EfficientNet

RETHINKING MODEL SCALING FOR CONVOLUTIONAL NEURAL NETWORKS

Presentation based on the paper: https://arxiv.org/pdf/1905.11946.pdf

Gene Olafsen

Overview

- To study and rethink the process of scaling up ConvNets.
- This paper proposes a new scaling method that uniformly scales all dimensions of depth/width/resolution using a simple yet highly effective compound coefficient.

Accomplishment

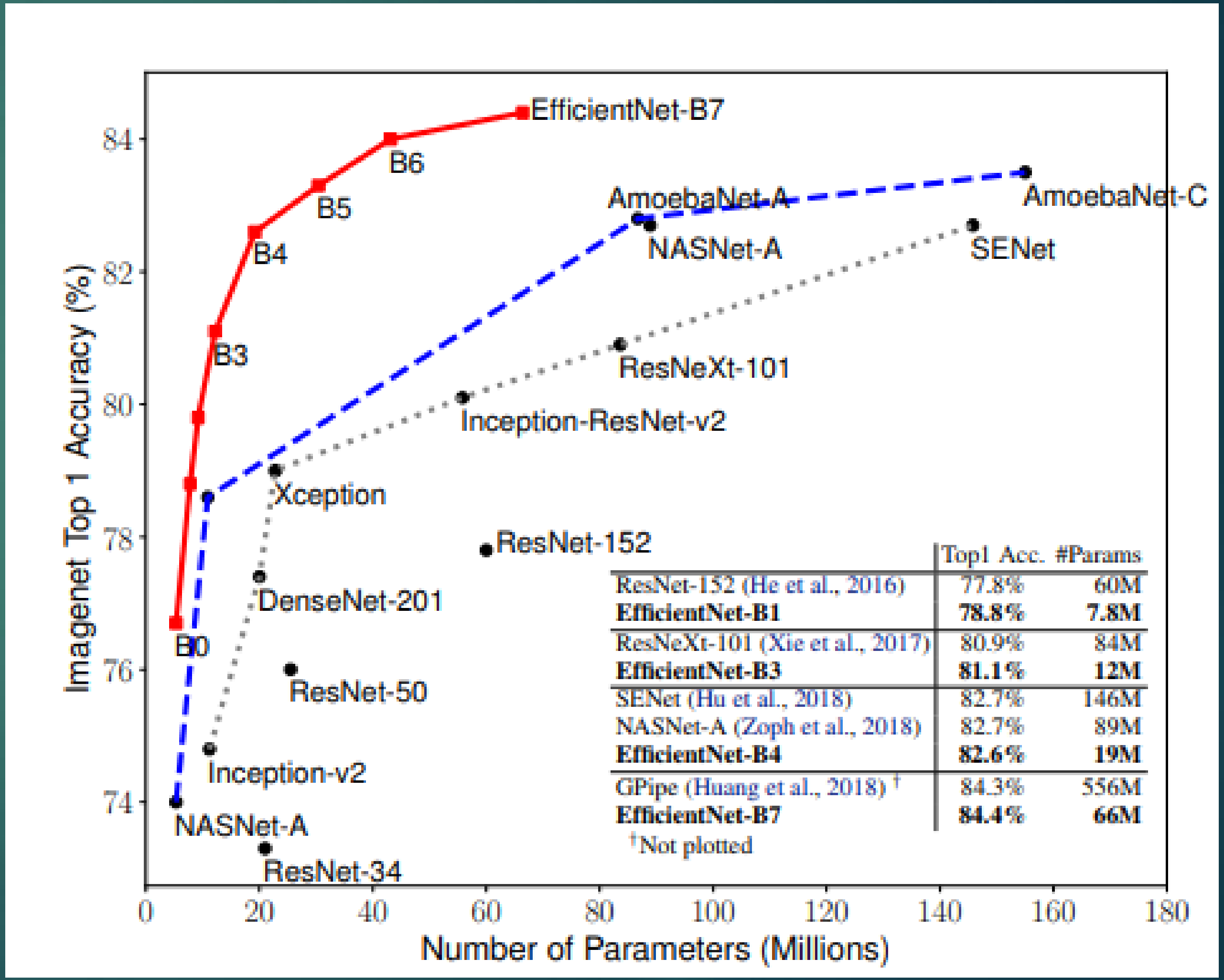
- This paper proposes a new scaling method that uniformly scales all dimensions of depth/width/resolution using a simple yet highly effective compound coefficient.
- First to empirically quantify the relationship among all three dimensions of network width, depth, and resolution.

EfficientNets

- The paper discusses how they use a neural architecture search to design a new baseline network and scale it up to obtain a family of models, called EfficientNets, which achieve much better accuracy and efficiency than previous ConvNets.

