Supplementary Material: Pruning from Scratch

Submission ID: 403

Effects of Pre-training on Pruning

In the main text, we explore the effects of pre-trained weights on pruned structures by visualizing the structure similarity matrices. Here we present more similar results of ResNet20 and ResNet56 models on CIFAR10 datasets.

Figure S1 and S2 show the results. All the pruned models are required to reduce 50% FLOPS of their original models on CIFAR10 dataset. In each figure, (a) we display the correlation coefficient matrix of the pruned models directly learned from randomly initialized weights (**"random"**) and other pruned models based on different checkpoints during pre-training (**"Epochs"**) (top-left). We display the correlation coefficient matrix of pruned structures from pre-trained weights in a finer scale (right). We show the channel numbers of each layer of different pruned structures (bottom-left). Red line denotes structure from random weights; (b) similar results from the experiment with a different random seed; (c) we display correlation coefficient matrices of all the pruned structure from five different random seeds. We mark the names of initialized weights used to get pruned structure below.

For ResNet20 and ResNet50, we observe the same phenomena with those in VGG16. First, that the pruned structures learned from random weights are not similar to all the network structures obtained from pre-trained weights. Second, the pruned model structures learned directly from random weights are more diverse with various correlation coefficients. Third, the pruned structure based on the checkpoints from near epochs are more similar with high correlation coefficients in the same experiment run.

The only difference between ResNet models with VGG16 is that the the similarities of the pruned structure based on the pre-trained weights of different random seeds are not as high as those of VGG16. This is mainly due to the fact that we only prune the layers on the residual branch in ResNet. In the case that the channel numbers of backbone layers are fixed, the number of channels of these pruned layers can have greater freedom of choice, so that they didn't converge to the same structure. However, the similarity between pruned structures based on pre-trained weights is still higher than that obtained from random weights. These results further validate our analysis in the main text.

Experiment Settings

Channel Gates Location

Following the same practice in Network Slimming (Liu et al. 2017), we associate the channel gates at the end of BatchNorm layer (Ioffe and Szegedy 2015) after each convolutional layer, since we can use the affine transformation parameters in BatchNorm to scale the channel output. For the residual block, we only associate gates in the middle layers of each block. For the depth-wise convolution block in MobileNetV1 (Howard et al. 2017), we associate gates at the end of the second BatchNorm layer. For the inverted residual block in MobileNetV2 (Sandler et al. 2018), we associate gates at the end of the first BatchNorm layer.

ImageNet Models Initialization

Table S1: ImageNet model initialization settings. C_{in} denotes the base channel number.

Model	FLOPS	C_{in}	Sparsity	Multiplier
MobileNet v1	150M	20	0.80	1.00
	286M	32	0.75	1.00
	567M	48	0.67	1.00
MobileNet v2	210M	32	0.75	0.75
	300M	40	0.80	0.90
	510M	50	0.80	1.30
ResNet50	1.0G	64	0.50	0.50
	2.0G	64	0.75	0.75
	3.0G	64	0.85	0.85
	4.1G	80	0.80	0.90

Table S1 summarizes ImageNet model initialization configurations.

Ablation Study

In the following sections, we explore the performance of our method under different channel expansion rate, pruning ratio

