Industrial Intrusion Detection Classifier Pruning via Hybrid-order Difference of Weights Based on Poisson Distribution

Xue-Jun LIU[1]*, Hao WANG[1], Hai-Ying LUAN[2], Yong YAN[1], Yun SHA[1]

[1] College of Information Engineering, Beijing Institute of Petrochemical Technology,
19 Qingyuan North Road, Daxing District, Beijing, 102617, China
lxj@bipt.edu.cn (**Corresponding author*), caseywang97@foxmail.com,
yanyong@bipt.edu.cn, shayun@bipt.edu.cn

[2] Fluid Drive and Car Equipment Technical Engineering Department, Beijing Research Institute of Automation for Machinery Industry Co., Ltd., Beijing, 100120, China
lhying1129@aliyun.com

Abstract: Pruning Techniques can greatly reduce the number of parameters and the computational load related to convolutional neural networks, which makes them suitable for edge industrial control systems with limited resources. However, they face the problem that the detection accuracy will be greatly reduced after pruning. Given the above, this paper proposes filter pruning via a technique called hybrid-order difference, which is based on Poisson distribution. According to this technique, some filters in each convolutional layer are removed, thus the number of parameters of the employed classifiers is highly reduced. The first-order and second-order difference for the L1-norm of filter parameters are calculated, they are given weights and they are converted into activity indices through the Min-Max function. The proposed method can fully explore the relationship between the weights and avoid the problem of threshold selection in pruning. Experiments were carried out on the LeNet-5, VGG16, ResNet18 and ResNet50 convolutional neural networks based on the 2019 Distributed Denial of Service dataset (the DDoS dataset) of the Canadian Institute for Cybersecurity, the 2014 experimental dataset of the Mississippi State University related to a natural gas pipeline (the gas dataset), and the dataset for an oil depot (the oil dataset). The results showed that the proposed method can effectively prune the employed intrusion detection classifiers, such as removing 83.74% of the Floating Point Operations (FLOPs) for VGG16 with only a 0.10% reduction of accuracy. As such, it greatly alleviates the load pressure generated by the above-mentioned classifiers in the context of edge industrial control systems.

Keywords: Neural network pruning, Poisson distribution, Hybrid-order difference, Edge industrial control system, Intrusion detection.

1. Introduction

Edge is defined as all network resource nodes between the data source and the cloud center (Ning et al., 2020). Edge computing is the process of extending computing services from a centralized cloud-based model to the network edge (Mansouri & Babar, 2021; Shi et al., 2016). In the industrial field, an edge device generates and collects a large amount of data that is initially preprocessed, analyzed on the edge of the network, and then transmitted to a centralized cloud in different ways to extract in-depth knowledge (Taneja et al., 2020; Liu et al., 2021). In recent years, convolutional neural networks have played an important role in industrial control, including data prediction (Shen, 2021), target detection (Wang et al., 2022), anomaly detection (Liu et al., 2018; Globa et al., 2006) and attack sample generation (Zhou et al., 2021). However, with the increasing depth of the network, the demand for computing ability increases gradually, making it difficult for complex networks to be deployed on all kinds of edges, such as industrial control sensors, smart robots, mobile phones, etc. Although reasonable resource allocation for edge devices can effectively improve their overall performance, reducing the number of neural network parameters is still the root of problems.

Researchers have proposed various methods for alleviating resource conflicts, such as more compact network frameworks (Yang et al., 2021), model quantification, knowledge distillation (Song et al., 2022), and network pruning. However, the classification of industrial datasets is different from various image classification tasks. Anomalies in edge industrial control systems often appear in some sensors or in sensors associated with a certain sensor, which resulted in abnormal data appearing in specific columns. Therefore, there will be more redundant features extracted from neural networks. These redundant features make some parameters of the filter in each convolutional layer update slowly or even remain unchanged for a long time. Secondly, in the edge industrial control system, the pruning method must be simple and efficient in order to ensure an optimal data transmission rate between the edge and the cloud with the purpose of facilitating the update and maintenance of intrusion detection classifiers. Based on this, the historical behavior of filter parameters under Poisson distribution is

https://doi.org/10.24846/v31i2y202201 ICI Bucharest © Copyright 2012-2022. All rights reserved

considered for selecting the active filters, which can effectively remove the redundant parameters of intrusion detection classifiers.

After network pruning, the performance degradation should be compensated by retraining the network. At present, there are two primary pruning methods, structured pruning and unstructured pruning (Li et al., 2016). To prune once and retrain means to prune filters of multiple layers at once and retrain them until the original accuracy is restored. To prune and retrain iteratively means to prune filters layer by layer or filter by filter and then retrain iteratively. The classifier is retrained before pruning the next layer so that the weights may adapt to the changes trigged by the pruning process. Obviously, the structured pruning requires less memory and time during training, and the unstructured pruning performs better during retraining. However, the proposed method considers the behavior of multiple filters at one time and pruning, so the former fits better.

Based on the above discussion, this paper proposed filter pruning through hybrid-order difference based on Poisson distribution. This channel-wise pruning method can effectively reduce the number of parameters. It brings about two contributions: (1) Calculating the first-order difference and second-order difference for the L1-norm of filter parameters, giving them weights and converting them into activity indices through the Min-Max function. (2) Verifying the method through two public datasets and an oil depot dataset.

