Analysis of Pruning Optimization Technology Based on Deep Learning

Tian Qin [1, †], Jiang Zhang [2, *, †], Xihua Zhu [3, †]

[1] International education, Nanchang HangKong University, Nanchang Jiangxi, 330063, China

[2.]Electronic engineering, King's College London, London, United Kingdom

[3.]Computer and softerware academy, Dalian Neusoft University of information, Dalian Liaoning, 11600, China

* Corresponding Author Email: K21026978@kcl.ac.uk

[†]These authors contributed equally.

Abstract. Neural network models have become a major topic in many areas in recent years due to their excellent problem-solving ability. However, during the training process in the real application, neural network models usually have relatively high requirements on the performance of the platform, which limits their applications, especially in battery-driven devices or embedded platforms with relatively weak performance. Therefore, the importance of optimizing and pruning neural network models is very important. This paper analyzes four categories of pruning techniques, including channel pruning, neuron pruning, weight pruning and layer pruning. This paper investigates its main ideas and particular pruning methods and compares the optimised effects of these pruning techniques in different neural network models through experiments. This paper discusses neuron pruning on the VGG-16 model, weight pruning on ResNet and AlexNet, and layer pruning on the YOLOV3 model, respectively. The parameter amount and inference speed before and after pruning are compared, and the accuracies drop before and after pruning are verified on data sets such as CIFAR-10.

Keywords: channel pruning, Neuron pruning, Weightpruning, layer pruning

1. Introduction

For the past few years, Convolutional Neural Networks (CNN) have acted in computer vision tasks, image classification, etc. with increasing importance. However, the powerful performances bring a range of problems, such as long computing time, large model size, high power consumption, etc.Due to these problems, CNN is hard to be employed in embedded systems. Therefore, pruning, as an effective method to reduce the model size, was proposed in Mozer,M. C. and Smolensky, P.'s research for the first time. With all kinds of emerged pruning, pruning has become a popular step to optimize the CNN model.

This paper mainly introduces a set of pruning approaches, which include structured pruning and unstructured pruning. They are different in the objects of pruning and stage. This paper lists the performances of these different pruning methods used on different datasets and models. This paper presents that these pruning methods all can reach great success in reducing the model size and have their own strengths.

2. Channel Pruning

Channel pruning is a type of structured pruning. As the name suggests, it prunes one or serval groups of convolutional channels to reduce the model size. Since the target of channel pruning is the channel, we are supposed to propose a bunch of standards to judge and weigh the importance or necessity of each channel and then prune the channels, which are less important among the whole neural network.

As for the standards or ways to weigh the channel, it has a lot of kindssuch as LASSO regression5[1], greedy algorithm [2], etc. In this paper, we mainly introduce a notion called "scaling factor," which is proposed for in-network slimming [3].

As seen in Figure 1[1], we imposed scaling factors gamma on every channel of batch normalization (BN), then rank these scaling factors, and finally prune the corresponding channels. Their scaling factors are relatively small according to the pruning ratio. Before the pruning, the model needs to go through sparsity training. It uses the L1 regularization on scaling factors aiming to make the gamma toward zero, which enables us to identify the insignificant channels. Namely, the closer the gamma is to 0, the less important the corresponding output of that channel is to the result. The loss function of this network is presented in Formula 1:

$$L = \sum_{(x,y)} l(f(x,W),y) + \lambda \sum_{\gamma \in \Gamma} g(\gamma) \text{ [1]} \quad (1)$$

$\sum_{(x,y)} l(f(x,W),y)$is the common networks' loss function, and $\lambda \sum_{\gamma \in \Gamma} g(\gamma)$is the regularization of the scaling factors. γ is the scaling factor and λis the penalty's sparsity.

As the structured pruning, channel pruning has advantages over the unstructured pruning. Although the unstructured pruning can decline dramatically parameters and FlOPsofthemodel, it needs particular hardware to accelerate, while the structured pruning can exploit the existing hardware configuration.

