Computer Networks 177 (2020) 107315

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

An effective convolutional neural network based on SMOTE and Gaussian mixture model for intrusion detection in imbalanced dataset

Hongpo Zhang [a,b,*], Lulu Huang [a,b], Chase Q. Wu [c], Zhanbo Li [a,b]

[a] School of Information Engineering, Zhengzhou University, Zhengzhou, Henan 450001, China
[b] Cooperative Innovation Center of Internet Healthcare, Zhengzhou University, Zhengzhou, Henan 450001, China
[c] Department of Computer Science, New Jersey Institute of Technology, Newark, NJ 07102-1982, USA

ARTICLE INFO

Keywords:
Network intrusion detection
Deep learning
Class imbalance
Gaussian mixture model
Convolutional neural network

ABSTRACT

Network Intrusion Detection System (NIDS) is a key security device in modern networks to detect malicious activities. However, the problem of imbalanced class associated with intrusion detection dataset limits the classifier's performance for minority classes. To improve the detection rate of minority classes while ensuring efficiency, we propose a novel class imbalance processing technology for large-scale dataset, referred to as SGM, which combines Synthetic Minority Over-Sampling Technique (SMOTE) and under-sampling for clustering based on Gaussian Mixture Model (GMM). We then design a flow-based intrusion detection model, SGM-CNN, which integrates imbalanced class processing with convolutional neural network, and investigate the impact of different numbers of convolution kernels and different learning rates on model performance. The advantages of the proposed model are verified using the UNSW-NB15 and CICIDS2017 datasets. The experimental results show that i) for binary classification and multiclass classification on the UNSW-NB15 dataset, SGM-CNN achieves a detection rate of 99.74% and 96.54%, respectively; ii) for 15-class classification on the CICIDS2017 dataset, it achieves a detection rate of 99.85%. We compare five imbalanced processing methods and two classification algorithms, and conclude that SGM-CNN provides an effective solution to imbalanced intrusion detection and outperforms the state-of-the-art intrusion detection methods.

1. Introduction

According to the prediction of Cisco [1], the number of networked smart devices will double from 2017 to 2022, resulting in a rapid increase in data traffic by five times. As the network scope and scale continue to expand, the threat of network intrusion has become far more serious than ever before. In this situation, the Intrusion Detection System (IDS), which is widely used to sniff and detect different types of network intrusion, is required to evolve with time to meet the growing demand of cybersecurity assurance. IDSs can be divided into Host-based IDS (HIDS) and Network-based IDS (NIDS) according to the data source-based methods [2]. HIDS mainly monitors system logs, system call traces, etc. [4,3]. Modern NIDS falls into two main categories, i.e., rule-based misuse detection and statistics-based anomaly detection [5]. The former works with a database that stores the attributes of all known attacks and classifies network traffic as "attack" if the extracted attributes match those in the database. This type of NIDS can identify known attacks efficiently and accurately but suffers from the incompetence to detect new ones, which is a critical issue in modern networks where 0-day attacks frequently occur. Therefore, the anomaly detection-based NIDS has increasingly attracted attention in recent years. Its basic principle is to detect anomaly in the features or distributions of network flows, which facilitates and enables the identification of unknown attacks.

Machine learning techniques have been extensively used in anomaly detection-based NIDS. Different machine learning models including support vector machine (SVM) [6–8], random forest (RF) [9–12], decision tree (DT) [13–15] have been employed to distinguish between normal and abnormal network activities. However, with the diversification of attack categories and the surge of network traffic, shallow learning such as traditional machine learning techniques is no longer suitable to meet the requirements of large-scale NIDS [2].

In recent years, with the advantage of fully automatic feature engineering, deep learning has been the focus of research. Convolutional neural networks (CNN) [16–19], multilayer perceptron (MLP) [20] and recurrent neural networks (RNN) [21,22] have also been employed in NIDS. Studies in [21] show that these deep learning-based NIDS can achieve better performance when working with big data.

However, these NIDS schemes still have some shortcomings. Firstly, most of them do not provide the exact attack type as the classification

* Corresponding author at: Cooperative Innovation Center of Internet Healthcare, Zhengzhou University, Zhengzhou, Henan 450001, China.
E-mail addresses: zhp@zzu.edu.cn (H. Zhang), llhuang@ha.edu.cn (L. Huang), chase.wu@njit.edu (C.Q. Wu), lzb@zzu.edu.cn (Z. Li).

https://doi.org/10.1016/j.comnet.2020.107315
Received 25 January 2020; Received in revised form 21 April 2020; Accepted 13 May 2020
Available online 16 May 2020

result. Since different attacks require different defence mechanisms in practical systems, a detection result of "normal" or "abnormal" is insufficient. Secondly, most of these schemes were tested with KDD99 [23] or NSL KDD datasets that were collected about 20 years ago. New attacks are emerging almost constantly and using outdated traffic data does not reflect the actual performance of NIDS when applied to modern networks. They only use a small part of the dataset for experiments, without considering the performance of the system in the big data environment. Lastly, they do not address the problem of class imbalance and its impact on the classification performance, which significantly reduces the detection rate, especially for minority classes. The focus of this work is to solve the problem of class imbalance in modern large-scale intrusion detection systems.

In recent years, the class imbalance problem has attracted a great deal of attention. The current solutions are mainly divided into four types: data-level methods, algorithm-level methods, cost-sensitive methods, and integration methods [24]. Among them, data-level methods are widely employed due to their advantages of being independent of algorithms and using simple operations. Its main idea is resampling, including over-sampling and under-sampling. The simplest resampling methods are ROS and RUS. Other representative methods are EasyEnsemble [25], BalanceCascade [25], SMOTE [26] and ADASYN [27], etc. Studies in [28] have shown that oversampling is the best way to handle imbalance in deep learning, and it does not cause overfitting of the model. However, the increase in the data volume would increase the time cost.

To solve the above challenges, we present a flow-based network intrusion detection model, SGM-CNN, which integrates class imbalance processing with deep learning, and evaluate the model on the UNSW-NB15 and CICIDS2017 datasets in binary-class and multi-class scenarios, respectively. We have also published a GitHub repository for reproducible demonstrations of intrusion detection with SGM-CNN [29]. Our main contributions are as follows:

(1) We present a novel class imbalance processing technology, SGM, for large-scale dataset. We first use SMOTE to oversample the samples of the minority class, then use a GMM to perform cluster-based under-sampling on the majority class, and ultimately balance various classes of data. This method can not only avoid the excessive time and space cost caused by oversampling, but also prevent random under-sampling from losing important samples. It significantly increases the detection rate of minority classes.
(2) We propose a flow-based network intrusion detection model, SGM-CNN, which fuses class imbalance processing technology SGM and 1D-CNN, to detect highly imbalanced network traffic with high accuracy. Also, we study the impact of different numbers of convolution kernels and different learning rates on model performance.
(3) We evaluate the performance of five imbalanced class processing methods and two machine learning classification algorithms (RF and MLP). Experimental results show that our model is superior to the state-of-the-art methods, and provides an effective method for large-scale imbalanced intrusion detection.

The rest of the paper is organized as follows. Section 2 discusses the related work of deep learning and class imbalance processing technology in NIDS. A description of the SGM-CNN model and dataset is provided in Section 3. Experimental results are presented in Section 4 and discussed in Section 5. The work is concluded in Section 6.