The structure of this paper is as follows. Section 2 introduces the current pruning methods. Section 3 presents the specific principle and implementation process for the proposed method. Section 4 discusses various experiments which were designed and conducted with the purpose of verifying the proposed method. Section 5 presents the conclusion of this paper and the proposals for future work.

2. Related Work

The network pruning can be traced back to the optimal brain damage approach proposed by Lecun et al. (1989), which could measure and prune low weights. Later, Hassibi & Stork (1992) proposed Optimal Brain Surgeon for removing unimportant weights determined by the second-order derivative information. The authors used the Hessian matrix to determine the importance of weight, which is a preliminary analysis in pruning. For unstructured pruning, some researchers regarded the parameters below a certain threshold as redundant parameters, which is a simple and effective pruning method. The sparse matrices generated by unstructured pruning slow down the network update and they are not conducive to the edge industrial control systems.

For structured pruning, the direction is relatively clear. The current research mainly focuses on formulating a more appropriate standard for evaluating the filters which are in convolutional layers. Li et al. (2016) proposed to select the filter based on the L1-norm and restore its performance through fine-tuning. This method is simple and effective, but it is difficult to apply to industrial control datasets. After some methods were put forward, researchers realized that better results could be obtained through global evaluation and dynamic pruning. For example, Wang et al. (2017) proposed global pruning. Abbasi-Asl & Bin (2021) proposed global importance indices to evaluate each filter. Luo et al. (2018) proposed to use the optimization algorithm to guide the selection of filters. Guo, Yao & Chen (2016) adopted the dynamic pruning method to recover the loss during pruning. However, with the proposed method focused on global evaluation and optimization, the cost of the pruning method is gradually increasing. So, Zhuo et al. (2018) proposed a pruning method based on sparse spectral clustering. Yang, Chen and Sze (2017) proposed that balancing the resources needed for calculation and data transfer could significantly reduce consumption. As deeper and more complex neural networks emerge, the pruning method for complex neural networks is also proposed. For example, He, Zhang & Sun (2017) used the LASSO constraint to select the coefficient channel to be deleted, which can be applied to the residual network. Wang et al. (2020) proposed a pruning method based on random weights. These pruning methods do not consider the complexity of pruning, so an efficient and effective pruning method is urgently needed at the edge with limited computing resources.

3. Methodology

3.1 Notation Explanation

In the process of pruning the intrusion detection classifier, certain notations were defined and their meaning was explained, as it is shown in Table 1.

3.2 The Pruning Process

The historical distribution of the parameters is analysed and manipulated as it is shown in Figure 1 and Table 2. The identities "S1", "S2" etc. in Figure 1 represent the steps for implementing the proposed method, and the identities in Table 2 represent the same. Equations for S2, S3, S4, S5, and S6 in Figure 1 are detailed in Section 3.3. The boxes in S1 represents the parameters of the filter. and in S1 and S6 represent the start and end times, respectively. The red cross in S7 is the deletion symbol, indicating that the filters corresponding to a low evaluation value are deleted after sorting.

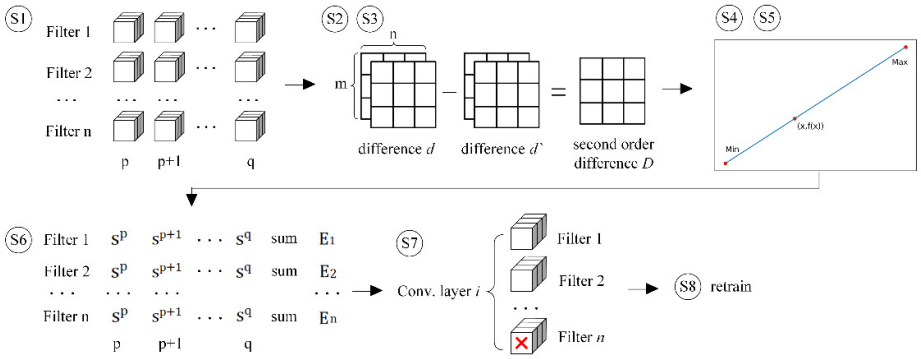

Figure 1. The proposed pruning method

Table 1. Notations and their meanings

Notations	Meanings
i^{th}	the convolution layer i
n^i	number of filters in the convolution layer i
x^i	the feature map input to the convolution layer i
x^{i+1}	the output feature map of the convolution layer i
h^i / w^i	the size of the feature map input to the convolution layer i
F_m^i	the filter m in the convolution layer i
$\omega_{i,j,k}$	the parameters of the position (i,j,k) in a filter
$d_{i,j,k}^t$	the first-order difference at the position (i,j,k) in the filter at time t
$D_{i,j,k}^t$	the second-order difference at the position (i,j,k) in the filter at time t
$W_{i,j,k}^t$	the weighted value at the position (i,j,k) in the filter at time t
$A_{i,j,k}^t$	the activity index at the position (i,j,k) in the filter at time t
E_i	the evaluation value of the filter i

Table 2. Pruning process

Input: weights of the model from p to q
S1: obtain the initial weight of the model
S2: calculate the first-order difference at the adjacent time
S3: calculate the second-order difference according to the first-order difference
S4: calculate the weighted value by weighting the first-order and second-order difference
S5: the activity index is obtained by mapping the weighted value
S6: calculate the evaluation value according to the activity index
S7: select the pruned filter according to the sort of the evaluation value
S8: retrain the pruned model
Output: pruned model

ICI Bucharest © Copyright 2012-2022. All rights reserved

3.3 The Pruning Theory

Assume that the given task is to prune filters in the convolutional layer i. For the convenience of narration, the convolution layer i is taken as an example without considering the pruning of other convolution layers. There are n^i filters in convolution layer i. x^i becomes x^{i+1} through n^i filters. The filter F_m^i consists of c convolutional kernels of size $a*b$. Therefore, the filter can be regarded as a block with side lengths a, b, and c, respectively. A filter contains $a*b*c$ parameters. So if a filter is pruned, the number of the parameters of the classifier will be reduced by $a*b*c$.

