**Frontiers in Computing and Intelligent Systems**
ISSN: 2832-6024 | Vol. 4, No. 2, 2023

# An Intrusion Detection Method based on Fusion Neural Network

**Xin Li [1], Hong Huang [1, 2], Guotao Yuan [1], Zhaolian Wang [1], Rui Du [1]**

[1] School of computer Science and Engineering, Sichuan University of Science & Engineering, Yibin 644000, China
[2] Key Laboratory of Higher Education of Sichuan Province for Enterprise Informationalization and Internet of Things, Yibin 644000, China

**Abstract:** Aiming at the problems of class imbalance, insufficient feature learning, weak generalization ability, and representation capability in existing intrusion detection models, we propose a multi-scale feature fusion Intrusion Detection Model (MSFF). This model combines multi-scale one-dimensional convolution and bidirectional long short-term memory (LSTM) networks, and incorporates residual connections with identity mappings to address the problem of network degradation. The multi-scale convolution captures feature representations at different levels, thereby improving the expressive power of the model. The WGAN-GP algorithm is employed to augment the minority samples and balance the dataset. By performing convolution operations and extracting local window features and global features using bidirectional LSTM units, the model effectively captures temporal information and long-term dependencies. Experimental results demonstrate significant performance improvement compared to a single model. The MSFF model achieves an accuracy of 99.50% and 94.73% in binary classification experiments on the NSL-KDD and UNSW-NB15 datasets, respectively, and an accuracy of 99.50% and 83.78% in multi-class classification experiments.

**Keywords:** Intrusion Detection; Multi-scale 1D Convolution; Bidirectional Long Short-Term Memory Network; Residual Connections.

## 1. Introduction

Since the popularization of the Internet, network security has been a hot topic of concern. At the end of 2020, hackers exploited vulnerabilities in the supply chain of SolarWinds, implanting malicious software and resulting in large-scale data breaches. In late 2021, Colonial Pipeline suffered a cyberattack that led to the shutdown of its pipeline system for oil transportation, causing a surge in oil prices and public panic in the eastern United States. Also in the same year, Code masters, a UK-based company, fell victim to a ransomware attack, resulting in a significant leak of user's private data. Therefore, in the current environment, it is an urgent problem to effectively prevent and detect network attacks.

Intrusion Detection Systems (IDS) monitor network data traffic to detect abnormal flows and prevent network intrusion activities. Several traditional statistical machine learning algorithms have been applied to network intrusion detection, such as K-nearest neighbors [1-2], support vector machines [3-4], naive Bayes [5], random forests [6-7], etc. These traditional machine learning approaches often focus on the relationship between individual features and the outcome during model training, while neglecting the relationships between different features and often requiring manual feature selection. In recent years, the diversity and complexity of network traffic data have grown exponentially, rendering traditional statistical machine learning models inadequate for the current data volume.

Deep learning methods, with their superiority in handling large amounts of nonlinear data and extracting advanced features from raw data, have been widely adopted in network intrusion detection. After being processed by deep learning methods, network traffic data can yield advanced features that are potentially related to attack behaviors, such as malicious software protocols, ports, packet sizes, etc.

## 2. Related Work

Compared to traditional feature engineering methods, deep learning methods can automatically learn more discriminative features from raw data, thereby improving the accuracy and efficiency of intrusion detection. Shu Hao [8] proposed an intrusion detection method that combines Bidirectional Long Short-Term Memory (BiLSTM) and attention mechanism to improve the detection performance on NSL-KDD dataset. Compared to the single BiLSTM method, it achieved improvements in accuracy and F1 score, but the multi-class classification results were not ideal. Li Haitao et al. [9] proposed a SEMI-GRU-based traffic anomaly detection method that combines multiple layers of Bidirectional Gated Recurrent Units (GRU) and an improved feedforward network, effectively improving the F1 score but with lower accuracy. Muhuri et al. [10] proposed a combination of Long Short-Term Memory (LSTM) and Genetic Algorithm (GA) for feature selection, achieving an accuracy of 93.88% on NSL-KDD, but they did not provide the detection rate. Yu [11] combined Deep Belief Network (DBN) and Bidirectional Gated Recurrent Units (BiGRU) in an algorithm model (DBN-BiGRU) and processed the raw feature analysis files of CICIDS2017 into well-structured temporal data, but no multi-class experiments were conducted. Khan [12] proposed a Hybrid Convolutional Recurrent Neural Network Intrusion Detection System (HCRNNIDS) where Convolutional Neural Network (CNN) captures local features through convolution and Recurrent Neural Network (RNN) captures temporal features to improve the performance and prediction of the intrusion detection system, achieving improvements in precision and other metrics, but again, no multi-class experiments were conducted.

