Journal of Computers Vol. 34 No. 5, October 2023, pp. 213-227
doi: 10.53106/199115992023103405017

Intrusion Detection Based on Feature Reduction and Model Pruning in Electricity Trading Network

Zhenzhen Liu[1], Rui Zhou[1], Kangqian Huang[1], Xin Hu[1],
Zhe Jiang[2], Binsi Cai[2*], Kaiguo Yuan[2]

[1] Information Data Department, Guangdong Power Exchange Center Co. Ltd.,
Guangdong 510080, China

clz0502@163.com, {zhourui, huangkangqian}@gd.csg.cn,278271099@qq.com

[2] A Key Laboratory of Trustworthy Distributed Computing and Service, Beijing University of Posts and Telecommunications, Beijing 100876, China

{jiangzhe1997, caibinsi, flyingdreaming}@bupt.edu.cn

Received 1 August 2023; Revised 1 September 2023; Accepted 28 September 2023

Abstract. The electricity trading network increases network flexibility and lowers trading costs with the aid of 5G and IOT technology. While it has improved trading efficiency and enhanced system intelligence, its security vulnerabilities pose significant challenges. In this study, we propose an intrusion detection method that focuses on feature reduction and model pruning in electricity trading network. The method effectively addresses the imbalance issue of the IDS2017 dataset by employing the SMOTE algorithm, reduces feature size and computational complexity through the application of PCA, autoencoder, and random forest techniques, and develops a lightweight intrusion detection model specifically designed for electricity trading network using model pruning and compression techniques. Experimental results demonstrate the effectiveness of the proposed model in detecting intrusions. The achieved precision, recall, F1 score, and false positive rate are at least 98.8%, 87.9%, 90.0%, and 0.08%, respectively. Furthermore, we conducted a comparative analysis of different pruning thresholds and determined that reducing the dimensionality to 49 dimensions yields superior model performance, making it particularly suitable for resource-constrained electricity trading network.

Keywords: electricity trading network, intrusion detection, feature selection, deep learning, convolutional neural network

1 Introduction

In recent years, the rapid growth and advancement of electricity trading network have revolutionized the energy industry, enabling efficient and dynamic electricity transactions [1]. With the development of digital technology and the rise of the trend towards intelligence, the electricity trading network in industrial control networks is undergoing digital transformation, utilizing an increasing amount of Internet of Things (IoT) technology. By connecting and integrating various IoT devices and sensors, it enables real-time monitoring and data transmission of power equipment status, load demand, and market information, enhancing system intelligence and responsiveness [2]. These sensors and actuators can sense their surrounding environment, communicate with each other, and exchange data through the Internet [3]. The IOT technology in electricity trading network enables people to monitor and control these devices remotely through the Internet and enables these devices to coordinate and cooperate automatically [4], thus helping to improve the efficiency of trade work.

Meanwhile, 5G technology is a new generation of mobile communication technology that offers faster speed, lower latency, greater connection density and stronger network capacity. These characteristics make 5G the driving force for the development of the electricity trading network [5]. 5G provides high-speed low-latency communication and supports massive connectivity for the power trading system, enabling real-time decision-making and more intelligent energy management. Electricity trading devices, equipped with sensors, controllers, and communication modules, enable the monitoring, control, and data transmission of power devices, thereby facilitating intelligent power trading and management.

IoT has gradually become an indispensable part of electricity trading network, and have benefited people in multiple ways. However, it also faces several inherent challenges [6]: 1) Connectivity of electricity devices:

* Corresponding Author

Due to their connectivity characteristics, these devices can easily connect to the Internet, but this also increases security risks [7]. Moreover, since these devices are often connected over unsecured channels like wireless networks, there is a possibility of confidential information being snooped during network communication. 2) The openness of electricity devices: electricity devices are typically open systems, allowing devices from different manufacturers to interconnect. While this interoperability enhances functionality, it also introduces complexity and greater security risks [8]. Additionally, hardware protocols, such as AMQP, MQTT, CoAP, and HTTP, add complexity to providing reliable network security methods [9]. 3) Resource-constrained electricity environment: electricity devices are usually embedded devices with limited hardware and software resources, resulting in restricted storage space and operating environments. This makes them more vulnerable to attacks [10]. Due to their limited computing resources and energy, electricity devices cannot support advanced security implementations. Consequently, they are more susceptible to attacks compared to traditional computing environments [11].

It is necessary to adopt more effective and intelligent methods to detect security threats in electricity trading network, among which intrusion detection systems (IDS) are a typical representative. Intrusion Detection System (IDS) is a network security technology used to monitor network traffic and identify suspicious or malicious activities [12]. IDS is a software application or tool used to monitor a network to ensure its security and detect any malicious activity that occurs [13]. Most IDS solutions have been adopted from existing computer networks, wireless sensor networks and mobile ad hoc networks. However, the unique characteristics of electricity trading network, such as connectivity to the global internet and lightweight resources, make the IDS proposed for these networks unsuitable for electricity trading applications [14]. Intrusion detection in the context of electricity trading network needs to take into account the unique characteristics of networks and devices, such as limited network bandwidth and limited device capabilities [15]. It is necessary to adopt intrusion detection technologies that are suitable for electricity trading scenarios. At the same time, it is necessary to consider the large amount of data and the large number of devices in the Internet of Things system, and to adopt a distributed intrusion detection system suitable for processing high-dimensional data to improve detection efficiency [16].

In this paper, we propose a novel anomaly detection framework for the limited computing and storage resources in electricity trading network. This study performs dimension reduction on the original data and uses CNN (Convolutional Neural Networks) to establish the original anomaly detection model. Then, the constructed model is pruned and quantized to make it more suitable for the electricity trading network.

The contributions of this paper are as follows:

- We used the IDS2017 dataset and addressed its imbalance issue by utilizing the SMOTE algorithm, which resulted in a more balanced distribution of the original features.
- To reduce the input data size and subsequent computational complexity of the high-dimensional data, we employed various dimensional reduction methods, including PCA, autoencoder, and random forest.
- Building upon the traditional CNN deep learning algorithm, we developed a lightweight intrusion detection model for electricity trading network with limited computational and resource capabilities. We achieved more efficient intrusion detection through model pruning and compression techniques.

The rest of the paper is organised as follows. Section II discussed the related work. The proposed model is presented in Section III. In Section IV, data collection strategies and datasets used are discussed. The analysis of the evaluation results are presented in Section V, with discussion and comparison results in Section VI. Finally, Section VII concludes the paper and offers ideas for future work.

