Hindawi
Security and Communication Networks
Volume 2018, Article ID 6026878, 13 pages
https://doi.org/10.1155/2018/6026878

Research Article

LA-GRU: Building Combined Intrusion Detection Model Based on Imbalanced Learning and Gated Recurrent Unit Neural Network

Binghao Yan and Guodong Han

National Digital Switching System Engineering & Technology Research Center, Zhengzhou, Henan 450001, China

Correspondence should be addressed to Binghao Yan; ndscybh@qq.com

Received 13 May 2018; Accepted 12 August 2018; Published 27 August 2018

Academic Editor: Bela Genge

The intrusion detection models (IDMs) based on machine learning play a vital role in the security protection of the network environment, and, by learning the characteristics of the network traffic, these IDMs can divide the network traffic into normal behavior or attack behavior automatically. However, existing IDMs cannot solve the imbalance of traffic distribution, while ignoring the temporal relationship within traffic, which result in the reduction of the detection performance of the IDM and increase the false alarm rate, especially for low-frequency attacks. So, in this paper, we propose a new combined IDM called LA-GRU based on a novel imbalanced learning method and gated recurrent unit (GRU) neural network. In the proposed model, a modified local adaptive synthetic minority oversampling technique (LA-SMOTE) algorithm is provided to handle imbalanced traffic, and then the GRU neural network based on deep learning theory is used to implement the anomaly detection of traffic. The experimental results evaluated on the NSL-KDD dataset confirm that, compared with the existing state-of-the-art IDMs, the proposed model not only obtains excellent overall detection performance with a low false alarm rate but also more effectively solves the learning problem of imbalanced traffic distribution.

1. Introduction

With the development of information technology, the Internet has penetrated into every aspect of people's work and life, bringing great convenience. However, attacks on the Internet have also emerged in an endless stream, and the means of invading the computer system have been increased, presenting a trend of intelligentialization and collectivization. Besides, the frequent occurrence of network attacks not only does damage the interests of netizens but also poses threats and challenges to social economy and national security. Therefore, in order to protect the security of the system, as an active and dynamic defense technology, the research and application of intrusion detection technology (IDT) are increasingly important. IDTs collect and analyze certain key information on the networks and hosts, then detect whether there is an event or behavior that violates the security policy, and alert the detected event.

Traditional IDTs include misuse detection and anomaly detection [1]. An intrusion detection model (IDM) based on misuse detection first encodes known attack signatures and possible system defects and stores them in the knowledge base, and then the monitored network traffic is matched with the attack patterns in the knowledge base. When some matches are successful, it indicates that intrusion behaviors have occurred and trigger a corresponding response mechanism at the same time. The advantage of this IDT is the fact that it can establish an effective IDM in a targeted manner with low false alarm rate. However, the disadvantage is that it is incapable of acting on unknown intrusions and variants of known invasive activities. By contrast, IDM based on anomaly detection establishes the normal working mode of the protected objects and believes that any behavior that deviates from the normal behavior pattern to a certain degree is an intrusion event. Its biggest strength is that it can detect attacks that

have never been seen before but has a high false positive rate.

So as to overcome the shortcomings of traditional IDSs, researchers have introduced many machine learning algorithms such as support vector machines (SVM) [2], artificial neural networks (ANN) [3], and decision trees (DT) [4] into the construction of IDMs and have achieved very satisfactory results. The machine learning algorithms have strong self-adaptive and dynamic learning ability, can quickly adjust the detection target according to the change of the network environment, and avoid the shortage of traditional IDTs that require a large amount of domain specialist knowledge. However, IDMs based on machine learning still have the following deficiencies.

(a) It is difficult to detect low-frequency attacks. First, anomaly traffic is far less than normal traffic in quantity, which makes it difficult for IDMs to capture classification features, resulting in the fact that the machine learning algorithms cannot effectively establish the detection model of the attack behaviors. Secondly, the machine learning algorithms do not take into account the overall data distribution in the detection process [5], so that the decision function will be biased towards the majority samples, and the low-frequency attack samples are ignored as noise points, thus producing the wrong detection results.

(b) IDMs do not consider temporal relationship between traffic samples. The existing IDMs generally consider the traffic data distribution as a whole to fit the sample values in the data space. Therefore, the model cannot form effective memory for the traffic samples that have been detected before, resulting in a low detection rate of the model and waste of computing resources in some cases.

Therefore, in order to deal with the above two problems at the same time, we propose a new combined IDM based on local adaptive synthetic minority oversampling technique (LA-SMOTE) algorithm and gated recurrent unit (GRU) neural network in this paper, named LA-GRU. The LA-GRU model consists of two phases. In the first phase, LA-SMOTE oversamples low-frequency attack samples adaptively to increase the number of low-frequency attack samples from the data level, which helps the classifiers to learn more fully the characteristics of low-frequency attack samples. In the second stage, we use GRU to mine temporal relationships between traffic samples and detect the new dataset generated in the first stage, and different experiments are performed on the NSL-KDD dataset to evaluate the performance of LA-GRU. The main contributions of this paper are as follows.

(a) This paper improves the synthetic minority oversampling algorithm (SMOTE) [6] and proposes a new local adaptive SMOTE algorithm (LA-SMOTE). LA-SMOTE divides low-frequency attack samples into different regions based on the difference in the number of low-frequency attack samples of the same class in the nearest neighbors, and then different synthetic strategies are used for low-frequency attack samples in different regions. Compared with the SMOTE algorithm, the proposed algorithm can achieve the same detection performance by synthesizing fewer new samples, thereby reducing computational costs and resource consumption.

(b) The GRU neural network based on deep learning theory is used to construct the IDM and perform intrusion detection because the temporal relationship between the traffic samples is an important classification feature, which can improve the overall detection ability and detection speed of the model. In addition, GRU has the ability of time series prediction and can detect unknown attack samples.

(c) In order to accelerate the training of the model while avoiding the training process falling into a local optimum, the adaptive moment estimation (Adam) gradient descent optimization algorithm [7] is used to assist the training of the GRU.

The remainder of the paper is organized as follows. Section 2 briefly introduces relevant works related to imbalanced learning problem and deep learning methods used in intrusion detection. The proposed model and methodology for intrusion detection are specified in Section 3 and then, in Section 4, introduction of our test bed, datasets, and experimental preparation process is presented. The experimental results, comparisons, and discussion are given in Section 5. Finally, we conclude and propose the next step in Section 6.

2. Relevant Work

Although the IDMs based on machine learning technologies have powerful detection capabilities and self-adaptive ability to face changes in the network environment, they are still affected by the impact of imbalanced data distribution. For example, Al-Yaseen et al. [8] used support vector machine and modified K-means algorithm to build a multilayer hybrid IDM. However, this model has a very poor detection rate for low-frequency attack samples of user to root attacks (U2R) and remote to local attacks (R2L) in the KDD CUP 99 dataset, only 21.93% and 31.39%, respectively, far below the detection rate of other high-frequency samples.