The majority of machine learning-based algorithms are built on the assumption of sufficient learning from a large amount of raw data. However, the problem of imbalanced

data distribution is prevalent [13]. Insufficient feature learning from the minority class samples leads to lower detection rates for these classes, which affects the performance of intrusion detection models and may result in highly threatening attack behaviors being misclassified as normal behavior, thereby adversely affecting the devices. Zhang [14] studied the combination of Gaussian Mixture Model-based SMOTE oversampling and clustering-based undersampling on the CICIDS2017 dataset. Although their model achieved good performance, they did not provide the F1 score. Jiang [15] combined One-Sided Selection (OSS) with Synthetic Minority Oversampling Technique (SMOTE) and incorporated a deep hierarchical model for spatial-temporal feature extraction. This method improved the metrics for the minority class samples, but the overall accuracy was not ideal. Toupas et al. [16] proposed the combination of SMOTE-ENN and Deep Neural Network (DNN) algorithm. However, they only reported the results of cross-validation and did not test on an independent test set.

In addressing the issue of imbalanced data, Generative Adversarial Networks (GANs) have unparalleled advantages. They can learn the data distribution and generate new samples from the learned distribution, making the generated samples more realistic and credible. In Shu et al. [17], GANs were used to generate adversarial samples to deceive Intrusion Detection Systems (IDS). Chen et al. [18] proposed an anti-intrusion detection autoencoder to learn the distribution of benign samples and generate real samples that attackers can use to bypass the existing IDS features. Although GANs have been used to address the issue of class imbalance, they still suffer from problems such as mode collapse and instability.

Despite the significant improvements in detection performance achieved by deep learning, existing intrusion detection methods still have some issues. (1) Insufficient feature learning and weak model representation. Previous research mostly focused on a single type of neural network. CNN-based models can accurately extract local spatial features but fail to learn temporal features, while RNN-based models can extract temporal features from data to analyze long-term dependencies but are not effective in extracting local spatial features. Moreover, RNN models can only learn unidirectional temporal features and do not fully consider the joint impact of preceding and subsequent information on the current state of traffic data. (2) Class imbalance. Imbalanced class distribution is prevalent in existing publicly available datasets. However, most researchers have focused on classifier models and paid less attention to the impact of class imbalance on the performance of minority class samples. (3) Focus on binary classification results or overall accuracy in multi-class scenarios, often overlooking the performance of individual class categories.

To address the above issues, this paper proposes a multi-scale feature fusion network intrusion detection model. The main contributions are as follows:

(1) Combining multi-scale one-dimensional convolution with BiLSTM and introducing residual connections to fully extract data features, thereby improving the model's representation and generalization capabilities.

(2) Using the WGAN-GP algorithm to balance the original dataset and improve the performance of minority class samples.

(3) Validating the effectiveness of the proposed model on the NSL-KDD and UNSW-NB15 datasets. Comparing the MSFF model with other classical machine learning and deep learning algorithms in binary and multi-class experiments.

# 3. Proposed Methodology

## 3.1. Multi-scale Feature Fusion Network Intrusion Detection Framework

To fully learn the features of network traffic data, we propose the fusion of multi-scale one-dimensional convolution (1D-CNN) and BiLSTM methods. We introduce residual connections with identity mapping between the 1D-CNN and BiLSTM layers to alleviate gradient vanishing and network degradation issues. The multi-scale 1D-CNN uses filters of different sizes to capture features at different scales. It can capture both local and larger contextual information. The overall framework of the Multi-scale Feature Fusion Intrusion Detection Model (MSFF-IDS) is shown in Figure 1, which consists of the data preprocessing module, the mixed sampling module, and the multi-scale feature fusion module.