2. Related work

Machine learning techniques have been used for anomaly detection-based NIDS since early2000*s* [7,9,12,14,30,31], because of the ability to mine hidden information on the differences between normal and malicious behaviours. Belouch et al. [32] proposed a two-stage NIDS based on the RepTree classification algorithm. The first phase identifies network traffic as normal or attack, and the second phase determines the specific attack type. On the UNSW-NB15 dataset, the detection accuracies for the first and second stages are 88.95% and 81.28%, respectively. Moustafa et al. [31] studied a real-time collaborative network forensic scheme (RCNF) using chi-square feature selection method and correntropy-variation. Koroniotis et al. [13] designed a system for forensics of IoT botnet activities based on machine learning. After using the information gain method to select ten vital features on the UNSW-NB15 dataset, it used four machine learning techniques, i.e., association rule mining (ARM), näive Bayes (NB), artificial neural network (ANN) and decision tree to classify the dataset. Simulation results show that the best choice to distinguish botnets from normal network traffic is the C4.5, which has an accuracy of 93.23%. Binbusayyis et al. [12] used the RF algorithm to verify their method based on integrated feature selection on four datasets including UNSW-NB15 and CICIDS2017 datasets. With the increase of network traffic, NIDS based on machine learning start experiencing some limitations. For example, the system's performance relies too much on feature selection methods, and it is challenging to achieve an overall accuracy beyond 95%.

More recently, with the advance in deep learning, researchers start to employ deep learning technologies in NIDS to cope with big data. Kwon et al. [33] proposed a fully connected neural network architecture. The simulation results on the NSL-KDD dataset reached an accuracy of 84.2%. Baig et al. [34] proposed a multi-class intrusion detection method based on cascaded artificial neural network. The results show that the method can effectively identify intrusion detection with an accuracy of 86.40% on the UNSW-NB15 dataset. Yin et al. [21] applied RNN to intrusion detection, and experimental results confirm that deep learning-based NIDS performs better when processing big data. Zhang et al. [20] used a denoising autoencoder (DAE) with a weighted loss function for feature selection and then used MLP to identify normal and abnormal data, with an accuracy of 98.80% on the UNSW-NB15 dataset. Osama Faker et al. [35] combined big data and deep learning technologies to propose an intrusion detection system based on K-means homogeneity metric feature selection, and used Deep Feed-Forward Neural Network (DNN), RF, and Gradient Boosting Tree (GBT) for binary classification and multi-classification. The experimental results on the UNSW-NB15 and CICIDS2017 datasets show that the system has high accuracy on three classification algorithms. Xiao et al. [17] reduced the dimensionality of data through principal component analysis and autoencoder, and used an improved Lenet-5 CNN to perform intrusion detection on KDD CUP99. The results show that CNN can perform better in intrusion detection, and more network traffic is more conducive to feature learning.

Class imbalance refers to the imbalance among various types of data, namely, there are many data in one class while very little in the other [36]. It affects the performance of classification algorithms with an assumption that the class distribution is balanced, hence degrading the detection rate, especially for minority classes. There is a serious class imbalance problem in intrusion detection dataset, which remains largely unexplored. Data-level processing is the most widely used method. David et al. [37] proposed Cluster-SMOTE that fuses oversampling and under-sampling techniques. Abdulhammed et al. [38] adopted the Uniform Distributed Based Balancing (UDBB) approach to construct a machine learning-based NIDS. The simulation results on the CICIDS2017 dataset show that UDBB can effectively alleviate the problem of imbalanced data distribution, where RF has the best performance with a multi-class accuracy of 99.60%. Some improvements have been made to the algorithm to solve the class imbalance problem. Zhu et al. [39] studied an improved NSGS-III feature selection algorithm, which uses probability-based bias selection to overcome class imbalance problems. The overall detection rate on the NSL-KDD dataset is 99.21%, and the feature selection algorithm has low computational complexity. Zhang et al. [18] proposed a parallel cross convolution neural network model (PCCN) for the imbalance problem in IDS. The experimental research on the CICIDS2017 dataset shows that PCCN is compared with other commonly used CNN and LSTM models show its superiority. Yang et al. [40]

combined an improved conditional variational autoencoder with a deep neural network and proposed an NIDS for rare attacks, referred to as ICVAE-DNN. ICVAE can explore the relationship between classes and data features to generate new attack samples to balance training data. It achieves an accuracy of 89.08% on the UNSW-NB15 dataset and 85.97% on the NSL KDDTest + dataset. Ullah et al. [41] used SMOTE and ENN to process imbalanced data in a 2-level hybrid NIDS. Gozde karatas et al. [42] used SMOTE to oversample the samples of minority classes, and then used six classic machine learning algorithms (i.e., DT, RF, K Nearest Neighbor, Adaboost, Gradient Boosting, and Linear Discriminant Analysis) for classification on the CSE-CIC-IDS2018 dataset. Experimental results show that the established model outperforms existing ones.

To solve the class imbalance problem in large-scale NIDS, we propose a data-level processing method combining SMOTE oversampling and clustering under-sampling based on Gaussian mixture model. Inspired by the existing research, we design a 1D-CNN for network traffic identification. We use the UNSW-NB15 dataset and CICIDS2017 datasets to test the model in binary classification and multiclass classification, and compare five imbalanced processing methods and two classification algorithms (RF and MLP).

3. Proposed solution

The architecture of the proposed NIDS, SGM-CNN, which combines class imbalance processing technology and CNN, is shown in Fig. 1. The system consists of three main modules: a data preprocessing module, an imbalance processing module and a classification decision module. The data preprocessing module is responsible for performing operations such as one-hot encoding, feature reduction, data standardization, and feature selection on the original data to make the data more conducive to the prediction of the model. The imbalance processing module mainly resamples the training dataset to alleviate the bias caused by the imbalance in the original dataset to the experimental results. We proposed a new method, SGM, which combines oversampling and under-sampling to achieve a completely balanced training dataset. SGM oversampling identifies minority classes using SMOTE and clustering under-sampling identifies majority classes based on a GMM. Finally, in the classification decision module, we design a six-layer 1D-CNN model. To test the performance in a modern network environment, we conduct binary-class and multi-class classifications on the UNSW-NB15 dataset and CICIDS2017 datasets, as detailed below.

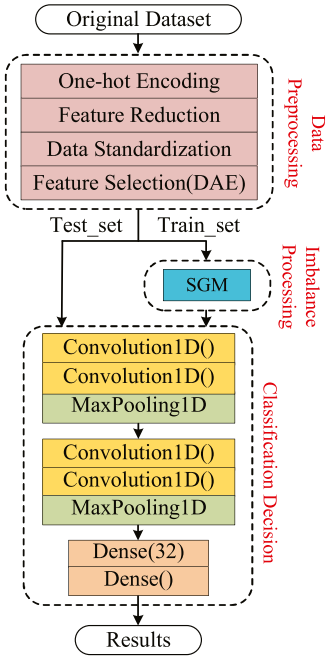

Fig. 1. A schematic diagram of the SGM-CNN NIDS model.

3.1. Dataset description

UNSW-NB15 dataset [43] was collected and distributed by the cybersecurity research group at the Australian Centre for Cyber Security (ACCS). This dataset contains a total of 2.54 million network traffic samples, involving nine attack categories. Each sample has 49 features, two of which are class label features. This dataset has serious class imbalances, in which normal traffic accounts for 87.35% of the entire dataset, and all attack traffic accounts for only 12.65% of the dataset. We use all samples of the UNSW-NB15 dataset in the experiment, and divide the dataset for training, validation, and testing at a ratio of 7:1:2. The proportion of each category in each subset is the same. The detailed data distribution of each class is shown in Table 1.

The CICIDS2017 dataset [44] was collected and compiled by the Canadian Cyber Security Institute with the help of the B-Profile system [45] at the end of 2017. The dataset contains 2,830,473 network traffic samples, including one benign class and 14 attack categories, with benign traffic accounting for 80.30% and attack traffic accounting for 19.70%. The attack categories cover the most common types of attacks, such as DoS, DDoS, Botnet, PortScan, Web Attacks, etc. The dataset extracts 84 features from the generated network traffic, of which the last column is the multiclass label [38]. In addition, compared with the publicly available datasets from 1998 to 2016, this dataset fully meets the 11 criteria for performance evaluation [46]. The CICIDS2017 dataset is split in the same way as the UNSW-NB15 dataset. The detailed data distribution of each class is provided in Table 2.