Therefore, researchers began to find a way out of this dilemma. Parsaei et al. [9] proposed combining the SMOTE with the cluster center and nearest neighbor to perform intrusion detection and improved the detection rate of low-frequency samples from the data level. From the algorithm level, Akyol et al. [10] constructed a multilevel hybrid classifier, based on the multilayer perceptron (selected as the first layer) and C4.5 Decision Tree Algorithm (selected as the second layer), which also achieved good detection results for low-frequency attack samples. Besides, feature selection methods are also helpful to improve the detection performance of low-frequency attacks. Papamartzivanos et al. [11] used decision tree and genetic algorithm to select features that are beneficial to detect low-frequency attacks, and then the detection rate of U2R and R2L is increased to 76.92% and 83.75%, respectively. Zhu et al. [12] proposed an improved nondominated sorting genetic algorithm (I-NSGA) to implement feature selection by adding a new niche preservation procedure in the traditional NSGA, and the experimental results showed that while maintaining high detection rate and low computational complexity, I-NSGA can deal with imbalanced problem very well.

As an emerging and popular theory, the development of deep learning also plays an important role in intrusion

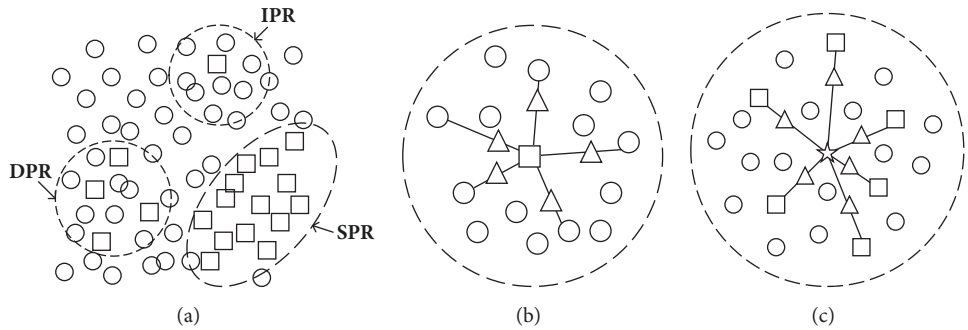

Figure 1: The schematic diagram of the LA-SMOTE algorithm. In the figure, circles denote high-frequency samples, rectangles denote low-frequency samples, triangles denote synthesized samples, and pentagon denotes the mean value point of the low-frequency samples in DPR. (a) Regional division. (b) Sample generation in IPR. (c) Sample generation in DPR.

detection, especially in the process of large-scale data processing. Wang et al. [13] proposed an IDM based on convolutional neural network to reduce the false alarm rate with millions of samples. Shone et al. [14] used nonsymmetric deep autoencoder and random forests to achieve unsupervised feature learning and intrusion detection, which achieved high detection rate while reducing the training time of the model. In addition to the field of traditional network security, Diro et al. [15] utilized the self-learning and compression characteristics of deep neural networks to propose a distributed IDM for the Internet of Things, and the experimental results have proved that the deep model is more effective than the previous shallow models. Similarly, Ma et al. [16] combined deep neural networks with spectral clustering algorithms to propose a hybrid intrusion detection model for wireless sensor networks and achieved higher detection rates and accuracy in real environments. Loukas et al. [17] developed a new vehicle-based IDS based on recurrent neural network and deep neural network architecture, which opened up a new research path for the safety of vehicle systems.

By summarizing and analyzing the current state-of-the-art methods, we believe that although existing IDMs have achieved good detection performance, there are still areas where improvement can be made. To the best of our knowledge, no one has yet built an IMD by combining the deep learning algorithm with the processing method of imbalanced data distribution, and at the same time it takes the temporal relationship between traffic into account, which are the research highlights of the proposed model in this paper.

3. Methodology

In this section, we introduce the basic theory and implementation rules of the improved LA-SMOTE algorithm and how to use the GRU to capture temporal information and perform anomaly detection of traffic samples.

3.1. LA-SMOTE Algorithm. As an effective oversampling algorithm, SMOTE [6] algorithm achieves the increment of low-frequency samples by performing random and linear interpolation between low-frequency samples and their congeneric nearest neighbors. The advantage of the SMOTE algorithm is that it generates new samples during the oversampling process, instead of monotonously replicating the original samples, which avoids the increase of erroneous samples and facilitating the classification process of the classifiers. However, the SMOTE algorithm not only does not consider the spatial distribution dispersion of the low-frequency sample and the problem of outliers processing but also increases the number of all low-frequency samples, resulting in a waste of some computing resources. Therefore, in order to solve the above problems and make oversampling algorithm suitable for intrusion detection, a new local adaptive SMOTE algorithm (LA-SMOTE) is proposed based on the SMOTE algorithm in this paper.

For each low-frequency sample $\boldsymbol{x}$, we select its k nearest neighbors and calculate the number of high-frequency samples contained in k nearest neighbors, which is denoted by k_h. Then, according to the size of k_h, we assign each low-frequency sample to different regions. We believe that there are differences in the difficulty of classification of low-frequency samples in different regions. Therefore, different methods are used to process low-frequency samples in different regions.

The LA-SMOTE algorithm is divided into the following three steps.

Step 1 (regional division). (a) If ($k_h = k$), it indicates that there are no congeneric samples in the k nearest neighbors of the low-frequency sample $\boldsymbol{x}$. We call such low-frequency samples outliers and assign them to the independent point region (IPR).

(b) If ($k/2 \le k_h < k$), it indicates that the number of high-frequency samples in the nearest neighbors is greater than the number of low-frequency samples. We call the low-frequency samples in such cases dangerous points and assign them to the danger point domain (DPR).

(c) If ($0 \le k_h < k/2$), it indicates that the number of high-frequency samples in the nearest neighbor is less than the number of low-frequency samples. We call the low-frequency samples in this case secure points and divide them into the safety point domain (SPR).

The schematic diagram of the regional division according to the above rules is shown in Figure 1(a).

Step 2 (sample generation). (a) For low-frequency samples belonging to IPR, unlike areas such as image processing, they should not be considered as noise and ignored in intrusion detection. Therefore, similar to the SMOTE algorithm, a new sample is generated between the low-frequency sample and its nearest neighbor high-frequency samples according to (1) and schematic diagram is shown in Figure 1(b).

$$\boldsymbol{x}_{new} = \boldsymbol{x} + u_{[0,1]} \left(\boldsymbol{x}_h - \boldsymbol{x} \right) \tag{1}$$

where $\boldsymbol{x}_{new}$ is the newly generated low-frequency sample, $\boldsymbol{x}_h$ is the high-frequency sample in IPR, and $u_{[0,1]} \in [0, 1]$ is a random number.

(b) For low-frequency samples that belong to DPR, they are mixed with high-frequency samples under such circumstances, and the spatial distribution dispersion of low-frequency sample set is high. Therefore, we calculate and select the mean of all low-frequency samples in DPR as the center of class, and new samples are inserted between the low-frequency samples and the center point to reduce the spatial distribution dispersion of the newly generated low-frequency sample set, so that the low-frequency samples are more easily identified. The generation rule is shown in (2) and (3), and the schematic diagram is shown in Figure 1(c).

$$\boldsymbol{x}_m = \frac{1}{k - k_h} \sum_{i=1}^{k-k_h} \boldsymbol{x}_i \tag{2}$$

$$\boldsymbol{x}_{new} = \boldsymbol{x} + u_{[0,1]} \left(\boldsymbol{x}_m - \boldsymbol{x} \right) \tag{3}$$

where $\boldsymbol{x}_m$ is the mean of all low-frequency samples in DPR.

(c) For low-frequency samples that belong to SPR, we believe that such low-frequency samples are not easily misclassified, so no action is required to generate new samples.

