Network Security Protection Model of Electric Power Information System Based on Hierarchical Weight Pruning Algorithm Optimizing OD-CNN Algorithm

Shengyang Wang

Information Engineering Institute, Nanyang Vocational College of Agriculture, Nanyang,473000, China
E-mail: shengyangwang22@sina.com

Received 14 June 2024; Accepted 30 May 2025

Abstract

The Internet of Things electric power information system is being used more and more frequently as a result of the advancement of IoT technology, and as a result, there is a higher chance that the network will try to eavesdrop on it. This research creates an Internet of Things intrusion detection model using a one-dimensional convolutional neural network and a hierarchical weight pruning technique to increase the protection level of the Internet of Things. The one-dimensional convolutional neural network model had the lowest accuracy and precision rate, and the other three models had only slight variations in recognition accuracy, according to the findings of the performance comparison studies. The recognition accuracy of the designed hybrid method, 1D convolutional neural network, Mobile Net, and Shuffle Net models is 92.6%, 90.7%, 91.2%, and 91.4% when the amount of test data is the entire test set, or 258. The designed hybrid algorithm, one-dimensional

Journal of Cyber Security and Mobility, Vol. 14_3, 673–700.
doi: 10.13052/jcsm2245-1439.1437

convolutional neural network, Mobile Net, and Shuffle Net models' average intrusion detection speeds during the entire experiment were 18.2 ms, 53.6 ms, 24.3 ms, and 29.5 ms, respectively. When the computational samples were the entire test set, their memory consumption was 2893 KB, 18602 KB, 3741 KB, and 4262 KB. However, the computational speed is faster, the model is simpler, and it is suitable to be deployed in scenarios requiring fast feedback to identify the network intrusion detection results. It can be seen that the detection accuracy of the power information system IoT network intrusion detection model designed in this research has a negligible difference compared with the mainstream and novel models in the current market.

Keywords: Layered weight pruning, power information systems, cybersecurity protection, one dimensional convolutional neural network, Internet of things.

1 Introduction

With the rapid development and wide application of the Internet of Things (IoT) technology, the process information system (PIS) in the power industry has gradually realized the integration of multi-device, real-time data interaction and automatic control. This integration significantly improves the efficiency and responsiveness of the system, allowing utilities to manage and dispatch more intelligently [1, 2]. However, the widespread use of the Internet of Things has also exposed many security vulnerabilities faced by PIS. First, the explosion in the number and diversity of types of IoT devices has led to a massive increase in the attack surface, with cyber attackers able to exploit weaknesses in various devices to infiltrate and disrupt. Second, many IoT devices are relatively weak in security protection, lacking the necessary authentication and data encryption measures, which makes it easy for attackers to obtain sensitive information or carry out malicious operations. In addition, due to the high interconnection between devices and systems, a lack of security in any one link can lead to a security risk for the entire system. Therefore, facing such challenges, it is particularly important to build an effective network intrusion detection and defense mechanism to ensure the safe and stable operation of PIS in the Internet of Things environment [3, 4]. The intrusion of PISIoT network is mainly reflected in network attacks, malware invasion, malicious data tampering and so on. Once these situations occur, all of them may have a serious impact on the functions of PIS, so that the power system cannot work normally, and may even cause serious damage

to the power facilities themselves [5]. Therefore, it is of high importance to establish an effective network intrusion detection (ID) and defence system.

In the context of IoT and its application in PIS, intrusion detection systems (IDS) are becoming more and more important. As sensors, smart devices, and control systems become more interconnected, the operational efficiency of power facilities has improved significantly, but at the same time, the probability and complexity of cyber attacks have increased, posing a serious challenge to the security of PIS. As a key technology to proactively monitor and respond to potential threats, IDS can quickly detect and identify abnormal activities in real time, prevent malicious attacks, and reduce the risk of system failures. Through the analysis of traffic and behavior patterns, IDS can not only detect known attacks in time, but also provide warnings for new attacks, so as to ensure the robust operation of PIS in the face of various network security threats. There are two main routes for the current mainstream IoT-ID techniques. The first is ID based on feature matching, which matches predefined rules with pre-defined features to identify threat behaviors [6]. However, this method is less adaptable and difficult to cope with complex and changing network environments and frequently changing attack patterns. The second is ID based on machine learning algorithms, which trains models by using large amounts of historical data to identify potential threat behaviors. The method is somewhat universal, but the quality of the training data, complexity of the model, and poor recognition speed makes the detection effect and accuracy of the model need to be improved. Because of this, this research aims to enhance the Convolutional neural network (CNN), which excels at processing nonlinear data characteristics, and create a compact model for IoT intrusion detection that has high recognition accuracy (PIS-IoT-ID).

The first part of this research is used to analyses the importance of PIS-IoT-ID and the way the technology is implemented, leading to the subsequent content. The second part is used to design the IoT intrusion detection (IoT-ID) model with hybrid One dimensional convolutional neural network (OD-CNN) and Layered weight pruning (LWP) algorithms, which is the innovation of this research. The third step is to use the designed intelligent IoT-ID model, which is also the innovation of this research. The third step is to use the designed intelligent detection model to conduct experiments on intrusion data identification in PISIoT and compare the experimental results with the computed results of common lightweight network ID models. The fourth part is to analyses the results obtained from the experiments and summaries the shortcomings of the study.

2 Related Works

PIS-IoT-ID is a research branch of network ID, which has been widely studied by experts and scholars due to its important role in protecting sensitive data of commercial organizations and maintaining normal operation of business. In order to more effectively identify cyberattacks, Li X et al. suggested a deep learning-based network ID classification model that integrates deep neural networks and long-short-term memory algorithms. Experiments show that the rate of this model on the dataset are better than the traditional long and short-term memory algorithm model. Moreover, the ID algorithm proposed in this study has a shorter classification test time and higher ID accuracy and recall compared to other algorithms such as LeNet. This study provides new perspectives and methods for network ID [7]. The research team led by Jiang K discovered that because of complex and time-varying network settings, network intrusion data are frequently buried behind a large number of normal samples, leading to a lack of model training examples and detection results with a high percentage of false alarms. Therefore, a network ID algorithm combining hybrid sampling and deep-level networks is proposed in the study. The performance of this network ID algorithm is validated by two network intrusion datasets, and the test results show that the algorithm achieves a classification accuracy of 83.58, due to the comparison model [8]. Traditional network ID techniques, according to Zhang H et al., struggle to handle high-speed network data and are incapable of spotting as-yet-unknown threats. According to the experimental findings, the suggested method's real-time detection efficiency is greater than that of conventional machine learning techniques [9]. Kim T et al. proposed a packet machine learning model based on delayed decision making and on the basis of this model a single packet integrated classifier with cumulative features. The authors then combined the use of a hybrid classifier based on packet characteristics and session-based characteristics to develop a new machine learning-based network ID system. Experimental results showed that the method was able to achieve very high detection accuracy unlike traditional methods [10]. The generalized average grey wolf method and Elastic Net shrinkage autoencoder were used to identify features for the network ID system that Moizuddin et al. proposed. two popular public network intrusion datasets were used for testing. This study provides new perspectives and methods for network ID, which will be an important reference for future research [11]. Alhajjar et al. explored the performance of Particle Swarm Optimization, Genetic Algorithms, and Deep Learning for network ID, and the authors applied them to two publicly

available datasets, and the results showed that machine learning based detection models are poor at detecting adversarial perturbation type intrusion problems [12].