The first-order difference of weights at position (i,j,k) is obtained as it is shown in equation (1). The second-order difference $D_{i,j,k}^t$ is obtained by the first-order difference between times t and $t-1$, as it is shown in equation (2). After that, the weighted value $W_{i,j,k}^t$ can be obtained according to equation (3).

$$d_{i,j,k}^t = \left|\omega_{i,j,k}^t - \omega_{i,j,k}^{t-1}\right| (t \geq 3) \tag{1}$$

$$D_{i,j,k}^t = \left|d_{i,j,k}^t - d_{i,j,k}^{t-1}\right| (t \geq 3) \tag{2}$$

$$W_{i,j,k}^t = (1-\alpha) d_{i,j,k}^t + \alpha D_{i,j,k}^t \quad (t \geq 3, 0 < \alpha < 1) \tag{3}$$

The weighted value is obtained by giving the second-order difference weight. With the change of a weight in a filter, it is difficult to judge the effect only by the first-order difference, so we use the second-order difference to measure the change rate of weights. Assuming that a weight changes from value ω_1 to value ω_2 from time t_1 to t_2, its growth mode can be linear, in steps, oscillatory, etc. Different growth methods have a different significance for feature extraction in a limited time. Therefore, the method for obtaining the evaluation value through the first-order difference and the second-order difference is proposed. For α, the value 0.2 is taken.

The weighted value is converted into activity index through the Min-Max function, as it is shown in equation (4), which avoids the step of threshold selection. The selection of activity threshold directly affects the activity index of the filter. So under different activity thresholds, the filters may be different. To avoid the error caused by the activity threshold, the evaluation value is mapped to a value between 0 and 1, as it is shown in equation (5), where *minimum* represents the minimum weight of a filter between time p and time q, and *maximum* represents the maximum weight. Typically, p and q are considered the start and end time, respectively. However, different frameworks adopt different stochastic methods when initializing parameters, which makes parameter updating unstable. Therefore, increasing p appropriately may reduce the error of randomization when pruning. The calculation method is shown in equations (6) and (7). For the classifiers with lots of weights, such as the neural network, a simple mapping relationship can reduce the memory consumption in the edge industrial control system.

$$A_{i,j,k}^t = f(W_{i,j,k}^t) \tag{4}$$

$$f(x) = \frac{x - minimum}{maximum - minimum} \tag{5}$$

$$minimum = min\{W^p, W^{p+1}, ..., W^q\} \tag{6}$$

$$maximum = max\{W^p, W^{p+1}, ..., W^q\} \tag{7}$$

The filter is evaluated according to equation (8), where a, b, and c denote the size of a filter. p and q represent the starting and ending time, respectively.

$$E = \sum_{t=p}^{p+q-1} \sum_{k=1}^{c} \sum_{j=1}^{b} \sum_{i=1}^{a} A_{i,j,k}^t \tag{8}$$

$$Z = M' \otimes X + b \tag{9}$$

$$Y = \partial(Z) \tag{10}$$

All of E make up a set $N = \{E_1, E_2, ..., E_n\}$. The top K values in N are set to 1 and the rest to 0. This 0-1 set can be expressed as N'. Essentially, N' is a mask set, where 1 means to retain the filter, and 0 means to remove it. All the parameters of the filter compose the tensor M, and the shape of M is $n*(a*b*c)$. After pruning, M becomes M', and the shape of M' is $K*(a*b*c)$. Equations (9) and (10) represent the forward propagation process for the convolutional network (Lecun et al., 1998). X represents the input, b denotes bias (a constant for promoting the model fitting), $\otimes$ denotes the convolutional operator and Z represents the output of convolutional layer. ∂ represents the activation function, which enables the neural network to fit the nonlinear function. For example, the Sigmoid function shown in equation

(11) is a common activation function, which was first used by Lecun et al. (1998). Y denotes the output of the activation function.

$$\partial(Z) = \frac{1}{1+e^{-Z}} \tag{11}$$

3.4 Pruning for Special Networks

If there are special structures such as basic blocks, bottleneck blocks (which comes from ResNet) or dense blocks (which comes from DenseNet) in the network, the pruning method needs to be adjusted according to the special blocks.

In the case of basic blocks and bottleneck blocks, there is a shortcut for each block that passes the information to the convolution layer behind. It is necessary to ensure that the number of pruning filters in the convolutional layer on the shortcut path be the same as that in the last convolution layer on the forward propagation path. The number of pruning filters in the first convolutional layer on the forward propagation path is optional. The above is shown in Figures 2 and 3. The rectangles in these figures represent the convolution layers. Shadows indicate that the number of filters in the different convolutional layers is the same, and the absence of shadows indicates that the number of filters in the different convolutional layers could be different.

