2025 International Conference on Intelligent Computing
July 26-29, Ningbo, China
https://www.ic-icc.cn/2025/index.php

Performance Optimization of Imbalanced Intrusion Detection Data Classification Based on Voting Approach

Guannan Wen[1] and Zhenzhou An[2(✉)]

[1] National Pilot School of Software, Yunnan University, Yunnan 650000, China
[2] Honghe University, Honghe 661400, China

Abstract. Intrusion detection systems (IDSs) often suffer from the category imbalance problem, i.e., malicious traffic is much less than normal traffic, which results in inefficient system detection. This paper proposed a data generation technique incorporating Pearson's correlation coefficient to solve this problem. At the same time, feature selection based on chi-square distribution and integrated learning based on voting method are also used for model optimization. In the data generation stage, the Pearson correlation coefficient-based sample similarity calculation was introduced to improve the Borderline-SMOTE method to generate high-quality data and reduce the negative impact of low-quality samples on the classification model. Different experiments were conducted for multiple machine learning methods in the model optimization phase and selected the most effective combination. Experiments on three public datasets, UNSW-NB15, NSL-KDD, and CICIDS-2017, proved the effectiveness of the method, especially in the detection of a few categories, which achieved significant improvement, especially in the UNSW-NB15 dataset, the F1 scores of the few categories of Analysis and Backdoor had 42% and 39% improvement, respectively. In addition, a new evaluation metric, Mean Category Accuracy (MCA), was proposed, which provides a more balanced assessment of the detection performance of all attack types.

Keywords: Intrusion detection, Machine learning, Ensemble learning.

1 Introduction

Intrusion detection is a technology that addresses network intrusions and other security issues by analyzing transmitted packets to differentiate between malicious and normal traffic. Network administrators can use this technology as a basis for devising appropriate countermeasures. Although intrusion detection systems can detect malicious traffic, their performance in modern neural network-based systems is heavily influenced by training data distribution and in the actual network environment, malicious traffic and normal traffic ratio is often very different, that is, today's intrusion detection system is faced with the class imbalance problem, addressing class imbalance effectively is critical for enhancing intrusion detection performance. Solving the class imbalance problem is an important breakthrough to improve intrusion detection capability. At the same time, with the development of artificial intelligence, intrusion

detection is also progressing after the introduction of machine learning. Detection models with different structures are constantly proposed, and the reasonable arrangement of the model structure is also an important influencing factor in improving the detection capability.

How to evaluate the model while training is also an important task related to the steps of evaluating the model and then optimizing the model. In intrusion detection, most researchers use the overall accuracy for model evaluation, but as mentioned above, the class imbalance problem is prevalent in intrusion detection, and using the overall accuracy for evaluation will lead to the inability to perceive the detection capability of the model correctly.

In order to address the above challenges, our research starts with solving the class imbalance problem, intrusion detection models, and how to evaluate an intrusion detection system more efficiently. Furthermore, the main contributions are:

- The generated data is cleaned by combining data up-sampling methods and data similarity algorithms, and the generated data is not completely used. Through the process, the poorer quality data in the generated data can be removed, thus removing its negative impact on the classification model.
- We use an ensemble learning model based on the voting method for training and detection, which eventually significantly improves a few classes' classification F1 scores of Analysis and Backdoor.
- We propose an attack-type-agnostic evaluation metric (MCA), which improves fairness by achieving 78.2% mean accuracy (vs. 63.7% in Stacked Ensemble [1]). With this evaluation method, the detection system's effectiveness against a wide variety of attack types can be more comprehensively reflected.

2 Related work

2.1 Class Imbalance Solutions

Although network attacks are common in cyberspace, they are still relatively few compared to normal network traffic data, especially for some infrequent attacks. For example, in the NSL-KDD dataset, the normal traffic data occupies 51.88% of the total data, while the corresponding DoS, Probe, R2L, and U2R attack data accounts for 35.95%, 9.48%, 2.61%, and 0.08%, respectively. The class imbalance problem is a common challenge in machine learning and artificial intelligence, and its occurrence can lead to the model not learning enough for a smaller number of classes, thus affecting the effectiveness of the final model.

To address class imbalance problem, the simpler methods are random up-sampling (ROS) and random down-sampling (RUS). However, the implementation of this method is simple, but its shortcomings are also very obvious: up-sampling to get the dataset of some samples repeated, the training of the model will have overfitting; down-sampling due to the loss of the data will make the model of the overall pattern of training is not sufficient. Chawla et al. [2] proposed artificial minority oversampling (SMOTE) for the class imbalance problem, which combined with random sampling methods to obtain good classification results in the case of class imbalance. Ma et al. [3] by

introducing the SMOTE technique into intrusion detection and combining it with reinforcement learning to achieve autonomous learning of the model and alleviate class imbalance.