2 Related Work

IDS solutions have received intensive attention from researchers and industries in the electricity trading network, and many IDS solutions have been proposed. Based on the detection approach, IDS solutions can be categorized into three approaches: signature, anomaly, and hybrid IDS model [17]. In general, the signature-based approach is effective for known attacks, while the anomaly-based is effective for unknown attacks [18]. However, due to the heterogeneity, dynamicity, and complex nature of the electricity network, the signature based approach is inefficient and ineffective because it requires continuous human interventions and knowledge expertise to extract attack patterns and signatures to update the IDS model [19]. Anomaly-based IDS detection gains advantages because it detects zero-day attacks and needs fewer human interventions [20]. The hybrid approach combines both signatures-based and anomaly-based approaches. However, because it is impractical to rely on pre-defined attack patterns (signature-based) intrusion detection in IoT, the utilization of the signature-based IDS is limited in electricity network [21]. To this end, anomaly intrusion detection systems play a vital role in intrusion detection in

electricity network.

Traditional methods of intrusion detection include signature analysis and threshold analysis methods. These analytical methods are based on a large number of existing abnormal samples, with corresponding thresholds and templates set in advance [22]. When a network packet matches these thresholds and templates, it will be identified as an attack behavior. This type of method usually has high accuracy and low false positive rates, but cannot detect unknown or variant attacks, so it is unable to effectively detect unknown attacks such as zero-day vulnerabilities [23]. Meanwhile, said criterion is also only a summary of the malicious traffic behavior found in the past and is typically difficult to quantify, so these methods cannot be readily adapted to the huge amount of network data. Therefore, some methods of log anomaly detection based on machine learning or deep learning have been proposed.

The intrusion detection method based on machine learning involves learning and analyzing a large amount of existing normal and abnormal network traffic data, and constructing a machine learning classification model through data processing, feature representation, and other processes to classify unknown network traffic [24]. Advancements in machine learning have produced models that effectively classify and cluster traffic for the purposes of network security. Early researchers attempted simple machine learning algorithms for classification-clustering problems in other fields, such as the k-Nearest Neighbor (KNN) [25] support vector machine (SVM) [26], with good results on KDD99, NSL-KDD, DARPA, and other datasets. these datasets are out of date, unfortunately, and contain not only normal data but also attack data that are overly simple. It is difficult to use these datasets to simulate today' s highly complex network environment. It is also difficult to achieve the expected effect using these algorithms to analyze malicious traffic in a relatively new dataset, as evidenced by our work in this study.

However, ML based methods have limited ability to capture features and it is difficult to analyze and extract hidden relationships. The development of deep learning in recent years has brought new breakthroughs to intrusion detection [27]. Deep learning is an end-to-end process that is capable of learning how to derive successful features from raw data, without taking time and labor-intensive hand-made applications [28]. Nowadays, deep learning approaches are leveraged in the intrusion detection domain to improve automation and accuracy.

Priya et al [29] suggested a DNN (Deep Neural Network) to recognize and forecast unexpected cyberattacks in IoMT (Internet of Medical Things) networks to establish reliable and productive IDS. The proposed DNN framework achieved improved accuracy and a 32% reduction in computation time, allowing quicker detection to prevent post-intrusion effects in critical cloud computing. Manimurugan et al [30] proposed a deep learning-based method Deep Belief Network (DBN) algorithm model for the intrusion detection system. In evaluating the performance of the proposed deep learning model DBN-IDS system and they used the CICIDS-2017 dataset for detection of attacks. They compared them with methods such as DNN, RNN, SVM etc, and their proposed model produced better results in all the parameters in relation to accuracy, recall, precision, F1-score, and detection rate.

Convolutional neural networks (CNNs), as the most widely used algorithm for analysing the data in depth using deep learning, have achieved excellent research results in computer vision, speech recognition, and natural language processing [31]. The CNN handles all kinds of datasets including network traffic dataset, and it is able to analyse the features. It can learn better features automatically than traditional feature selection algorithms [32]. In fact, the structure of packets and traffic is very similar to words, sentences, and articles compared to bytes in a network stream. Therefore, CNNs can not only select features but also classify the traffic data. The more traffic data, the more useful features the CNN can learn, the better classification the CNN performs [33]. Hence, CNN is suitable for the massive network environment. Besides, compared with other DL algorithms, the greatest advantage of CNN is that it shares the same convolutional kernels, which would reduce the number of parameters and calculation amount of training once greatly, it can more quickly identify attack type of traffic data.

In the context of the electricity trading network, the task of intrusion detection is more complex because of the large number of IoT devices and the large amount of data per device. Many studies combine deep learning and feature selection related technologies to improve the performance of intrusion detection. These feature selection algorithms can select the most useful features for intrusion detection, thereby reducing the impact of noise and redundant data and improving the performance of the model [34]. Xiao et al [35] proposed a new deep learning approach using CNN that selects the relevant features from dataset which has a huge data by incorporating the dimensionality reduction. Their approach has been improved the classification accuracy and also reduces the classification time. Selvakumar and Muneeswaran [36] developed a firefly algorithm which works based on filter and wrapper method for selecting the best features. They have selected only 10 features from KDD Cup dataset for detecting the attackers by using the mutual information value of features, wrapper-based Bayesian network and C4.5 algorithm. Their system computational cost was reduced. Kim and Cho [37] proposed a new CNN-

based system for improving the classification accuracy. Their system selects the probable and nonlinear features automatically using GA from the complex dataset by applying the correlation values of features.

Based on previous research, we utilize CNN to achieve efficient extraction of traffic features and compare several feature dimensionality reduction methods to evaluate the performance of different feature selection approaches. Additionally, while model pruning techniques have been widely applied in the field of image processing, they are rarely used in intrusion detection. Considering the limited computation and storage resources in IoT environments, this study applies model pruning methods to achieve efficient intrusion detection.

3 Lightweight Intrusion Detection Methods

In this section, we describe the system architecture in detail, and mainly introduce the core technologies of the model. The diagram of our model is shown in Fig. 1. It is divided into four parts in total. The first part is processing the data. In order to overcome the problem of large dimension, feature reduction is followed by data processing to eliminate features. The third part is using the CNN classifier to detect network intrusion. Finally, we utilize model pruning techniques to make the model more lightweight, making it more suitable for IoT environments. The following subsections will detail each step in the proposed method.

