AutoSlim: Towards One-Shot Architecture Search for Channel Numbers

Jiahui Yu, and Thomas Huang
University of Illinois at Urbana-Champaign

Prerequisite:

Slimmable networks [1, 2] introduced an approach to train a single weight-shared model that can instantly adjust the runtime width with equally or even better predictive accuracy compared with the same architectures that are trained individually. In this work, we take advantages of slimmable networks to search better channel configurations.

Abstract:

A simple and one-shot solution, named ***AutoSlim***, is presented to search channel configurations in a neural network for achieving ***better accuracy*** under ***constrained resources***. Instead of training many network samples and searching with reinforcement learning, we train a single slimmable network to ***approximate*** the network accuracy of different channel configurations. We then iteratively evaluate the trained slimmable model and greedily slim the layer with minimal accuracy drop. By this single pass, we can obtain the optimized channel configurations under different resource constraints. We present experiments with MobileNet v1, MobileNet v2, ResNet-50 and RL-searched MNasNet on ImageNet classification.

1 Motivation

What is the goal of this work?

- We study how to set the number of channels in a neural network to achieve better accuracy under constrained resources (FLOPs, latency, memory footprint or model size).

Why do we want to search #channels in a network?

- The most common constraints, i.e., latency, FLOPs and runtime memory footprint, are all bound to the number of channels.
- For example, in a single convolution or fully-connected layer, the FLOPs (number of Multiply-Adds) increases linearly by the output channels.
- The memory footprint can also be reduced by reducing the number of channels in bottleneck convolutions for most vision applications.
- Despite its importance, the number of channels has been chosen mostly based on heuristics in previous methods.