Banitaba FS et al. proposed stochastic computing (SC) as a new mechanism to enhance model reliability and resist adversarial attacks in order to solve the security problem of neural networks (NN). The main goal of the study was to evaluate the effectiveness of SC in mitigating the negative impact of attacks on NN results. Through a series of rigorous experiments and evaluations, the resilience of neural networks using SC under adversarial attacks is explored. The research results show that SC can introduce an effective defense layer and significantly reduce the vulnerability of the network to the change of results under attack [13]. Nouraniboosjin S et al. address the privacy and security of medical data by proposing a new protocol that combines blockchain technology and cryptographic principles to eliminate reliance on a single trusted third party and provide efficient privacy-protecting access control mechanisms. The security of searchable encryption schemes is enhanced by combining blockchain technology with other cryptographic principles to ensure that patient privacy is protected and by eliminating the need for trusted third parties through blockchain technology [14]. Larijani A. et al. proposed A comprehensive privacy protection solution for smart meter data privacy to enhance the security of smart grid data. With a focus on confidentiality, integrity and anonymity of messages, the aim is to develop a dynamic pricing platform to coordinate supply and demand and improve facility efficiency. Compared with previous studies, this study solves the problems of ineffective customer authentication and high computing cost in the operation center, and adopts the methods of mutual authentication and key negotiation to improve the system efficiency and reduce the time complexity. The solution enables power companies to send control commands to smart meters more quickly, reducing communication load and computational complexity, and enhancing the system's ability to withstand various types of attacks [15].

To sum up, although the current research on PIS-IoT-ID has made some progress, there are still some significant shortcomings. First of all, many existing network intrusion detection algorithms are not efficient and accurate when processing high-speed real-time data, and it is difficult to adapt to complex and changeable attack patterns, resulting in serious false positives and missed positives. Secondly, the existing model is usually unable to achieve good comprehensive performance in the face of multiple intrusions, especially in the aspect of feature selection and parameter adjustment, it

still relies too much on manual process and lacks automation and intelligent support. In addition, most of the current research focuses on the optimization of a single algorithm, which fails to effectively integrate multiple technical means to achieve comprehensive protection. Therefore, by combining LWP algorithm and OD-CNN, this study aims to build a lightweight and efficient intrusion detection model to improve the real-time identification capability of power information system security threats in the Internet of Things environment, so as to better protect the stable operation of the system. Through the feature extraction capability of the proposed method and the advantages of deep learning, the identification accuracy of various network attacks is improved. Through the implementation of LWP algorithm, on the basis of ensuring the model detection performance, the redundant parameters in the neural network can be effectively reduced, the lightweight degree of the model can be improved, and the computing speed can be accelerated to adapt to the resource-limited Internet of Things environment.

3 PIS-IoT-ID Model Based on Improved OD-CNN Algorithm

The PIS-IoT-ID task requires high computational speed for the model, so the detection model needs to achieve a lightweight setup [16, 17]. Here, OD-CNN is used to construct the basic PIS-IoT-ID structure, and LWP algorithm is used to construct the network layer of OD-CNN correspondingly, and finally the complete ID model is constructed. The purpose of this research is to significantly compress the algorithm computation and volume without affecting the algorithm's detection performance, so that the model has the ability to quickly detect IoT intrusion behaviours.

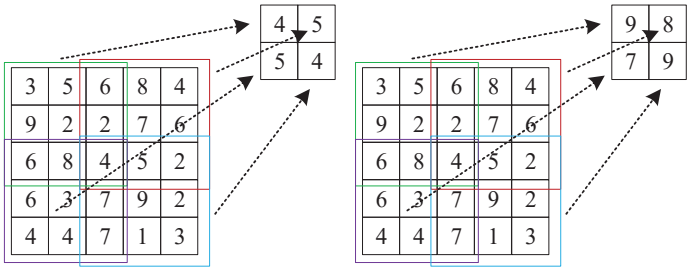
3.1 OD-CNN Algorithm Design for IoT-IDs

The reason for choosing OD-CNN to build the network ID model in this research is that this type of algorithm is friendlier to low computing power hardware devices. Compared to traditional 2D CNN algorithms, OD-CNN has a higher computational efficiency ratio, low memory footprint and simpler maintenance [18]. Now the first design to build the PIS-IoT-ID structure of the OD-CNN, the core structure in the OD-CNN is the convolutional unit, the various parameters of the convolutional unit can be debugged through a number of experiments to determine the optimal solution. In addition, the mode of manual debugging of this parameter will bring more potential errors to the algorithm, and here it is more reasonable to use Equation (1) to

calculate the input dimension dim_o of the convolutional unit.

$$dim_o = 1 + (dim_{in} + 2size_{padding} - size_{kernel})/s \tag{1}$$

$size_{padding}$ and $size_{kernel}$ in Equation (1) correspond to the specific size of the padding and the size of the convolution kernel, respectively. The contribution of Equation (1) to the input dimension of the convolution unit in optimizing OD-CNN is mainly reflected in its precise calculation of the dimension of the input feature map in the convolution operation, thereby effectively guiding the design and parameter adjustment of the convolution kernel. Specifically, this equation takes into account the size of the convolution kernel, the stride length, and the specific configuration of the filling operation to ensure a reasonable reduction or maintenance of the data dimension during the convolution processing. This process can retain key information to the greatest extent while reducing computational complexity, thus avoiding the waste of computing resources caused by excessive dimensions. The most common pooling structures in CNN networks are generally divided into average pooling, which uses the average of all values in the window as the output value, and maximum pooling, which uses the maximum value within the window as the output value, as shown in Figure 1. Average pooling and maximum pooling have different roles in the feature extraction process. Average pooling retains the global information of the region by calculating the average of all values within the local region, which makes it more suitable for capturing the overall features while reducing the influence of noise. In contrast, Max pooling focuses on extracting the most significant features within a local region. By retaining only the maximum value, it enhances the sensitivity to important features, thereby helping to improve the model's recognition ability and robustness for key features.



(a) Average pooling (b) Max pooling

Figure 1 Shows the calculation principles of average pooling and maximum pooling in CNN.

Considering the large dimensionality of the input data for the IoT-ID model, average pooling needs to be inserted into the front structure of the OD-CNN to reduce the amount of computation, and maximum pooling needs to be added to the back structure of the OD-CNN to further extract key features. In IoT-IDs, the applicability of OD-CNN to low-power devices is crucial because its lightweight design ensures efficient operation in resource-limited environments and reduces the demand for computing power and electricity. The OD-CNN model also needs to have a fully connected layer added at the end because it can globally map the previously extracted features to the same feature space, which has a significant impact on the output quality of the final classification result. The fully connected layer plays a key role in improving the overall performance and classification accuracy of OD-CNN. It is responsible for integrating the features extracted through convolution and pooling layers, not only providing a global view of the output of the previous layers, but also enhancing the nonlinear representation ability of the model. By further processing these extracted features, the fully connected layer can effectively identify complex input patterns, thereby improving the classification accuracy of the model. As can be observed, the convolutional layer, pooling layer, spreading layer, and fully connected layer should all be included in the proposed OD-CNN algorithm [19]. The main function of the convolutional layer is to learn and extract features from the original data; the pooling layer serves to retain the main features while reducing the dimensionality of the data and parameters to prevent overfitting; the spreading layer is responsible for transforming the outputs of the previous layer into a one-dimensional variable. The more convolutional layers there are, the more complex the features the network can learn. Shallow networks learn local and basic information, while deeper networks learn broader and more advanced information. In order to extract more low-dimensional information, the shallow convolutional layer uses a large number of small-scale convolutional kernels. Two pooling layers are introduced after the convolutional layer due to the small amount of data and susceptibility to overfitting. Combining the above and the lightweight design goal, the network structure of OD-CNN is designed, as shown in Table 1.