3.2. Data preprocessing

The data preprocessing module contains four steps: one-hot encoding, feature reduction, data standardization, and feature selection (DAE). First of all, one-hot encoding, which maintains the disordered characteristics of nominal features, is used to quantify three nominal features in the UNSW-NB15 dataset that cannot be processed by machine learning algorithms. The three features “proto”, “state” and “service” contain 135, 16 and 13 different values, respectively. After applying one-hot encoding, the feature dimension of the UNSW-NB15 dataset changes from 47 to 208. One-hot encoding is also used in class label numeralization of two datasets. Note that the “Flow Bytes” and “Flow Packets” features of the CICIDS2017 dataset contain “infinity” values. We replace such “infinity” values with the maximum value of their column plus one, and fill missing values (NaN) with zero to avoid errors during model training.

Then, we drop redundant and meaningless features in the dataset. Particularly, we drop six features in the UNSW-NB15 dataset according to Zhang et al. [20], which are “srcip”, “sport”, “dsport”, “dstip”, “ltime”, and “stime”. After that, the feature dimension of the UNSW-NB15 dataset is reduced to 202. For the CICIDS2017 dataset, we drop six features, namely, “Flow ID”, “Source IP”, “Source Port”, “Destination IP”, “Protocol”, and “Time stamp” according to Abdulhammed et al. [38], hence reducing its feature dimension to 77.

Next, we standardize all remaining features according to Eq. (1), and normalize them to a Gaussian distribution with a mean of 0 and a variance of 1:

$$x' = \frac{x - \mu}{\delta}, \tag{1}$$

where x is the original feature, x' is the normalized feature, and μ and δ are the mean and standard deviation of the feature, respectively. The normalized data maintains the same linear relationship as the original data [17]. Data normalization helps improve the convergence speed and accuracy of the model.

The final step is to perform feature selection. Since the CICIDS2017 dataset has total 77 features at this time, much less than the UNSW-NB15

Table 1
The number of samples in each class of UNSW-NB15 dataset.

Class	Trainset-size	Testset-size	Validationset-size	Total
Normal	1,553,132	443,755	221,874	2,218,761
Generic	150,836	43,097	21,548	215,481
Exploits	31,167	8,906	4,452	44,525
Fuzzers	16,972	4,849	2,425	24,246
DoS	11,449	3,269	1,635	16,353
Reconnaissance	9,791	2,797	1,399	13,987
Analysis	1,874	535	268	2,677
Backdoors	1,630	466	233	2,329
Shellcode	1,057	303	151	1,511
Worms	122	35	17	174
Total	1,778,030	508,012	254,001	2,540,044

Table 2
The number of samples in each class of the CICIDS2017 dataset.

Class	Trainset-size	Testset-size	Validationset-size	Total
BENIGN	1,591,167	454,620	227,310	2,273,097
DoS Hulk	161,751	46,215	23,107	231,073
PortScan	111,251	31,786	15,893	158,930
DDoS	89,618	25,606	12,803	128,027
DoS GoldenEye	7,205	2,059	1,029	10,293
FTP-Patator	5,516	1,588	794	7,898
SSH-Patator	4,128	1,179	590	5,897
DoS slowloris	4,057	1,159	580	5,796
DoS Slowhttptest	3,849	1,100	550	5,499
Bot	1,376	393	197	1,966
Web Attack Brute Force	1,055	301	151	1,507
Web Attack XSS	457	130	65	652
Infiltration	26	7	3	36
Web Attack Sql Injection	15	4	2	21
Heartbleed	8	2	1	11
Total	1,981,519	566,149	283,075	2,830,743

Table 3
The selected features and corresponding weights of the UNSW-NB15 dataset.

Feature	Dtcpb	Stcpb	Service_-	Dload	Dmeansz	Service_dns
Weight	21.338	20.295	15.901	14.440	13.455	12.939
Feature	Smeansz	Sload	Trans_depth	Sttl	Service_ftp-data	ct_ftp
Weight	12.197	11.979	10.821	9.827	9.787	9.723

dataset, and the training time for this amount of data is within an acceptable range, we do not perform feature selection on the CICIDS2017 dataset. For the UNSW-NB15 dataset, we use the DAE in [20] to select 12 features with the highest weight out of 202 features for subsequent models. Table 3 displays the 12 selected features and their weights, three of which belong to "Service".

3.3. Class imbalance processing

The number of abnormal samples in the intrusion detection dataset is inherently small. For example, the "Worms" in the UNSW-NB15 dataset has only 174 samples, accounting for 0.00685% of the entire dataset. So it is insuitable to use the under-sampling alone. However, using over-sampling only would introduce too much redundant data and increase space and time costs. In this paper, we propose a new method SGM that combines SMOTE and GMM-based clustering under-sampling to resample all categories of samples to a uniform number of instance $I_{Resample}$ [38]. The definition of $I_{Resample}$ is provided in Eq. (2):

$$I_{Resample} = int\left(\frac{N}{C}\right), \tag{2}$$

where N is the total number of samples in the training set, and C is the number of classes.

SGM uses SMOTE to oversample classes with less than $I_{Resample}$ to $I_{Resample}$. SMOTE is a classic oversampling method proposed by Chawla et al. in 2002 [26]. SMOTE increases the quantity of minority class samples by "synthesizing" minority class samples. The so-called "synthesis" is to generate a sample that does not exist in the original dataset, instead of simply copying samples such as ROS, to avoid overfitting in the process of building a classification model. SMOTE works by taking a random point between the minority class sample and its k-nearest neighbours as a new synthetic sample.

For classes with more samples than $I_{Resample}$, we use a GMM-based clustering method to under-sample the majority class samples to $I_{Resample}$. GMM is a parameterized probability distribution model that represents a linear combination of multiple Gaussian distribution functions. Assuming that all samples come from multiple Gaussian distributions with different parameters, and samples that belong to the same distribution are divided into the same cluster, GMM returns the probability that sample x belongs to different clusters according to Eq. (3):

$$P(x|\theta) = \sum_{k=1}^{K} \alpha_k \Phi(x|\theta_k), \tag{3}$$

where α_k is a coefficient, $\alpha_k \geq 0$, $\sum_{k=1}^{K} \alpha_k = 1$, $\theta_k = (\mu_k, \sigma_k^2)$; $\Phi(x|\theta_k)$ is the Gaussian distribution density, called the kth sub-model, and is given by:

$$\Phi(X|\theta_k) = \frac{1}{\sqrt{2\pi}\sigma_k} \exp\left(-\frac{(x-\mu_k)^2}{2\sigma_k^2}\right). \tag{4}$$

Class imbalance processing is only performed on the training set $D = \{D_i, i = 1, 2, \cdots, C\}$ to ensure the authenticity of the model test results. The SGM method first calculates $I_{Resample}$. For each class of data D_i in the training set, if the number of samples in D_i is less than $I_{Resample}$, then we use SMOTE to oversample D_i to balance with $I_{Resample}$. If the number of samples in D_i is more than $I_{Resample}$, we use GMM to cluster D_i into C clusters. Then we randomly select $\frac{I_{Resample}}{C}$ samples from each cluster and merge them into D_i', which is a subset of D_i. At this time, the number of samples in D_i' is in balance with $I_{Resample}$, so that we obtain a balanced training set D'. SGM not only avoids the excessive time and space cost caused by using oversampling alone, but also prevents random undersampling from losing important samples. It significantly improves the detection rate of minority classes. The pseudocode of SGM is provided in Algorithm 1.

