Rethinking the Value of Network Pruning

Zhuang Liu 1∗ , Mingjie Sun 2∗, Tinghui Zhou 1 , Gao Huang 2 , Trevor Darrell 1
1 University of California, Berkeley
2 Tsinghua University

ICLR 2019; NIPS 2018 Workshop on Compact Deep Neural Networks (best paper award)

Outlines

- Typical pruning pipeline vs rethinking paper’s finding
- A call back to pruning algorithms + experiments
    - 3 predefined target architectures
    - 3 automatically discovered target architectures
    - Transfer learning to object detection
- Conclusions and discussions

Typical pruning pipeline

- 3-stage pipeline:

  Training --- Pruning --- Fine-tuning

- Two common beliefs:
  - One can safely remove a set of redundant parameters without significantly hurting the accuracy, when starting with training a large, over-paramterized network
  - Both the pruned architecture and its associated weights are essential for obtaining the final efficient model

Rethinking paper's finding

Conclusion:

Fine-tuning a pruned model only gives comparable or even worse performance than training that model with randomly initialized weights.

- CIFAR-10/-100, ImageNet
- VGG, ResNet, DenseNet

Pruning algorithms

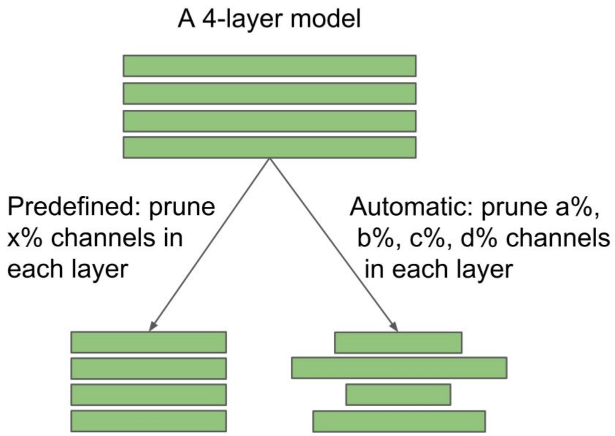


Figure 2: Difference between predefined and non-predefined (automatically discovered) target architectures. The sparsity x is user-specified, while a, b, c, d are determined by the pruning algorithm.

- Predefined: prunes *locally*
- Automatic: prunes *globally*

predefined target architecture 1

L1-norm based filter pruning

- ICLR 2017
- A certain percentage of filters with smaller L1 norm will be pruned at each layer
- data-free

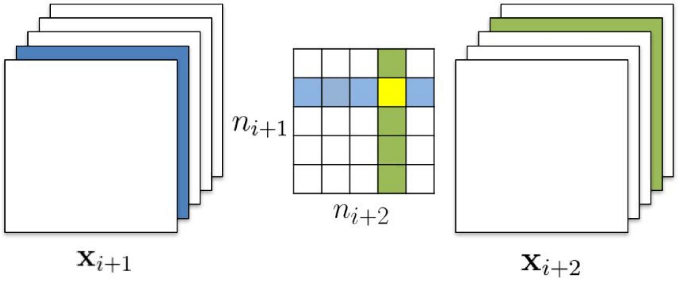
Li, H., Kadav, A., Durdanovic, I., Samet, H., & Graf, H. P. (2016). Pruning filters for efficient convnets. *arXiv preprint arXiv:1608.08710*.

predefined target architecture 1

L1-norm based filter pruning

Two strategies for layer-wise filter selection:

- *Independent (VGG or AlexNet)*
- *Greedy (ResNet)*

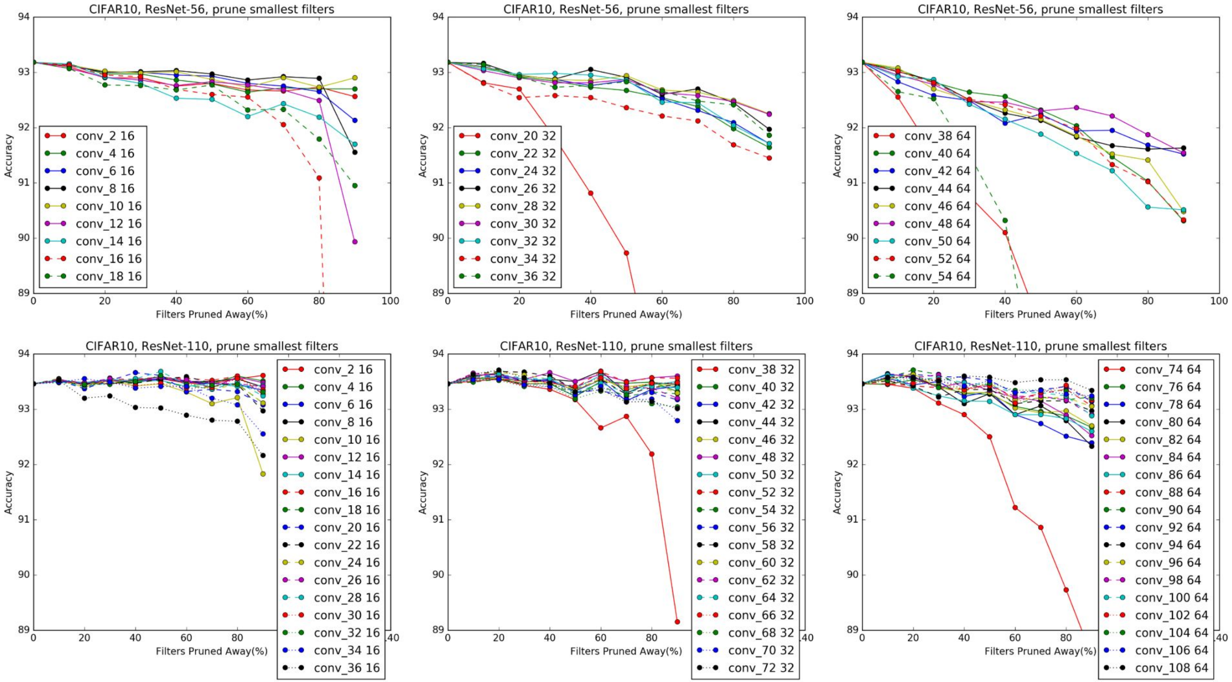


predefined target architecture 1

L1-norm based filter pruning

Two strategies for pruning the filters across multiple layers

- *Prun once and retrain (non-sensitive layers)*
- *Prun and retrain iteratively (sensitive layers)*



predefined target architecture 1

L1-norm based filter pruning

Scratch-E: same epoch as large unpruned model
Scratch-B: same computation budget as large unpruned model, more epochs

Dataset	Model	Unpruned	Pruned Model	Fine-tuned	Scratch-E	Scratch-B	Pruned %
CIFAR-10	VGG-16	93.63 (±0.16)	VGG-16-A	93.41 (±0.12)	93.62 (±0.11)	**93.78** (±0.15)	64%
	ResNet-56	93.14 (±0.12)	ResNet-56-A	92.97 (±0.17)	92.96 (±0.26)	**93.09** (±0.14)	9.4%
			ResNet-56-B	92.67 (±0.14)	92.54 (±0.19)	**93.05** (±0.18)	13.7%
	ResNet-110	93.14 (±0.24)	ResNet-110-A	93.14 (±0.16)	**93.25** (±0.29)	93.22 (±0.22)	2.3%
			ResNet-110-B	92.69 (±0.09)	92.89 (±0.43)	**93.60** (±0.25)	32.4%
ImageNet	ResNet-34	73.31	ResNet-34-A	72.56	72.77	**73.03**	7.6%
			ResNet-34-B	72.29	72.55	**72.91**	10.8%

Table 1: Results (accuracy) for L_1-norm based channel pruning (Li et al., 2017). "Pruned Model" is the model pruned from the large model. Configurations of Model and Pruned Model are both from the original paper.

predefined target architecture 2

ThiNet

- ICCV 2017

- Greedily prunes the channel that has the smallest effect on the next layer activation values

$$\arg\min_{T} \sum_{i=1}^{m} \left(\sum_{j \in T} \hat{\mathbf{x}}_{i,j} \right)^2$$
$$\text{s.t.} \quad |T| = C \times (1 - r), \quad T \subset \{1, 2, \ldots, C\}.$$

  where *m* is the number of training samples, *T* is the subset of removed channels, *r* is a predefined compression rate, and *x* is the resulting feature map after convolution