Although the proposed SMOTE technique relieves the class imbalance problem, its drawbacks are obvious: on the one hand, this method increases the likelihood of overlap between classes, and on the other hand, the generated samples do not provide useful information. Generative Adversarial Network (GAN) is a data generation model proposed by Goodfellow et al. [4], who optimize each other by combining the generative and discriminative models in a mutual confrontation. The proposed model is a good direction for solving the class imbalance problem, and there are many research results generated by using the improvement of GAN in some recent articles. For example, Park et al. [5] have greatly improved the classification effect of minority classes by utilizing the Boundary Equilibrium Generative Adversarial Network (BEGAN) to train and generate data for the minority classes and combining it with neural networks. Meanwhile, various variants of GAN have been applied in the field of intrusion detection, such as GMM-WGAN [6], CTGAN [7], CWVAEGAN [8], AE-WGAN [9], and various variants [10,11]. Among them, the CWVAEGAN proposed by He et al. [8] incorporates the variational autoencoder (VAE) into the generator phase of the generative adversarial network, which makes the network further robust to noise by obtaining a probabilistic representation of the encoded data. Wang et al. [12] utilized the samples to train a twin self-encoder and used its input as the input to the network in 2024 to design S2CGAN. The authors also utilized similarity features as auxiliary information to generate the data. In the same year, Bai et al. [13] extended the WGAN framework by integrating a classifier that handles all traffic data types to design SC-GAN, which significantly improved the accuracy of intrusion detection with the generated data.

As opposed to using various methods to generate training data, some methods address the class imbalance problem by starting from the model itself. Thakkar et al. [14] applied ensemble learning to intrusion detection by applying multiple neural network models to the data in the training phase and then in the final classification phase by using a majority voting mechanism to classification. Siddique et al. [15] also performed well on the NSL-KDD, UNSW-NB15 and CICIDS2017 datasets using sampleless and self-supervised learning. At the loss function level, Zhang [16], Imrana [17], and others have also made several contributions to the solution of the class imbalance problem by improving the focus loss and controlling the model training in the backpropagation phase.

To summarize, most of the existing methods for solving the class imbalance problem take the approach of expanding the dataset through certain strategies such as up-sampling, SMOTE, or neural networks but ignore whether these artificially generated data fit the original data well enough, in other words, whether these generated data bring a positive impact on the model. How to solve this problem may be an important research direction.

2.2 Machine Learning-based

In the early developmental stages of intrusion detection, some traditional machine learning methods were commonly used to detect network data. Mukkamala et al. [18] and Ben et al. [19] firstly applied machine learning methods such as support vector machines and decision trees to the field of intrusion detection at the beginning of the 20th century, more and more research results have emerged. Lin et al. citelin2015cann proposed a cluster centroid and nearest neighbour (CANN) feature representation method by improving the feature representation method, which is based on calculating the distance between each data and the cluster centroid and the distance between the data's nearest neighbours in the cluster respectively, and using the one-dimensional distance feature formed by the sum of the two distances for K nearest neighbour classification, and finally obtaining the best results on the KDD- Cup99 dataset with better results than KNN, SVM.

Combinatorial machine learning usually performs better than individual machine learning methods in intrusion detection; Aburomman et al. [20] optimally obtain the optimal weights of various machine learning algorithms in the combinatorial machine learning by introducing the Particle Swarm Optimization algorithm (PSO) and using the PSO algorithm to control the weights of combinatorial machine learning. Finally, the Weighted Majority Algorithm (WMA) is tested to defeat the Weighted Majority Algorithm (WMA) on five randomly sampled subsets on the KDD99 dataset. Machine learning-based intrusion detection is applied to a wide variety of networks; Kolias et al. [21] utilized multiple machine learning methods on a dataset in a wireless network environment to provide a solid experimental foundation for the security of protocol iterations in wireless networks. Wathiq et al. [22] provided a solid experimental foundation for the security of protocol iterations in wireless networks by combining the Support Vector Machines and Extreme Learning Machines to form a multi-layer hybrid intrusion detection model; this model can have good detection of existing attacks as well as unknown attacks. Wathiq also uses a modified K-means algorithm to reduce the dataset, which shortens the training time and improves the detection performance. Experiments on the KDD Cup99 dataset showed that the model achieved the best results.

Although machine learning-based detection models have achieved good results in the field of intrusion detection, machine learning models are very dependent on the setting of hyperparameters; good initial hyperparameter settings can often obtain better results, how to find the appropriate hyperparameters is a time-consuming and laborious work. Therefore, in intrusion detection, such as high real-time requirements of the scene, how to shorten the speed of model training and inference is also an urgent problem to be solved.

2.3 Neural Network-based

Recent studies show that deep learning-based IDSs detect attacks more effectively in IoT networks. For example, Alkahtani et al. [23] introduced in their research an intrusion detection framework for IoT networks that employs three deep learning approaches: convolutional neural networks (CNNs), long-short-term memory networks (LSTMs), and a hybrid model combining CNNs and LSTMs (CNN-LSTM). When

tested on the IoTID20 dataset, the framework showed that LSTM achieved the highest accuracy (99.20%), followed by the hybrid model CNN-LSTM (98.0%), and lastly, CNN (96.60%). However, the study used an unbalanced dataset, which might have yielded better results if it had been balanced. Another study by Alqahtani et al. [24] proposed a novel hybrid optimization LSTM approach, where CNN extracts spatial and temporal relevant features from IoT datasets. In contrast, LSTM is used to predict intrusion attacks. The model incorporates the firefly swarm optimization technique for feature selection to reduce computational overhead. When evaluated on two popular network intrusion datasets, UNSWNB15 and NSL-KDD, the deep learning model performs excellently with a prediction accuracy 98.89%. However, this study used an unbalanced dataset in the pilot phase, and it is believed that better results would have been obtained if data balancing treatment was also combined. Another study by Alqahtani et al. [24] proposed a novel hybrid optimization LSTM approach where CNN extracts spatially and temporally relevant features from IoT datasets. In contrast, LSTM is used to predict intrusion attacks. In order to reduce the computational overhead in the feature selection phase, the model incorporates the firefly swarm optimization algorithm. In the final test phase, the model shows excellent performance for both datasets, UNSW-NB15 and NSL-KDD, with a prediction accuracy of 98.89%. However, it performs poorly in recognizing novel attack types due to overfitting. In addition, when the model was evaluated in a real-time IoT environment, its training time was long; hence, the overall performance was poor.