The specific components of the OD-CNN network are designed in detail in the next step, and the loss function $loss$ of the algorithm is constructed in a two-paradigm form, as shown in Equation (2).

$$loss = \sum_{n=1}^{N} \|Y_n - \Gamma(X_n)\|^2 \tag{2}$$

Table 1 OD-CNN network structure for power information system IoT intrusion detection

Layer Number	Hierarchy Type	Layer Name	Number of Neurons	Neuron Size	Output Data Size
*1	Convolutional layer	Conv_1	64	8	256,64
*2	Convolutional layer	Conv_2	64	8	256,64
*3	Average pooling layer	Avg_pool1	1	4	128,64
*4	Convolutional layer	Conv_3	64	3	128,64
*5	Maximum pooling layer	Max_pool1	1	3	64,64
*6	Flattening layer	Fla_1	/	/	1,1024
*7	Fully connected layer	FC_1	1	1024	1,1

$\Gamma(X_n)$ in Equation (2) describes the output of the neural network. For the purpose of preventing neural network overfitting, the OD-CNN network needs to be accompanied by a random discard module after each convolutional layer. Let the probability that each neuron stops working be p, and the neuron i is associated with Equations (3) and (4) before the loss of computation.

$$z_i^{(l+1)} = b_i^{(l+1)} + w_i^{(l+1)} y^l \tag{3}$$

In Equation (3) $z_i^{(l+1)}$ is the total input corresponding to the ith neuron in the $z_i^{(l+1)}$th layer, y^l is the output of the lth layer connected to the corresponding neuron, $w_i^{(l+1)}$ is the neuron weight coefficient, and $b_i^{(l+1)}$ is the bias coefficient of the corresponding neuron.

$$y_i^{(l+1)} = f(z_i^{(l+1)}) \tag{4}$$

$y_i^{(l+1)}$ in Equation (4) is the corresponding prediction result of this neuron and $f(\cdot)$ represents the activation function. When random discarding is performed according to the discard probability p, the output of the neuron is completed by Equations (5) and (6).

$$z_i^{(l+1)} = b_i^{(l+1)} + w_i^{(l+1)} \tilde{y}^{(l)} \tag{5}$$

$\tilde{y}^{(l)}$ in Equation (5) is the neuron after the completion of random discarding in layer l.

$$y_i^{(l+1)} = f(z_i^{(l+1)}) \tag{6}$$

For the purpose of preventing the neural network from overfitting, the algorithm also needs to add a regularization term, and the commonly

used regularization methods are one-paradigm regularization L_{re1} and two-paradigm regularization, which are computed as shown in Equations (7) and (8), respectively.

$$L_{re1} = L + \lambda\|\theta\| \tag{7}$$

Equation (7) in which θ is the parameter to be optimized, λ is the weight attenuation coefficient and λ loss function.

$$L_{re2} = L + \lambda\|\theta\|^2 \tag{8}$$

The objective function is set to be λ. The optimized optimal network parameters θ^* need to be such that Equation (9) holds.

$$\theta^* = \arg\min_{\theta} L \tag{9}$$

Here the network is optimized using the gradient descent method, where the computation involves first deriving λ to obtain the gradient and then updating the model using the gradient until the model finishes converging, see Equation (10).

$$\theta_{j+1} = \theta_j + \frac{lr \cdot \partial L}{\partial \theta_j} \tag{10}$$

Where θ_j is the model parameters at iteration j, $\frac{\partial L}{\partial \theta_j}$ is the calculated gradient, and lr is the learning rate of the neural network.

3.2 ID Model for OD-CNN with Hybrid LWP Algorithm

The LWP method is added into the OD-CNN algorithm in order to increase the computational effectiveness of the programmed even more. The lightweight design of OD-CNN ensures effective feature extraction while maintaining computational efficiency by optimizing the network structure and reducing the number of parameters. This design concept, by using small-scale convolutional kernels and a reasonable hierarchical architecture, not only reduces the consumption of computing resources, but also enables rapid screening and retention of important local features. Pruning is a light weighting algorithm, which cuts out the weights that do not satisfy the criteria by setting a measure so that they are no longer involved in the subsequent computation and training. The pruning rate, which measures the model's level of light weighting and pruning strength, is calculated as the ratio of the number of weights that were pruned to the total weights. Now according to the pruning rate for weight screening, that is, for the network layer that

will be pruned, each time the pruning step, according to the absolute value of the weights of the size of the order of the smallest s $\times$ weights of the weights to 0, that is, these weights are pruned, and no longer participate in the subsequent computation and training, where weights represent the number of the total number of the weights in the network layer. Weight ranking based on absolute value can effectively identify redundant weights that contribute less to the overall model performance by evaluating the absolute value of each weight. This strategy ensures that weights with smaller absolute values are preferentially removed during the pruning process, thereby maximizing the retention of high-impact parameters that are crucial to learning ability and classification performance.

Typically, the pruning operation starts from 0 and gradually increases the pruning rate, which has the advantage of reserving time for model repair, as pruning weights can reduce the network learning ability and affect network performance. Multiple small-volume pruning allows time to be set aside between pruning's for the network to retrain, adjust the remaining weights, and restore the network effect. Small-scale pruning, in terms of retraining and restoring network performance, gradually removes redundant weights and gives the model sufficient time to adjust and optimize the remaining parameters. This can effectively prevent the performance drop that may be caused by one-time large-scale pruning. The retraining process carried out after each small-scale pruning enables the network to relearn and adjust the relationships after pruning, thereby maintaining the expressive ability and classification performance of the model.

