Frontiers of Information Technology & Electronic Engineering
www.jzus.zju.edu.cn; engineering.cae.cn; www.springerlink.com
ISSN 2095-9184 (print); ISSN 2095-9230 (online)
E-mail: jzus@zju.edu.cn

CRGT-SA: an interlaced and spatiotemporal deep learning model for network intrusion detection

Jue CHEN, Wanxiao LIU, Xihe QIU[†‡], Wenjing LV, Yujie XIONG
School of Electronic and Electric Engineering,
Shanghai University of Engineering Science, Shanghai 310027, China
[†]E-mail: qiuxihe@sues.edu.cn

Received May 30, 2024; Revision accepted Sept. 18, 2024; Crosschecked

Abstract: To meet the challenge of widely existing and frequently changing network attacks, intrusion detection systems (IDSs) are introduced to recognize intrusions and to protect computer networks. Among all these IDSs, conventional machine learning methods rely on shallow learning and have unsatisfactory performance. Unlike machine learning methods, deep learning methods are the mainstream methods because of their capability to handle mass data without prior knowledge of specific domain expertise. Concerning deep learning, long short-term memory (LSTM) and temporal convolutional networks (TCNs) can be used to extract temporal features from different angles, while convolutional neural networks (CNNs) are valuable for learning spatial properties. Based on the above, this paper proposes a novel interlaced and spatiotemporal deep learning model called Cnn Rnn Gated Tcn-self attention (CRGT-SA), which combines the CNN with Gated TCN and RNN (LSTM) modules to learn spatiotemporal properties, and imports the self-attention mechanism to select significant features. More specifically, our proposed model splits the feature extraction into multiple steps with a gradually increasing granularity, and executes each step with combined CNN, LSTM, and Gated TCN modules. Our proposed CRGT-SA model is validated using the UNSW-NB15 data set and compared with other compelling techniques, including traditional machine learning and deep learning models as well as state-of-the-art deep learning models. According to the simulation results, our proposed model exhibits the highest accuracy and F1-score among all the compared methods. More specifically, our proposed model achieves 91.5% and 90.5% accuracy for binary and multi-class classifications, respectively, and demonstrates its ability to protect the Internet from complicated network attacks. Moreover, we conducted another series of experiments on the NSL-KDD data set; the simulation results of comparison with other models further prove the generalization ability of our proposed model.

Key words: Intrusion detection; Deep Learning; convolutional neural network; Long short-term memory; Temporal convolutional network
https://doi.org/10.1631/FITEE.2400459 **CLC number:**

1 Introduction

A latest report from Juniper Research states that there will be 84 billion network connections in 2024 (Oseni et al., 2023). Meanwhile, network attacks are constantly evolving and pose a significant threat to a wide variety of cutting-edge technologies, such as smart hospitals, power, medical, and campuses (Lv et al., 2021). For example, attacks against critical infrastructures used for power generation can lead to loss of property or even personal safety (Cook et al., 2020).

Intrusion detection systems (IDSs) are responsible for identifying intrusions that evade security techniques, and providing a vital second level of resistance to protect computer networks (Fang et al., 2021). Multiple recent IDS studies adopt machine

[‡] Corresponding author

learning and deep learning methods to solve security issues (Al-Garadi et al., 2020; Chen et al., 2022). Conventional machine learning methods include logistic regression (LR), Gaussian Naive Bayes (GNB), K-nearest neighbors (KNN), adaptive boosting (AdaB), and random forest (RF), and almost all the methods mentioned above use shallow learning which relies on manual feature engineering to extract features (Vasilomanolakis et al., 2015). Because of the huge amount of data volume, shallow learning is unable to solve real-time problems (Laghrissi et al., 2021). As a result, these methods achieve unsatisfactory performance in identifying different types of cyber attacks (Wang Ket al., 2023).

Unlike machine learning methods, deep learning methods have become the most dominant roles in the field of intrusion detection, because of its ability to deal with mass data without prior knowledge on specific domain expertise. For instance, convolutional neural networks (CNNs), long short-term memory (LSTM) and temporal convolutional networks (TCNs), as standard deep learning technologies, can deal with network intrusions with different degrees of difficulty, complexity, and distributivity (Wang XFet al., 2020). To achieve high accuracy in detecting and classifying different types of cyber attacks, this study proposes a new network intrusion detection approach with a hybrid deep learning model.

Among deep learning technologies, the recurrent neural network (RNN) is a kind of dynamic and feed-forward neural network, and is capable of learning sequential data over timesteps (Aldweesh et al., 2020). As an enhanced version of RNN, LSTM can use a gating mechanism to learn long-term dependencies. The TCN can be regarded as an alternative to the RNN with the advantage of processing inputs in parallel as well as extracting high-dimensional abstract features from raw data (Fenghour et al., 2021). As a result, we combine LSTM with a Gated TCN to make full use of time series prediction of LSTM combined with the feature extraction and fusion of the TCN, which can finally improve the performance of processing time series data.

On the other hand, the CNN comprises convolutional layers, pooling layers, and (optional) fully connected layers. Among these layers, convolutional layers contain filters to extract features. The pooling layer then selects features from the convolutional layer through sub-sampling (Aldweesh et al., 2020). In general, the CNN fits multi-dimensional data well when extracting spatial features. Moreover, when the input of the neural network is multiple vectors of different sizes that are related to each other, the self-attention mechanism can be used to make the IDS notice relationships between different parts of the input, and then select more important features.

In summary, this paper proposes a novel and hybrid deep learning model called Cnn Rnn Gated Tcn-self attention (CRGT-SA) to achieve network intrusion detection. The proposed model uses a CNN to learn spatial features, combines Gated TCN with RNN to extract temporal characteristics, interlaces them to generate the integrated features, and imports self-attention to select more important features. The main contributions of this paper are summarized as follows:

1. A novel IDS model that combines CNN, LSTM, with Gated TCN is proposed to improve the capability of learning spatiotemporal characteristics from network traffic in a hierarchical manner.More specifically, our proposed model splits the feature extraction into multiple steps with a gradually increasing granularity, and executes each step through a combined CNN, LSTM and Gated TCN modules.

2. A self-attention mechanism is imported for calculating the weight which reflects the importance of each feature, and makes the neural network focus on the most important features, which finally improves the IDS performance.

3. A series of experiments are conducted on the UNSW-NB15 data sets. The simulation results verify that our proposed model achieves superior performance on intrusion detection compared with other traditional machine learning and state-of-the-art deep learning models. More specifically, our proposed model achieves accuracy of 91.5% and 90.5% in the binary and multi-class classifications, respectively. To further prove the generalization ability of our proposed model, we conducted another series of experiments on the NSL-KDD data set, the simulation results show that our proposed model still achieves the best performance of all the compared models no matter in binary or multi-class classifications.

2 Related works

Artificial intelligence approaches have continuously been applied to developing reliable IDSs. For example, Francois et al. proposed a semi-supervised ensemble approach based on random partitioning binary trees for intrusion detection. Moreover, detection when facing collective anomalies was improved through taking into account the relative frequency of visits to the leaves of the trees (Marteau, 2021). To enhance the performance of single-learners, Ghada et al. built an ensemble learning model which is composed of principal component analysis (PCA), a Support Vector Machine and a neural network to detect attacks on the internet of things (IoT) (Abdelmoumin et al., 2022). Ali et al. presented an IDS based on soft voting to select optimal supervised classifiers to maximize accuracy and minimize false alarm rates. Furthermore, they adopted different sampling methods to solve the data imbalance problem (Khan et al., 2023). Qi et al.(2022) merged local sensitive hash (LSH), isolated forest and PCA together to efficiently detect attacks in Industry 4.0; these components operate on multi-aspect data, catch group anomalies, and reduce dimensionality for correlations between different attributes. However, these classical machine learning methods can only learn shallow features, which limits learning ability and detection accuracy.