Algorithm 1 SGM.

Input:
Training set $D = \{D_i, i = 1, 2, \cdots, C\}$;
C = the total number of classes;
$|D| = N$; #the total number of samples
Output:
a balanced training set D';

$I_{Resample} = int(\frac{N}{C})$
for $i \leftarrow 1$ to C **do**
  if $|D_i| < I_{Resample}$ **then**
    $D_i' = SMOTE(D_i, I_{Resample})$ #Use SMOTE to oversample D_i, so that $|D_i'| = I_{Resample}$
  end if
  if $|D_i| > I_{Resample}$ **then**
    $G_k = GMM(D_i, C)$ #Use GMM to cluster D_i into C clusters, $k = 1, 2, \cdots, C$
    for $k \leftarrow 1$ to C **do**
      $G_k' = Resample(G_k, \frac{I_{Resample}}{C})$ #Randomly select $\frac{I_{Resample}}{C}$ samples from G_k
    end for
    $D_i' = Concatenate(G_k')$
  end if
  $D' = Concatenate(D_i')$
end for
return D'

3.4. Convolutional neural network

Convolutional Neural Network (CNN) is one of the representative algorithms for deep learning. It is a type of deep feedforward neural network with convolution calculations. CNN has the capability of representation learning, and can classify input information according to its hierarchical structure.

CNN includes the convolutional layer, the pooling layer and the dense layer. The combination of multiple convolutional layers makes CNN feature extraction. Convolution operations can not only strengthen the original data features but also relatively reduce the noise. We use 1D convolution in this work. The convolution of a signal sequence x_1, x_2, $\cdots$ at time t can be expressed by:

$$y_t = f\left(\sum_{k=1}^{m}(w_k \bigotimes x_{t-k+1}) + b_t\right), \tag{5}$$

where y_t is the output feature at time t, $f(x)$ is a nonlinear activation function, $w_k(k = 1, 2, \cdots, m)$ is a filter (or convolution kernel) of length m, and b_t is the offset vector at time t. The convolution kernel and the output feature of the previous layer or the original feature perform a convolution operation to obtain the current feature. The nonlinear activation process removes redundant information and retains the mapping information of the original data features, and also enhances the ability for nonlinear expression of the network model. The main difference between the dense layer and the convolutional layer is that the dense layer learns the global pattern from the input feature space, while the convolutional layer learns the local pattern.

Since there are still many features after the convolution layer, the network is very complicated. At this time, the pooling layer needs to under-sample the features to reduce the network complexity and avoid overfitting. The most frequently used pooling methods are Max-Pooling and Average Pooling [4]. In this work, we use Max-Pooling, as shown in Eq. (6), taking the maximum value of all neurons in a region $\mathfrak{R}$ as the output of that region:

$$Y_i = \underset{i \in \mathfrak{R}}{max}\ x_i. \tag{6}$$

We design a six-layer 1D-CNN suitable for NIDS. The network structure is shown in the Classification Decision module of Fig. 1. The data is simply mapped into two-dimensional information as input to the network. The first four layers are convolutional layers, where every two convolutional layers are followed by a Max-Pooling layer and a Dropout layer with a parameter of 0.2. Multiple stacked convolutional layers have fewer parameters and more nonlinear transformations than a large-size convolutional layer, and can provide stronger feature learning capabilities [47]. The last two layers are dense layers, which integrate the previously learned local features into global features, of which the fifth layer has 32 neural units. The sixth layer is the output layer, which is mainly used for classification prediction. The detailed parameter selection process is described in the next section.

4. Experimental analysis

We implement the proposed NIDS model in Python and conduct experiments on the UNSW-NB15 and CICIDS2017 datasets on a PC with an Ubuntu 64-bit operating system to evaluate its efficacy for detecting modern attacks. The specific system environment parameters are provided in Table 4. When class imbalance processing is performed on the training set, the data of each category is resampled to $I_{Resample}$. Also, during the model training, the epoch is set to be 100 and batch_size is set to be 256.

4.1. Evaluation metrics

In the experiments, in addition to Accuracy (ACC), false alarm rate (FAR), and detection rate (DR), we also consider three other performance indicators commonly used in class imbalance systems, namely, Recall, Precision and F_1 score.

For each attack type, we consider the samples as positive ones and the others as negative ones. ACC is defined as the percentage of correctly classified ones among all samples. DR is the ratio of positive samples that are correctly detected, reflecting the ability of NIDS to identify specific types of attacks. FAR is defined as the ratio of negative samples that are wrongly judged as positive ones. Recall, which is essentially DR, refers to the ratio of the number of positive class samples that are correctly predicted to the total number of positive class samples. Precision refers to how many of the samples that the model judges to be positive are truly positive samples. F_1 score is the harmonic average of Precision and

Table 4
Experimental environment.

Project	Environment/Version
Operating system	Ubuntu
Framework	Kreas 2.2.4
Backend	Engine Tensorflow
CPU	i9-9900KF
GPU	RTX2080Ti
Memory	64GB

Table 5
Comparison of ACC, DR and F_1 score in CNN model with different numbers of convolution kernels on the UNSW-NB15 dataset(%).

Convolution Kernel Number	16-16-32-32	16-16-64-64	32-32-64-64	32-32-128-128	64-64-128-128
ACC	99.04	98.95	**99.08**	99.10	99.10
DR	94.05	95.99	**96.45**	96.07	95.77
F_1 score	96.10	95.86	**96.36**	96.41	96.43

Table 6
Comparison of ACC, DR and F_1 score at different learning rates of the CNN model on the UNSW-NB15 dataset(%).

Learning-Rate	0.5	0.1	0.05	0.03	0.01	0.008	0.005	0.002	0.0005
ACC	99.10	99.09	99.07	99.09	99.04	**99.08**	99.08	99.08	99.08
DR	96.13	95.96	94.57	96.38	96.68	**96.45**	96.11	96.31	94.67
F_1 score	96.43	96.39	96.25	96.39	96.22	**96.36**	96.35	96.38	96.28

Table 7
CNN model structure and parameter settings.

Layer	Type	Kernel/Pool_size	Strides	Active Function	Output
L1	Input	-	-	-	F*1
L2	Conv1D	3	1	Relu	F*32
L3	Conv1D	3	1	Relu	F*32
L4	MaxPooling	2	2	Relu+Dropout(0.2)	(F/2)*32
L5	Conv1D	3	1	Relu	(F/2)*64
L6	Conv1D	3	1	Relu	(F/2)*64
L7	MaxPooling	2	2	Relu+Dropout(0.2)	(F/4)*64
L8	Dense	-	-	Relu	32
L9	Dense	-	-	Softmax	*C*

Recall. In multi-classification, to more reasonably evaluate the detection performance of the model on the imbalanced dataset, each index is calculated using a weighted average method according to the number of samples in each category. The formula of each indicator is shown in Eqs. (7)–(11).

$$ACC = \frac{TP + TN}{TP + TN + FP + FN}, \tag{7}$$

$$DR = \frac{TP}{TP + FN}, \tag{8}$$

$$FAR = \frac{FP}{FP + TN}, \tag{9}$$

$$Precision = \frac{TP}{TP + FP}, \tag{10}$$

$$F_1 score = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall}, \tag{11}$$

where TP/FP and TN/FN are the numbers of samples, which are correctly and wrongly predicted to be positive and negative ones, respectively.

