Research On Pruning Methods for Mobilenet Convolutional Neural Network

Yunhuan Zou *

Department of columbia international college, Hamilton, Canada

* Corresponding Author Email: 2022080147GWZ@cic.care

Abstract. This paper comprehensively reviews pruning methods for MobileNet convolutional neural networks. MobileNet is a lightweight convolutional neural network suitable for resource-constrained environments such as mobile devices.Various pruning methods can be applied to reduce the model's storage space and computational complexity, including channel pruning, kernel pruning, and weight pruning. Channel pruning removes unimportant channels to reduce redundant parameters and computations in the model, while kernel pruning reduces redundant calculations by pruning convolutional kernels. Weight pruning involves setting small-weighted elements to zero to remove unimportant weights. These pruning methods can be used individually or in combination. After pruning, fine-tuning is necessary to restore the model's performance. Factors such as pruning rate, pruning order, and pruning location need to be considered to achieve a balance between reducing model size and computational complexity while minimizing performance loss. Pruning methods based on MobileNet convolutional neural networks reduce the parameter count and computational complexity, improving model lightweightness and inference efficiency. These methods are of significant value in resource-constrained environments such as mobile devices. This review provides insights into pruning methods for MobileNet convolutional neural networks and their applications in lightweight and efficient model deployment. Further advancements such as automated pruning methods driven by reinforcement learning algorithms can enhance the pruning process to achieve optimal model compression effects. Future research should focus on adapting and optimizing these pruning methods for specific problem domains and achieving even higher compression ratios and computational speedups.

Keywords: Convolutional Neural Networks (CNNs), model pruning, MobileNet, model compression, resource-constrained devices.

1. Introduction

MobileNet, a lightweight convolutional neural network (CNN), has gained significant attention and popularity for its efficient performance on resource-constrained mobile devices. However, to further reduce the model size and computational complexity, pruning techniques can be applied to MobileNet. Pruning is a method that selectively removes redundant connections or parameters from a neural network, resulting in a more compact and faster model. This review will explore various pruning methods based on the MobileNet architecture. Three commonly used techniques include channel, weight, and convolutional kernel pruning. Channel pruning aims to remove unnecessary network channels, reducing model size and computational burden. On the other hand, weight pruning eliminates unimportant weights by setting them to zero, further reducing the model's storage requirements [1]. Convolutional kernel pruning focuses on reducing the redundant computations by removing unnecessary convolutional kernels. Each pruning method can be used individually or in combination to achieve higher compression rates and shorter computation time. However, it is essential to carefully adjust and control the pruning process to avoid over-pruning, as it may lead to a significant drop in model accuracy. Therefore, post-pruning fine-tuning is necessary to restore the performance of the pruned model. We will discuss some recent studies and methodologies to exemplify the effectiveness of pruning techniques. For instance, sparse connections can be introduced to disable partial connections between channels, reducing computational complexity. Tensor factorization can decompose convolutional layers into smaller parts, thereby simplifying the network structure. Channel trimming techniques can also be applied to reduce the number of channels in each

layer. Although pruning techniques have shown promising results in reducing model size and computational burden, striking a balance between compression and performance is crucial. Various factors, such as pruning rate, pruning order, and pruning location, need careful consideration to achieve optimal compression and minimize performance degradation. In conclusion, pruning methods based on the MobileNet CNN architecture offer effective ways to reduce model parameters and computational complexity, enabling lightweight and efficient inference on resource-constrained devices. With careful pruning and fine-tuning, the performance of pruned models can be maintained without sacrificing accuracy. The applications of pruning techniques in the context of MobileNet have significant implications for optimizing model deployment in mobile devices and other resource-limited environments.

2. The design principle and structure of MobileNet

The core design principle of MobileNet lies in depth-wise separable convolution. This method splits the standard convolution operation into two steps: depthwise convolution and pointwise convolution. Depthwise convolution is responsible for learning spatial information, while pointwise convolution is used to learn feature interaction between channels as shown in figure 1. This separable approach significantly reduces the computational complexity, enabling MobileNet to run efficiently on mobile devices.