Step 3 (add timestamp for new samples). The IDM presented in this paper regards the temporal relationship between network traffic as part of the classification feature; however, the new samples are synthesized artificially using the LA-SMOTE algorithm on the basis of the actual samples, and there is no realistic temporal relationship between new samples and original samples. Therefore, it is necessary to add timestamp for the new synthesized samples thus making them suitable for the GRU.

From Step 2, we can know that there is a linear relationship between the new samples in the IPR and DPR and the original low-frequency samples used to generate new samples; thus we believe that these new samples not only have similar classification characteristics with the original low-frequency samples but also are closer in timing relationship. Therefore, after input low-frequency samples, we input new samples generated by them accordingly. In particular, the multiple new samples synthesized by the same original sample in IPR are inputted in sequence according to the order of their generation.

3.2. Gated Recurrent Unit. Deep neural network (DNN) breaks through the limitations of shallow networks in terms of sample classification and feature extraction and has a strong ability of nonlinear fitting. However, the traditional DNNs do not take into account the temporal relationship between the classified samples, resulting in the loss of some information in classification process. The proposal and development of the recurrent neural network (RNN) [18] effectively solve the problem of timing dependence. RNN introduces the feedback connection between the hidden layer units, so that the network can retain the learned information to the current moment and determine the final output results of the network together with the input of the current moment. The effectiveness of the RNN in solving timing problems has been validated in many research areas and has yielded many encouraging results, such as image description [19], speech recognition [20], and machine translation [21]. In addition, RNN is also used for the prediction of events related to time series, such as stock prediction [22, 23]. However, in the process of training, RNN faces the problem of vanishing gradients, which leads to the failure of the network to converge normally, and RNN cannot overcome the impact of long-term dependencies [24].

Many network structures for improving the RNN are proposed, and one of the most widely used and effective structures is the Long Short Term Memory (LSTM) [25]. The difference between LSTM and RNN is that LSTM adds a "processor" to judge whether the information is useful or not, which is called cell. A cell includes three gates, called input gate, forget gate, and output gate, respectively. When past and new information enter the cell of LSTM, it can be judged according to the rules whether it is useful, only the information that is consistent with the authentication of the algorithm will be left, and the incompatible information will be forgotten through the forget gate. The current state-of-the-art research results and literatures [26–28] show that LSTM is an effective technology for solving long-term dependencies, and, moreover, the problem of vanishing gradients can also be alleviated through gating mechanism.

As the most popular variant of LSTM, gated recurrent unit (GRU) [29] simplifies the gated structure in LSTM's cell and uses reset gate and update gate to replace three gates in LSTM, in which the reset gate determines how to combine the new input information with the previous memory, and the update gate defines how much of the previous information needs to be saved to the current time step. GRU has also achieved satisfactory results in many popular research areas [30, 31], and, compared with LSTM, GRU can save more training time and computing resources.

Illustration for the unfolded structures of GRU is shown in Figure 2. The basic calculation process of GRU is shown in the following equations.

(a) Candidate State

$$\widetilde{h}_t = g\left(W_{fh} x_t + W_{rh} \left(h_{t-1} \odot r_t \right) + \varphi_h \right) \tag{4}$$

(b) Reset Gate

$$r_t = f\left(W_{fr} x_t + W_{rr} h_{t-1} + \varphi_r \right) \tag{5}$$

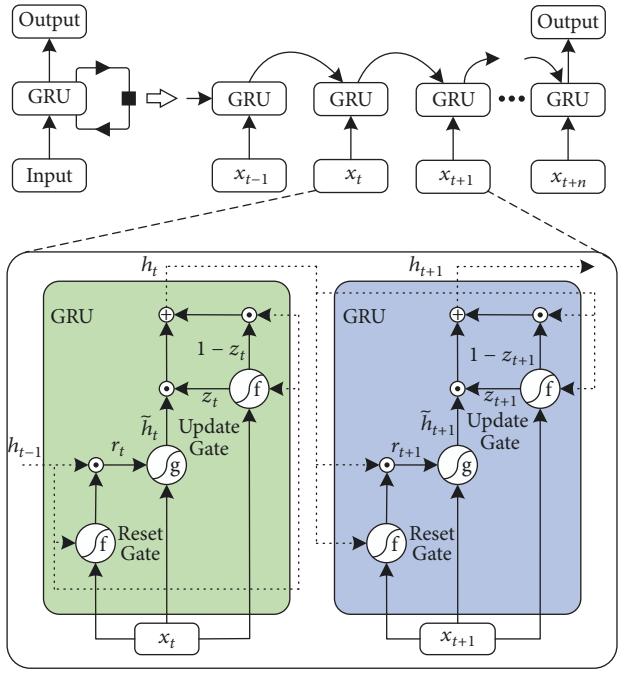

FIGURE 2: Illustration for the unfolded structures of GRU.

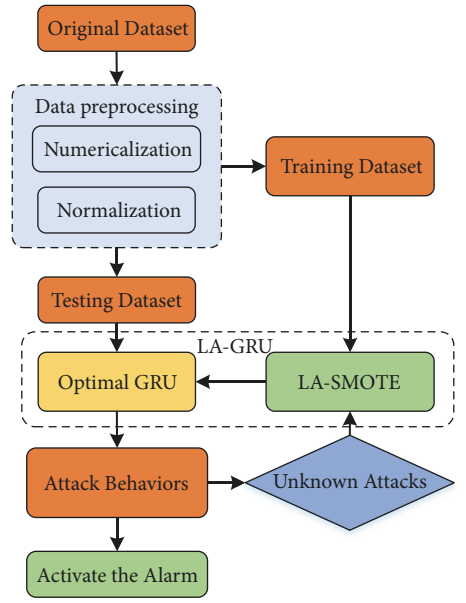

FIGURE 3: The framework of the proposed LA-GRU IDM.

(c) Update Gate

$$z_t = f\left(W_{fz}x_t + W_{rz}h_{t-1} + \varphi_z\right) \tag{6}$$

(d) Current State

$$h_t = \left(1 - z_t\right) \odot \tilde{h}_t + z_t \odot h_{t-1} \tag{7}$$

where r_t and z_t denote the output vector of the reset gate and update gate at current time step t, while h_t and $\tilde{h}_t$ denote the state vector and the candidate state vector at the current time step t, respectively. $\varphi_h \in R^{n\times 1}$, $\varphi_r \in R^{n\times 1}$, and $\varphi_z \in R^{n\times 1}$ are the bias vectors. $W_{fh} \in R^{n\times m}$, $W_{fr} \in R^{n\times m}$, and $W_{fz} \in R^{n\times m}$ are the weight matrices of the feed-forward connections, respectively. Besides, $W_{rh} \in R^{n\times n}$, $W_{rr} \in R^{n\times n}$, and $W_{rz} \in R^{n\times n}$ are the weight matrices of the recurrent connections, respectively. In particular, weights sharing mechanism is applied to different time steps t. $\odot$ represents the operational rule of element-wise multiplications between vectors. $g(\cdot)$ and $f(\cdot)$ denote neuronal activation functions, where $g(\cdot)$ is the tanh function and $f(\cdot)$ is the sigmoid function in this paper.

Due to the multiobjective classification involved in the experiments, Softmax classifier is used as the final output layer of GRU in this paper:

$$softmax\left(y_i\right) = \frac{e^{y_i}}{\sum_{i=1}^{I} e^{y_i}} \tag{8}$$

where y_i represents the ith element of the GRU's output vector and satisfies $\sum_{i=1}^{I} softmax(y_i) = 1$. I is the dimension of the output vector.