2 Previous Heuristics on Setting #Channels

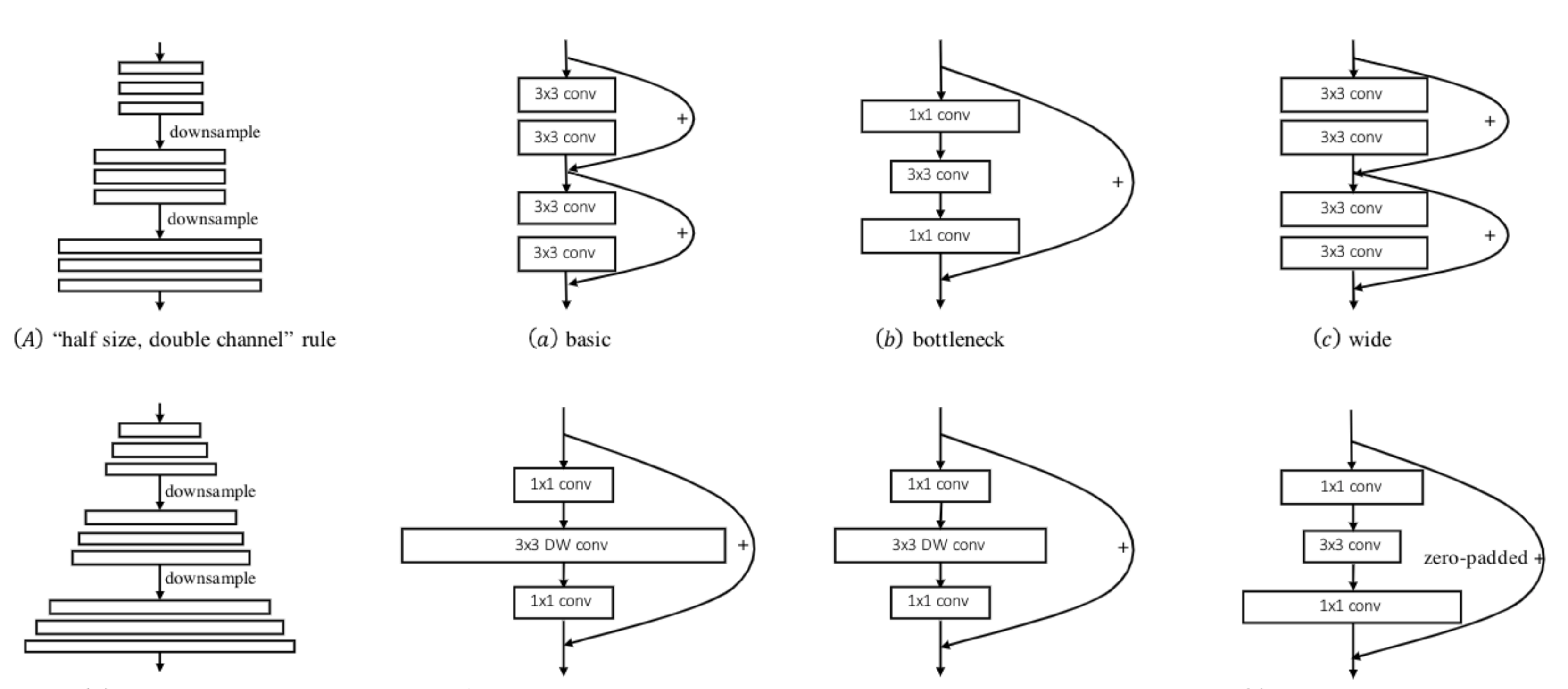
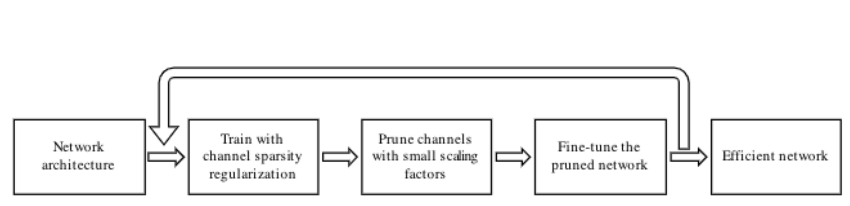
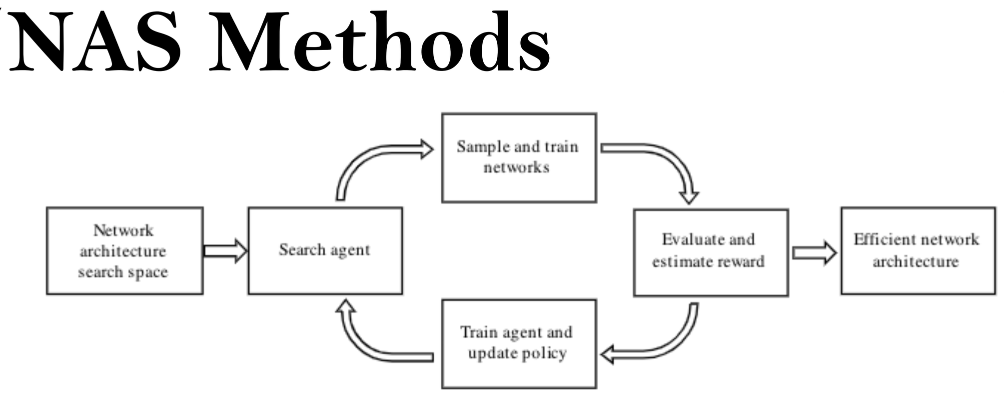


Figure 1: Various heuristics for setting channel numbers across entire network $((A)-(B))$ (Simonyan & Zisserman, 2014; Han et al., 2017; Zhang et al., 2017a), and inside network building blocks $((a)-(f))$ (Sandler et al., 2018; He et al., 2016; Han et al., 2017; Zhang et al., 2017a; Tan et al., 2018; Cai et al., 2018).

3 Previous Pruning/NAS Methods



(a) The pipeline of network pruning methods (Liu et al., 2017b).