Unlike traditional machine learning methods, deep learning has received extensive attention in the domain of intrusion detection because of its powerful learning ability and independence from feature engineering, which can further improve the accuracy. For instance, Diro et al. proposed a distributed deep learning-based attack detection architecture for fog computing in IoT, which exchanged parameters and models between worker and master nodes. In terms of the deep learning models, the stacked autoencoder and softmax regression are used for feature engineering and attack classification, respectively (Abeshu and Chilamkurti, 2018). Kumar et al.(2022) integrated blockchain with a deep learning technique to realize the privacy-preserving intrusion detection in the Internet of Vehicles (IoV). More specifically, the blockchain and LSTM modules are used to transmit data securely and identify cyber-attacks. Ayodeji et al. designed a CNN-based IDS for IOV and employed the SHapley Additive exPlanations (SHAP) mechanism to interpret how a feature value increases or decreases a model's prediction (Oseni et al., 2023). Nie et al.(2022) constructed an intrusion detection model based on a generative adversarial network (GAN) to detect a single attack and multiple attacks for secure social IoT . Taking the data and concept drifts into consideration, Wahab et al. presented drift detection and outlier detection methods to study the change in the variances of the features over time and identify the outliers that diverge both from historical and temporally close data points, respectively. To counter drifts, this paper discussed an online deep neural network (DNN)-based model to adjust the sizes of hidden layers and to realize intrusion detection dynamically (Wahab, 2022). FatimaEzzahra et al. first used PCA and mutual information (MI) for dimensionality reduction and feature selection, respectively. On that basis, they implemented a LSTM-based model to realize intrusion detection (Laghrissi et al., 2021). Because of the small number of labeled IoT traffic records, Abdel-Basset et al. introduced a hierarchical semi-supervised training model to realize intrusion detection. Moreover, this paper imported a multi-scale residual temporal convolutional model and an improved attention mechanism to fine-tune the network capability in learning spatiotemporal representations and estimate the importance score of different features, respectively (Abdel-Basset et al., 2021). Vinayakumar et al. established a hybrid network IDS using a DNN with five hidden layers that can handle mass data in real time. To prove the universality of the DNN-based model, contrast experiments were conducted on multiple public datasets including KDDCup 99, NSL-KDD, UNSW-NB15, Kyoto, WSN-DS, and CICIDS 2017 (Vinayakumar et al., 2019).

On the basis of machine learning and deep learning methods, multiple hybrid learning models have been proposed for intrusion detection that combined multiple machine learning or deep learning algorithms to make full use of each one's advantage. Mehedi et al. combined deep CNN with LSTM to create an effective network intrusion detection approach for big data scenarios, in which CNN and weight-dropped LSTM are used for extracting meaningful features and retaining long-term dependencies, respectively (Hassan et al., 2020). Ashfaq et al. built a convolutional RNN-based IDS to predict and classify malicious cyber-attacks. In this system, the CNN and RNN modules are responsible for capturing local features and temporal features, respectively (Khan, 2021). Cao et al.(2022) designed a network intrusion detection approach merging a CNN with GRUs to solve the issue of low classification accuracy. After solving the data imbalance and realizing the feature selection, the CNN and GRU modules were used to extract spatial features and long-distance dependent information features, respectively . Mambwe et al. adopted multiple different RNN types, including LSTM, GRU, and RNN to compose a network intrusion detection framework and compare the performance of these RNN models. Moreover, an XGBoost-based algorithm was implemented for feature selection to reduce the feature space of data sets (Kasongo, 2023). Jian et al.(2019) presented a multi-path IDS composed of two models: coupled data embedding (CDE) and coupled outlier scoring of high-dimensional data (COSH) for clustering and outlier detection, respectively .

Even though existing related works have combined CNN with RNN modules for intrusion detection, most of them simply structure these networks in tandem, leading to the loss of temporal or spatial information. Unlike the methods mentioned above, our proposed model divides the feature extraction into multiple steps with gradually increasing granularity, and performs each step using a combined CNN, Gated TCN, and LSTM block, which can maintain the spatiotemporal characteristics of network traffic effectively. Moreover, this paper introduces the self-attention mechanism to select the most significant features.

3 Proposed model

3.1 Overview

As mentioned before, most existing related works that combine CNN with RNN modules for intrusion detection simply structure them in tandem. For instance, in the hierarchical spatial-temporal features-based IDS (HAST-IDS) architecture (Wang Wet al., 2018) (as shown in Fig. 1), only after the CNN block has learned low-level spatial features from network traffic, will it be handed over to the RNN module for subsequent processing. It can be inferred that a loss of temporal information may exist due to the CNN module, which degrades the performance of the following RNN module and the whole system.

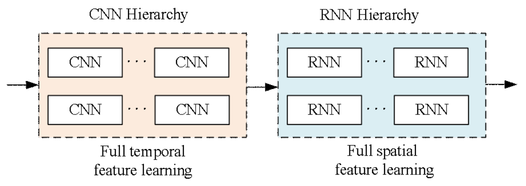


Fig. 1 Diagram of the HAST-IDS model. HAST-IDS, hierarchical spatial-temporal features-based IDS; IDS; intrusion detection system

In consideration of the above issues, this paper proposes a novel and hybrid deep learning model called CRGT-SA, which integrates CNN, RNN, Gated TCN and self-attention into an entirety, as shown in Fig. 2. Instead of allowing CNN to achieve full learning, our proposed model intertwines the CNN with Gated TCN and RNN (LSTM) modules, that is, feature extraction is divided into multiple steps with gradually increasing granularity, and each step is executed through a combined CNN, Gated TCN, and LSTM block. Moreover, the self-attention mechanism is introduced to measure the importance of the input features, so the network will emphasize the most significant features.

As can be seen from the dashed lines labeled “increase granularity”, the learning begins with coarse-grained learning, hence the output of the CNN module will still reserve the temporal information for the following Gated TCN and LSTM modules. The learning granularity becomes more fine-grained as the learning continues. However, at each level, CNN, Gated TCN, and LSTM learn spatiotemporal characteristics at the same granularity. Consequently, these modules can learn to their full extent independently. Moreover, the batch normalization, dimension reshaping, and dropout are useful for handling covariance shift, data reshaping, and over-fitting, respectively.

3.2 Batch normalization

When the DNN is adopted, the range of input value varies dynamically from layer to layer in the training stage, which makes learning efficiency be highly dependent and leads to unstable learning outcomes. To make the sample data more stable and accelerate the convergence rate of the DNN, batch normalization is adopted to modulate the output of

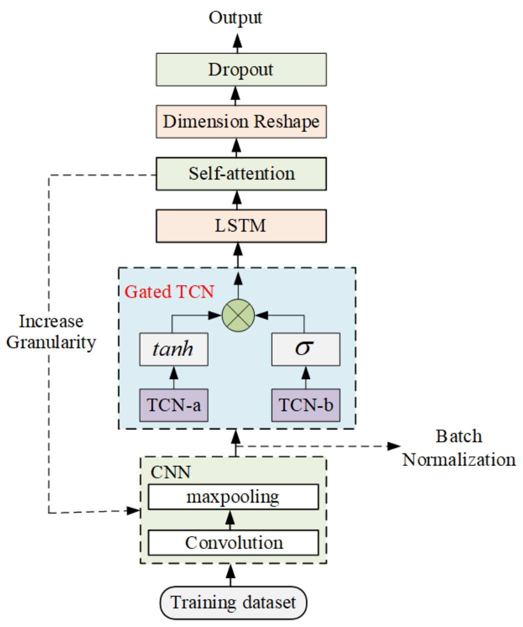


Fig. 2 Diagram of the CRGT-SA model. CRGT-SA, CNN RNN Gated Tcn-self attention

the CNN to satisfy the requirement of the Gated TCN requirement in our proposed model, and the corresponding formula is shown as follows:

$$\hat{x} = \frac{x - \mu_B}{\sqrt{\delta_B^2 + \varepsilon}}, \tag{1}$$

where x is data, and μ_B and δ_B correspond to the batch mean and batch standard deviation, respectively. Note that ε is used to ensure a nonzero denominator, which can be ignored. Based on $\hat{x}$, the normalization generates the output $\hat{y}$ through the formula:

$$\hat{y} = \gamma\hat{x} + \beta, \tag{2}$$

where both γ and β are trained for a better learning effect.

