

Article

TGA: A Novel Network Intrusion Detection Method Based on TCN, BiGRU and Attention Mechanism

Yangyang Song ¹, Nurbol Luktarhan ^{1,*}, Zhaolei Shi ¹ and Haojie Wu ² 

¹ College of Information Science and Engineering, Xinjiang University, Urumqi 830046, China; song_yy@stu.xju.edu.cn (Y.S.); shizhaolei@stu.xju.edu.cn (Z.S.)

² School of Software, Xinjiang University, Urumqi 830046, China; 107552104325@stu.xju.edu.cn

* Correspondence: nurbol@xju.edu.cn

Abstract: With the increasing complexity of the network environment, the types of network attacks are gradually increasing. Network intrusion detection systems can detect and identify network attacks effectively. However, the existing methods have some limitations, focusing only on local or global temporal features of network traffic. To address the above issues, we present a novel network intrusion detection model (TGA) based on Temporal Convolutional Network (TCN), Bidirectional Gated Recurrent Unit (BiGRU), and self-attention mechanism. TCN extracts local temporal information from network traffic sequences, while BiGRU extracts global temporal information from network traffic sequences. However, TCN and BiGRU do not consider the weights of features when extracting them, so an attention mechanism is added. The feature vectors obtained in TCN and BiGRU are fused and then input into the self-attention mechanism to capture the correlation between different positions in the sequence and reassign the weights of the temporal features to further enhance the model's capabilities. Lastly, it is delivered to the classifier to classify different network traffic classes. Our method achieves 97.83% accuracy on the public CSE-CIC-IDS2018 dataset. After extensive experiments, our idea proved to be reasonable and practical.

Keywords: network intrusion detection; temporal convolutional network; bidirectional gated recurrent unit; self-attention mechanism



Citation: Song, Y.; Luktarhan, N.; Shi, Z.; Wu, H. TGA: A Novel Network Intrusion Detection Method Based on TCN, BiGRU and Attention Mechanism. *Electronics* **2023**, *12*, 2849. <https://doi.org/10.3390/electronics12132849>

Academic Editor: Suleiman Yerima

Received: 24 May 2023

Revised: 25 June 2023

Accepted: 25 June 2023

Published: 27 June 2023



Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

With the fast growth of Internet of Things (IoT) techniques, the emergence of the Internet has significantly changed people's lives. The Internet has penetrated almost every aspect of our daily life. This means anyone can easily access any network at a low cost without any complicated procedures. However, with this comes an increasing number of attack tactics by hackers and the increasing complexity of cyber-security issues that people face. The emergence of any vulnerability or threat can affect the entire network [1]. The diverse categories of cyber attacks can severely impact individuals and risk property damage to society and the country. Cyber-security thus becomes an issue that cannot be ignored. In order to avoid the impact of cyber-attacks as much as possible, effective means of defense are needed. Currently, many methods are used to prevent hacking, such as firewalls, anti-virus software, access control mechanisms, and other technologies. However, most of these technical means are based on passive defense strategies and rely heavily on historical traffic databases, which require more proactive countermeasures in the face of ever-changing network attacks. Network intrusion detection can actively identify and determine the security vulnerabilities in the network and promptly respond to achieve adequate network protection [2], which is a crucial technique in cybersecurity. Network intrusion detection usually involves feature extraction and network traffic classification. First, features need to be extracted from the dataset to minimize data complexity and construct meaningful information; second, benign and attack traffic is classified using feature

information. In this way, network intrusions can be detected and prevented. Therefore, network intrusion detection has become a hot direction of interest for many researchers [3].

As artificial intelligence technology advances rapidly, machine learning (ML) has achieved outstanding results in many fields [4]. Furthermore, ML techniques have become prevalent in network intrusion detection to identify network attacks. However, this method is shallow learning, which usually requires manual intervention for feature selection, tends to ignore the correlation between feature data, and does not efficiently address the issue of classifying massive data traffic, leading to low accuracy in identifying network attacks. Later, with the advance of computer hardware, deep learning (DL) was proposed and widely used in image classification and emotion recognition. DL can autonomously learn the feature information of traffic data, which solves the problem of manual selection of traffic features in ML and has an excellent performance in handling large amounts of data. Applying the advantages of DL to the network intrusion detection task can enable better network traffic classification.

Since the relationship between traffic samples collected in adjacent periods is very close, the temporal relationship of network traffic should be considered in the network traffic classification task. Temporal Convolutional Network (TCN) and Bidirectional Gated Recurrent Unit (BiGRU) are DL models for processing time series data, and they have different structures and advantages. However, using a single structure results in models that can only focus on local or global temporal features of network traffic. In addition, TCN and BiGRU do not consider the weights of the features when extracting them. Based on the above analysis, the following are the primary contributions of this paper:

- This paper proposes an algorithmic model based on TCN, BiGRU, and attention mechanism (TGA). The temporal features of traffic sequences are extracted using both TCN and BiGRU models simultaneously, which can improve the effectiveness of sequence data modeling by using their respective advantages to some extent. After that, their outputs are fused, and the correlation between different positions in a sequence is captured by adding a self-attention mechanism to enhance the model's expressiveness.
- The presented method was evaluated on the CSE-CIC-IDS2018 dataset at 97.83% accuracy. Compared with existing methods, our proposed model has shown much better results.

The remainder of the paper is shown below: in Section 2, we introduce the related work on network intrusion detection; in Section 3, we present a novel network intrusion detection model with TCN, BiGRU, and the self-attention mechanism. Additionally, we describe the dataset and the data pre-processing process; in Section 4, we present the experimental process and performance results. The experimental process includes the experimental environment, hyperparameter settings, and the analysis of experimental results; lastly, we discuss the conclusions and the directions of future works in Section 5.

2. Related Work

Network intrusion detection technology is a network security mechanism that monitors network traffic and system activity, then identifies and responds to unauthorized access, malicious behavior, or attacks [5]. It protects computers from network intrusion attacks and is one of the essential tools for protecting network security. ML approach techniques have found extensive application in network intrusion detection in the last two decades. Tavallaee et al. [6] evaluated the capabilities of some ML algorithms, which included decision tree (DT), Naive Bayes (NB), and SVM. Hamed et al. [7] used principal component analysis and LDA to obtain four-dimensional data. During the classification stage, the four-dimensional samples are first fed to a Bayesian classifier, and then the samples predicted to be normal are utilized for secondary detection using CFKNN. Mahfouz et al. [8] comprehensively evaluated contemporary ML classifiers for detecting network traffic intrusions. They assess the classifiers in terms of various dimensions, such as feature selection, sensitivity to hyperparameter selection, and the challenge of class imbalance in intrusion detection. Zwane et al. [9] used seven ML classifiers, such as multilayer perceptron, SVM,

Bayesian network, RF, AdaBoost, Bootstrap aggregation, and DT, using Weka in their analysis.