Meanwhile, Abdel-Basset et al. [25] proposed a multi-scale residual temporal convolution module in a semi-supervised deep learning intrusion detection (SS-Deep-ID) approach. The module aims to learn spatio-temporal features and incorporates an attention mechanism to extract local features. This SS-DeepID model is evaluated on two network intrusion datasets, CI-CID2017 and CI-CID2018, and the results show that it improves detection efficiency and accuracy. Similarly, the model is time-consuming in detecting real-time traffic data.Xiao et al. [26] employed two dimensionality reduction techniques, namely Principal Component Analysis (PCA) and Autoencoder (AE), to evaluate the impact of reducing the feature set on the classification of intrusion attacks. The obtained feature set is fed into a Convolutional Neural Network (CNN) model for attack classification. The model was tested on the standard KDD-CUP99 dataset, and the results showed that the accuracy of AE (0.940%) was higher than PCA (0.930%). However, the detection accuracy of this model was relatively low compared to other existing models. In contrast, Safiullah et al. [27] proposed an intrusion detection framework using deep convolutional neural networks (CNNs) for binary and multiple classification. The framework was tested on the IoTID20 dataset with different batch sizes, and performance was evaluated using standard metrics. The model achieved up to 99% accuracy, F1 score, precision, and recall in the binary classification task but lower values in the multi-classification task, ranging from 70% to 97%.

Neural networks are a growing technology with the development of big data, which means that neural networks rely on a large amount of data to get good results. Intrusion detection, which is plagued by class imbalance problems, does not seem suitable for using the technology. Therefore, neural network models often need to be used with data balancing techniques. At the same time, in many research results, the binary

classification results are always better than the results of multi-classification, which also reflects that the generation of data is not a perfect fit to the original data to solve the quality of the generated data is an important prerequisite for improving the performance of neural networks.

3 Methodology

3.1 Data pre-processing

In network data traffic, data is composed in an intricate form, where categorical features such as 'proto', 'service', and 'state'. For similar cases, we use label encoding for these categorical features, which maps each category to an integer value incremented from 0 so that each data item has a unique numeric representation. For example, for the 'proto' field, TCP corresponds to 0, UDP to 1. Through the label encoding process, these categorical features are transformed into features that can be recognized and processed by the model.

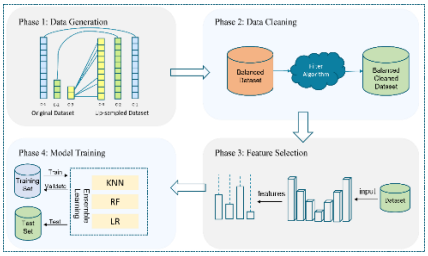

Fig. 1. Framework of the Proposed Method.

Data preprocessing is not just about numerical encoding the data because there may be special values such as Null, Nan (Not a number) or Inf in the raw network data. Since the model fails to recognize these data, we discard them.

3.2 Data generation

After data preprocessing, to solve the problem of class imbalance, we improved BorderlineSMOTE by introducing the Pearson correlation coefficient for data similarity calculation.

SMOTE method is an oversampling technique proposed by Nitesh Chawla et al. [2] in 2002 and widely used, which generates samples by calculating inter-sample differences after obtaining the closest neighbouring samples.

Based on the SMOTE method, Hui Han et al. [28] argued that samples located at decision boundaries are more likely to be misclassified, so they proposed an improved scheme, BorderlineSMOTE. This method generates samples by identifying borderline samples.

Although the SMOTE method effectively solves the class imbalance problem, it will inevitably affect the dataset in the sample generation. Therefore, we use the Pearson correlation coefficient-based sample similarity calculation method to constrain the generated data after using the Borderline-SMOTE method to generate samples from the original data. Firstly, we select a sample from the generated data set and find its k

nearest neighbours; subsequently, we calculate the similarity between this sample and its k neighbours and compare the calculated results with the threshold value set in advance. Subsequently, the similarity between this sample and its k neighbouring samples is calculated. The calculated result is compared with the threshold value set beforehand. If it is smaller than the threshold value, the non-compliant sample is discarded, and vice versa; it is retained. After generating the samples more compact between classes and more obvious features, this operation makes the dataset more compact, so it has better results in the subsequent data classification stage.

Algorithm. 1. Borderline-SMOTE-Pearson Correlation Filter

Input: Original dataset D, similarity threshold θ
Output: Filtered dataset D_{fil}

```
Dsyn←BorderlineSMOTE(D)
Dcom←Concatenate(D,Dsyn)
Initialize empty list Dfil
for xi in Dcom do
    choose k nearest neighbors to Dsubset
    satisfied←True
    for si in Dsubset do
        ri←caculatePearsonCorrelation(xi,si)
        if ri < θ then
            satisfied ← False
            break
        end if
    end for
    if satisfied then
        append xi to Dfil
    end if
end for
return Dfil
```

3.3 Feature selection

After the above data preprocessing and data generation stages, we obtain a class-balanced dataset. Subsequently, we enter the third stage of the method - feature selection- and choose the feature selection method based on the chi-square test. The chi-square test is a nonparametric test, and nonparametric tests can use sample data to make inferences about the overall distribution pattern when the overall variance is poorly known or unknown. Since network traffic is diverse in a network environment, knowing the distribution pattern in advance is impossible. Therefore, the chi-square test can be applied in this situation.