4.2. Hyper-parameter optimization

The number of convolution kernels in the CNN's convolution layer directly affects the classification result of the model. To obtain a better CNN model, we perform a set of binary classification experiments on the UNSW-NB15 dataset after feature selection. In this set of experiments, we only vary the number of convolution kernels. The number of neural units in the first dense layer is 32, and there are two neural units in the output layer. The optimization algorithm uses "nadam", the learning rate is 0.008, and the loss function uses "categorical_crossentropy". Table 5 shows the comparison of ACC, DR and F_1 score in CNN model with different numbers of convolution kernels on the UNSW-NB15 dataset. We observe that the difference in ACC is insignificant, but the difference in DR is obvious. We achieve the best performance when the number of convolution kernels in four convolution layers is 32, 32, 64, and 64, respectively, resulting in ACC of 99.08%, DR of 96.45%, and F_1 score of 96.36%.

"Nadam" is the best optimization algorithm at present [48], but different learning rates also affect the results. We also conduct comparative experiments on the learning rates of different values. We keep the number of convolution kernels (32-32-64-64) and other parameters unchanged, and set the values of the learning rate to be 0.5, 0.1, 0.05, 0.03, 0.01, 0.008, 0.005, 0.002, and 0.0005, respectively. Table 6 shows the comparison of ACC, DR and F_1 score results of CNN model at different learning rates on the UNSW-NB15 dataset. These results show that ACC does not change much at different learning rates. The learning rate of 0.008 leads to a better performance of the model and hence is used in the final model.

In the above two experiments, we explain the hyper-parameter selection of the CNN model. Table 7 shows the parameter settings of each layer in the final CNN model. The number of convolution kernels in the four 1D convolution layers is 32-32-64-64, and the Max-Pooling layer under-samples the parameters of the convolution layer by two times. Except for the output layer using Softmax as the activation function, the other layers use Relu. Also, there is a Dropout layer with the parameter of 0.2 behind each pooling layer to prevent overfitting. Note that the parameter F denotes the feature dimension of the input data. F is 12 on the UNSW-NB15 dataset and 77 on the CICIDS2017 dataset. The parameter C denotes the number of classes: C is 2 in binary classification, and 10 and 15 in multi-classification for UNSW-NB15 and CICIDS2017, respectively.

4.3. Binary classification

In the binary classification experiment, to demonstrate the effectiveness of SGM, we compare five different class imbalanced processing techniques, including three famous oversampling technologies, namely, ROS, SMOTE, and ADASYN, and two others by replacing GMM in Algorithm 1 with RUS and K-means for under-sampling, respectively. Also, to evaluate the effectiveness of the proposed CNN model, we compare two machine learning classification algorithms, namely, RF and MLP. RF is the most representative method in integrated technology [49]. RandomizedSearchCV [50] is used for parameter optimization to determine the RF optimal parameters in binary classification. Finally,

Table 8
Performance evaluation of the proposed model in binary classification on the UNSW-NB15 dataset (%).

Model	Datasets	ACC	DR	FAR	Precision	F_1 score	Train-Times(s)	Test-Time(s)
RF	ROS	98.67	99.99	1.52	90.49	95.01	15.64	0.52
	SMOTE	98.68	99.99	1.52	90.53	95.02	16.49	0.52
	ADASYN	98.66	99.87	1.52	90.49	94.95	18.13	0.53
	RUS+SMOTE	98.66	99.87	1.51	90.52	94.97	9.51	0.23
	K-means+SMOTE	98.68	99.99	1.51	90.53	95.03	9.03	0.23
	SGM	**98.68**	**99.99**	**1.51**	**90.54**	**95.03**	9.28	0.25
MLP	ROS	98.66	**99.86**	1.51	90.52	94.96	34.43	6.29
	SMOTE	98.67	99.80	1.50	90.61	94.98	48.62	6.26
	ADASYN	98.65	99.85	1.52	90.49	94.94	47.81	7.19
	RUS+SMOTE	98.65	99.82	1.52	90.49	94.93	7.77	2.71
	K-means+SMOTE	98.71	99.84	1.46	90.84	95.13	7.61	2.67
	SGM	**98.74**	99.82	**1.42**	**91.08**	**95.25**	7.90	2.70
CNN	ROS	98.68	99.99	1.52	90.53	95.02	71.62	18.12
	SMOTE	98.78	99.93	1.39	91.22	95.38	82.75	55.49
	ADASYN	98.70	**99.99**	1.48	90.72	95.13	73.06	17.95
	RUS+SMOTE	98.78	99.91	1.39	91.25	95.39	47.53	8.27
	K-means+SMOTE	98.76	99.97	1.41	91.10	95.33	47.13	8.12
	SGM	**98.82**	99.74	**1.31**	**91.66**	**95.53**	48.56	8.39

Table 9
Performance evaluation of the proposed model in multi-class classification on the UNSW-NB15 dataset using CNN, highlighting the DR of each class (%).

Class	12	ROS	SMOTE	RUS+SMOTE	K-means+SMOTE	SGM
Normal	1.00	0.98	0.98	0.98	0.98	0.98
Generic	0.98	0.97	0.97	0.97	0.97	0.97
Exploits	0.85	0.46	0.47	0.46	0.44	0.45
Fuzzers	0.38	0.72	0.69	0.65	0.66	0.67
DoS	0.02	0.37	0.42	0.37	0.31	0.39
Reconnaissance	0.75	0.83	0.81	0.81	0.82	0.82
Analysis	0.00	0.36	0.20	0.22	0.38	0.27
Backdoors	0.05	0.44	0.56	0.60	0.49	0.51
Shellcode	0.02	0.76	0.87	0.88	0.87	0.88
Worms	0.00	0.83	0.77	0.83	0.83	0.83
DR	97.73	96.48	96.54	96.25	96.17	**96.54**
ACC	97.73	96.48	96.54	96.25	96.17	**96.54**
Precision	97.48	98.25	98.25	98.27	98.27	**98.30**
F_1 score	97.35	97.20	97.25	97.09	97.05	**97.26**
Train-Time(s)	47.54	427.99	387.39	48.21	47.58	47.22
Test-Time(s)	17.85	67.24	52.63	8.30	8.28	8.26

we set n_estimators to be 25, max_depth of the decision tree to be 20, and use “sqrt” to determine max-features. MLP is a typical neural network with multiple hidden layers. In MLP, we set three hidden layers with 128, 64, and 32 neural units, respectively. The parameter settings of the output layer are the same as CNN.

Table 8 shows the binary classification results on the UNSW-NB15 dataset, where the bold part is the optimal result on a certain index. We observe in Table 8 that the best classification results are obtained by using the SGM-CNN model. ACC, DR, FAR, Precision, and F_1 score reach 98.82%, 99.74%, 1.31%, 91.66%, and 95.53%, respectively, where DR is 3.29% higher than the original data. With the MLP model, SGM achieves a DR, which is 0.04% lower than that of ROS, but significantly outperforms ROS in terms of other indicators, where the DR-related F_1score is 0.29% higher. With the CNN model, SGM does not achieve as good a DR as the other imbalanced methods, but yields an F_1score, which is 0.51% higher than that of ADASYN. With the RF model, SGM has the best DR. For the MLP and CNN models, our method achieves a slightly lower DR but higher ACC, Precision and FAR than ROS and ADASYN. Note that F_1score is the harmonic mean of Precision and DR. The high F_1score indicates that our method has a better overall performance than the other methods in comparison.

In terms of the overall performance, no matter which classification algorithm is used, such as RF, MLP, and CNN, SGM consistently achieves the best results. In this set of experiments, the combined performance of under-sampling and oversampling is better than using oversampling alone. Clustering under-sampling reduces the number of samples in the training set, making the time cost of the classification model significantly lower. When using CNN for classification, the test time of the SGM method is only 8.39s, while the test time of the SMOTE is 55.49s. The classification effect of our proposed 1D-CNN is better than RF and MLP. In terms of time cost, CNN takes the longest time, followed by MLP and RF. One of the reasons is that it takes more time with multiple convolutional layers of CNN.