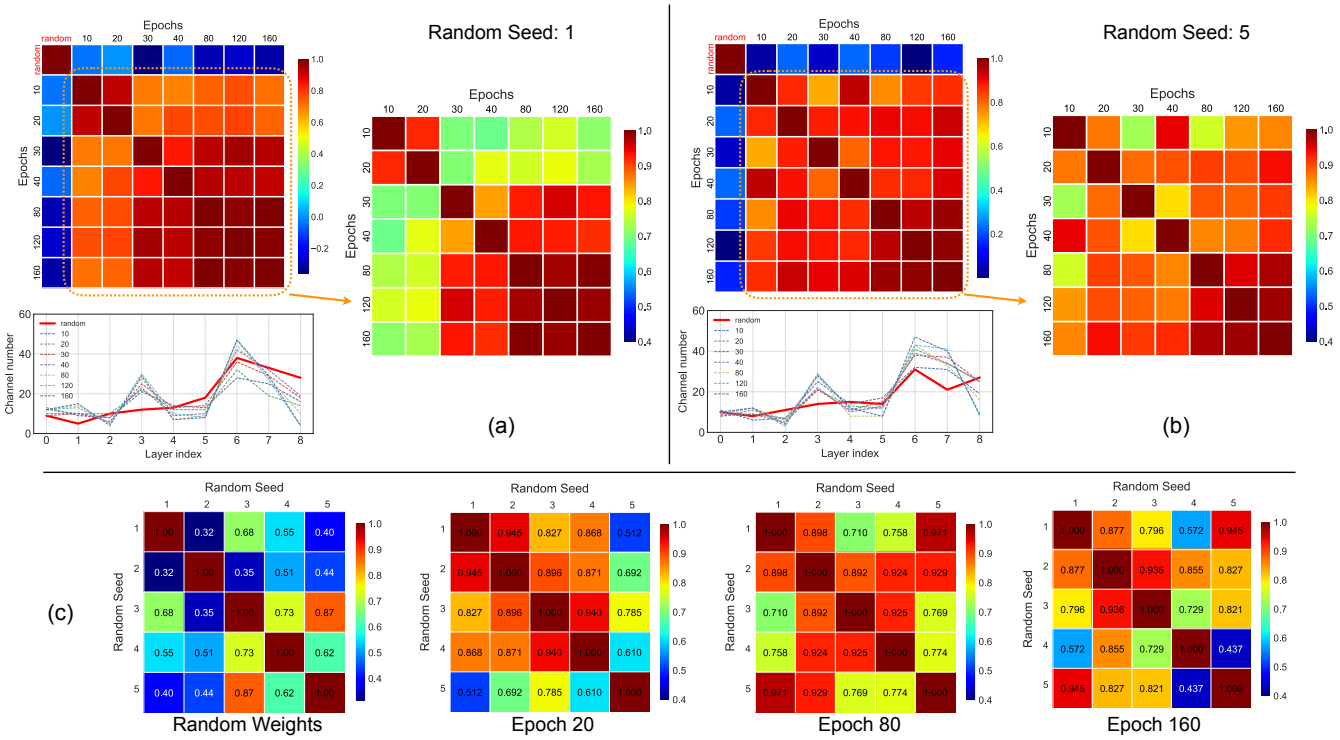

Figure S1: Exploring the effect of pre-trained weights on pruned structure by using **ResNet20** model.