- not data-free
  (Imagenet, 10 images per class for importance evaluation)

Luo, J. H., Wu, J., & Lin, W. (2017). Thinet: A filter level pruning method for deep neural network compression. *arXiv preprint arXiv:1707.06342*.

predefined target architecture 2

ThiNet

Dataset	Unpruned	Strategy	Pruned Model		
ImageNet	VGG-16		VGG-Conv	VGG-GAP	VGG-Tiny
	71.03	Fine-tuned	−1.23	−3.67	−11.61
	71.51	Scratch-E	−2.75	−4.66	−14.36
		Scratch-B	**+0.21**	**−2.85**	**−11.58**
	ResNet-50		ResNet50-30%	ResNet50-50%	ResNet50-70%
	75.15	Fine-tuned	−6.72	−4.13	−3.10
	76.13	Scratch-E	−5.21	−2.82	−1.71
		Scratch-B	**−4.56**	**−2.23**	**−1.01**

#parameters

VGG:

Original 138.34M
VGG-Conv 131.44M
VGG-GAP 8.32M
VGG-Tiny 1.32M

predefined target architecture 3

Regression based Feature Reconstruction

- ICCV 2017
- Prunes channels by minimizing the feature map reconstruction error of the next layer, with LASSO regression
- not data-free
  (Imagenet, 5 images per class for importance evaluation)

He, Y., Zhang, X., & Sun, J. (2017, October). Channel pruning for accelerating very deep neural networks. In *International Conference on Computer Vision (ICCV)* (Vol. 2, No. 6).

predefined target architecture 3

Regression based Feature Reconstruction

Dataset	Unpruned	Strategy	Pruned Model
ImageNet	VGG-16		VGG-16-5x
	71.03	Fine-tuned	−2.67
	71.51	Scratch-E	−3.46
		Scratch-B	**−0.51**
	ResNet-50		ResNet-50-2x
	75.51	Fine-tuned	−3.25
	76.13	Scratch-E	−1.55
		Scratch-B	**−1.07**

So far ...

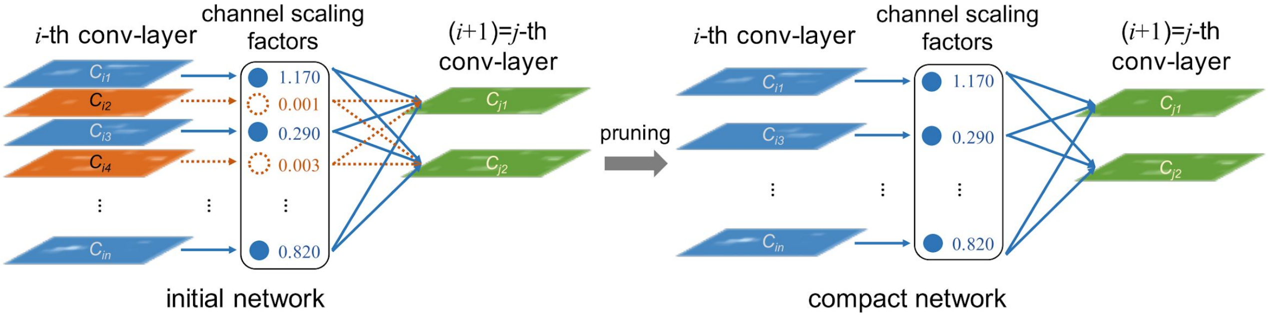
- Two common beliefs:
    - One can safely remove a set of redundant parameters without significantly hurting the accuracy, when starting with training a large, over-optimized network
      ---> Large model training at the first stage is not necessary
    - Both the pruned architecture and its associated weights are essential for obtaining the final efficient model
      ---> Preserved weights are not essential, only the architecture matters

automatically discovered target architecture 1

Network Slimming

- ICCV 2017

- Uses L1 sparsity on channel-wise scaling factors from BN layers to measure feature map importance, and prunes channels with lower scaling factors