4.4. Multiclass classification

In the multiclass classification experiments, we use the same comparative experiment with the binary classification scenario except for ADASYN. The ADASYN method takes too long to process the data. The number of neural units in the output layer of the CNN and MLP is ten on the UNSW-NB15 dataset and 15 on the CICIDS2017 dataset, and the remaining parameters are the same as those in the binary classification. For the RF algorithm, we also use RandomizedSearchCV to determine its optimal parameters in multi-classification. We set n_estimators to be 157 and max_depth of the decision tree to be 27, and select the max-features decision method as “auto”.

The ten classification results on the UNSW-NB15 dataset of CNN are shown in Table 9, which highlights the DR of each class. The experimental results in Table 9 clearly show that the CNN model after class imbalance processing significantly improves the detection rate of attack classes, especially for “Backdoor”, “Shellcode” and “Worms”. The SGM method achieves the overall best performance in terms of DR, Precision,

Table 10
Multi-class classification performance comparison on the UNSW-NB15 dataset between SGM-CNN, RF and MLP (%).

Model	RF					MLP					CNN
	ROS	SMOTE	RUS+SMOTE	K-means+SMOTE	SGM	ROS	SMOTE	RUS+SMOTE	K-means+SMOTE	SGM	SGM
DR	95.79	95.79	95.89	95.80	96.18	**96.59**	96.56	96.51	96.55	96.56	96.54
Precision	98.06	97.95	97.96	97.94	98.11	98.08	98.17	98.19	98.21	98.23	**98.30**
F_1 score	96.63	96.61	96.69	96.62	96.87	97.21	97.22	97.21	97.21	97.23	**97.26**

Table 11
Performance evaluation of the proposed model in multi-class classification on the CICIDS2017 dataset using CNN, highlighting the DR of each class (%).

Class	77	ROS	SMOTE	RUS+SMOTE	K-means+SMOTE	SGM
BENIGN	1.00	1.00	1.00	1.00	1.00	1.00
DoS Hulk	1.00	1.00	1.00	1.00	1.00	1.00
PortScan	1.00	1.00	1.00	1.00	1.00	1.00
DDoS	1.00	1.00	1.00	1.00	1.00	1.00
DoS GoldenEye	1.00	1.00	1.00	1.00	1.00	1.00
FTP-Patator	1.00	1.00	1.00	1.00	1.00	1.00
SSH-Patator	1.00	1.00	1.00	1.00	1.00	1.00
DoS slowloris	0.99	0.99	0.99	1.00	1.00	1.00
DoS Slowhttptest	0.99	0.99	0.99	1.00	1.00	1.00
Bot	0.98	1.00	1.00	1.00	1.00	1.00
Web Attack Brute Force	0.97	0.69	0.69	0.49	0.45	0.50
Web Attack XSS	0.04	0.52	0.47	0.84	0.86	0.82
Infiltration	0.57	0.71	0.71	0.71	0.71	0.71
Web Attack Sql Injection	0.75	1.00	1.00	1.00	1.00	1.00
Heartbleed	0.50	0.50	0.50	1.00	1.00	1.00
DR	99.93	99.76	99.78	99.72	99.70	**99.85**
ACC	99.93	99.76	99.78	99.72	99.70	**99.85**
Precision	99.93	99.81	99.82	99.81	99.81	**99.88**
F_1 score	99.92	99.78	99.79	99.75	99.74	**99.86**
Train-Time(s)	43.27	558.84	625.28	58.53	58.13	67.16
Test-Time(s)	55.23	57.91	58.12	18.26	18.53	45.23

Table 12
Performance comparison of multiclass classification on the CICIDS2017 dataset between SGM-CNN, RF and MLP (%).

Model	RF					MLP					CNN
	ROS	SMOTE	RUS+SMOTE	K-means+SMOTE	SGM	ROS	SMOTE	RUS+SMOTE	K-means+SMOTE	SGM	SGM
DR	92.85	92.87	94.19	92.51	93.08	99.54	99.51	99.63	99.60	99.64	**99.85**
Precision	97.47	97.43	98.01	96.65	97.03	99.70	99.72	99.76	99.73	99.76	**99.88**
F_1 score	94.75	94.74	95.81	94.11	94.67	99.60	99.60	99.68	99.65	99.69	**99.86**

and F_1 score of 96.54%, 98.30%, and 97.26%, respectively. The SGM method is slightly better than SMOTE in the overall performance, but the SGM method greatly reduces the test time, from 52.63s of SMOTE to 8.26s. The detection rate of “Analysis”, “DoS” and “Explores” does not exceed 50%, which is one of the main factors limiting the overall performance. One reason is that there is some similarity between the features of the data. Especially for “Exploits”, the DR decreases after imbalanced processing, indicating that SMOTE oversampling this class of data makes its boundary with the neighbouring class more blurred.

Table 10 shows the performance comparison on the UNSW-NB15 dataset of the SGM-CNN model with RF and MLP. From Table 10, we conclude that the effect of MLP is similar to CNN, but CNN is slightly better than MLP, and the performance of RF is the worst. SGM significantly improves classification performance when using RF classification algorithm. The DR of SGM-CNN is 0.05% lower than ROS-MLP. Considering DR, Precision, and F_1 score, the SGM-CNN model achieves the best overall performance, whether combined with machine learning or deep learning.

The 15-class classification results on the CICIDS2017 dataset of CNN are shown in Table 11. The results in Table 11 clearly show that the CNN model after class imbalance processing improves the DR of attack classes, especially for “Web Attack XSS”, “Infiltration” and “Web Attack Sql Injection”. In particular, the DR of Heartbleed increases from 50% to 100%. The DR of “Web Attack Brute Force” decreases by about 50%, which is the main factor limiting the overall DR. The SGM method achieves the best performance in terms of DR, Precision, and F_1 score of 99.85%, 99.88%, and 99.86%, respectively. The test time of the SGM method is shorter than the test time of oversampling alone, but longer than RUS + SMOTE and K-means + SMOTE. From Tables 9 and 11, we observe that with the CNN classification algorithm, the oversampling method alone is better than the combination of RUS or K-means under-sampling and SMOTE, but is not as good as the combination of GMM-based clustering under-sampling and SMOTE.

Table 12 shows the performance comparison of SGM-CNN with RF and MLP on the CICIDS2017 dataset. From Table 12, we observe that CNN has a significantly better effect than MLP and RF, and the performance of RF is still the worst. SGM yields the best performance with CNN and MLP, but the results with RF are not as good as RUS + SMOTE. These results show that SGM-CNN has a better overall performance than all other models.

5. Discussion

The experimental results show that our proposed method SGM, combining SMOTE oversampling and GMM-based clustering under-sampling, significantly improves the DR of attacks and effectively

Table 13
A comparison of the proposed model and previous studies (%).

Dataset		Model	ACC	DR	FAR	F_1 score
UNSW-NB15	Binary Classification	RepTree [32]	88.95	-	-	-
		DAE+MLP [20]	98.80	94.43	**0.57**	95.20
		IG+ANN [13]	97.04	-	1.48	-
		In this study	**98.82**	**99.74**	1.31	**95.53**
	Multiclass Classification	CSCADE-ANN [34]	86.40	86.74	13.10	-
		RCNF [31]	95.98	-	**4.02**	-
		ICVAE-DNN [40]	89.08	95.68	19.01	90.61
		SCM3+RF [12]	95.87	**97.80**	7.70	-
		In this study	**96.54**	96.54	-	**97.26**
CICIDS2017	Multiclass Classification	PCA+RF/BN [38]	99.60/94.80	-	-	98.80/96.40
		AE+RF/BN [38]	99.60/95.30	-	-	99.60/99.60
		DNN/RF [35]	99.57/92.72	-	-	-
		PCCN [18]	**99.87**	98.21	-	98.65
		In this study	99.85	**99.85**	-	**99.86**

reduces time cost. There are two reasons why the SGM method is superior. First, in GMM-based clustering under-sampling, GMM computes the probability that each sample belongs to each cluster. Compared to K-means clustering, which randomly selects the initial clustering center, and RUS, which randomly deletes samples, GMM is more general (it can find clusters of different sizes and ellipsoid shapes) and has more stable performance. Second, using SMOTE to oversample minority classes to the uniform number of instances reduces the risk of introducing too much redundant data.