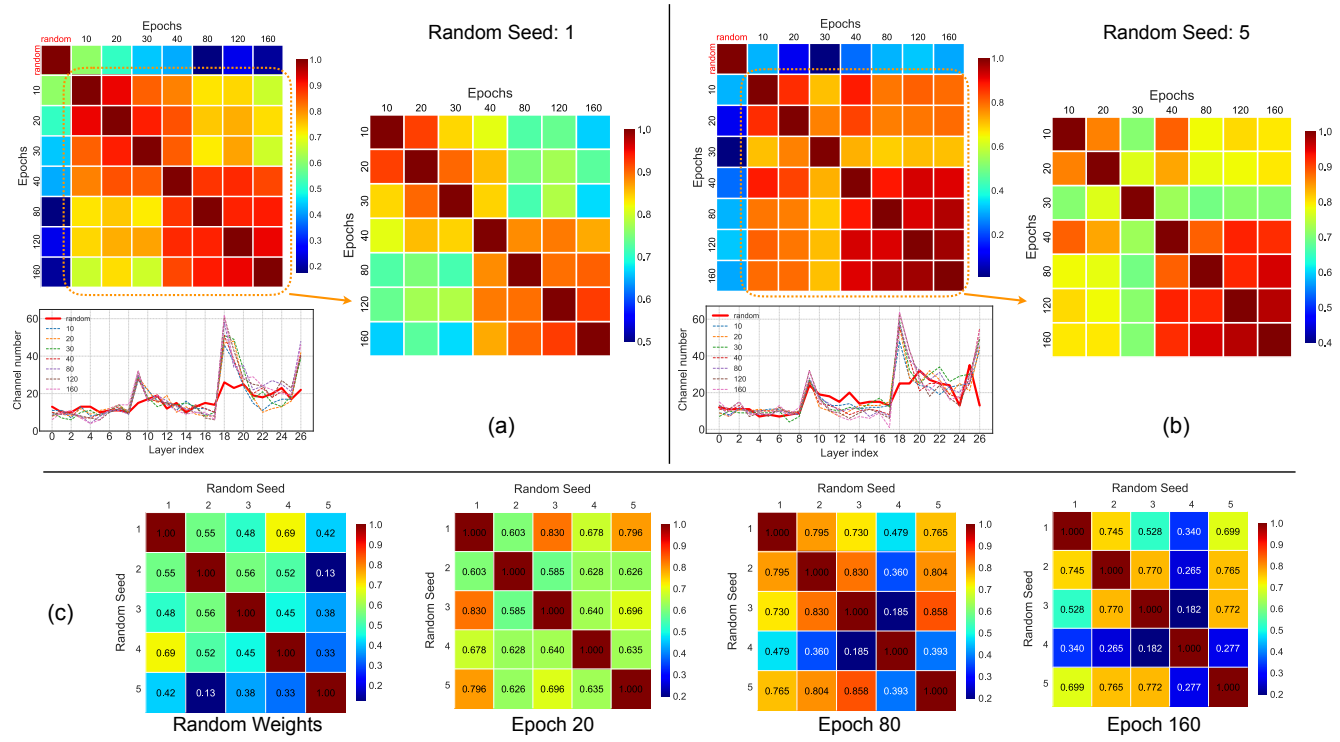

Figure S2: Exploring the effect of pre-trained weights on pruned structure by using **ResNet56** model.

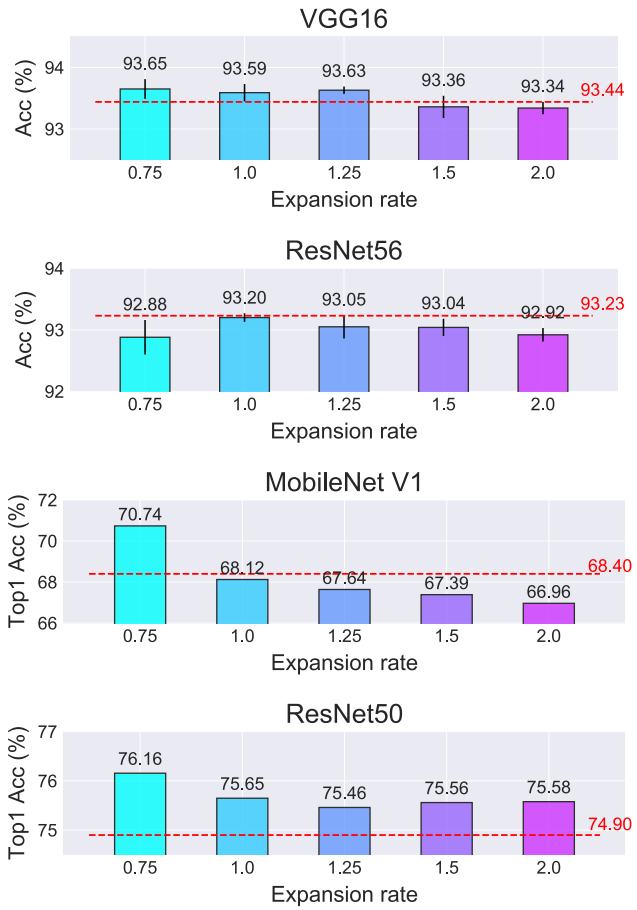

Figure S3: Effects of different expansion rate on the pruned model accuracy. Red dotted lines denote the baseline full models accuracy. VGG16 and ResNet56 models are trained on CIFAR10 dataset for five runs. MobileNet V1 and ResNet50 models are trained on ImageNet.