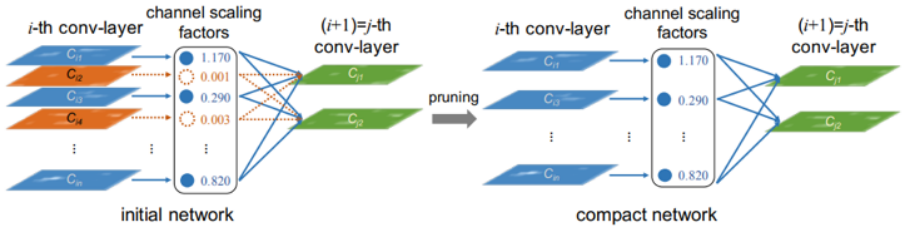

Figure 1 Pruning process [3]

3. Neuron pruning

Neuron pruning has been developed for many years. In the 2010s, it was demonstrated that neural nets with many layers, neurons, and weights may attain state-of-the-art performance in a variety of applications when trained on big data sets utilizing GPUs and employing a huge net with many layers, neurons, and weights [4]. In this way, we can compress the large networks and use them on mobile phones or other portable devices. Therefore, people pay more attention to the neural network. Neural network pruning removes unnecessary neurons and weights from a big network before learning the proper amount of neurons and weights on its own or avoiding overfitting while keeping the network's performance as much as feasible, as seen in Figure 2. For the network to reach the same performance, weight initialization is required to improve pruning efficiency. When we train a trimmed network from the ground up, we find that it has a higher proportion of zero activation neurons than the weight initialised version. This means that a retrained network that hasn't had its weights initialised is far less efficient. It has been discovered that pruning or compressing a huge network is significantly easier than writing a tiny network directly.

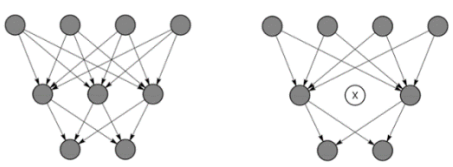

Figure 2 Neuron pruning process [4]

The enormous quantity of parameters in deep neural networks has resulted in substantial redundancy in several deep learning models. Not only does an over-parameterized model waste

memory and computing resources, but it also leads to a substantial overfitting problem. As a result, several scholars in this subject have looked toward lowering the number of factors. However, the stud on optimising the number of neurons still exists. The majority of prior efforts to improve network architectures have been divided into two categories: high-level architectural design and low-level weight trimming [5]. There are 13 convolutional layers and three fully linked layers in the VGG-16 network. For non-linear mapping, each of the layers is followed by a ReLU layer. The VGG-16 is regarded as a typical network that has been used in a variety of applications, including object categorization and localisation. As a result, this model is ideal for neuron pruning experiments [5].

4. Weight pruning

Weight pruning is an optimization method that reduces the number of neural network parameters and increases performance by removing the unnecessary weight parameters. Neural networks often contain too many parameters, which causes the neural network to perform a lot of unnecessary computations and take up unnecessary memories. A simple weight pruning method is proposedbyHan, S., Pool, J., Tran, J. and Dally, W [6], which uses three steps. The first step is to learn the importance of connections. The second step is to remove those with weights below the threshold, and the third step is to learn the parameters of each connection. In this way, a dense neural network is successfully converted into a sparse network. This simple method has been extended and used. One of the energy-aware pruning methods [7] can effectively remove those non-zero weights. As a result, following pruning, AlexNet's energy consumption can be decreased by 3.7 times, which is 1.7 times less than pruning utilizing other prominent network pruning approaches. Even for a compact CNN model like GoogLeNet, this pruning strategy can save the 1.6 times amount of energy. Thismethodcan significantly lowertheenergy consumption of the neuralnetworkmodel,whichhighlybenefitedthedeploymentandapplicationof battery-powerdevicesforNNs.Another pruning method proposed by Wenet al.[8] introduced the Struct Sparsity Learning method[8]. This pruning method kept the structure of the neural. This SSL method uses off-the-shelf libraries to speed up the computation of AlexNet's CPU and GPU convolutional layers by an average of 5.1x and 3.1x, respectively. These are about twice as fast as non-structured sparsity. It also improves classification accuracy by regularizing the DNN structure. Regularization on layer depth reduces the 20-layer Deep Residual Network (ResNet) to 18 layers for CIFAR-10, while boosting accuracy from 91.25 per cent to 92.60 per cent, which is still greater than the original ResNet' with 32 layers. The SSL technique decreases errors by 1% for AlexNet. By usingtheweightpruningmethods, the inferring speed, memory usage, energy consumption andotherperformancecanbesignificantlyboosted.