To further demonstrate the effectiveness of our proposed SGM-CNN intrusion detection model, as shown in Table 13, we compare with several state-of-the-art intrusion detection methods using the UNSW-NB15 and CICIDS2017 datasets. Whether compared with traditional machine learning technologies (such as RF and RepTree) or with deep neural networks, our model has demonstrated its superiority, especially in improving the detection rate of various categories.

6. Conclusions

Considering class imbalance in intrusion detection dataset, which seriously affects the DR of each class, we proposed a flow-based intrusion detection model SGM-CNN that combines class imbalance processing technology SGM and convolutional neural network. We designed a new class imbalance processing technology SGM for large-scale data, using SMOTE oversampling and GMM-based clustering under-sampling to resample all class samples to a uniform number of instances. Compared with five types of imbalanced processing technologies and two widely-used classification algorithms, for both binary and multiclass classification, the proposed SGM-CNN model achieves the best performance. Especially in multi-classification, SGM significantly improves the DR of attack classes. The DR of the SGM-CNN model on the UNSW-NB15 dataset for binary and ten classifications reaches 99.74% and 96.54%, respectively. The DR, Precision and F_1score of SGM-CNN are 99.85%, 99.88% and 99.86%, respectively, in the multi-classification experiment on the CICIDS2017 dataset. Our model is superior to the state-of-the-art methods and points to a promising direction for future intrusion detection against large imbalanced data sets. In our future research, we plan to explore other feature selection methods to improve the detection performance.

Data Availability

The UNSW-NB15 dataset used to support the findings of this study is available at https://www.unsw.adfa.edu.au/unsw-canberra-cyber/cybersecurity/ADFA-NB15-Datasets/. The CICIDS2017 dataset used to support the findings of this study is available at https://www.unb.ca/cic/datasets/ids-2017.html.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

The research is partly supported by the Integration of Cloud Computing and Big Integration of Cloud Computing and Big Data, Innovation of Science and Education (grant number 2017A11017), the Key Research, Development, and Dissemination Program of Henan Province (Science and Technology for the People) (grant number 182207310002), and the Key Science and Technology Project of Xinjiang Production and Construction Corps (grant number 2018AB017).

Supplementary material

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.comnet.2020.107315 .

References

[1] Cisco, Cisco visual networking index: global mobile data traffic forecast update, 2017–2022, 2019, (https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/white-paper-c11-738429.html). Online; accessed 18 February 2019.
[2] H. Liu, B. Lang, Machine learning and deep learning methods for intrusion detection systems: a survey, Appl. Sci. 9 (2019) 4396, doi:10.3390/app9204396.
[3] W. Haider, J. Hu, J. Slay, B.P. Turnbull, Y. Xie, Generating realistic intrusion detection system dataset based on fuzzy qualitative modeling, J. Netw. Comput. Appl. 87(2017) 185–192.
[4] N.N. Tran, R. Sarker, J. Hu, An approach for host-based intrusion detection system design using convolutional neural network, in: J. Hu, I. Khalil, Z. Tari, S. Wen (Eds.), Mobile Networks and Management, Springer International Publishing, 2018, pp. 116–126.
[5] A.M. Mahfouz, D. Venugopal, S.G. Shiva, Comparative analysis of ML classifiers for network intrusion detection, in: Fourth International Congress on Information and Communication Technology, Springer, 2020, pp. 193–207.
[6] A.F.M. Agarap, Acm, A neural network architecture combining gated recurrent unit (GRU) and support vector machine (SVM) for intrusion detection in network traffic data, in: Proceedings of the 2018 10th International Conference on Machine Learning and Computing, in: ICMLC 2018, Association for Computing Machinery, New York, NY, USA, 2018, pp. 26–30, doi:10.1145/3195106.3195117.
[7] M.S.M. Pozi, M.N. Sulaiman, N. Mustapha, T. Perumal, Improving anomalous rare attack detection rate for intrusion detection system using support vector machine and genetic programming, Neural Process. Lett. 44(2) (2017) 279–290. 10.1007/s11063-015-9457-y
[8] Y.-W.C. Wei Zong, W. Susilo, Interactive three-dimensional visualization of network intrusion detection data for machine learning, Future Gener. Comput. Syst. 102 (2020) 292–306, doi:10.1016/j.future.2019.07.045.
[9] P. Sangkatsanee, N. Wattanapongsakorn, C. Charnsripinyo, Practical real-time intrusion detection using machine learning approaches, Comput. Commun. 34 (18) (2011) 2227–2235.
[10] J. Zhang, M. Zulkernine, A. Haque, Random-forests-based network intrusion detection systems, IEEE Trans. Syst. Man Cybern.Part C 38 (5) (2008) 649–659, doi:10.1109/tsmcc.2008.923876.