The chi-square test is a commonly used hypothesis testing method, which obtains the $\chi 2$ value (1) by processing the observed and expected frequencies to determine whether the data features are correlated or not to achieve the purpose of feature selection further.

$$\chi^2 = \sum \frac{(Observation\ frequency - Expected\ frequency)^2}{Expected\ frequency} = \sum_{i=1}^{r} \sum_{j=1}^{c} \frac{(O_{i,j} - E_{i,j})^2}{E_{i,j}} \quad (1)$$

where $O_{i,j}$ is the value of the cell in the observed frequency table, $E_{i,j}$ is the value of the cell in the expected frequency table, r is the number of rows, and c is the number of columns.

3.4 Evaluation criteria

When dealing with multicategorization tasks with class imbalance, traditional evaluation metrics such as overall accuracy may mask performance problems in minority classes due to the presence of majority classes. In this case, even though the model performs well on the majority class, its prediction ability on the minority class may be poor, leading to poor modeling in practical applications. Therefore, to assess the model's performance on the class imbalance problem in a more comprehensive and fair way, this study adopts a new evaluation scheme: calculating the accuracy of each class and finding its mean, i.e., mean class accuracy (MCA).

Definition and calculation of mean class accuracy:

Mean class accuracy is an evaluation metric for multi-categorization tasks considering the class imbalance problem. It calculates the model's accuracy in each category and averages these accuracies to get the final evaluation value. The specific steps are as follows Eq. (2):

- Category Accuracy Calculation: Calculate the model's accuracy for each category. This is calculated by taking the ratio of the number of samples correctly predicted by the model for that category to the total number of samples in that category.
- Finding the Mean: Add up the accuracies for all categories and divide by the total number of categories to get the category accuracy mean.

$$MCA = \frac{\sum_{i=1}^{k} Acc_i}{k} \quad (2)$$

where Acc_i denotes the accuracy of the samples of the ith class, and k denotes the number of classes in the entire dataset. The advantage of using this metric is that it gives equal weight to each category regardless of its sample size, thus avoiding the excessive influence of the majority of categories on the overall evaluation results. While using the mean value of category accuracy to evaluate the experimental results, this paper also employs the commonly used precision (Pre), recall (Rec) and F1 score to evaluate the results.

In order to argue for the effect of weights in MCA, we conduct the following hypothetical experiment. We assume the existence of two types of data, A and B, where A is the minority class, and B is the majority class, in the case where the weights are biased towards the minority and majority classes and different accuracy rates, respectively:

As shown in the table below, the final results are more balanced when the weights of the minority and majority classes are the same, and when the weights are somewhat skewed, there may be extreme values that lead to an inability to evaluate the final results well. As a result, we used MCA with the same weights to evaluate the experimental results in this paper.At the same time, a point that cannot be ignored in the field of

intrusion detection is that the majority class tends not to be very low (usually above 80%), and this was confirmed in the subsequent experimental phase. Therefore, it is reasonable for MCA to choose equal weights.

Table 1. Weighting Argumentation Experiments.

Accuracy of A,B	Weight(1,5)	Weight(5,1)	Weight(1,1)
100%,20%	33.3%	86.6%	60%
0%,100%	83.3%	16.7%	50%
100%,100%	100%	100%	100%
20%,100%	86.6%	33.3%	60%

Table 2. Distribution of Datasets

Dataset	**Category**	**Original Samples**	**Balanced Samples**
UNSW-NB15	Normal	93000	93000
	Generic	58871	58871
	Exploits	44525	44525
	Fuzzers	24246	25767
	DoS	16353	25767
	Reconnaissance	13987	25767
	Analysis	3677	25767
	Backdoor	2329	25767
	Shellcode	1511	25767
	Worms	174	25767
CIC-IDS-2017	BENIGN	2271312	2271312
	Dos Hulk	230124	230124
	PortScan	158804	188524
	DDoS	128025	188524
	DoS GoldenEye	10293	188524
	FTP-Patator	7935	188524
	SSH-Patator	5897	188524
	DoS slowloris	5796	188524
	DoS Slowhttptest	5499	188524
	Bot	1956	188524
	Web Attack Brute Force	1507	188524
	Web Attack XSS	652	188524
	Infiltration	36	188524
	Web Attack Sql Injection	21	188524
	Heartbleed	11	188524
NSL-KDD	Normal	93000	93000
	Generic	58871	58871
	Exploits	44525	44525
	Fuzzers	24246	25767
	DoS	16353	25767

4 Experimental results

4.1 Datasets

In our previous work, we introduced how to perform data cleaning, data generation, and feature extraction for network traffic data. Next, we validate our proposed method using three public datasets, which are UNSW-NB15, NSL-KDD and CIC-IDS-2017.

For these three datasets, we have performed up-sampling using the method proposed in the paper, and the final results as shown in Table 2.

Table 3. Configuration Specifications of Voting Ensemble Architectures.