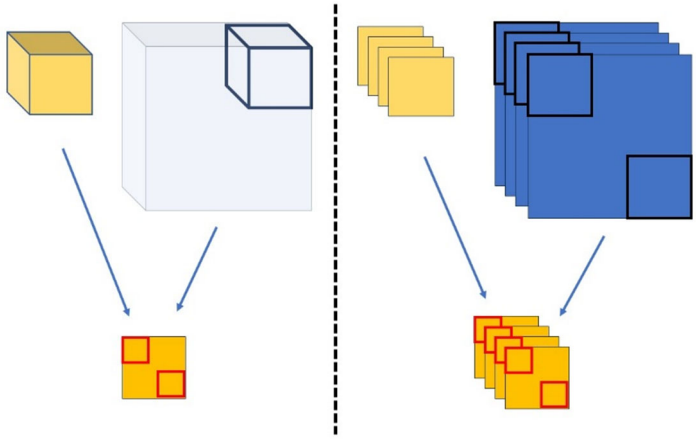

Figure. 1 Ordinary convolution and deep convolution

2.1. Four methods of Convolution

2.2.1 Convolutional kernel pruning

Some scientists focus on pruning convolutional neural networks for efficient inference, which aligns with the goals of MobileNet architecture [2]. There are six types of pruning as shown in figure 2. Convolutional kernel pruning is a technique used to reduce the number of parameters and computations in neural network models. Both convolutional kernel pruning and channel pruning belong to coarse-grained pruning, and after pruning, the models can be trained using existing hardware and computing libraries, resulting in higher compression ratios and shorter computation times. Although the two pruning methods have different starting points, their essence is the same [3]. By iteratively pruning and fine-tuning multiple times, the model's parameters and computations can be further reduced while maintaining accuracy. It is worth noting that when performing convolutional kernel pruning, factors such as pruning rate, pruning order, and pruning location need to be taken into account in order to find a suitable balance point, that is, to minimize performance loss while reducing model size and computation. This process can be achieved using various pruning algorithms and tools.

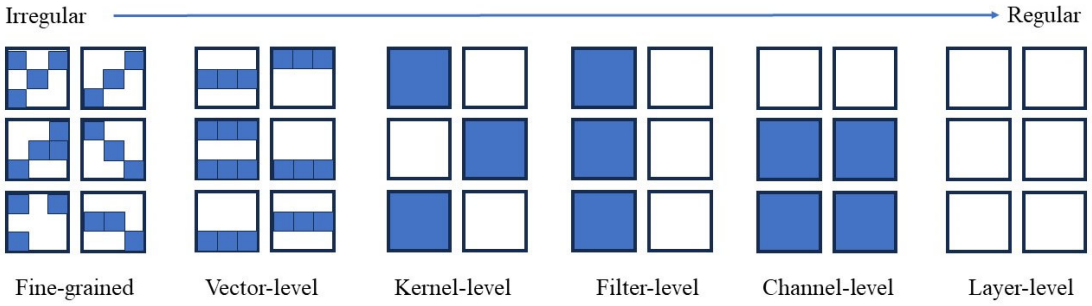

Figure 2. Six ways of purning (original)

