Dynamic Channel Pruning with Adaptive Weight Learning

Shuanglin Wu[1], Chao Xiao[1], Jungang Yang[1], and Wei An[1]

[1]National University of Defense Technology

October 30, 2022

Abstract

Dynamic channel pruning is widely used in model compression to improve the efficiency of neural networks. Although dynamic pruning can remove redundant channels dynamically, the parameters still remain unchanged, which can limit the performance as the response of each neuron changes with different inputs. In this paper, we propose a novel dynamic channel pruning method with adaptive weight learning, which can adaptively adjust both the parameters and widths of the filters simultaneously. Specifically, we design an adaptive-weight convolution module, which can be customized for different inputs under the guidance of global context information to capture representative local patterns and synthesize interested features. At the same time, we utilize a channel importance prediction module to predict the saliency of each channel. Based on the channel saliency, unimportant channels can be removed dynamically to speed up the convolution. These two modules work jointly to achieve a good trade-off between model performance and computational complexity. Experiments on image classification and object detection tasks demonstrate that our method can greatly reduce the computational burden while maintaining the performance, which outperforms state-of-the-art methods.

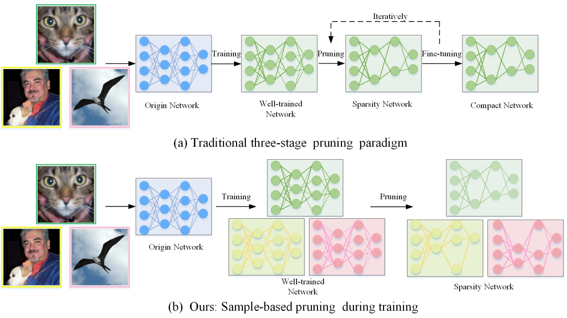
Dynamic Channel Pruning with Adaptive Weight Learning

Shuanglin Wu, Chao Xiao, Jungang Yang, and Wei An
National University of Defense Technology, College of Electronic Science and Technology, Hunan, China
Email: yangjungang@nudt.edu.cn

Dynamic channel pruning is widely used in model compression to improve the efficiency of neural networks. Although dynamic pruning can remove redundant channels dynamically, the parameters still remain unchanged, which can limit the performance as the response of each neuron changes with different inputs. In this paper, we propose a novel dynamic channel pruning method with adaptive weight learning, which can adaptively adjust both the parameters and widths of the filters simultaneously. Specifically, we design an adaptive-weight convolution module, which can be customized for different inputs under the guidance of global context information to capture representative local patterns and synthesize interested features. At the same time, we utilize a channel importance prediction module to predict the saliency of each channel. Based on the channel saliency, unimportant channels can be removed dynamically to speed up the convolution. These two modules work jointly to achieve a good trade-off between model performance and computational complexity. Experiments on image classification and object detection tasks demonstrate that our method can greatly reduce the computational burden while maintaining the performance, which outperforms state-of-the-art methods.

Introduction: Deep convolutional neural networks (CNNs) have achieved excellent performance in various computer vision tasks, *e.g.,* image classification [5], object detection [18] and object tracking [1]. However, the great performance usually comes from larger and deeper networks with the requirement of more memory and computation resources, which brings significant challenges for deployment on limited hardware devices and low-power edge computing applications.

In recent years, model compression techniques have been widely explored to reduce the computational budget of CNNs, including weight quantization [2], low-rank decomposition [13], network pruning [16] and knowledge distillation [9]. Among them, channel pruning have been widely studied for its effectiveness to improve model efficiency and compatibility with other compression methods, which can be divided into static [7, 11]and dynamic pruning methods [4, 6]. Static pruning methods usually follows a three-stage pruning paradigm(as shown in Fig. 1(a)): training network from scratch firstly, then removing the "unimportant" filters by measure the importance based on a specific criterion, and finally fine-tuning the model to recover the performance. To avoid a sharp decline of performance, pruning and fine-tuning are usually performed iteratively, which makes static pruning a cumbersome process. Moreover, after static pruning, model parameters and structures are changed and shared for different inputs, which brings several drawbacks. Firstly, since parameters and structures treat all inputs equally, the inputs with different features can not be discriminated accurately. Secondly, due to the permanent removal of some channels, model capacity will be lost. In contrast, dynamic channel pruning methods (as shown in Fig. 1(b)) learn the importance of each channel and skip the "unimportant" ones adaptively, which can remedy the aforementioned deficiencies of static pruning. Although existing dynamic pruning methods can adjust the channels adaptively for different inputs, the model parameters are still shared for different inputs, which tends to limit the performance.