Ensemble ID	Component Models	Parameter Configuration
VE-1	KNN, RF, DNN	KNN: k=5; RF: n_estimators=100; DNN: 3 layers (256-128-64), dropout=0.3
VE-2	KNN, RF, DNN, SVM, NB	+ SVM: RBF kernel (γ=0.1, C=1.0);+ NB: Gaussian var_smoothing=1e-9
VE-3	KNN, RF, DNN, SVM, GD	+ GD: learning_rate=0.01, n_iter=500
VE-4	KNN, RF, LR	+ LR: L2 penalty (C=1.0), max_iter=1000
VE-5	KNN, RF, LR, SVM	SVM parameters tuned via grid search

4.2 Experimental results

This section describes the final results of the above-proposed method. We conduct experiments on several public datasets to prove the method's effectiveness and compare them comprehensively with existing methods. The experiments in this paper will be conducted in the following three steps:

- Categorize against the original dataset
- Classify the dataset for balanced processing
- Perform metrics evaluation

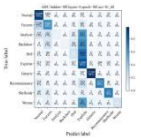

Fig. 2. GRU model classification performance on UNSW-NB15.

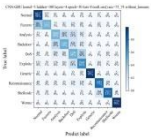

Fig. 3. CNN-GRU model classification performance on UNSW-NB15.

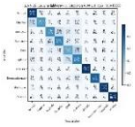

Fig. 4. KNN-RF-LR model classification performance on UNSW-NB15.

Comparative experiments with different models. Regarding model selection, we conducted training and classification experiments for several different models. First, we conducted experiments using CNN Figure 38 and GRU Figure 2 models

alone.Subsequently, we obtained the same results for the combination of the two models Figure 3. The results show that the CNN and GRU models cannot obtain good results. Results and the CNN-GRU model that combines the two are still not good enough to classify individual categories, although the overall effect has been improved.

Subsequently, we experimented with the integrated model based on the voting method. We tried a combination of machine learning methods as shown in Table 3. By observing the experimental results of multiple combinations, it is seen that the best results were obtained using the combination of KNN, RF and LR.

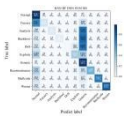

Fig.5. KNN-RF-DNN-SVM-NB model classification performance on UNSW-NB15.

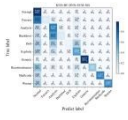

Fig.6. KNN-RF-DNN-SVM-GD model classification performance on UNSW-NB15.

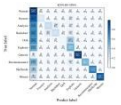

Fig. 7. KNN-RF-DNN model classification performance on UNSW-NB15.

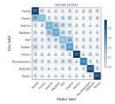

Fig. 8. KNN-RF-LR-SVM model classification performance on UNSW-NB15.

After selecting KNN, RF, and LR through the combination of different machine learning methods mentioned above, we tuned their parameters to improve model performance. As shown in Figure. 9-14, the optimal detection was achieved with KNN (n_neighbors=7) and RF (n_estimators=100). For LR, we set max_iter=10000 to ensure convergence.

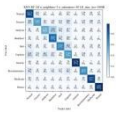

Fig. 9. Classification results for neighbors=3 estimators=50.

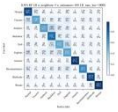

Fig. 10. Classification results for neighbors=3 estimators=100.

Fig. 11. Classification results for neighbors=5 estimators=50.

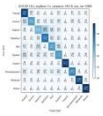

Fig. 12. Classification results for neighbors=5 estimators=100.

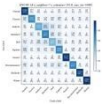

Fig. 13. Classification results for neighbors=7 estimators=50.

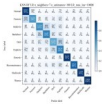

Fig. 14. Classification results for neighbors=7 estimators=100.

Comparative experiments with different oversampling methods. In the field of data imbalance by various oversampling methods, this subsection focuses on experiments for BorderlineSMOTE, ADASYN, and SVMSMOTE to determine which method will be used for oversampling in this paper.

As shown in Figure 15-Figure 18, both ADASYN and SVMSMOTE do not perform as well as BorderlineSMOTE 19 in the setting of this paper, and therefore in the up-sampling stage, the choice of the BorderlineSMOTE.

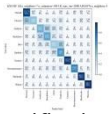

Fig. 15. Classification results for ADASYN(neighbors=5).

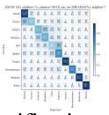

Fig. 16. Classification results for ADASYN(neighbors=7).

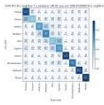

Fig. 17. Classification results for SVMSMOTE(neighbors=5).

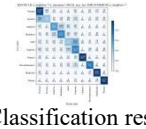

Fig. 18. Classification results for SVMSMOTE(neighbors=7).

Comparative experiments with different similarity calculation method. The following experiments were conducted to demonstrate the effectiveness of introducing the similarity calculation method and justify why the Pearson correlation coefficient was chosen. First, we trained the model without processing the data generated by BorderlineSMOTE, as shown in Figure. 19 and Figure. 20 show that the model prediction accuracy is improved after the introduction of the Pearson correlation coefficient. Secondly, we performed experiments using different correlation calculations for data constraints. Thus, as shown by Figure 20-Figure 23, the model had the best performance with the Pearson correlation coefficient.

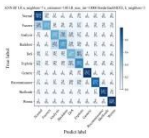

Fig. 19. Classification results while BorderlineSMOTE-only.

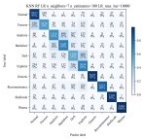

Fig. 20. Classification results for orderlineSMOTE and Pearson correlation coefficient.

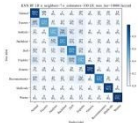

Fig. 21. Classification results for BorderlineSMOTE and Jaccard similarity coefficient.

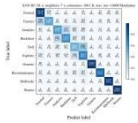

Fig. 22. Classification results for BorderlineSMOTE and Manhattan distance.

