Hindawi
Mobile Information Systems
Volume 2023, Article ID 1594622, 11 pages
https://doi.org/10.1155/2023/1594622

Research Article

Network Intrusion Anomaly Detection Model Based on Multiclassifier Fusion Technology

Feilu Hang,[1] **Wei Guo**,[1] **Hexiong Chen**,[1] **Linjiang Xie**,[1] **Xiaoyu Bai**,[2] **and Yao Liu**[2]

[1]*Information Center, Yunnan Power Grid Co. Ltd., Kunming, Yunnan 650034, China*
[2]*Network and Data Security Key Laboratory of Sichuan Province, University of Electronic Science and Technology of China, Chengdu, Sichuan 610054, China*

Correspondence should be addressed to Xiaoyu Bai; 202121090126@std.uestc.edu.cn

Received 8 August 2022; Revised 14 September 2022; Accepted 28 September 2022; Published 8 April 2023

Academic Editor: Sai Zou

With the increasing development of the industrial Internet, network security has attracted more and more attention. Among the numerous network security technologies, anomaly detection technology based on network traffic has become an important research field. At present, a large number of methods for network anomaly detection have been proposed. Most of the better performance detection methods are based on supervised machine learning algorithms, which require a large number of labelled data for model training. However, in a real network, it is impossible to manually filter and label large-scale traffic data. Network administrators can only use unsupervised machine learning algorithms for actual detection, and the detection effects are much worse than supervised learning algorithms. To improve the accuracy of the unsupervised detection methods, this study proposes a network anomaly detection model based on multiple classifier fusion technology, which applies different fusion techniques (such as Majority Vote, Weighted Majority Vote, and Naive Bayes) to fuse the detection results of the five best performing unsupervised anomaly detection algorithms. Comparative experiments are carried out on three public datasets. Experimental results show that, in terms of RECALL and AUC score, the fusion model proposed in this study achieves better performance than the five separate anomaly detection baseline algorithms, and it has better robustness and stability, which can be effectively applied to a wide range of network anomaly detection scenarios.

1. Introduction

With the advancement of network technology, an increasing number of users are connected to the Internet. As the Internet grows in size, the threats from attackers and criminal enterprises have also increased accordingly. The increasing number of these threats makes firewall and intrusion detection systems (IDS) become one of the foundational technologies of network security. However, intrusion detection systems on the market primarily exploit the signatures of known attacks to detect anomalies. Such systems require frequent updates of the rule database and signatures, and unknown attacks cannot be detected. Anomaly detection systems can effectively discover the attack behavior by modelling normal behaviors and detecting anomalous behavior which are different from normal behavior. Although anomaly detection systems are conceptually attractive, there are many technological issues to overcome before they can be widely adopted, such as high false positive rates and the inability to scale to gigabit speeds. At present, anomaly detection methods are mainly divided into two categories: anomaly detection based on statistics and anomaly detection based on machine learning.

Haystack [1] is one of the earliest examples of a statistical anomaly intrusion detection system. Haystack defines a series of values that are considered normal for each feature. If, during a session, a feature is outside the normal range, the topic's anomaly score increases, and it is designed to detect six types of intrusions. The Statistics Package Anomaly Detection Engine (SPADE) [2] is a statistical anomaly detection system. SPADE was one of the first systems to propose the use of the anomaly scoring concept to detect

port scans, rather than using the traditional method of looking at p attempts within q seconds. In [2], the authors use a simple frequency-based method to calculate the "anomaly score" of a packet. The less views a given packet has, the higher its anomaly score is.

Machine learning is designed to answer many of the same questions as statistics. However, unlike statistical methods, which tend to focus on understanding the processes that generate data, machine learning techniques focus on building a system that improves performance based on previous results. The Bayes Network is a graph model that encodes the probabilistic relationships between target variables. When combined with statistical techniques, Bayes Network has advantages in data analysis [3]. Some researchers have taken inspiration from Bayes statistics to create anomaly detection models. Valdes [4] has developed an anomaly detection system that uses a naive Bayes Network to perform intrusion detection on traffic bursts. Bayesian techniques are also often used to classify and suppress false positive regions. Kruegel et al. [5] proposed a multisensor fusion method in which the outputs of different IDS sensors are aggregated together to generate a single alarm. While intrusion detection using the Bayes Network is effective in some applications, its limitations should be considered in practice. Unfortunately, typical network structures are complex, and choosing an accurate model of behavior is a daunting task.

To solve the problem of high-dimensionality datasets, the researchers developed a dimensionality reduction technique called principal component analysis (PCA) [6]. Shyu et al. [7] proposed an anomaly detection scheme in which PCA is used as an anomaly detection scheme and applied to reduce the dimension of audit data and obtain a classifier. Extreme Learning Machine (ELM) and NSL Knowledge Discovery Data Mining [8] have been identified as criteria for evaluating network intrusion detection mechanisms. The researchers also implemented a random forest classifier on an IDS dataset sample [9, 10]. In addition, the available datasets need to be continuously updated based on the dynamic characteristics of the malware attack.

Some researchers have proposed implementing deep learning models and deep neural networks (DNN) to develop flexible and dynamic IDS systems that can successfully detect and classify capricious cyberattacks [10–13]. In harmony with the use of DNN, convolutional neural network has been identified as an advanced and superior technique for extracting features from intrusion datasets for classification [14]. Since this method [15] provides visual detection of network intrusions, it can be justified as a real-time solution for deploying intrusion detection systems [10]. The authors in [16] explored the different deep learning methods used in IDS and proposed comparisons and analyses. The authors of [17] detected intrusion based on Spark-Chi-SVM technology.

The authors in [18–21] propose several hybrid classification models that classify datasets using naturally inspired algorithms such as cuckoo search, BAT, fireflies, and genetic algorithms [10]. The differential performance of different classifiers is related to several factors, such as the statistical distributional properties of the categorical data, prior knowledge, the size of the training data samples, and the structure of the classifier itself. In anomaly detection, different intrusion identification results can be obtained using different classifiers, and these results are often highly complementary. Therefore, the fusion of the decisions of multiple classifiers can effectively improve the detection effect of anomalies. Moreover, the fusion of multiple classifiers can also improve the robustness of the classification system. In the selection of fusion methods, Dainotti et al. [22] summarize some of the common fusion methods, including majority voting, weighted majority voting, Naive Bayes, behavioral knowledge space (BKS), Wernecke's (WER) method, and the Oracle (ORA) method.