Applying ML techniques to network traffic classification tasks can achieve high accuracy. However, these ML methods still have some limitations, such as manually selecting features. In addition, ML methods do not produce satisfactory results when handling high volumes of data. With the fast growth of artificial intelligence technology, DL is prevalent in many fields. DL techniques can autonomously learn feature information of traffic data, which solves the problem of manual selection of traffic features in ML and has an excellent performance in processing large volumes of data. DL techniques are currently receiving significant attention in research on network intrusion detection.

Vinayakumar R et al. [10] presented a CNN-based network intrusion detection model. This paper evaluates the effectiveness of various DL models, including MLP, CNN, and CNN-RNN. The study uses the KDDCup99 dataset for experimentation, and the results indicate that CNN can automatically extract helpful feature information from the data. However, they did not consider that network traffic has contextual relationships. Yan et al. [11] applied NSL-KDD and UNSW-NB15 datasets, and four DL models were analyzed for their performance. These include restricted Boltzmann machine (RBM), multilayer perceptron (MLP), sparse autoencoder (SAE), and MLP with feature embedding. However, their experiments were not evaluated on the newer intrusion dataset. Liu et al. [4] presented a CNN combined with CBAM for a network intrusion detection model, and this paper used the newer dataset: CSE-CIC-IDS2018. Finally, the presented model was compared with DNN and CNN, proving the proposed model's validity. Usama et al. [12] introduced a generative adversarial network (GAN), which can be utilized for intrusion detection with solid resistance to adversarial attacks. However, the method could be more challenging to parameterize in most cases and suffers from training instability.

Due to the contextual relationship of network traffic, the temporal relationship should be considered when extracting features. TCN and RNN frameworks are widely used to process time series data. Kong et al. [13] proposed an integrated deep generative model, which captures time-series dependencies by combining a generative adversarial network built by a BiLSTM and an attention mechanism. M S et al. [14] presented a network intrusion warning prediction approach based on GRU. This model learns dependencies in a series of security alerts and outputs possible future alerts based on the history of alerts from attack sources. Zhang et al. [15] introduced a new detection approach built on TCN. The TCN is utilized as a predictor, allowing for the mapping of series-to-series and the projections of a user's future behavior based on their current actions. This paper presents a network intrusion detection algorithm combining TCN, BiGRU, and self-attention mechanism.

3. Methodology

3.1. Model Architecture

In this work, we presented the TGA model. The architecture of the proposed model is shown in Figure 1. First, the pre-processed dataset is input to the constructed model, and the model consists of three modules. The TCN network model builds the first module, which extracts the local temporal features from the network traffic after several convolutional operations. The BiGRU network model establishes the second module, which extracts the global temporal features from the network traffic. The two different temporal features captured are merged to extend the information of the characteristics. After that, the fused feature vectors are fed into the self-attention mechanism for capturing correlations between different positions in the sequence and reassigning the weights of the temporal features. As a result, the highest prominent characteristics are preserved to the maximum extent. This can boost the efficiency and robustness of the model. Eventually, a softmax layer is employed as a classifier for the network traffic classification task.

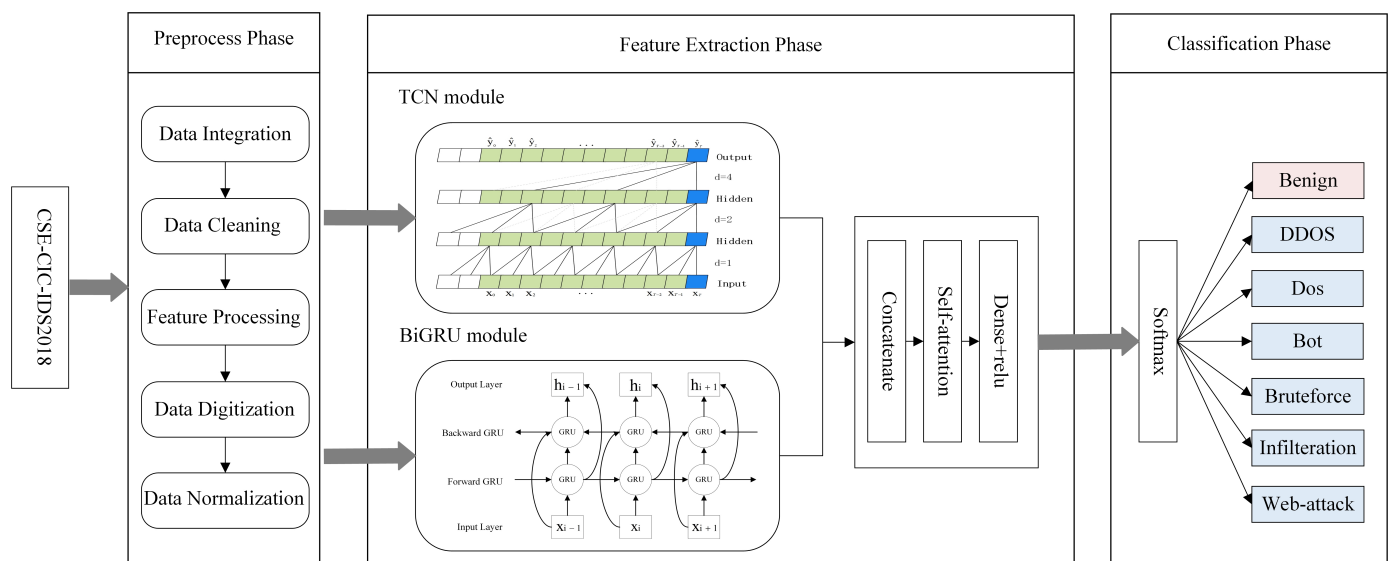


Figure 1. The architecture of the TGA model.

3.2. Datasets

The domain of network intrusion detection has plenty of mainstream datasets, such as KDDCup99 [16], NSL-KDD [17], UNSW-NB15 [18], and CSE-CIC-IDS2018 [19], but the KDDCup99 dataset contains numerous redundant and duplicate traffic. The NSL-KDD dataset improves on the KDDCup99 dataset by deleting duplicate, redundant, and irrelevant traffic samples. UNSW-NB15 is a newer dataset established by the Australian Cyber Security Centre's Cyber Scope Lab in 2015. However, these datasets all contain old-fashioned and unreliable traffic and do not quite match current real attacks. The Canadian Cyber Security Laboratory proposed the CSE-CIC-IDS2018 dataset in 2018. This dataset collects ten days of samples containing the latest attack traffic and has a sufficiently large sample data size. This paper utilizes the CSE-CIC-IDS2018 dataset to evaluate our presented method.

