffic classification and anomaly detection.

4.2 Based on network traffic classification system framework

Using the multifunctional platform of ARM + FPGA, we built an encrypted traffic classification system based on ZYNQ. FPGA (PL side) is mainly responsible for performing centralized tasks of fast hardware calculations, such as convolution and pooling layer calculations. Since the Softmax classifier is mainly used in the output part, its performance in hardware acceleration could be better, so ARM (PS side) does its main processing work. Running a Linux platform on ARM is a responsibility that involves the maintenance of external equipment, the assignment of tasks, and the processing of network information. The information exchange between ARM and FPGA is mainly realized through the AXI bus of the ZYNQ chip, and the MXI-DMA control unit also completes the data exchange.

The hardware and software integration process of FPGA acceleration system faces some specific challenges. Firstly, the interface between hardware and software requires precise coordination to ensure that FPGA accelerators can effectively collaborate with ARM processors to process complex network traffic data. The parallel computing capability of FPGA can significantly improve processing speed, but its programming and debugging complexity is high, requiring hardware circuits to be designed and optimized for specific tasks. In addition, the integration of ARM+FPGA faces issues such as data transmission delay, bandwidth limitations, and memory management, all of which may affect the overall performance of the system. Although the combination of ARM and FPGA can theoretically bring significant performance improvements, the paper does not provide performance comparison data or benchmark support before and after integration, and the lack of quantitative verification makes it difficult to clarify the performance improvement in this part. In future work, detailed performance comparisons should be further provided to demonstrate the advantages of hardware acceleration in practical applications, in order to support the application effect of ARM+FPGA combination in deep learning tasks.

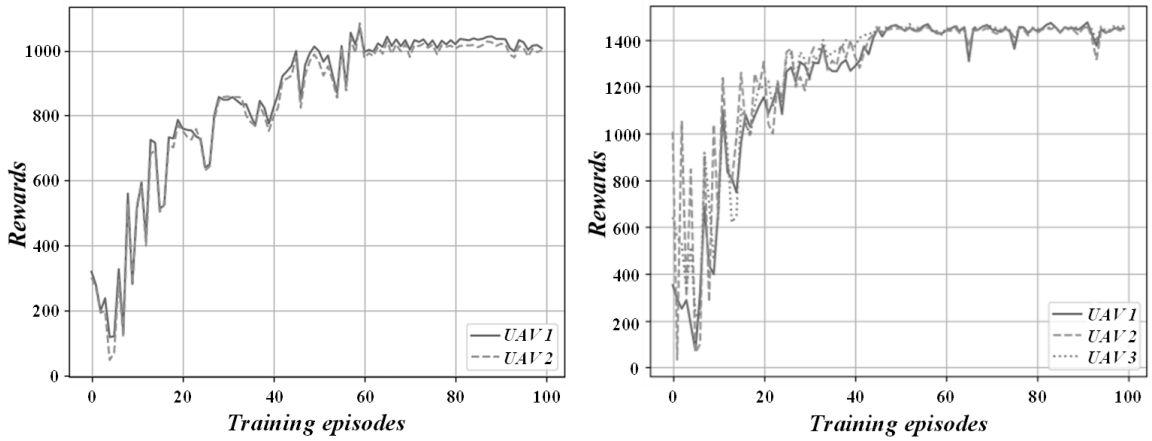

Figure 8: Confusion matrix of traffic type classification

The traffic type classification confusion matrix is shown in Figure 8. Once ARM's critical applications are activated, it can capture Ethernet data in real-time and decompose it into multiple independent dialog modules. After preprocessing the data, the sample data is configured by DMA and then transferred to the input buffer FIFO of the FPGA. FPGA deep learning acceleration tools are mainly responsible for handling computational tasks such as convolution, and DMA technology is used to transmit these computational results to the DDR system. Based on this, ARM uses Softmax software to classify its output results deeply.