The pruning operation in this study is non-linearly varying, which is inconsistent with conventional processing. The nonlinear change pruning strategy gradually optimizes the pruning rate, enabling the network to gradually reduce redundant parameters while maintaining the learning ability, thereby reducing the computational cost. This strategy allows for more aggressive pruning in the initial stage to eliminate a large number of unnecessary weights. In the later stage, as the learning ability of the network improves, the pruning speed is automatically slowed down to fully utilize and optimize the remaining weights. The initial training step in the algorithm is t_0, the initial pruning rate is s_i, and after every Δt training steps, a pruning operation is carried out and updated. In the process of completing the n round of pruning, the pruning rate of the current round will gradually grow from s_i to the end of the branch after the pruning rate gradually increases from the initial value to the target sparse value of s_f. Specifically, in the t step of pruning, the current pruning rate is calculated according to

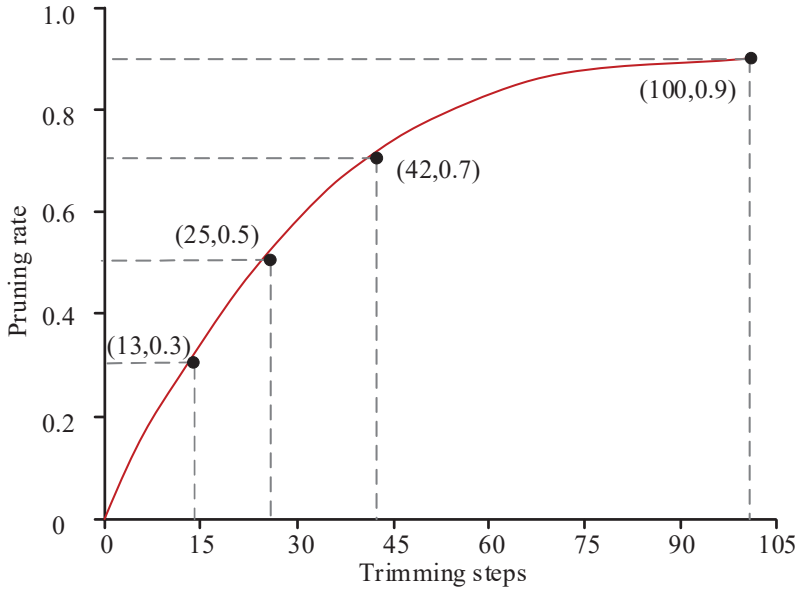


Figure 2 Curve of the pruning rate change function with the number of pruning steps.

Equation (11).

$$s_t = f(t) = (-s_f + s_i)(1 + (t_0 - t)/(n\Delta t))^3 + s_f \tag{11}$$

t in Equation (11) belongs to the set of $\{t_0, t_0 + \Delta t, \ldots, t_0 + n\Delta t\}$, and $f(t)$ represents the pruning rate change function. The function curve of $f(t)$ is shown in Figure 2. In Figure 2, the pre-pruning stage, the overall growth of pruning rate is rapid, but the slope gradually decreases, and in the late pruning stage, the pruning rate gradually converges to a certain value. That is to say, in the initial stage, there are a lot of redundant weights in the network, which can prune the network more drastically, and at this time there are a lot of network parameters, and the algorithm performance recovers quickly. However, after that the remaining weights become less and less, and at this time the number of weights per pruning needs to be reduced.

The research focuses on the differences in importance among different network layers, and independently defines the pruning rate parameter for each layer. Let the set of network layers be $\theta = \{\mathrm{Conv}_1\ \mathrm{Conv}_2\ \mathrm{Conv}_3\ \mathrm{FC}_1\}$, where Conv is the convolutional layer and FC is the fully connected layer. The initial pruning rate and the target pruning rate of each layer are $\mathrm{s}^l_{\mathrm{initial}}$ and $\mathrm{s}^l_{\mathrm{final}}$ respectively. The pruning rate of the l layer at step t is shown in

Equation (12).

$$\mathrm{s}^l(t) = f^l(t) = s^l_{\text{initial}} + (s^l_{\text{final}} - s^l_{\text{initial}}) \cdot \left(\frac{t}{T}\right)^{\alpha} \tag{12}$$

In Equation (12), T represents the total number of pruning steps, and α represents the parameter of the growth rate of the pruning rate. For each layer l, sort according to the absolute value of the weight to determine the pruning weight. Let the weight matrix of layer l be $\mathbf{W}^l \in \mathbb{R}^{n\times m}$, and flatten it into a one-dimensional vector $\mathbf{w}^l = \mathrm{vec}(\mathbf{W}^l)$. The importance scores of each weight are calculated as shown in Equation (13).

$$\mathrm{Score}(w^l_i) = |w^l_i| \tag{13}$$

In Equation (13), w^l_i represents the i-th weight value in the l-th layer. After sorting in ascending order of scores, the weights with the lowest scores are selected for pruning and pruning, and the remaining weights are retrained to restore performance. The sparsity regularization term is introduced into the loss function, as shown in Equation (15).

$$\theta_{total} = \theta_{CE} + \lambda \sum_{l\in\theta} \sum_{i} |w^l_i| \tag{14}$$

In Equation (14), θ_{CE} represents the cross-entropy loss and λ represents the sparsity coefficient. During retraining, freeze the pruned weights (keep them at zero) and only update the remaining weights.

For convolutional layers, deeper layers tend to assume a more important role. Therefore, when performing a pruning operation, more weights of the deeper convolutional layers need to be retained compared to the shallower ones. Whereas, fully connected layers have a huge number of parameters and one can try to use a larger pruning rate. For pooling and spreading layers, they do not contain parameters, so they are not processed. Based on the above considerations, a LWP algorithm is now proposed to lighten the one-dimensional CNN with the following pruning configurations for each layer. Firstly, for the shallow depth Conv_1 and Conv_2 layers, in order to ensure that as much information as possible is extracted, a large number of small-size convolutional kernels are used, and higher starting and target pruning rates are set for these two network layers, which are 0.5 and 0.8, respectively. The initial and target pruning rates are set relatively high at the Conv_1 and Conv_2 layers, which may lead to the loss of key information and thereby affect the learning performance of the model. These layers are mainly

responsible for extracting the basic local features. Excessive pruning may weaken their sensitivity to important signals, thereby resulting in a reduction in the richness and diversity of the extracted features. Since shallow networks are mainly responsible for learning the basic patterns in images or data, such as edges and textures, these local features usually change within a small range. Therefore, small-scale convolutional kernels can accurately capture these changes and have stronger universality at the same time. As for the deep Conv_3 layer, this layer is used to integrate low-dimensional features and extract higher-level information. Therefore, the convolution kernel size of this layer is larger and more important. Therefore, the convolution kernel size of this layer is larger and more important. In order to ensure the reliability of high-dimensional information, in the pruning process of Conv_3 layer, the pruning intensity should be set at a weaker level, and the starting pruning rate and the target pruning rate are set to 0.1 and 0.5, respectively. Finally, since the output data of Fla_1 layer is not all valid, it is easy to have more unnecessary parameters. Therefore, the fully connected layer FC_1 should be set with the highest pruning rate, and its starting pruning rate is set to 0.6 and 0.9 respectively.

Now OD-CNN with hybrid LWP algorithm is used as the core to construct the PIS-IoT-ID model, which consists of dataset loading, feature selection, data pre-processing, model pre-training, model lightening, and ID module as shown in Figure 3.

The training set and test set are divided up, and the data integrity is checked by the dataset loading module. In the feature selection module, a genetic algorithm is used to automatically pick the sliced data in order to

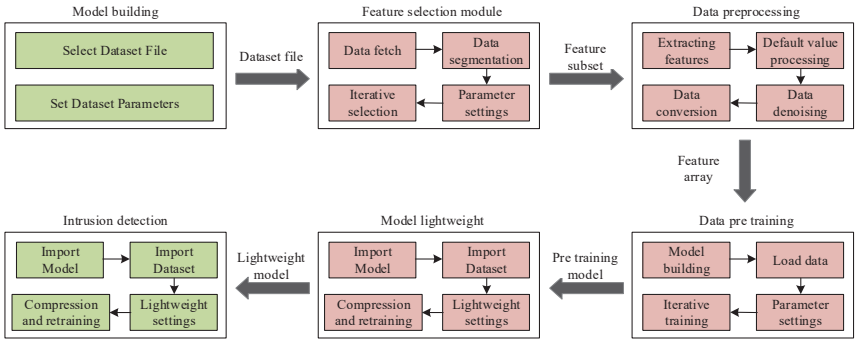


Figure 3 Structure of IoT intrusion detection model for power information system.