The datasets contain two types of different files: Pcap format files and CSV format files. CSV format files are generated by CICFlowMeter-V3 that contain 10 CSV files, about 16,000,000 instances. Malicious traffic is divided into six categories: Bruteforce, Dos, DDOS, Infiltration, Bot, and Web-attacks [19]. These kinds of attacks are relatively frequent in real network environments and have a considerable impact on the system's normal function. They have a broad impact range and can cause severe issues such as user data leakage and system vulnerability exploitation. Several features [20,21] of the CSE-CIC-IDS2018 dataset are illustrated in Table 1. For this experiment, only 8 CSV files were used. We have divided the dataset into three sets; 10% is used for testing, and the percentage of samples allocated to training and validation stages is 9:1.

Table 1. Several features details of the CSE-CIC-IDS2018 dataset.

Feature name	Feature Short Description	Data Type
Dst Port	The destination port number	Integer
Protocol	Protocols used for connection	Integer
Flow Duration	Duration that connection occurred	String
Tot Fwd Pkts	Total number of packages in the forward-looking direction	Integer
Tot Bwd Pkts	Total number of packets in the back direction	Integer
ToLen Fwd Pkts	Total length of forward packets	Integer
Fwd Pkt Len Max	Maximum size of forward mode network packets	Float
Bwd Pkt Len Mean	Mean size of backward mode network packets	Float
...	...	...
Active mean	Mean time a flow was active before it becomes idle	Float
Idle Std	Standard deviation time a flow was idle before it becomes active	Float
Idle Min	Minimum idle time for a flow before it becomes active	Float
Label	Describes whether the file is attack or benign	char

3.3. Data Preprocess

In general, the network traffic dataset is not used directly as an input into the model and requires data preprocessing work, which includes the following steps:

- **Data Integration:** In this experiment, we selected eight days of traffic samples from the CSE-CIC-IDS2018 dataset, and these eight CSV files were first integrated together. After that, 14 different tags were reclassified into seven categories, including one benign and six malicious tags.
- **Data Cleaning:** The presence of anomalous traffic in the dataset can adversely impact the training, so the rows containing NaN, Infinity, and null values are removed from this experiment. Table 2 shows the distribution of each category before and after cleaning. Before data cleaning, there were 5,567,951 benign samples and 2,103,173 malicious samples. After data cleaning, there are 5,538,387 benign samples and 2,102,418 malicious samples.
- **Feature Processing:** There are 79 attributes and one label in the CSE-CIC-IDS2018 dataset. Timestamp has been removed since it has a slight impact on network traffic identification.
- **Data Digitization:** The one-hot encoding method converts categorical features into a numeric variable.
- **Data Normalization:** If data features are not normalized, the model will assume that features taking larger values have a greater impact on the results. Normalizing data features can balance the size of each feature value and ensure that the DL model converges when backpropagating. In our experiment, the Min-Max normalization methodology is chosen to normalize the data on the dataset. Thus, each feature that has numerical values is normalized to the [0,1] interval, which can be defined as follows:

$$x_n = \frac{x - x_{\min}}{x_{\max} - x_{\min}} \quad (1)$$

where x_n is the normalized value, x is the initial value to be processed, $x_{\max}$ and $x_{\min}$ are the maximum and minimum data values in the current attribute.

Table 2. Distribution of data before and after cleaning.

Class	Attack Type	Before	After
Benign	-	5,567,951	5,538,387
DDoS	DDoS attack-HOIC	686,012	685,993
-	DDoS attack-LOIC-UDP	1730	1729
Dos	DoS attacks-Hulk	461,912	461,905
-	DoS attacks-GoldenEye	41,508	41,505
-	DoS attacks-Slowloris	10,990	10,989
-	DoS attacks-SlowHTTPTest	139,890	139,880
Bot	Bot	286,191	286,170
Bruteforce	SSH-Brute force	187,589	187,576
-	FTP-Brute force	193,360	193,342
Infiltration	Infiltration	93,063	92,403
Web-attack	Brute Force -Web	611	609
-	Brute Force -XSS	230	230
-	SQL Injection	87	87

3.4. Background on Neural Networks

3.4.1. Temporal Convolutional Networks

TCN is a model presented by Bai et al. [22]. It is a neural network architecture suitable for solving sequential problems. TCN can be computed in parallel, has high computational efficiency, and can handle long time series. TCN performs well in many time-series data

processing tasks, and TCN mainly comprises causal convolution, dilated convolution, and residual block.

- **Causal Convolution**

Causal convolution performs convolution operations in a strict temporal order, only on data from past time steps. However, it should be noted that causal convolution is susceptible to the limitations of the received domain, so it can only process shorter historical information for prediction.

- **Dilated Convolution**

TCN introduces dilated convolution to enable the model to better capture the dependencies on long input data. The dilated convolution increases the “dilated rate” parameter compared with the traditional convolution, which can enhance the ability to extract features effectively and enable the model to expand the perceptual field of the convolution kernel and catch the long-term time-dependent relationship without increasing the number of parameters. When performing time series forecasting, assuming given a sequence of inputs series [22]: $x_0, x_1, x_2, \dots, x_{T-2}, x_{T-1}, x_T$. output series: $\hat{y}_0, \hat{y}_1, \hat{y}_2, \dots, \hat{y}_{T-2}, \hat{y}_{T-1}, \hat{y}_T$. The input sequence can be interpreted as a record of past data. The procedure aims is to develop a formulism of generating novel potential information, also referred to as an output sequence, that is built on past data [15]. This can be represented as follows:

$$\hat{y}_0, \hat{y}_1, \hat{y}_2, \dots, \hat{y}_{T-2}, \hat{y}_{T-1}, \hat{y}_T = f(x_0, x_1, x_2, \dots, x_{T-2}, x_{T-1}, x_T) \quad (2)$$

The Figure 2 illustrates the process of predicting an input sequence. When $d = 1$, every point of the input needs to be sampled; when $d = 2$, every two points are sampled as an input, and when $d = 4$, every four points are sampled as an input.

