DESNN Algorithm For Communication Network Intrusion Detection

Fulai Liu
Northeastern University - Qinhuangdao Campus
Jialiang Xu (jialiang9102@163.com)
Northeastern University https://orcid.org/0000-0001-6453-0527
Lijie Zhang
Northeastern University
Ruiyan Du
Northeastern University - Qinhuangdao Campus
Zhibo Su
Northeastern University
Aiyi Zhang
Northeastern University
Zhongyi Hu
Northeastern University

Research Article

Keywords: Intrusion detection, Deep neural network, Dynamic pruning rule, Neural network compression

Posted Date: June 2nd, 2021

DOI: https://doi.org/10.21203/rs.3.rs-471468/v1

Noname manuscript No.
(will be inserted by the editor)

DESNN Algorithm for Communication Network Intrusion Detection

Fulai Liu[1,2,*], Jialiang Xu[2,*], Lijie Zhang[2,*] Ruiyan Du[1,2], Zhibo Su[2], Aiyi Zhang[2], Zhongyi Hu[2]

Received: date / Accepted: date

Abstract Intrusion detection is a crucial technology in the communication network security field. In this paper, a dynamic evolutionary sparse neural network (DESNN) is proposed for intrusion detection, named as DESNN algorithm. Firstly, an ensemble neural network model is constructed, which is processed by a dynamic pruning rule and further divided into advantage subnetworks and disadvantage subnetworks. The dynamic pruning rule can effectively reduce the subnetworks weight parameters, thereby increasing the speed of the subnetworks intrusion detection. Then considering the subnetworks performance loss caused by the dynamic pruning rule, a novel evolutionary mechanism is proposed to optimize the training process of the disadvantage subnetworks. The weight of the disadvantage subnetworks approach the weight of the advantage subnetworks by the evolutionary mechanism, such that the performance of the ensemble neural network can be improved. Finally, an optimal subnetwork is selected from the ensemble neural network, which is used to detect multiple types of intrusion. Experiments show that the proposed DESNN algorithm improves intrusion detection speed without causing significant performance loss compare with other fully-connected neural network models.

Keywords Intrusion detection · Deep neural network · Dynamic pruning rule · Neural network compression

1 Introduction

Intrusion detection system (IDS) plays a crucial role in defending communication network [1-2]. Several state-of-art intrusion detection algorithms are proposed in recent years [3-7]. However, the rapid development and popularization of the network bring many challenges [8], such as: 1) the intrusion detection performance is unsatisfied due to the diversification of attack types; 2) The detection speed of the IDS is reduced due to the volume of data both stored and passing through networks persistently increase. Fortunately, the deep learning (DL) has the ability to learn and model complex nonlinear relationships that allow it to solve high-dimensional classification or prediction problems, the DL-based IDS is deployed as a potential solution to effectively detect the network intrusions.

It has been proved that the DL-based IDS can effectively detect intrusions recent years. Invoking the modified recurrent neural network (RNN) based on machine learning (ML), an intrusion detection algorithm is presented [3]. This algorithm improves the accuracy of the intrusion detection. Combining the deep belief network (DBN) and the ensemble support vector machine (SVM), a new model is proposed for the intrusion detection problem [4]. This model uses the DBN to extract data features which are used by the ensemble SVM to determine intrusion types. A novel intrusion detection model is introduced, which joints nonsymmetric deep autoencoder (NDAE) and random forests classification algorithm (RF) [5]. This model offers high levels of accuracy, precision and recall together. An intrusion detection model is proposed [6], which is based on fully-connected network, variational autoencoder (AE) and Sequence-to-Sequence (Seq2Seq). Experimental results show that this model has better intrusion detection performance compared with other models. The DL-based IDS performs well in classification performance and can accurately identify attack types.

Though the aforementioned methods are promising, the neural network has considerable redundant parameters, which may take more time to detect the types of intrusion. Neural network pruning can effectively improve the real-time performance of intrusion detection while maintaining detection accuracy. The existence of redundant connections

[1] Engineer Optimization & Smart Antenna Institute, Northeastern University at Qinhuangdao, Qinhuangdao, China.
· [2] School of Computer Science and Engineering, Northeastern University, Shenyang, China.
*Corresponding author: Fulai Liu (fulailiu@126.com), Jialiang Xu (jialiang9102@126.com), Lijie Zhang (zhanglijie0401@163.com). Fulai Liu and Lijie Zhang have contributed equally to this study.

in neural networks is proved [8]. Therefore, with a proper strategy, it is possible to compress neural networks without significantly losing their prediction accuracy. A data-free model compression method is introduced [9]. The square difference of neural network output is used to determine whether the neuron connection is deleted or not, such that the impact of data on pruning is avoided. A simple regularization method based on soft weight-sharing is presented [10], which includes both quantization and pruning in network retraining procedure. A network compression method is proposed [11], named as dynamic network surgery (DNS) method, which can significantly reduce network complexity by dynamic pruning. However, it is difficult to determine the appropriate weight evaluation criteria, therefore the pruning process may result in the decline of neural network performance.

In order to solve the problem that pruning may damage the detection performance of neural networks, a dynamic evolutionary sparse neural network (DESNN) is proposed. Firstly, a dynamic pruning (DP)-based ensemble neural network model is constructed to detect intrusion. Then a novel evolutionary mechanism is proposed to optimize the training process of the ensemble neural network. Finally, an optimal subnetwork is selected from the ensemble neural network, which can detect intrusion faster and better than the other models.

The outline of the paper is organized as follows. The problem model is described in Sect. 2. Section 3 introduces the proposed DESNN algorithm. Section 4 shows some experiment results, the conclusion is given in Sect. 5.

2 System Model

Deep learning (DL) can be utilized in intrusion detection for both dimensionality reduction and classification tasks, which automatically learns complex features from multi-dimensional and large-scale data [12-13]. Thus, deep learning models can be trained with large amounts of historical data to build an intrusion detection model. The model classifies the new traffic into either the normal or anomaly class. If a multi-class classification is used, the model can further classify the infected traffic to different classes and subclasses of attacks. Fig. 1 illustrates the overall architecture of a DL-based IDS.

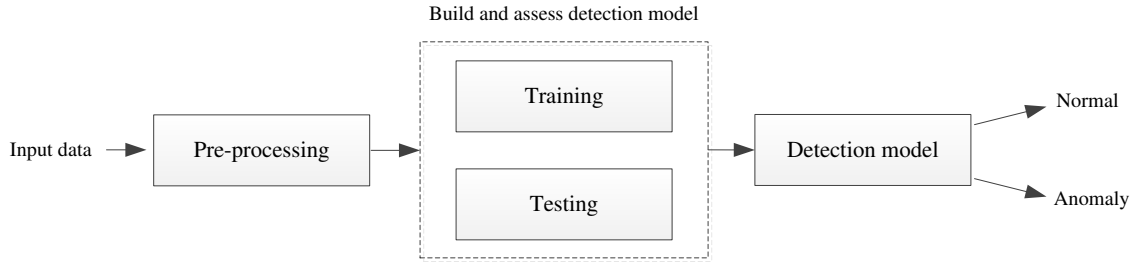

Fig. 1: DL-based IDS architecture

KDD-Cup 99 data set is used as the input data to verify the performance of the proposed algorithm. This data set is collected from a simulated United States air force over nine weeks of communication network connection data, which can be divided into labeled training data and unlabeled test data. The test data and training data have different probability distributions, and the test data contains some intrusion types that do not appear in the training data, which makes intrusion detection more realistic.

The date pre-processing including the numericalization of text features and the normalization of numerical features. The text features are converted into the numerical features, furthermore all the numerical features are quantified in the same range through normalization.