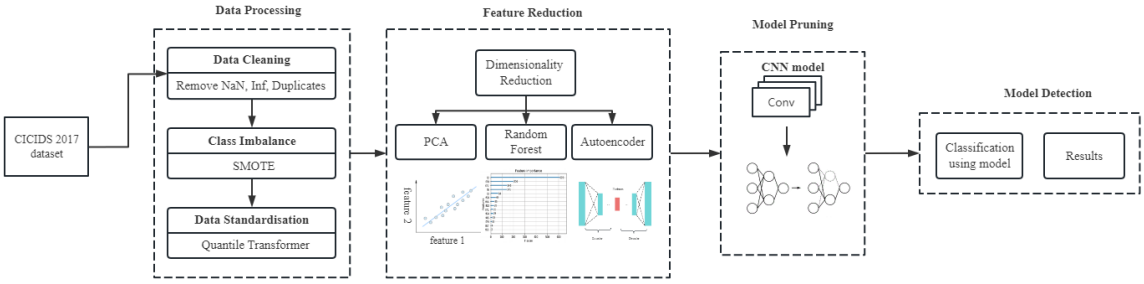

Fig. 1. The system architecture of our proposed method

3.1 SMOTE

In the field of intrusion detection, there is a problem of data imbalance due to the relatively small amount of malicious attack data compared to normal data. In the problem of data imbalance, the number of samples for a few categories is small, resulting in the trained classifier leaning too much towards the majority category and ignoring the samples for a few categories. This may lead to high misjudgment rates in practical applications. Thus, several oversampling techniques have emerged and become an essential preparatory step of applying any classification task considering imbalanced datasets. In this study, we used SMOTE technology to balance the dataset and improve the performance and robustness of the classifier to address this issue.

SMOTE sampling technology solves this problem by artificially synthesizing new minority class samples. SMOTE synthetically creates new samples considering the feature space of the instances rather than the instances themselves as a whole. This technique focuses on the features of the instances and the relationships between them to fulfil the minority class with additional instances. Specifically, it first randomly selects a sample from a few categories, then randomly selects a sample from its nearest neighbor, and multiplies the difference between these two samples by a random number to generate a new sample. Through this method, a few category samples in the dataset can be manually expanded, making the proportion of category samples in the training set more balanced. In this way, SMOTE creates new instances of the minority class to give richer information to the classification technique and thus enhances the quality of the prediction.

3.2 Feature Reduction

In the field of machine learning and data mining, due to the high-dimensional problem of feature dimensions, there will be many problems, such as training time, overfitting and noise. To address these issues, feature reduction technology is needed to reduce the high-dimensional feature space to a lower dimensional space. In IoT intrusion detection, feature dimensionality reduction technology can help us reduce the number of features and improve the efficiency and performance of intrusion detection systems. Feature dimensionality reduction techniques can be divided into two categories: feature selection and feature extraction. Feature selection is a method of directly selecting the most representative features from the original features, reducing the original feature set to a smaller subset. Common feature selection techniques include Filter, Wrapper, and Embedded methods. Feature extraction is a method of converting original features into new features through a certain algorithm, extracting the most representative information. Commonly used feature extraction techniques include PCA, LDA, etc. Here we choose the feature dimensionality reduction method used in this study for introduction.

PCA: Principal Component Analysis (PCA) is the most commonly used linear dimensionality reduction method in machine learning. It is widely used in data analysis and preprocessing. PCA aims to map high-dimensional data to low-dimensional space representations through linear projection. PCA is a statistical procedure which uses an orthogonal transformation. PCA converts a group of correlated variables to a group of uncorrelated variables. In order to reduce the dimensionality of the initial variables while preserving the variance in the sample as much as possible, many highly correlated variables can be transformed into independent or uncorrelated variables.

Suppose that a dataset has m objects x_1, x_2, x_m, and each object contains n variables. To obtain the n0 (n0 < n) principal components, the process is based on the following steps:

Standardization of the raw data as follows: The raw data should have unit variance and zero mean.

$$Z_{ij} = \frac{x_{ij} - \overline{x}_j}{s_j}, \quad i = 1,2,...,m; \quad j = 1,2,...,n. \tag{1}$$

$$where\ \overline{x}_j = \frac{\sum_{i=1}^{m} x_{ij}}{m},\ s_j^2 = \frac{\sum_{i=1}^{m} (x_{ij} - \overline{x}_j)^2}{m-1}. \tag{2}$$

Calculate the co-variance matrix of the raw data as follows:

$$R = \frac{Z^T Z}{m-1'}. \tag{3}$$

Calculate the eigenvector and eigenvalue of the co-variance matrix. Finding the eigenvalues $a_1 \geq a_2 \geq \ldots \geq a_j$ of R and the corresponding unit eigenvector.

$$a_1 = \begin{Bmatrix} \alpha_{11} \\ \alpha_{21} \\ \vdots \\ \alpha_{n1} \end{Bmatrix},\ a_2 = \begin{Bmatrix} \alpha_{12} \\ \alpha_{22} \\ \vdots \\ \alpha_{n2} \end{Bmatrix},\ ...,\ a_n = \begin{Bmatrix} \alpha_{1n} \\ \alpha_{2n} \\ \vdots \\ \alpha_{nn} \end{Bmatrix}. \tag{4}$$

Top k eigenvector of co-variance matrix are chosen as the principal components. These will be the new, original basis for the data. The Calculation of corresponding vector is given in Equation 5.

$$t_i = \alpha_{i1} Z_1 + \alpha_{i2} Z_2 + ... + \alpha_{in} Z_n,\ i = 1,2,...,k. \tag{5}$$

In this way if the raw data is with n dimensionality, it will be reduced to a new k dimensional representation of the data. The principal component n0 (n0 < n) is used as a new data vector to replace the original data. After feature extraction, the unimportant and redundant feature can be removed to the greatest extent.

Random Forest: The random forest is a collection of unpruned decision trees for classification or regression purposes. By employing a tree classification algorithm, random forest constructs numerous classification trees, each of which is created using a distinct bootstrap sample from the original data. When the new data requires classification, it is evaluated by each of the trees in the forest. Based on each tree's decision regarding the data's class, a vote is cast. Finally, the forest selects the class that receives the most votes as the classification for the new data.

The two commonly used variable important measures in RF are Gini importance index and permutation importance index (PIM). Gini importance index is directly derived from the Gini index when it is used as a node impurity measure. A feature's importance value in a single tree is the sum of the Gini index reduction over all nodes in which the specific feature is used to split. The overall variable importance for a feature in the forest is defined as the summation or the average of its importance value among all trees in the forest. Permutation importance measure (PIM) is arguably the most popular variable's importance measure used in RF. The RF algorithm does not use all training samples in the construction of an individual tree. That leaves a set of out of bag (OOB) samples, which can be used to measure the forest's classification accuracy. In order to measure a specific feature's importance in the tree, randomly shuffle the values of this feature in the OOB samples and compare the classification accuracy between the intact OOB samples and the OOB samples with the particular feature permutated. In this work, we have used PIM to select an important subset of feature among all features.

STEP 1: For each decision tree in the random forest, calculate its out-of-bag (OOB) error using the corresponding OOB data and denote it as err_{OOB1}.