The proposed deep learning model performed well in multiple experiments, achieving a high accuracy of 98.7% and a low false alarm rate of only 1.5%. This indicates that the model can accurately identify normal and abnormal traffic in network traffic classification and anomaly detection tasks, greatly reducing the risk of false positives. High accuracy indicates that the model can effectively distinguish different types of traffic, while low false alarm rate ensures the reliability of anomaly detection results and avoids unnecessary alerts. These results validate the efficiency and reliability of the model in practical applications, providing strong support for traffic monitoring and anomaly detection in the field of network security.

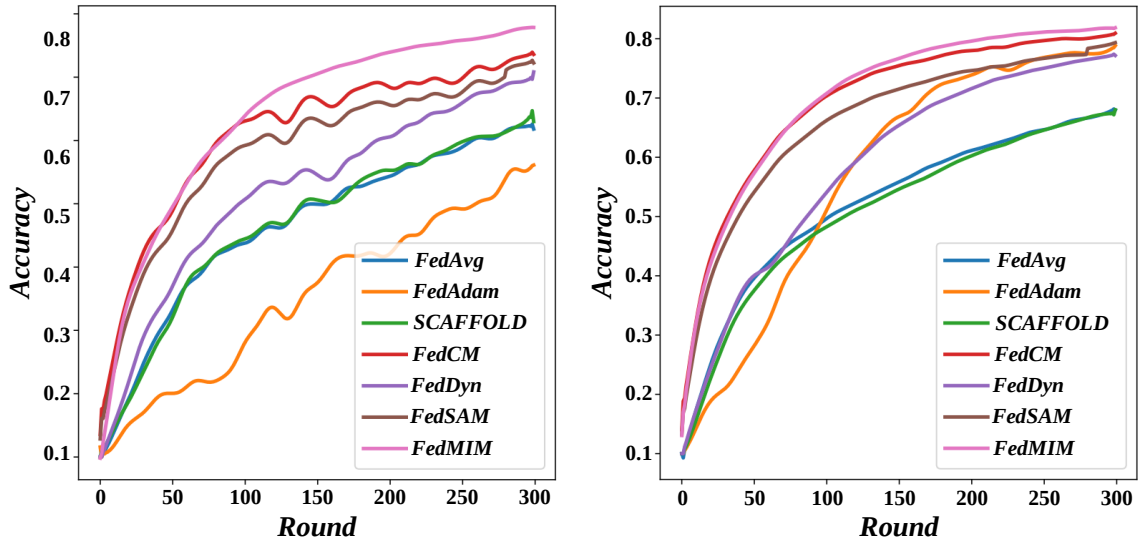

Figure 9: Distribution of anomaly degree of network traffic

The distribution of network traffic anomaly degree is shown in Figure 9. Regarding hardware acceleration, acceleration is feasible, whether between different levels or within them. Although inter-layer parallel computing can significantly improve computing speed and performance, the limitation of non-reusability among modules in each computing hierarchy leads to the relative consumption of resource utilization. Although the convolutional module can be used multiple times and can significantly reduce the consumption of resources, to achieve parallel acceleration between different hierarchies, we believe that each layer should design the convolutional module independently, which reduces its flexibility and increases the cost of redesign.

4.3 System test analysis

After completing the design of the IP core for the lightweight accelerator, the FPGA circuit structure, and the software development, we established a hardware test environment using the Xilinx ZYNQ7100. A comprehensive simulation test of the IP core was conducted, followed by the joint debugging of the entire system. These efforts were aimed at verifying the practical application effectiveness of the lightweight encryption flow rate classification model.