The neural network is trained to find the optimal weight of the neural network, the objective function can be expressed as follows

$$\mathbb{W}^* = \arg\max_{\mathbb{W}} \left[\frac{1}{N} \sum_{i}^{N} \sum_{j}^{J} (y_{i,j} \log(f_j(x_i;\mathbb{W})) + (1-y_{i,j}) \log(1-f_j(x_i;\mathbb{W}))) \right] \tag{1}$$

where N denotes the number of intrusion samples. J represents the number of intrusion types. $y_{i,j}$ is the expected probability with the ith sample belongs to the intrusion type j. $f_j(x_i;\mathbb{W})$ stands for the predicted probability. $\mathbb{W}$ indicates the neural network weight set. $\mathbb{W}^*$ means the optimal neural network weight set. Through the neural network training process, the neural network learns knowledge from the data sample, such that the neural network weight $\mathbb{W}$ gradually converge to the optimal weight $\mathbb{W}^*$.

After the neural network training process, the performance of the neural network is tested under the unlabeled test data. If the intrusion detection performance of the neural network meets the IDS requirements, it can be used as a detection model to detect intrusion types. If not, the neural network needs to be further trained.

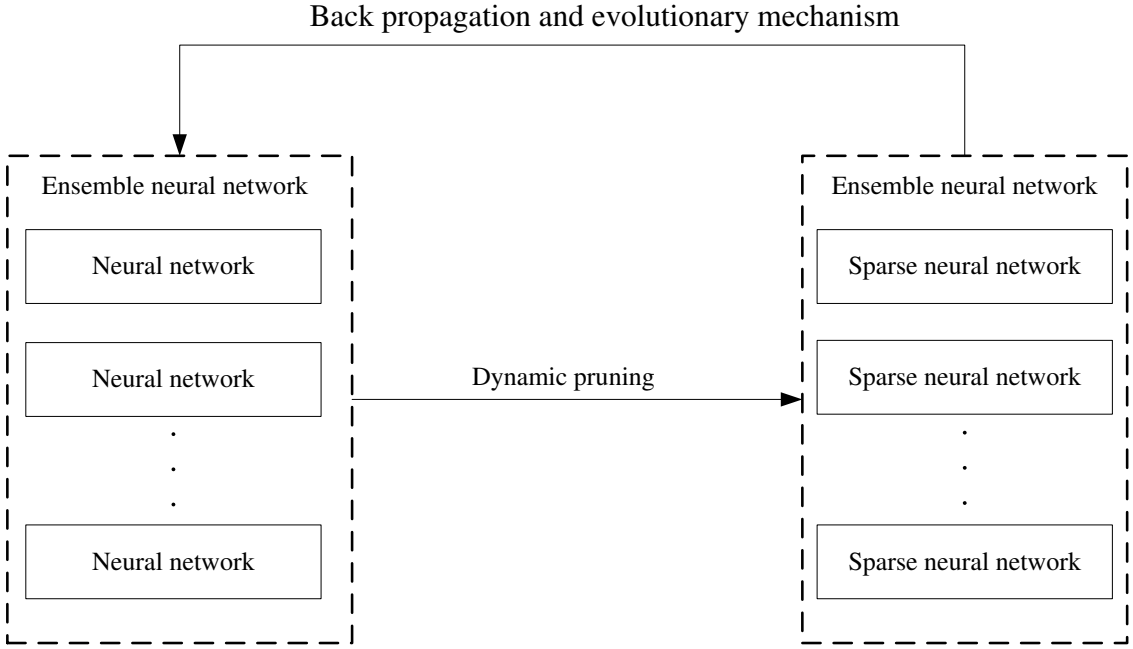

Fig. 2: Ensemble neural network based on dynamic pruning rule and evolution mechanism

3 Algorithm Formula

Redundant connections between neurons may increase intrusion detection time. Combining a dynamic pruning rule and an evolution mechanism, a DESNN algorithm is proposed to eliminate the redundant connections issue without causing significant performance loss. Ensemble neural network based on dynamic pruning rule and evolution mechanism is shown in Fig 2. The process of eliminating redundant connections in the neural network is expressed as an optimization problem. In this section, the optimization problem, the dynamic pruning rule and the evolution mechanism are clarified.

3.1 Optimization Problem

The DESNN algorithm reduces redundant neuron connections through an unstructured pruning. Fig 3 shows neural network with the unstructured pruning. The corresponding optimization problem can be expressed as

$$\min_{\mathbf{W}_k,\mathbf{T}_k} L(\mathbf{W}_k \odot \mathbf{T}_k) \tag{2}$$

where $L(\cdot)$ stands for a loss function. $\mathbf{W}_k$ denotes the matrix of neural network connection weights in the kth layer. $\odot$ represents the Hadamard product operator. $\mathbf{T}_k = h_k(\mathbf{W}_k)$ indicates the filter matrix. $h_k(\cdot)$ is a pruning rule. The following describes that the problem (2) is optimized by the dynamic pruning rule.

3.2 Dynamic Pruning Rule

In this paper, a dynamic pruning rule based on weight contribution is used to find redundant connections of neurons. The neural network is pruned by the pruning rule $h_k(\cdot)$, which can be expressed as

$$h_k(W_k^{(a,b)}) = \begin{cases} 1 & if |W_k^{(a,b)}| \geq p_k \\ 0 & if |W_k^{(a,b)}| < p_k \end{cases}, \forall (a,b) \in \mathbb{I} \tag{3}$$

where $\mathbf{W}_k^{(a,b)}$ denotes the neuron connection with indicator (a,b) in the kth layer. $\mathbf{T}_k^{(a,b)}$ stands for the filter matrix element with indicator (a,b) in the kth layer. Set $\mathbb{I}$ represents all the entry indices in matrix $\mathbf{W}_k$. a represents the input neuron indicator, b is the output neuron indicator. p_k denotes the pruning threshold of the kth layer.

In the proposed pruning rule $h_k(\cdot)$, the pruning threshold p_k is determined by pruning rate α and the total number of $\mathbf{W}_k$ elements n. Sort the absolute value of all elements in the weight matrix $\mathbf{W}_k$ by the ascending order as a weight

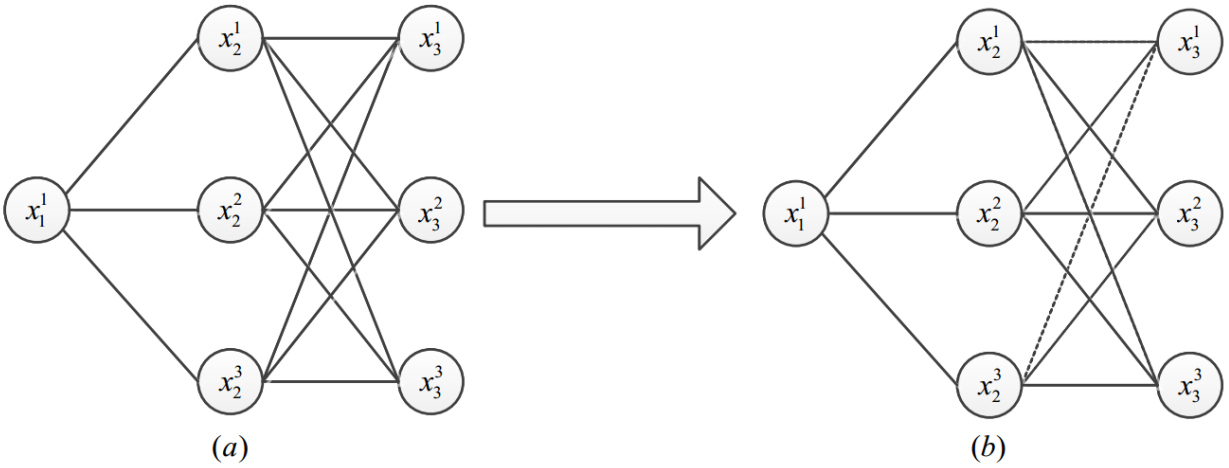

Fig. 3: Neural network with unstructured pruning

contribution vector $\mathbf{c}_k$. Then the pruning threshold p_k is set as the $(n \times \alpha)$th value in the contribution vector $\mathbf{c}_k$. Then the weight matrix $\mathbf{W}_k$ is processed by expression (3), the filter matrix element $\mathbf{T}_k^{(a,b)}$ is 1 if the weight matrix element $\mathbf{W}_k^{(a,b)}$ has a higher contribution to the neural network; the value of $\mathbf{T}_k^{(a,b)}$ is 0 otherwise.