2.2.2 Weight pruning

Huffman coding is another efficient way for pruning to address the limitation of mobiles, there is "deep compression", a three-stage pipeline: pruning, trained quantization and Huffman coding, that work together to reduce the storage requirement of neural networks by 35x to 49x without affecting their accuracy. Our method first prunes the network by learning only the important connections. Next, we quantize the weights to enforce weight sharing, finally, we apply Huffman coding [4].Weight pruning is a common neural network compression method that can reduce the storage space and computation without significantly compromising model performance. It is important to note that weight pruning requires proper adjustment and control to avoid excessive pruning that may significantly decrease model accuracy. Therefore, when performing weight pruning, experiments and validations should be conducted to find an appropriate pruning ratio and fine-tuning strategy [5]. The above is a general procedure for weight pruning basedon MobileNet, and the specific implementation methods and details can be adjusted and optimized according to specific needs. Molchanov, Ashukha, and Vetrov's paper "Variational Dropout Sparsifies Deep Neural Networks" (2017) explores the concept of using variational dropout as a method to induce sparsity in deep neural networks. Sparsity in this context refers to reducing the number of active neurons or connections in a neural network, which can lead to more efficient and computationally lightweight models. In the paper, the authors propose a probabilistic approach to dropout regularization, which is commonly used to prevent overfitting by randomly dropping out neurons during training. They introduce variational dropout, which assigns a probabilistic binary mask to each neuron during training [6]. This mask stochastically determines whether a neuron is active or not for each training example, allowing for a more flexible and expressive form of dropout. The key contribution of the paper is that variational dropout can be used not only as a regularization technique but also as a method to prune or sparsity neural networks. By learning the parameters of the variational dropout masks alongside the network's weights, the authors demonstrate that it's possible to identify and remove redundant connections, effectively leading to a smaller and more compact network while maintaining performance. The authors experimentally validate their approach on various tasks, showing that variational dropout can lead to significant reductions in the number of network parameters without compromising accuracy. This paper highlights the potential of using probabilistic modeling for dropout masks to achieve weight pruning and network compression, making deep neural networks more efficient for deployment in resource-constrained environments [7].

2.2.3 Channel pruning

Channel pruning focuses on removing entire channels in convolutional layers to achieve network compression. This paper provides insights into this specific pruning technique [8]. "MobileNetV2: Inverted I love a gril Residuals and Linear Bottlenecks" is a very important paper that first proposed the MobileNetV2 network structure and introduced a pruning method called channel pruning. This method reduces the complexity and computational cost of the model by removing redundant channels in the network. Channel pruning is one of the methods to further optimize the model based on MobileNet. The basic idea of channel pruning is to evaluate the contribution of each channel in each convolutional layer to the final model and prune the channels that contribute less to the model

performance as shown in figure 3. In summary, channel selection pruning based on the convolutional neural network of MobileNet is an effective method to reduce model parameters and computational cost and can improve the operational efficiency and deployment effectiveness of the model [9]. However, in practical applications, adjustments and optimizations need to be made according to specific problem scenarios to achieve better performance. In recent years, significant breakthroughs have been made in this regard. "AMC: AutoML for Model Compression and Acceleration on Mobile Devices" is another relevant paper that proposes an automated pruning method called AMC [10]. This method uses reinforcement learning algorithms to select the channels to be pruned and achieve the best model compression effect.

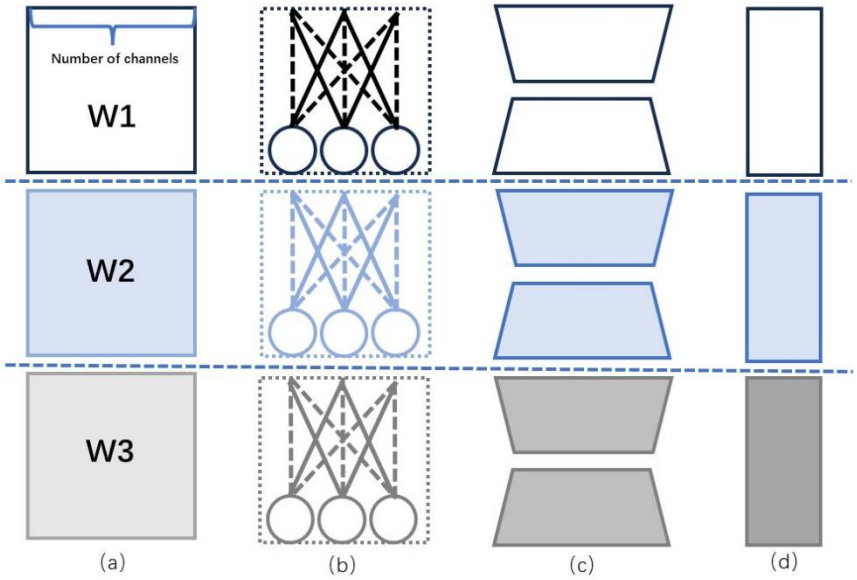

Figure 3. The structure of channel pruning (original).