Liu, Z., Li, J., Shen, Z., Huang, G., Yan, S., & Zhang, C. (2017, October). Learning efficient convolutional networks through network slimming. In *Computer Vision (ICCV), 2017 IEEE International Conference on* (pp. 2755-2763). IEEE.

automatically discovered target architecture 1

Network Slimming

Dataset	Model	Unpruned	Prune Ratio	Fine-tuned	Scratch-E	Scratch-B
CIFAR-10	VGG-19	93.53 (±0.16)	70%	93.60 (±0.16)	93.30 (±0.11)	**93.81** (±0.14)
	PreResNet-164	95.04 (±0.16)	40%	94.77 (±0.12)	94.70 (±0.11)	**94.90** (±0.04)
			60%	94.23 (±0.21)	94.58 (±0.18)	**94.71** (±0.21)
	DenseNet-40	94.10 (±0.12)	40%	94.00 (±0.20)	93.68 (±0.18)	**94.06** (±0.12)
			60%	**93.87** (±0.13)	93.58 (±0.21)	93.85 (±0.25)
CIFAR-100	VGG-19	72.63 (±0.21)	50%	72.32 (±0.28)	71.94 (±0.17)	**73.08** (±0.22)
	PreResNet-164	76.80 (±0.19)	40%	76.22 (±0.20)	76.36 (±0.32)	**76.68** (±0.35)
			60%	74.17 (±0.33)	75.05 (± 0.08)	**75.73** (±0.29)
	DenseNet-40	73.82 (±0.34)	40%	**73.35** (±0.17)	73.24 (±0.29)	73.19 (±0.26)
			60%	72.46 (±0.22)	72.62 (±0.36)	**72.91** (±0.34)
ImageNet	VGG-11	70.84	50%	68.62	70.00	**71.18**

automatically discovered target architecture 2

Sparse Structure Selection

- ECCV 2018
- A generalization of network slimming
- Other than channels, pruning be done on residual blocks in ResNet by adding scaling factor after each residual block

Huang, Z., & Wang, N. (2018). Data-driven sparse structure selection for deep neural networks. In *Proceedings of the European Conference on Computer Vision (ECCV)* (pp. 304-320).

automatically discovered target architecture 2

Sparse Structure Selection

Dataset	Model	Unpruned	Pruned Model	Pruned	Scratch-E	Scratch-B
ImageNet	ResNet-50	76.12	ResNet-41	75.44	75.61	**76.17**
			ResNet-32	74.18	73.77	**74.67**
			ResNet-26	71.82	72.55	**73.41**

automatically discovered target architecture 3

Non-structured Weight Pruning

- NIPS 2015
- Prunes individual weights that have small magnitudes
- L2 norm > L1 norm
- No applicable with general GPU/CPU

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* (pp. 1135-1143).

automatically discovered target architecture 3

Non-structured Weight Pruning

Dataset	Model	Unpruned	Prune Ratio	Fine-tuned	Scratch-E	Scratch-B
CIFAR-10	VGG-19	93.50 ($\pm$0.11)	30%	93.51 ($\pm$0.05)	**93.71** ($\pm$0.09)	93.31 ($\pm$0.26)
			80%	93.52 ($\pm$0.10)	**93.71** ($\pm$0.08)	93.64 ($\pm$0.09)
	PreResNet-110	95.04 ($\pm$0.15)	30%	95.06 ($\pm$0.05)	94.84 ($\pm$0.07)	**95.11** ($\pm$0.09)
			80%	**94.55** ($\pm$0.11)	93.76 ($\pm$0.10)	94.52 ($\pm$0.13)
	DenseNet-BC-100	95.24 ($\pm$0.17)	30%	95.21 ($\pm$0.17)	95.22 ($\pm$0.18)	**95.23** ($\pm$0.14)
			80%	95.04 ($\pm$0.15)	94.42 ($\pm$0.12)	**95.12** ($\pm$0.04)
CIFAR-100	VGG-19	71.70 ($\pm$0.31)	30%	71.96 ($\pm$0.36)	72.81 ($\pm$0.31)	**73.30** ($\pm$0.25)
			50%	71.85 ($\pm$0.30)	73.12 ($\pm$0.36)	**73.77** ($\pm$0.23)
	PreResNet-110	76.96 ($\pm$0.34)	30%	76.88 ($\pm$0.31)	76.36 ($\pm$0.26)	**76.96** ($\pm$0.31)
			50%	**76.60** ($\pm$0.36)	75.45 ($\pm$0.23)	76.42 ($\pm$0.39)
	DenseNet-BC-100	77.59 ($\pm$0.19)	30%	77.23 ($\pm$0.05)	77.58 ($\pm$0.25)	**77.97** ($\pm$0.31)
			50%	77.41 ($\pm$0.14)	77.65 ($\pm$0.09)	**77.80** ($\pm$0.23)
ImageNet	VGG-16	71.59	30%	73.68	72.75	**74.02**
			60%	**73.63**	71.50	73.42
	ResNet-50	76.15	30%	**76.06**	74.77	75.70
			60%	**76.09**	73.69	74.91