In the case of DenseNet, the information fusion method for dense blocks is concat, which is different from that of basic blocks or bottleneck blocks. So the number of filters does not need to remain the same, but it is preferred to keep the same number of filters to maintain a stable information transmission, as it is shown in Figure 4.

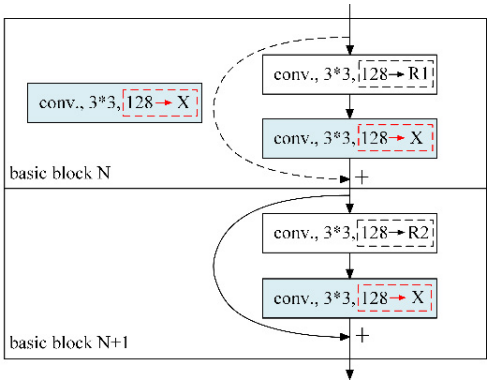

Figure 2. Pruning for basic blocks

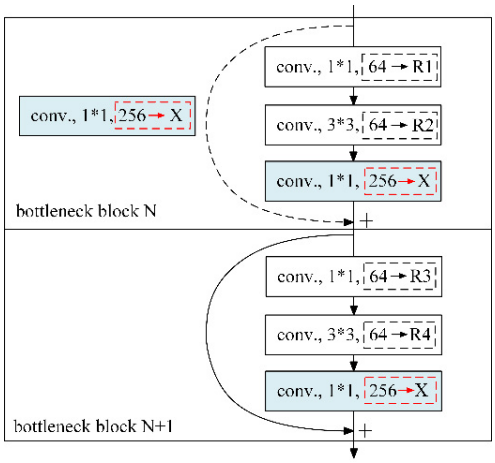

Figure 3. Pruning for bottleneck blocks

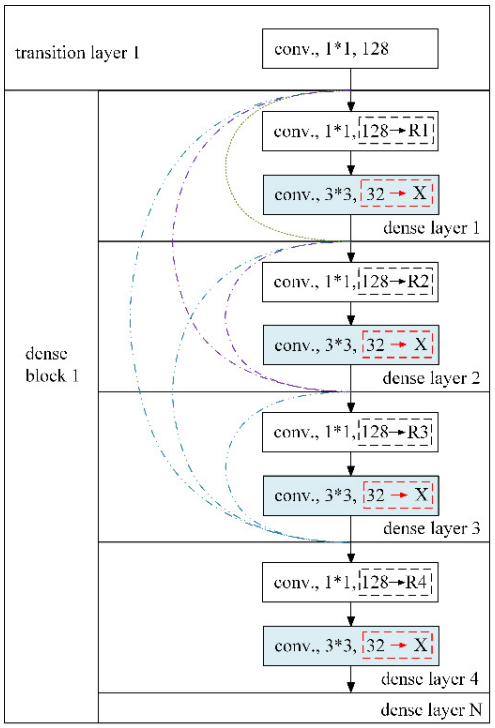

Figure 4. Pruning for dense blocks

4. Experiments

4.1 Preparation

The experiments were run in Windows 10 with CPU AMD R7 4800H, GPU NVIDIA GeForce RTX 2060, and 16G memory. The employed program was Python 3.7. The deep learning framework were PyTorch 1.9.0 and TorchVision 0.10.0.

The experiments only verified the pruning effect. The network was pruned when the loss converged and the accuracy was within an acceptable range.

ICI Bucharest © Copyright 2012-2022. All rights reserved

The evaluation method for pruning included the parameters (Paras), the FLOPs, and the accuracy change. The fully connected layers and the feature engineering such as feature selection or data preprocessing were not considered.

Three datasets were used, namely the DDoS dataset, the gas dataset, and the oil dataset. The gas dataset is from the Mississippi State University Key Infrastructure Protection Center for intrusion detection and assessment in industrial control systems. These datasets can be used to help researchers evaluate the performance of supervisory control and data acquisition intrusion detection system (SCADA IDS) by using actual SCADA attack modes. The CICDDoS2019 dataset contains benign and the most up-to-date common DDoS attacks, which resemble real-world data. It also includes the results of the network traffic analysis using CICFlowMeter-V3 with labeled flows based on the time stamp, source, and destination IPs, source and destination ports, protocols, and attacks. In the oil dataset, there are 131 attributes, including inlet pressure of oil pump, outlet pressure of oil pump, etc. The attack types are set to seven, as it is shown in Table 3.

4.2 LeNet-5 on Industrial Datasets

The pruning effect of LeNet-5 convolutional neural network was verified on the basis of three industrial datasets, and the experimental results are shown in Figure 5 and Table 4. The figures labeled by A, B, and C illustrate the results of LeNet-5 for the DDoS dataset, the gas dataset, and the oil dataset, respectively. The sequences of figures labeled by 1 (A-1, B-1 and C-1) show the trends for Acc Top 1 and the loss during pruning.