(a) Traditional three-stage pruning paradigm

(b) Ours: Sample-based pruning during training

Fig 1 *Different pruning paradigms.*

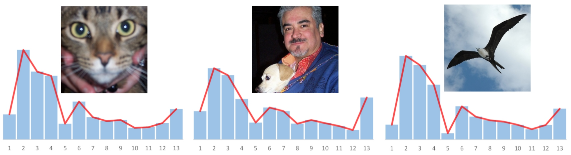


Fig 2 *Response in the last channel of each activation layer of the pre-trained VGG-16 for different input images.*

Figure 2 shows the response values in the last channel of each activation layer of the pre-trained VGG-16 for different inputs. It can be observed that the responses of each neuron changes with different inputs, *i.e.,* different structures are activated for different inputs. Based on this motivation, we design a novel dynamic channel pruning method which can dynamically remove redundant channels for higher efficiency and adaptively adjust parameters for better performance. Our proposed method consists of an adaptive-weight convolution module for adaptive parameter learning and a channel importance prediction module for the removal of unimportant channels. Combining these two modules organically, our pruning method can significantly reduce the computational complexity as well as maintain the performance. The contribution of our method is as follows:

- We propose a novel dynamic channel pruning method, which can adaptively adjust the model parameters and structures for different inputs to reduce the computational burden and alleviate performance degradation.
- We design an adaptive-weight convolution module and a channel importance prediction module. These two modules work jointly to achieve dynamic channel pruning with a good trade-off between performance and computational complexity.
- Extensive experiments on two tasks (*i.e.,* image classification and object detection) demonstrate both the effectiveness and efficiency of our proposed dynamic pruning method.

Dynamic Channel Pruning with Adaptive Weight Learning: We propose a novel dynamic channel pruning method with adaptive weight learning. The overall architecture is shown in Fig. 3. In this section, we will introduce our method in detail.

Notation and Preliminaries: For a L-layer convolutional neural network, we denote the i-th convolution layer as $f_i : x_{i-1} \in \mathbb{R}^{C_{i-1}\times H_{i-1}\times W_{i-1}} \rightarrow x_i \in \mathbb{R}^{C_i\times H_i\times W_i}$, where x_{i-1}, x_i denote the input feature map and the output feature map, respectively. C, H and W represents channel number, height and width of the feature map. $\theta_0^i \in \mathbb{R}^{C_i\times C_{i-1}\times k\times k}$ represents the convolution kernel, and k is the size of the convolution kernel. Then, we define i-th convolutional layer as follows:

$$f_i(x_{i-1}) = conv_i(x_{i-1}, \theta_0^i) \tag{1}$$

Further, convolutional layer with batch normalization is defined as follows:

$$x_i = \gamma_i \cdot norm(\mathrm{conv}_i(x_{i-1}, \theta_0^i)) + \beta_i \tag{2}$$

where $norm$ is the standardized normalization. The trainable parameter γ_i and β_i represent the scaling factor and bias, respectively.

Adaptive-weight Convolution Module: To learn more prominent features for different inputs, we design an adaptive-weight convolution module, which redistributes convolution weight based on different features. The adaptive weight module consists of three parts, including context fusion submodule, channel interaction submodule and coefficient generation submodule, which are shown successively in the three dashed box of adaptive weight convolution module in Fig. 3.

To extract the context information of the input feature map, we first employ a average pooling layer to reduce its spatial resolution to $C_{i-1} \times k \times k$. Then we feed it to context fusion submodule to