3.3 Gated temporal convolutional network (TCN)

The TCN can be used to solve a time series prediction problem, which is composed of a 1D convolution layer with the same input and output lengths. Fig. 3 shows a diagram of the dilated casual convolution with the kernel size to be 2. The inputs are selected every d steps and a standard 1D convolution is applied to the selected inputs. Given a one-dimensional sequence of inputs $x \in R^{\mathrm{T}}$ and filter $f \in R^{\mathrm{K}}$, the representation of the dilated casual convolution operation of x with f at step t is shown

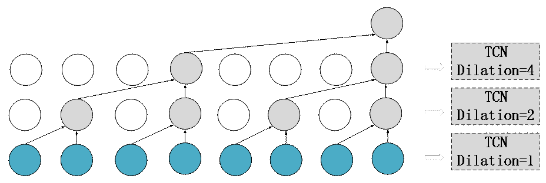


Fig. 3 Dilated causal convolution

in the following formula:

$$x * f\left(\mathrm{t}\right) = \sum_{s=0}^{\mathrm{K}-1} f\left(s\right) x\left(\mathrm{t} - d \times s\right), \tag{3}$$

where d is the dilation factor. As the name suggests, the dilated causal convolution is a combination of dilated convolution and causal convolution. In dilated convolution, interval sampling is allowed, whereas in causal convolution, data at time t of layer i only depends on the effect of time t of layer $i-1$ and previous values, which is a strict time-constrained model that discards the influence of future data during training. Moreover, the residual network is adopted to transmit data across layers and ensure the consistency of input and output.

On that basis, we import the Gated TCN in this paper to extract complicated temporal features, where only an output gate is involved in our proposed model. Given the input $X \in R^{\mathrm{N\times D\times S}}$, it takes the following formula:

$$h = g\left(\theta_1 * X + b\right) \odot \sigma\left(\theta_2 * X + c\right), \tag{4}$$

where θ_1, θ_2, b, and c are model parameters, $\odot$ is the element-wise product, $g(\cdot)$ is an activation function of the outputs, and $\theta(\cdot)$ is the sigmoid function.

3.4 Convolutional neural network (CNN)

A CNN can handle dense connections between layers of deep neural networks, and consists of convolutional layers, pooling layers, and optional fully connected layers. Convolutional layers directly receive multi-dimensional inputs, and convolve between the inputs to generate feature maps. Pooling layers perform sub-sampling for feature maps to reduce the dimensionality (Gan et al., 2022). Because a packet is stored in a one-dimensional format, we use the following formula to process the input vector g with a filter f of size m:

$$\left(f * g\right)\left(i\right) = \sum_{j=1}^{m} g\left(i\right) \cdot f\left(i - j + m/2\right). \tag{5}$$

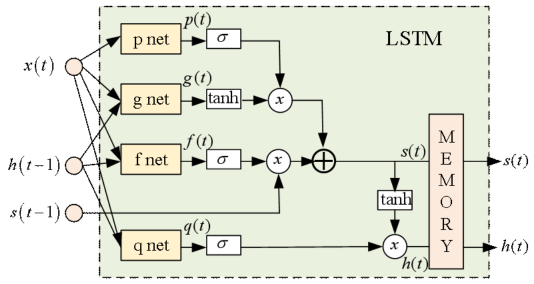


Fig. 4 Diagram of LSTM. LSTM; long short-term memory

Moreover, the rectified linear unit (ReLU) is chosen as the activation function in consideration of its high speed convergence.

$$f(z) = \max(0, z). \tag{6}$$

3.5 Long short-term memory (LSTM)

LSTM belongs to the gated RNN, which controls the feedback with multiple gate functions to reserve long-term instead of short-term dependencies. Fig. 4 shows a diagram of LSTM, which is composed of four connected subnetworks (represented as p-net, g-net, f-net, and q-net), multiple control gates, and a memory component.

All the subnetworks have a unified structure, as shown in the following formula:

$$b + U \times x\left(t\right) + W \times h\left(t-1\right), \tag{7}$$

where $x(t)$, $h(t-1)$, and b are current input, previous output and bias, respectively; $\boldsymbol{U}$ and $\boldsymbol{W}$ are the weight matrix for the current input and recurrent weight matrix for the previous output, respectively. Note that four subnetworks are different in b, $\boldsymbol{U}$, and $\boldsymbol{W}$. The outputs of four subnetworks are executed by two types of controlling gates, i.e., σ and tanh, to calculate the feedback $s(t)$ from the previous learning and the current output $h(t)$:

$$s\left(t\right) = \sigma\left(f\left(t\right)\right) * s\left(t-1\right) + \sigma\left(p\left(t\right)\right) * \tanh g\left(t\right), \tag{8}$$

$$h\left(t\right) = \tanh s\left(t\right) * \sigma\left(q\left(t\right)\right). \tag{9}$$

3.6 Self-attention

The function of the self-attention mechanism is to enable the model to learn to assign weights for input signals on its own, i.e., the different dimensions of the input signal are scored and features are weighted according to scores, which can emphasize the influence of significant features. As shown in Fig. 5, the self-attention mechanism can generate weights for different connections dynamically, which can be used as a layer in a neural network, as specified in the following formula:

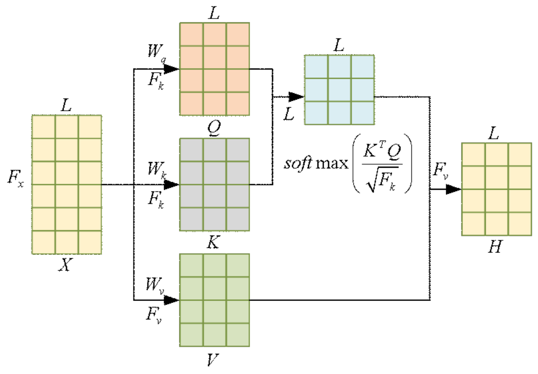


Fig. 5 Diagram of the self attention mechanism

$$\text{Attention}(\text{Q}, \text{K}, \text{V}) = \text{soft}\max\frac{\text{K}^{\text{T}}\text{Q}}{\sqrt{F_k}}\text{V}, \tag{10}$$

where Q, K, and V represent the matrices of query vectors, key vectors, and value vectors, respectively, and $\sqrt{F_k}$ is the vector length.

3.7 Output

On one hand, due to the variation in learning granularity among different steps, the possibility in which the output size of one step (each step is performed by a combined CNN, Gated TCN, and LSTM block) does not match the anticipated input size of the next step exists. Therefore, the dimension reshape layer is added to adjust the data for the following module. On the other hand, a typical problem of learning mass data with a DNN is overfitting, where the network learns too well from the training data, which limits the capability of recognizing variables from new data samples. In this situation, the dropout layer is added to randomly remove multiple connections from the DNN to reduce the possibility of overfitting problem.