The cross-entropy loss function is selected for the error function. Besides, in order to accelerate the gradient descent in the process of error backpropagation, the adaptive moment estimation (Adam) [7] algorithm is used as optimizer to make the GRU converge as soon as possible and avoid parameter values falling into the local optimal.

4. Experimental Setup

4.1. LA-GRU Overview. The framework of the proposed LA-GRU intrusion detection model is shown in Figure 3. First, the preprocessing operations including numericalization and normalization are performed on the original dataset so that the samples are suitable for oversampling algorithm and GRU network. Then we divide the processed dataset into training datasets and testing datasets. After that, we use the proposed LA-SMOTE algorithm to process the low-frequency attack samples in the training datasets and use the new datasets after oversampling to train the GRU network so that optimal weights are obtained. Testing datasets are used to verify the validity of the model. Finally, it sends alerts to user if the attack behaviors are detected, and the unknown attack samples which are detected in the testing datasets are reinput into the LA-GRU model. After processing by the LA-SMOTE algorithm, the unknown attack samples are input into the GRU network to continue fine-tuning weights in order to improve the robustness of the model.

Figure 4 shows the placement of the LA-GRU model proposed in this paper in the actual network environment. Since the primary purpose of IDM is to discover attacks that are internal to the system or host, the LA-GRU should be placed behind the firewall and complement each other's advantages. In the real application process, the LA-GRU is trained and tuned using the manually processed actual network traffic dataset, and the obtained optimal model is

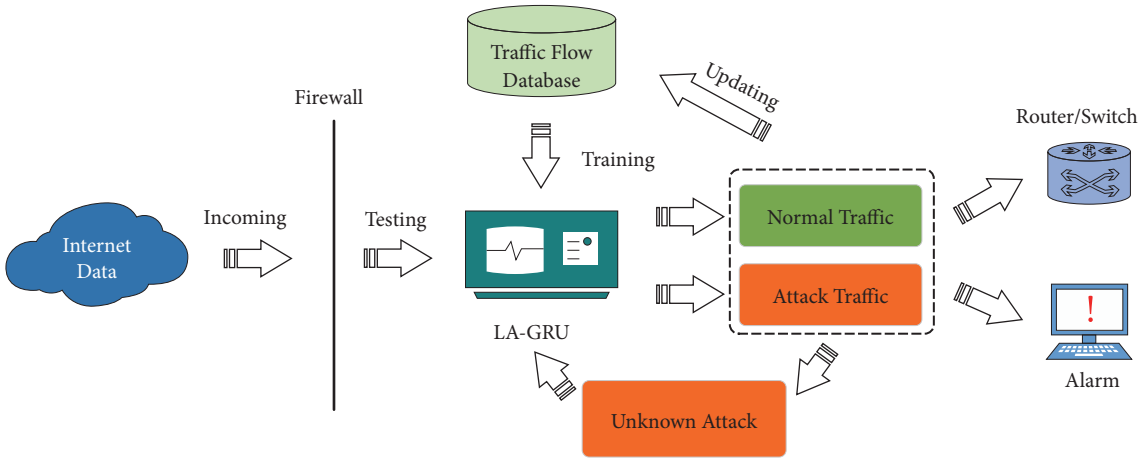

Figure 4: The network architecture of the proposed LA-GRU IDM.

Table 1: Distribution of samples in NSL-KDD dataset.

Dataset	Normal	Attack				Total
		DOS	Probing	R2L	U2R	
Training	67343	45927	11656	995	52	125973
Testing	9691	7457	2421	2754	220	22543

Table 2: Size and distribution of samples in new dataset.

No.	Training set		Testing set	
	Normal	Attacks	Normal	Attacks
1	12323	2853	5324	4645
2	13864	3298	4795	3926
3	10756	2901	5674	4209
4	11348	2988	5352	3874
5	10466	3109	5168	3982

placed at the important nodes of the network to detect the data transmitted through the Internet or other hosts. For traffic flow that is detected and considered to be normal behavior, it is allowed to flow through the node and be transmitted to the destination via router or switch. For traffic flow considered to be attack behavior, the user is alerted through the computer interaction interface and the corresponding measures are taken to prevent the protected network from being affected. In addition, the database needs to be updated periodically and the LA-GRU is also updated during the detection process to maintain the stability.

4.2. Dataset. The simulation experiments in this paper are based on the NSL-KDD [32] dataset to verify the validity of the proposed model. The NSL-KDD dataset is improved on the KDD CUP 99 dataset [33] and is one of the most commonly used standard datasets for intrusion detection. Besides, the NSL-KDD dataset removes the redundant data contained in the KDD CUP 99 dataset and adds new attack types and samples not included in the training dataset into the testing dataset to verify the detection performance of the IDM for unknown attacks. The NSL-KDD dataset consists of 125973 training samples and 22543 testing samples, in which the attack samples are divided into four types: denial of service attacks (DOS), probing attacks (Probe), remote to local attacks (R2L), and user to root attacks (U2R); the specific distribution of the attack types and numbers is shown in Table 1.

To accelerate the experimental process, we sample the original samples in NSL-KDD dataset randomly and reassemble the extracted samples into multiple new independent datasets, as shown in Table 2. The samples in the new datasets after sampling still maintain the original arrangement order, in order to preserve the temporal relationship between the samples. In particular, because U2R's training samples are not enough to sample, they are all retained in every new training dataset. Moreover, independent and repeated experiments on each new dataset are carried out several times in the experiment, and 10-fold cross-validation is performed at the same time to ensure that the final experimental results have good unbiasedness and reference.

4.3. Data Preprocessing. Each sample in the NSL-KDD dataset consists of 41 attribute features, including 38 numeric attribute features and 3 symbolic attribute features. Data preprocessing includes two steps: (a) transform the symbolic attribute features into numeric attribute features and (b) perform the minimum-maximum normalization on the feature values.

(a) Numeralization. The input of the classifier requires that the data must be a numeric vector, so that three symbolic features of *Protocol_type* (including 3 different symbolic feature values), *Service* (including 70 different symbolic feature values), and *Flag* (including 11 different symbolic feature values) in the NSL-KDD dataset need to be mapped into binary numeric features. For example, we use (0,0,1),

Table 3: Size and distribution of samples in new dataset.

Algorithm	Parameter	Value
LA-SMOTE	Number of nearest neighbors	65
	Over-sampling rate	600%
GRU	Number of nodes in input layer	122
	Number of neurons in hidden layer	75
	Number of neurons in output layer	5
	Batch size	500
Adam	Step size	0.001
	First-order exponential damping decrement	0.9
	Second-order exponential damping decrement	0.999
	Non-zero constant	10^{-8}

(0,1,0), and (1,0,0) to represent TCP, UDP, and ICMP in *Protocol_type*, respectively. Therefore, after the operation of numericalization is completed, the feature dimension of the dataset increases from 41 to 122 dimensions. Similarly, the category labels of attacks are transformed into numerical vectors in the same way.