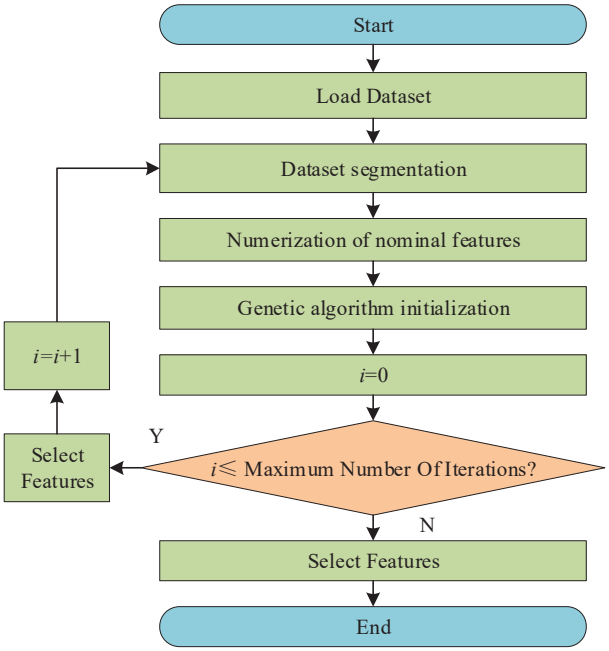


Figure 4 Data Feature selection process based on genetic algorithm.

prevent the subjective inaccuracy brought on by manual feature selection. The calculating procedure is illustrated in Figure 4.

The main tasks of the data preprocessing module are data extraction, data de-weighting, data de-noising, and numerical normalization, so that the subsequent detection model can better identify the input data. As shown in Figure 5, firstly, the pre-training model loading operation is required, and the user needs to choose one of the recent models and the best-performing model to load, and then enter into the data light weighting stage, which is mainly completed by the proposed LWP algorithm. The outputs of the model lightening section are the parameter file of the lightened model, the model structure data and the lightening process log. The LWP method significantly reduces the computational complexity and memory usage of the OD-CNN algorithm by identifying and removing redundant weights that contribute less to the model performance. This method allows the model to maintain a high accuracy rate while reducing the actual computing requirements, enabling the network to operate more efficiently in a resource-constrained environment.

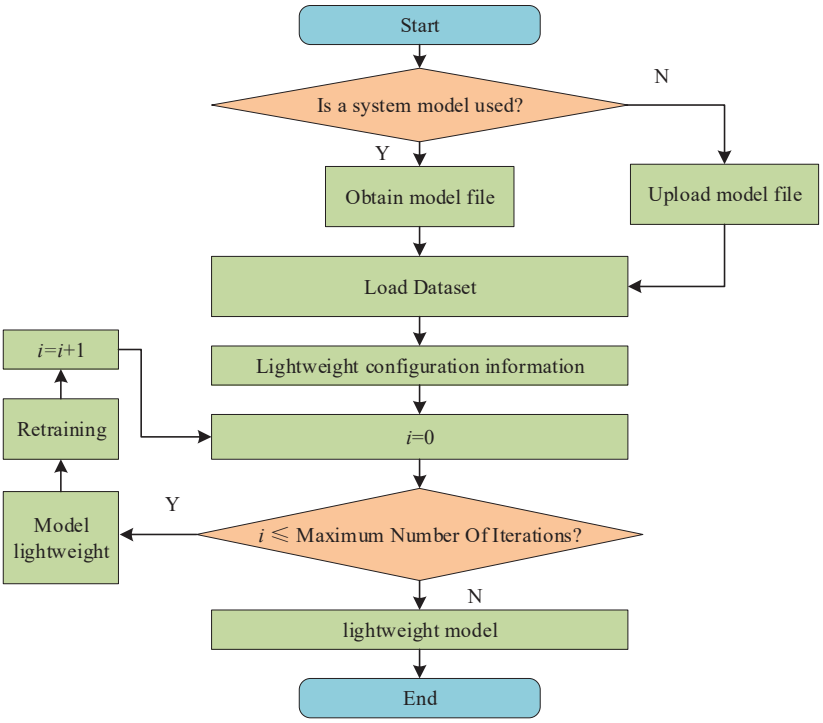


Figure 5 Business process of the lightweight section of the model.

The ID module firstly needs to load the lightweight model as well as the preprocessed dataset, then run the model and finally output the ID binary classification judgement result on the dataset, i.e., to judge whether each sample data in the PISIoT is intrusion attack data or not. Finally, the detection accuracy evaluation metrics used in this research are selected, which are Accuracy (ACC), Precision (PRE), Recall (REC), and F-Measure (F1). ACC is calculated according to Equation (15).

$$ACC = \frac{TP + TN}{TP + TN + FP + FN} \tag{15}$$

In Equation (15), TP and TN are the number of cases that are accurately judged as positive cases and accurately judged as negative cases, respectively, and FP and FN are the number of cases. The PRE calculation method is shown in Equation (16).

$$PRE = \frac{TP}{TP + FP} \tag{16}$$

Similarly, the REC calculation is shown in Equation (17).

$$REC = \frac{TP}{TP + FN} \tag{17}$$

The F1 indicator is a mixture of both PRE and REC and is calculated in Equation (18).

$$PRE = \frac{2PRE \cdot REC}{PRE + REC} \tag{18}$$

4 PIS-IoT-ID Model Performance Testing

Now an experiment is designed to validate the performance of this designed PIS-IoT-ID model, the model is provided with Python language implementation. The model is deployed in the following environment: the hardware platform is a personal desktop computer, the operating system is Win10 Professional, the memory is 32GB, and the central processor is Intel Core i7. The data set used in the study comes from the actual operating environment of various power information systems in China, covering a variety of intrusion types and normal behavior samples to ensure the validity and adaptability of the research model. The dataset contains 862 samples, including normal operation data and samples of multiple types of cyber attacks, such as denial-of-service attacks, active attacks, passive attacks, and vulnerability attacks. These attack samples simulate various threats that can occur in the real world and embody common attack patterns in the iot environment. Each sample is composed of multiple features, including network traffic characteristics, device status indicators and communication protocol content. These features reflect the behavior of the system in normal and abnormal states, and can provide rich information for the subsequent model training. In addition, the feature screening process uses an automated genetic algorithm to select the most influential features for intrusion detection, thereby improving the performance of the model. In terms of labeling, each sample in the dataset was reviewed and labeled by a professional, and the labeling included classified information on attack types and normal behavior. This labeling method ensures the accuracy and reliability of samples, and enables the model to correctly learn the characteristics of different attack behaviors in the training process. The data set is split into a test set and training set at a 3:7 ratio. The high-performance and lightweight Mobile Net, Shuffle Net, and the same structure OD-CNN neural network without LWP algorithm are selected to construct the ID model with the same architecture. The hyperparameters of

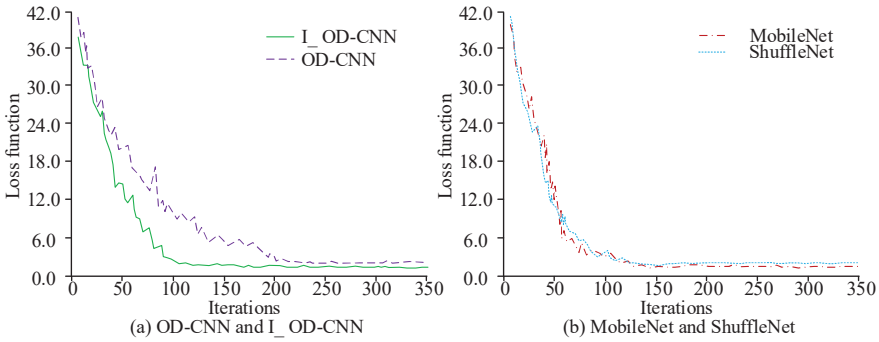


Figure 6 Change of Loss function of each model in training stage.