5. layer pruning

During the development process of the CNN model, a unique but useful structure, called "shortcut" or "short path", has been purposed by Kaiming He, who won the 2017 ImageNet Prize. To deal with the degeneration problem in the Deep Learning model, the Resnet model was published by Kaiming He and his team members in 2015. In the Resnet team's experiment, they found a very interesting phenomenon. With the constantly increasing depth of the Resnet model, the accuracy of the Resnet model didn’t improve with the depth. It gradually decreased when the depth was deeper. This phenomenon is a conflict with the ideal Deeper Neural Networks that want to be more accurate. Thus, they call this phenomenon Degeneration. In the past, people believed that complex Neural Networks with a deeper depth can map data to higher-level to achieve data classification through a series of activation and feature extraction. However, the Resnet team concluded that complex Neural Networks cannot handle Identity mapping (y = x). Such huge data needs an amount of math to calculate, and it exceeds the processing power of Neural Networks. The recent team wants to solve the problem, so they put the concept of "shortcut connection" and simulate Alexnet Network to establish a Resnet

network model with shortcut connection through eight-building layers that contain residual blocks to finish down sampling.

As a Convolutional Neural Network, its memory size, accuracy and amount of calculation strictly limit to deploy models to mobile devices. However, pruning technology can use some algorithms or rules to delete some useless layers and parameters in Neural Networks to make sure that CNN models are more compact with less computational and smaller search complexity and can work on edge devices better. Normally, pruning technology has four types, including weight-level, kernel-level, channel-level and layer-level. Among these pruning technologies, layers pruning can compress the length of the Neural Network. In the Darknet, the structure hierarchy will evaluate convolution, batch normalization, and LeakyReLU before the shortcut layer, rank the γ value [11] of each layer, and then prune the layer with the lowest value. In addition, such methods are always used for YOLOV3 Neural Network, and it can reduce the length of the Neural Network to a large extent. To maintain the model's integrity, the shortcut layer and two convolutional layers should be pruned at the same time. The length of the YOLOV3 Neural Network can be compressed.

6. Case study

6.1. neuron pruning in modelVGG-16

To determine the proportion of zero activations in a neuron following ReLU mapping, first, compute the average percentage of zeros using Formula 2.

$$\mathrm{APoZ}_c^{(i)} = \mathrm{APoZ}\left(O_c^{(i)}\right) = \frac{\sum_k^N \sum_j^M f(O_{c,j}^{(i)}(k)-0)}{N \times M} \tag{2}$$

The relevance of each neuron in the network can be assessed based on the APoZ concept. The APoZ of each neuron was computed to demonstrate that the reported output of certain neurons in big networks is mainly zero, and it was discovered that 631 neurons in the VGG-16 network had an APoZ of more than 90% [4].

The network pruning method consists of three main steps, including Conventional Training, Prune Neurons, and Weight Initialization. The last two steps are repeated.

After 6 iterations of trimming, the model eliminates more than half of the total number of parameters and achieves a compression rate of 2.59, with the trimmed network having 23% higher Top-1/Top-5 accuracy than the original VGG-16 model.

Although around 500 neurons in CONV5-3 and F C6 were pruned throughout each iteration, the accuracy did not decline much, as presented in Table 1. Rather than a week of retraining, this modest loss inaccuracy may be compensated with rapid fine-tuning. Finally, the pruning accuracy surpasses the original data, and there are fewer parameters. The prior model's strong initialization provides a promising starting point for the reduced model. Furthermore, having fewer parameters in FC6 also decreases the possibility of overfitting, which can improve accuracy [4].

Table 1. 6 iterations result [4]

Network (CONV5-3,FC6)	Compression Rate	Before Fine-tuning(%)		After Fine-tuning(%)	
		Top-1 Accuracy	Top-5 Accuracy	Top-1 Accuracy	Top-5 Accuracy
(512,4096)	1.00	68.36	88.44	68.36	88.44
(488,3477)	1.19	64.09	85.90	71.17	90.28
(451,2937)	1.45	66.77	87.57	71.08	90.44
(430,2479)	1.71	68.67	89.17	71.06	90.34
(420,2121)	1.96	69.53	89.49	71.05	90.30
(400,1787)	2.28	68.58	88.92	70.64	89.97
(390,1513)	2.59	69.29	89.07	70.44	89.79

6.2. pruning technology based on VGG16/ResNet-50

This paper introduces a bunch of methods of channel pruning. To evaluate their performance, we tested the different pruning models using ThiNet, as we explained in this paper on ImageNet. This test used the pre-trained model on resized centre-cropped images, which would serve to imply the accuracy of the original VGG-16 model. The test on ResNet also followed this tactic.