In view of this, this study proposes a network intrusion anomaly detection model MF based on multiclassifier fusion. Three fusion techniques (majority voting, weighted majority voting, and Naive Bayes) were used to fuse the decisions of different anomaly detection methods based on unsupervised learning that have performed well in recent years, such as Lightweight Online Detector of Anomalies (LODA), AutoEncoder, PCA, Histogram-based Outlier Score (HBOS), and iForest (Isolation Forests). The major contributions and findings of this study are listed below:

(1) In real-world network environments, the large scale of labelled data is rare, and the supervised learning methods are not suitable. However, using only unsupervised learning methods leads to poor performance and low accuracy of the model due to the lack of guidance from labelled data. The MF model proposed in this study provides a framework to ensemble various anomaly detection classifiers to form heterogeneous or homogeneous modelling backgrounds for final decision-making and significantly improves the overall detection performance; in other words, it improves the RECALL and AUC metrics.

(2) In real-world network environments, intrusions change rapidly and new attacks are endlessly emerging. Different detection classifiers tend to be biased in their detection performance, and perhaps one of the anomaly detection classifiers would work well for a particular intrusion detection job. The MF model proposed in this study can remedy the shortcomings of a single anomaly detection method and realize the complementarity of different detection methods by using multiclassifier fusion technology.

The remainder of the study is organized as follows. Section 2 presents the anomaly detection methods based on unsupervised learning used in this study. Section 3 describes the multiclassifier fusion technique, as well as the MF model architecture and procedure. Section 4 presents the three anomaly detection datasets. Section 5 presents the experimental configuration, discussion, and findings. Finally, Section 5 presents our conclusions and future work.

2. Anomaly Detection Methods Based on Unsupervised Learning

This section introduces five anomaly detection algorithms based on unsupervised learning that have the best detection effect and fast detection speed and are suitable for large-scale networks. Unsupervised learning methods have no training or learning phase and do not require data labelling, which is more suitable for the detection requirements of a real network. And these five detection methods all assign an outlier score to each of the data points. This is an advantage compared to binary output methods because, additionally, the outlier score allows for estimating the reliability or certainty of the prediction.

2.1. LODA. A lightweight online detector of anomalies (LODA) [23] can be used to quickly process a large stream of data generated by continuously changing behaviors. It consists of a collection of k one-dimensional histogram $\{h_i\}^k$; each histogram approximates the probability density of input data projected into a single projection vector $\{w_i \in R^d\}_{i=1}^k$. The output of LODA is the average of the logarithms of probability density on multiple histograms, thereby improving the performance of a single histogram detector.

The input to LODA is sample x, and the output $f(x)$ represents the average of the logarithms of the estimated probabilities projected by the sample into different projection vectors. Using $\hat{p}_i$ represents the probability estimated by the ith histogram, W_i represents the corresponding projection vector. The output $f(x)$ of LODA can be written as follows:

$$f(x) = -\frac{1}{k}\sum_{i=1}^{k} \log \hat{p}_i\left(x^T w_i\right). \tag{1}$$

If different histograms are considered to be independent of each other, (2) can be used to integrate the probability densities of multiple histograms:

$$f(x) = -\log p\left(x^T w_1, x^T w_2, x^T w_i\right), \tag{2}$$

where $(x^T W_1, x^T W_2, \ldots, xW_k)$ represents the joint probability of the projection. (2) shows that the output of LODA is proportional to the negative log likelihood of the sample, which means that the less likely the sample is to appear, the higher its outliers are.

2.2. Autoencoder. An autoencoder [24] is an unsupervised learning model that essentially uses a neural network to generate a low-dimensional representation of a high-dimensional input. Autoencoder is similar to principal component analysis (PCA), but autoencoder utilizes a nonlinear activation function, thus overcoming the limitation that PCA can only do feature linear transformation.

An autoencoder is composed of two parts, an encoder and a decoder. The encoding procedure performs a dimension reduction on the training data and projects training data in the latent space, where the features of the training data are preserved. A decoder can be constructed to recover the data using the features in the latent space. The difference between the original input vector and the reconstruction vector is called the reconstruction error. If the features of the sample are all numerical variables, the mean squared error (MSE) or the mean absolute error (MAE) can be used as the reconstruction error. For example, the input sample is $X = (X_1, X_2, ..., X_m)$. The result of autoencoder reconstruction is $X^R = (X_1^R, X_2^R, ..., X_m^R)$. The reconstruction error is MSE $1/m \sum_{i=1}^{m} (X_i - X_i^R)^2$.

The reconstruction error was used as the anomaly score. Data points with high reconstruction are considered to be anomalies. Only data with normal instances are used to train the autoencoder. After training, the autoencoder will reconstruct normal data very well, while failing to do so with anomaly data, which the autoencoder has not encountered.

2.3. PCA. Principal component analysis (PCA) [25] is the most common method of data dimensionality reduction. Generally, in anomaly detection scenarios, noise, outlier, and anomaly are different representations of the same thing. Since PCA can recognize noise, it can naturally detect anomalies. PCA maps the data to low-dimensional feature space and then checks the deviation of each data point from the other data on different dimensions of the feature space.

For a feature vector e_j, the deviation degree d_{ij} of the data sample x_i in direction j can be calculated by using

$$d_{ij} = \frac{\left(x_i^T \cdot e_j\right)^2}{\lambda_j}, \tag{3}$$

where λ_j mainly plays a role in normalization, which can make the deviation degrees in different directions comparable. After that, the anomaly score of sample x_i is calculated, as shown in Equation (4). If the score is greater than the threshold, the sample x_i is judged as an anomaly:

$$\text{Score}\left(x_j\right) = \sum_{j=1}^{n} d_{ij}. \tag{4}$$

2.4. HBOS. HBOS (Histogram-based Outlier Score) [26] is a combination of univariate methods that cannot model dependencies between features, but is faster and friendly to large datasets. HBOS calculates an outlier score by creating a univariate histogram for every single feature of the dataset. It assumes that the features are independent. The drawback of assuming feature independence becomes less severe when the dataset has a high number of dimensions due to a larger sparsity.