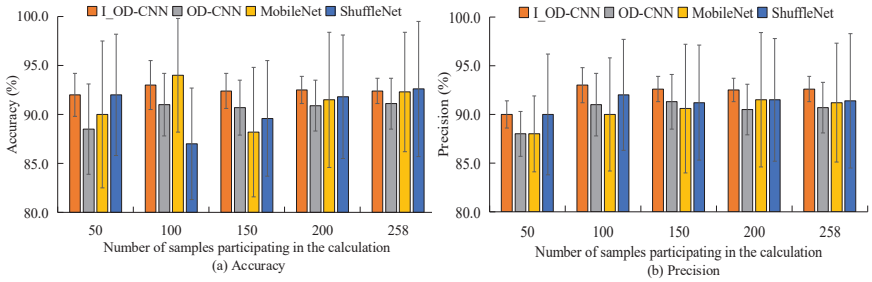


Figure 7 Accuracy and variation of accuracy of each model during the testing phase.

each algorithm are determined according to multiple debugging to take the optimal solution.

The horizontal and vertical axes in Figure 6 represent the number of iterations and the loss function, respectively. I_OD-CNN represents the algorithm model designed in this time in Figure 6. In Figure 6, I_OD-CNN converges significantly faster compared to OD-CNN before improvement, and there is no obvious difference in its convergence speed compared to the other two algorithms. Specifically, the loss function values of each algorithm after convergence are all at about 1.5, and all models except the OD-CNN model finish convergence when the number of iterations is at about 140.

Figure 7 displays the accuracy and precision information for each model on the test set following the training. Different subplots in Figure 7 are used to distinguish different metrics. Figure 7 demonstrates that while the recognition accuracy of the other two models hardly differs from that of the I_OD-CNN model, the overall accuracy and precision of the OD-CNN model is much lower than that of the I_OD-CNN. For example, when the amount of test data

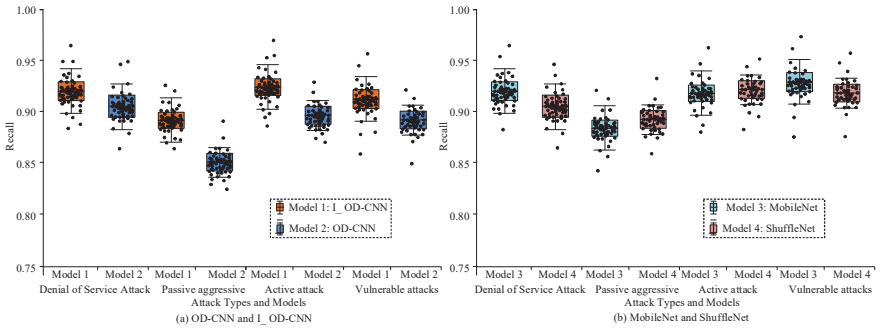


Figure 8 Recall rate changes of each model during the testing phase.

is the whole test set, i.e., 258, the recognition accuracies of I_OD-CNN, OD-CNN, Mobile Net, and Shuffle Net models are 92.6%, 90.7%, 91.2%, and 91.4%, respectively. Meanwhile, the accuracy of I_OD-CNN model reached 92.6%, which not only shows its effectiveness and reliability in the task of intrusion identification, but also emphasizes the importance of the model in practical application scenarios. The high accuracy means that the model can effectively identify most of the potential network attacks and reduce the false positive rate, thus ensuring the security and stability of the power information system.

The statistical recall metrics are displayed in Figure 8 after which you may compare the recognition precision of each model on various IoT attack types. The horizontal axis in Figure 8 shows various common types of IoT network attack problems, which are also present in the dataset, and the horizontal axis shows the recall rate. Observation of Figure 8 shows that the recognition recall of the OD-CNN model on various types of intrusion attack problems is lower than the rest of the models, and the recall of the remaining three models on various types of attack problems varies slightly, with values roughly around 92%.

Then, as shown in Figure 9, contrast the Receiver operating characteristic curves (ROC) of each model's Area Under Curve (AUC). In Figure 9, different line types stand in for various IoT-ID models, and the horizontal and vertical axes show the false positive rate and the true positive rate, respectively. Figure 9 shows that the AUCs of the I_OD-CNN, OD-CNN, Mobile Net, and Shuffle Net models are 0.69, 0.57, 068, and 0.70, respectively, and that the OD-CNN model has the lowest AUC. The difference between the AUCs of the OD-CNN, Mobile Net, and Shuffle Net models is also very small.

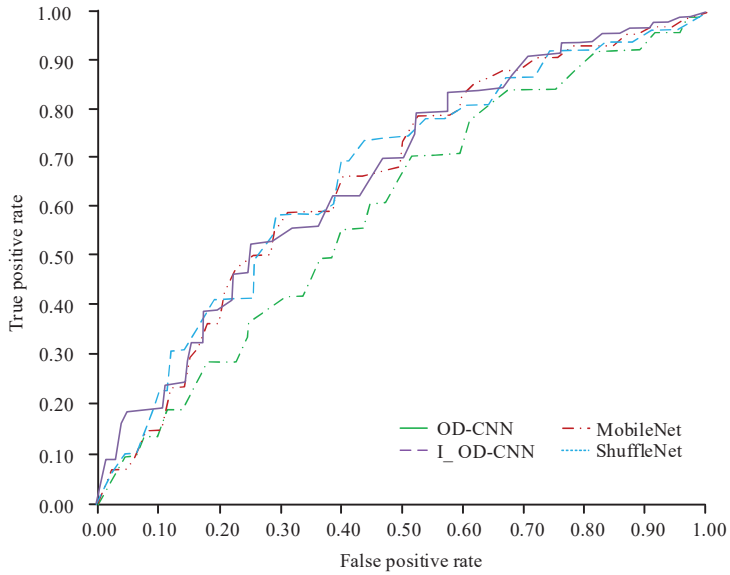


Figure 9 Comparison of ROC and AUC of various models during the testing phase.