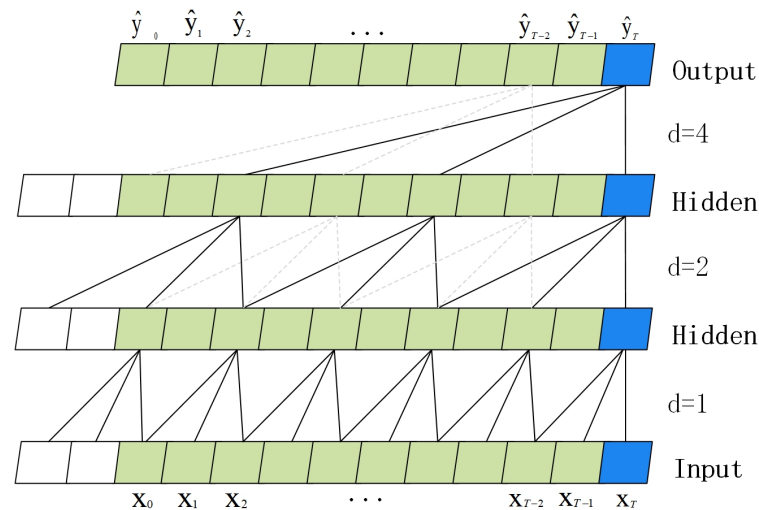


Figure 2. Dilated convolution of TCN model.

- **Residual Block**

The architecture of the residual block in the TCN model is illustrated in Figure 3, $\hat{z}^{(i-1)}$ means the residual block of input, and $\hat{z}^{(i)}$ denotes the residual block output. Two nonlinear dilated causal convolution layers are included in one residual block, WeightNorm is applied to the convolutional filters, and the activation function is chosen as ReLU. In addition, the Dropout [23] layer was used for regularization after the activation function layer. Considering the potential variability in the number of input and output channels, a 1×1 convolutional layer is used to adjust the quantity of channels.

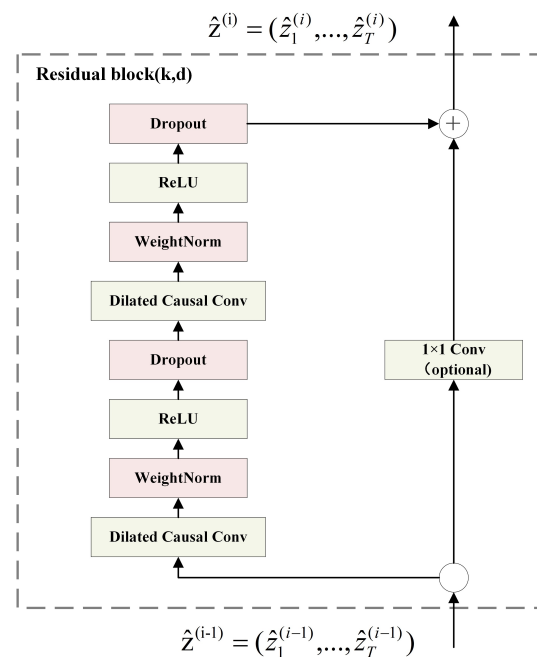


Figure 3. Residual block of TCN model.

3.4.2. Bidirectional Gated Recurrent Unit

Compared to the RNN framework, TCN may not be effective at catching long-term reliability information. However, the RNN framework is able to catch timing information of arbitrary length [24]. GRU [25] and LSTM [26] belong to the RNN network architecture, and both introduce gating mechanisms to control the flow of information. The difference is that LSTM has three gates, comprising a single input gate, forget gate, and output gate, while GRU is simplified from LSTM and contains just a reset gate and update gate. Compared with LSTM, GRU has a more straightforward model structure with fewer parameters, and the GRU block diagram is displayed in Figure 4.

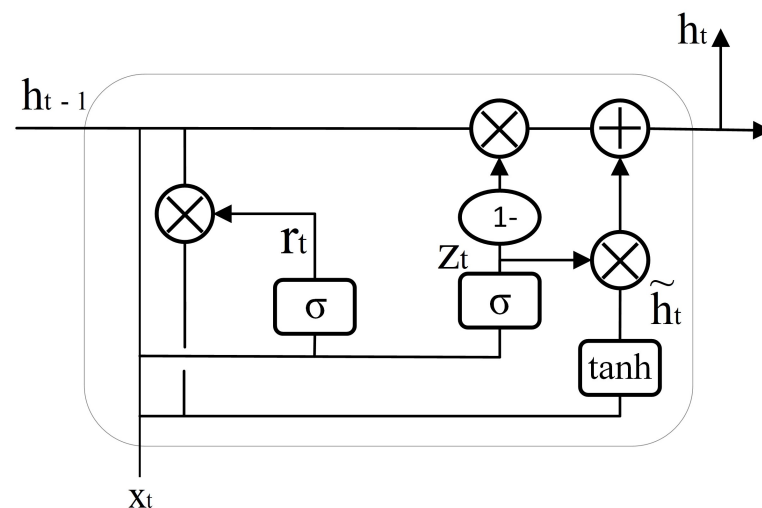


Figure 4. The block diagram of GRU.

The calculation formulas of GRU are shown in Equations (3)–(6). r and z denote the reset gate and update gate, respectively. $\tilde{h}$ represents the candidate output, and h represents

the output. W is the parameter to be determined and represents the input, σ denotes the Sigmoid function, and the subscripts t and $t - 1$ denote the present and previous moment.

$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t]) \quad (3)$$

$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t]) \quad (4)$$

$$\tilde{h}_t = \tanh(W_{\tilde{h}} \cdot [r_t \cdot h_{t-1}, x_t]) \quad (5)$$

$$h_t = (1 - z_t) \cdot h_{t-1} + z_t \cdot \tilde{h}_t \quad (6)$$

To increase the capabilities of the prediction, the model needs to learn the contextual information thoroughly and extract deeper traffic characteristics [27]. However, traditional models like GRU and LSTM only process data forward-to-backwardly [28], which can cause them to miss important information from earlier time points relevant to the current figures. To address this limitation, BiGRU was presented, which incorporates both forward and backward propagation directions in a single layer, using GRU units to learn information from past and future time points. The general architecture of BiGRU is illustrated in Figure 5.

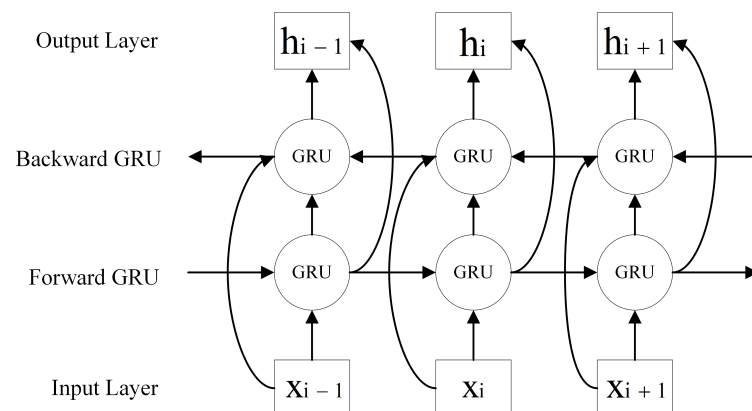


Figure 5. The block diagram of BiGRU.