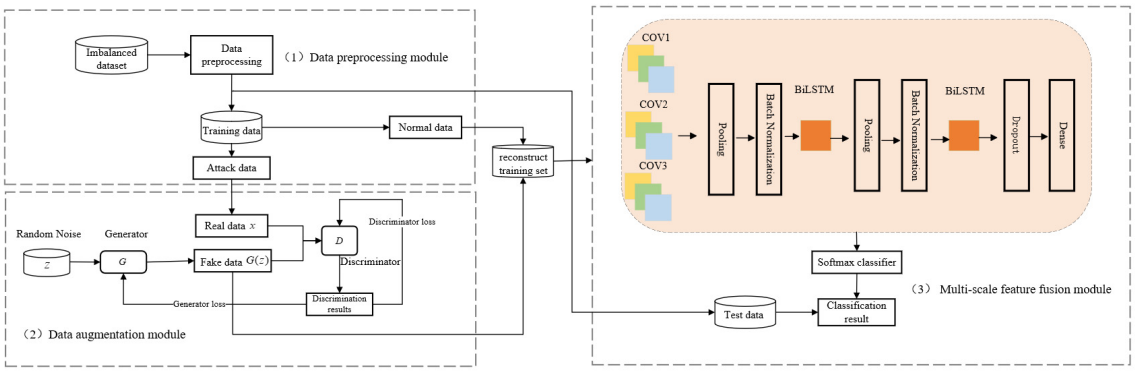


**Fig 1.** MSFF-framework

The data preprocessing module converts the character-based features in the original data into numerical features through one-hot encoding and then applies Min-Max normalization to the data with the same attributes.

In the data augmentation module, Generative Adversarial Networks (GANs) are used to generate synthetic data through the competition between a generator and a discriminator. The generator generates fake data based on input random noise, while the discriminator distinguishes between the generated fake data and real samples. The loss function is updated based on the discriminator's results. To address the issues of gradient vanishing and model collapse that can occur in GANs, the model utilizes the Wasserstein distance from Wasserstein GAN (WGAN) to measure the distance between the generated distribution and the real distribution. This replaces the JS divergence and KL divergence used in GANs. The

Wasserstein distance provides a smooth representation of the proximity between the two distributions. Additionally, to overcome the limitations and difficulties of weight clipping in WGAN, WGAN-GP introduces gradient penalty techniques to train the generator and discriminator more stably, thereby improving the performance of the generative adversarial network.

The multi-scale feature fusion module focuses on different types of features through multiple one-dimensional convolutional layers with different filters. Network traffic data may contain various types of features such as source addresses, destination addresses, protocol types, and port numbers. By designing filters of different sizes, the module specifically attends to a certain type of feature and extracts its representation. This combination enhances the model's ability to learn different types of features and captures diverse feature patterns in network traffic data. Then, two layers of BiLSTM are used to extract temporal patterns and global features from the network traffic data. Finally, the fused features are passed through a softmax classifier to obtain classification results.

The implementation process of network intrusion detection based on the Multi-scale Feature Fusion (MSFF) is shown in Algorithm 1.

Algorithm 1: Network Intrusion Detection Implementation Based on MSFF

Input: Original dataset

Output: Classification results

Step 1: Data preprocessing

1. Encode and convert the original dataset into numerical values.

2. Normalize the dataset after numerical conversion.

Step 2: Construct a balanced dataset

3. Use the SMOTE-Tomek algorithm to perform mixed sampling on the training set and reconstruct the unprocessed test set into a balanced dataset.

Step 3: Build the model

4. Use multi-scale convolution to extract local spatial features and implement parameter sharing.

5. Add pooling layers to prevent overfitting.

6. Add batch normalization layers to normalize the parameters of the previous layer and prevent training slowdown.

7. Add BiLSTM layers to learn from forward and backward temporal sequences.

8. Introduce residual connections with identity mappings between the input layer and BiLSTM layers.

9. Repeat steps 5-7 and add dropout layers and fully connected layers.

10. Use a softmax classifier for classification.

Step 4: Train the model

11.Set experimental hyperparameters and network structure parameters as described in section 3.3.

Step 5: Test the model

12. Feed the test set into the trained model and obtain the classification results.

## 3.2. Multi-scale Feature Fusion Network

The Multi-scale Feature Fusion (MSFF) network module is the core component of MSFF-IDS. In this module, the multi-scale 1D convolution is utilized to consider features of different scales, enriching the model's representation capability of input data. By introducing filters of different scales in the convolutional layer, a more comprehensive feature representation is provided, enhancing the model's expression and generalization ability. The residual connection, which passes the original feature information directly between the input layer and BiLSTM, improves feature expression and model convergence speed, while addressing the issue of gradient vanishing or exploding commonly encountered in traditional deep neural networks. The structure of the residual module is shown in Figure 2. The following sections will provide an introduction to the multi-scale convolutional layer and the identity mapping.