Updating the weight matrix $\mathbf{W}_k$ and the filter matrix $\mathbf{T}_k$ through the stochastic gradient descent (SGD) method, then the back propagation process of the proposed neural network can be expressed as

$$\mathbf{W}_k^{(a,b)} = \mathbf{W}_k^{(a,b)} - \beta \frac{\partial L(\mathbf{W}_k \odot \mathbf{T}_k)}{\partial (\mathbf{W}_k^{(a,b)} \mathbf{T}_k^{(a,b)})}, \ \forall (a,b) \in \mathbb{I} \tag{4}$$

where β is a positive learning rate. Set $\mathbb{I}$ represents all the entry indices in matrix $\mathbf{W}_k$.

Due to the filter matrix $\mathbf{T}_k$ is introduced into the back propagation process of the neural network, the weight optimization of sparse neural network $\{\mathbf{W}_k \odot \mathbf{T}_k : k \in \mathbb{K}\}$ is different from the weight optimization of fully connected neural network $\{\mathbf{W}_k : k \in \mathbb{K}\}$, set $\mathbb{K}$ denotes all the entry indices of the neural network layer. The expression (4) updates all weight elements in the weight matrix $\mathbf{W}_k$, not only the important parameters, but also the unimportant parameters of $\mathbf{W}_k$. The neural network is processed by the dynamic pruning rule to realize the dynamic connection and inhibition of neurons, which enhances the flexibility of pruning.

3.3 Evolutionary Mechanism

Although the dynamic pruning rule can effectively reduce redundant connections, the pruning rate α affects the pruning effect. Excessive pruning rate may cause important connections of the neural network to be inhibited, such that affects the intrusion detection performance of the neural network. In order to address this problem, an ensemble neural network model is constructed, furthermore the proposed evolutionary mechanism is used to optimize the back propagation of the ensemble neural network, which can be described as follows

$$\mathbf{W}_k^d = (\mathbf{W}_k^{op} + \mathbf{W}_k^d)/2 \tag{5}$$

where $\mathbf{W}_k^{op}$ denotes the weight matrices of the advantage subnetworks in the kth layer. $\mathbf{W}_k^d$ represents the weight matrices of the advantage subnetworks in the kth layer. These subnetworks in the ensemble neural network are divided into advantage and disadvantage subnetworks according to their instinct performance. The expression (5) can realize that the disadvantage subnetworks weight matrices $\mathbf{W}_k^d$ approximate the advantage subnetworks weight matrices $\mathbf{W}_k^{op}$ in each epoch of training process. Since the the disadvantage subnetworks weight matrices $\mathbf{W}_k^d$ is closer to the local optimal value, the performance of the disadvantage subnetworks is improved.

4 Experiments

In this section, several experimental results are provided to evaluate the performance of the proposed DESNN algorithm. In the following simulation experiments, the software is set as follows. Python: 3.7.6; keras=2.3.1. The hardware is set as follows. CPU Intel®Xeon®E5-2560, GPU NVIDIA Tesla K80, RAM 64GB.

4.1 Dataset

The intrusion detection dataset KDD-Cup 99 is widely used to verify the performance of intrusion detection algorithms. The details of the dataset are shown in Table 1.

Table 1: KDD-CUP 99 dataset details

Data type	Training set	Testing set
Normal	97278	60593
Dos	391458	229853
Probe	4107	4166
Probe	1126	16189
U2R	52	228
Total	494021	311029

4.2 Model Parameters

To ensure the reproducibility of the experiment, the parameters of the DNN model set in this paper are shown in Table 2. The model is a shallow model, which not only has better intrusion detection classification performance, but also has faster intrusion detection speed. In order to ensure the performance of the model, the parameters of the model are selected based on performance in multiple experiments.

Table 2: DNN model

Layer setting	Layer parameter
Dense	1344(41×32+32)
ReLU	
Dense	2112(32×64+64)
ReLU	
Dense	4160(64×64+64)
ReLU	
Dense	2080(64×32+32)
ReLU	
Dense	165(32×5+5)
Softmax	5

Although the model processed by the dynamic pruning rule can effectively reduce neuron connections, its intrusion detection performance may decrease. In order to solve this problem, this paper proposes the DESNN algorithm. Some hyperparameters are required in the implementation of the proposed algorithm, which have a certain influence on the neural network model. The hyperparameter of the DESNN algorithm are shown in Table 3.

Table 3: Hyperparameter of the DESNN algorithm

Hyperparameter	Value
Iteration	50×10
Subnetwork number	10
Batch size	128
Pruning rate	0 : 0.1 : 1
Optimizer	RMSprop

where iteration is 50×10, 50 represents the training epoch of ensemble neural network, 10 denotes the retraining epoch of the subnetwork. The retraining epoch can be understood as the pruning interval, which affects the degree of the subnetwork convergence. The weight of the subnetwork can converge a better local optimal value with the retraining epoch value increase. The subnetwork number represents the number of subnetwork in the ensemble neural network. The pruning rate represents the rate of deleted neural network parameters.

4.3 The evaluation indicator

This paper evaluates the network performance from two perspectives, including the classification performance and the running time. Invoking [14-18], the evaluation indicators of classification performance adopt accuracy, precision, recall, F-Score, which can be expressed as

$$A_{acc} = \frac{T_p + T_n}{T_p + T_n + F_p + F_n} \tag{6}$$

where A_{acc} represents the classification accuracy of the neural network. T_p and T_n are the number of normal and intrusion samples correctly classified, respectively. F_p and F_n are the number of normal and intrusion samples mistakenly classified.

$$P_{precision} = \frac{T_p}{T_p + F_p} \tag{7}$$

where $P_{precision}$ denotes the ratio of the T_p to the total number of intrusion samples.

$$R_{recall} = \frac{T_p}{T_p + F_n} \tag{8}$$

where R_{recall} represents the ratio of T_p to the predicting the correct samples.

$$F_{score} = \frac{2 \times T_p}{2 \times T_p + F_n + F_p} \tag{9}$$

where F_{score} is the harmonic average of A_{acc} and R_{recall}.

In order to evaluate the running time of intrusion detection model, the running time of the model code as an evaluation indicator. The running time in the experimental results is obtained by using a python time module.

4.4 Experiment Results

Experiment 1: Network degradation phenomenon

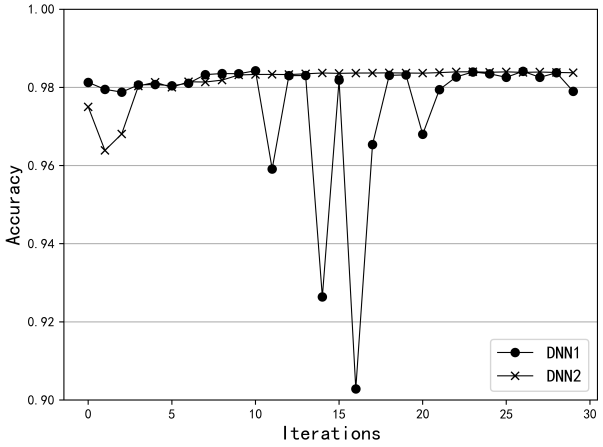

Fig. 4: Accuracy as functions of iterations for different pruning rates

Fig 4 shows the accuracy as functions of iterations for different pruning rates. The pruning rate used by DNN1 is equal to 0.5, the pruning rate used by DNN2 is equal to 0.3. It can be observed that the accuracy of DNN may be reduced by the dynamic pruning rule, named as degradation phenomenon. Due to the selection of the unsuitable pruning threshold p_k, some important connections in the neural network are deleted, which leads to the occurrence of degradation.

Experiment 2: Accuracy comparison of the DNN, the DESNN and the DPNN.