3.4.3. Self-Attention Mechanism

The self-attention mechanism is a neural-network-based modeling technique for learning relationships between elements in sequential data. The self-attention mechanism was initially proposed in natural language processing [29]. However, it is also used successfully in other areas, such as image generation and super-resolution [30], where it has shown significant potential. The self-attention mechanism works by calculating the similarity between each element and other elements and converting these similarities into a weighted contextual representation. The self-attention mechanism has the advantages of parallel computation, long-range dependency capture, and good interpretability.

3.4.4. Softmax Layer

The softmax layer is an output layer in the neural network and is used in multi-category tasks. In the softmax layer, for each input sample, the neural network calculates the score corresponding to each category. It outputs a probability distribution such that the sum of the scores of all categories is 1. This output allows us to utilize the cross-entropy loss function to compute the gap between the network output and the true label, and the weight parameters of the neural network are renewed by the back-propagation algorithm.

4. Experimental Method

We present a traffic classification model using TCN, BiGRU, and self-attention mechanisms to boost the network intrusion detection accuracy. To validate the effectiveness of our proposed model, we outline the experimental procedure, we followed for network intrusion detection using the CSE-CIC-IDS2018 dataset, as well as the results obtained from the experiments conducted in this section.

4.1. Experimental Environment

The experiments were conducted with Python version 3.8 on an Ubuntu 20.04 operating system. The hardware specification is a single 3080 GPU with 12 GB of memory. Tensorflow version 2.9.0 and Keras version 2.9.0 were used for all experiments. Furthermore, the other major model parameters are given below: the batch size is 512, the epoch number is 30, and Adam is selected as the optimizer for parameter learning. Our model uses a cross-entropy loss function. Minimizing the cross-entropy loss enables the model to predict results closer to the actual labels.

4.2. Evaluation Indicator

To evaluate the effectiveness of our proposed model, we visualize the detection outcomes by utilizing a confusion matrix, which is given in Table 3. TP refers to the count of correctly classified benign samples, while FP represents the number of attack samples incorrectly classified as benign. TN denotes the number of correctly classified attack samples, and FN represents the count of benign samples incorrectly classified as an attack. Furthermore, we also employed four evaluation metrics: accuracy, precision, recall, and f1-score [31].

Table 3. Confusion matrix.

Actual \ Predict	Predict		
	Benign	Malware	
Benign	TP	FN	
Malware	FP	TN	

4.3. Effect of Hyperparameters

4.3.1. Learning Rate

In our experiments, the learning rate is a critical hyperparameter, which controls the number of updates to the model parameters in each iteration. A suitable learning rate can accelerate the model training process and improve the model's performance. To find a suitable learning rate, multiple sets of controlled trials are usually required for hyperparameter tuning.

We compared the performance of various learning rates on the traffic classification results. We set the learning rate to four values: 0.1, 0.01, 0.001, and 0.0001. As shown in Figure 6, the model's performance is improving as the learning rate decreases. When the learning rate was set to 0.001, precision, recall, and f1-score were the highest; 97.85%, 97.83%, and 97.57%, respectively, reached the best performance. Which were higher than the learning rate was 0.1, increased by 0.37%, 0.57%, and 0.7%, respectively. When the learning rate was 0.01, the values of precision, recall, and f1-score were 97.58%, 97.44%, and 97.08%, respectively. When it decreases to 0.0001, the model's performance decreases, and the four evaluation metrics are 97.66%, 97.67%, and 97.39%, respectively. Therefore, the learning rate is set to 0.001.

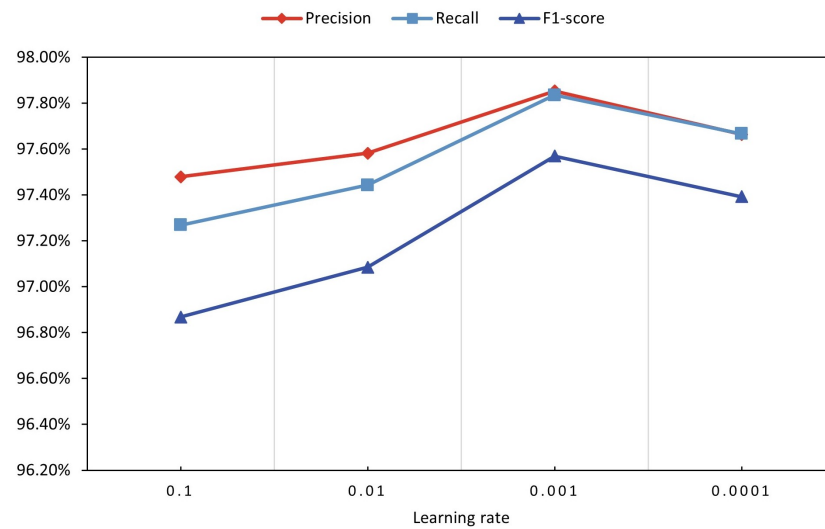


Figure 6. The accuracy changes with learning rates different.

4.3.2. Nb_Stacks

The Nb_stacks parameter specifies the number of stacks in TCN. Figure 7 displays the capabilities of the proposed model in various Nb_stacks. The figure indicates that the model outperformed when Nb_stacks is 2, and there is a slight improvement in precision, recall, and f1-score compared with the rest two groups. Therefore, we choose Nb_stacks equal to 2 for our experiments.

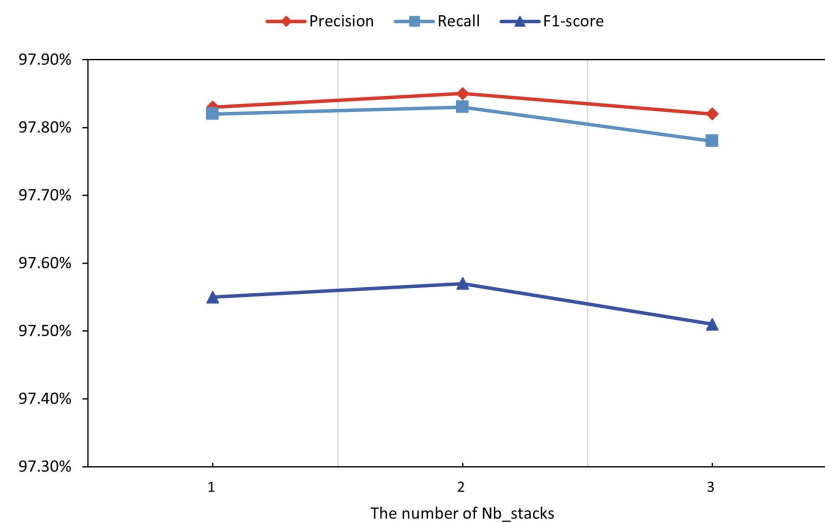


Figure 7. The accuracy changes with Nb_stacks difference.