(b) The pipeline of network architecture search methods (Tan et al., 2018; He et al., 2018)

4 *AutoSlim* Algorithm (Ours)

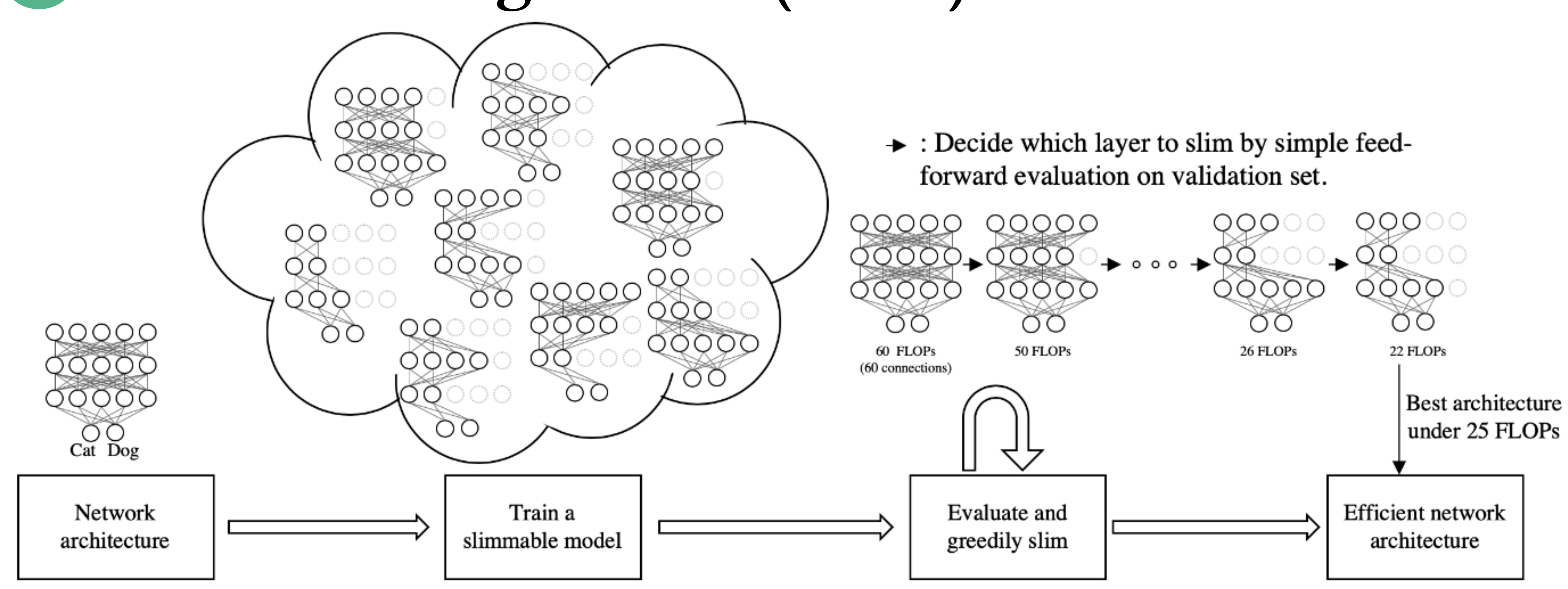


Figure 2: The flow diagram of our proposed approach *AutoSlim*.

AutoSlim has two essential steps:

1. Given a network architecture (e.g., MobileNets, ResNets), we first train a slimmable model for a few epochs (e.g., 10% to 20% of full training epochs). During the training, many different sub-networks with diverse channel configurations have been sampled and trained. Thus, after training one can directly sample its sub-network architectures for instant inference, using the correspondent computational graph and same trained weights.
2. Next, we iteratively evaluate the trained slimmable model on the validation set. In each iteration, we decide which layer to slim by comparing their feed-forward evaluation accuracy on validation set. We greedily slim the layer with minimal accuracy drop, until reaching the efficiency constraints. No training is required in this step.