Two different methods can be used to construct histograms: the static bin width and the dynamic bin width [27].

(1) The static bin width: data are grouped using bins of the same width. A rectangle is drawn in each interval, whose height is proportional to the number of points that fall into the interval. Equal binning assumes that

each bin is equally likely, but this assumption is usually not met.

(2) The dynamic bin width: the width of bins depends on the values of data and may not necessarily be equal to the specified number of bins. In the case of long-tail distributions with repetitive integers, some bins may contain more values than specified.

In both the static and dynamic cases, it is necessary to specify the number of bins k. It is recommended that the k value should be equal to the square root of all data points. The height of every single bin in the histogram represents the density estimation. To ensure an equal weight for each feature, the histograms are normalized in such a way that the maximum height of the bin would be equal to one. Then, calculated values are inverted so that anomalies have a high score and normal instances have a low score. The anomaly score of each sample x_i is calculated according to Equation (5). The higher the HBOS score, the greater the anomaly degree of the sample:

$$HBOS(x_i) = \sum_{j=0}^{a} \log\left(\frac{1}{\text{hist}_j(x_i)}\right), \tag{5}$$

where a is the number of features/histograms, x_j is the vector of features, and $\text{hist}_j(x_j)$ represents the density estimation of feature instance x_i in the jth histogram.

2.5. iForest. Isolation Forest (iForest) is a fast outlier detection method based on an ensemble, with linear time complexity and high accuracy, which meets the needs of big data processing. iForest was first proposed in 2008, and then in 2012, an improved version was proposed [28], which is suitable for anomaly detection of continuous data. iForst is different from other anomaly detection algorithms to characterize the degree of alienation between data samples based on quantitative indicators such as distance and density and detects outliers by isolating sample points.

The algorithm utilizes a binary search tree structure called an isolation tree (iTree) to isolate samples. An iForest consists of multiple isolation trees, which are created by choosing attributes and the values of attributes randomly. At each node in the isolation trees, the instance set is divided into two parts based on the chosen attributes and their values. Here, the attributes are selected randomly, and the split value for this selected attribute is selected randomly as well between the minimum value and maximum value of this selected attribute. Commonly, anomalous instances are those objects whose attribute values are very different from the normal instances and are easier to be divided than normal instances. In the process of isolation, they are also closer to the root and more easily divided than normal instances.

3. Network Intrusion Anomaly Detection Model Based on Multiclassifier Fusion

The network intrusion anomaly detection model based on multiclassifier fusion is described in the following section.

3.1. Anomaly Detection Model Based on Multiclassifier Fusion. The anomaly detection method is based on network traffic to extract the content features, essential features, and traffic features from the original network traffic data and detect anomalies from the feature space composed of these three types of features. In intrusion detection, the features associated with different network behavior have different meanings, and it is difficult for a single classifier to effectively deal with the combination of these features with different meanings. A multiple classifier system is needed, whose outputs are combined in some way to obtain a final classification decision under different situations. In an ensemble anomaly detection system, each base classifier will focus on different aspects of the data.

To overcome the limitations of a single classifier, this study proposed a network intrusion anomaly detection model (MF) based on the multiclassifier fusion method. The MF model intelligently combines the decision outputs of multiple classifiers according to selected fusion rules and achieves better detection performance than a single classifier. The overall architecture of the MF model is shown in Figure 1.

The algorithm selection criteria for the MF model are that there is no strong dependence between individual classifiers and that a series of individual classifiers can be generated in parallel. These algorithms are preferably heterogeneous, which means that the types of these algorithms are not the same as the five algorithms chosen in this study. In anomaly detection, different individual classifier results can be obtained using different classifiers, and these results are often highly complementary. Thus, several well-performing heterogeneous anomaly detection algorithms can potentially yield better anomaly detection models.

After data collection and feature extraction, the dataset is divided into two subsets: a training set and a testing set (unsupervised machine learning produces predictions during training). The purpose of the training set is to apply the multiclassifier fusion technique and obtain the confusion matrix of multiple anomaly detection algorithms. The role of the test set is to fuse the outputs of multiple anomaly detection methods using the confusion matrix and obtain the final detection results.

3.2. Multiclassifier Fusion Methods. A confusion matrix is a standard format for expressing accuracy evaluation, which is represented by a matrix with n rows and n columns. Each column of the confusion matrix represents the predicted category, and the total number of data in each column represents the number of data predicted to be in that category. Each row represents the true attribution category of the data, and the total number of data in each row represents the number of data instances in that category.

To ensure the optimal performance, the multiclassifier fusion method should be able to select the subset of classifiers that is optimal in the sense that it produces the highest possible performance for a particular combiner. In this study, three multiclassifier fusion methods are carried out in the MF model, such as majority voting, weighted majority