4 Experimental setup and evaluations

4.1 Data set

The evaluation of an intrusion detection model is strongly associated with the chosen data set. Multiple data sets for intrusion detection contain a large amount of superfluous data, making experimental results untrustworthy (Zhuo et al., 2017). To ensure the effectiveness of experiments in our study, we choose the UNSW-NB15 data set (Moustafa and Slay, 2016) without any redundancy, which was produced by the australian center for cyber security (ACCS) in 2015. The data samples were first collected from the Internet, and then simulated in a lab to generate the data set. There are 9 UNSW-NB15 attack types: denial of service (DoS), Exploits, Generic, Shellcode, Reconnaissance, Backdoor, Worms, Analysis, and Fuzzers, and the corresponding proportions are 6.35%, 17.28%, 22.85%, 0.59%, 5.43%, 0.90%, 0.07%, 1.14%, and 9.41%, respectively. The remaining samples belong to normal traffic. Table 1 illustrates the specific description of each attack type.

4.2 Data preprocessing

Before inputting data to our proposed model, the raw data should be cleaned, labeled, and annotated first. More specifically, the data preprocessing stage contains three steps: conversion, standardization, and cross validation.

1. Conversion: To make the experiments more effective, we need data to agree with the input format required by the neural network. The original network traffic data contains classification features in the form of text information, which cannot be directly processed by our proposed model. As a result, the text information needs to be transformed to numerical values, which can be implemented through the use of “get dummies” in Pandas.

2. Standardization: The means and standard derivations of input data may differ, which can lead to inefficient learning. As a result, we zoom the input data to ensure that the means and standard derivations are 0 and 1, respectively.

3. Cross validation: UNSW-NB15 contains $2,540,044$ samples, and we employ the Stratified K-Fold Cross Validation strategy to split all these samples into k groups, with $k-1$ groups and one group for training and validating, respectively.

4.3 Experimental setup

For the sake of proving the effectiveness of our proposed CRGT-SA model, we implement this model with PyTorch, Keras, and Scikit-Learn packages, and run the CPU @ 3.20 GHz and 16.0 GB RAM on the HP EliteDesk 800 G2 SFF desktop equipped with Intel (R) Core (TM) i5-6500. Table 2 displays a summary of our proposed model. Moreover, RMSprop is applied to optimize the weight and bias when training our proposed model, with the learning and dropout rates set as 0.001 and 0.5, respectively. The pseudocode of the training procedure is shown in Algorithm 1.

Algorithm 1 Training Procedure of CRGT-SA

```
Require:
    Training data set, learning rate, training epochs,
    batch size, testing data set.
Ensure:
    Classification results of testing data sets.
 1: Preprocess training data set, including the completion of missing values,
    the label coding of discrete features, the matrix of reshaping the input vector;
 2: for $i$ in number of training epochs do
 3:   for $j$ from 1 to $n$ do
 4:     Start: $K = N/\text{Batch}$;
 5:     Split training data set into $K$-groups;
 6:     Load proposed model;
 7:     Fit model with $K-1$ group;
 8:     Validate model with remaining $\text{K}^{\text{th}}$ group;
 9:   end for
10: end for
11: Test model on testing data set;
```

4.4 Evaluation metrics

To measure the performance of the CRGT-SA model, and to compare with other machine learning and deep learning models, we need to compute the accuracy, precision, recall and F1-score for each model, which are explained and derived below.

1. Accuracy: this corresponds to the ratio of the number of correctly identified traffic records to the total number of traffic records.

$$\text{ACC} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}. \tag{11}$$

2. Precision: this corresponds to the ratio of the number of correctly detected network attacks to the total number of traffic records identified as network

Table 1 Description of attack types in the UNSW-NB15 data set

Attack type	Description
Denial of Service (DoS)	Disguise as real hosts to access resources
Exploits	Control resources through vulnerabilities
Generic	Hash collision based on block cipher
Shellcode	Exploit vulnerabilities in software to execute code
Reconnaissance	Collect information illegally
Backdoor	Bypass security mechanisms to access data or control resources illegally
Worms	Spread through media
Analysis	Infiltrate web programs through scripts
Fuzzers	Input unexpected data and observe the output to find the vulnerabilities

Table 2 Summary of the CRGT-SA model

Layer	Name	Type	Number of parameters
1	CNN	CRGT-SA-Block	254.6K
	Batch normalization		
	TCN		
	LSTM		
	Self-Attention		
2	CNN	CRGT-Block	1.3M
	Batch normalization		
	TCN		
	LSTM		
	Self-Attention		
3	CNN	CRGT-Block	9.0M
	Batch normalization		
	TCN		
	LSTM		
	Self-Attention		
4	conv	Sequential	262.0K
5	avg_pool	AvgPool1d	0
6	drop_out	Dropout	0
7	out	Sequential	65.0K
Trainable params			10.3M
Total params			10.3M
Total estimated model params size			41.249M

attacks.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}. \tag{12}$$

3. Recall: This corresponds to the ratio of the number of correctly detected network attacks to the total number of network attacks.

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}. \tag{13}$$

4. F1-score: This corresponds to the harmonic mean of the precision and recall.

$$\text{F1} - \text{score} = 2 \times \left(\frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}\right). \tag{14}$$

In these formulas, TP is the number of network attacks correctly detected as intrusions, TN is the number of normal traffic records correctly classified, FP is the number of normal traffic records wrongly classified as network attacks, and FN is the number of network attacks wrongly classified as normal traffic.

4.5 Baseline methods

To demonstrate the validity of the CRGT-SA model, we implemented a set of the most advanced machine learning and deep learning models for comparison. Machine learning models include LR, GNB, KNN, AdaB and RF, which are introduced as follows in brief:

1. Logistic regression (LR): this model estimates the probability of an event according to a given data set with independent variables. Considering that the result is a probability, then the dependent variables are in the range of 0 to 1.

2. Gaussian naive bayes (GNB): this model assumes that the conditional probability of each feature dimension is subject to Gaussian distribution, calculates the posterior probability for the new sample under a certain feature distribution according to the Bayes formula, and finally determines the category of the sample by maximizing the posterior probability.

3. K-Nearest neighbors (KNN): to find the category for a new input instance with a given training data set, the model firstly finds K instances closest to the target instance, and then classifies the target instance into the class to which most of these K instances belong.

4. AdaB: this model is an iterative algorithm that trains different classifiers on a unified training data set, and then combines these classifiers into a stronger classifier to generate final results.

5. Decision tree (DT): this model is a method of approximating to discrete function values. The model processes the data by inductive algorithm, generates rules and DTs, and finally analyzes the new data according to the decision. In essence, a DT uses multiple rules to classify data.

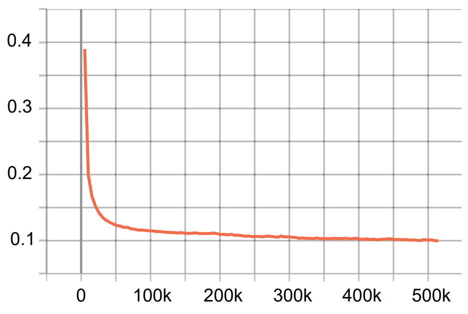
6. Random forest (RF): this model is a supervised learning method that can summarize rules from a series of characterized and labeled data and represent these rules through the structure by a tree diagram. In the tree, each internal node, branch, and leaf node correspond to a judgment on an attribute, an output of a judgment result, and a classification result, respectively.