Table 3. Abnormal states of an oil depot

Number	Events	Results
1	Increase / decrease the integral coefficient of the pipeline system.	The pressure on a device is out of balance in the process of transporting oil.
2	Distort the adjustment mode of the pipeline system from automatic to manual.	The pressure on a device is out of balance in the process of transporting oil.
3	Distort the integral coefficient of the pipeline system frequently.	The pressure on a device fluctuates up and down in the process of transporting oil.
4	Distort the output pressure of the pipeline system.	The alarm system is activated. The pressure on a device is abnormal.
5	Turn on the switch to fill up a device with oil when it is not necessary.	The pressure on a device increases sharply.
6	Distort the frequency of variable frequency pump in a pipeline system.	The pressure on a device is out of balance in the process of transporting oil.
7	Distort the oil flow in the process of receiving oil.	The data is abnormal when a device is transporting oil.

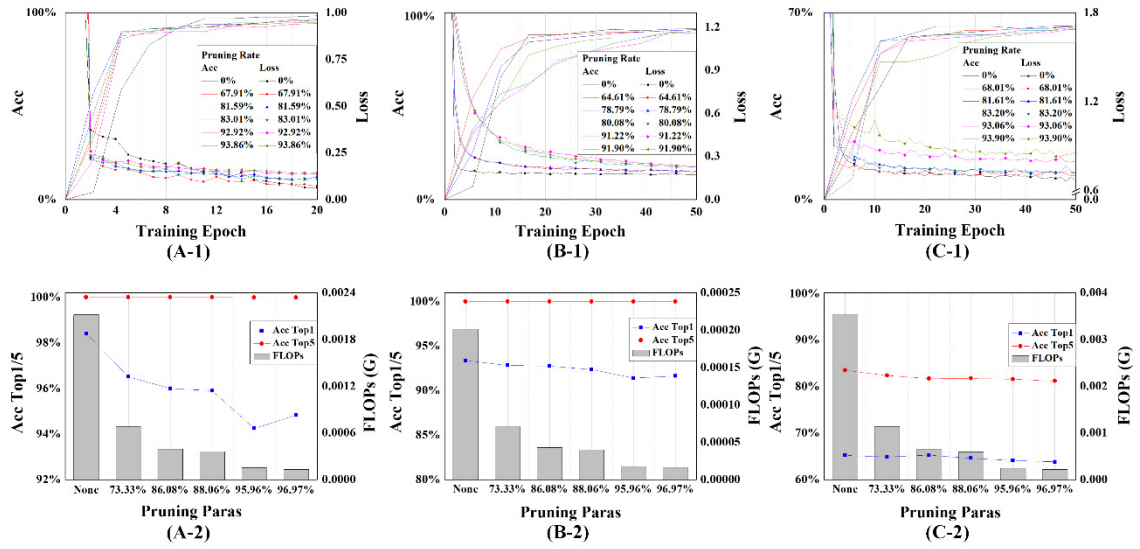

Figure 5. The trends for various indices in the experiments based on LeNet-5

Table 4. The final results in the experiments based on LeNet-5

LeNet-5 on DDoS Dataset	Pruning FLOPs (%)	None	67.91	81.59	83.01	92.92	93.86
	Δ Acc Top1 (%)	98.40	-1.89	-2.4	-2.5	-4.16	-3.57
	Δ Acc Top5 (%)	100.00	0.00	0.00	0.00	-0.01	-0.01
	Loss	0.0641	0.0723	0.1213	0.1097	0.1345	0.1435
LeNet-5 on Gas Dataset	Pruning FLOPs (%)	None	64.61	78.79	80.08	91.22	91.90
	Δ Acc Top1 (%)	93.38	-0.55	-0.62	-1.01	-2.34	-1.75
	Δ Acc Top5 (%)	100.00	0.00	0.00	0.00	0.00	0.00
	Loss	0.1714	0.1915	0.1975	0.2283	0.2304	0.2337
LeNet-5 on Oil Dataset	Pruning FLOPs (%)	None	68.01	81.61	83.20	93.60	93.90
	Δ Acc Top1 (%)	65.20	-0.33	+0.06	-0.62	-1.12	-1.40
	Δ Acc Top5 (%)	83.50	-1.12	-1.78	-1.74	-1.92	-2.28
	Loss	0.6619	0.6982	0.7155	0.7201	0.7988	0.8567

The horizontal axis represents the training epoch. The left vertical axis represents the accuracy. The right vertical axis represents the loss. The sequences of figures labeled by 2 (A-2, B-2 and C-2) show the variations of Acc Top1, Acc Top5, and FLOPs. The horizontal axis represents the percentage of pruning parameters. The left vertical axis represents the Acc Top1 and the Acc Top5. The right vertical axis represents the pruning FLOPs. Table 4 shows the final results for the pruned FLOPs, Acc Top1, Acc Top5, and loss for different pruning rates. The meanings of the experimental diagram in sections 4.3 and 4.4 are the same as those described in this section.

In the experiments on the DDoS dataset, when the pruning rate reaches 67.91%, the amount of parameters is reduced by 73.33%, and Acc Top1 is only reduced by 1.89%. In the experiments on the gas dataset, when the pruning rate reaches 91.90%, the amount of parameters is reduced by 96.97%, and Acc Top1 is only reduced by 1.75%. The experiments on the oil dataset show that the accuracy may increase when redundant parameters are pruned. For example, the accuracy (Acc Top1) increased by 0.06%, when the amount of parameters was reduced by 81.61%.

4.3 VGG16 on Industrial Datasets

The pruning effect of VGG16 was verified based on three industrial datasets, and the experimental results are shown in Figure 6 and Table 5.