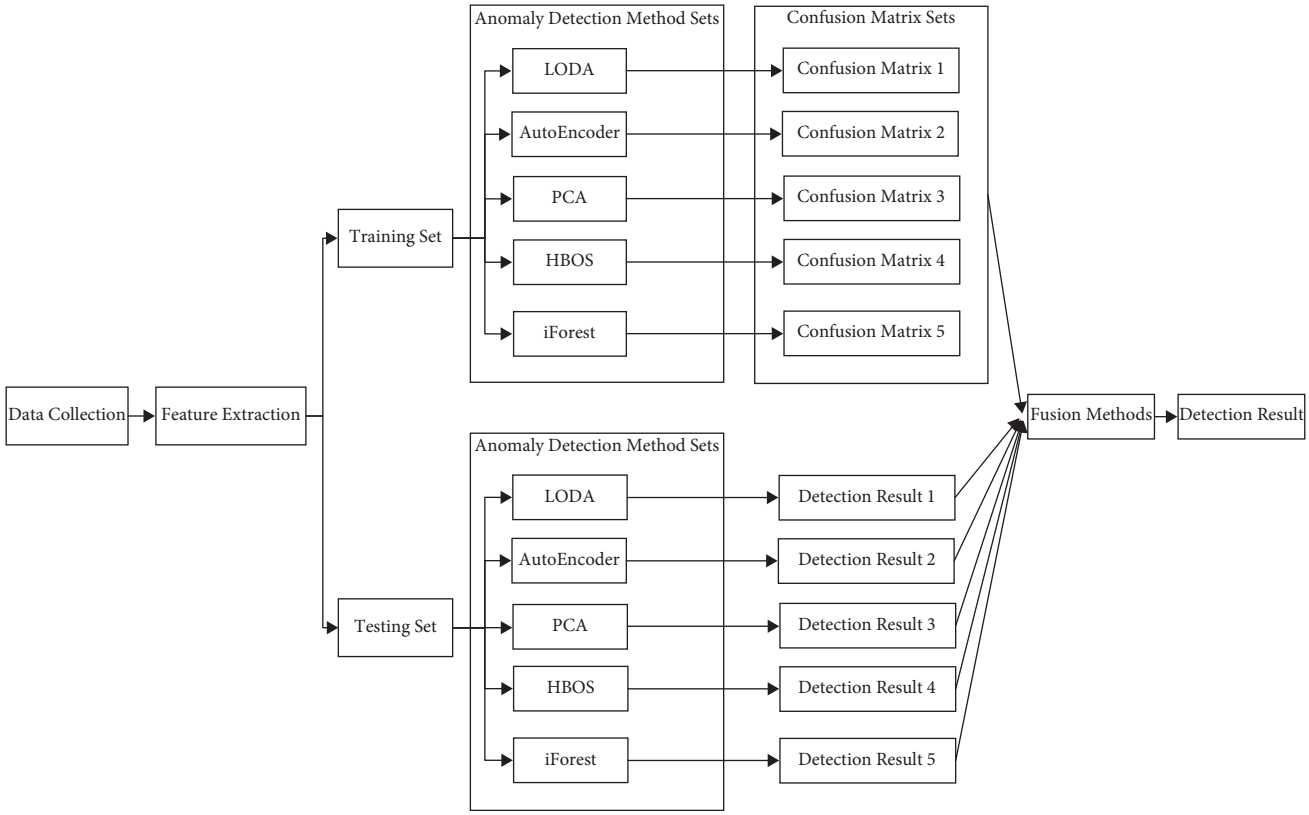

Figure 1: MF model overall architecture.

voting, and Naive Bayes. To better evaluate the performance of these three methods, we assume that the kth classifier gets the training set as input, the predicted confusion matrix is E^k, and $e^k{}_{ij}$ represents the percentage of the samples in class i is predicted as class j in the kth classifier.

3.2.1. Majority Voting. The number of admissible classifiers voted for a special class is assessed. The class with more votes is selected as the ultimate decision. However, in the case of an equal number of votes in two or three classes, the number of admissible classifiers voted against each class is counted and the class with the minimum vote is selected as the final decision. The majority voting can be written formally as follows:

$$C = \underset{i}{\operatorname{argmax}} \sum_{k} V_i^k, \tag{6}$$

where the combiner classifies the sample into class C in the manner shown in Equation (6). If the kth classifier predicts that the class of the sample is i, then V_i^k is 1, otherwise, it is 0. The algorithm implementation of the MV is shown in Algorithm 1.

3.2.2. Weighted Majority Voting. If many classes receive the same number of votes, $e^k{}_{ii}$ is used for tie-breaking; that is, the voting rights of each classifier are weighted by a number representing the confidence of the classifier for its vote. The vote given by each classifier is weighted by the confidence level assessed by the confusion matrix, and the combiner classifies the sample into class C in the manner shown as

$$C = \underset{i}{\operatorname{argmax}} \sum_{k} e_{ii}^k \cdot V_i^k. \tag{7}$$

The algorithm implementation of the WMV is shown in Algorithm 2.

3.2.3. Naive Bayes. Based on the priori probability of a sample, the Naive Bayes classifier calculates its posterior probability using the Bayes formula. The posterior probability is the probability that the sample belongs to a certain class. The method selects the class with the largest posterior probability as the decision class for the sample.

When the kth classifier predicts that the class of the sample is j, combined with the prediction results of all classifiers, the probability $p(i)$ that the sample belongs to the class i is shown as

$$p(i) = \frac{M_i \cdot e_{ij}^k}{\sum_{h=1}^{m} M_h \cdot e_{hj}^k} \tag{8}$$

where M_i is the number of samples that belong to class i. Applying Bayes formula and assuming that the classifiers are independent, the maximizing posterior probability is shown as [29]

Input: the results set predicted by each base classifiers Y;
Output: final prediction result C;
(1) $a\ [2] \leftarrow \{0, 0\}$;
(2) for y in Y do
(3) $a\ [y] \leftarrow a\ [y] + 1$;
(4) end for
(5) $C \leftarrow \text{argmax}\ (a)$
(6) return C

Algorithm 1: Majority voting

Input: the results set predicted by each base classifiers $Y = \{y_1, y_2, \ldots, y_n\}$, confusion matrix set $E = \{e^1, e^2, \ldots, e^n\}$;
Output: final prediction result C;
(1) $a\ [2] \leftarrow \{0, 0\}$;
(2) for i in $\{0, 1\}$ do
(3) for j in $\{1, 2, \ldots, n\}$ do
(4) if $y_j == i$ then
(5) $v_j \leftarrow 1$;
(6) else
(7) $v_j \leftarrow 0$;
(8) end if
(9) $a\ [i] \leftarrow a\ [i] + v_j * e^j_{ii}$;
(10) end for
(11) end for
(12) $C \leftarrow \text{argmax}\ (a)$;
(13) return C;

Algorithm 2: Weighted majority voting

Input: the results set predicted by each base classifiers $Y = \{y_1, y_2, \ldots, y_n\}$, confusion matrix set $E = \{e^1, e^2, \ldots, e^n\}$, the number of samples of each types $M = \{m_0, m_1\}$;
Output: final prediction result C;
(1) $a\ [2] \leftarrow \mathbf{\{0, 0\}}$;
(2) for i in $\{0, 1\}$ do
(3) $c_i \leftarrow 1$;
(4) for j in $\{1, 2, \ldots, n\}$ do
(5) $k \leftarrow y_j$;
(6) $c_i \leftarrow c_i * e^j_{ik}$;
(7) end for
(8) $a\ [i] \leftarrow m_i * c_i$;
(9) end for
(10) $C \leftarrow \text{argmax}\ (a)$;
(11) return C;

Algorithm 3: Naive Bayes

$$C = \underset{i}{\text{argmax}}\, M \prod_{k=1}^{N} e^k_{ij}. \tag{9}$$

The algorithm implementation of the NB is shown in Algorithm 3.