5 Experimental results and analysis

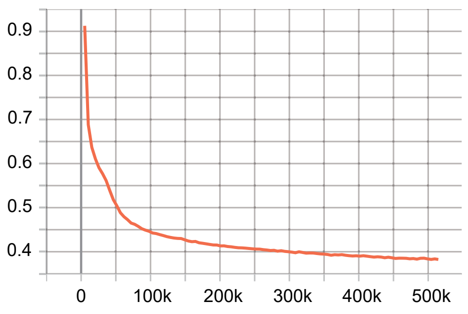
In this section, we describe our implementation of our proposed CRGT-SA model, and measure the corresponding loss trend and confusion matrix. Moreover, we conduct a comparative experiment and an ablation experiment against recent standard and state-of-the-art methods to prove the effectiveness of our proposed model.

5.1 Loss trend

The loss curves over the number of steps for our proposed CRGT-SA model is depicted in Fig. 6, and the relationship between steps and epochs is shown in Fig. 7. As shown in Fig. 6, the left and right subfigures depict the loss curves for binary and multi-class classifications, respectively. When the number of steps reaches 550, the loss tends to converge. As shown in Fig. 7, the number of steps is in linear relation to the number of epochs, and 550 steps corresponds to 100 epochs. Therefore, we set the number of epochs to be 100 in the following experiments. More specifically, in terms of the binary classification, the difference between the model's predicted value and the true value stabilizes at 0.1, while in terms of the multi-class classification, the difference stabilizes below 0.4. It can be explained because the latter is more difficult than the former,; especially, the sample size of several cyberattacks is too small to detect accurately in the multi-class classification.



(a)



(b)

Fig. 6 Loss curve on binary classification (a) and multi-class classification (b)

5.2 Comparative analysis with traditional machine learning models

In this subsection, we compare the performance of our proposed CRGT-SA model against traditional machine learning models, and the comparative results of the binary and multi-class classifications are shown in Tables 3 and 4, respectively. As shown in Table 3, the accuracy of traditional machine learning algorithms is between 71.6% and 87.7%, and the F1-score is between 79.2% and 91.2%. Among these algorithms, LR and GNB have poor effects, the accuracy and F1-socre of KNN and AdaB are acceptable, and RF has the best performance. It can be inferred that the ensemble learning involved in RF makes it more tolerant to noise and outliers. In contrast, the accuracy and F1-score of the CRGT-SA model are 91.5% and 91.6%, respectively. In other words, the accuracy and F1-score of our proposed model are better by at least 3.8% and 0.4%, respectively, which indicates that this model achieves the best performance on binary classification among all the models.

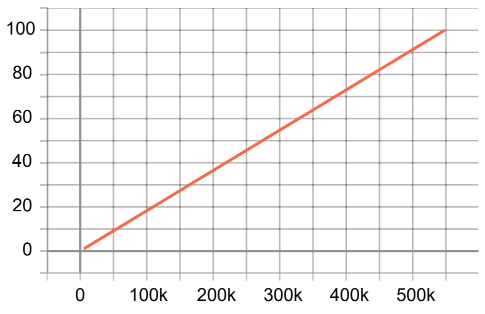


Fig. 7 Relationship between epochs and steps

Table 3 Comparative analysis with traditional machine learning models of binary classification

Methods	Accuracy	F1-score
LR	0.753	0.792
GNB	0.716	0.818
KNN	0.829	0.869
AdaB	0.839	0.884
RF	0.877	0.912
CRGT-SA	**0.915**	**0.916**

Table 4 Comparative analysis with traditional machine learning models of multi-class classification

Methods	Accuracy	F1-score
LR	0.561	0.428
GNB	0.085	0.13
KNN	0.652	0.638
AdaB	0.631	0.557
RF	0.736	0.695
CRGT-SA	**0.905**	**0.743**

As shown in Table 4, the accuracy of traditional machine learning algorithms is between 8.5% and 73.6%, and the F1-score is between 13% and 79.5%. When compared to the binary classification, the accuracy and F1-score of multi-class classification obviously decrease. Among these algorithms, GNB corresponds to the poorest performance, which may be attributable to the fact that the GNB assumes that features are independent of each other. When the number of features is large or features are in correlation with each other, the classification effect of the GNB may not perform well. Because this study does not adopt the feature dimension reduction, and features may be relative to each other in the UNSW-NB15 data set, then the GNB obviously lags behind other algorithms. In contrast, the accuracy and F1-score of the CRGT-SA model are 90.5% and 74.3%, respectively. In other words, the accuracy and F1-score of our proposed model are better by at least 16.9% and 4.8%, respectively, indicating that this model achieves the best performance on multi-class classification among all the models.

In addition, it can be observed that the difference between the accuracy and F1-score of our proposed CRGT-SA model in the multi-class classification is 16.2%, which cannot be ignored. The F1-score is a harmonic average of precision and recall, which indicates that this metric is more sensitive to class imbalances. In the chosen UNSW-NB15 data set, as the Backdoor, Analysis, and Worm attacks make up only 1.41%, 1.63%, and 0.11% of all the abnormal traffic, it is difficult for models to detect these three types of cyberattacks. As a result, the poor performance on minority classes results in a reduction of F1-score, which should be improved in our future works.

5.3 Comparative analysis with state-of-the-art deep learning models

To further prove the superiority of our proposed CRGT-SA model, we compare it with state-of-the-art deep learning models, including HAST (Wang Wet al., 2018), LuNet (Wu and Guo, 2019), and MSCNN-LSTM (Zhang et al., 2020), because all these models (including our proposed model) utilize the CNN and RNN to learn spatial and temporal features, respectively. More specifically, both the HAST and MSCNN-LSTM combine CNN with RNN in tandem. By contrast, LuNet intertwines the CNN and RNN modules, and is used as a reference of our proposed model. The simulation results are shown in Table 5, where it can be observed that the HAST model corresponds to the poorest performance, the performances of the LuNet and MSCNN-LSTM are acceptable, and the CRGT-SA has the best performance; and the reason analysis is summarized as follows:

5.3.1 Comparison of HAST with MSCNN-LSTM

Although both two models stack all LSTM layers after CNN layers, they differ in the convolutional kernels of each layer. The HAST model always adopts the same scale, while the MSCNN-LSTM utilizes the multi-scale convolutions effectively. As a result, the MSCNN-LSTM is better than the HAST model in the aspect of accuracy and F1-score.

5.3.2 Comparison of HAST, MSCNN-LSTM with LuNet, CRGT-SA

The architecture of the HAST and MSCNN-LSTM models may cause the CNN layers to drop out the temporal information, leading to ineffective learning for LSTM layers. However, the LuNet and CRGT-SA intertwine CNN with LSTM layers to capture both spatial and temporal features sufficiently. As a result, the LuNet and CRGT-SA are better than the HAST and MSCNN-LSTM due to dividing the feature extraction into multiple steps and the combined CNN+RNN block in each step.

5.3.3 Comparison of LuNet with CRGT-SA

On the basis of LuNet, our proposed CRGT-SA integrates CNN, RNN, Gated TCN, and self-attention into a single model. Similarly, our proposed model can retain spatiotemporal properties at each step to ensure that all layers learn to full capacity. To better capture the temporal information, our proposed model combines Gated TCN with LSTM modules to extract features from different angles, which can exploit the advantages of both. Moreover, our proposed model imports the self-attention mechanism to select significant features from network traffic data to help recognize network attacks. As a result, our proposed model improves the accuracy by 10.3% when compared with LuNet. However, our proposed model decreases the F1-score by 2.5% when compared with LuNet. The reason can be explained as follows, Because the architecture of our proposed model is more complicated than LuNet, with the Gated TCN and self-attention mechanism involved, then the probability of over-fitting increases. Overfitting means that the model performs well on training data, but poorly on previously unseen test data. To reduce the risk of over-fitting, larger data sets are required. However, because minor classes exist in the UNSW-NB15 data set, then the overfitting may be unavoidable. As a result, the severity of overfitting for our proposed CRGT-SA model may be larger than LuNet, leading to poorer performance on the F1-score of our proposed model, although the difference is still acceptable.