Then the computational speed and the lightweight level of each model are compared, and the statistical results of the computational speed are shown in Figure 10. In Figure 10, the horizontal and vertical axes represent the sample size to be computed as well as the computational speed, respectively, and the computational speed is in units of units/ms, and the grey dotted line is used to assist in viewing the maximum and minimum computational speeds of the corresponding models. According to Figure 10, the I_OD-CNN model developed for this study has a computational speed that is noticeably faster than the other models, whereas the OD-CNN model has the slowest computational speed among the four IoT-ID models. Specifically, the average ID speeds of the I_OD-CNN, OD-CNN, Mobile Net, and Shuffle Net models are 18.2/ms, 53.6/ms, 24.3/ms, and 29.5/ms, respectively.

Table 2 now compares the amount of computational memory used by each model. Table 2 shows that the I_OD-CNN model has the lowest average memory consumption, followed by the Mobile Net model and Shuffle Net model, and that the OD-CNN model has a much higher computational memory consumption. When the computational sample is the full test set, the memory consumption of I_OD-CNN, OD-CNN, Mobile Net, and Shuffle Net models are 2893 KB, 18602 KB, 3741 KB, and 4262 KB, respectively.

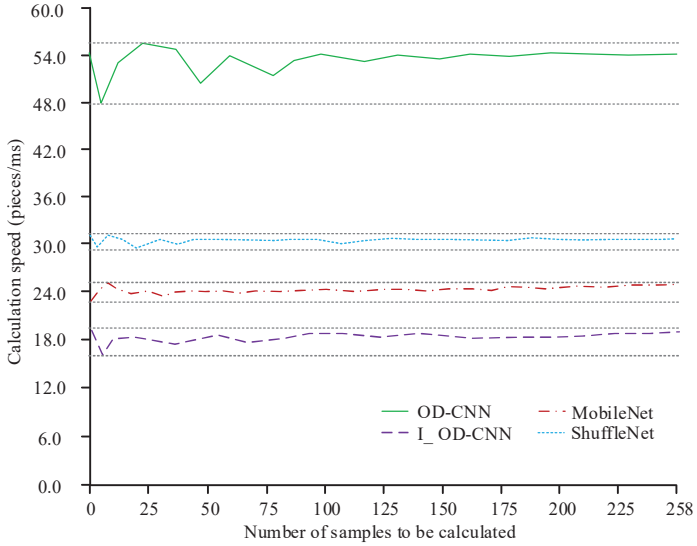


Figure 10 Comparison of calculation speeds among different models during the testing phase.

Table 2 Calculation memory consumption of each model during the testing phase

Sample	Average Memory Consumption (KB)				Memory Consumption Standard Deviation (KB)			
Size	I_OD-CNN	OD-CNN	MobileNet	ShuffleNet	I_OD-CNN	OD-CNN	MobileNet	ShuffleNet
30	322	2038	401	452	7	29	22	25
60	615	4062	809	903	15	58	45	48
90	943	6155	1220	1361	21	99	63	71
120	1264	8164	1641	1840	27	136	81	96
150	1610	10316	2042	2351	31	164	102	120
180	1913	12350	2486	2866	39	194	125	136
210	2236	14421	2905	3269	36	230	149	163
240	2561	16580	3315	3805	45	276	165	205
258	2893	18602	3741	4262	42	307	184	189

To increase the reliability of the test results, another practical test experiment was carried out. A domestic PIS was selected and the system was attacked using the various attack types listed in Figure 8. In order to be as close as possible to the actual network attack situation, the attack model is as follows: every 5 seconds the system has a 50% probability of being attacked, and each experiment lasts for 10,000 seconds. After the experiments were completed, the attack recognition accuracy of each model was counted in Table 3. According to Table 3, the I_OD-CNN model created for this study

Table 3 PIS network security test results

Attack Type	I_OD-CNN	OD-CNN	Mobile Net	Shuffle Net
Denial of Service Attack	97.2%	87.1%	93.6%	95.3%
Passive aggressive	94.8%	83.4%	88.2%	91.5%
Active attack	98.4%	86.2%	91.3%	95.8%
Vulnerable attacks	96.1%	88.5%	94.2%	95.7%

Table 4 Comparison of the advancement of different models

Indicator	I_OD-CNN	OD-CNN	RF	SVM
Accuracy rate of adversarial attack (%)	89.2	75.4	68.5	72.1
Delay (ms)	18.2	53.6	42.8	12.4
Real-time Response (FPS)	54.9	18.7	23.4	80.6
Model Volume (MB)	2.8	17.6	15.2	0.3
Memory usage (KB)	2893	18602	10500	850

has the best overall attack recognition accuracy in a variety of simulated assault situations and has the best capacity to secure the PIS.

The research integrates the proposed I_OD-CNN model into the network security protection system of the actual power information system. The typical power information system adopts a hierarchical distributed architecture, which is specifically divided into the perception layer, edge layer, core layer and communication network. To verify the superiority of I_OD-CNN, random forest (RF), Support vector Machine (SVM), and unpruned OD-CNN were selected as comparison models, and performance tests were conducted under the same experimental environment. The specific comparison indicators include the ability to resist adversarial attacks (the classification accuracy of the model on adversarial samples), latency (the time consumption of a single inference), real-time response capability (the number of frames processed per second), and scalability (the model size and memory usage). The specific comparison results are shown in Table 4.

It can be seen from the results in Table 4 that the I_OD-CNN model is superior to the unpruned OD-CNN, RF and SVM models in terms of the ability to resist adversarial attacks, latency and real-time response capabilities. Specifically, the classification accuracy of I_OD-CNN on adversarial samples reached 89.2%, which was higher than 75.4% of OD-CNN and 68.5% of RF, demonstrating its stronger resistance. Meanwhile, the inference delay of I_OD-CNN is only 18.2 milliseconds, which is much lower than that of OD-CNN and RF, indicating its strong adaptability to real-time scenes. Moreover,

its real-time response capability of 54.9 frames per second is also superior to that of the unpruned OD-CNN. Furthermore, the model volume and memory usage of I_OD-CNN are 2.8 MB and 2893 KB respectively, demonstrating the lightweight advantage of its design and enabling it to operate efficiently in the power information system with limited resources. Overall, these results indicate that the I_OD-CNN model provides an effective and efficient network security protection solution for power information systems.

The above experimental results indicate that the method proposed in the study has certain advantages in dealing with circumvention techniques (such as perturbation-based attacks). This powerful resistance capability may stem from the model's adoption of advanced feature extraction and pruning strategies, which can effectively identify and classify malicious samples even under minor disturbances. In terms of ensuring data security, the key lies in implementing strict data encryption technologies and real-time monitoring mechanisms. Encryption technology can protect sensitive data in power information systems and prevent unauthorized access and data leakage. At the same time, global and regional regulations, such as the General Data Protection Regulation (GDPR), should be followed to ensure the protection of users' privacy rights and data security rights. This includes taking necessary technical and organizational measures in all aspects of data collection, storage and processing to ensure compliance. Therefore, by combining the powerful attack and defense capabilities of I_OD_CNN with the comprehensive strategies of data encryption and regulatory compliance, a stable and secure network environment can be provided for the power information system.