The accuracy of the DNN may be degraded due to the dynamic pruning rule, the DESNN algorithm is proposed to solve this problem.

Fig 5 shows the accuracy as functions of iterations for different pruning rates, the xlable represents pruning rate, the range is [0.1, 0.9] and the interval value is equal to 0.1, the ylable denotes accuracy of neural network. The DESNN stands for the DNN processed by the DESNN algorithm, the DPNN represents the DNN processed by the dynamic pruning rule.

In the Fig 5, the accuracy of the DPNN and the DNN are generated by averaging 5 independent runs. The accuracy of the DESNN is the optimal network accuracy in the ensemble neural network, which is also generated by averaging 5 independent runs. The accuracy of the DNN is 0.97633. When the pruning rate are equal to 0.2 and 0.3,

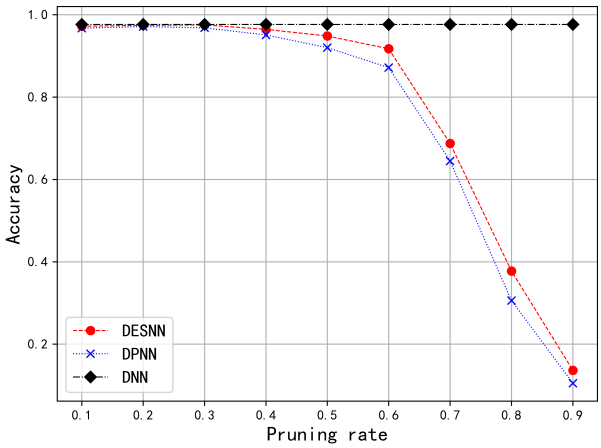

Fig. 5: Accuracy versus pruning rate for different neural network model

the corresponding accuracy of the DESNN are 0.97426 and 0.97535, close to the DNN and higher than the DPNN that accuracy are 0.97126 and 0.96820. When the pruning rate is greater than 0.3, the accuracy rate gradually decreased with the increase of pruning rate. Experiment 2 shows that the DESNN algorithm can reduce the damage caused by dynamic pruning rule to the network.

Experiment 3: Weight distribution diagram of the DESNN and fully-connected DNN.

In the experiment 3, the weight elements of the DESNN and the fully-connected DNN are extracted, which frequency distribution histogram (FDH) are drawn.

Fig 6 represents the FDH of the fully-connected DNN weight elements and Fig 7 denotes the FDH of the DESNN weight elements.

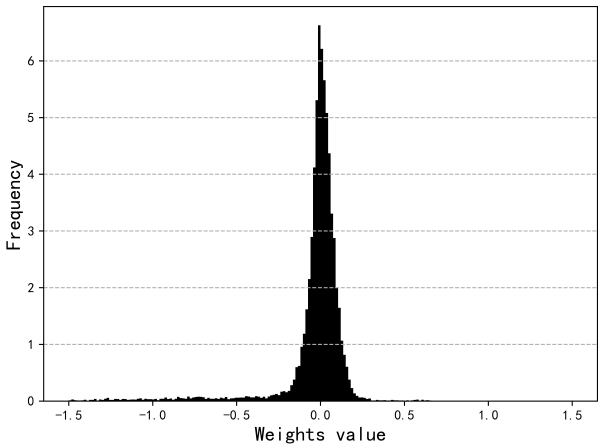

Fig. 6: FDH of the fully-connected DNN weight elements

It can be seen from Fig 6 and Fig 7 that the distribution of weight elements is normal distribution. The reason is that the weights of the neural network are initialized according to the Gaussian distribution. The training process adjust the weights to make the neural network converge gradually, but the training process does not change the weight distribution type of the neural network.

Fig 8 shows the weight distribution of the two neural networks, with the xlable denotes the weight value and the ylabel represents the unit frequency. It can be seen that the weight distribution of DESNN is thinner and higher than fully-connected neural network. The fully-connected DNN is processed by DESNN algorithm that its weight elements approach to 0 as a whole. According to the expression (3), neuron connections with smaller contributions are deleted to speed up the operation of the neural network; Neuron connections with higher contributions are reserved to ensure the intrusion detection performance of the neural network.

Experiment 4: Performance comparison between the proposed algorithm and other intrusion detection models

This experiment tests the classification performance and the running time of various deep learning models. The model paremeters of the DESNN is shown in Table 2 and the hyperparameter of the DESNN algorithm is shown in

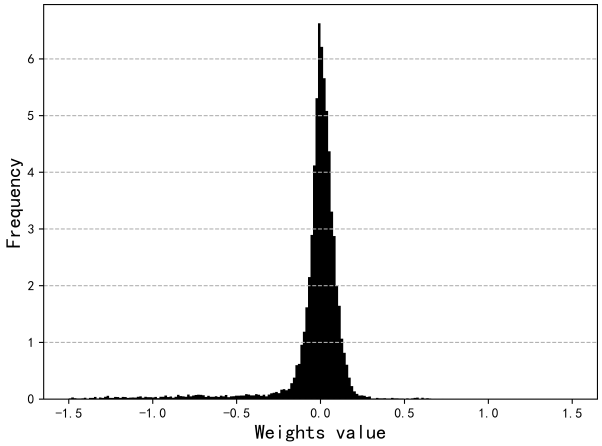

Fig. 7: FDH of the DESNN weight elements

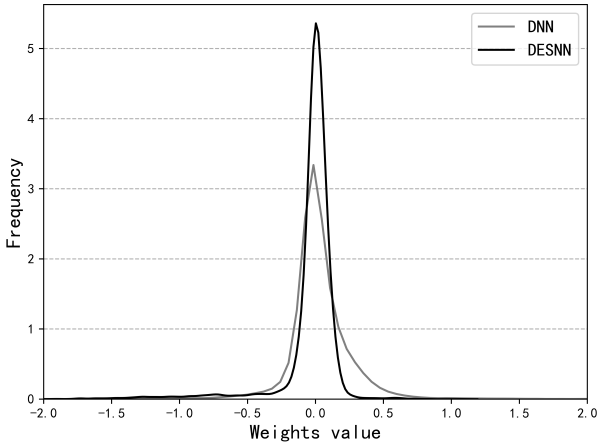

Fig. 8: Comparison of weight distribution between the two networks

Table 3, the pruning rate is equal to 0.3. It is worth noting that in order to allow the data to be entered into the 2D-CNN model, we use the data slicing to convert 1D data into 2D data.

Table 4: Classification performance and running time

Metric	DNN	LSTM	1D-CNN	2D-CNN	**DESNN**
Accuracy	0.9066	0.9201	0.9167	0.9153	**0.9168**
Precision	0.9946	0.9958	0.9952	0.9989	**0.9948**
Recall	0.8887	0.9045	0.9008	0.8957	**0.9012**
F-score	0.9387	0.9479	0.9457	0.9445	**0.9457**
Running time (s)	31.56	42.57	42.40	61.68	**20.71**

It can be seen from Table 4 that the accuracy of all models exceeds 90% and the model with the highest accuracy is the LSTM. The accuracy of the DESNN is higher than the DNN, indicating that the DESNN algorithm can improve the accuracy of the DNN. Because the DNN may be overfitting, and pruning is a means to solve overfitting. From the perspective of running time, the DESNN is lower than all models. Since the number of parameters of the model is directly related to the running time. In this experiment, the weight parameter of the DNN is only 10 thousand, while the LSTM is 69 thousand and the 2D-CNN is 30 thousand, the operating speed of the DNN is lower than that of the LSTM and the 2D-CNN. If the pruning rate is equal to 0.3, then the weight parameters of the DESNN is 7 thousand, so its running time is lower than the DNN.