5 ImageNet Classification Results

ImageNet classification results with various network architectures. Blue indicates the network pruning methods, Cyan indicates the network architecture search methods and Red indicates our results using *AutoSlim*.

Group	Model	Parameters	Memory	CPU Latency	FLOPs	Top-1 Err. (gain)
200M FLOPs	ShuffleNet v1 1.0×	1.8M	4.9M	46ms	138M	32.6
	ShuffleNet v2 1.0×	-	-	-	146M	30.6
	MobileNet v1 0.5×	1.3M	3.8M	33ms	150M	36.7
	MobileNet v2 0.75×	2.6M	8.5M	71ms	209M	30.2
	AMC-MobileNet v2	2.3M	7.3M	68ms	211M	29.2 (1.0)
	MNasNet 0.75×	3.1M	7.9M	65ms	216M	28.5
	AutoSlim-MobileNet v1	1.9M	4.2M	33ms	150M	32.1 (4.6)
	AutoSlim-MobileNet v2	4.1M	9.1M	70ms	207M	27.0 (3.2)
	AutoSlim-MNasNet	4.0M	7.5M	62ms	217M	26.8 (1.7)
300M FLOPs	ShuffleNet v1 1.5×	3.4M	8.0M	60ms	292M	28.5
	ShuffleNet v2 1.5×	-	-	-	299M	27.4
	MobileNet v1 0.75×	2.6M	6.4M	48ms	325M	31.6
	MobileNet v2 1.0×	3.5M	10.2M	81ms	300M	28.2
	NetAdapt-MobileNet v1	-	-	-	285M	29.9 (1.7)
	AMC-MobileNet v1	1.8M	5.6M	46ms	285M	29.5 (2.1)
	MNasNet 1.0×	4.3M	9.8M	76ms	317M	26.0
	AutoSlim-MobileNet v1	4.0M	6.8M	43ms	325M	28.5 (3.1)
	AutoSlim-MobileNet v2	5.7M	10.9M	77ms	305M	25.8 (2.4)
	AutoSlim-MNasNet	6.0M	10.3M	71ms	315M	25.4 (0.6)
500M FLOPs	ShuffleNet v1 2.0×	5.4M	11.6M	92ms	524M	26.3
	ShuffleNet v2 2.0×	-	-	-	591M	25.1
	MobileNet v1 1.0×	4.2M	9.3M	64ms	569M	29.1
	MobileNet v2 1.3×	5.3M	14.3M	106ms	509M	25.6
	MNasNet 1.3×	6.8M	14.2M	95ms	535M	24.5
	AutoSlim-MobileNet v1	4.6M	9.5M	66ms	572M	27.0 (2.1)
	AutoSlim-MobileNet v2	6.5M	14.8M	103ms	505M	24.6 (1.0)
	AutoSlim-MNasNet	8.3M	14.2M	95ms	532M	24.5
Heavy Models	ResNet-50	25.5M	36.6M	197ms	4.1G	23.9
	ResNet-50 0.75×	14.7M	23.1M	133ms	2.3G	25.1
	ResNet-50 0.5×	6.8M	12.5M	81ms	1.1G	27.9
	ResNet-50 0.25×	1.9M	4.8M	44ms	278M	35.0
	Pruned-ResNet-50 [Yihui He et al.]	-	-	-	≈2.0G	27.2
	AutoSlim-ResNet-50	23.1M	32.3M	165ms	3.0G	24.0
		20.6M	27.6M	133ms	2.0G	24.4
		13.3M	18.2M	91ms	1.0G	26.0
		7.4M	11.5M	69ms	570M	27.8

References:

[1] Yu, Jiahui, et al. "Slimmable neural networks." International Conference on Learning Representations (ICLR), 2019.
[2] Yu, Jiahui, and Thomas Huang. "Universally slimmable networks and improved training techniques." International Conference on Computer Vision (ICCV), 2019.

The 5th Workshop on Energy Efficient Machine Learning and Cognitive Computing @ NeurIPS 2019