The experiments on the DDoS dataset and the gas dataset show that there are many redundant parameters in the case of VGG16. The fluctuations of Acc Top1 do not exceed ±2.45%. In the experiments on the oil dataset, when the pruning rate reaches 97.61%, the Acc Top1 is only reduced by 0.82%.

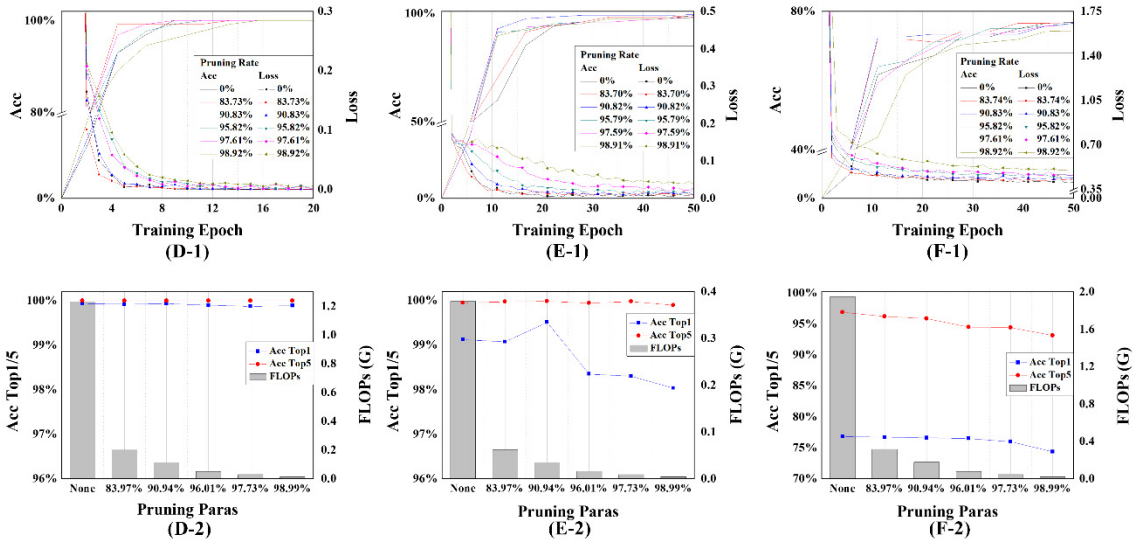

Figure 6. The trends for various indices in the experiments based on VGG16

ICI Bucharest © Copyright 2012-2022. All rights reserved

Table 5. The final results in the experiments based on VGG16

VGG16 on DDoS Dataset	Pruning FLOPs (%)	None	83.73	90.83	95.82	97.61	98.92
	Δ Acc Top1 (%)	99.94	-0.02	-0.01	-1.09	-0.07	-0.05
	Δ Acc Top5 (%)	100.00	0.00	0.00	0.00	0.00	0.00
	Loss	0.0001	0.0002	0.0002	0.0002	0.0011	0.0025
VGG16 on Gas Dataset	Pruning FLOPs (%)	None	83.70	90.82	95.79	97.59	98.91
	Δ Acc Top1 (%)	99.13	-0.06	0.38	-0.78	-0.83	-1.09
	Δ Acc Top5 (%)	99.95	+0.02	+0.03	-0.01	+0.03	-0.05
	Loss	0.0108	0.0090	0.0089	0.0267	0.0280	0.0382
VGG16 on Oil Dataset	Pruning FLOPs (%)	None	83.74	90.83	95.82	97.61	98.92
	Δ Acc Top1 (%)	76.83	-0.10	-0.21	-0.30	-0.82	-2.45
	Δ Acc Top5 (%)	96.91	-0.69	-1.03	-2.41	-2.48	-3.76
	Loss	0.3935	0.4212	0.4390	0.4664	0.4586	0.5046

4.4 ResNet on Industrial Datasets

The pruning effects of ResNet18 and ResNet50 were verified for the oil datasets, and the experimental results are shown in Figure 7 and Table 6.

The experiments of ResNet18 on the oil dataset show that the amount of parameters is reduced by 83.9%, and the Acc Top1 is only reduced by 2.28% when the pruning rate reaches 83.16%. When 95.62% of the FLOPs of ResNet50 were pruned, the Acc Top1 was not reduced by more than 2%.

The proposed method was compared with two other methods for the oil dataset, as it is shown in Table 7. L1-Norm prunes filters by calculating the absolute difference of weights. HRank prunes filters by calculating the average rank of multiple feature maps. The features of the industrial control dataset were analysed, in order to better extract the abnormal features and prune the redundant parameters. As it can be seen, the proposed method performed better than the other two methods.