5 Conclusion

In this paper, the DESNN algorithm is proposed by combining the evolutionary mechanism with the dynamic pruning rule to solve the problem that network performance degradation caused by inappropriate pruning threshold. The proposed DESNN algorithm constructs a DP-based ensemble neural network which can be divided into the advantage subnetworks and the disadvantage subnetworks according to the intrusion detection performance. Furthermore, a new evolution mechanism is proposed and used to optimize the training process of the disadvantaged network, such that the performance of the ensemble neural network can be improved. The proposed algorithm not only effectively improves the intrusion detection performance compared with the fully connected neural network, but also improves the intrusion detection speed compared with all other neural network models.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgements

This work was supported by the National Natural Science Foundation of China under Grant No.61971117, by the Natural Science Foundation of Hebei Province (Grant No. F2020501007).

References

1. Ahmim A, Derdour M, Ferrag M A. (2018). An intrusion detection system based on combining probability predictions of a tree of classifiers. *International Journal of Communication Systems*, 31(9).
2. Ahmad Z, Khan A S, Shiang C W, et al. Network intrusion detection system: A systematic study of machine learning and deep learning approaches. *Transactions on Emerging Telecommunications Technologies*. 32(1). 2020.
3. Yin C L, Zhu Y F, Fei J L, et al. A Deep Learning Approach for Intrusion Detection Using Recurrent Neural Networks. *IEEE Access*. 5: 21954-21961. 2017.
4. Marir N, Wang H, Feng G, et al. Distributed Abnormal Behavior Detection Approach based on Deep Belief Network and Ensemble SVM using Spark. *IEEE Access*. 6: 59657-59671. 2018.
5. Wei P, Li Y, Zhang Z, et al. An Optimization Method for Intrusion Detection Classification Model Based on Deep Belief Network. *IEEE Access*. 7: 87593-87605. 2019.
6. Malaiya R K, Kwon D, Suh S C, et al. An Empirical Evaluation of Deep Learning for Network Anomaly Detection. *IEEE Access*. 7: 140806-140817. 2019.
7. Shone N, Ngoc T N, Phai V D, et al. A Deep Learning Approach to Network Intrusion Detection. *IEEE Transactions on Emerging Topics in Computational Intelligence*. 2(1), 41-50. 2018.
8. Denil M, Shakibi B, Dinh L, et al. Predicting Parameters in Deep Learning. *Conference and Workshop on Neural Information Processing Systems*. 2148-2156. 2013.
9. Srinivas S, Babu R V. Data-free parameter pruning for Deep Neural Networks. *British Machine Vision Conference*. 2830-2838. 2015.
10. Ullrich K, Meeds E, Welling M. Soft weight-sharing for neural network compression. *International Conference on Learning Representations*. 2017.
11. Guo Y W, Yao A B, Chen Y R. Dynamic Network Surgery for Efficient DNNs. *Conference and Workshop on Neural Information Processing Systems*. 1379-1387. 2016.
12. Aldweesh A, Derhab A, Emam A Z. Deep learning approaches for anomaly-based intrusion detection systems: A survey, taxonomy, and open issues. *Knowledge-Based Systems*. 189. 2020.
13. Vani R. Towards Efficient Intrusion Detection using Deep Learning Techniques: A Review. *International Journal of Advanced Research in Computer Science and Electronics Engineering*. 6(10): 375-384. 2017.
14. Kang M J, Kang J W, Tang T. Intrusion Detection System Using Deep Neural Network for In-Vehicle Network Security[J]. *Plos One*. 11(6): e0155781. 2016.
15. Kim J, Kim J, Thu H, Kim H. Long short term memory recurrent neural network classfier for intrusion detection. *International Conference on Platform Technology and Service*. 1-5. 2016.
16. Feng F, Liu X, Yong B, et al. Anomaly detection in ad-hoc networks based on deep learning model: A plug and play device. *Ad Hoc Networks*. 84: 82-89. 2019.
17. Liu G J, Zhang J B. CNID: Research of Network Intrusion Detection Based on Convolutional Neural Network. *Discrete Dynamics in Nature and Society*. 2020.
18. G. Marín, P. Casas, Rawpower. Deep learning based anomaly detection from raw network traffic measurements, in: ACM SIGCOMM 2018 Conference on Posters and Demo. 7: 75–77. 2018.