STEP 2: Randomly permute the distribution of the i-th feature value X of all N original samples, and calculate the OOB error again. This is denoted as err_{OOB}.

STEP 3: Assuming there are Ntree trees in the random forest, the importance of feature X is calculated as $\frac{\Sigma err_{OOB1} - err_{OOB2}}{N_{tree}}$. The reason why this expression can be used as a measure of the importance of the corresponding feature is that if the OOB accuracy significantly decreases after introducing random noise to a particular feature, it indicates that this feature has a significant impact on the classification results of the samples, meaning it has a higher importance.

Autoencoder: Fig. 2 shows the structure of an Autoencoder, which consists of two parts: the encoder and the decoder. The encoder component transforms high-dimensional data to a lower-dimensional representation, while the decoder component maps the lower-dimensional data back to the original higher-dimensional space. The compressed layer represents the lower-dimensional data produced by the encoding process. The entire neural network is trained to learn the identity function $X_{out} = X$, by adjusting the weights and biases of each unit, in order to optimize the network's parameters.

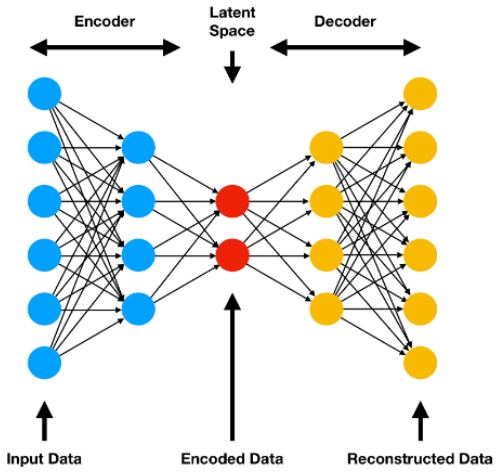

Fig. 2. The architecture of Autoencoder which consists of an encoder and a decoder.

An encoder and a decoder are represented as (6) and (7) respectively. W and b are weight matrix and bias matrix of encoder respectively. W^T and b' are weight matrix and bias matrix of decoder respectively.

$$E(x) : W(x) + b. \tag{6}$$

$$D(x) : W^T(x) + b'. \tag{7}$$

Auto-encoder identifies the difference between X and X_{out} as loss function, Mean Absolute Error (MAE) and Mean Square Error (MSE) are widely used loss functions in autoencoders. Equation (8) represents the loss function of auto-encoders.

$$\min(f_{loss} : (W^T(Wx + b) + b'), x). \tag{8}$$

$$C_{AE} = W_{AE} * I + b. \tag{9}$$

Autoencoder is typically used to reduce the dimensionality of high-dimensional data while retaining important features of the data. Through this process, an autoencoder can effectively reduce the dimensionality of high-dimensional data while preserving its important features. This makes it easier to visualize and analyze the data and can be used for downstream tasks in machine learning to improve performance and reduce the risk of overfitting.

3.3 Intrusion Detection

Convolutional neural networks (CNN) are widely used in the field of intrusion detection in the context of the electricity trading network. CNN can automatically learn features, adapt to dynamic environments, process large-scale data, and save computing resources. This makes it an important and applicable deep learning model in the field of electricity trading network intrusion detection. This subsection explains in detail about the existing convolutional neural network (CNN) model which is used for classifying the records as normal and attacker classes.

Fig. 3 describes the structure of the CNN which is composed of the input layer, convolutional layer, pooling layer, fully connected layer, and output layer. The main function of a convolutional layer is to extract features, and it can obtain many effective ones without any manual intervention. Convolutional layers always serve as the beginning structure of the whole network. Convolution kernel size can be defined to extract local features with different sizes and different convolution kernels can represent different features. The results output from the convolutional layer are mapped by a nonlinear operation, which is usually achieved through the activation function such as rectified linear unit (ReLU), Tanh, and Sigmoid function. The main purpose of the pooling layer is to reduce the dimension of current data, which is actually the process of sampling. Pooling layer not only retains the main features but also greatly reduces the computation of the model. Max pooling and average pooling are two major operations in this layer. One calculates the maximum of local units and the other figures out the average in the feature map. Finally, the fully connected (FC) layer determines the output category which the input belongs.

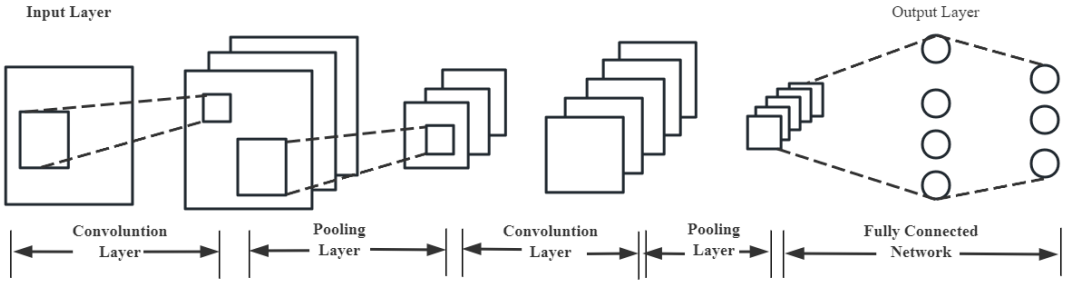

Fig. 3. The architecture of CNN

The CNNs of different structures have varying numbers of convolution layers and pooling layers. Assume that the input characteristic of the CNN is feature map of the layer i is M_{i-1} , the weight coefficient is indicated by W_i , the operation symbol $\otimes$ represents the convolution operation and the deviation quantity is b_i. Then, the convolution process can be expressed as

$$M_i = f(M_{i-1} \otimes W_i + b_i). \tag{10}$$

The convolutional layer extracts different feature information of the data matrix M_{i-1} by specifying different window values, and extracts different features M_i in the data through different convolution kernels. In the convolution operation, the same convolution kernel follows the principle of "parameter sharing" that is, sharing the same weight and offset which markedly reduces the number of parameters of the entire neural network. The pooling layer usually samples the feature map in accordance with different sampling rules after the convolutional layer.

The sampling criterion generally selects the maximum or mean value of the window region. The pooling layer mainly reduces the dimension of the feature, thereby decreasing the influence of redundant features on the model.

After several rounds of convolution and pooling, the CNN ultimately outputs the final result using the softmax function. The softmax function was used to calculate the probability distribution of an N-dimensional vector, which can be defined as:

$$O_i = \frac{e^{z_i}}{\sum_{i=1}^{M} e^{z_i}}. \tag{11}$$

where i and z_i are the output from pervious layers, O_i indicates the output of softmax function, and M is the total number of output nodes.