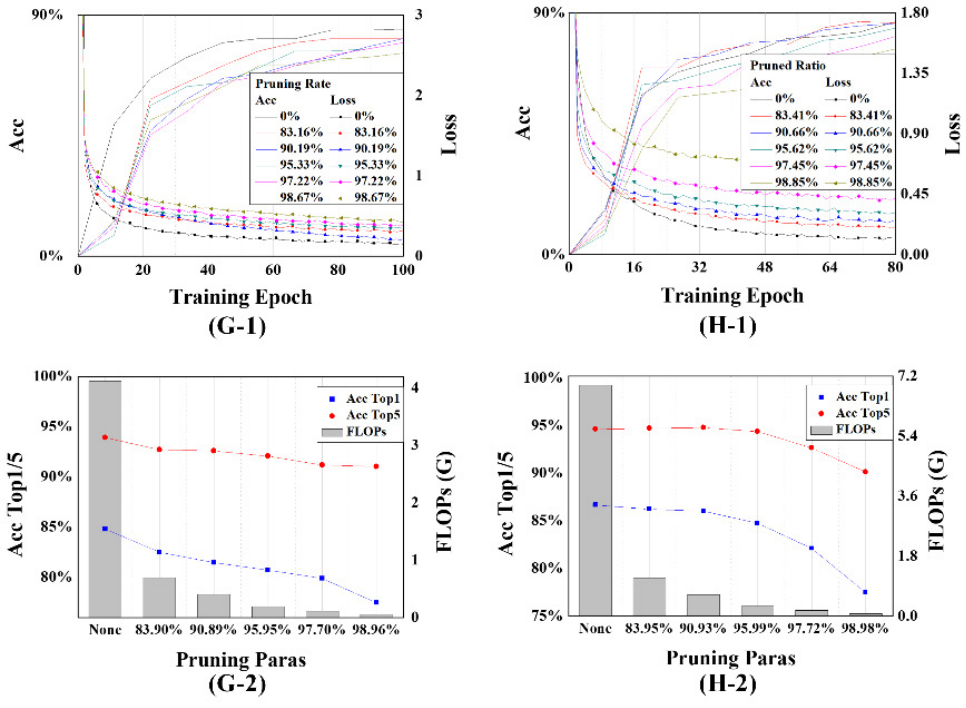

Figure 7. The trends for various indices in the experiments based on ResNet18 and ResNet50

Table 6. The final results in the experiments based on ResNet18 and ResNet50

ResNet18 on Oil Dataset	Pruning FLOPs (%)	None	83.16	90.19	95.33	97.22	98.67
	Δ Acc Top1 (%)	84.79	-2.28	-3.33	-4.07	-4.91	-7.33
	Δ Acc Top5 (%)	93.91	-1.20	-1.32	-1.85	-2.73	-2.89
	Loss	0.1483	0.3090	0.2118	0.3373	0.3838	0.4236
ResNet50 on Oil Dataset	Pruning FLOPs (%)	None	83.41	90.66	95.62	97.45	98.85
	Δ Acc Top1 (%)	86.65	-0.43	-0.64	-1.90	-4.54	-9.18
	Δ Acc Top5 (%)	94.62	+0.08	+0.15	-0.26	-1.96	-4.50
	Loss	0.1166	0.2099	0.2584	0.3137	0.4183	0.6360

Table 7. Comparison of the proposed method with other methods

Experiments	**Pruning FLOPs (%)**	**Δ Acc Top1 (%)**		
		L1-Norm	**HRank**	**The Proposed Method**
LeNet-5 on Oil Dataset	83.2	-6.7	-1.88	-0.62
VGG16 on Oil Dataset	95.82	-4.43	-1.65	-0.3
ResNet18 on Oil Dataset	83.16	-8.92	-2.33	-2.28
ResNet50 on Oil Dataset	95.62	-6.83	-3.13	-1.9

5. Conclusion

In order to reduce parameters and FLOPs of the neural networks and alleviate load pressure on the edge industrial control systems, this paper proposes the pruning of filters through hybrid-order difference based on Poisson distribution. The experiments carried out show that the proposed method can prune various intrusion detection classifiers at a high rate and make the pruned classifier perform excellently in the edge industrial control system with limited resources, which may have a certain guiding significance for the construction of lightweight industrial control systems. Future work may focus on analyzing the distribution of parameters on the overall experimental results by adjusting the different pruning rates for different convolutional layers.

Acknowledgements

This research was funded by means of two projects, that is BIPTACF-008 and the National Science and Technology Major Project of the Ministry of Science and Technology of China (No.2018AAA0102900).

REFERENCES

Abbasi-Asl, R. & Bin, Y. (2021). Structural Compression of Convolutional Neural Networks with Applications in Interpretability, *Frontiers in Big Data*, *4*: 704182. DOI: 10.3389/fdata.2021.704182

Globa, L. S., Demidova, Y. A. & Ternovoy, M. Y. (2006). Network Anomaly Detection using Neural Networks. In *2006 16th International Crimean Microwave and Telecommunication Technology* (pp. 412-413). DOI: 10.1109/CRMICO.2006.256445

Guo, Y., Yao, A. & Chen, Y. (2016). Dynamic Network Surgery for Efficient DNNs. In *Proceedings of the 30th International Conference on Neural Information Processing Systems (NIPS)*, (pp. 1387-1395).

Hassibi, B. & Stork, D. G. (1992). Second order derivatives for network pruning: optimal brain surgeon, *Advances in Neural Information Processing Systems*. Morgan Kaufmann Publishers.

He, Y., Zhang, X. & Sun, J. (2017). Channel Pruning for Accelerating Very Deep Neural Networks. In *2017 International Conference on Computer Vision (ICCV)*, (pp. 1398-1406). IEEE Computer Society. DOI: 10.1109/ICCV.2017.155

Lecun, Y., Bottou, L., Bengio, Y. & Haffner, P. (1998). Gradient-based learning applied to document recognition, *Proceedings of the IEEE*, *86*(11), 2278-2324. DOI:10.1109/5.726791

Lecun, Y., Denker, J. S. & Solla, S. A. (1989). Optimal brain damage, *Advances in Neural Information Processing Systems*, *2*(279), 598-605.