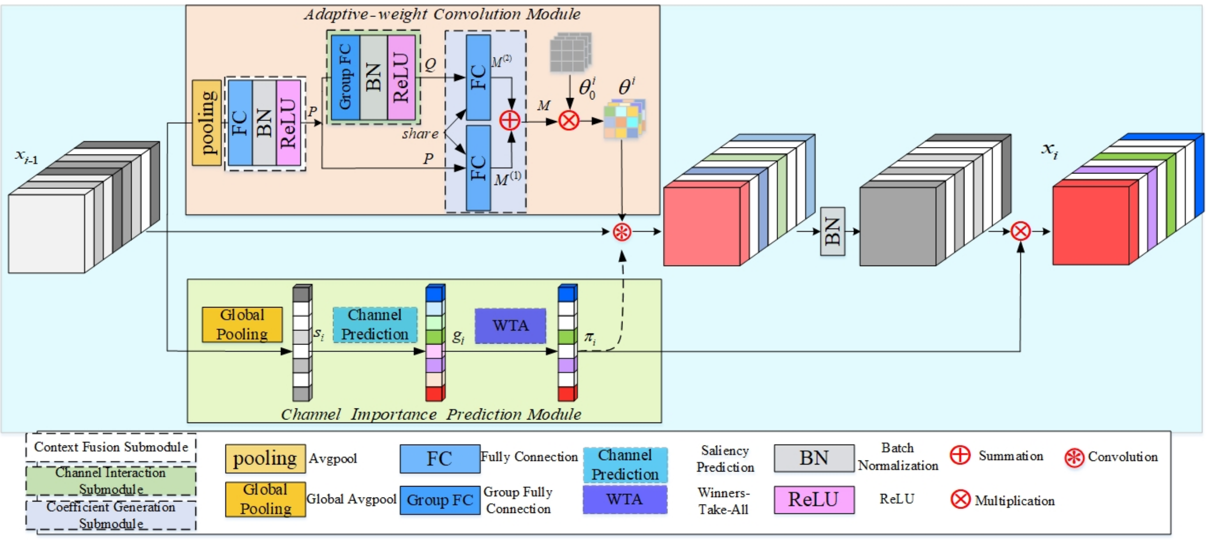


Fig 3 *The framework of dynamic channel pruning with adaptive weight learning. The colored blocks indicate the importance of the corresponding channels, and the white ones indicate all-zero channels.*

extract context information by projecting the adjusted feature map to $P \in \mathbb{R}^{C_{i-1} \times d}$, and the default size of d is set to $k^2/2$. After that, the channel interaction submodule utilizes a group linear layer to project the input channel C_{i-1} into the output dimension of C_i. The input feature P is transformed into feature $Q \in \mathbb{R}^{C_i \times d}$. Next, the coefficient generation submodule takes P and Q as inputs, and decodes them from one-dimension vector d to $k \times k$. Then we use these two groups features to obtain feature $M \in \mathbb{R}^{C_i \times C_{i-1} \times k \times k}$. Finally, the weight adjusted convolution θ^i can be obtained by mutiplying θ_0^i with M element-wise, which is equivalent to assigning weight to the convolution kernel in a pixel-wise manner. We replace θ_0^i with θ^i in the convolution layer to obtain the adaptive convolution. In this way, we can adjust parameters for different inputs utilizing the adaptive weight convolution module.

Channel Importance Prediction Module : We design a channel importance prediction module to evaluate the importance of each channel. We first compress the input feature map by Global Average Pooling (GAP), which projects each channel of input feature map into a scalar, *i.e.,* the input feature map is projected to a column vector $s_i \in \mathbb{R}^{C_{i-1}}$.

Then, we use a fully connected layer to project s_i into $g_i \in \mathbb{R}^{C_i}$, which represents the importance of each channel. Without elaborately designing parameter evaluation criteria, our channel importance module can learn important channels adaptively based on the input features, and can be optimized with the network to improve the accuracy of prediction.

Dynamic Channel Pruning : We use wta (winner-take-all) function $wta_{\lceil k \rceil}$ [17] to sort the channel importance of the feature map. $wta_{\lceil k \rceil}$ preserves only top k values, and sets all the rest to zero, which can be described as

$$\pi_i(x_{i-1}) = wta_{\lceil \alpha C_i \rceil}(g_i) \qquad (3)$$

We set pruning rate to α. $wta_{\lceil k \rceil}$ function sorts the importance of g_i to $\pi_i \in \mathbb{R}^{C_i}$, in which $C_i - \lceil \alpha C_i \rceil$ values are set to zero. π_i guides the network to skip the unimportant convolution channels and makes network sparse. Finally, we replace the γ_i of BN layer with π_i, which can filter out the unimportant channels to reduce the computational cost. Combining the adaptive-weight convolution module and channel importance prediction module, the proposed dynamic pruning method with adaptive weight can be formulated as follows. :

$$f_i(x_{i-1}) = \pi_i \cdot norm(\text{conv}_i(x_{i-1}, \theta^i)) + \beta_i \qquad (4)$$

Using the proposed dynamic pruning method, we can dynamically adjust the parameters and channels of the convolution, which can not only reduce the computational burden but also maintain the performance.

Experiments: We conduct experiments on two representative tasks to verify the proposed method, *i.e.,* image classification and object detection. For image classification, we conduct experiment on ResNet-20 and ResNet-56 for CIFAR-10 dataset [10]. For object detection, we evaluate our method on Centernet [21] with Resnet-50 as backbone for PASCAL VOC dataset [3].