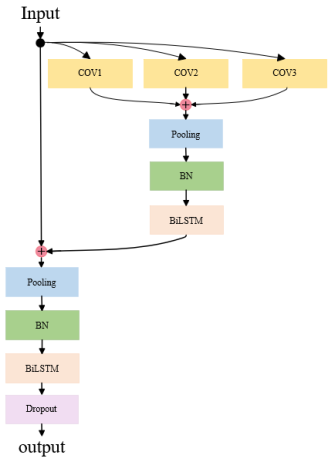


**Fig 2.** Residual module structure

### 3.2.1. Multi-scale Convolutional Layer

One-dimensional convolution primarily focuses on local details, while multi-scale one-dimensional convolution can provide a more comprehensive description of the relationships between global and local features. To effectively extract the local spatial features of network traffic data, this study adopts multiple filters of different scales (16, 32, 64) to capture features at different levels.

In multi-scale convolution, both the Concatenate function and the Add function can be used for feature fusion. The Concatenate function concatenates the feature maps of different scales along the channel dimension, while the Add function performs element-wise addition to combine the feature maps of multiple scales, preserving the information of the original features. Therefore, this study adopts the Add function for the fusion of multi-scale convolutional layers. It is important to note that when using the Add function to merge filters of different scales, they need to be adjusted to have the same number of channels.

### 3.2.2. Identity Mapping

The identity mapping, also known as skip connection, refers to directly passing the input to the output of certain layers in a neural network. It introduces shortcut connections in deep neural networks to enhance model convergence and accelerate training. The expression for the identity mapping is shown in Equation (1).

$$y = F(x) + x \tag{1}$$

Where $x$ represents the input and $F(x)$ represents the transformation operation of the network. Through the identity mapping, the network adds the input to the output of the transformation operation to obtain the final output $y$. Even if the transformation operation of the network cannot perfectly fit the data, the information from the input is still preserved and transmitted to the output through the identity mapping, retaining the original input information. Additionally, the

identity mapping does not introduce additional parameters or computational complexity, yet it helps the network learn and optimize more effectively. By utilizing the identity mapping, the network can be stacked deeper, improving the representation power and performance of the network.

# 4. Experimental Setup and Analysis

## 4.1. Dataset Description

The NSL-KDD dataset is an improvement and enhancement of the KDDCUP99 dataset, aiming to improve the accuracy and reliability of network intrusion detection. The dataset contains a total of 41 network traffic features, including 34 continuous features, 6 discrete features, and 1 binary feature. The NSL-KDD dataset is a commonly used dataset in the field of network intrusion detection. In this study, the KDDTrain+ and KDDTest+ subsets of the NSL-KDD dataset were used as the training and testing sets for the model, respectively. The class distribution and quantity are shown in Table 1.

**Table 1.** NSL-KDD categories and quantities

| | Total | Normal | Dos | Probe | R2L | U2L |
|---|---|---|---|---|---|---|
| Train | 125973 | 67343 | 45927 | 11656 | 995 | 52 |
| Test | 22544 | 9711 | 7458 | 2421 | 2754 | 200 |

The UNSW-NB15 dataset is a network intrusion detection dataset developed by the University of New South Wales in Australia. It consists of 49 network traffic features, including 45 continuous features, 3 discrete features, and 1 binary feature. UNSW-NB15 contains a wider range of attack types and network traffic features, allowing for a more comprehensive evaluation of intrusion detection models. In the experiments, the UNSW_NB15_training-set and UNSW_NB15_testing-set provided by the dataset creators were used as the training and testing sets, respectively. The class distribution and quantity are shown in Table 2.