Li, H., Kadav, A., Durdanovic, I., Samet, H. & Graf, H. P. (2016). Pruning filters for efficient ConvNets. In *5th International Conference on Learning Representations (ICLR)*, (pp. 1-13). DOI: 10.48550/arXiv.1608.08710

ICI Bucharest © Copyright 2012-2022. All rights reserved

Liu, J., Guo, J., Orlik, P., Shibata, M., Nakahara, D., Mii, S. & Takáč, M. (2018). Anomaly Detection in Manufacturing Systems Using Structured Neural Networks. In *2018 13th World Congress on Intelligent Control and Automation (WCICA)*, (pp. 175-180). DOI: 10.1109/WCICA.2018.8630692

Liu, X., Wang, H., Zhang, X., Luan, H., Sha, Y. & Yan, Y. (2021). A Method Based on Multiple Population Genetic Algorithm to Select Hyper-Parameters of Industrial Intrusion Detection Classifier, *Studies in Informatics and Control*, *30*(3), 39-49. DOI: 10.24846/v30i3y202104

Luo, J., Zhang, H., Zhou, H., Xie, C., Wu, J. & Lin, W. (2018). ThiNet: Pruning CNN Filters for a Thinner Net, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *41*, 2525-2538.

Mansouri, Y. & Babar, M. A. (2021). A review of edge computing: Features and resource virtualization, *Journal of Parallel and Distributed Computing*, *150*, 155-183. DOI: DOI: 10.1016/j.jpdc.2020.12.015

Ning, H., Li, Y., Shi, F. & Yang L. (2020). Heterogeneous edge computing open platforms and tools for internet of things, *Future Generation Computer Systems-The International Journal of eScience*, *106*, 67-76. DOI: 10.1016/j.future.2019.12.036

Shen, B. & Ge, Z. (2021). Weighted Nonlinear Dynamic System for Deep Extraction of Nonlinear Dynamic Latent Variables and Industrial Application, *IEEE Transactions on Industrial Informatics*, *17*(5), 3090-3098. DOI: 10.1109/TII.2020.3027746

Shi, W., Cao, J., Zhang, Q., Li, Y. & Xu, L. (2016). Edge Computing: Vision and Challenges, *IEEE Internet of Things Journal*, *3*(5), 637-646. DOI: 10.1109/jiot.2016.2579198

Song, J., Chen, Y., Ye, J. & Song, M. (2022). Spot-Adaptive Knowledge Distillation, *IEEE Transactions on Image Processing*, *31*, 3359-3370. DOI: 10.1109/TIP.2022.3170728

Taneja, M., Byabazaire, J., Jalodia, N., Davy, A., Olariu, C. & Malone, P. (2020). Machine learning based fog computing assisted data-driven approach for early lameness detection in dairy cattle, *Computers and Electronics in Agriculture*, *171*, 105286. DOI: 10.1016/j.compag.2020.105286

Wang, Y., Chen, X., Wang, F., Song, M. & Yu, C. (2022). Meta-Learning Based Hyperspectral Target Detection Using Siamese Network, *IEEE Transactions on Geoscience and Remote Sensing*, *60*, 1-13. DOI: 10.1109/TGRS.2022.3169970

Wang, Y., Zhang, X., Xie, L., Zhou, J., Su, H, Zhang, B. & Hu, X. (2020). Pruning from scratch. In *Proceedings of the AAAI Conference on Artificial Intelligence*, *34*(7), (pp. 12273-12280).

Wang, Z., Zhu, C., Xia, Z., Guo, Q. & Liu, Y. (2017). Towards thinner convolutional neural networks through gradually global pruning. In *2017 International Conference on Image Processing (ICIP)*, (pp. 3939-3943). IEEE. DOI: 10.1109/ICIP.2017.8297021

Yang, T. J., Chen, Y.-H. & Sze, V. (2017). Designing Energy-Efficient Convolutional Neural Networks using Energy-Aware Pruning. In *2017 Conference on Computer Vision and Pattern Recognition (CVPR)*, (pp. 6071-6079). IEEE. DOI: 10.1109/CVPR.2017.643

Yang, R., Lu, X., Huang, J., Zhou, J., Jiao, J., Liu, Y., Liu, F., Su, B. & Gu, P. (2021). A Multi-Source Data Fusion Decision-Making Method for Disease and Pest Detection of Grape Foliage Based on ShuffleNet V2, *Remote Sensing*, *13*(24). DOI: 10.3390/rs13245102

Zhou, W., Kong, X., Li, K., Li, X., Ren, L., Yan, Y., Sha, Y., Cao, X. & Liu, X. (2021). Attack sample generation algorithm based on data association group by GAN in industrial control dataset, *Computer Communications*, *173*(9), 206-213. DOI: 10.1016/j.comcom.2021.04.014

Zhuo, H., Qian, X., Fu, Y., Yang, H. & Xue, X. (2018). SCSP: Spectral Clustering Filter Pruning with Soft Self-adaption Manners, ArXiv, abs/1806.05320. DOI: 10.48550/arXiv.1806.05320