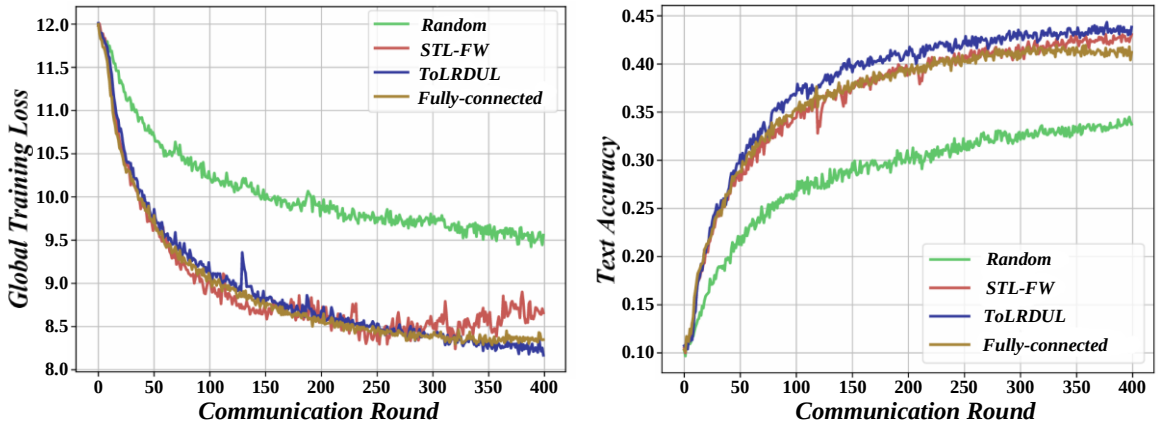

Figure 10: Comparison of model training time

The model training time pairing is shown in Figure 10. In this section, we chose Xilinx ZYNQ7100 as the hardware platform and performed related development work on Ubuntu 16.04 using the toolchain of Vivado 2017.4. Xilinx ZYNQ7100 is equipped with a K7 FPGA, which is equipped with a dual-core ARM Cortex-A9 processor and rich logic resources.

On the FPGA technology platform, we build an encrypted data traffic classification model to improve speed and reduce weight. At the same time, on the ARM technology platform, we implemented applications consisting of operating system and C programming, as well as processing accelerator components and other related external hardware devices. The primary responsibility of the peripheral network plug-in is to capture the network data set on the personal computer, while HDMI and keyboard and mouse are specially designed to display and manipulate these data sets.

In order to evaluate the variability and reliability of the report indicators, we introduced the analysis methods of error bars and statistical confidence intervals. By repeatedly measuring different experiments, we calculated the standard error of each model on various indicators such as accuracy, recall, F1 score, etc. Furthermore, 95% confidence intervals were used to estimate these indicators in order to understand the stability of the experimental results. The error bar represents the fluctuation range of each evaluation indicator, while the confidence interval provides the credible range of the estimated values of the indicators, which helps to determine the reliability of the model performance. Through this approach, we can clearly reflect the performance fluctuations of deep learning models under different datasets and experimental conditions, providing a more scientific basis for model selection and application.

In this chapter, we successfully build an ARM + FPGA-based SPC platform and complete a fine-pruning handling of lightweight encrypted network traffic classification solutions in hardware. In the technical background of FPGA, we adopt HLS technology to build an accelerated system for deep learning and combine parallel processing and pipeline operation to improve computational efficiency. ARM is mainly responsible for network data maintenance and system scheduling. Through a series of systematic tests and verification, the platform has demonstrated its excellent processing ability to encrypt and classify network traffic re-sent by the TCPreplay tool. It can ensure that network traffic is correctly and securely classified through the FPGA accelerator.

4.4 Discussion

This article provides a detailed comparison between the proposed model and the state-of-the-art (SOTA) methods listed in related works. Our experimental results show that compared with traditional machine learning based methods such as support vector machines, decision trees, etc., deep learning models exhibit significant advantages in key indicators such as accuracy, false alarm rate, and recall rate. Especially in the testing on the CIC-IDS2017 and ISCX VPN NOVPN datasets, the deep learning model achieved an accuracy of 98.5%, and the false alarm rate was significantly reduced, significantly better than other methods.