4.4. Result and Analysis

This section compares the proposed algorithm with the single algorithms TCN, BiGRU, and TCN-BiGRU. After careful analysis, we find that using a single TCN is capable of extracting the local temporal features, but it ignores the global temporal features. By combining TCN with BiGRU, we are able to extract both local and global temporal features of the traffic, improving the model's performance. Finally, by adding self-attention, the model can focus on more essential feature data, and the model's performance is further improved.

As shown in Figure 8, when combining TCN and BiGRU, the performance of precision, recall, and f1-score are 97.71%, 97.68%, and 97.40%, respectively, which are higher than those of TCN and BiGRU alone. Adding self-attention improves the value of four evaluation metrics by 0.14%, 0.14%, 0.14%, and 0.15%, respectively. The experiments demonstrate that

our proposed algorithm model combining TCN, BiGRU, and self-attention can efficiently enhance the traffic performance of multi-classification.

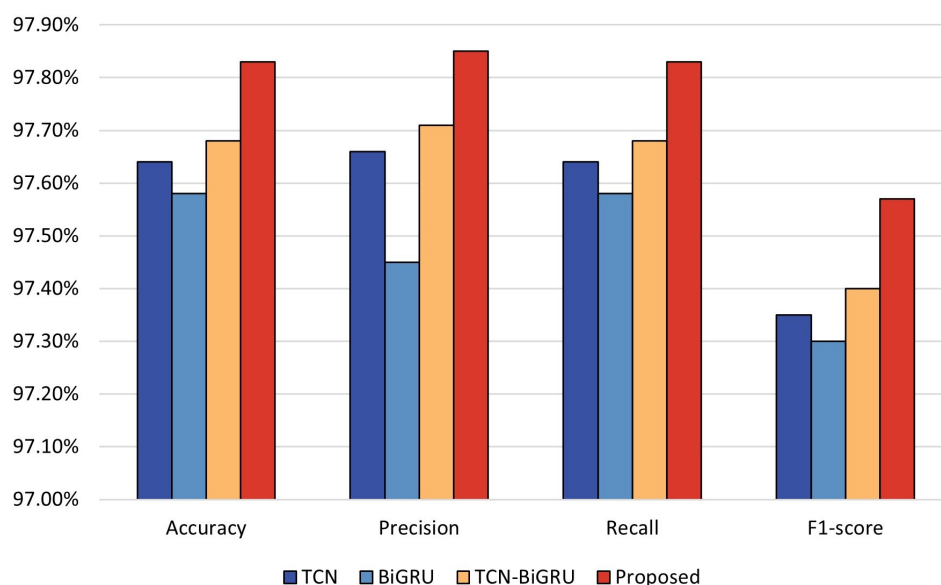


Figure 8. Experimental results based on multiple classifications.

To better demonstrate the capabilities of the presented algorithm in network intrusion detection, We compare the performance of the TGA model with that of recently developed algorithms, for example, LSTM + AM [32], DNN [33], CNN [34], DAE + DNN [35], CNN + CBAM [4], and ID-RDRL [20], all of them using various DL approaches. Experiments indicate that our proposed algorithm is superior to other baseline models regarding accuracy, precision, recall, and f1-score.

As illustrated in Table 4, the value of four evaluation metrics for LSTM + AM [32] are 96.2%, 96%, 96%, and 93%, respectively. Moreover, our method improves these four metrics by improving 1.63%, 1.85%, 1.83%, and 4.57%. In addition, the accuracy value of our proposed algorithm improves by 0.55% over DNN [33]. ID-RDRL [20] only published the accuracy and f1-score, which are 96.2% and 94.9%. The TGA model's accuracy increased by 1.63%, and the f1-score increased by 2.67%. The above analysis shows that the proposed approach has yielded better results in all four evaluation metrics than in other papers, which validates the proposed model's effectiveness in intrusion detection tasks.

Table 4. Comparison of the results with other methods.

Method	Accuracy	Precision	Recall	F1-Score
LSTM + AM [32]	96.2%	96%	96%	93%
DNN [33]	97.28%	N/A	N/A	N/A
CNN [34]	91.5%	N/A	N/A	N/A
DAE + DNN [35]	95.79%	95.38%	95.79%	95.11%
CNN + CBAM [4]	97.29%	97.26%	97.29%	96.88%
ID-RDRL [20]	96.2%	N/A	N/A	94.9%
Proposed	97.83%	97.85%	97.83%	97.57%

Figure 9 shows the confusion matrix, and Table 5 illustrates the classification report of our presented model. Combining Table 5 and Figure 9, the detection capacity of the presented model in the multi-category case can be further determined. Among the samples in the benign category, precision of 98.73%, recall and f1-score values over 99%. The precision, recall, and f1-score values for DDOS and Bot attack categories exceeded 99%, indicating that the model can detect these two attack types very well. The detection results for the Dos and the Bruteforce attack category samples show that some samples are difficult to distinguish, which may be due to the similar attack principles of these two types of

attack samples. Besides, the Web-attack samples are a minority of the total samples. The proportion of data samples in the training process is tiny, which causes the model to be biased toward the majority of samples, also known as a benign sample. Therefore, the classification of infiltration is not effective. An imbalance problem in the dataset can cause infiltration to misclassify into benign samples. This indicates that sample imbalance is still the primary problem faced in network intrusion detection, and we will focus on treating sample imbalance in future research. The problem of data imbalance can be solved in future research using oversampling methods such as random oversampling or SMOTE methods.

Table 5. Classification report.