**Table 2.** UNSW-NB15 categories and quantities

| Category | Training set | Testing set |
|---|---|---|
| Normal | 56000 | 37000 |
| Analysis | 2000 | 677 |
| Backdoor | 1746 | 583 |
| Dos | 12264 | 4089 |
| Exploits | 33393 | 11132 |
| Fuzzers | 18184 | 6062 |
| Generic | 40000 | 18871 |
| Reconnaissance | 10491 | 3496 |
| Shellcode | 1133 | 378 |
| Worms | 130 | 44 |
| Total | 175341 | 82332 |

## 4.2. Preprocessing

Data preprocessing involves performing numerical encoding and normalization on two benchmark datasets, NSL-KDD and UNSW-NB15. The specific steps are as follows:

(1) Numerical Encoding. Non-numerical features in the dataset need to be converted into numerical features. Taking the UNSW-NB15 dataset as an example, 'proto', 'state', and 'service' are non-numerical features, while the rest are numerical features. One way to handle this is by using One-Hot Encoding. 'proto' represents transaction protocol types, including 133 different types such as tcp, udp, arp. 'state' represents states and related protocols, including 13 different types such as CON, ECO, RST. 'service' includes 11 different types of services such as http, ftp, ssh, dns.

(2) Normalization. Due to the significant differences in the numerical feature values across dimensions, for example, the feature 'sybets' ranges from 0 to 129652. To scale the data to the same range and ensure that different features have equal impact on the model training results, Min-Max normalization is applied after numerical encoding. The normalization operation is represented by Equation (2).

$$X_{[i]}' = \frac{X_{[i]} - X_{\min}}{X_{\max} - X_{\min}} \tag{2}$$

Where $X_{[i]}$ represents the feature value to be normalized, $X_{\min}$ and $X_{\max}$ represent the minimum and maximum values of that feature attribute, respectively.

## 4.3. Experimental Settings

**Table 3.** Parameter Settings

| Network Architecture Layers | NSL-KDD | UNSW-NB15 |
|---|---|---|
| | Parameter Values | Parameter Values |
| InputLayer | 122,1 | 196,1 |
| Convolution1D | 16,122, same, relu | 16,196, same, relu |
| | 32,122, same, relu | 16,196, same, relu |
| | 64,122, same, relu | 16,196, same, relu |
| MaxPooling | 5 | 10 |
| BatchNormalization | - | - |
| Bidirectional (LSTM) | 61 | 98 |
| MaxPooling | 5 | 5 |
| BatchNormalization | - | - |
| Bidirectional (LSTM) | 122 | 128 |
| Dropout | 0.5 | 0.2 |
| Dense | 5 | 10 |
| activation_layer | softmax | softmax |

Experimental setup: Hardware environment: Intel(R) Core i7-8550U @ 1.80GHz Quad-Core, 16GB RAM. Software environment: Windows 10 operating system, TensorFlow 2.1, and Python 3.6.3.

As described in section 4.2, after data preprocessing, the NSL-KDD dataset has 122 dimensions, and the UNSW-NB15 dataset has 196 dimensions. Due to the need for feature fusion and the introduction of residual connections in the MSFF model, a simple sequential model structure is not sufficient. Therefore, a functional model structure is adopted.

The MSFF model consists of multiple one-dimensional convolutional layers with different scales of filters (16, 32, and 64) and two BiLSTM layers. The model also incorporates skip connections (identity mapping) between the input and BiLSTM layers. Techniques such as max pooling, batch normalization, and dropout are utilized. The specific parameter settings are shown in Table 3.

## 4.4. Evaluation Metrics

To comprehensively evaluate the performance of the MSFF model, we conducted comparative experiments using multiple metrics, including Accuracy, Precision, Recall, False Positive Rate (FPR), and F1 Score. The formulas for these metrics are as follows:

$$Accuracy = \frac{TN + TP}{TN + TP + FN + FP} \tag{3}$$

$$Precision = \frac{TP}{TP + FP} \tag{4}$$

$$Recall = \frac{TP}{TP + FN} \tag{5}$$

$$FPR = \frac{FP}{FP + TN} \tag{6}$$

$$F1 - score = \frac{2 * Precision * Recall}{Precision + Recall} \tag{7}$$

Where:

TP: True Positives (the number of correctly predicted positive samples); TN: True Negatives (the number of correctly predicted negative samples); FP: False Positives (the number of incorrectly predicted positive samples); FN: False Negatives (the number of incorrectly predicted negative samples)

## 4.5. Binary Classification Results

This section presents the experiments conducted on the publicly available network intrusion detection datasets, NSL-KDD and UNSW-NB15, for binary classification.