Network Pruning As Architecture Search

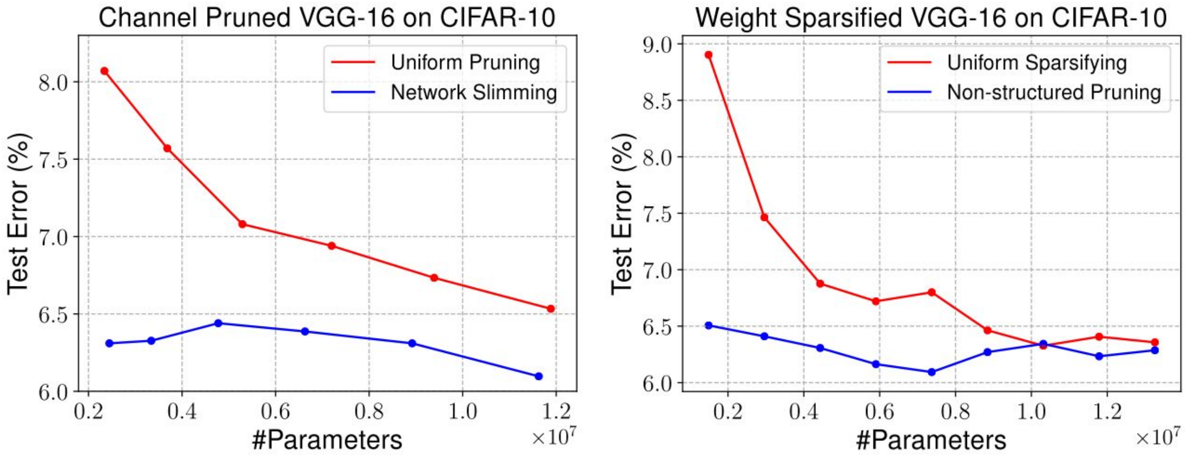


Figure 3: Pruned architectures obtained by different approaches, all *trained from scratch*, averaged over 5 runs. Architectures obtained by automatic pruning methods (*Left:* Network Slimming (Liu et al., 2017), *Right:* Non-stuctured weight pruning (Han et al., 2015)) have better parameter efficiency than uniformly pruning channels or sparsifying weights in the whole network.

Take-Home Message

- There is no clear benefit from following typical pruning pipelines with *predefined models* in order to get efficient networks, train small models from scratch instead

- Use *automatic pruning approaches* as architecture search

- When a pre-trained large model is given and little or no training budget is available, use conventional pruning methods instead of training from scratch

Discussions

- Most of prior works say scratch < fine tuning
  - Not training for a long time with scratch-B
  - Simpler-than-standard data augmentation

- Different learning rate schedules for fine tuning

Dataset	Pruned Model	Fine-tune	Scratch-E	Scratch-B	Fine-tune-restart
CIFAR-10	VGG-16-A	93.41(±0.12)	93.62(±0.11)	93.78(±0.15)	93.80(±0.07)
CIFAR-10	ResNet-56-A	92.97(±0.17)	92.96(±0.26)	93.09(±0.14)	93.46(±0.21)
CIFAR-10	ResNet-56-B	92.67(±0.14)	92.54(±0.19)	93.05(±0.18)	93.29(±0.19)
CIFAR-10	ResNet-110-A	93.14(±0.16)	93.25(±0.29)	93.22(±0.22)	93.55(±0.17)
CIFAR-10	ResNet-110-B	92.69(±0.09)	92.89(±0.43)	93.60(±0.25)	93.51(±0.15)

- Scratch-B? Or train every model until convergence?