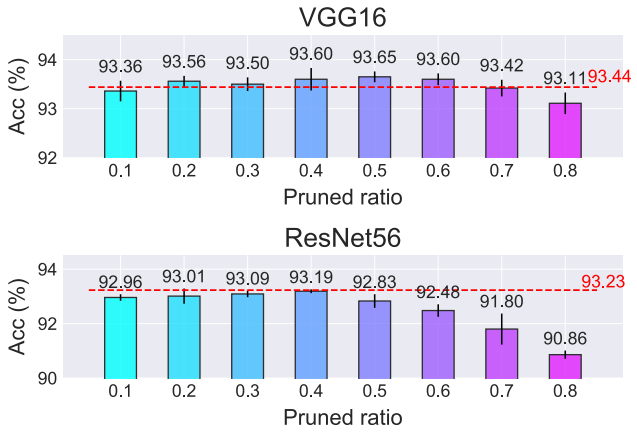

Figure S4: Effects of different pruning ratio on the model accuracy. Red dotted lines denote the baseline full models accuracy. All the models are trained on CIFAR10 dataset for five runs.

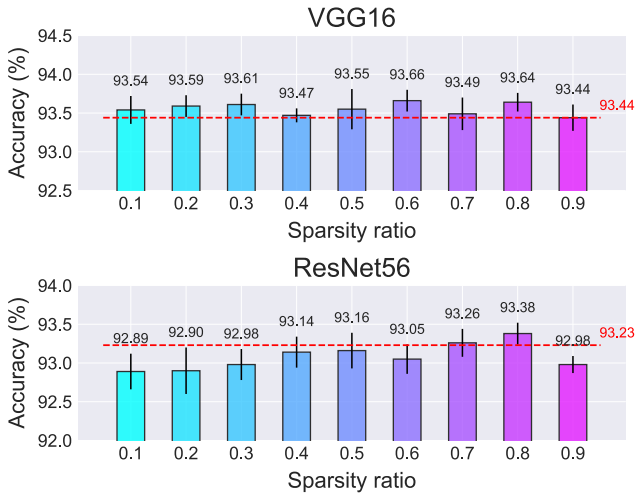

Figure S5: Effects of different sparsity ratio on the model accuarcy. Red dotted lines denote the accuracy of baseline full models.

and sparsity level.

Channel Expansion Rate

We have proposed to use a width multiplier to enlarge the channels of each layer as channel expansion uniformly in the previous section. We further investigate the effect of different expansion rate to the final pruned model accuracy. Figure S3 displays the results. All the pruned models are required to reduce 50% FLOPS compared to the full models. From the figure, we find that a general trend of the influence is that when the expansion rate is too large, the pruned model performance will deteriorate. We also surprisingly notice that using the channel shrinkage ($0.75\times$ expansion) can even achieve higher pruned model performance in some situations. This is because the preset reduced model capacity can limit the search space, which makes the pruning algorithm easier to find efficient structures.

Pruning Ratio

In this section, we explore the performance of the pruned model under different pruning ratio. Figure S4 displays the results. For each pruned model, the channel importance is learned by setting predefined sparsity ratio r as $1-\text{pruning\_ratio}$. Also, all the models are trained under the same hyper-parameter settings with budget training scheme. From the figure, we conclude that our method is robust under different pruning ratio. Even under the extreme situation where a large portion of FLOPS is reduced, our method still achieves comparable prediction performance.

Sparsity Ratio

In this section, we explore the effects of different sparsity ratio on the performance of the pruned model. The predefined sparsity ratio r is utilized to restrict the overall sparsity of channel importance value. Figure S5 summarizes the results. All the models are required to reduce 50% FLOPS of

the original full models. From the figure, we observe that the final pruned model accuracy is not very sensitive to the sparsity ratio, though a small sparsity level may have the negative impact on the performance. This demonstrates that our method is stable for a range of sparsity ratio and does not require hyper-parameter tuning.

References

Howard, A. G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; and Adam, H. 2017. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*.

Ioffe, S., and Szegedy, C. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, 448–456.

Liu, Z.; Li, J.; Shen, Z.; Huang, G.; Yan, S.; and Zhang, C. 2017. Learning efficient convolutional networks through network slimming. In *Proceedings of the IEEE International Conference on Computer Vision*, 2736–2744.

Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; and Chen, L.-C. 2018. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 4510–4520.