3.4 Model Pruning

Deep learning models have achieved tremendous success in fields such as image recognition, natural language processing, and speech recognition due to their powerful learning and inference capabilities. However, deep neural network models typically require a large number of parameters and have significant computational costs, which limit their practical deployment and implementation. Intrusion detection functions are mostly integrated into IoT devices such as IDS and IPS, which not only require accurate detection but also need to consider efficiency and cost issues. In today's context of massive data scale, complex data features, and diversified attack methods, intrusion detection methods that balance accuracy and efficiency are more suitable for specific scenarios with limited resources. In IoT intrusion detection, the application of techniques such as model pruning, compression, and quantization to CNN models can improve the efficiency and performance of the model, making it suitable for resource-constrained IoT environments. Therefore, to reduce the parameters and computational resources of deep neural networks and improve the efficiency and generalization performance of the model detection, this study performs model pruning based on the previous anomaly detection framework.

Model pruning reduces the size and computational complexity of the model by removing redundant and unimportant parameters. Common pruning methods include structural pruning and weight pruning. Structural pruning reduces the model size by removing entire neurons or layers, while weight pruning reduces the parameter count by setting the weights to zero. In intrusion detection, model pruning can reduce storage space and computational requirements, improving the operational efficiency of the model on IoT devices. Based on the pre-built anomaly detection architecture, this research adopts weight pruning based on L1 regularization to achieve model lightweighting. This method removes the connections with smaller weight values in the network by introducing L1 regularization, reducing the model size and computational complexity, thus achieving model compression and acceleration effects. As Fig. 4 shows, the pruning process in this study is as follows:

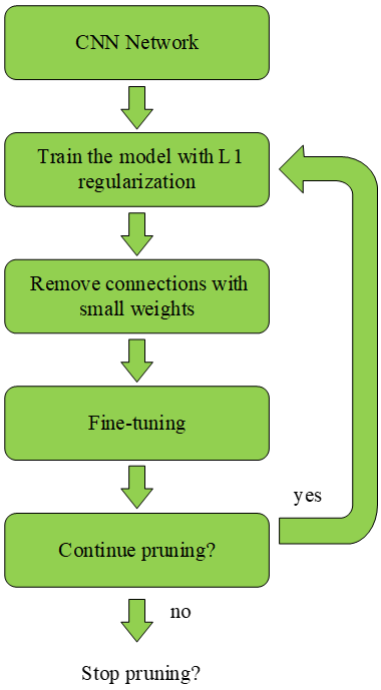

Fig. 4. The pruning process in this study

Step 1: Train the model with L1 regularization: On the previously built neural network model architecture for log anomaly detection, add L1 regularization term to the weight matrix of the model. The L1 regularization term is calculated by multiplying the L1 regularization coefficient with the sum of absolute values of all elements in the weight matrix. This term is then added to the original loss function, resulting in the following loss function L:

$$L = Loss + \lambda \sum |wi|, \tag{12}$$

where λ is the L1 regularization coefficient, and $|wi|$ represents the i-th element of the weight matrix.

Step 2: Remove connections with small weights: After training, for each weight matrix, sort the absolute values of the weights in descending order and remove a portion of the weights with the smallest absolute values, so that the remaining weights account for a predetermined threshold ratio of the total weights. This study evaluates the optimal pruning threshold by comparing different thresholds. After removing the connections with small weights, some unused connections are formed. These connections can be directly removed after the model training by setting their corresponding masks to 0, ensuring that their weights will not be updated in subsequent processes.

Step 3: Fine-tune the pruned model: This study fine-tunes the pruned model using the previous training data through multiple iterations of epochs. The above steps can be repeated multiple times, obtaining a more compact model through pruning, followed by retraining and fine-tuning until the desired model size and accuracy requirements are met.

4 Experimental Evaluation

In this section, we first describe the dataset used for our experiments and related data processing method. The we discusses the environmental settings, the evaluation measures, the Intrusion detection of conducting the experiments using the proposed approach, the comparative experiments with different parameters, comparison between the proposed approach and other standard algorithms, and a summary of the discussion.

4.1 Dataset

The dataset used in our experiment is CIC-ISDS-2017, which contains benign and the latest common attacks, similar to real world data. The CIC-IDS-2017 dataset provides multiple labeled data files, each containing a large number of network traffic records. The distribution of this CIC-ISDS-2017 is shown in Table 1. Each record includes various feature fields such as source IP address, destination IP address, source port, destination port, protocol type, packet length, and label. By analyzing these features, models can be built to detect and classify different types of network attacks. CIC-ISDS-2017 dataset reflects real traffic scenarios in real networks as well as newer means of attack. Meanwhile it contains benign traffic and up-to-date common attack traffic representative of a real network environment.

Table 1. Distribution of CIC-ISDS-2017 dataset

Traffic type	Traffic family	Count
Benign	Benign	2359087
Intrusion	DoS Hulk	231072
	PortScan	158930
	DDoS	41835
	DoS GoldenEye	10293
	FTP-Patator	7938
	SSH-Patator	5897
	DoS slowloris	5796
	DoS Slowhttptest	5499
	Bot	1966
	Infiltration	36
	Heartbleed	11
	Web Attack Brute Force	1507
	Web Attack Sql Injection	652
	Web Attack XSS	21

4.2 Feature Processing

Class Imbalance: In the context of the electricity trading environment, due to the large volume of data and its inherent imbalance, it is common to encounter situations where the number of normal samples far exceeds that of abnormal samples. This imbalance poses a challenge as machine learning models may not adequately learn from the rare abnormal samples during the training process, thereby impacting the accuracy of intrusion detection. As highlighted above, IDS datasets generally suffer from higher class imbalance compared to datasets from other domains. In our experiments, CICIDS2017 has 90% benign samples, with the rest containing different types of attacks, which can be showed as Fig. 5. This can fundamentally be attributed to the nature of these attacks, even in real-world deployments where the majority of the network traffic is expected to be benign.

Here, we address the imbalance issue in the dataset by focusing on the minority class and applying oversampling techniques. By utilizing the SMOTE algorithm for oversampling, we can enhance the recognition capability of the intrusion detection system for minority class abnormal samples in the context of the electricity trading network. This approach improves the accuracy and robustness of the model.

Data Standardisation: IDS datasets are sometimes high dimensional in nature (e.g. CICIDS2017 dataset has 78 features). In the case of such higherdimensional datasets without any form of normalization, machine learning models in general do not optimise very well or take much longer to train. We standardized the CICIDS2017 dataset to make it easier for the model to capture important patterns and regularities in the data. This helps improve the performance and accuracy of the intrusion detection model.