Image Classification: CIFAR-10 contains 50000 training images and 10000 test images of 10 categories, in which the size of a single image is 32×32. We use ResNet-20, ResNet-56 as baseline networks. We utilize the SGD optimizer with momentum at 0.9 and weigtht dacay at 5×10^{-4}. The initial learning rate is set to 0.1, and dropped by 10× at epoch 120, 180, and 220. The total training epochs are set to 300. The batch-size is set to 128.

Comparison to the State-of-the-arts: We compare the proposed method with both static (*e.g.,* FPGM [7], DHP [11], HRank [12], AdaPruner [15]) and dynamic channel pruning methods (*e.g.,* SFP [6], FBS [4]) on CIFAR-10. The quantitative results is shown in Table. 1.

It can be observed that, for ResNet-20, after pruning 60.3% FLOPs, the sparse network achieves 91.17% Top-1 accuracy, which is slightly lower than DHP (91.17% v.s. 91.54%). But our method reduces more FLOPs (51.8% v.s. 60.3%), which demonstrates the effectiveness of our method. For ResNet-56, our method suffers only a slight performance degradation (93.59% v.s. 93.7%) while pruning 60.8% FLOPs, which demonstrates that our method can achieve a good trade-off between model efficiency and performance. We attribute the reason to the adaptive adjustment of both parameters and structures of convolution layers. More importantly, our method requires no fine-tuning to recover performance and can prune redundant channels by directly training the network.

Table 1. Comparison of the pruned ResNet with different methods on CIFAR-10.

Model	Method	Dynamic	Top-1 Acc.(%)	FLOPs↓(%)
ResNet-20	Baseline	-	91.53	-
	FPGM [7]	×	90.44	54.0
	DHP [11]	×	**91.54**	51.8
	SFP [6]	✓	90.83	42.2
	FBS [4]	✓	90.97	53.1
	Ours	✓	91.17	**60.3**
ResNet-56	Baseline	-	93.7	-
	FPGM [7]	×	93.49	52.6
	HRank [12]	×	93.17	50.0
	AdaPruner [15]	×	93.49	50.0
	SFP [6]	✓	92.26	52.6
	FBS [4]	✓	93.52	53.6
	Ours	✓	**93.59**	**60.8**

In addition, we visualizes channel numbers of pruned ResNet-20. The comparion of each layer between our method and baseline is shown in Fig. 4. It can be observed that our method can remove lots of redundant channels at each layer to reduce the computational complexity.

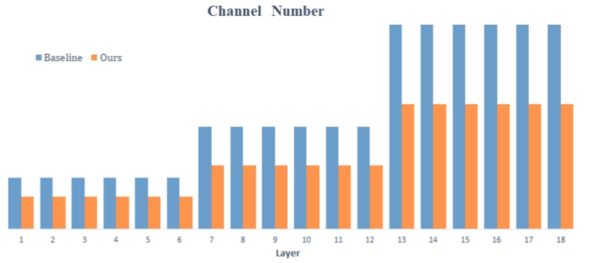


Fig 4 *Comparion of layer-wise channel numbers between baseline and our method.*

Ablation Study: We conduct ablation study to evaluate the effectiveness of the adaptive-weight convolution module and channel importance prediction module on CIFAR-10. The quantitive results are shown in Table 2. It can be observed that, with adaptive weight convolution module only, the FLOPs only increses 0.04%, while the Top-1 accuracy can be improved by 1.67%, which demonstrates the effectiveness of our proposed module in improving the performance. With channel importance prediction module only, the Top-1 accuracy declines 0.38% with a reduction of 60.35% FLOPs. Combining these two modules, we can achieve 91.17% Top-1 accuracy with a reduction of 60.32% FLOPs. The results demonstrate the effectiveness our proposed method.

Table 2. Ablation study on CIFAR-10 for ResNet-20, and pruning ratio $\alpha = 0.6$.

Adaptive Conv	Dynamic Channel	Top-1 Acc(%)	FLOPs(%)
×	×	91.53	-
✓	×	93.26	0.04↑
×	✓	91.01	60.35 ↓
✓	✓	91.17	60.32 ↓

Object Detection: We apply the proposed method to object detection to analyze the generalization of our method. We use CenterNet with ResNet-50 as the baseline and evaluate the performance on PASCAL VOC dataset. The PASCAL VOC dataset contains 16551 training images and 4962 testing images of 20 categories. Mean average precision (mAP) at IoU threshold 0.5 is utilized as the evaluation metric. Following the training settings of CenterNet, we employ SGD with the momentum 0.9 and the weight decay 5×10^{-4} as the optimizer. The initial learning rate is set to 1.25×10^{-4} and multiplied by a factor of 0.1 at epoch 45 and 60. The batch size is set to 32 and the training is stopped after 70 epochs.