	Precision	Recall	F1-Score	Support
Benign	98.73%	99.95%	99.34%	553,839
DDOS	100%	100%	100%	68,772
Dos	96.39%	89.30%	92.71%	553,839
Bot	99.97%	99.87%	99.92%	28,617
Bruteforce	83.70%	94.38%	88.71%	38,092
Infiltration	91.92%	23.63%	37.59%	9240
Web-attack	82.14%	49.46%	61.74%	93

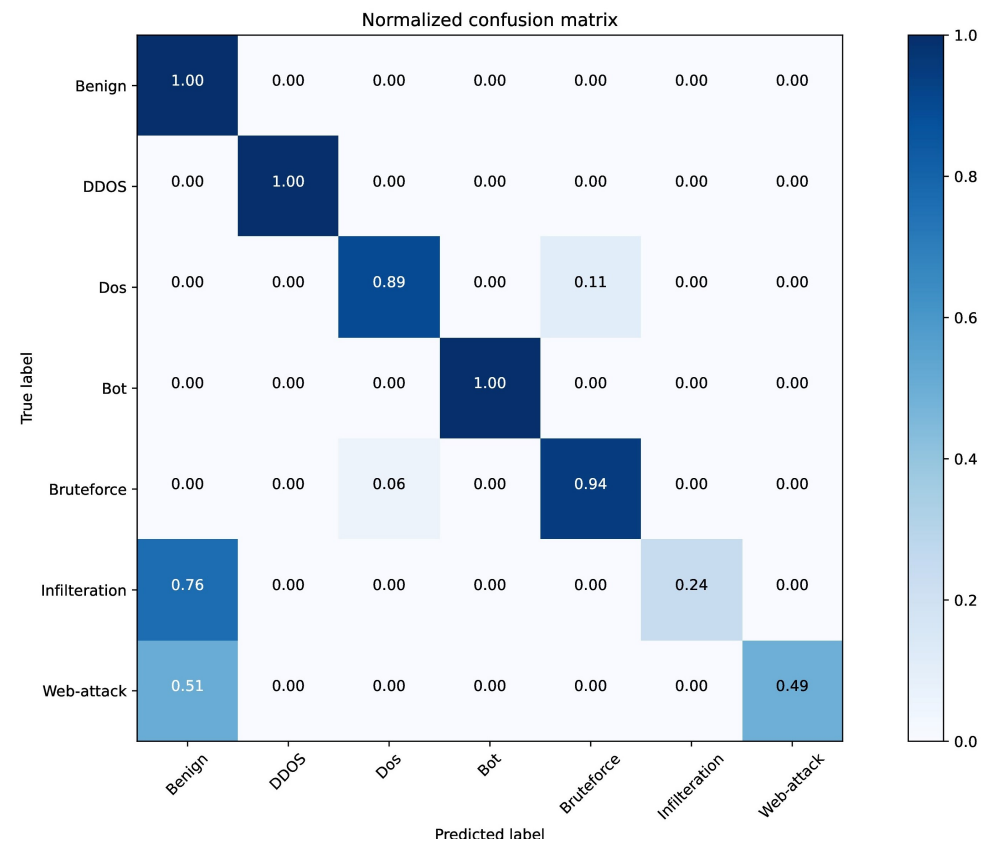


Figure 9. The confusion matrix of proposed model.

5. Conclusions

In this work, we present a model that combines TCN, BiGRU, and a self-attention mechanism to boost the performance of traffic classification. Our model utilizes TCN and BiGRU to extract local and global temporal features of network traffic simultaneously. First of all, these two features are integrated in order to broaden the feature information. Second, our model correspondingly assigns different weight parameters to the features through a self-attention mechanism, whereby the most prominent features are preserved to the utmost and then fed into the fully connected layer. Finally, the softmax layer is a classifier to identify the category to which the traffic belongs. The results of the experiments indicate

that our presented model achieves relatively good effects for multi-classification on the dataset. The value of the accuracy reached 97.83%, exceeding that of using individual TCN and BiGRU methods. However, it also has some limitations; the proportion of benign samples to malicious samples is imbalanced in the dataset, leading to the less effective classification of small samples. For future research, we will pay attention to treating sample imbalance, for example, introducing the random oversampling or SMOTE method to achieve the effect of a balanced dataset. Furthermore, we regard a more lightweight intrusion detection system as needing to be applied in the real network environment.

Author Contributions: Conceptualization, Y.S. and N.L.; methodology, Y.S.; software, Y.S.; validation, Y.S., Z.S. and H.W.; formal analysis, Y.S.; investigation, Z.S.; resources, Y.S.; data curation, H.W.; writing—original draft preparation, Y.S.; writing—review and editing, Y.S.; visualization, Z.S. and H.W.; supervision, N.L.; project administration, N.L.; funding acquisition, N.L. All authors have read and agreed to the published version of the manuscript.

Funding: This work was funded in part by National Social Science Fund of China under Grant 20&ZD293.

Data Availability Statement: This paper uses a publicly available dataset for the analysis of the results. The data can be found here: <https://www.unb.ca/cic/datasets/ids-2018.html> (accessed on 1 June 2022).

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are employed in this paper:

TCN	Temporal Convolutional Network
BiGRU	Bidirectional Gated Recurrent Unit
GRU	Gated Recurrent Unit
LSTM	Long short-term Memory Network
CBAM	Convolutional Block Attention Module
ML	Machine Learning
DL	Deep Learning
TP	True Positives
TN	True Negatives
FP	False Positives
FN	False Negatives
AM	Attention Model

References

1. Sun, P.; Liu, P.; Li, Q.; Liu, C.; Lu, X.; Hao, R.; Chen, J. DL-IDS: Extracting features using CNN-LSTM hybrid network for intrusion detection system. *Secur. Commun. Netw.* **2020**, *2020*, 8890306. [CrossRef]
2. Khraisat, A.; Gondal, I.; Vamplew, P.; Kamruzzaman, J. Survey of intrusion detection systems: Techniques, datasets and challenges. *Cybersecurity* **2019**, *2*, 20. [CrossRef]
3. Kornaropoulos, E.M.; Papamanthou, C.; Tamassia, R. The state of the uniform: Attacks on encrypted databases beyond the uniform query distribution. In Proceedings of the 2020 IEEE Symposium on Security and Privacy (SP), IEEE, San Francisco, CA, USA, 18–21 May 2020; pp. 1223–1240.
4. Liu, Y.; Kang, J.; Li, Y.; Ji, B. A network intrusion detection method based on CNN and CBAM. In Proceedings of the IEEE INFOCOM 2021-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), IEEE, Vancouver, BC, Canada, 10–13 May 2021; pp. 1–6.
5. Kim, J.; Kim, J.; Thu, H.L.T.; Kim, H. Long short term memory recurrent neural network classifier for intrusion detection. In Proceedings of the 2016 International Conference on Platform Technology and Service (PlatCon), IEEE, Jeju, Republic of Korea, 15–17 February 2016; pp. 1–5.
6. Tavallaee, M. *An Adaptive Hybrid Intrusion Detection System*; The University of New Brunswick: Fredericton, NB, Canada, 2011.
7. Pajouh, H.H.; Javidan, R.; Khayami, R.; Dehghantanha, A.; Choo, K.K.R. A two-layer dimension reduction and two-tier classification model for anomaly-based intrusion detection in IoT backbone networks. *IEEE Trans. Emerg. Top. Comput.* **2016**, *7*, 314–323. [CrossRef]