To validate the effectiveness of the proposed model, we compared it with three classical machine learning algorithms, two individual deep learning models, and two recently proposed deep learning models. The comparison results are shown in Table 4. The two deep learning models are BCNN-DFS proposed in reference [19] and SEMI-GRU proposed in reference [9].

**Table 4.** Binary classification comparison of models

| Model | NSL-KDD | | | | | UNSW-NB15 | | | |
|---|---|---|---|---|---|---|---|---|---|
| | Acc/% | Pre/% | Recall/% | F1/% | | Acc/% | Pre/% | Recall/% | F1/% |
| SVM | 87.33 | 87.96 | 87.33 | 85.01 | | 62.00 | 62.00 | 60.00 | 60.98 |
| RF | 80.45 | 97.05 | 67.72 | 70.77 | | 80.45 | 97.05 | 67.72 | 79.77 |
| J48 | 81.53 | 97.14 | 69.61 | 81.10 | | 76.95 | 70.05 | 99.98 | 82.69 |
| CNN | 93.80 | 98.80 | 93.40 | 96.02 | | 91.20 | 87.53 | 96.17 | 91.59 |
| LSTM | 94.26 | 99.05 | 90.79 | 94.74 | | 89.90 | 88.90 | 97.30 | 92.90 |
| BCNN-DFS | 90.14 | 90.00 | 90.00 | 90.00 | | 89.26 | 89.00 | 89.00 | 89.00 |
| SEMI-GRU | 92.18 | 93.74 | 92.43 | 93.08 | | 88.11 | 96.29 | 81.56 | 88.13 |
| MSFF | 99.50 | 99.32 | 99.59 | 99.45 | | 94.73 | 94.51 | 91.20 | 92.83 |

From Table 4, it can be observed that the proposed model in this paper performs well compared to other models. It achieves an accuracy of 99.50%, precision of 99.32%, recall of 99.59%, and F1 score of 99.45%. On the other hand, RF, SVM, and J48 exhibit lower accuracy with values of 80.45%, 87.33%, and 81.53%, respectively. This is because they are unable to fully learn the data features. When compared to the individual models CNN and LSTM, the MSFF model outperforms them in all performance metrics except for a slightly lower recall in the UNSW-NB15 dataset, indicating the MSFF model's ability to effectively extract network data features. Compared to the SEMI-GRU model, the MSFF model demonstrates an improvement of 7.32% in accuracy, 5.58% in precision, 7.16% in recall, and 6.37% in F1 score on the NSL-KDD dataset. Similarly, on the UNSW-NB15 dataset, the MSFF model shows an enhancement of 6.62% in accuracy, 9.64% in recall, and 4.70% in F1 score. Overall, the MSFF model exhibits significant improvements in all the evaluated metrics.

## 4.6. Multi-classification Results

This section presents the experiments conducted on the publicly available network intrusion detection datasets, NSL-KDD and UNSW-NB15, for multi-class classification.

To validate the superiority of the MSFF model in improving the performance of minority classes, we compared

it with the SVM model, the ID-GAN model proposed in [19], and the STACON-ATT model proposed in [20] on the NSL-KDD dataset. The experimental results are shown in Table 5.

**Table 5.** Multi-class performance comparison of different models on NSL-KDD

| Category | Evaluation metrics | SVM | ID-GAN | STACON-ATTN | MSFF |
|---|---|---|---|---|---|
| Total | Accuracy/% | 77.36 | -- | 80.76 | 99.50 |
| | FPR/% | 1.97 | -- | 3.11 | 0.41 |
| Normal | Precision/% | 67.00 | 75.10 | 75.00 | 99.32 |
| | Recall/% | 98.00 | 96.64 | 97.00 | 99.55 |
| | F1/% | 80.00 | 84.52 | 85.00 | 99.43 |
| DoS | Precision/% | 97.00 | 96.18 | 97.00 | 99.82 |
| | Recall/% | 82.00 | 83.70 | 78.00 | 99.88 |
| | F1/% | 89.00 | 89.50 | 86.00 | 99.85 |
| Probe | Precision/% | 85.00 | 82.75 | 70.00 | 99.36 |
| | Recall/% | 65.00 | 84.01 | 87.00 | 99.19 |
| | F1/% | 74.00 | 83.38 | 78.00 | 99.27 |
| R2L | Precision/% | 95.00 | 90.85 | 93.00 | 95.14 |
| | Recall/% | 8.00 | 34.60 | 32.00 | 89.62 |
| | F1/% | 15.00 | 50.12 | 48.00 | 92.30 |
| U2R | Precision/% | 0.00 | 62.00 | 34.00 | 52.63 |
| | Recall/% | 0.00 | 15.50 | 7.00 | 83.33 |
| | F1/% | 0.00 | 24.80 | 12.00 | 64.51 |