Dimensionality Reduction: In this study, we previously provided a detailed introduction to relevant feature selection techniques. In this study, we employed methods such as Autoencoder, PCA, and Random Forest to perform dimensionality reduction on the original data. By dimensionality reduction, we are able to reduce the 78 dimensions of the original dataset to the desired dimensions, effectively reducing computational complexity and facilitating its use as input for the CNN model. Here, we employ different dimensionality reduction methods to

reduce the features to 64, 49, 36, and 25 dimensions, respectively, in order to evaluate the effects of different dimensionality reduction methods and different dimensions.

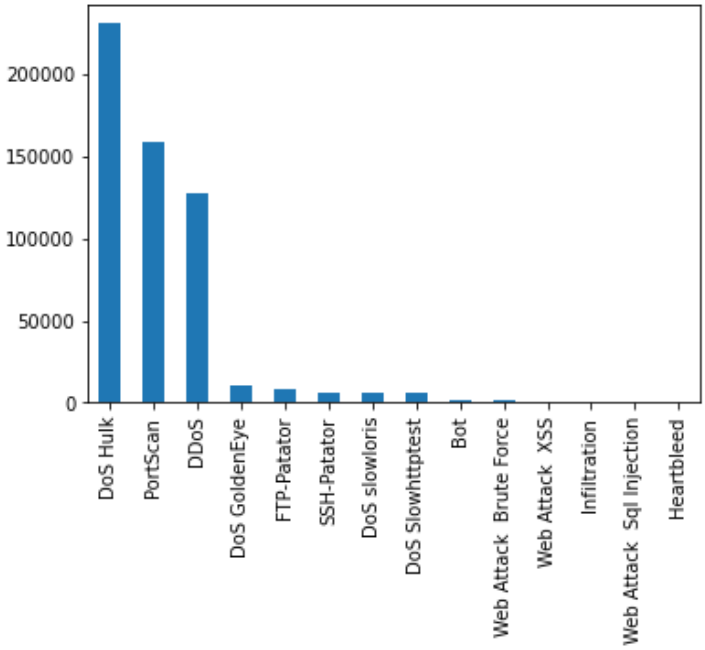

Fig. 5. Distribution of anomaly categories

4.3 Evaluation Metrics

Our experiment result is evaluated according to precision, recall, F1 score, false positive rate (FPR). These indexes were calculated by TP, FP, TN and FN, where TP represents that the predicted value and the actual value both are true; FP represents that the predicted value is true, while the actual value is false; FN indicates that the predicted value is false, but the actual value is true; TN means the predicted and true values both are false. Precision is calculated by Eq.(12), which measures the proportion of correct predictions among all predictions. Recall is calculated by Eq.(13), which shows the detected true anomalies in all real anomalies. F1-measure is calculated by Eq.(14), which represents the combination of the precision and recall. FPR is calculated by Eq.(15), which measures the proportion of negative instances that are incorrectly predicted as positive.

$$Precision = \frac{TP + TN}{TP + FP + FN + TN'}. \tag{13}$$

$$Recall = \frac{TP}{TP + FN'}. \tag{14}$$

$$F1 - measure = \frac{2\,Precision \cdot Recall}{Precision \cdot + Recall'}. \tag{15}$$

$$FPR = \frac{FP}{FP + TN'}. \tag{16}$$

4.4 Evaluation Results

Model Parameters. The various experiments have been conducted by using the Keras Library with Python programming language to implement the feature reduction algorithm, intrusion detection and the model pruning in a CPU which is Intel(R) Core(TM) i7-7800X CPU @ 3.50GHz with 64 GB RAM and GTX2080Ti. In addition, the full dataset has been considered for performing training and testing processes in the ratio of 70% and 30%, respectively.

Performance Evaluation. First, we use different dimension reduction methods to evaluate the performance of the proposed model including PCA, random forest (RF), and Autoencoder. In addition, the feature reduction methods were used to reduce the original data to different feature dimensions, specifically 36 dimensions and 64 dimensions in our experiment. The model uses training data to train and verifies the training results on the test set. Precision, Recall, F1 Score, and FPR are used to measure the performance of the model. The final experimental results are listed in Table 2.

Table 2. Comparison of model performance among different dimension reduction methods

Technique	Precision	Recall	F1 Score	FPR
PCA (36)	99.2%	94.4%	98.0%	0.06%
PCA (64)	99.6%	97.4%	97.5%	0.03%
RF (36)	99.2%	95.3%	95.7%	0.05%
RF (64)	99.5%	96.7%	96.5%	0.03%
Autoencoder (36)	99.2%	98.3%	96.6%	0.05%
Autoencoder (64)	98.8%	97.9%	90.0%	0.08%

The experimental results show that the proposed model efficiently detects intrusion data by dimensionality reduction. Precision, Recall, F1 Score, and FPR can reach 98.8%, 87.9%, 90.0%, and 0.08%. Based on the experimental results of three feature reduction methods for intrusion detection, PCA showed the best performance, achieving an F1 score of 98.0% and the False Positive Rate of 0.03%. Random Forest performed relatively well, while Autoencoder had the lowest performance. This result indicates that the low-dimension feature dataset after PCA realizes redundancy removal in network traffic and obtains better detection results. PCA tends to preserve the discriminative features that contribute significantly to separating normal and intrusive instances, leading to high performance. Although random forest may not provide the same level of dimension reduction as PCA, its ability to handle nonlinear relationships and feature interactions could still lead to reasonably good performance in intrusion detection. Autoencoder might have failed to capture the relevant information and discard the noise adequately, resulting in a less discriminative feature representation for intrusion detection. Additionally, the complexity of the Autoencoder model might not have been suitable for this dataset, leading to suboptimal results.

From the above results, it can be observed that as the data dimensionality decreases, the model performance gradually declines. We selected the PCA dimensionality reduction method, which exhibited the best performance, to examine the specific impact of different dimensions on model performance. The detailed results are shown in the following Table 3.

Table 3. Comparison of model performance based on PCA with different dimensions

Methods	Precision	Recall	F1 Score	FPR
PCA (64)	99.6%	97.4%	97.5%	0.03%
PCA (49)	99.4%	97.4%	97.5%	0.03%
PCA (36)	99.2%	94.4%	98.0%	0.06%
PCA (25)	98.6%	93.4%	92.5%	0.08%

From the results, it can be observed that as the dimensionality decreases, PCA discards some of the original data's information, which may result in the model being unable to fully capture the complexity and variability of the data. Therefore, higher dimensions such as 36 and 49 dimensions often better retain important features that contribute to model classification and recognition. Lower dimensions such as 25 dimensions may oversimplify the model and fail to adapt to the complexity of the data. This can lead to underfitting or inaccurate predictions when dealing with new samples. Among the given results, reducing the dimensionality to 49 dimensions shows

relatively better model performance with higher Precision, Recall, and F1 Score, while maintaining a lower FPR. Therefore, based on the provided dimension selections, reducing to 49 dimensions may be the optimal choice.