4. Dataset

In this study, a series of experiments are carried out on three public available datasets which are widely used in the field of network intrusion detection. The three datasets are CICIDS2017 [30], UNSW-NB 15 [31], and KDDCUP 99. A detailed description of the three datasets is shown in Table 1.

4.1. CICIDS2017. CICIDS2017 is an IDS domain dataset that has been collected for a total of 5 days up to 5 p.m. on Friday, July 7, 2017. Monday is a normal day that includes only normal traffic. Some of the most common attacks, such as DoS, DDoS, Brute Force, XSS, SQL injection, infiltration, port scanning, and botnets were performed on Tuesday,

Table 1: Public dataset description.

Dataset	Dimension	Number
CICIDS2017	83	592782
UNSW-NB 15	196	185423
KDDCUP 99	118	50000

Table 2: Experimental configuration environment.

Hardware/software	Configuration/version
Pyod	1.0.0
CPU	2.3 GHz dual core intel core i5
Memory	16 GB 2133 MHz LPDDR3
Operating system	macOS big sur

Table 3: Anomaly detection algorithm performance comparison (anomaly ratio of 0.05).

Dataset	Performance metrics	LODA	AE	PCA	HBOS	iForest	MV	WMV	NB
CICIDS2017	AUC	0.635	0.576	0.538	0.632	0.657	0.763	0.749	0.852
	RECALL	0.497	0.438	0.401	0.531	0.492	0.794	0.821	0.886
	Time (seconds)	0.511	8.369	0.732	0.902	7.436	7.082	8.198	8.073
UNSW-NB 15	AUC	0.675	0.739	0.852	0.724	0.913	0.871	0.912	0.925
	RECALL	0.724	0.791	0.893	0.852	0.937	0.897	0.943	0.961
	Time (seconds)	0.867	10.023	0.992	1.942	8.241	6.924	8.245	8.128
KDDCUP 99	AUC	0.942	0.923	0.924	0.938	0.953	0.910	0.928	0.956
	RECALL	0.964	0.957	0.940	0.964	0.983	0.924	0.965	0.995
	Time (seconds)	0.735	9.710	1.324	1.672	8.092	7.942	8.124	8.206

Wednesday, Thursday, and Friday morning and afternoon, respectively. The dataset is fully labelled and provides the CICFlowMeter software that triages traffic data and extracts more than 80 stream signatures, which are available on the Canadian Cybersecurity Institute website.

4.2. UNSW-NB 15. The UNSW-NB 15 dataset was created by the Australian Center for Cyber Security (ACCS) using the IXIA PerfectStorm tool, which can be used to generate raw network traffic mixed with normal activity and attack behavior. 100 GB of raw traffic (PCAP files) was captured using the tcpdump tool, and 49 stream signatures were extracted using the Argus and Bro-IDS tools, covering 9 attacks: Fuzzers, Analysis, Backdoors, DoS, Exploits, Generic, Reconnaissance, Shellcode, and Worms.

4.3. KDDCUP 99. The KDD CUP 99 dataset is the data used in the 1999 KDD CUP competition. In 1998, the U.S. defense advanced program (DARPA) conducted an intrusion detection assessment program at the Lincoln Laboratory at MIT. Lincoln Labs built a network environment that simulated the U.S. Air Force's local area network, collected network connection and system audit data for 9 weeks, and simulated various user types, various network traffic, and attack methods to make it resemble a real network environment. The raw data collected through tcpdump are divided into two parts: 7 weeks of training data, which contains about 5,000,000 million network connection records; the remaining 2 weeks of test data contains approximately 2,000,000 million network connection records.

5. Results and Discussion

5.1. Performance Metrics. RECALL and AUC (Area under Curve) are two important metrics used to evaluate the performance of anomaly detection algorithms. The time overhead is used to measure the time efficiency of each algorithm running.

(1) RECALL, also known as the detection rate, reflects the ability of the classifier or model to correctly predict positive samples (abnormal samples), that is, the proportion of the total number of positive samples predicted as positive to the total number of positive samples. The higher the value, the better the performance.

(2) AUC (Area under Curve): in a prediction, if it is an anomalous score value, a score is generally selected as a threshold, and the score above this threshold is abnormal, the score below this threshold is determined to be normal. At this time, according to the different values of the threshold, different FPR (False Positive Rate) and TPR (True Positive Rate) values can be obtained, and all values are plotted with FPR as the horizontal axis, TPR as the vertical axis, that is, the ROC curve, and the area covered by the curve downward is the AUC value.

Table 4: Anomaly detection algorithm performance comparison (anomaly ratio of 0.1).