Table 5 Comparative analysis with state-of-the-art deep learning models

Method	Accuracy	F1-score
HAST	0.814	0.530
LuNet	0.802	0.768
MSCNN-LSTM	0.898	0.668
CRGT-SA	**0.905**	**0.743**

5.4 Ablation studies

In this subsection, ablation experiments are executed to analyze the contributions of different components of our proposed CRGT-SA model for recognizing different types of network attacks. To investigate the function of components involved in our proposed CRGT-SA model, which contains CNN, TCN, LSTM blocks, and a self-attention mechanism, we compare it with CNN, CNN-TCN, CNN-LSTM, and CNN-TCN-LSTM models. We evaluate these models, and calculate the corresponding accuracies and F1-scores for binary and multi-class classifications as shown in Tables 6 and 7, respectively.

Among all the models, our proposed CRGT-SA model achieves the best performance on accuracy in both binary and multi-class classifications. This can be explained because our proposed model is composed of multiple interlaces modules, with each module learning one type of feature. In general, a more complicated model typically has stronger representation and can learn more complex patterns and functions, which is useful for capturing subtle data set differences, leading to an improvement in accuracy. In terms of the F1-score, our proposed model is 2.5% lower than the CNN-LSTM model. The reason was explained in the previous subsection, i.e., minor classes of the data set results in overfitting. However, the difference between two models on the F1-score is still acceptable.

To explore the function of the self-attention mechanism, we compared the performance of our proposed CRGT-SA model with and without the self-attention module and recorded the achieved performance. It can be observed that inclusion of the self-attention module improves the performance by 8.3% on accuracy in binary classification, and improves the performance by 15.7% and 4.6% on accuracy and F1-score in multi-class classification, respectively. This explains the importance of the self-attention mechanism in selecting significant features.

Table 6 Ablation studies of binary classification

Method	Accuracy	F1-score
CNN	0.841	0.885
CNN-TCN	0.821	0.895
CNN-LSTM	0.848	0.903
CNN-TCN-LSTM	0.832	0.916
CRGT-SA	**0.915**	**0.916**

Table 7 Ablation studies of multi-class classification

Method	Accuracy	F1-score
CNN	0.772	0.732
CNN-TCN	0.761	0.694
CNN-LSTM	0.802	0.768
CNN-TCN-LSTM	0.748	0.697
CRGT-SA	**0.905**	**0.743**

5.5 Confusion matrix

The confusion matrices of using our proposed CRGT-SA model used on the UNSW-NB15 data set for binary and multi-class classifications are shown in Figs 8 and 9, respectively. As shown in Fig. 8, our proposed model can correctly detect most normal and abnormal traffic, specifically, 99.34% of normal traffic is correctly classified. As shown in Fig. 9, our proposed model can correctly detect most of the normal traffic, as well as Reconnaissance, DoS, Exploits, Fuzzers, and Generic attacks. However, it is difficult for our proposed model to discover Backdoor, Analysis, Worms, and Shellcode attacks, and most of these attacks are classified as Exploits attacks. This can be explained because the five attacks mentioned above all rely on exploiting security vulnerabilities in the target system or software, and hence Backdoor, Analysis, Worms, and Shellcode attacks are similar to Exploit attacks in signature. Moreover, the Backdoor, Analysis, Worms, and Shellcode attacks make up only 0.90%, 1.14%, 0.07%, and 0.59% of all the traffic in the data set, respectively, and provide insufficient data for the training stage. Therefore, the Backdoor, Analysis, Worms, and Shellcode attacks are wrongly recognized as Exploit attacks by the CRGT-SA model.

As one of our future directions, we plan to improve the accuracy of our proposed CRGT-SA model in detecting four minor classes of cyber-attacks, which will require us to master their differences more precisely at first. In terms of Backdoor and Exploits attacks, the former focuses on embedding backdoors in the system for long-term access and control, whereas the latter focuses on exploiting the vulnerability itself to perform malicious actions. In terms of Analysis and Exploits attacks, the former aims to improve the security of the system and protect the privacy of users, wherear the latter aims to perform malicious actions. In terms of Worms and Exploits attacks, the former type primarily infects the system by replicating and spreading itself and forming a worm network, whereas the latter primarily exploits specific vulnerabilities to perform malicious operations. In terms of Shellcode and Exploits attacks, the former can be considered as a subset or special case of the latter. On that basis, we plan to import the GAN module into our proposed model to generate realistic data and expand the diversity of samples for these minor classes.

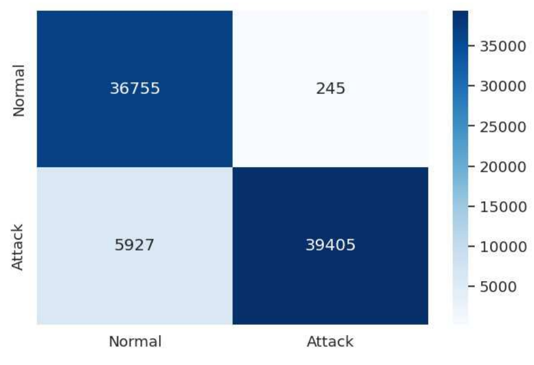


Fig. 8 Confusion matrix on binary classification

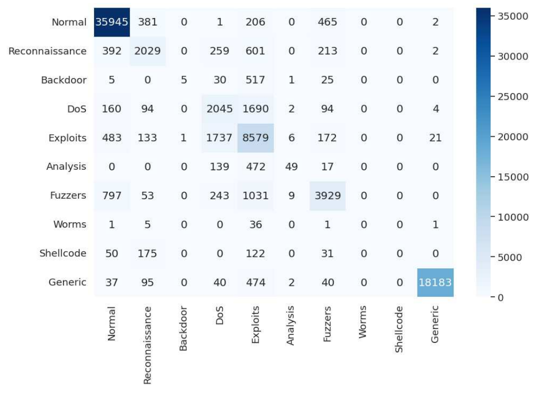


Fig. 9 Confusion matrix on multi-class classification

5.6 Experimental results and analysis on the nsl-kdd data set

To demonstrate the generalization ability of our proposed CRGT-SA model, we conducted experiments on the NSL-KDD data set to compare the performance of our proposed model with traditional and state-of-the-art machine learning and deep learning models.

5.6.1 Comparative analysis with traditional machine learning models on the nsl-kdd data set

Similar to Subsection 5.2, we conducted experiments to compare the performance of our proposed CRGT-SA model with traditional machine learning models including LR, GNB, KNN, AdaB, and RF on the NSL-KDD data set, and the comparative results of the binary and multi-class classifications are shown in Tables 8 and 9, respectively. It can be observed that our proposed model achieves the best performance in all the compared models in both binary or multi-class classifications. On the contrary, the GNB model has the poorest performance on both scenarios among all the models. This can be explained because the NSL-KDD data set contains a large amount of network traffic data that is highly correlated and redundant. However, the GNB model assumes that all features are independent, which violates the characteristics of this data set, leading to a low accuracy and F1-score. Moreover, the AdaB model performs poorly on the multi-class classification, because it is a method based on weak classifier integration, which relies on the predictive performance of a weak classifier. In the multi-class classification scenario, weak classifiers have limited ability to distinguish complex patterns and multicategories, which finally results in low accuracy and F1-score by this model.