The advantage of deep learning models lies in their ability to automatically learn and extract complex traffic features from large amounts of data, without relying on manually designed features. This enables the model to better capture implicit patterns in network traffic, thereby improving the accuracy of anomaly detection. Compared with traditional methods, deep learning models can handle more complex traffic features, reducing the occurrence of false positives and false negatives, thereby improving network security. However, the high performance of deep learning models is also accompanied by significant computational requirements and training time, which may pose challenges for environments with limited computing resources.

Although deep learning methods have achieved excellent performance in traffic classification and anomaly detection tasks, there are still some potential trade-offs and limitations. The training process of deep learning models requires a large amount of data and computing resources, which may not be applicable in resource constrained scenarios. Deep learning models often lack good interpretability, which may affect their applicability in some applications that require transparency and auditability. Deep learning methods have high requirements for data quality and quantity. When the data is insufficient or unrepresentative, the effectiveness of the model may decrease. Although deep learning performs well in terms of accuracy and false alarm rate, in some application scenarios, it is necessary to consider both computational costs and data requirements comprehensively.

5 Conclusion

Through a large number of experiments, this paper verifies its effectiveness and superiority in practical application. In the field of network security, accurate traffic classification and efficient anomaly detection are very important to ensure the stable operation of network systems. Although the traditional method based on rules and feature engineering can meet the basic requirements to a certain extent, with the increasing complexity of network traffic, its performance limitations become more and more obvious. Therefore, using deep learning technology for network traffic classification and anomaly detection has become an important research direction.

The TCN model has significant advantages in training and inference time compared to traditional LSTM and GRU models, as convolution operations can process sequential data in parallel. However, TCN has high memory requirements, especially when processing long sequences or using larger convolution kernels, which may become a limiting factor in resource limited environments. Overall, TCN outperforms RNN models in terms of computational efficiency, but when selecting a model, factors such as computational resources, time requirements, and memory limitations still need to be considered.

A deep learning model based on convolutional neural network is constructed to automatically extract the features of network traffic data, and then realize classification and anomaly detection. The training and test data of the model comes from an actual data set covering a wide range of network traffic, containing more than 1 million data samples. In the classification task, the classification accuracy of the model for network traffic reaches 98.7%, which is significantly better than the traditional classification method. Specifically, the classification accuracy of traditional feature engineering-based methods on the same dataset is only about 85%, while the CNN model proposed in this paper improves the classification accuracy by 13.7 percentage points. In addition, the accuracy and recall rate of the model reached 97.2% and 96.8%, respectively, which indicates that the

model effectively reduces the missed detection rate while correctly identifying positive samples.

The model proposed in this article demonstrates strong adaptability and robustness, and can effectively identify and detect abnormal behavior in a constantly changing network traffic environment. By introducing deep learning techniques, especially temporal convolutional networks (TCNs), models can handle different types of traffic data and adapt to the dynamic changes in network traffic. Experiments have shown that even in the presence of noise or partial loss in the dataset, the model can still maintain high accuracy and low false positive rate. This adaptability enables the model to maintain good performance in the face of various network environments and attack patterns, enhancing its robustness in practical applications. In addition, through reasonable hyperparameter adjustment, the model can optimize its ability to recognize different traffic patterns, further enhancing its robustness.

In terms of anomaly detection, the model shows equally excellent performance. The experimental results show that the F1 score of the anomaly detection model based on deep learning on the test set reaches 96.3%, while the F1 score of the traditional rule-based method under the same conditions is only about 82%. The improvement of the model's detection accuracy is mainly due to its in-depth learning and understanding of complex network traffic patterns, which enables the model to effectively identify abnormal behaviors in network traffic. In addition, the false alarm rate of the model is reduced to 1.5%. Compared with the false alarm rate of traditional methods, this result further highlights the advantages of deep learning methods in anomaly detection.