(b) Normalization. Perform the minimum-maximum normalization on the original feature values so that the values of the features are in the same order of magnitude, which is suitable for the comparative evaluation of the detection performance and reduces the overhead of the computational resources. Linearly map the numeric feature values to the range [0, 1] as shown in (9).

$$x_{new} = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{9}$$

4.4. Metrics. In order to show the experimental results more intuitive, evaluation indicators including ACC (accuracy), DR (detection rate), and FAR (false alarm rate) are adopted to evaluate the detection performance and effectiveness of the proposed model and compare it with other state-of-the-art intrusion detection models and methods. The calculation rules of these evaluation indicators are obtained as follows:

$$ACC = \frac{TN + TP}{TN + TP + FN + FP} \tag{10}$$

$$DR = \frac{TP}{TP + FN} \tag{11}$$

$$FAR = \frac{FP}{FP + TN} \tag{12}$$

where TP represents the true positive, which means normal samples are predicted as normal correctly, TN represents the true negative, which means attack behaviors are correctly detected, FP represents the false positive, which means attack behaviors are classified as normal mistakenly, and FN represents false negative, which means normal samples are predicted as malicious wrongly. Besides, the training time and testing time for different models are denoted by t_{train} and t_{test}, respectively.

4.5. Model Parameters. The parameters of the model proposed in this paper mainly include the following:

(a) The number of neurons in the input layer, output layer, and hidden layer of the GRU network and batch size during network training.

(b) The number of nearest neighbors and oversampling rate of the LA-SMOTE algorithm.

(c) The parameters of Adam algorithm, including step size and first-order and second-order exponential damping decrement and nonzero constant.

The specific parameter values used in the experiments are shown in Table 3. Among them, the number of neurons in hidden layer of the GRU, the number of nearest neighbors, and oversampling rate of the LA-SMOTE algorithm need to be determined through experiments. The selection process of these parameters is described in Sections 5.1, 5.2, and 5.3, respectively.

The number of neurons in the input layer and output layer is 122 and 5, respectively, because the input vector is 122-dimensional and the experiments in this paper are five-category classification. Besides, existing studies have shown [34] that the batch size affects the final experimental results. A small batch size can easily affect the optimization of the model and lead to overfitting, while a large batch size decreases the model's convergence speed. Therefore, given the size of the dataset and the memory capacity used in our experiments, the batch size is set to 500. The parameters in Adam algorithm are based on the author's recommendation [7].

5. Experimental Results and Discussion

The simulation software platform we have used in our experiments is TensorFlow, which is one of the most convenient and popular deep learning frameworks. Meanwhile, a desktop, with Intel Core i7-8700K hexa-core processor, 128G SSD, 16G RAM, and Windows 10 operating system, is selected as the hardware platform to run the empirical experiments. The experiments in this paper can be divided into two parts: the selection process of optimal parameters and the comparison of different models. All experiments are based on new datasets produced by NSL-KDD. Specifically, the content of the experiments mainly includes the following items:

Table 4: Experimental results of different numbers of hidden neurons.

Network Structure	ACC(%)	DR(%)	FAR(%)
[122, 200, 5]	99.33	99.28	0.098
[122, 150, 5]	99.32	99.16	0.106
[122, 122, 5]	99.24	99.16	0.112
[122, 105, 5]	99.13	99.05	0.128
[122, 90, 5]	99.11	98.97	0.131
[122, 75, 5]	99.04	98.92	0.134
[122, 60, 5]	98.32	97.93	0.137
[122, 45, 5]	91.34	90.99	0.343
[122, 35, 5]	88.90	88.09	0.441
[122, 20, 5]	84.41	83.21	0.562
[122, 5, 5]	72.57	71.74	0.741

(a) Evaluate the impact of the number of neurons in hidden layer on the experimental results and select the optimal value.

(b) Evaluate the impact of the number of nearest neighbors on the experimental results and select the optimal value.

(c) Evaluate the impact of oversampling rate on the experimental results and select the optimal value.

(d) Obtain the best experimental results of proposed model and compare with other state-of-the-art IDMs based on imbalanced learning and shallow and deep learning methods, respectively.

5.1. Impact of the Number of Hidden Neurons. The number of input neurons in the GRU network is determined by the dimension of the input vector, and the number of output neurons is determined by the output class. However, there is no fixed selection method for the number of hidden neurons, which needs to be adaptively adjusted according to different experimental objectives. Table 4 shows the effect of different numbers of hidden neurons on the detection results of the model and Figure 4 shows the training time and testing time.

From Table 4, it can be concluded that as the number of hidden neurons increases, the overall detection performance of the model is also improved. However, the training time and testing time of the model have a nonlinear relationship with the number of hidden neurons as can be seen in Figure 5. When the number of hidden neurons exceeds a certain threshold, the training time and testing time of the model increase dramatically, but the detection performance does not change significantly. For example, for the experiments in this paper, the ACC and DR of the model only increase by 0.22% and 0.36%, respectively, when the number of hidden neurons increases from 75 to 200, but the training time and testing time increase by 7.09 times and 2.41 times.

The reason for the above experimental results is mainly because there are three activation functions in the network structure of GRU, and each activation function requires a certain number of neurons to implement the function of the gating mechanism, and the number of neurons required for each activation function is the same, so that the time required for model training and testing is greatly influenced by the number of hidden neurons.

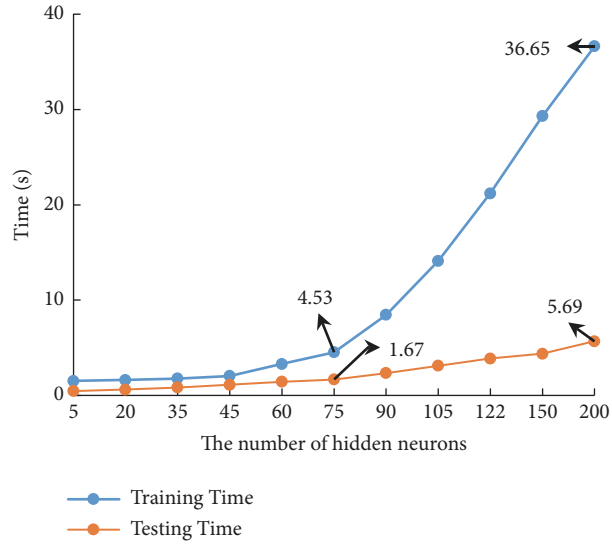

Figure 5: The effect of the number of hidden neurons on the training and testing time.

5.2. Impact of the Number of Nearest Neighbors. Before oversampling, firstly LA-SMOTE algorithm needs to select the nearest neighbors for each low-frequency attack sample and then divides the low-frequency samples into different sets according to the number of high-frequency samples contained in the nearest neighbor set and takes different methods to improve the number of low-frequency samples. Therefore, the number of nearest neighbors will affect the process of dividing the sample set and further change the incremental mode of the low-frequency sample.

On the one hand, if more nearest neighbors are selected, the proportion of high-frequency samples contained in the nearest neighbor set will increase, which may cause low-frequency samples that originally belong to the SPR to be grouped into DPR, and eventually unnecessary new samples are synthesized, resulting in a rise in false alarm rates. On the other hand, the insufficient number of nearest neighbors may result in the low-frequency samples in the DPR being reclassified into SPR and failing to synthesize new samples effectively.

We summarized the experimental results for the number of nearest neighbors in the range of 0 to 100 and Figure 6 clearly shows the effect of different numbers of nearest neighbors on the detection performance of the proposed model. It can be seen from (a) and (b) in Figure 6, as the number of nearest neighbors increases, the ACC and DR gradually increase until a steady state is reached. Additionally, Figure 6(c) shows that the FAR has continuously declined to the minimum point and then increases slightly again. Therefore, after comprehensive consideration, we believe that the detection performance of the model reaches the optimal result when the number of nearest neighbors is 65.