Dataset	Performance metrics	LODA	AE	PCA	HBOS	iForest	WV	WMV	NB
CICIDS2017	AUC	0.629	0.643	0.683	0.513	0.753	0.789	0.819	0.842
	RECALL	0.389	0.421	0.569	0.492	0.628	0.853	0.853	0.896
	Time (seconds)	0.441	6.649	0.484	0.197	7.391	3.724	3.621	3.604
UNSW-NB 15	AUC	0.721	0.691	0.783	0.689	0.927	0.919	0.924	0.943
	RECALL	0.752	0.803	0.902	0.762	0.932	0.934	0.951	0.958
	Time (seconds)	0.529	8.236	0.692	0.384	9.242	4.045	4.578	4.248
KDDCUP 99	AUC	0.928	0.956	0.931	0.928	0.954	0.894	0.927	0.961
	RECALL	0.952	0.969	0.950	0.955	0.985	0.961	0.985	0.993
	Time (seconds)	0.502	7.923	0.669	0.328	8.927	3.928	4.029	4.293

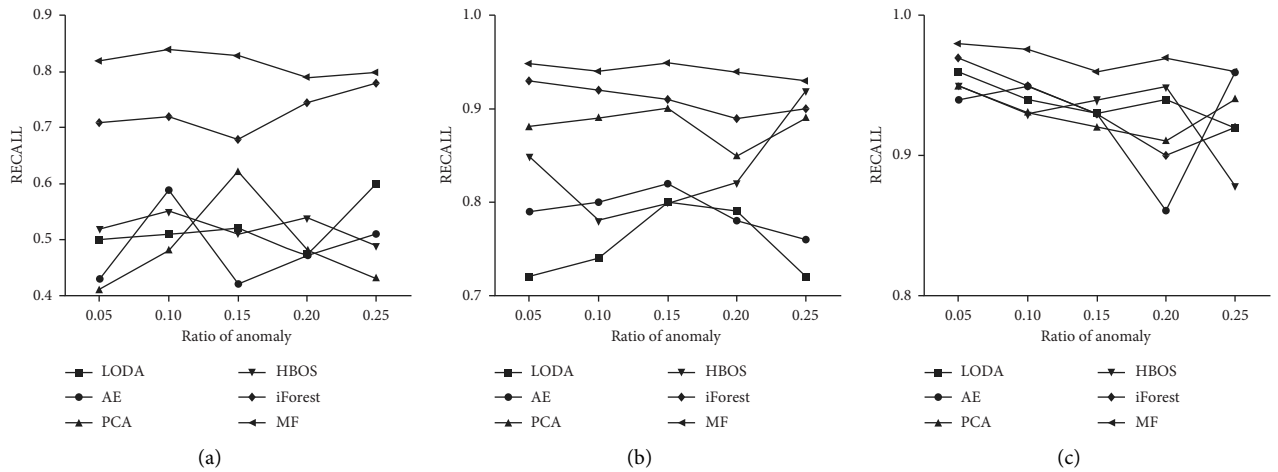

Figure 2: RECALL of different datasets varies with the anomaly rate. (a) CICIDS2017. (b) UNSW-NB 15. (c) KDDCUP 99.

(3) Time is also known as time overhead, during which an algorithm is executed.

5.2. Experimental Configuration Environment. The experimental configuration environment is given in Table 2. This study uses a well-known python third-party library for anomaly detection, PyOD, which runs on a PC machine. The hardware configuration is 2.3 GHz dual core Intel Core i5, 16 GB of 2133 MHz LPDDR3 memory, and the macOS Big Sur operating system without GPU acceleration. In addition, since the base classifier references the third-party python library, Pyod, the anomaly rate is the only parameter setting of the algorithms, which is reflected in Section 5.4.

5.3. Performance Comparison. Most of the real network traffic is normal, with a small percentage of attacks or anomalous traffic. Therefore, the vast majority of datasets widely used in the field of anomaly detection are data imbalances, the proportion of normal samples is very large, and the proportion of abnormal samples is very small. The anomaly rate represents the proportion of anomalous samples to the total number of samples. Since, in real network datasets, the ratio of anomalous data is small, the ratio of normal data is large. This experiment sets the anomaly rate to two smaller values (0.05 and 0.1) to simulate the real environment [32], comparing the performance of five anomaly detection baseline algorithms and three fusion methods on different datasets. The results are shown in Tables 3 and 4, respectively.

The motivation we use the multiclassifier fusion technique is to improve the accuracy of anomaly detection; hence, AUC and RECALL are the two most important metrics. In addition, the time overhead for each fusion technique should also be used as an indicator to measure its performance.

Among the three fusion methods, NB (Naive Bayes) performs better than MV (Majority Voting) and WMV (Weighted Majority Voting) in both RECALL and AUC; besides, the fusion takes a similar amount of time. So, the MF model in this study selects Naive Bayes as the fusion method.

On the contrary, multiclassifier fusion models combine multiple classifiers and must ensure that the overall detection is better than the best single classifier. If the overall performance of the fusion model is slightly better than the worst classifier, but slightly worse than the best classifier, then, in this case, it is better to let the best classifier work on its own and not participate in the mix. From Tables 3 and 4, it can be observed that the overall detection performance of NB is better than that of a single classifier in all three datasets, while the detection

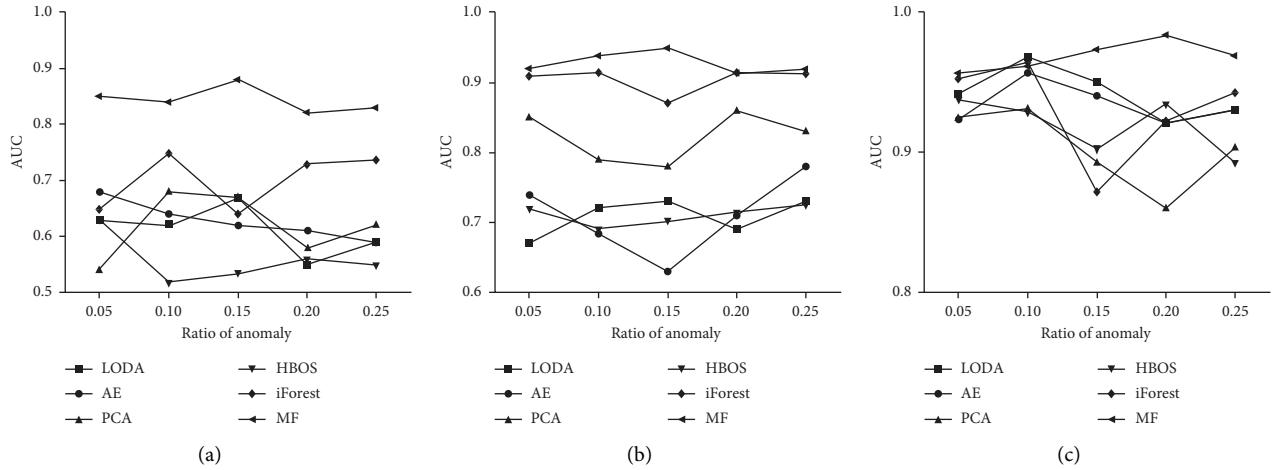

Figure 3: AUC of different datasets varies with the anomaly rate. (a) CICIDS2017. (b) UNSW-NB 15. (c) KDDCUP 99.

performance of MV and WMV is sometimes worse than that of a single detection algorithm.