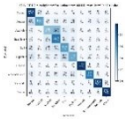

Fig. 23. Classification results for BorderlineSMOTE and Cosine similarity.

Comparative experimentas for different thresholds of Pearson's correlation coefficient. In order to determine the prediction results of the final model for Pearson's correlation coefficient with different thresholds, several sets of experiments were conducted, and the results were obtained as shown in the table below:

Table 4. Comparative experiments for different thresholds of Pearson's correlation coefficient.

Class	Threshold=0.25	Threshold=0.3	Threshold=0.35	Threshold=0.4	Threshold=0.45	Threshold=0.5
Normal	95	95	95	94	94	94
Fuzzers	58	59	58	56	57	56
Analysis	60	63	63	60	58	56
Backdoor	68	69	69	65	67	65
Dos	54	53	54	54	53	55
Exploits	69	69	66	66	67	68
Generic	97	97	97	97	98	98
Reconnaissance	79	80	77	80	79	83
Shellcode	97	97	97	97	97	97
Worms	99	100	98	99	98	98
MCA	77.6	**78.2**	77.4	76.8	76.8	77

As can be seen from the table, the model ultimately obtained the best results when the threshold value was 0.3, so 0.3 was ultimately chosen as the threshold value.

Comparison experiment. Figure 24-26 demonstrates the classification results of the UNSW-NB15, NSL-KDD, and CIC-IDS2017 datasets under the method proposed in this paper.

Fig. 24. Classification results for the UNSW-NB15 dataset.

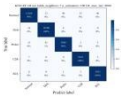

Fig. 25. Classification results for the NSL-KDD dataset.

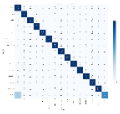

Fig. 26. Classification results for the CIC-IDS-2017 dataset.

Figs. 27-30 show the comparison for UNSW-NB15 dataset between our method and the existing methods Stacked ensemble[1], HDLBID[29] and SKM-XGB[30], and it can be seen through the figure that our method in this paper has a significant improvement in the recall and f1 scores compared to the existing methods, especially in the three categories of Analysis, Backdoor and Dos attack types have 42%, 39% and 25% improvement in detection, respectively.

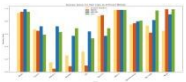

Fig. 27. Accuracy comparison across methods.

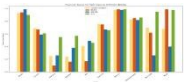

Fig. 28. Precision comparison across methods.

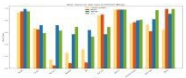

Fig. 29. Recall comparison across methods.

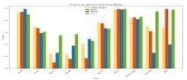

Fig. 30. F1 comparison across methods.

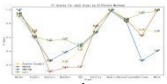

Fig. 31. F1 comparison across methods.

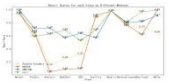

Fig. 32. Recall comparison across methods.

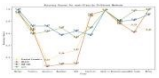

Fig. 33. Accuracy comparison across methods.

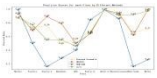

Fig. 34. Precision comparison across methods.

Table 5. Results of Comparison Experiments on the UNSW-NB15 Dataset.

Methods	MCA	MCA(F1)	MCA(recall)	MCA(precision)
Stacked ensemble[1]	63.7	64.7	63.2	74.0
HDLBID[29]	61.6	62.4	61.1	**79.8**
SKM-XGB[30]	77.2	58.4	77.2	54.4
Ours	**78.2**	**75.3**	**78.3**	76.4

In Table 5, we can see that the method in the paper obtains the highest rating under the scoring criteria of Mean Class Accuracy (MCA), the evaluation metric proposed in this paper. This shows that the method in this paper has improved in dealing with class imbalance, which is also supported by the combination of comparative experiments such as F1 shown above.

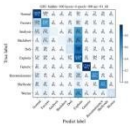

Fig. 35. UNSW-NB15 raw data classification results.

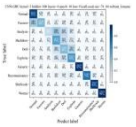

Fig. 36. Classification results after sampling on UNSW-NB15.

Ablation experiments. Ablation experiments were conducted using the method proposed in this paper. As mentioned in the previous section, the classification results are not very satisfactory when the data generation method is not taken due to the class imbalance problem prevailing in the dataset in the intrusion detection domain. Below 35 is a graph of the classification results performed for the original dataset:

As shown in Figure 35, the classification model performs poorly on the original dataset due to class imbalance. Similar to Normal, Generic, and other more numerous classes have higher accuracy rates reaching above 90, but some few classes have even single-digit accuracy rates. Therefore, in the follow-up study, we introduced data

generation techniques and feature extraction for further research. Some of the classification results are plotted below:

Table 6. Results of Comparison Experiments on the CIC-IDS-2017 Dataset.

Class	Open-set ID[31]	MLP-Based ID[32]	Ours
BENIGN	99.4	99.9	**100**
Dos Hulk	99.5	99.7	**99.8**
PortScan	99.6	99.9	**100**
DDoS	**99.8**	**99.8**	98.8
DoS GoldenEye	99.8	99.9	**100**
FTP-Patator	99.4	99.9	**100**
SSH-Patator	99.2	99.9	**100**
DoS slowloris	99.8	99.9	**100**
DoS Slowhttptest	99.8	99.9	**100**
Bot	68.0	99.6	**99.9**
Web Attack Brute Force	-	**99.2**	98.7
Web Attack XSS	99.8	99.6	**99.9**
Web Attack Sql Injection	-	99.9	**100**
Infiltration	96.3	99.9	**100**
Heartbleed	**100**	**100**	66.7

Table 7. Results of Comparison Experiments on the NSL-KDD Dataset.