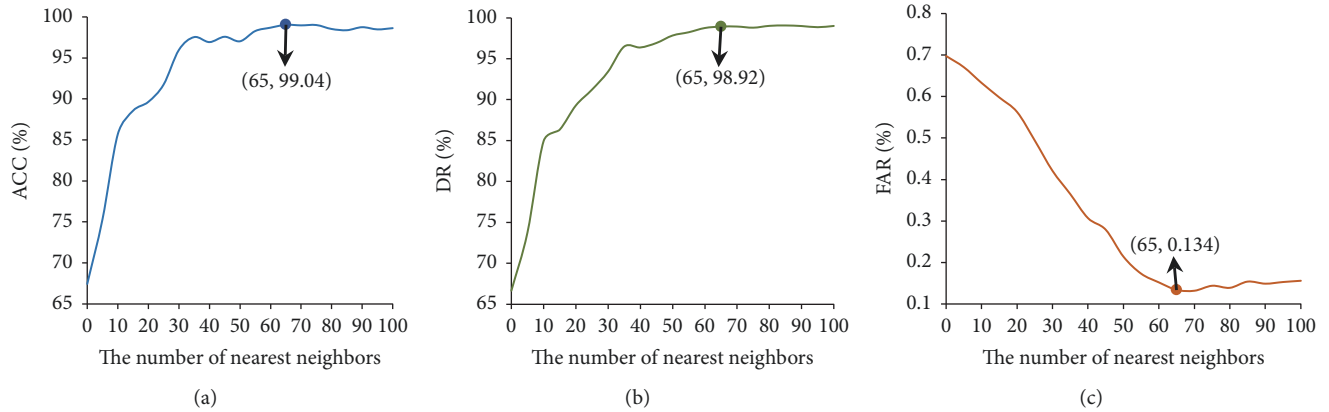

Figure 6: Experimental results of the proposed model with different number of nearest neighbors in LA-SMOTE. (a) ACC. (b) DR. (c) FAR.

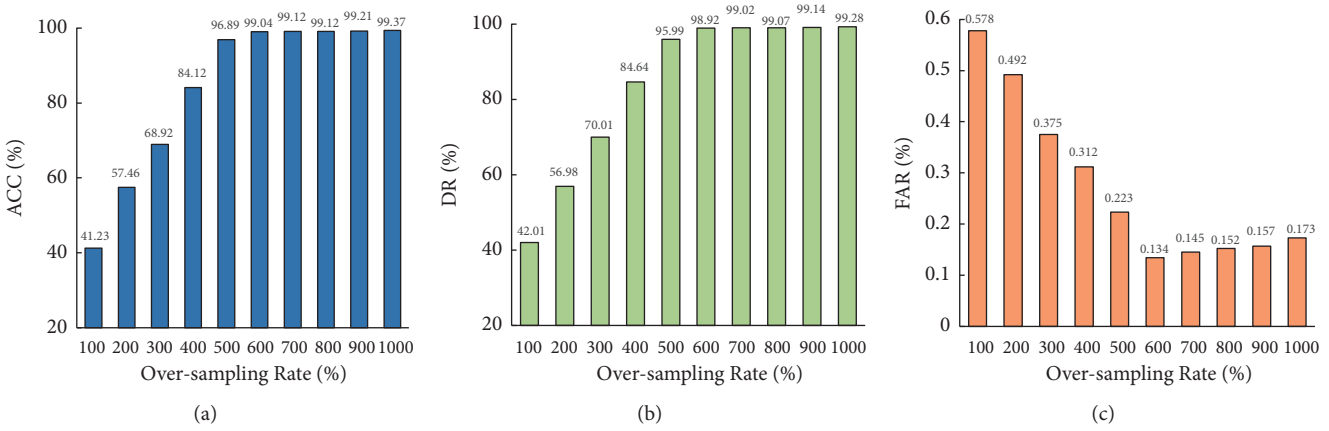

Figure 7: Experimental results of the proposed model with different oversampling rate in LA-SMOTE. (a) ACC. (b) DR. (c) FAR.

5.3. Impact of the Size of Oversampling Rate. The oversampling rate in the LA-SMOTE algorithm is a crucial parameter that has a direct impact on the model's performance and detection results. Figure 7 shows the effect of different oversampling rates (range from 100% to 1000%) on the performance of the model. From Figure 7, we can see that when the oversampling rate is less than 600%, although the ACC and DR are in a rising trend, the model does not achieve the optimal results, indicating that attack behaviors cannot be effectively classified with these values of the oversampling rate. Conversely, when the oversampling rate is higher than 600%, the ACC and DR of the model remain basically unchanged, but the increase in FAR is more obvious, which is mainly because LA-SMOTE has produced superfluous new samples.

In a word, although the classification accuracy and detection rate of low-frequency attack samples are proportional to the number of low-frequency attack samples, that is, the more low-frequency attack samples we have, the higher ACC and DR will be, but, at the same time, the excess new generated attack samples will also lead to the enhancement of FAR and the increase of the training time of the model, so, in the follow-up experiments, the oversampling rate of 600% is adopted.

5.4. Comparison of Different Methods. In the first place, in order to verify the validity of the proposed LA-SMOTE algorithm, we set up the comparative experiments to test the effect of different oversampling algorithms on the performance of the IDM and the experimental results are shown in Table 5. In Table 5, *GRU+Nothing* indicates that no oversampling algorithm is used during the establishment of the model, while *GRU+Oversampling* indicates that only the basic oversampling algorithm is used to replicate the low-frequency samples. Similarly, *GRU+SMOTE* and *GRU+LA-SMOTE* indicate the use of SMOTE and LA-SMOTE algorithms, respectively, to process low-frequency samples. All oversampling algorithms used the same oversampling rate during the experiments.

From the experimental results in Table 5, it can be seen that the four methods have approximately the same detection performance for the three high-frequency sample types of Normal, DOS, and Probing. However, for R2L and U2L, the low-frequency attack samples in NSL-KDD dataset, the use of oversampling algorithms greatly improves the detection performance of the IDM. Furthermore, the performance of SMOTE algorithm and LA-SMOTE algorithm is better than that of the basic oversampling algorithm.

Table 5: Experimental results of different oversampling algorithms on the detection performance of the model.

Method	Metrics	Category					GRU's Training Time
		Normal	DOS	Probing	R2L	U2L	
GRU+Nothing	DR(%)	99.19	99.08	98.94	49.32	37.85	3.88s
	FAR(%)	--	0.026	0.026	0.694	0.843	
GRU+Oversampling	DR(%)	99.15	99.14	99.21	88.59	83.73	5.39s
	FAR(%)	--	0.020	0.019	0.158	0.193	
GRU+SMOTE	DR(%)	99.19	99.24	99.16	98.78	98.24	5.31s
	FAR(%)	--	0.018	0.027	0.041	0.049	
GRU+LA-SMOTE	DR(%)	99.21	99.16	99.20	98.34	98.61	4.53s
	FAR(%)	--	0.021	0.025	0.036	0.052	

Table 6: Comparison of overall detection performance among different IDMs.