Discussions

- Significantly pruned?
  What if the pruned models still have enough capacity to keep good accuracy?

Dataset	Model	Unpruned	Prune Ratio	Fine-tuned	Scratch-E	Scratch-B
CIFAR-10	PreResNet-164	95.04 (±0.16)	80%	91.76 (±0.38)	93.21 (±0.17)	**93.49** (±0.20)
			90%	82.06 (±0.92)	87.55 (±0.68)	**88.44** (±0.19)
	DenseNet-40	94.10 (±0.12)	80%	92.64 (±0.12)	93.07 (±0.08)	**93.61** (±0.12)
CIFAR-100	DenseNet-40	73.82 (±0.34)	80%	69.60 (±0.22)	71.04 (±0.36)	**71.45** (±0.30)

Table 13: Results (accuracy) for Network Slimming (Liu et al., 2017) when the models are significantly pruned. "Prune ratio" stands for total percentage of channels that are pruned in the whole network. Larger ratios are used than the original paper of Liu et al. (2017).

Discussions

- Lottery Ticket Hypothesis
  Dense randomly-initialized feed-forward networks contain subnetworks (winning tickets) that - when trained in isolation- arrive at comparable test accuracy in a comparable number of iterations

  Pruned non-structured models,
  trained from scratch < trained from winning tickets initialization

Frankle, J., & Carbin, M. (2018). The Lottery Ticket Hypothesis: Finding Sparse, Trainable Neural Networks.

Discussions

- Lottery Ticket

Adam with a small learning rate

Dataset	Model	Unpruned	Prune Ratio	Lottery Ticket	Random Init
CIFAR-10	VGG-19	93.50 (±0.11)	30%	**93.69** (±0.13)	93.63 (±0.16)
			80%	93.58 (±0.15)	**93.65** (±0.19)
	PreResNet-110	95.04 (±0.15)	30%	94.89 (±0.14)	**94.97** (±0.10)
			80%	**93.87** (±0.15)	93.79 (±0.17)
CIFAR-100	VGG-19	71.70 (±0.31)	30%	**72.57** (±0.58)	**72.57** (±0.23)
			50%	**72.75** (±0.22)	72.31 (±0.19)
	PreResNet-110	76.96 (±0.34)	30%	76.41 (±0.15)	**76.60** (±0.10)
			50%	**75.61** (±0.12)	75.48 (±0.17)

Table 11: Experiments on the lottery ticket hypothesis (Anonymous, 2019) with non-structured weight pruning (Han et al., 2015). 'Lottery Ticket" refers to training the pruned models with the original initialization as in Anonymous (2019). "Random Init" refers to training the pruned models with weights randomly re-initialized, as in all other experiments in this paper.

Dataset	Model	Unpruned	Pruned Model	Lottery Ticket	Random Init
CIFAR-10	VGG-16	93.63 (±0.16)	VGG-16-A	**93.62** (±0.09)	93.60 (±0.15)
	ResNet-56	93.14 (±0.12)	ResNet-56-A	92.72 (±0.10)	**92.75** (±0.26)
			ResNet-56-B	92.78 (±0.23)	**92.90** (±0.27)
	ResNet-110	93.14 (±0.24)	ResNet-110-A	**93.21** (±0.09)	**93.21** (±0.21)
			ResNet-110-B	93.15 (±0.12)	**93.37** (±0.29)

Table 12: Experiments on the lottery ticket hypothesis (Anonymous, 2019) with L_1-norm based filter pruning (Li et al., 2017). "Lottery Ticket" refers to training the pruned models with the original initialization as in Anonymous (2019). "Random Init" refers to training the pruned models with weights randomly re-initialized, as in all other experiments in this paper.