Table 8 Comparative analysis with traditional machine learning models of binary classification on the NSL-KDD data set

Method	Accuracy	F1-score
LR	0.753	0.742
GNB	0.565	0.387
KNN	0.775	0.769
AdaB	0.776	0.773
RF	0.780	0.767
CRGT-SA	**0.907**	**0.918**

Table 9 Comparative analysis with traditional machine learning models of multi-class classification on the NSL-KDD data set

Method	Accuracy	F1-score
LR	0.750	0.706
GNB	0.442	0.461
KNN	0.754	0.720
AdaB	0.468	0.405
RF	0.758	0.716
CRGT-SA	**0.867**	**0.866**

5.6.2 Comparative analysis with state-of-the-art deep learning models on the nsl-kdd data set

Similar to Subsection 5.3, we conducted experiments to compare the performance of our proposed CRGT-SA model with state-of-the-art deep learning models including HAST, LuNet, and MSCNN-LSTM on the NSL-KDD data set, and the comparative results are shown in Table 10. It can be observed that our proposed model achieves the best performance among all the compared models. When compared with the UNSW-NB15 data set, our proposed model obtains a higher accuracy and F1-score on the NSL-KDD data set, because this data set only contains four categories of network attacks and three of them belong to major classes, and are easier to classify.

5.6.3 Ablation studies on the NSL-KDD data set

Similar to Subsection 5.4, we conducted ablation experiments to compare the performance of our proposed CRGT-SA model with CNN, CNN-TCN, CNN-LSTM, and CNN-TCN-LSTM models on the NSL-KDD data set, and the comparative results of the binary and multi-class classifications are shown in Tables 11 and 12, respectively. It can be observed that our proposed model achieves the best performance of all the compared models. Moreover, CNN-LSTM always has a high accuracy and F1-score when compared with CNN-TCN in binary and multi-class classifications. Even though both LSTM and TCN models are used for capturing temporal characteristics, they are different in learning long-term dependencies. The LSTM model designs a special gating mechanism to selectively remember or forget past information, which is effective in capturing long-term dependencies. However, the TCN model relies on the extended convolution operations to learn temporal characteristics, because a simple convolution cannot effectively capture the long-term dependencies.

Table 10 Comparative analysis with state-of-the-art deep learning models on the NSL-KDD data set

Method	Accuracy	F1-score
HAST	0.799	0.771
LuNet	0.813	0.774
MSCNN-LSTM	0.849	0.786
CRGT-SA	**0.867**	**0.866**

Table 11 Ablation studies of binary classification on the NSL-KDD data set

Method	Accuracy	F1-score
CNN	0.813	0.816
CNN-TCN	0.843	0.743
CNN-LSTM	0.824	0.828
CNN-TCN-LSTM	0.840	0.847
CRGT-SA	**0.907**	**0.918**

Table 12 Ablation studies of multi-class classification on the NSL-KDD data set

Method	Accuracy	F1-score
CNN	0.779	0.807
CNN-TCN	0.781	0.787
CNN-LSTM	0.836	0.851
CNN-TCN-LSTM	0.794	0.822
CRGT-SA	**0.867**	**0.866**

Confusion matrix on the NSL-KDD data set

The confusion matrices of using our proposed CRGT-SA model on the NSL-KDD data set for binary and multi-class classifications are shown in Figs 10 and 11, respectively. It can be observed that our proposed model can correctly classify normal traffic as well as three abnormal traffic types including dos, probe and r2l attacks, because all of them belong to majority classes of the data set. However, most u2r traffic is recognized as normal traffic by our proposed model. This is because in the NSL-KDD data set, only 119 of 148, 517 pieces of data are labeled as u2r attacks, a proportion of 0.08%. In terms of the category with less data, our proposed model cannot be trained enough, resulting in poor classification.

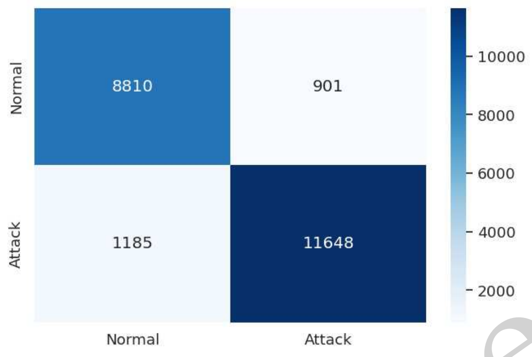


Fig. 10 Confusion matrix on binary classification of the NSL-KDD data set

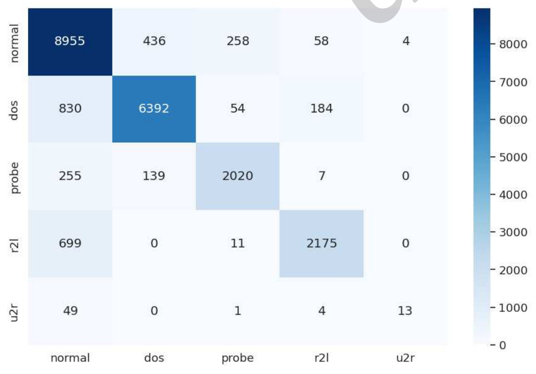


Fig. 11 Confusion matrix on multi-class classification of the NSL-KDD data set

6 Conclusions and future works

This study designs a novel and hybrid deep learning model called CRGT-SA to achieve network intrusion detection. To extract sufficient spatiotemporal properties from network traffic data, our proposed model intertwines the CNN with Gated TCN and LSTM modules. More specifically, we divide the feature extraction into multiple steps with gradually increasing granularity, and execute each step using combined CNN, LSTM, and Gated TCN modules. On that basis, the self-attention mechanism is introduced to select significant features from network traffic data. In the experiments, we evaluate and compare our proposed CRGT-SA model with traditional machine learning models and state-of-the-art deep learning models. According to the simulation results, our proposed model achieves the highest accuracy and F1-scores among all the models, achieving accuracy of 91.5% and 90.5% for binary and multi-class classifications, respectively. Moreover, the generalization ability of our proposed model is further demonstrated through another series of experiments on the NSL-KDD data set in comparison with other models.

However, as can be seen from the structure of our proposed CRGT-SA model, the computational complexity cannot be ignored. To enhance the computational efficiency, we plan to import the process of feature selection into the data preprocessing stage in future works, such as adopting the genetic algorithm to reduce dimensionality. Moreover, as can be observed from the confusion matrix, our proposed CRGT-SA model cannot detect Backdoor, Analysis, Worms, and Shellcode attacks efficiently because of insufficient data in the training stage. To cope with this problem, we plan to import the GAN module into our proposed model to generate samples of rare attack categories to expand the data set between the data preprocessing and intrusion detection stages.

Contributors

Jue CHEN and Yurui MA designed the research. Yurui MA processed the data. Jue CHEN and Yurui MA drafted the paper. Junhao WU helped organize the paper. Xihe QIU and Wenjing LV put forward advice. Jue CHE revised and finalized the paper.

Conflict of interest

All the authors declare that they have no conflict of interest.

Data availability

Data available on request from the authors.