The Fig. 6 illustrates the detection performance of the model against different types of attacks. Overall, the model demonstrates strong performance in detecting various types of attacks, such as BENIGN, DDoS, and DoS attacks etc. However, there are some specific attack types, such as Bot, Web Attack Brute Force, where the model's performance is relatively lower. The lower detection performance can be attributed to insufficient training data, so it limits the expressive capability of features. Moreover, complex or evolving attack patterns can have diverse and evolving patterns, making them more challenging to detect accurately.

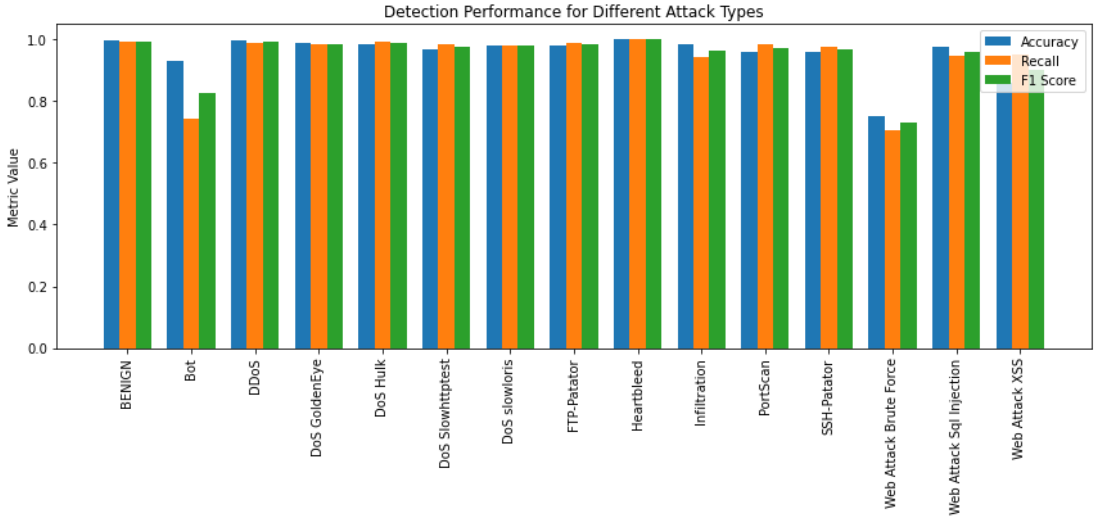

Fig. 6. The detection performance on different types of attacks

Finally, we compared the impact of model pruning on the performance of intrusion detection models. We mainly used different pruning thresholds to evaluate the performance changes of the models. The specific results are shown in the graph. Additionally, in assessing the performance metrics of the models, we included training time and model size, which are important for future deployment.

Table 4. Comparison of model performance based on PCA with different thresholds

Thresholds	Precision	Recall	F1 Score	FPR	Time (s)
10%	98.7%	92.5%	92.7%	0.03%	376.2
20%	98.9%	94.3%	95.2%	0.05%	352.0
30%	98.7%	94.7%	96.5%	0.07%	340.1
40%	99.1%	95.2%	96.7%	0.06%	355.3
50%	99.1%	94.9%	95.0%	0.08%	352.9
60%	99.1%	92.8%	91.5%	0.06%	352.4
70%	99.1%	93.5%	92.3%	0.06%	354.6
80%	99.1%	95.0%	95.1%	0.06%	352.8
90%	99.0%	93.2%	91.0%	0.06%	353.3

As the pruning threshold decreases, indicating higher compression (e.g., from 90% to 10%), there is a trend of decreasing precision, recall, and F1 score. The experimental results of the comparison are shown in Table 4 above. This suggests that aggressive model pruning negatively affects the model's ability to accurately detect intrusions. As the pruning threshold becomes more stringent, the model discards more weights and connections, leading to a loss of important information and affecting the model's performance. The false positive rate (FPR) remains relatively consistent across different pruning thresholds, with values ranging from 0.03% to 0.09%. This indicates that model pruning does not significantly impact the model's ability to correctly identify benign in-

stances. The inference time decreases as the pruning threshold decreases, indicating that model pruning reduces the computational resources required for intrusion detection. The model size decreases as the pruning threshold decreases, indicating that model pruning effectively reduces the storage space needed for the model.

A balance needs to be achieved between model size, inference time, and performance metrics. In our experiment, the optimal pruning threshold was determined to be 40%. We achieved an F1 score of 96.7% and an FPR of 0.06%. Additionally, the model training time was 355 seconds. At this threshold, the model strikes a balance between performance and model size, making it more suitable for deployment in IoT environments.

5 Conclusion

With the rapid development of electricity trading network, ensuring the security and integrity of the system has become a critical concern. Intrusion detection systems play a crucial role in mitigating these threats, but traditional IDS solutions are inadequate for electricity trading applications due to unique network characteristics and resource limitations. This paper proposes a novel anomaly detection framework that leverages dimension reduction and Convolutional Neural Networks to develop a lightweight intrusion detection model suitable for the limited computing and storage resources in electricity trading network, resulting in efficient and effective threat detection. We proposed a lightweight intrusion detection model with limited computational and resource capabilities based on feature reduction and model pruning. The experimental results indicate that the proposed model not only considerably improves the classification detection performance of the intrusion network traffic but also significantly reduces the classification time, which can satisfy the real-time requirements of the intrusion detection system. In the upcoming work, we can utilize more methods such as model distillation to explore further advancements in model pruning. This is also a promising area of research with great potential.

6 Acknowledgement

This work was supported by the China Souther Power Grid Technological Proiect with the Proiect No. GDKIXM20210107.