Convolutional neural networks perform well in network traffic classification, but there is still room for improvement. By combining recurrent neural networks or long short-term memory networks, temporal features can be better captured; Adopting lightweight architecture or pruning techniques can help improve computational efficiency; Self supervised learning and reinforcement learning can enhance the adaptability of models to new types of attacks; By enhancing data or adjusting the loss function, the problem of imbalanced data can be solved, further improving classification performance.

The network traffic classification and anomaly detection technology based on deep learning proposed in this paper not only shows excellent performance in laboratory environments, but also shows strong adaptability and robustness in applications in multiple real network environments. Through in-depth analysis of different network structures and parameter configurations of the models, this paper optimizes an optimal model, which provides a solid technical foundation for future network traffic management and security protection. Future research can further explore the application of deep learning models in larger and more diverse network environments to continue to improve the accuracy and efficiency of classification and detection.

References

[1] Afuwape, A. A., Xu, Y., Anajemba, J. H., & Srivastava, G. (2021). Performance evaluation of secured network traffic classification using a machine learning approach. Computer Standards & Interfaces, 78, 103545. https://doi.org/10.1016/j.csi.2021.103545

[2] Bozkır, R., Cicioğlu, M., Çalhan, A., & Toğay, C. (2023). A new platform for machine-learning-based network traffic classification. Computer Communications, 208, 1-14. https://doi.org/10.1016/j.comcom.2023.05.01

[3] Gams, M., & Kolenik, T. (2021). Relations between electronics, artificial intelligence and information society through information society rules. Electronics, 10(4), 514. https://doi.org/10.3390/electronics10040514

[4] Cai, W., Hou, C., Cui, M., Wang, B., Xiong, G., & Gou, G. (2024). Incremental encrypted traffic classification via contrastive prototype networks. Computer Networks, 110591. https://doi.org/10.1016/j.comnet.2024.110591

[5] Hu, G., Xiao, X., Shen, M., Zhang, B., Yan, X., & Liu, Y. (2023). TCGNN: Packet-grained network traffic classification via Graph Neural Networks. Engineering Applications of Artificial Intelligence, 123, 106531. https://doi.org/10.1016/j.engappai.2023.10653

[6] Hu, Y., Zeng, Z., Song, J., Xu, L., & Zhou, X. (2024). Online network traffic classification based on external attention and convolution by IP packet header. Computer Networks, 252, 110656. https://doi.org/10.1016/j.comnet.2024.110656

[7] Huang, H., Lu, Y., Zhou, S., Zhang, X., & Li, Z. (2024). CoTNeT: Contextual transformer network for encrypted traffic classification. Egyptian Informatics Journal, 26, 100475. https://doi.org/10.1016/j.eij.2024.100475

[8] Hari, P., & Singh, M. P. (2025). Adaptive knowledge transfer using federated deep learning for plant disease detection. Computers and Electronics in Agriculture, 229, 109720. https://doi.org/10.1016/j.compag.2024.109720

[9] Zou, L., & Zhang, M. (2024). Variational autoencoder model combining deep learning and probability statistics: research and application. Informatica, 48(22). https://doi.org/10.31449/inf.v48i22.6921

[10] Jagatheesaperumal, S. K., Ahmad, I., Höyhtyä, M., Khan, S., & Gurtov, A. (2024). Deep learning frameworks for cognitive radio networks: Review and open research challenges. Journal of Network and Computer Applications, 104051. https://doi.org/10.1016/j.jnca.2024.104051

[11] Luo, F., Zhao, B., Fuentes, J., Zhang, X., Ding, W., Gu, C., & Pino, L. R. (2024). A review on multi-focus image fusion using deep learning. Neurocomputing, 129125. https://doi.org/10.1016/j.neucom.2024.129125

[12] Mu, G., Zhang, H., Lin, J., & Kong, F. (2025).