Method		ACC(%)	DR(%)	FAR(%)	Dataset	Training Dataset Size
Imbalanced Learning	CANN+SMOTE[9]	98.99	99.56	0.557	NSL-KDD	125,973
	MHCVF[10]	98.04	95.57	1.38	KDD CUP 99	494,021
	DENDRON[11]	97.55	95.97	1.08	NSL-KDD	125,973
	I-NGSA[12]	99.37	99.24	N/A	NSL-KDD	125,973
Shallow Learning	SVM[2]	94.22	92.99	3.46	KDD CUP 99	145,585
	OS-ELM[3]	98.66	99.01	1.74	NSL-KDD	125,973
	TLMD[4]	93.32	93.11	0.761	KDD CUP 99	86,000
	GA-LR[35]	99.90	99.81	0.105	KDD CUP 99	494,021
Deep Learning	CNN+LSTM[13]	99.68	97.78	0.07	KDD CUP 99	2,466,929
	S-NADE[14]	97.85	97.85	2.15	KDD CUP 99	494,021
	DNN[15]	99.20	99.27	0.85	NSL-KDD	125,973
	SCDNN[16]	92.03	92.23	7.90	NSL-KDD	62,986
Proposed Method	LA-GRU	99.04	98.92	0.134	NSL-KDD	73,906

In terms of the training time of the model, although the oversampling algorithms increase the time required for model training, we believe that the significant improvement in the detection performance of the low-frequency attack is worthwhile. Specifically, the training time of GRU based on the LA-SMOTE algorithm presented in this paper is shorter than that of the SMOTE algorithm and the basic oversampling algorithm, which shows that the LA-SMOTE algorithm is more effective in synthesizing new samples and can make the classifier achieve the same detection effect by generating fewer new samples compared to the existing oversampling algorithms.

To further demonstrate the effectiveness of the proposed method, different IDMs proposed in the existing literatures have been selected for comparison, including imbalanced data learning methods, shallow learning methods, and deep learning methods. Besides, in order to make the comparison experiments more detailed and cover more state-of-the-art IDMs, the methodologies using the KDD CUP 99 dataset for experiments are also added to the experimental comparison.

The overall detection performance of different models is shown in Table 6. From the comparison results in Table 6, it can be seen that, compared with other imbalanced learning methods, the proposed LA-GRU model uses the least training samples to achieve the optimal FAR. In terms of ACC and DR, the I-NSGA model proposed in [12] and the CANN+SMOTE model proposed in [9] have achieved the best results, respectively. In contrast, although the ACC and DR of the proposed LA-GRU are slightly inferior to those of the above two detection models, we believe that they are still promising results (around 99%) and reach the average detection level.

In comparison experiments between the LA-GRU model and the shallow learning models, the GA-LR reported in [32] obtained the best results in all three evaluation indicators and are slightly higher than the model we proposed. But GA-LR used nearly 500,000 samples during the training process, far more than the number of training samples we used. Therefore, in the same experimental environment, we think that the training time of the GA-LR model is more than the model presented in this paper, which also proves that the LA-GRU's convergence ability is better than the GA-LR, and it is more in line with the real-time requirement of real network.

The similar results also appear in the comparison experiments between LA-GRU and other deep learning models. For these deep learning models, both the CNN+LSTM [13] and DNN [15] obtain higher ACC than the proposed LA-GRU. In addition, the DR obtained by the DNN model is 0.35% higher than that of the proposed LA-GRU, and the FAR of CNN+LSTM is 0.64% lower than that of LA-GRU slightly.

Table 7: Comparison of detection rate among different IDMs for five-category classification.

Method		Normal	DOS	Probe	R2L	U2R
Imbalanced Learning	CANN+SMOTE[9]	N/A	N/A	N/A	92.97	55.91
	MHCVF[10]	94.29	99.99	99.39	80.00	84.05
	DENDRON[11]	98.98	95.94	97.17	83.75	76.92
	I-NGSA[12]	99.58	99.59	96.47	84.61	88.95
Shallow Learning	SVM[2]	96.16	98.06	57.36	22.24	14.29
	OS-ELM[3]	99.07	99.14	90.35	56.75	78.10
	TLMD[4]	99.24	98.57	93.77	56.20	75.71
	GA-LR[32]	99.97	99.98	98.44	95.48	52.17
Deep Learning	CNN+LSTM[13]	N/A	99.10	83.35	74.19	64.25
	S-NADE[14]	99.49	99.79	98.74	9.31	0.00
	DNN[15]	97.43	99.5	99.00	91.00	91.00
	SCDNN[16]	98.42	97.23	80.23	11.4	6.88
Proposed Method	LA-GRU	99.21	99.16	99.20	98.34	98.61

However, the number of training samples used by the above two models in [13, 15] is also much larger than our model. In particular, although the SCDNN model proposed in [16] is the only model in all comparison experiments that uses less training samples than the proposed LA-GRU, its overall detection performance is also the worst among all models.

Table 7 provides the comparison of detection rate between different IDMs for five-category classification. From Table 7, we can clearly see that the LA-GRU model not only performs very well in high-frequency attack detection, but, more importantly, in the low-frequency attack detection, the detection rate of 98.34% and 98.61% for R2L and U2R is obtained, respectively, which exceeds all the other IDMs and became the best experimental results.

To sum up, through the analysis of the comparison experimental results in Tables 6 and 7, we think that although, compared with other IDMs, the proposed LA-GRU does not achieve the best results in the overall detection performance, it still meets the expected requirements and has achieved good results. The advantage of LA-GRU is that its detection rate of low-frequency attack samples, for R2L and U2R, is much higher than all other IDMs. Moreover, the number of training samples used in our experiment is also less than the average level of other IDMs, which means that LA-GRU can reach the optimal state faster and reduce the consumption of computing resources. Therefore, we believe that the LA-GRU model has achieved the best comprehensive detection performance among the existing IDMs.

6. Conclusion and Future Work

In this paper, we propose a new combined IDM called LA-GRU to solve the existing problems in the IDMs constructed by utilizing machine learning theory. First, based on SMOTE algorithm, a new local adaptive SMOTE algorithm is proposed to solve the problem of poor detection rate of low-frequency attack traffic. Then, the GRU neural network is used as the classifier to complete the anomaly traffic detection while mining the existing temporal relationship between input samples. The experimental results of simulation verification on NSL-KDD dataset show that, on the one hand, compared with SMOTE algorithm, LA-SMOTE algorithm adjusts the imbalanced fitting trend of machine learning algorithms for high-frequency and low-frequency traffic and improves the detection rate of low-frequency attacks by generating fewer new samples, thereby reducing resource consumption and speeding up the classification process of the classifier. On the other hand, compared with other mainstream and state-of-the-art IDMs, the use of the GRU makes the IDM achieve outstanding and satisfactory results in terms of both overall detection performance and multiobjective detection because the GRU has utilized the temporal relationship existing within the traffic.

However, there are still many deficiencies in our work. In the future plans, the first intention is to verify the validity of the proposed model ulteriorly by using new intrusion detection dataset or real-world network traffic. Second, we will use new deep learning methods to achieve feature compression of input samples rather than intrusion detection, making IDMs close to meeting real-time requirements.

Data Availability

The discrete datasets including numeric data and symbolic data used to support the findings of this study are openly available at https://github.com/defcom17/NSL_KDD.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Science Technology Major Project of China (2016ZX01012101), National Natural Science Foundation Project of China (61572520), and National Natural Science Foundation Innovation Group Project of China (61521003).

References

[1] W. Lee and S. J. Stolfo, "A framework for constructing features and models for intrusion detection systems," *ACM Transactions on Information and System Security*, vol. 3, no. 4, pp. 227–261, 2000.