References

[1] T. Ahmad, D. Zhang, C. Huang, H. Zhang, N. Dai, Y. Song, H. Chen, Artificial intelligence in sustainable energy industry: Status Quo, challenges and opportunities, Journal of Cleaner Production 289(2021) 125834.
[2] S.S. Ali, B.J. Choi, State-of-the-art artificial intelligence techniques for distributed smart grids: A review, Electronics 9(6)(2020) 1030.
[3] N. Mishra, S. Pandya, Internet of things applications, security challenges, attacks, intrusion detection, and future visions: A systematic review, IEEE Access 9(2021) 59353-59377.
[4] S. Venkatraman, B. Surendiran, Adaptive hybrid intrusion detection system for crowd sourced multimedia internet of things systems, Multimedia Tools and Applications 79(5-6)(2020) 3993-4010.
[5] M. Agiwal, N. Saxena, A. Roy, Towards connected living: 5G enabled internet of things (IoT), IETE Technical Review 36(2)(2019) 190-202.
[6] A.R.H. Hussein, Internet of things (IOT): Research challenges and future applications, International Journal of Advanced Computer Science and Applications 10(6)(2019) 77-82.
[7] N.M. Karie, N.M. Sahri, W. Yang, C. Valli, V.R. Kebande, A review of security standards and frameworks for IoT-based smart environments, IEEE Access 9(2021) 121975-121995.
[8] K. Sha, W. Wei, T.A. Yang, Z. Wang, W. Shi, On security challenges and open issues in Internet of Things, Future generation computer systems 83(2018) 326-337.
[9] N. Naik, Choice of effective messaging protocols for IoT systems: MQTT, CoAP, AMQP and HTTP, in: Proc. 2017 IEEE international systems engineering symposium (ISSE), 2017.
[10] M. Abomhara, G.M. Køien, Cyber security and the internet of things: vulnerabilities, threats, intruders and attacks, Journal of Cyber Security and Mobility 4(1)(2015) 65-88.
[11] D.H.D.L. Iglesia, G.V. González, A.S. Mendes, D.M. Jiménez-Bravo, A.L. Barriuso, Architecture to embed software agents in resource constrained internet of things devices, Sensors 19(1)(2019) 100.
[12] F. Sabahi, A. Movaghar, Intrusion detection: A survey, in: Proc. 2008 Third International Conference on Systems and

Networks Communications, 2008.
[13] M. Hawedi, C. Talhi, H. Boucheneb, Security as a service for public cloud tenants (SaaS), Procedia computer science 130(2018) 1025-1030.
[14] L. Wallgren, S. Raza, T. Voigt, Routing attacks and countermeasures in the RPL-based internet of things, International Journal of Distributed Sensor Networks 9(8)(2013) 794326.
[15] B.B. Zarpelão, R.S. Miani, C.T. Kawakani, S.C. de Alvarenga, A survey of intrusion detection in Internet of Things, Journal of Network and Computer Applications 84(2017) 25-37.
[16] K. Zhao, L. Ge, A survey on the internet of things security, in: Proc. 2013 Ninth international conference on computational intelligence and security, 2013.
[17] A. Patcha, J.M. Park, An overview of anomaly detection techniques: Existing solutions and latest technological trends, Computer networks 51(12)(2007) 3448-3470.
[18] Z. Inayat, A. Gani, N.B. Anuar, S. Anwar, M.K. Khan, Cloud-based intrusion detection and response system: open research issues, and solutions, Arabian Journal for Science and Engineering 42(2)(2017) 399-423.
[19] T.D. Nguyen, S. Marchal, M. Miettinen, H. Fereidooni, N. Asokan, A.R. Sadeghi, DÏoT: A federated self-learning anomaly detection system for IoT, in: Proc. 2019 IEEE 39th International conference on distributed computing systems (ICDCS), 2019.
[20] M.A. Alsoufi, S. Razak, M.M. Siraj, I. Nafea, F.A. Ghaleb, F. Saeed, M. Nasser, Anomaly-based intrusion detection systems in iot using deep learning: A systematic literature review, Applied sciences 11(18)(2021) 8383.
[21] H.H. Pajouh, R. Javidan, R. Khayami, A. Dehghantanha, K.K.R. Choo, A two-layer dimension reduction and two-tier classification model for anomaly-based intrusion detection in IoT backbone networks, IEEE Transactions on Emerging Topics in Computing 7(2)(2019) 314-323.
[22] A. Verma, V. Ranga, Statistical analysis of CIDDS-001 dataset for network intrusion detection systems using distance-based machine learning, Procedia Computer Science 125(2018) 709-716.
[23] H. Liu, B. Lang, Machine learning and deep learning methods for intrusion detection systems: A survey, applied sciences 9(20)(2019) 4396.
[24] M. Li, Application of CART decision tree combined with PCA algorithm in intrusion detection, in: Proc. 2017 8th IEEE international conference on software engineering and service science (ICSESS), 2017.
[25] H. Xu, C. Fang, Q. Cao, C. Fu, L. Yan, S. Wei, Application of a distance-weighted KNN algorithm improved by mothflame optimization in network intrusion detection, in: Proc. 2021 IEEE 4th International Symposium on Wireless Systems within the International Conferences on Intelligent Data Acquisition and Advanced Computing Systems (IDAACS-SWS), 2018.
[26] S. Teng, N. Wu, H. Zhu, L. Teng, W. Zhang, SVM-DT-based adaptive and collaborative intrusion detection, IEEE/CAA Journal of Automatica Sinica 5(1)(2018) 108-118.
[27] 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.
[28] K. Lakshmanna, R. Kaluri, N. Gundluru, Z.S. Alzamil, D.S. Rajput, A.A. Khan, M.A. Haq, A. Alhussen, A review on deep learning techniques for IoT data, Electronics 11(10)(2022) 1604.
[29] S.P.R.M., P.K.R. Maddikunta, M. Parimala, S. Koppu, T.R. Gadekallu, C.L. Chowdhary, M. Alazab, An effective feature engineering for DNN using hybrid PCA-GWO for intrusion detection in IoMT architecture, Computer Communications 160(2020) 139-149.
[30] S. Manimurugan, S. Al-Mutairi, M.M. Aborokbah, N. Chilamkurti, S. Ganesan, R. Patan, Effective attack detection in internet of medical things smart environment using a deep belief neural network, IEEE Access 8(2020) 77396-77404.
[31] J. Hou, B. Adhikari, J. Cheng, DeepSF: deep convolutional neural network for mapping protein sequences to folds, Bioinformatics 34(8)(2018) 1295-1303.
[32] H. Tian, S.C. Chen, M.L. Shyu, Evolutionary programming based deep learning feature selection and network construction for visual data classification, Information systems frontiers 22(5)(2020) 1053-1066.
[33] K. Wu, Z. Chen, W. Li, A novel intrusion detection model for a massive network using convolutional neural networks, IEEE Access 6(2018) 50850-50859.
[34] Y. Zhou, G. Cheng, S. Jiang, M. Dai, Building an efficient intrusion detection system based on feature selection and ensemble classifier, Computer networks 174(2020) 107247.
[35] Y. Xiao, C. Xing, T. Zhang, Z. Zhao, An intrusion detection model based on feature reduction and convolutional neural networks, IEEE Access 7(2019) 42210-42219.
[36] B. Selvakumar, K. Muneeswaran, Firefly algorithm based feature selection for network intrusion detection, Computers & Security 81(2019) 148-155.
[37] J.Y. Kim, S.B. Cho, Exploiting deep convolutional neural networks for a neural-based learning classifier system, Neurocomputing 354(2019) 61-70.