Figures

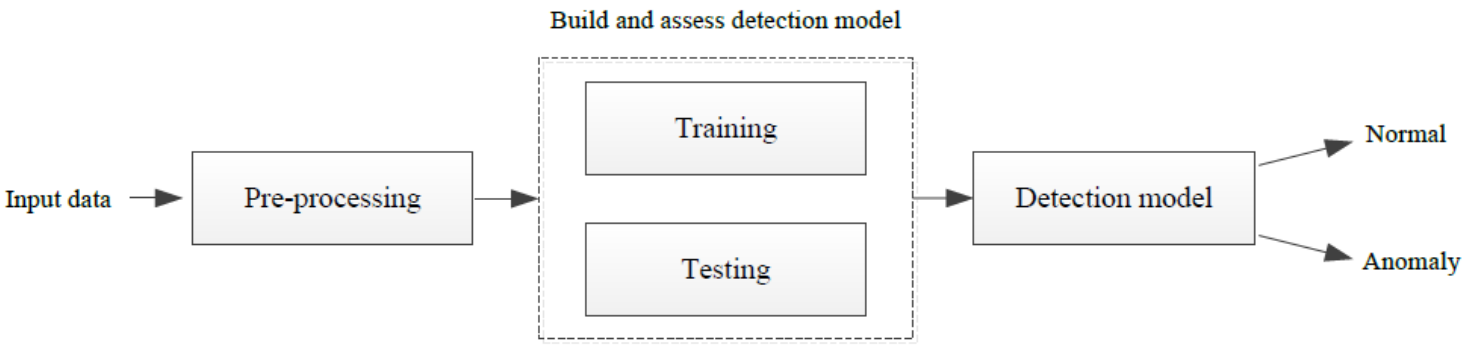

Figure 1

DL-based IDS architecture

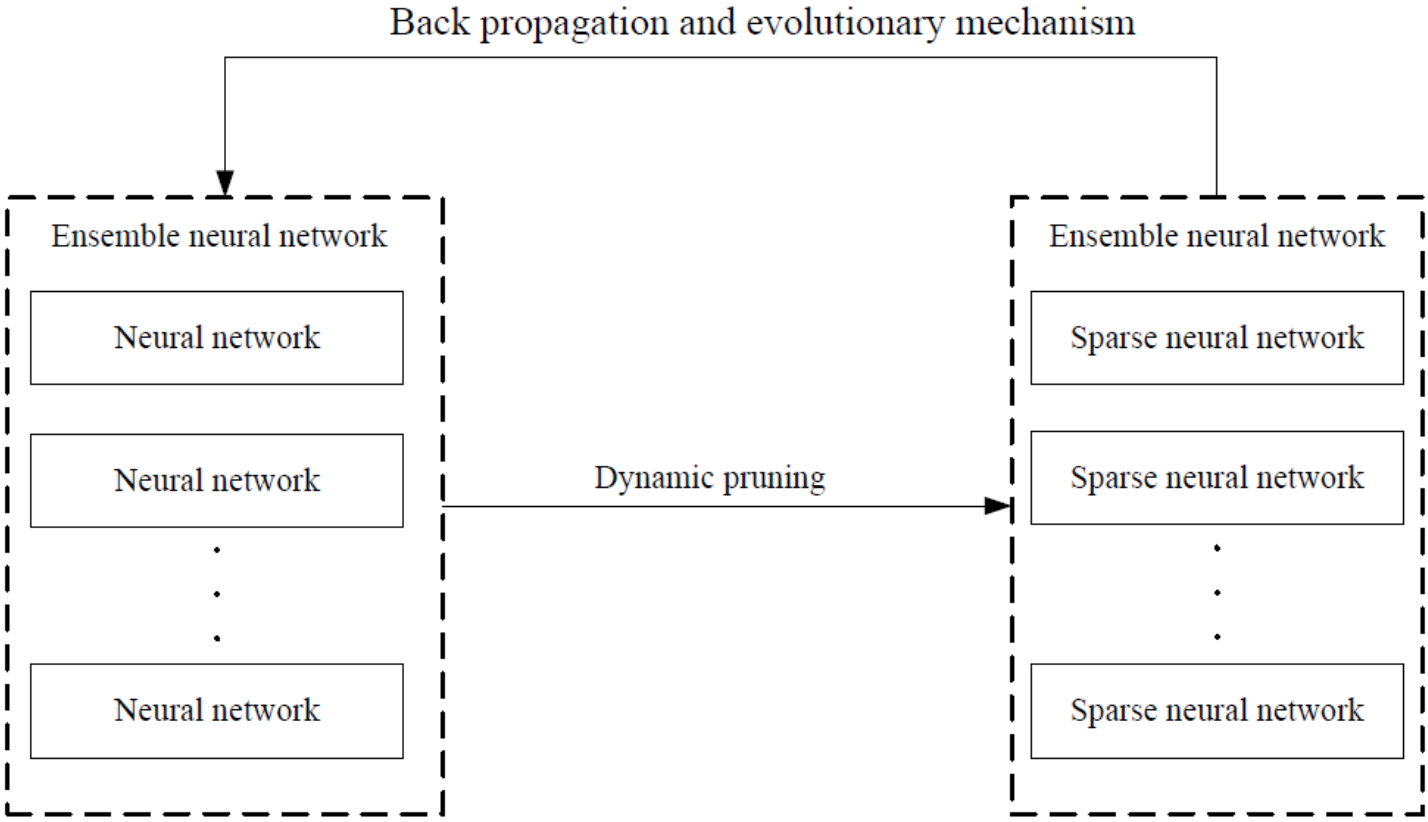

Figure 2

Ensemble neural network based on dynamic pruning rule and evolution mechanism

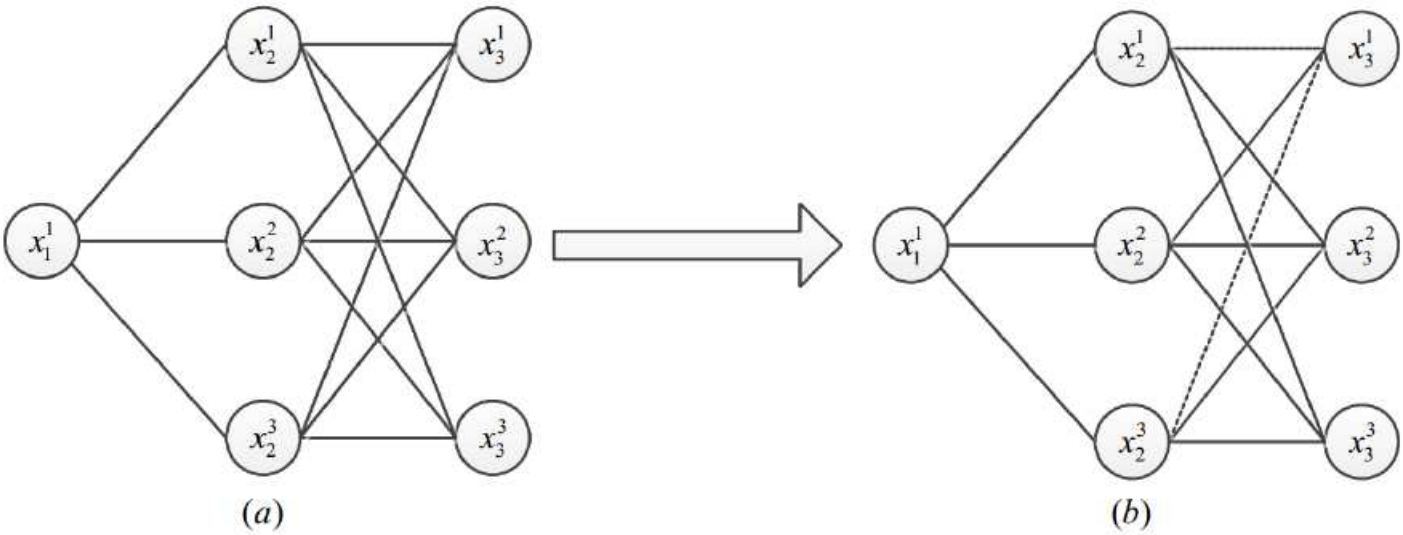

Figure 3