Class	HDLBID[29]	MLP-Based ID[32]	SKM-XGB[30]	Ours
Normal	**99.5**	99.4	98.9	99.1
Dos	**99.8**	99.5	**99.8**	**99.8**
Probe	99.2	**99.5**	99.3	98.9
R2L	91.6	99.2	96.6	**99.8**
U2R	99.8	**99.9**	86.0	99.7

After introducing the data generation method and the feature extraction module, the classification effect has been significantly improved, but the classification effect of individual categories is still poor. We speculate whether low-quality samples generated in the data generation stage may contaminate the generated data, resulting in these categories with not very distinctive features being affected and thus poorly classified. Therefore, we introduced a sample constraint scheme to constrain the generated samples for further experiments. For the above research method, we obtain the following classification result graph, which outperforms the existing classification methods under the new evaluation metrics:

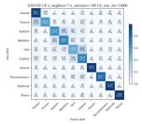

Fig. 37. Classification results for the UNSW-NB15 dataset.

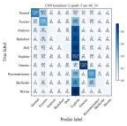

Fig. 38. CNN model classification performance on UNSW-NB15.

Time Complexity Analysis. To evaluate the time complexity of the method in this paper, we conducted several experiments with the set parameters. We averaged them to obtain the time consumption table shown below:

Table 8. Time consumption of experimental steps.

Data Generation	Similarity Calculation	Model Training	Model Prediction
3.22	2986.48	1075.43	24.02

Note: All time values are given in seconds (s).

5 Conclusion

The class imbalance problem is one of the non-negligible problems in intrusion detection systems, and how to solve the problem efficiently is an important topic in intrusion detection. In this paper, we improve the BorderlineSMOTE approach by constraining the generated samples to obtain higher-quality training data. In the subsequent stages, we conduct experiments for various models and select a more effective model architecture while confirming the approach's effectiveness. Finally, we propose the Mean Category Accuracy (MCA) evaluation criterion, under which our approach outperforms the existing intrusion detection schemes. At the same time, there is a significant improvement in the F1 scores for the two attack types, Analysis and Backdoor (42% and 39%, respectively). As a result, the method proposed in this paper significantly improves the classification results of a few classes. It provides new alternatives for relevant applications of intrusion detection systems. In the follow-up work, we will work on new data generation methods, plan to improve the data generation method based on Generative Adversarial Networks (GAN), and propose methods to generate more effective data.

Acknowledgements. This work is supported by Yunnan Key Laboratory of Smart City in Cyberspace Security (No.202105AG070010).

References

1. Ali Mohammed Alsaffar, Mostafa Nouri-Baygi, and Hamed M Zolbanin. Shielding networks: enhancing intrusion detection with hybrid feature selection and stack ensemble learning.Journal of Big Data, 11(1):133, 2024.
2. Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. Journal of artificial intelligence research, 16:321– 357, 2002.

3. Xiangyu Ma and Wei Shi. Aesmote: Adversarial reinforcement learning with smote for anomaly detection. IEEE Transactions on Network Science and Engineering, 8(2):943–956, 2020.
4. Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. Communications of the ACM, 63(11):139–144, 2020.
5. Cheolhee Park, Jonghoon Lee, Youngsoo Kim, Jong-Geun Park, Hyunjin Kim, and Dowon Hong. An enhanced ai-based network intrusion detection system using generative adversarial networks. IEEE Internet of Things Journal, 10(3):2330–2345, 2022.
6. Jiyuan Cui, Liansong Zong, Jianhua Xie, and Mingwei Tang. A novel multi-module integrated intrusion detection system for high-dimensional imbalanced data. Applied Intelligence, 53(1):272–288, 2023.
7. Dhiraj Ganji and Chandranil Chakraborttii. Towards data generation to alleviate privacy concerns for cybersecurity applications. In 2023 IEEE 47th Annual Computers, Software, and Applications Conference (COMPSAC), pages 1447–1452. IEEE, 2023.
8. Jiaxing He, Xiaodan Wang, Yafei Song, Qian Xiang, and Chen Chen. Network intrusion detection based on conditional wasserstein variational autoencoder with generative adversarial network and one-dimensional convolutional neural networks. Applied Intelligence, 53(10):12416–12436, 2023.
9. Mohammad Arafah, Iain Phillips, Asma Adnane, Wael Hadi, Mohammad Alauthman, and Abedal-Kareem Al-Banna. Anomaly-based network intrusion detection using denoising autoencoder and wasserstein gan synthetic attacks. Applied Soft Computing, 168:112455, 2025.
10. Auangkun Rangsikunpum, Sam Amiri, and Luciano Ost. Bids: An efficient intrusion detection system for in-vehicle networks using a two-stage binarised neural network on low-cost fpga. Journal of Systems Architecture, 156:103285, 2024.
11. Gyurin Byun, Huigyu Yang, Syed M Raza, Moonseong Kim, Min Young Chung, and Hyunseung Choo. Generative spatiotemporal image exploitation for datacenter traffic prediction. Computer Networks, 254:110755, 2024.
12. Caihong Wang, Du Xu, Zonghang Li, and Dusit Niyato. Effective intrusion detection in highly imbalanced iot networks with lightweight s2cgan-ids. IEEE Internet of Things Journal, 2023.
13. Wuxia Bai, Kailong Wang, Kai Chen, Shenghui Li, Bingqian Li, and Ning Zhang. Sc-wgan: Gan-based oversampling method for network intrusion detection. In International Conference on Engineering of Complex Computer Systems, pages 23–42. Springer, 2024.
14. Ankit Thakkar and Ritika Lohiya. Attack classification of imbalanced intrusion data for iot network using ensemble-learning-based deep neural network. IEEE Internet of Things Journal, 10(13):11888–11895, 2023.
15. Dina Ayesha S and Siddique AB. Fs3: Few-shot and self-supervised framework for efficient intrusion detection in internet of things networks. In Proceedings of the 39th Annual Computer Security Applications Conference, pages 138–149, 2023.
16. Jiawei Zhang, Rui Chen, Yanchun Zhang, Weihong Han, Zhaoquan Gu, Shuqiang Yang, and Yongquan Fu. Mf2pose: Multi-task feature fusion pseudo-siamese network for intrusion detection using category-distance promotion loss. Knowledge-Based Systems, 283:111110, 2024.
17. Yakubu Imrana, Yanping Xiang, Liaqat Ali, Adeeb Noor, Kwabena Sarpong, and Muhammed Amin Abdullah. Cnn-gru-ff: a double-layer feature fusion-based network intrusion detection system using convolutional neural network and gated recurrent units. Complex & Intelligent Systems, pages 1–18, 2024.