References

Abdel-Basset M, Hawash H, Chakrabortty RK, et al., 2021. Semi-supervised spatiotemporal deep learning for intrusions detection in IoT networks. *IEEE Int Things J*, 8(15):12251-12265. https://doi.org/10.1109/JIOT.2021.3060878

Abdelmoumin G, Rawat DB, Rahman A, 2022. On the performance of machine learning models for anomaly-based intelligent intrusion detection systems for the internet of things. *IEEE Int Things J*, 9(6):4280-4290. https://doi.org/10.1109/JIOT.2021.3103829

Abeshu A, Chilamkurti N, 2018. Deep learning: the frontier for distributed attack detection in fog-to-things computing. *IEEE Commun Mag*, 56(2):169-175. https://doi.org/10.1109/MCOM.2018.1700332

Aldweesh A, Derhab A, Emam AZ, 2020. Deep learning approaches for anomaly-based intrusion detection systems: a survey, taxonomy, and open issues. Knowl Based Syst, 189:105124. https://doi.org/10.1016/j.knosys.2019.105124

Al-Garadi MA, Mohamed A, Al-Ali AK, et al., 2020. A survey of machine and deep learning methods for internet of things (IoT) security. *IEEE Commun Surv Tutorials*, 22(3):1646-1685. https://doi.org/10.1109/COMST.2020.2988293

Cao B, Li CH, Song YF, et al., 2022. Network intrusion detection model based on CNN and GRU. *Appl Sci*, 12(9):4184. https://doi.org/10.3390/app12094184

Chen J, Xiong YJ, Qiu XH, et al., 2022. A cross entropy based approach to minimum propagation latency for controller placement in software defined network. *Comput Commun*, 191:133-144. https://doi.org/10.1016/j.comcom.2022.04.030

Cook AA, Misirli G, Fan Z, 2020. Anomaly detection for IoT time-series data: A survey. *IEEE Int Things J*, 7(7):6481-6494. https://doi.org/10.1109/JIOT.2019.2958185

Fang LM, Li Y, Liu Z, et al., 2021. A practical model based on anomaly detection for protecting medical IoT control services against external attacks. *IEEE Trans Ind Inf*, 17(6):4260-4269. https://doi.org/10.1109/TII.2020.3011444

Fenghour S, Chen DQ, Guo K, et al., 2021. Deep learning-based automated lip-reading: a survey. *IEEE Access*, 9:121184-121205. https://doi.org/10.1109/ACCESS.2021.3107946

Gan BQ, Chen YQ, Dong QP, et al., 2022. A convolutional neural network intrusion detection method based on data imbalance. *J Supercomput*, 78(18):19401-19434. https://doi.org/10.1007/s11227-022-04633-x

Hassan MM, Gumaei A, Alsanad A, et al., 2020. A hybrid deep learning model for efficient intrusion detection in big data environment. *Inf Sci*, 513:386-396. https://doi.org/10.1016/j.ins.2019.10.069

Jian SL, Pang GS, Cao LB, et al., 2019. CURE: flexible categorical data representation by hierarchical coupling learning. *IEEE Trans Knowl Data Eng*, 31(5):853-866. https://doi.org/10.1109/TKDE.2018.2848902

Kasongo SM, 2023. A deep learning technique for intrusion detection system using a recurrent neural networks based framework. *Comput Commun*, 199:113-125. https://doi.org/10.1016/j.comcom.2022.12.010

Khan MA, 2021. HCRNNIDS: hybrid convolutional recurrent neural network-based network intrusion detection system. *Processes*, 9(5):834. https://doi.org/10.3390/pr9050834

Khan MA, Iqbal N, Imran N, et al., 2023. An optimized ensemble prediction model using automl based on soft voting classifier for network intrusion detection. *J Network Comput Appl*, 212:103560. https://doi.org/10.1016/j.jnca.2022.103560

Kumar R, Kumar P, Tripathi R, et al., 2022. P2SF-IoV: a privacy-preservation-based secured framework for internet of vehicles. *IEEE Trans Intell Transport Syst*, 23(11):22571-22582. https://doi.org/10.1109/TITS.2021.3102581

Laghrissi F, Douzi S, Douzi K, et al., 2021. Intrusion detection systems using long short-term memory (LSTM). *J Big Data*, 8(1):65. https://doi.org/10.1186/s40537-021-00448-4

Lv ZH, Qiao L, Li JH, et al., 2021. Deep-learning-enabled security issues in the internet of things. *IEEE Int Things J*, 8(12):9531-9538. https://doi.org/10.1109/JIOT.2020.3007130

Marteau PF, 2021. Random partitioning forest for point-wise and collective anomaly detection-application to network intrusion detection. *IEEE Trans Inf Forensics Secur*, 16:2157-2172. https://doi.org/10.1109/TIFS.2021.3050605

Moustafa N, Slay J, 2016. The evaluation of network anomaly detection systems: statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 data set. *Inf Secur J A Global Perspect*, 25(1-3):18-31. https://doi.org/10.1080/19393555.2015.1125974

Nie LS, Wu YX, Wang XJ, et al., 2022. Intrusion detection for secure social internet of things based on collaborative edge computing: a generative adversarial network-based approach. *IEEE Trans Comput Soc Syst*, 9(1):134-145. https://doi.org/10.1109/TCSS.2021.3063538

Oseni A, Moustafa N, Creech G, et al., 2023. An explainable deep learning framework for resilient intrusion detection in IoT-enabled transportation networks. *IEEE Trans Intell Transport Syst*, 24(1):1000-1014. https://doi.org/10.1109/TITS.2022.3188671

Qi LY, Yang YH, Zhou XK, et al., 2022. Fast anomaly identification based on multiaspect data streams for intelligent intrusion detection toward secure industry 4.0. *IEEE Trans Ind Inf*, 18(9):6503-6511. https://doi.org/10.1109/TII.2021.3139363

Vasilomanolakis E, Karuppayah S, MÍźhlh?user M, et al., 2015. Taxonomy and survey of collaborative intrusion detection. *ACM Comput Surv*, 47(4):55. https://doi.org/10.1145/2716260

Vinayakumar R, Alazab M, Soman KP, et al., 2019. Deep learning approach for intelligent intrusion detection system. *IEEE Access*, 7:41525-41550. https://doi.org/10.1109/ACCESS.2019.2895334

Wahab OA, 2022. Intrusion detection in the IoT under data and concept drifts: online deep learning approach. *IEEE Int Things J*, 9(20):19706-19716. https://doi.org/10.1109/JIOT.2022.3167005

Wang K, Zhang AH, Sun HR, et al., 2023. Analysis of recent deep-learning-based intrusion detection methods for in-vehicle network. *IEEE Trans Intell Transport Syst*, 24(2):1843-1854. https://doi.org/10.1109/TITS.2022.3222486

Wang W, Sheng YQ, Wang JL, et al., 2018. HAST-IDS: learning hierarchical spatial-temporal features using deep neural networks to improve intrusion detection. *IEEE Acess*, 6:1792-1806. https://doi.org/10.1109/ACCESS.2017.2780250

Wang XF, Han YW, Leung VCM, et al., 2020. Convergence of edge computing and deep learning: a comprehensive survey. *IEEE Commun Surv Tutorials*, 22(2):869-904. https://doi.org/10.1109/COMST.2020.2970550

Wu PL, Guo H, 2019. LuNet: a deep neural network for network intrusion detection. IEEE Symp Series on Computational Intelligence, p.617-624. https://doi.org/10.1109/SSCI44817.2019.9003126

Zhang JW, Ling Y, Fu XB, et al., 2020. Model of the intrusion detection system based on the integration of spatial-temporal features. *Comput Secur*, 89:101681. https://doi.org/10.1016/j.cose.2019.101681

Zhuo XY, Zhang JL, Son SW, 2017. Network intrusion detection using word embeddings. IEEE Int Conf on Big Data, p.4686-4695. https://doi.org/10.1109/BigData.2017.8258516