Neural network with unstructured pruning

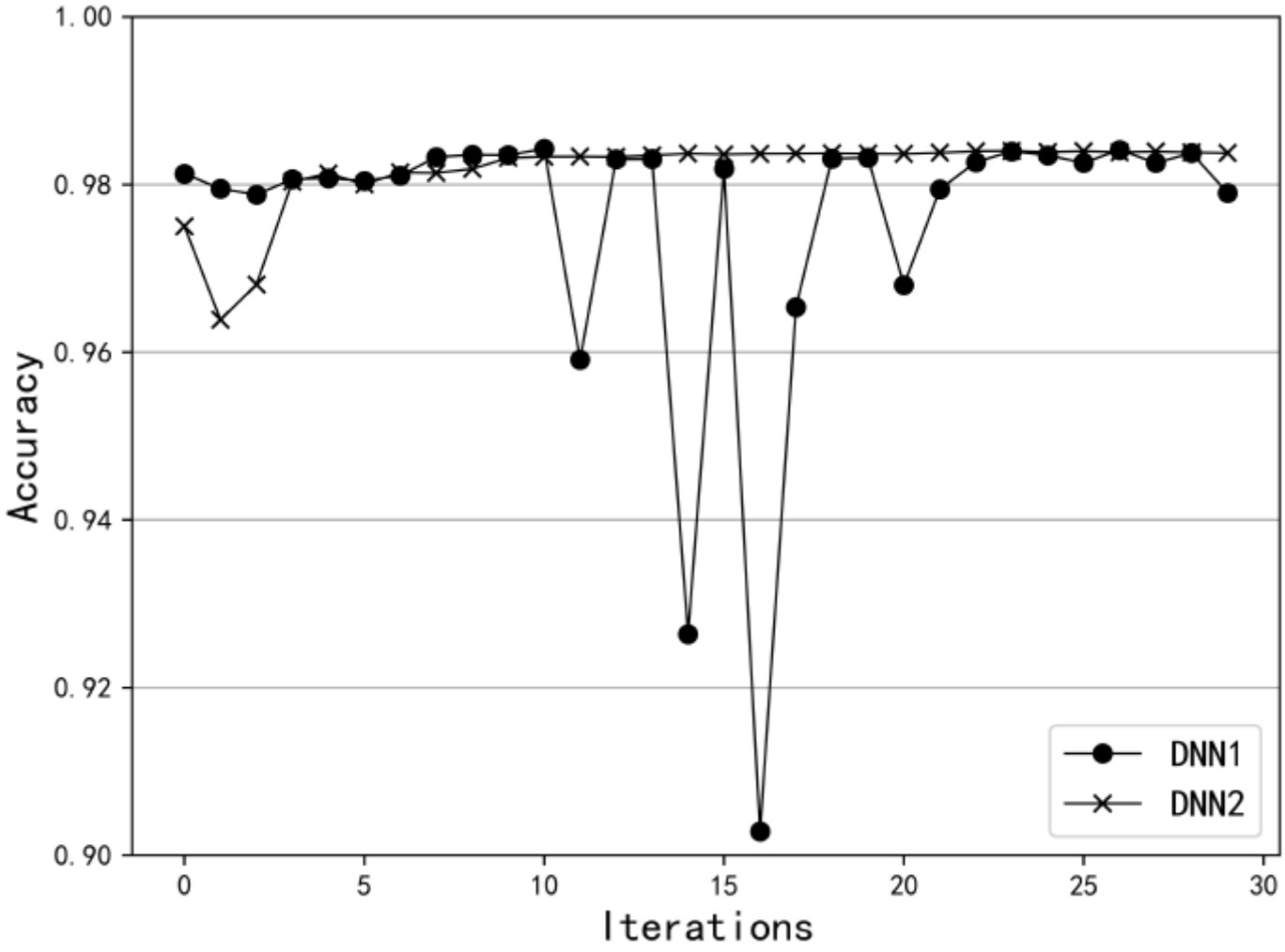

Figure 4

Accuracy as functions of iterations for different pruning rates

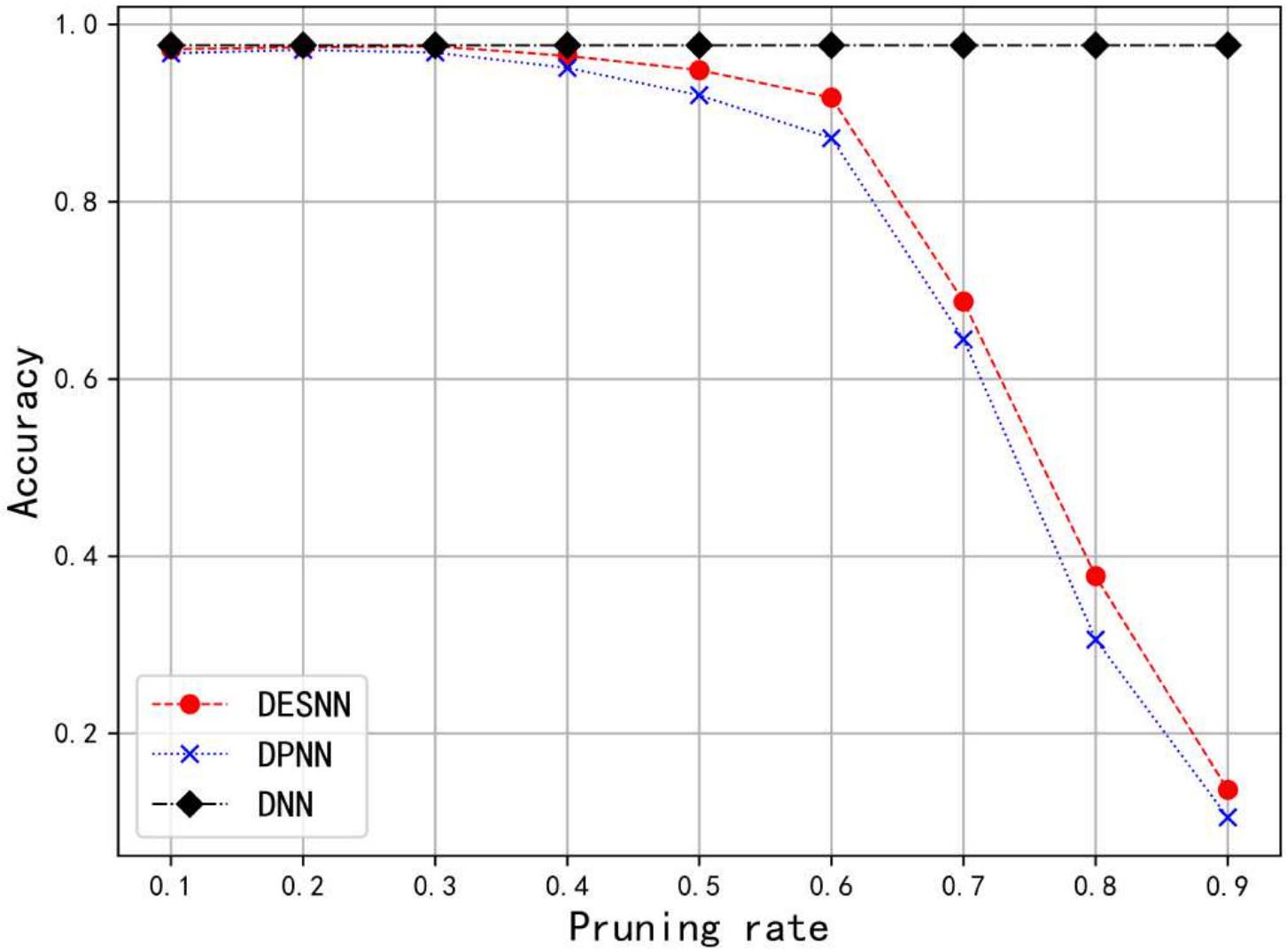

Figure 5

Accuracy versus pruning rate for different neural network model