In terms of time overhead, this study randomly samples 100,000 pieces of data in each of the three datasets, and the time costs of the three basic anomaly detectors, HBOS, LODA, and PCA, are close to each other and significantly faster than AE and iForest. This is related to the fact that the first three algorithms are suitable for large amounts of data. The time of the last two datasets (UNSW-NB 15 and KDDCUP 99) was slightly higher than that of the first dataset, CIDIDS2017, due to the fact that the dimensions of the latter two datasets were higher than those of CICIDS2017. It should be noted that the times of MV, WMV, and NB represent only the fusion time of their respective results and do not include the training time of the base classifier.

As shown in Table 3, when the anomaly rate is set to 0.05, the MF model proposed in this study is superior to the other five anomaly detection baseline algorithms in terms of the performance of both the AUC and RECALL indicators.

On the CICIDS2017 dataset, the RECALL and AUC of the MF were 0.886 and 0.852, respectively, which were 0.355 higher than the HBOS (0.531) with the highest RECALL rate and 0.195 higher than iForest (0.657), which had the best AUC performance.

On the UNSW-NB 15 dataset, the RECALL and AUC of the MF were 0.961 and 0.925, respectively, an increase of 0.24 and 0.12 compared to iForest (0.937 and 0.913), which had the highest RECALL and AUC indicators.

On the KDDCUP 99 dataset, the recall and AUC of MF were 0.995 and 0.956, respectively, an increase of 0.12 and 0.03 compared with iForest (0.983 and 0.953), which had the highest RECALL and AUC indicators.

As shown in Table 4, when the anomaly rate is set to 0.1, a similar conclusion can be seen from Table 4: MF outperforms the other five baseline methods in both AUC and RECALL in three datasets.

The results of Tables 3 and 4 prove that the network intrusion anomaly detection model MF proposed in this study based on multiclassifier fusion technology has more advantages in anomaly detection and performs better than the five state-of-the-art unsupervised learning algorithms.

5.4. Performance Comparison at Different Anomaly Rates. In order to better verify the model proposed in this study, this experiment also compares the detection results of five baseline algorithms for anomaly detection and the MF model in the case of continuous changes in the anomaly rate on three public datasets.

As Figures 2 and 3 shown, with the increase in the anomaly rate, the values of AUC and RECALL of the other five baseline methods are always lower than the MF model and have obvious fluctuations, which means the instability of performance. In contrast, the performance (AUC and RECALL) of the MF model was stable, and the polyline is flatter. This also verifies that the MF model has good robustness and can better adapt to the dynamic changes in attack scale and attack numbers in different network environments.

6. Conclusions

Anomaly detection refers to the detection of data or behaviors that do not conform to normal expected patterns in a large amount of data and is widely used in the field of data security and network security. Aiming at the problem that the detection effect of unsupervised learning methods cannot meet the anomaly detection requirements of the real network environment, this study proposes a network intrusion anomaly detection model MF based on multiclassifier fusion technology. The model can use different fusion methods, such as majority voting fusion, weighted majority voting fusion, and Naive Bayes fusion, to intelligently fuse the detection results of multiple anomaly detection baseline methods (such as LODA, AutoEncoder, PCA, HBOS, and iForest) to obtain a higher detection effect than a single detection method. Finally, the MF model is compared with other anomaly detection baseline methods on three public network intrusion anomaly detection datasets, such as CICIDS2017, UNSW-NB 15, and KDDCUP

99. Experimental results show that the network intrusion anomaly detection model MF proposed in this study based on multiclassifier fusion technology successfully improves the accuracy and effectiveness of detection and has good robustness.

In future research, more advanced intelligent fusion technology will be used to fuse the results of the baseline method to obtain better performance.

Data Availability

All data included in this study can be obtained from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest regarding the publication of this study.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (nos. 62072074, 62076054, 62027827, 61902054, and 62002047), the Frontier Science and Technology Innovation Projects of National Key R&D Program (no. 2019QY1405), the Sichuan Science and Technology Innovation Platform and Talent Plan (no. 2020TDT00020), and the Sichuan Science and Technology Support Plan (no. 2020YFSY0010).

References

[1] S. E. Smaha, "Haystack: an intrusion detection system," in *Proceedings of the IEEE Fourth Aerospace Computer Security Applications Conference*, pp. 37–44, Orlando, FL, September 1988.

[2] S. Staniford, J. A. Hoagland, and J. M. McAlerney, "Practical automated detection of stealthy portscans," *Journal of Computer Security*, vol. 10, no. 1-2, pp. 105–136, 2002.

[3] D. Heckerman, *A Tutorial on Learning with Bayesian Networks*, Technical Report MSRTR-95-06, Microsoft Research, Bengalaru, India, 1995.

[4] K. S. Valdes, "Adaptive model-based monitoring for cyber attack detection," in *Proceedings of the Recent Advances in Intrusion Detection Toulouse*, pp. 80–92, France, October 2000.

[5] D. M. Kruegel, W. Robertson, and F. Valeur, "Bayesian event classification for intrusion detection," in *Proceedings of the 19th Annual Computer Security Applications Conference*, Las Vegas, NV, December 2003.

[6] S. Wold, K. Esbensen, and P. Geladi, "Principal component analysis," *Chemometrics and Intelligent Laboratory Systems*, vol. 2, no. 1-3, pp. 37–52, 1987.

[7] M.-L. Shyu, S.-C. Chen, K. Sarinnapakorn, and L. Chang, "A novel anomaly detection scheme based on principal component classifier," in *Proceedings of the IEEE Foundations and New Directions of Data Mining Workshop*, pp. 172–179, Melbourne, FL, USA, November 2003.