[11] A. Tesfahun, D.L. Bhaskari, Intrusion detection using random forests classifier with SMOTE and feature reduction, in: Proceedings of the 2013 International Conference on Cloud & Ubiquitous Computing & Emerging Technologies, in: CUBE '13, IEEE Computer Society, USA, 2013, pp. 127–132, doi:10.1109/CUBE.2013.31.
[12] A. Binbusayyis, T. Vaiyapuri, Identifying and benchmarking key features for cyber intrusion detection: an ensemble approach, IEEE Access 7 (2019) 106495–106513, doi:10.1109/ACCESS.2019.2929487.
[13] N. Koroniotis, N. Moustafa, E. Sitnikova, J. Slay, Towards developing network forensic mechanism for botnet activities in the IoT based on machine learning techniques, Mob. Netw. Manage. 235 (2018) 30–44, doi:10.1007/978-3-319-90775-8_3.
[14] N. Farnaaz, M. Jabbar, Random forest modeling for network intrusion detection system, Procedia Comput. Sci. 89 (2016) 213–217, doi:10.1016/j.procs.2016.06.047.
[15] T.T. Bhavani, M.K. Rao, A.M. Reddy, Network intrusion detection system using random forest and decision tree machine learning techniques, in: First International Conference on Sustainable Technologies for Computational Intelligence, Springer Singapore, Singapore, 2020, pp. 637–643.
[16] Y. Liu, C. Wang, Y. Zhang, J. Yuan, Multiscale convolutional CNN model for network intrusion detection, Comput. Eng. Appl. 55 (3) (2019) 90, doi:10.3778/j.issn.1002-8331.1712-0021.
[17] Y. Xiao, C. Xing, T. Zhang, Z. Zhao, An intrusion detection model based on feature reduction and convolutional neural networks, IEEE Access 7 (2018) 342210–42219. 10.1109/ACCESS.2019.2904620
[18] Z. Yong, C. Xu, G. Da, S. Mei, T. Yinglei, W. Xiaojuan, PCCN: Parallel cross convolutional neural network for abnormal network traffic flows detection in multi-class imbalanced network traffic flows, IEEE Access 7 (2019) 119904–119916, doi:10.1109/ACCESS.2019.2933165.
[19] Y.Z.L.H. X. Zhang J. Chen, J. Lin, A multiple-layer representation learning model for network-based attack detection, IEEE Access 7 (2019) 91992–92008.
[20] H. Zhang, C.Q. Wu, S. Gao, Z. Wang, Y. Xu, Y. Liu, An effective deep learning based scheme for network intrusion detection, in: 2018 24th International Conference on Pattern Recognition (ICPR), 2018, pp. 682–687, doi:10.1109/ICPR.2018.8546162.
[21] C. Yin, Y. Zhu, J. Fei, X. He, A deep learning approach for intrusion detection using recurrent neural networks, IEEE Access 5 (2017) 21954–21961, doi:10.1109/ACCESS.2017.2762418.
[22] M. Sheikhan, Z. Jadidi, A. Farrokhi, Intrusion detection using reduced-size RNN based on feature grouping, Neural Comput. Appl. 21 (6) (2012) 1185–1190, doi:10.1007/s00521-010-0487-0.
[23] N. Moustafa, 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, Inf. Secur. J. 25 (1–3) (2016) 18–31, doi:10.1080/19393555.2015.1125974.
[24] M. Galar, A. Fernandez, E. Barrenechea, H. Bustince, F. Herrera, A review on ensembles for the class imbalance problem: bagging-, boosting-, and hybrid-based approaches, IEEE Trans. Syst. Man Cybern.Part C 42 (4) (2012) 463–484.
[25] X.Y. Liu, J.X. Wu, Z.H. Zhou, Exploratory undersampling for class-imbalance learning, IEEE Trans. Syst. Man Cybern.Part B 39 (2) (2009) 539–550, doi:10.1109/tsmcb.2008.2007853.
[26] N.V. Chawla, K.W. Bowyer, L.O. Hall, W.P. Kegelmeyer, SMOTE: Synthetic minority over-sampling technique, J. Artif. Intell. Res. 16 (1) (2002) 321–357.
[27] H. He, B. Yang, E.A. Garcia, S. Li, ADASYN: Adaptive synthetic sampling approach for imbalanced learning, in: Neural Networks, 2008. IJCNN 2008. (IEEE World Congress on Computational Intelligence). IEEE International Joint Conference on, 2008, pp. 1322–1331, doi:10.1109/IJCNN.2008.4633969.
[28] M. Buda, A. Maki, M.A. Mazurowski, A systematic study of the class imbalance problem in convolutional neural networks, Neural Netw. 106 (2018) 249–259, doi:10.1016/j.neunet.2018.07.011.
[29] H. Zhang, L. Huang, SGM-CNN, 2020, (https://github.com/zhang-hongpo/SGM-CNN). Accessed 18 April 2020.
[30] U. Ravale, N. Marathe, P. Padiya, Feature selection based hybrid anomaly intrusion detection system using k-means and RBF kernel function, Procedia Comput. Sci. 45 (2015) 428–435, doi:10.1016/j.procs.2015.03.174.
[31] N. Moustafa, J. Slay, RCNF: Real-time collaborative network forensic scheme for evidence analysis (2017). arXiv:1711.02824.
[32] M. Belouch, S. El, M. Idhammad, A two-stage classifier approach using REPTree algorithm for network intrusion detection, Int. J. Adv. Comput. Sci. Appl. 8 (6) (2017) 389–394, doi:10.14569/IJACSA.2017.080651.
[33] D. Kwon, H. Kim, J. Kim, S.C. Suh, I. Kim, K.J. Kim, A survey of deep learning-based network anomaly detection, Cluster Comput. (2017), doi:10.1007/s10586-017-1117-8.
[34] E.M. El-Sayed, M., M.M. Awais, B.M. Mirza, A multiclass cascade of artificial neural network for network intrusion detection, J. Intell. Fuzzy Syst. 32 (4) (2017) 2875–2883, doi:10.3233/JIFS-169230.
[35] O. Faker, E. Dogdu, Intrusion detection using big data and deep learning techniques, 2019, doi:10.1145/3299815.3314439.
[36] N. Japkowicz, The class imbalance problem: Significance and strategies, in: Proceedings of the International Conference on Artificial Intelligence. IC-AI'2000, vol. 1, 2000, pp. 111–117.
[37] D.A. Cieslak, N.V. Chawla, A. Striegel, Combating imbalance in network intrusion datasets, in: 2006 IEEE International Conference on Granular Computing, 2006, pp. 732–737, doi:10.1109/GRC.2006.1635905.
[38] R. Abdulhammed, H. Musafer, A. Alessa, M. Faezipour, A. Abuzneid, Features dimensionality reduction approaches for machine learning based network intrusion detection, Electronics 8 (3) (2019) 27, doi:10.3390/electronics8030322.
[39] Y. Zhu, J. Liang, J. Chen, Z. Ming, An improved NSGA-III algorithm for feature selection used in intrusion detection, Knowl. Based Syst. 116 (2016) 74–85, doi:10.1016/j.knosys.2016.10.030.
[40] Y. Yang, K. Zheng, C. Wu, Y. Yang, Improving the classification effectiveness of intrusion detection by using improved conditional variational autoencoder and deep neural network, Sensors 19(11) (2019) 2528. 10.3390/s19112528
[41] I. Ullah, Q. Mahmoud, A two-level hybrid model for anomalous activity detection in IoT networks, 2019, pp. 1–6, doi:10.1109/CCNC.2019.8651782.
[42] G. Karatas, O. Demir, O.K. Sahingoz, Increasing the performance of machine learning-based IDSs on an imbalanced and up-to-date dataset, IEEE Access 8 (2020) 32150–32162.
[43] N. Moustafa, J. Slay, IEEE, UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set), in: IEEE: 2015 Military Communications and Information Systems Conference, 2015.
[44] A.H.L. Iman Sharafaldin, A.A. Ghorbani, Toward generating a new intrusion detection dataset and intrusion traffic characterization, ICISSP, 2018.
[45] A.H.L.I. Sharafaldin, A. Gharib, A. Ghorbani, Towards a reliable intrusion detection benchmark dataset, Softw. Netw. (2017) 177–200, doi:10.13052/jsn2445-9739.2017.009.
[46] A. Gharib, I. Sharafaldin, A. Habibi Lashkari, A. Ghorbani, An evaluation framework for intrusion detection dataset, 2016, pp. 1–6, doi:10.1109/ICISSEC.2016.7885840.
[47] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition (2015). arXiv:1409.1556v6.
[48] A. Tahmassebi, A.H. Gandomi, S. Fong, A. Meyer-Baese, S.Y. Foo, I. Olier, Multi-stage optimization of a deep model: a case study on ground motion modeling, PLoS ONE 13 (9) (2018) 1–23, doi:10.1371/journal.pone.0203829.
[49] L. Breiman, Random forests, Mach. Learn. 45 (1) (2001) 5–32, doi:10.1023/a:1010933404324.
[50] J. Bergstra, Y. Bengio, Random search for hyper-parameter optimization, J. Mach. Learn. Res. 13 (1) (2012) 281–305.

Hongpo Zhang received the B.S. degree in computer science and application from Zhengzhou University of Technology, China, in 1997. He is currently pursuing the Ph.D. degree at the State Key Laboratory of Mathematical Engineering and Advanced Computing, China. He has been conducting research at Zhengzhou University since 1997. His research interests include intelligent healthcare and network security.

Lulu Huang received the B.S. degree from Jiangxi University of Science and Technology of Information Security in 2018. She is currently pursuing the M.S. degree in computer science and technology at Zhengzhou University. Her current research interests include intrusion detection and machine learning.

Chase Q. Wu is currently a Professor and the Associate Chair in the Department of Computer Science and the Director of the Center for Big Data at New Jersey Institute of Technology (NJIT). He joined NJIT in fall 2015 from the University of Memphis, where he is an Associate Professor in the Department of Computer Science. His research interests include big data, high-performance networking, parallel and distributed computing, sensor networks, scientific visualization, and cybersecurity.

Zhanbo Li is currently the director of the Network Management Center of Zhengzhou University and the dean of the School of Distance Education. He obtained the master degree from Sun Yat-Sen University in 1993. His research is mainly focused on cybersecurity.