18. Srinivas Mukkamala, Guadalupe Janoski, and Andrew Sung. Intrusion detection using neural networks and support vector machines. In Proceedings of the 2002 International Joint Conference on Neural Networks. IJCNN'02 (Cat. No. 02CH37290), volume 2, pages 1702–1707. IEEE, 2002.
19. Nahla Ben Amor, Salem Benferhat, Zied Elouedi, and Khaled Mellouli. Decision trees and qualitative possibilistic inference: Application to the intrusion detection problem. In Symbolic and Quantitative Approaches to Reasoning with Uncertainty: 7th European Conference, ECSQARU 2003 Aalborg, Denmark, July 2-5, 2003 Proceedings 7, pages 419–431. Springer, 2003.
20. Abdulla Amin Aburomman and Mamun Bin Ibne Reaz. A novel svm-knn-pso ensemble method for intrusion detection system. Applied Soft Computing, 38:360–372, 2016.
21. Constantinos Kolias, Georgios Kambourakis, Angelos Stavrou, and Stefanos Gritzalis. Intrusion detection in 802.11 networks: Empirical evaluation of threats and a public dataset. IEEE Communications Surveys & Tutorials, 18(1):184–208, 2015.
22. Wathiq Laftah Al-Yaseen, Zulaiha Ali Othman, and Mohd Zakree Ahmad Nazri. Multi-level hybrid support vector machine and extreme learning machine based on modified k-means for intrusion detection system. Expert Systems with Applications, 67:296–303, 2017.
23. Hasan Alkahtani and Theyazn HH Aldhyani. Intrusion detection system to advance internet of things infrastructure-based deep learning algorithms. Complexity, 2021(1):5579851, 2021.
24. Abdulrahman Saad Alqahtani. Retracted article: Fso-lstm ids: hybrid optimized and ensembled deep-learning network-based intrusion detection system for smart networks. The Journal of Supercomputing, 78(7):9438–9455, 2022.
25. Mohamed Abdel-Basset, Hossam Hawash, Ripon K Chakrabortty, and Michael J Ryan. Semi-supervised spatiotemporal deep learning for intrusions detection in iot networks. IEEE Internet of Things Journal, 8(15):12251–12265, 2021.
26. Yihan Xiao, Cheng Xing, Taining Zhang, and Zhongkai Zhao. An intrusion detection model based on feature reduction and convolutional neural networks. IEEE Access, 7:42210–42219, 2019.
27. Safi Ullah, Jawad Ahmad, Muazzam A Khan, Eman H Alkhammash, Myriam Hadjouni, Yazeed Yasin Ghadi, Faisal Saeed, and Nikolaos Pitropakis. A new intrusion detection system for the internet of things via deep convolutional neural network and feature engineering. Sensors, 22(10):3607, 2022.
28. Hui Han, Wen-Yuan Wang, and Bing-Huan Mao. Borderline-smote: a new over-sampling method in imbalanced data sets learning. In International conference on intelligent computing, pages 878–887. Springer, 2005.
29. KG Raghavendra Narayan, Rakesh Ganesula, Tamminaina Sai Somasekhar, Srijanee Mookherji, Vanga Odelu, Rajendra Prasath, and Alavalapati Goutham Reddy. Attenuating majority attack class bias using hybrid deep learning based ids framework. Journal of Network and Computer Applications, 230:103954, 2024.
30. Mohammad Kazim Hooshmand, Manjaiah Doddaghatta Huchaiah, Ahmad Reda Alzighaibi, Hasan Hashim, El-Sayed Atlam, and Ibrahim Gad. Robust network anomaly detection using ensemble learning approach and explainable artificial intelligence (xai). Alexandria Engineering Journal, 94:120–130, 2024.
31. Wei Yu, Zhixiang Chen, Hui Wang, Zeyu Miao, and Dake Zhong. Industrial network intrusion detection in open-set scenarios. International Journal of Information Security, 24(1):1–16, 2025.
32. Sarra Cherfi, Ali Lemouari, and Ammar Boulaiche. Mlp-based intrusion detection for securing iot networks. Journal of Network and Systems Management, 33(1):20, 2025.