Based on Table 2, we can see that the ThiNet-GAP only has 8.32M parameters, which is far smaller than the original with 138.34M parameters. Although the channel pruning dramatically reduced model size, the Top-1/Top-5 accuracies are 67.34% and 87.92% respectively, which is little difference compared to the original VGG-16 model. The speed-up is also presented in Table 3, the f./b. (forward/backwards running time) of the original model are 189.92ms/407.56ms respectively. However, the ThiNet-GAP's f./b. is 71.73ms/145.51ms, which demonstrates the pruned model has a greater speed.

Table 2. Pruning results of VGG-16 on ImageNet using ThiNet [2].

Model	Top-1	Top-5	#Param.	#FLOPs	f./b. (ms)
Original	68.34%	88.44%	138.34M	30.94B	189.92/407.56
ThiNet-Conv	69.80%	89.53%	131.44M	9.58B	76.71/152.05
Trainfrom scratch	67.00%	87.45%	131.44M	9.58B	76.71/152.05
ThiNet-GAP	67.34%	87.92%	8.32M	9.34B	71.73/145.51
ThiNet-Tiny	59.34%	81.97%	1.32M	2.01B	29.51/55.83
SqueezeNet	57.67%	80.39%	1.24M	1.72B	37.30/68.62

6.3. Extreme pruning technology based on the YOLOV3 model

In the YOLOV3 method, a new Neural backbone with a shortcut connection called Darkent-53 has been designed. The Darknet-53 model's [11] Convolutional layers include the Conv2D layer, the BN layer, and the LeakyReLU layer. Now, Darknet-53 is used as the backbone of the YOLOV3 convolutional neural network. As the most fully convolutional neural network, YOLOV3 doesn't have a max-pooling layer but uses shortcuts and residuals [9][10] to form the residual blocks. The residual blocks are stacked to finish the down sampling process and to reduce the image's height and width. Through using shortcut connection and residual blocks, YOLOV3 can decrease the data transmission in the process of Forwarding calculations and Backpropagation.

The YOLOV3 model has some changes to a large extent compared with the YOLOV1 and YOLOV2 models. As a One-stage Neural Network [9], YOLOV3 has a higher object detection speed, but its detection accuracy still needs to enhance. Thus, it adopts the Feature Pyramid Network (FPN) structure for extracting feature maps, in which images that are inputted into the Neural Network need to go through a series of complex processes. First, images will go through a down sampling process from top to bottom, these images' height and width will shrink to 13×13, and their channels will be expanded to 1024[11]. This data collected through an up sampling process from bottom to top can expand images' height and width. After these procedures, the entire model can get some low-level features and high-level features, however, such features cannot improve the accuracy of the Neural Network. Therefore, these features need to through lateral connection, and low-level features need to be fused with other features as a stage in the same layer to increase the model's accuracy. Besides,

each layer needs to make a separate prediction. A one-stage model structure like YOLOV3 can use Feature Pyramid Network to improve the accuracy rate of the whole network. Furthermore, when data is Backpropagation [11], these shortcuts that are established between the networks can reduce the impact of the parameter. Such structure can help prevent the entire network's degeneration with the depth of the model.

6.4. Weight pruning on LeNet models

To have a better neural network structure and make the entire model has a faster training speed with smaller memorization on the hard devices, we may refer to the weight pruning technologies based on the LeNet model trained on the ministry database. In this network structure, we will receive a series of different parameters after every training. However, among these parameters, the magnitude of the change related to the parameters is very tiny, which means that such parameters are not necessary for the LeNet model. We can delete them to reduce the volume of the network and the number of calculations. The small magnitude of the change will dramatically delay gradient updates in the after processes of backpropagation and gradient descent to a large extent. In the LeNet model, when parameters finished a forward training process, we can set a different label for each layer in the whole network structure and get each layer's weights and bias. Through comparing these weights, we can find the biggest changes in weights and define them as the more important parameters. Furthermore, in order to delete those weights that are unnecessary for the network, we can set a threshold. It could be the smallest parameter in the layer or the entire model. Every weight bigger than the threshold can be defined as an important parameter and continue through the process of gradient updates. Besides, every time after we finished a process of training epoch, some parameters will be kept and the backpropagation and loss function cannot be calculated. Thus, we can prune such parameters after every training, when the network achieves iterated over a certain number of times. When the network finishes its calculation, some weights will be close to zero. We can delete these weights that are close to zero, it does not merely reduce the network's structure volume, however, it decreases the times of calculation [12]. This finishes the pruning of the LeNet model.