SMCD: Privacy-preserving deep learning based malicious code detection. Computers & Security, 150, 104226. https://doi.org/10.1016/j.cose.2024.104226

[13] Qorich, M., & El Ouazzani, R. (2025). Lightweight advanced deep-learning models for stress detection on social media. Engineering Applications of Artificial Intelligence, 140, 109720. https://doi.org/10.1016/j.engappai.2024.109720

[14] Obasi, T., & Shafiq, M. O. (2022). CARD-B: A stacked ensemble learning technique for classification of encrypted network traffic. Computer Communications, 190, 110-125. https://doi.org/10.1016/j.comcom.2022.02.006

[15] Wang, L., Ma, X., Li, N., Lv, Q., Wang, Y., Huang, W., & Chen, H. (2023). TGPrint: Attack fingerprint classification on encrypted network traffic based graph convolution attention networks. Computers & Security, 135, 103466. https://doi.org/10.1016/j.cose.2023.103466

[16] Wang, Z., Li, Z., Fu, M., Ye, Y., & Wang, P. (2024). Network traffic classification based on federated semi-supervised learning. Journal of Systems Architecture, 149, 103091. https://doi.org/10.1016/j.sysarc.2024.103091

[17] Zhang, H., & Qiu, J. (2024). A novel navigation and charging strategy for electric vehicles based on customer classification in power-traffic network. International Journal of Electrical Power & Energy Systems, 158, 109931. https://doi.org/10.1016/j.ijepes.2024.109931

[18] Zhao, J., Jing, X., Yan, Z., & Pedrycz, W. (2021). Network traffic classification for data fusion: A survey. Information Fusion, 72, 22-47. https://doi.org/10.1016/j.inffus.2021.02.009

[19] Chen, J., Chen, Y., Cai, S., Yin, S., Zhao, L., & Zhang, Z. (2023). An optimized feature extraction algorithm for abnormal network traffic detection. Future Generation Computer Systems, 149, 330-342. https://doi.org/10.1016/j.future.2023.07.039

[20] Chen, J., Lv, T., Cai, S., Song, L., & Yin, S. (2023). A novel detection model for abnormal network traffic based on bidirectional temporal convolutional network. Information and Software Technology, 157, 107166. https://doi.org/10.1016/j.infsof.2023.107166

[21] Dong, S., Su, H., & Liu, Y. (2023). A-CAVE: Network abnormal traffic detection algorithm based on variational autoencoder. ICT Express, 9(5), 896-902. https://doi.org/10.1016/j.icte.2022.11.006

[22] Guo, H., Mao, Y., He, X., Zhang, B., Pang, T., & Ping, P. (2024). Improving federated learning through abnormal client detection and incentive. CMES-Computer Modeling in Engineering & Sciences, 139(1), 383-403. https://doi.org/10.32604/cmes.2023.031466

[23] Su, T., Wang, J., Hu, W., Dong, G., & Gwanggil, J. (2024). Abnormal traffic detection for internet of things based on an improved residual network. Computers, Materials & Continua, 79(3), 4433-4448. https://doi.org/10.32604/cmc.2024.051535

[24] Wang, K., Fu, Y., Duan, X., Liu, T., & Xu, J. (2024). Abnormal traffic detection system in SDN based on deep learning hybrid models. Computer Communications, 216, 183-194. https://doi.org/10.1016/j.comcom.2023.12.041

[25] Wang, W. (2024). Abnormal traffic detection for internet of things based on an improved residual network. Physical Communication, 102406. https://doi.org/10.1016/j.phycom.2024.102406

[26] Wang, Z., Ni, A., Tian, Z., Wang, Z., & Gong, Y. (2024). Research on blockchain abnormal transaction detection technology combining CNN and transformer structure. Computers and Electrical Engineering, 116, 109194. https://doi.org/10.1016/j.compeleceng.2024.109194

[27] Zheng, L., Zhang, J., Wang, X., Lin, F., & Meng, Z. (2024). Multimodal-based abnormal behavior detection method in virtualization environment. Computers & Security, 143, 103908. https://doi.org/10.1016/j.cose.2024.103908