[8] I. Ahmad, M. Basheri, M. J. Iqbal, and A. Rahim, "Performance comparison of support vector machine, random forest, and extreme learning machine for intrusion detection," *IEEE Access*, vol. 6, pp. 33789–33795, 2018.

[9] K. Park, Y. Song, and Y. G. Cheong, "Classification of attack types for intrusion detection systems using a machine learning algorithm," in *Proceedings of the 2018 IEEE Fourth International Conference on Big Data Computing Service and Applications (BigDataService)*, pp. 282–286, Bamberg, Germany, March 2018.

[10] S. Bhattacharya, P. K. R. Maddikunta, P. K. R. Maddikunta et al., "A novel PCA-firefly based XGBoost classification model for intrusion detection in networks using GPU," *Electronics*, vol. 9, no. 2, p. 219, 2020.

[11] R. Vinayakumar, M. Alazab, K. P. Soman, P. Poornachandran, A. Al-Nemrat, and S. Venkatraman, "Deep learning approach for intelligent intrusion detection system," *IEEE Access*, vol. 7, pp. 41525–41550, 2019.

[12] M. Alazab, S. Venkatraman, P. Watters, M. Alazab, and A. Alazab, "Cybercrime: the case of obfuscated malware," in *Global Security, Safety and Sustainability & E-Democracy*, pp. 204–211, Springer, Thessaloniki, Greece, 2011.

[13] S. Huda, J. Abawajy, M. Alazab, M. Abdollalihian, R. Islam, and J. Yearwood, "Hybrids of support vector machine wrapper and filter based framework for malware detection," *Future Generation Computer Systems*, vol. 55, pp. 376–390, 2016.

[14] R. U. Khan, X. Zhang, M. Alazab, and R. Kumar, "An improved convolutional neural network model for intrusion detection in networks," in *Proceedings of the 2019 Cybersecurity and Cyberforensics Conference(CCC)*, pp. 74–77, Melbourne, Australia, May 2019.

[15] R. Vinayakumar, M. Alazab, K. P. Soman, P. Poornachandran, and S. Venkatraman, "Robust intelligent malware detection using deep learning," *IEEE Access*, vol. 7, pp. 46717–46738, 2019.

[16] G. Karatas, O. Demir, and O. K. Sahingoz, "Deep learning in intrusion detection systems," in *Proceedings of the 2018 International Congress on Big Data, Deep Learning and Fighting Cyber Terrorism (IBIGDELFT)*, pp. 113–116, ANKARA, Turkey, December 2018.

[17] S. M. Othman, F. M. Ba-Alwi, N. T. Alsohybe, and A. Y. Al-Hashida, "Intrusion detection model using machine learning algorithm on Big Data environment," *J. Big Data*, vol. 5, no. 1, p. 34, 2018.

[18] T. R. Gadekallu and N. Khare, "Cuckoo search optimized reduction and fuzzy logic classifier for heart disease and diabetes prediction," *International Journal of Fuzzy System Applications*, vol. 6, no. 2, pp. 25–42, 2017.

[19] G. T. Reddy and N. Neelu, "Hybrid firefly-bat optimized fuzzy artificial neural network based classifier for diabetes diagnosis," *International Journal of Intelligent Engineering and Systems*, vol. 10, no. 4, pp. 18–27, 2017.

[20] N. Khare and G. T. Reddy, "Heart disease classification system using optimised fuzzy rule based algorithm," *International Journal of Biomedical Engineering and Technology*, vol. 27, no. 3, pp. 183–202, 2018.

[21] G. T. Reddy, M. P. K. Reddy, K. Lakshmanna, D. S. Rajput, R. Kaluri, and G. Srivastava, "Hybrid genetic algorithm and a fuzzy logic classifier for heart disease diagnosis," *Evolutionary Intelligence*, vol. 13, no. 2, pp. 185–196, 2019.

[22] A. Dainotti, A. Pescapé, and C. Sansone, "Early classification of network traffic through multi-classification," in *Proceedings of the International Workshop on Traffic Monitoring and Analysis*, Springer, Berlin, Heidelberg, 2011.

[23] D. C. Le and N. Zincir-Heywood, "Anomaly detection for insider threats using unsupervised ensembles," *IEEE*

Transactions on Network and Service Management, vol. 18, no. 2, pp. 1152–1164, 2021.

[24] H. D. Nguyen, K. P. Tran, S. Thomassey, and M Hamad, "Forecasting and Anomaly Detection approaches using LSTM and LSTM Autoencoder techniques with the applications in supply chain management," *International Journal of Information Management*, vol. 57, Article ID 102282, 2021.

[25] R. Patil, R. Biradar, V. Ravi, P. Biradar, and U Ghosh, "Network traffic anomaly detection using PCA and BiGAN," *Internet Technology Letters*, vol. 5, no. 1, p. e235, 2022.

[26] O. Abdelrahman and P. Keikhosrokiani, "Assembly line anomaly detection and root cause analysis using machine learning," *IEEE Access*, vol. 8, Article ID 189661, 2020.

[27] N. Paulauskas and A. Baskys, "Application of histogram-based outlier scores to detect computer network anomalies," *Electronics*, vol. 8, no. 11, p. 1251, 2019.

[28] F. T. Liu, K. M. Ting, and Z. H. Zhou, "Isolation-based anomaly detection," *ACM Transactions on Knowledge Discovery from Data*, vol. 6, no. 1, pp. 1–39, 2012.

[29] L. I. Kuncheva, *Combining Pattern Classifiers: Methods and Algorithms*, Wiley-Interscience, The Netherland, 2004.

[30] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," *ICISSp*, vol. 1, pp. 108–116, 2018.

[31] N. Moustafa and J. Slay, "The evaluation of Network Anomaly Detection Systems: statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 data set," *Information Security Journal: A Global Perspective*, vol. 25, no. 1-3, pp. 18–31, 2016.

[32] Y. Wu, LiWei, M. Ni, and Z. Xu, "Anomaly detection model based on one-class support vector machine fused deep auto-encoder," *Computer Science*, vol. 49, no. 3, pp. 144–151, 2022.