7. Conclusion

This paper discusses several pruning technologies based on VGG-16, ResNet-50, Yolov3-416, and LeNet-300-100 neural networks. This paper finds that pruning technologies can reduce the number of parameters to a large extent. Firstly, this paper analyzes that Neuron pruning based on the VGG-16 model will use APoZ to measure the filters that exist in the neural network, and then decided to prune or keep them. Secondly, channel pruning technology also decreases the size of parameters in the ThiNet-GAP, which is smaller than the original parameters. However, through Neuron pruning and Channel pruning, the accuracy of Top1/Top5 can still maintain at 67.34% and 87.92%. Furthermore, extreme pruning is also an indispensable part of the YOLOV3 model. To reduce the entire volume of the YOLOV3 structure and make it easily deployed on mobile devices, layer pruning technology can prune almost 80% volume. YOLOV3 model still has smaller parameters, and less time cost and its accuracy rate didn't change. At last, weight pruning on the LeNet-300-100 and LeNet-5 dramatically decrease the volume of parameters and FLOP. After pruning the weights from different layers, the whole LeNet model's accuracy still can reach to 82%.

References

[1] Y.,He, X.,Zhang, &J. Sun, (2017). Channel pruning for accelerating very deep neural networks. In Proceedings of the IEEE international conference on computer vision (pp. 1389-1397).

[2] J. Luo, H., J., Wu, &W. Lin, (2017). Thanet: A filter level pruning method for deep neural network compression. In Proceedings of the IEEE international conference on computer vision (pp. 5058-5066).

[3] Z., Liu, J., Li, Z., Shen, G., Huang, S., Yan, &C. Zhang, (2017). Learning efficient convolutional networks through network slimming. In Proceedings of the IEEE international conference on computer vision (pp. 2736-2744).

[4] H Hu, R Peng, YW Tai, CK Tang.: Network Trimming: A Data-Driven Neuron Pruning Approach towards Efficient Deep Architectures. arXiv preprint arXiv:1607.03250, (2016)

[5] Miguel Á. Carreira-Perpiñán, YerlanIdelbayev; Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018, pp. 8532-8541

[6] S., Han, J., Pool, J.,Tran,W. Dally,: Learning both weights and connections for the efficient neural network. In: Advances in Neural Information Processing Systems (NIPS). pp. 1135–1143 (2015)

[7] T.J., Yang, Y.H., Chen, V.Sze,: Designing energy-efficient convolutional neural networks using energy-aware pruning. arXiv preprint arXiv:1611.05128 (2016)

[8] W., Wen, C., Wu, Y., Wang, Y., Chen, H.Li,: Learning structured sparsity in deep neural networks. In: Advances in Neural Information Processing Systems. pp. 2074–2082 (2016)

[9] X., Gong, L.,Ma, &H. Ouyang, (2020, February). An improved method of Tiny YOLOV3. In IOP Conference Series: Earth and Environmental Science (Vol. 440, No. 5, p. 052025). IOP Publishing.

[10] J., Redmon, &A. Farhadi, (2018). Yolov3: An incremental improvement. arXiv preprint arXiv:1804.02767.

[11] Q. -C. Mao, H. -M. Sun, Y. -B. Liu and R. -S. Jia, "Mini-YOLOv3: Real-Time Object Detector for Embedded Applications," in IEEE Access, vol. 7, pp. 133529-133538, 2019, doi: 10.1109/ACCESS.2019.2941547.

[12] X. Ma, G. Yuan, S. Lin, Z. Li, H. Sun and Y. Wang, "ResNet Can Be Pruned 60×: Introducing Network Purification and Unused Path Removal (P-RM) after Weight Pruning," 2019 IEEE/ACM International Symposium on Nanoscale Architectures (NANOARCH), 2019, pp. 1-2, doi: 10.1109/NANOARCH47378.2019.181304.