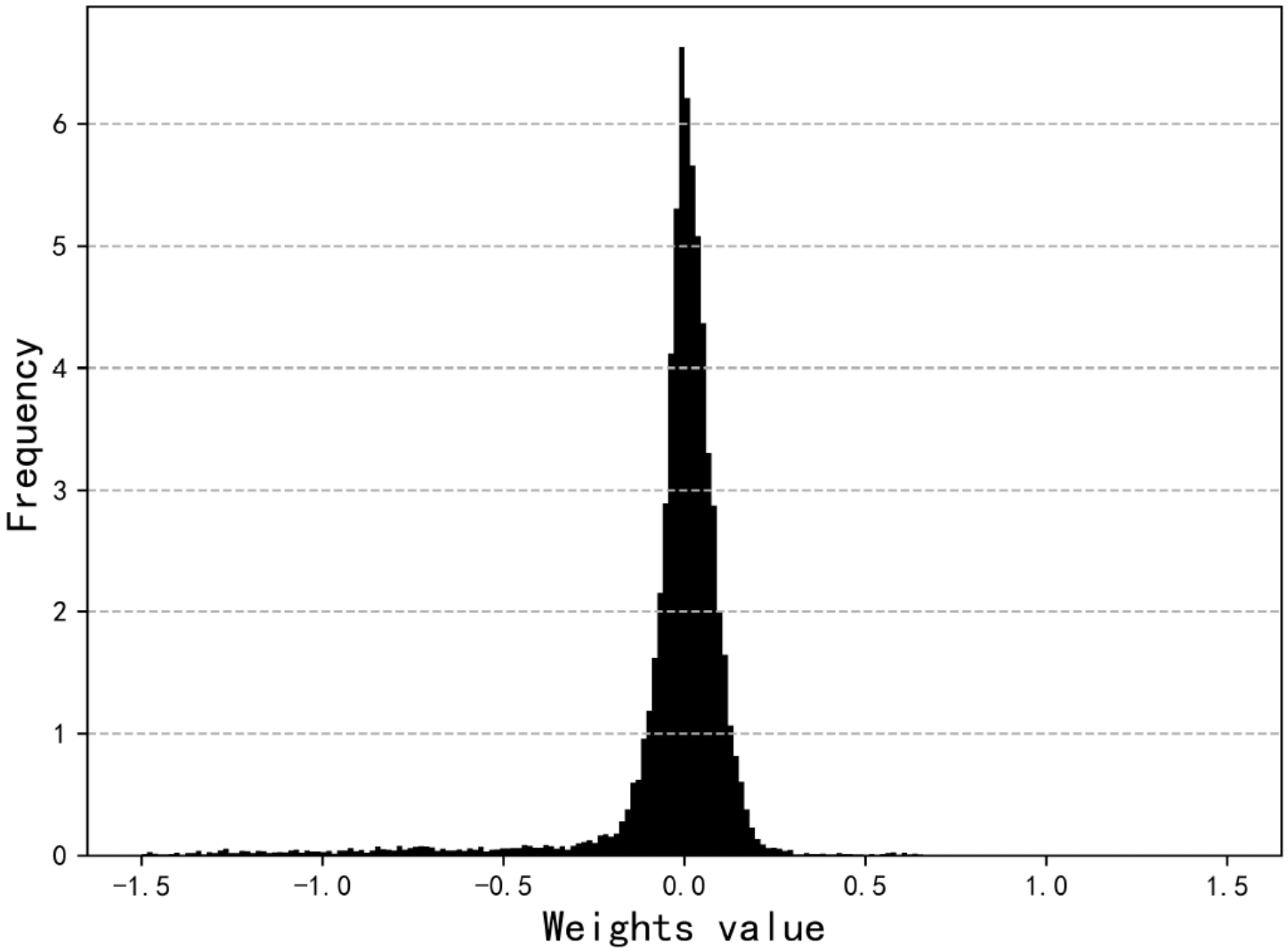

Figure 6

FDH of the fully-connected DNN weight elements

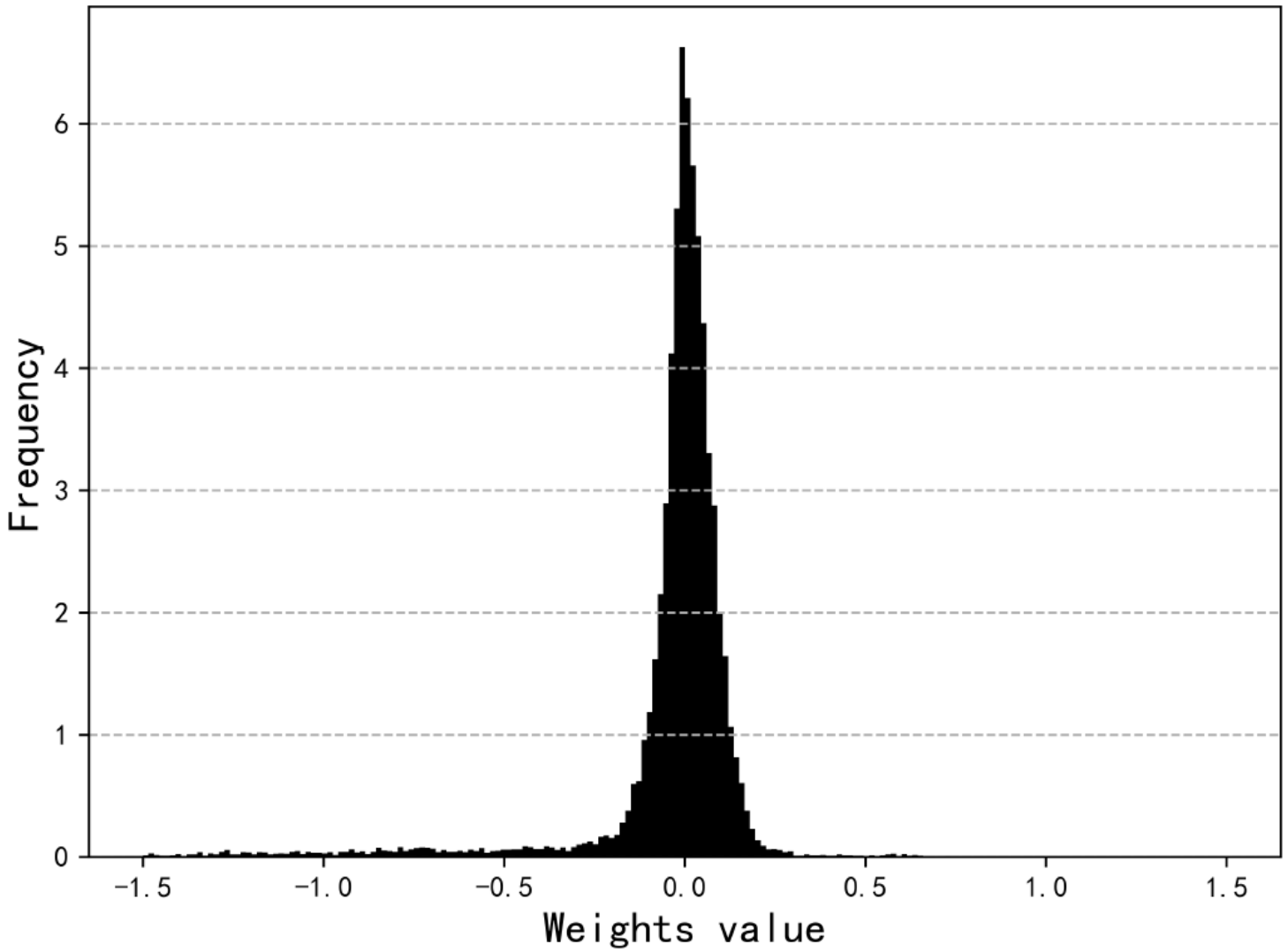

Figure 7

FDH of the DESNN weight elements

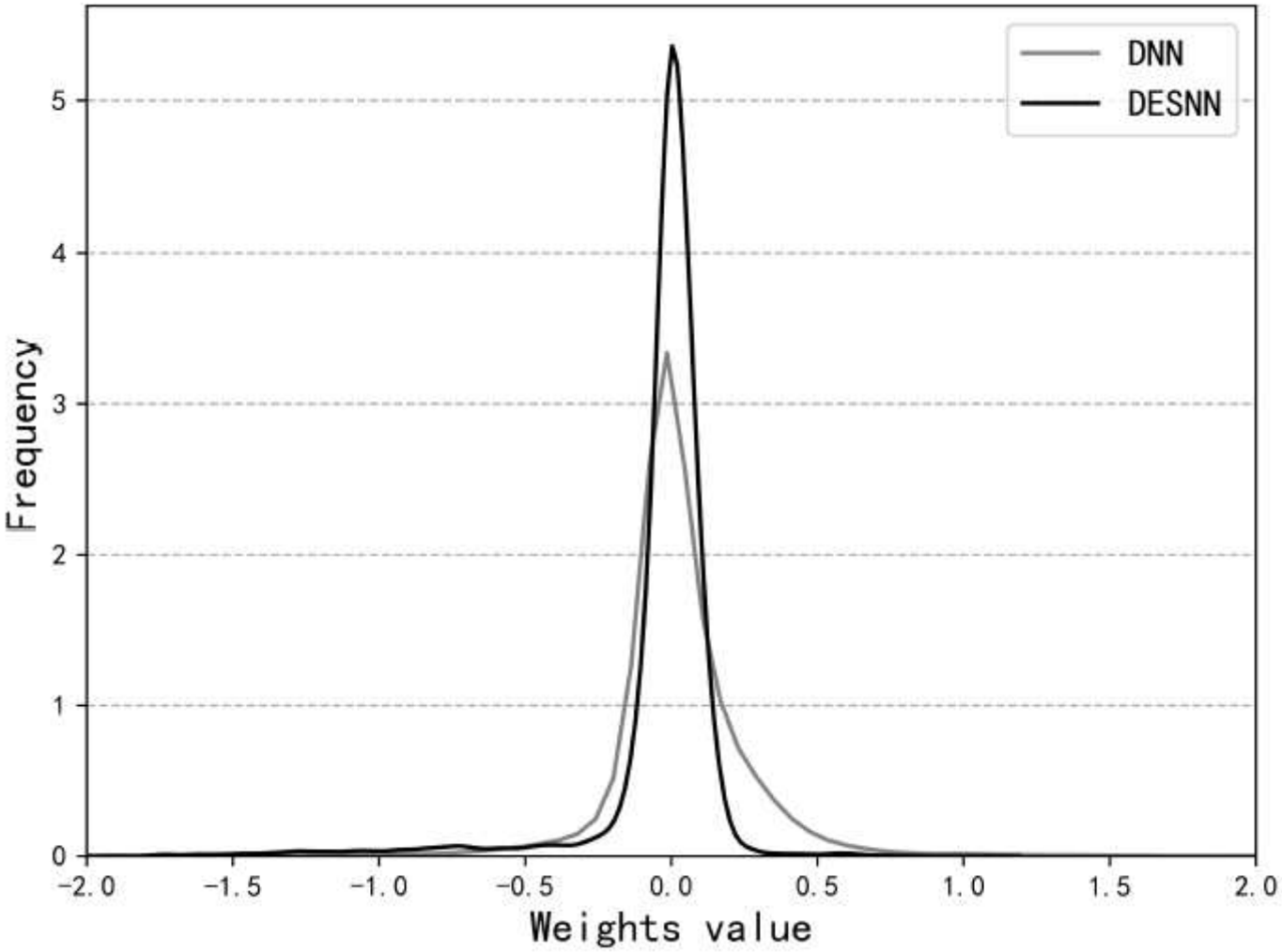

Figure 8

Comparison of weight distribution between the two networks