5 Conclusion

In this research, the ID model of PISIoT network with hybrid LWP algorithm and OD-CNN algorithm is designed and the results of information recognition experiments are as follows. I_OD-CNN converges substantially more quickly during the training phase than OD-CNN did before improvement, and it no longer differs significantly from the other two algorithms in terms of convergence speed. With the exception of the OD-CNN model, all models converge when there have been roughly 140 iterations. After convergence, the values of each model's loss function are all close to 1.5. While the recognition accuracy of the other two models barely differs from the I_OD-CNN model, the OD-CNN model's overall accuracy and precision are much lower than those of the I_OD-CNN model. The recognition accuracies of

the I_OD-CNN, OD-CNN, Mobile Net, and Shuffle Net models are 92.6%, 90.7%, 91.2%, and 91.4%, respectively, when the amount of test data is the entire test set, or 258. Additionally, the recognition recall of the OD-CNN model performs worse than the other models in all types of intrusion assault situations; the recall of the other three models performs slightly better, with a value of about 92%. The I_OD-CNN model had the lowest AUC, and the differences between the AUCs of the OD-CNN, Mobile Net, and Shuffle Net models were incredibly small. The corresponding AUCs of the ROCs for the I_OD-CNN, OD-CNN, Mobile Net, and Shuffle Net models were 0.69, 0.57, 068, and 0.70, respectively. I_OD-CNN, OD-CNN, Mobile Net, and Shuffle Net models' average ID speeds throughout the trial were 18.2 ms, 53.6 ms, 24.3 ms, and 29.5 ms, respectively. The I_OD-CNN model had the lowest average memory consumption, followed by the Mobile Net model and the Shuffle Net model. The OD-CNN model had a much higher computational memory usage. The memory requirements of the I_OD-CNN, OD-CNN, Mobile Net, and Shuffle Net models are 2893 KB, 18602 KB, 3741 KB, and 4262 KB, respectively, when the computational samples constitute the whole test set. Based on the aforementioned experimental data, it can be seen that the ID model of the PISIoT network designed in this research has good detection accuracy and that it performs significantly better than the conventional model in terms of computational speed, computational memory consumption, and level of model structure simplification.

Although the research has made some progress in constructing the PIS-IoT network intrusion detection model, there are still several limitations. First, the relatively small size of the data set used can lead to sample imbalances or biases, which affect the model's ability to generalize and its adaptability to new types of attacks. In addition, existing models mainly rely on static features and fail to fully consider the importance of time series data in intrusion detection, which may limit the ability to identify dynamic changes in attack behavior. To solve these problems, future research can be improved from the following aspects: First, transfer learning technology can be applied to improve the performance of the model on a small data set. For example, by pre-training the model on a large relevant data set, the learned features are transferred to the target task, thereby effectively improving the recognition effect on a specific small-scale data set. This not only helps reduce the need for large amounts of labeled data, but also improves the detection of rare attack patterns. Secondly, for time data analysis in intrusion detection, we can study the architecture combining recurrent neural network (RNN) and OD-CNN. RNN is good at processing time series data and can

capture the dynamic characteristics of attack behavior over time, which is complementary to the spatial feature extraction of OD-CNN. Such integration will help improve the recognition accuracy of complex attack sequences. Finally, combined with anomaly detection techniques, self-supervised learning methods can be introduced to identify minimal differences between normal and abnormal behavior. By establishing a model of normal behavior, the system can actively identify and mark potential unknown attacks in the detection process, so as to further improve the detection rate and reduce the false positive rate.

Acknowledgement

The research thanks the support from Guizhou Wujiang Hydropower Development Co., Ltd.

References

[1] Jia H, Liu J, Zhang M, He X H, Sun W Xl. Network intrusion detection based on IE-DBN model. Computer Communications, 2021, 178(Oct):131–140.

[2] Balamurugan E, Mehbodniya A, Kariri E, Yadav K, Kumar A, Haq M A. Network optimization using defender system in cloud computing security-based intrusion detection system with game theory deep neural network (IDSGT-DNN). Pattern recognition letters, 2022, 156(Apr):142–151.

[3] Chen Y, Lin Q, Ji J, Wei W, Wong K C, Coello C A C. Intrusion detection using multi-objective evolutionary convolutional neural network for Internet of Things in Fog computing. Knowledge-based systems, 2022, 244(May 23):108505.1–108505.14.

[4] Guo Y, Mustafaoglu Z, & Koundal D. Spam Detection Using Bidirectional Transformers and Machine Learning Classifier Algorithms. Journal of Computational and Cognitive Engineering, 2022, 2(1), 5–9.

[5] Hidayat I, Ali M Z, Arshad A. Machine Learning-Based Intrusion Detection System: An Experimental Comparison. Journal of Computational and Cognitive Engineering, 2022, 2(2):88–97.

[6] Liu Z, Su N, Qin Y, Lu J H, Li X F. A Deep Random Forest Model on Spark for Network Intrusion Detection. Mobile Information Systems, 2020, 2020(Pt.3):6633252.1–6633252.16.

[7] Li X, Zhang S. Network Intrusion Detection Methods Based on Deep Learning. Recent patents on engineering, 2021, 15(4): e210421180688.1–e210421180688.9. DOI: 10.2174/1872212114999200403092708.
[8] Jiang K, Wang W, Wang A, Wu H B. Network Intrusion Detection Combined Hybrid Sampling with Deep Hierarchical Network. IEEE Access, 2020, 8:32464–32476.
[9] Zhang H, Li Y, Lv Z, Kumar A. A real-time and ubiquitous network attack detection based on deep belief network and support vector machine. IEEE/CAA Journal of Automatica Sinica, 2020, 7(3):790–799. DOI: 10.1109/JAS.2020.1003099.
[10] Kim T, Pak W. Real-time network intrusion detection using deferred decision and hybrid classifier. Future Generation Computer Systems, 2022, 132:51–66.
[11] Moizuddin M D, Jose M V. A bio-inspired hybrid deep learning model for network intrusion detection. Knowledge-based systems, 2022, 238(Feb.28):107894.1–107894.20.
[12] Alhajjar E, Maxwell P, Bastian N. Adversarial machine learning in Network Intrusion Detection Systems. Expert Systems with Applications, 2021, 186(2):115782.1–115782.13.
[13] Banitaba F S, Aygun S, Najafi M H. Late breaking results: Fortifying neural networks: Safeguarding against adversarial attacks with stochastic computing. arXiv preprint arXiv:2407.04861, 2024.
[14] Nouraniboosjin S, Yousefi M, Meisami S, Yousefi M, Meisami S. Empowering healthcare: a blockchain-based secure and decentralized data sharing scheme with searchable encryption. International Journal on Cybernetics & Informatics (IJCI), 2024, 13(13):47.
[15] Larijani A, Dehghani F. A computationally efficient method for increasing confidentiality in smart electricity networks. Electronics, 2023, 13(1):170.
[16] Pawlicki M, Chora M, Kozik R. Defending network intrusion detection systems against adversarial evasion attacks. Future Generation Computer Systems, 2020, 110(11):148–154.
[17] Zhou Y, Mazzuchi T A, Sarkani S. M-AdaBoost-A Based Ensemble System for Network Intrusion Detection. Expert Systems with Applications, 2020, 162(Dec):113864.1–113864.15.

[18] Wu C, Li W. Enhancing intrusion detection with feature selection and neural network. International Journal of Intelligent Systems, 2021, 36(7):3087–3105.

[19] Alvares C, Dinesh D, Alvi S, Gautam T, Hasib M, Raza A. Dataset of attacks on a live enterprise VoIP network for machine learning based intrusion detection and prevention systems. Computer Networks, 2021, 197(Oct.9):108283.1–108283.6.