[2] M. A. M. Hasan, M. Nasser, B. Pal, and S. Ahmad, "Support vector machine and random forest modeling for intrusion detection system (IDS)," *Journal of Intelligent Learning Systems & Applications*, vol. 6, no. 1, pp. 45–52, 2014.

[3] R. Singh, H. Kumar, and R. K. Singla, "An intrusion detection system using network traffic profiling and online sequential extreme learning machine," *Expert Systems with Applications*, vol. 42, no. 22, pp. 8609–8624, 2015.

[4] Y. Yuan, L. Huo, and D. Hogrefe, "Two layers multi-class detection method for network intrusion detection system," in *Proceedings of the 2017 IEEE Symposium on Computers and Communications, ISCC 2017*, pp. 767–772, Greece, July 2017.

[5] B. Krawczyk, "Learning from imbalanced data: Open challenges and future directions," *Progress in Artificial Intelligence*, vol. 5, no. 4, pp. 1–12, 2016.

[6] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: synthetic minority over-sampling technique," *Journal of Artificial Intelligence Research*, vol. 16, no. 1, pp. 321–357, 2002.

[7] D. P. Kingma and J. L. Ba, "Adam: A method for stochastic optimization," in *Proceedings of the 3rd International Conference on Learning Representations*, pp. 1–13, 2015.

[8] W. L. Al-Yaseen, Z. A. Othman, and M. Z. A. Nazri, "Multi-level hybrid support vector machine and extreme learning machine based on modified K-means for intrusion detection system," *Expert Systems with Applications*, vol. 67, pp. 296–303, 2017.

[9] M. Reza, S. Miri, and R. Javidan, "A hybrid data mining approach for intrusion detection on imbalanced NSL-KDD dataset," *International Journal of Advanced Computer Science and Applications*, vol. 7, no. 6, pp. 20–25, 2016.

[10] A. Akyol, M. Hacibeyoglu, and B. Karlik, "Design of multilevel hybrid classifier with variant feature sets for intrusion detection system," *IEICE Transaction on Information and Systems*, vol. E99D, no. 7, pp. 1810–1821, 2016.

[11] D. Papamartzivanos, F. Gómez Mármol, and G. Kambourakis, "Dendron: Genetic trees driven rule induction for network intrusion detection systems," *Future Generation Computer Systems*, vol. 79, no. 2, pp. 558–574, 2018.

[12] Y. Zhu, J. Liang, J. Chen, and Z. Ming, "An improved NSGA-III algorithm for feature selection used in intrusion detection," *Knowledge-Based Systems*, vol. 116, pp. 74–85, 2017.

[13] W. Wang, Y. Sheng, J. Wang et al., "HAST-IDS: Learning hierarchical spatial-temporal features using deep neural networks to improve intrusion detection," *IEEE Access*, vol. 6, pp. 1792–1806, 2017.

[14] N. Shone, T. N. Ngoc, V. D. Phai, and Q. Shi, "A deep learning approach to network intrusion detection," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 2, no. 1, pp. 41–50, 2018.

[15] A. A. Diro and N. Chilamkurti, "Distributed attack detection scheme using deep learning approach for Internet of Things," *Future Generation Computer Systems*, vol. 82, pp. 761–768, 2017.

[16] T. Ma, F. Wang, J. Cheng, Y. Yu, and X. Chen, "A hybrid spectral clustering and deep neural network ensemble algorithm for intrusion detection in sensor networks," *Sensors*, vol. 16, no. 10, pp. 1701–1724, 2016.

[17] G. Loukas, T. Vuong, R. Heartfield, G. Sakellari, Y. Yoon, and D. Gan, "Cloud-Based Cyber-Physical Intrusion Detection for Vehicles Using Deep Learning," *IEEE Access*, vol. 6, pp. 3491–3508, 2017.

[18] B. A. Pearlmutter, "Gradient calculations for dynamic recurrent neural networks: A survey," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 6, no. 5, pp. 1212–1228, 1995.

[19] A. Karpathy and L. Fei-Fei, "Deep visual-semantic alignments for generating image descriptions," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 4, pp. 664–676, 2014.

[20] H.-Y. Lee, B.-H. Tseng, T.-H. Wen, and Y. Tsao, "Personalizing Recurrent-Neural-Network-Based Language Model by Social Network," *IEEE/ACM Transactions on Audio Speech and Language Processing*, vol. 25, no. 3, pp. 519–530, 2017.

[21] Z. Tan, J. Su, B. Wang, Y. Chen, and X. Shi, "Lattice-to-sequence attentional Neural Machine Translation models," *Neurocomputing*, vol. 284, pp. 138–147, 2018.

[22] A. K. Parida, R. Bisoi, P. K. Dash, and S. Mishra, "Times series forecasting using Chebyshev functions based locally recurrent neuro-fuzzy information system," *International Journal of Computational Intelligence Systems*, vol. 10, no. 1, pp. 375–393, 2017.

[23] D. K. Bebarta, B. Biswal, and P. K. Dash, "Polynomial based functional link artificial recurrent neural network adaptive system for predicting indian stocks," *International Journal of Computational Intelligence Systems*, vol. 8, no. 6, pp. 1004–1016, 2015.

[24] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 5, no. 2, pp. 157–166, 2002.

[25] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[26] R. Kadari, Y. Zhang, W. Zhang, and T. Liu, "CCG supertagging with bidirectional long short-Term memory networks," *Natural Language Engineering*, vol. 24, no. 1, pp. 1–14, 2018.

[27] K. Chen and Q. Huo, "Training Deep Bidirectional LSTM acoustic model for LVCSR by a context-sensitive-chunk BPTT approach," *IEEE/ACM Transactions on Audio, Speech and Language Processing*, vol. 24, no. 7, pp. 1185–1193, 2016.

[28] B. Fan, L. Xie, S. Yang, L. Wang, and F. K. Soong, "A deep bidirectional LSTM approach for video-realistic talking head," *Multimedia Tools and Applications*, vol. 75, no. 9, pp. 5287–5309, 2016.

[29] K. Cho, B. Van Merriënboer, C. Gulcehre et al., "Learning phrase representations using RNN encoder-decoder for statistical machine translation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014*, pp. 1724–1734, Qatar, October 2014.

[30] M. Ravanelli, P. Brakel, M. Omologo, and Y. Bengio, "Light gated recurrent units for speech recognition," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 2, no. 2, pp. 92–102, 2018.

[31] P.-S. Kim, D.-G. Lee, and S.-W. Lee, "Discriminative context learning with gated recurrent unit for group activity recognition," *Pattern Recognition*, vol. 76, pp. 149–161, 2018.

[32] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," in *Proceedings of the 2nd IEEE Symposium on Computational Intelligence for Security and Defence Applications*, pp. 1–6, IEEE, July 2009.

[33] S. J. Stolfo, W. Fan, W. Lee, A. Prodromidis, and P. K. Chan, "Cost-based modeling for fraud and intrusion detection: Results from the JAM project," in *Proceedings of the DARPA Information Survivability Conference and Exposition, DISCEX 2000*, pp. 130–144, USA, January 2000.

[34] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *Journal of Machine Learning Research*, vol. 13, pp. 281–305, 2012.

[35] C. Khammassi and S. Krichen, "A GA-LR wrapper approach for feature selection in network intrusion detection," *Computers & Security*, vol. 70, pp. 255–277, 2017.