8. Mahfouz, A.M.; Venugopal, D.; Shiva, S.G. Comparative analysis of ML classifiers for network intrusion detection. In *Fourth International Congress on Information and Communication Technology: ICICT 2019*; Springer: Singapore, 2020; Volume 2, pp. 193–207.
9. Zwane, S.; Tarwireyi, P.; Adigun, M. Performance analysis of machine learning classifiers for intrusion detection. In *Proceedings of the 2018 International Conference on Intelligent and Innovative Computing Applications (ICONIC)*, IEEE, Mon Tresor, Mauritius, 6–7 December 2018; pp. 1–5.
10. Vinayakumar, R.; Soman, K.; Poornachandran, P. Applying convolutional neural network for network intrusion detection. In *Proceedings of the 2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, IEEE, Udupi, India, 13–16 September 2017; pp. 1222–1228.
11. Yan, J.; Jin, D.; Lee, C.W.; Liu, P. A comparative study of off-line deep learning based network intrusion detection. In *Proceedings of the 2018 Tenth International Conference on Ubiquitous and Future Networks (ICUFN)*, IEEE, Prague, Czech Republic, 3–6 July 2018; pp. 299–304.
12. Usama, M.; Asim, M.; Latif, S.; Qadir, J.; Ala-Al-Fuqaha. Generative adversarial networks for launching and thwarting adversarial attacks on network intrusion detection systems. In *Proceedings of the 2019 15th International Wireless Communications & Mobile Computing Conference (IWCMC)*, IEEE, Tangier, Morocco, 24–28 June 2019; pp. 78–83.
13. Kong, F.; Li, J.; Jiang, B.; Wang, H.; Song, H. Integrated generative model for industrial anomaly detection via Bidirectional LSTM and attention mechanism. *IEEE Trans. Ind. Inform.* **2021**, *19*, 541–550. [[CrossRef](#)]
14. Ansari, M.S.; Bartoš, V.; Lee, B. GRU-based deep learning approach for network intrusion alert prediction. *Future Gener. Comput. Syst.* **2022**, *128*, 235–247. [[CrossRef](#)]
15. Zhang, J.; Chen, G.; Zhao, H.; Ye, Y. Research on Network Traffic Anomaly Detection Method Based on Temporal Convolutional Network. In *Proceedings of the 2022 IEEE 8th International Conference on Computer and Communications (ICCC)*, IEEE, Chengdu, China, 9–12 December 2022; pp. 590–598.
16. Tavallae, M.; Bagheri, E.; Lu, W.; Ghorbani, A.A. A detailed analysis of the KDD CUP 99 data set. In *Proceedings of the 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications*, IEEE, Ottawa, ON, Canada, 8–10 July 2009; pp. 1–6.
17. Dhanabal, L.; Shantharajah, S. A study on NSL-KDD dataset for intrusion detection system based on classification algorithms. *Int. J. Adv. Res. Comput. Commun. Eng.* **2015**, *4*, 446–452.
18. Moustafa, N.; Slay, J. 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. Glob. Perspect.* **2016**, *25*, 18–31. [[CrossRef](#)]
19. Sharafaldin, I.; Lashkari, A.H.; Ghorbani, A.A. Toward generating a new intrusion detection dataset and intrusion traffic characterization. *ICISSp* **2018**, *1*, 108–116.
20. Ren, K.; Zeng, Y.; Cao, Z.; Zhang, Y. ID-RDRL: A deep reinforcement learning-based feature selection intrusion detection model. *Sci. Rep.* **2022**, *12*, 15370. [[CrossRef](#)]
21. Khan, M.A.; Kim, J. Toward developing efficient Conv-AE-based intrusion detection system using heterogeneous dataset. *Electronics* **2020**, *9*, 1771. [[CrossRef](#)]
22. Bai, S.; Kolter, J.Z.; Koltun, V. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv* **2018**, arXiv:1803.01271.
23. Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* **2014**, *15*, 1929–1958.
24. Zhang, D.; Yang, J.; Li, F.; Han, S.; Qin, L.; Li, Q. Landslide Risk Prediction Model Using an Attention-Based Temporal Convolutional Neural Network Connected to a Recurrent Neural Network. *IEEE Access* **2022**, *10*, 37635–37645. [[CrossRef](#)]
25. Chung, J.; Gulcehre, C.; Cho, K.; Bengio, Y. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv* **2014**, arXiv:1412.3555.
26. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [[CrossRef](#)]
27. Zheng, W.; Cheng, P.; Cai, Z.; Xiao, Y. Research on Network Attack Detection Model Based on BiGRU-Attention. In *Proceedings of the 2022 4th International Conference on Frontiers Technology of Information and Computer (ICFTIC)*, IEEE, Qingdao, China, 2–4 December 2022; pp. 979–982.
28. Li, L.; Hu, M.; Ren, F.; Xu, H. Temporal Attention Based TCN-BIGRU Model for Energy Time Series Forecasting. In *Proceedings of the 2021 IEEE International Conference on Computer Science, Artificial Intelligence and Electronic Engineering (CSAIEE)*, IEEE, Virtual Conference, 20–22 August 2021; pp. 187–193.
29. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. *Adv. Neural Inf. Process. Syst.* **2017**, *30*.
30. Child, R.; Gray, S.; Radford, A.; Sutskever, I. Generating long sequences with sparse transformers. *arXiv* **2019**, arXiv:1904.10509.
31. Hodo, E.; Bellekens, X.; Hamilton, A.; Tachtatzis, C.; Atkinson, R. Shallow and deep networks intrusion detection system: A taxonomy and survey. *arXiv* **2017**, arXiv:1701.02145.
32. Lin, P.; Ye, K.; Xu, C.Z. Dynamic network anomaly detection system by using deep learning techniques. In *Proceedings of the Cloud Computing—CLOUD 2019: 12th International Conference, Held as Part of the Services Conference Federation, SCF 2019*, San Diego, CA, USA, 25–30 June 2019; Springer: Cham, Switzerland, 2019; pp. 161–176.
33. Ferrag, M.A.; Maglaras, L.; Moschoyiannis, S.; Janicke, H. Deep learning for cyber security intrusion detection: Approaches, datasets, and comparative study. *J. Inf. Secur. Appl.* **2020**, *50*, 102419. [[CrossRef](#)]

34. Kim, J.; Kim, J.; Kim, H.; Shim, M.; Choi, E. CNN-based network intrusion detection against denial-of-service attacks. *Electronics* **2020**, *9*, 916. [[CrossRef](#)]
35. Kunang, Y.N.; Nurmaini, S.; Stiawan, D.; Suprpto, B.Y. Attack classification of an intrusion detection system using deep learning and hyperparameter optimization. *J. Inf. Secur. Appl.* **2021**, *58*, 102804. [[CrossRef](#)]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.