Table 3. Results on the PASCAL VOC. CenterNet with ResNet-50 is utilized as baseline.

Method	mAP(%)	FLOPs↓(%)	$\Delta(mAP)$(%)
Baseline	76.46	-	-
Ours	75.29	63.70	-1.17

The quantitative results are shown in Table. 3. Compared to the baseline, after pruning 63.70% FLOPs, the mAP only decreases by 1.17%, which shows that the proposed method has a good generalization ability and can also achieve good performance in the downstream tasks of computer vision.

Conclusion: In this paper, we propose a novel dynamic channel pruning method which learns sample-dependent convolution weight and customizes convolution channels for different inputs, and provides more elaborate decisions for dynamic channel pruning. Specifically, based on different input features, the filter parameters and widths are customized for each sample to obtain a high-performance sparse network. Extensive experiments on different computer vision tasks shows that our method is competitive with state-of-the-art methods.

References

1. Ziang Cao, Ziyuan Huang, Liang Pan, Shiwei Zhang, Ziwei Liu, and Changhong Fu. Tctrack: Temporal contexts for aerial tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14798–14808, 2022.
2. Yoojin Choi, Mostafa El-Khamy, and Jungwon Lee. Towards the limit of network quantization. *arXiv preprint arXiv:1612.01543*, 2016.
3. Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010.
4. Xitong Gao, Yiren Zhao, Łukasz Dudziak, Robert Mullins, and Cheng-zhong Xu. Dynamic channel pruning: Feature boosting and suppression. *arXiv preprint arXiv:1810.05331*, 2018.
5. Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.
6. Yang He, Guoliang Kang, Xuanyi Dong, Yanwei Fu, and Yi Yang. Soft filter pruning for accelerating deep convolutional neural networks. *arXiv preprint arXiv:1808.06866*, 2018.
7. Yang He, Ping Liu, Ziwei Wang, Zhilan Hu, and Yi Yang. Filter pruning via geometric median for deep convolutional neural networks acceleration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4340–4349, 2019.
8. Yihui He, Xiangyu Zhang, and Jian Sun. Channel pruning for accelerating very deep neural networks. In *Proceedings of the IEEE international conference on computer vision*, pages 1389–1397, 2017.
9. Geoffrey Hinton, Oriol Vinyals, Jeff Dean, et al. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2(7), 2015.
10. Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.
11. Yawei Li, Shuhang Gu, Kai Zhang, Luc Van Gool, and Radu Timofte. Dhp: Differentiable meta pruning via hypernetworks. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part VIII 16*, pages 608–624. Springer, 2020.
12. Mingbao Lin, Rongrong Ji, Yan Wang, Yichen Zhang, Baochang Zhang, Yonghong Tian, and Ling Shao. Hrank: Filter pruning using high-rank feature map. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1529–1538, 2020.
13. Shaohui Lin, Rongrong Ji, Chao Chen, Dacheng Tao, and Jiebo Luo. Holistic cnn compression via low-rank decomposition with knowledge transfer. *IEEE transactions on pattern analysis and machine intelligence*, 41(12):2889–2905, 2018.
14. Xudong Lin, Lin Ma, Wei Liu, and Shih-Fu Chang. Context-gated convolution. In *European Conference on Computer Vision*, pages 701–718. Springer, 2020.
15. Xiangcheng Liu, Jian Cao, Hongyi Yao, Wenyu Sun, and Yuan Zhang. Adapruner: Adaptive channel pruning and effective weights inheritance. *arXiv preprint arXiv:2109.06397*, 2021.
16. Jian-Hao Luo, Jianxin Wu, and Weiyao Lin. Thinet: A filter level pruning method for deep neural network compression. In *Proceedings of the IEEE international conference on computer vision*, pages 5058–5066, 2017.
17. E Majani, Ruth Erlanson, and Yaser Abu-Mostafa. On the k-winners-take-all network. *Advances in neural information processing systems*, 1, 1988.
18. Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.
19. Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.
20. Yehui Tang, Yunhe Wang, Yixing Xu, Yiping Deng, Chao Xu, Dacheng Tao, and Chang Xu. Manifold regularized dynamic network pruning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5018–5028, 2021.
21. Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. Objects as points. *arXiv preprint arXiv:1904.07850*, 2019.