Table 5 shows that the overall accuracy of the proposed model in this paper reaches 99.50%, with a false positive rate of 0.41%. Compared to other models, the proposed model achieves the highest precision, recall, and F1 score in the Normal, DoS, Probe, R2L, and U2R categories. The precision rates are 99.32%, 99.82%, 99.36%, 95.14%, and 52.63%, respectively, while the recall rates are 99.55%, 99.88%, 99.19%, 89.62%, and 83.33%, respectively. The F1 scores are 99.43%, 99.85%, 99.27%, 92.30%, and 64.51%, respectively. Based on the comparison results, the MSFF model demonstrates excellent performance in recognizing minority classes.

**Table 6.** Comparison of F1 scores (%) of different models on UNSW-NB15

| Category | RF | AlexNet | CNN | LSTM | MSFF |
|---|---|---|---|---|---|
| Normal | 85.86 | 86.85 | 85.20 | 88.41 | 91.85 |
| Generic | 98.98 | 95.26 | 97.31 | 98.48 | 98.89 |
| Exploits | 63.37 | 66.95 | 67.73 | 58.19 | 74.23 |
| Fuzzers | 28.13 | 26.59 | 34.68 | 28.31 | 65.02 |
| DoS | 19.04 | 24.16 | 5.33 | 6.10 | 31.76 |
| Reconnaissance | 58.11 | 53.37 | 75.21 | 56.65 | 83.91 |
| Analysis | 6.93 | 4.88 | 2.09 | 2.91 | 15.29 |
| Backdoor | 4.33 | 3.62 | 2.53 | 2.90 | 15.81 |
| Shellcode | 32.59 | 23.84 | 24.64 | 18.41 | 61.26 |
| Worms | 6.05 | 3.17 | 1.58 | 1.95 | 60.21 |

To verify the applicability of the MSFF model, more in-depth comparative experiments were conducted on the UNSW-NB15 dataset. The F1 score, which combines precision and recall, effectively evaluates the performance of the model. Table 6 presents the comparison of F1 scores for different categories in the dataset under different models. From Table 6, it can be observed that the categories Analysis, Backdoor, and Worms have relatively smaller F1 scores compared to other categories like Normal and Generic. Compared to other models, the MSFF model shows significant improvement in F1 scores for the Analysis, Backdoor, Worms, and Shellcode categories, ranging from 8.36% to 13.20%, 11.48% to 13.28%, 54.16% to 58.63%, and 28.67% to 42.85%, respectively. This indicates that the MSFF model effectively enhances the detection performance for minority class samples, demonstrating the applicability of this approach.

# 5. Summary

In response to the insufficient feature learning, representation capability, and generalization ability of existing intrusion detection models, a novel approach is proposed in this work. It combines multi-scale 1D-CNN and BiLSTM to extract fused features from the data, while incorporating residual connections with identity mapping to enhance model performance. Moreover, considering the problem of data imbalance in existing public network intrusion datasets, the WGAN-GP algorithm is employed to improve the detection performance for minority class samples. The proposed MSFF model is validated on the NSL-KDD and UNSW-NB15 datasets. The results demonstrate that the MSFF model outperforms several classical machine learning methods and popular deep learning approaches in both binary

and multi-class experiments, achieving high accuracy, precision, recall, and F1 scores. In future work, further exploration will be conducted to investigate algorithms that are better suited to address data imbalance and enhance the detection performance for anomalous traffic. Additionally, the use of updated and more realistic datasets will be explored for validation purposes.

# Acknowledgments

Sichuan Science and Technology Program Project (2020YFG0151); Key Laboratory Project of Enterprise Informatization and IoT Measurement and Control Technology in Sichuan Province (2021WZY01).