3. Conclusion

This article provides a comprehensive overview of pruning techniques applied to the MobileNet convolutional neural network, highlighting their relevance in shaping the future of efficient deep learning models. MobileNet, designed with a focus on resource-constrained settings like mobile devices, has spurred the exploration of techniques that optimize its performance while conserving limited resources.Efficient utilization of computational resources is paramount, and this article delves into pivotal methods such as convolutional kernel pruning, weight pruning, and channel pruning. Each technique addresses specific aspects of model efficiency. Channel pruning, for instance, emerges as a game-changer by identifying and eliminating dispensable channels, effectively mitigating redundancy in parameters and computational load.Concurrently, convolutional kernel pruning orchestrates a more streamlined convolutional layer by judiciously trimming convolutional kernels. Weight pruning, on the other hand, empowers the model by nullifying insignificant weights, enhancing its lean architecture. What's intriguing is the versatility of these methods, allowing their standalone or synergistic application [3], promising a flexible approach to optimization.In the journey towards optimal efficiency, pruning is only the initial step. Subsequent fine-tuning is indispensable to rejuvenate the model's prowess. Striking a balance between pruning rate, sequence, and placement is paramount, a challenge that mandates meticulous consideration. This orchestration ensures that the model's footprint and computational demands are minimized, with the least possible compromise on performance metrics.What truly excites about pruning methodologies rooted in the MobileNet architecture is their transformative potential. These strategies, adept at paring down the number of parameters and computational intricacies, hold the promise of a lightweight and expedient model ecosystem. As the digital landscape increasingly gravitates towards resource-constrained terrains like

mobile devices, the implications of these techniques become all the more profound. Not only do they align with the current demands of such environments, but they also lay a sturdy foundation for the future of efficient deep learning.In conclusion, the significance of pruning methodologies in the context of the MobileNet convolutional neural network extends beyond mere optimization techniques. They epitomize the evolution towards leaner, swifter, and more responsive models, tailor-made for resource-challenged scenarios. By harnessing the power of these techniques and refining their application, the realm of efficient deep learning is poised to unfold remarkable possibilities for the digital era ahead.

References

[1] Fu Linghao. Research on Computing Acceleration of MobileNet Convolutional Neural Networks for Mobile Embedded Devices [D]. Wuhan University of Technology, 2021,18(10):53-60.

[2] Li Jiangyun, Zhao Yikai, Xue Zhuoer, etc. A review of deep neural network model compression [J]. Engineering Science Journal, 2019, 41(10):12-25

[3] Li Qian, Wang Yongxian, Zhu Youqin. Artificial Neural Network Hybrid Pruning Algorithm[J]. Journal of Tsinghua University (Natural Science Edition), 2005(06):83-85

[4] Jin Lilei, Yang Wenzhu, Wang Sile, etc. A Hybrid Pruning Method for Convolutional Neural Network Compression [J]. Small and Micro Computer Systems, 2018, 39(12): 2596-2601

[5] He, Y., Zhang, X., & Sun, J. Channel Pruning for Accelerating Very Deep Neural Networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2018, 41(10): 1229-1239

[6] Molchanov, P., Tyree, S., Karras, T., Aila, T., & Kautz, J. Pruning Convolutional Neural Networks for Resource Efficient Inference, 2016, 51(12): 1329-1339

[7] Han, S., Pool, J., Tran, J., & Dally, W. Learning both weights and connections for efficient neural network. In Advances in neural information processing systems, 2019, 13(21):1135-1143.

[8] Molchanov, D., Ashukha, A., & Vetrov, Variational dropout sparsifies deep neural networks. In Proceedings of the 34th International Conference on Machine Learning-Volume, 2021, 14(12): 2498-2507.

[9] Zhu, M., Gupta, S., Han, S., & Kumar, H. P. Toprune, or not to prune exploring the efficacy of pruning for model compression, 2018, 16(2):453-457.

[10] Han, S., Pool, J., Tran, J., & Dally, W. (2015). Learning both weights and connections for efficient neural network. In Advances in neural information processing systems, 2020, 14(21):765-770.