# References

[1] Wazirali R. An improved intrusion detection system based on KNN hyperparameter tuning and cross-validation[J]. Arabian Journal for Science and Engineering, 2020, 45(12): 10859-10873.

[2] Liu G, Zhao H, Fan F, et al. An enhanced intrusion detection model based on improved kNN in WSNs[J]. Sensors, 2022, 22 (4): 1407.

[3] Mohammadi M, Rashid T A, Karim S H T, et al. A comprehensive survey and taxonomy of the SVM-based intrusion detection systems[J]. Journal of Network and Computer Applications, 2021, 178: 102983.

[4] Wang H, Gu J, Wang S. An effective intrusion detection framework based on SVM with feature augmentation[J]. Knowledge-Based Systems, 2017, 136: 130-139.

[5] Tabash M, Abd Allah M, Tawfik B. Intrusion detection model using naive bayes and deep learning technique[J]. Int. Arab J. Inf. Technol., 2020, 17(2): 215-224.

[6] Balyan A K, Ahuja S, Lilhore U K, et al. A hybrid intrusion detection model using ega-pso and improved random forest method[J]. Sensors, 2022, 22(16): 5986.

[7] Anton S D D, Sinha S, Schotten H D. Anomaly-based intrusion detection in industrial data with SVM and random forests[C]//2019 International conference on software, telecommunications and computer networks (SoftCOM). IEEE, 2019: 1-6.

[8] Shu Hao, Wang Chen, Shi Yan. Intrusion Detection Based on BiLSTM and Attention Mechanism[J]. Computer Engineering and Design, 2020, 41(11): 3042-3046.

[9] Li Haitao, Wang Ruimin, Dong Weiyu, Jiang Liehui. A Semi-Supervised Network Traffic Anomaly Detection Method based on GRU[J]. Computer Science, 2023, 50(03): 380-390.Kumar J, Goomer R, Singh A K. Long short term memory recurrent neural network (LSTM-RNN) based workload forecasting model for cloud datacenters[J]. Procedia Computer Science, 2018, 125: 676-682.

[10] Yu X, Li T, Hu A. Time-series Network Anomaly Detection Based on Behaviour Characteristics[C]. In 2020 IEEE 6th International Conference on Computer and Communications (ICCC). Chengdu, China: IEEE, 2020: 568-572.

[11] Khan M A. HCRNNIDS: hybrid convolutional recurrent neural network-based network intrusion detection system[J]. Processes, 2021, 9(5): 834.

[12] Rodda S, Erothi U S R. Class imbalance problem in the network intrusion detection systems[C]//2016 international conference on electrical, electronics, and optimization techniques (ICEEOT). Ieee, 2016: 2685-2688.

[13] Zhang H, Huang L, Wu C Q, et al. An effective convolutional neural network based on SMOTE and Gaussian mixture model for intrusion detection in imbalanced dataset[J]. Computer Networks, 2020, 177: 107315.

[14] Jiang K, Wang W, Wang A, et al. Network intrusion detection combined hybrid sampling with deep hierarchical network[J]. IEEE access, 2020, 8: 32464-32476.

[15] Toupas P, Chamou D, Giannoutakis K M, et al. An intrusion detection system for multi-class classification based on deep neural networks[C]//2019 18th IEEE International Conference on Machine Learning And Applications (ICMLA). IEEE, 2019: 1253-1258.

[16] Al-Turaiki I, Altwaijry N. A convolutional neural network for improved anomaly-based network intrusion detection[J]. Big Data, 2021, 9(3): 233-252.

[17] Shu D, Leslie N O, Kamhoua C A, et al. Generative adversarial attacks against intrusion detection systems using active learning [C]// Proceedings of the 2nd ACM workshop on wireless security and machine learning. 2020: 1-6.

[18] Chen J, Wu D, Zhao Y, et al. Fooling intrusion detection systems using adversarially autoencoder[J]. Digital Communications and Networks, 2021, 7(3): 453-460.

[19] Yin Chuanlong. Research on Network Anomaly Detection Technology based on Deep Learning [D]. Information Engineering University of Strategic Support Force, 2018.

[20] Dong Weiyu, Li Haitao, Wang Ruimin, Ren Huajuan, Sun Xuekai. Network Traffic Anomaly Detection Model based on Stacked Convolutional Attention[J]. Computer Engineering, 2022, 48(09): 12-19.