EfficientNet-B7

- EfficientNet-B7 achieves stateof-the-art 84.4% top-1 / 97.1% top-5 accuracy on ImageNet, while being 8.4x smaller and 6.1x faster on inference than the best existing ConvNet.



	Top1 Acc.	#Params
ResNet-152 (He et al., 2016)	77.8%	60M
EfficientNet-B1	**78.8%**	**7.8M**
ResNeXt-101 (Xie et al., 2017)	80.9%	84M
EfficientNet-B3	**81.1%**	**12M**
SENet (Hu et al., 2018)	82.7%	146M
NASNet-A (Zoph et al., 2018)	82.7%	89M
EfficientNet-B4	**82.6%**	**19M**
GPipe (Huang et al., 2018) †	84.3%	556M
EfficientNet-B7	**84.4%**	**66M**

†Not plotted

Transfer Learning Advantage

- The EfficientNets also transfer well and achieve state-of-the-art accuracy on CIFAR-100 (91.7%), Flowers (98.8%), and 3 other transfer learning datasets, with an order of magnitude fewer parameters.
- Transfer learning is a machine learning method where a model developed for a task is reused as the starting point for a model on a second task.
- Pre-trained models are used as the starting point on computer vision and natural language processing tasks given the vast compute and time resources required to train such neural network models.

Common Scaling Techniques

- The most common way to scale up ConvNets is by increasing their depth.
- Second most common way to scale ConvNets is to increase image resolution.

Scaling Depth Issues

- Deeper ConvNet can capture richer and more complex features, and generalize well on new tasks. However, deeper networks are also more difficult to train due to the vanishing gradient problem.
- Creative Resolution: skip connections, etc.

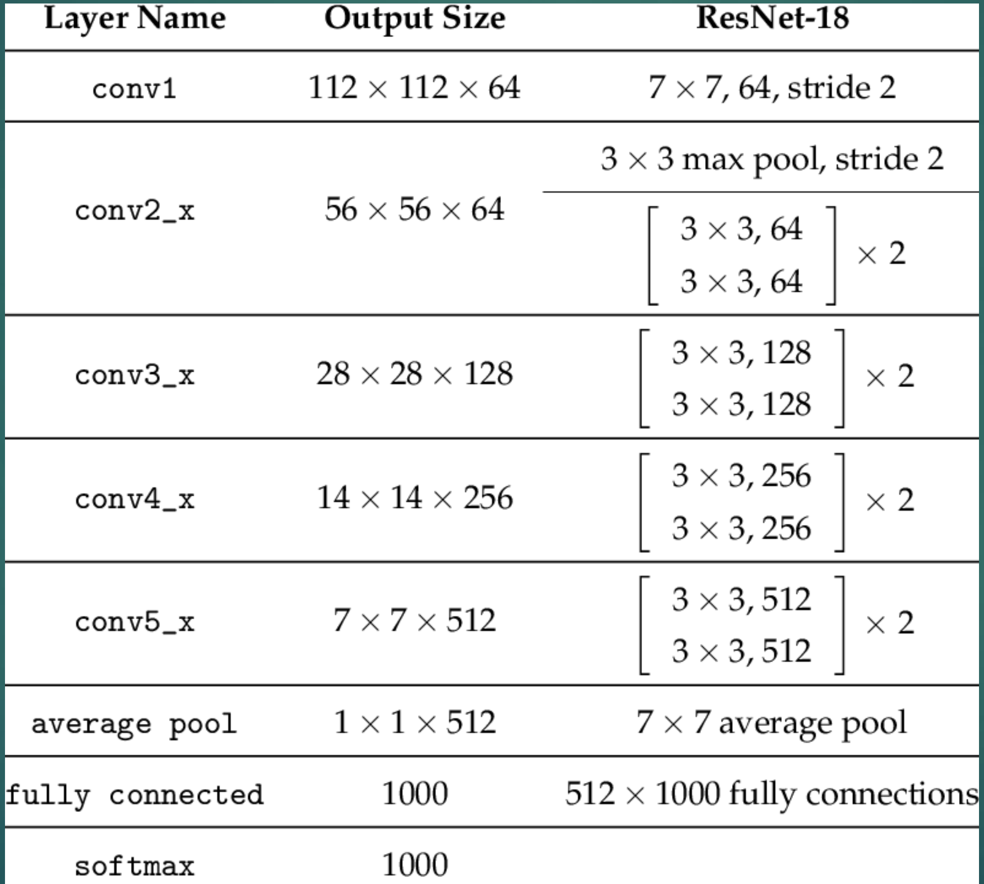
Scaling Width Issue

- Scaling network width is commonly used for small size models.
- However, extremely wide but shallow networks tend to have difficulties in capturing higher level features.

Scaling Resolution Issues

- Indeed higher resolutions improve accuracy, but the accuracy gain diminishes for very high resolutions (where hi-res is denoted by 560x560)

ResNet 18 – Example ConvNet

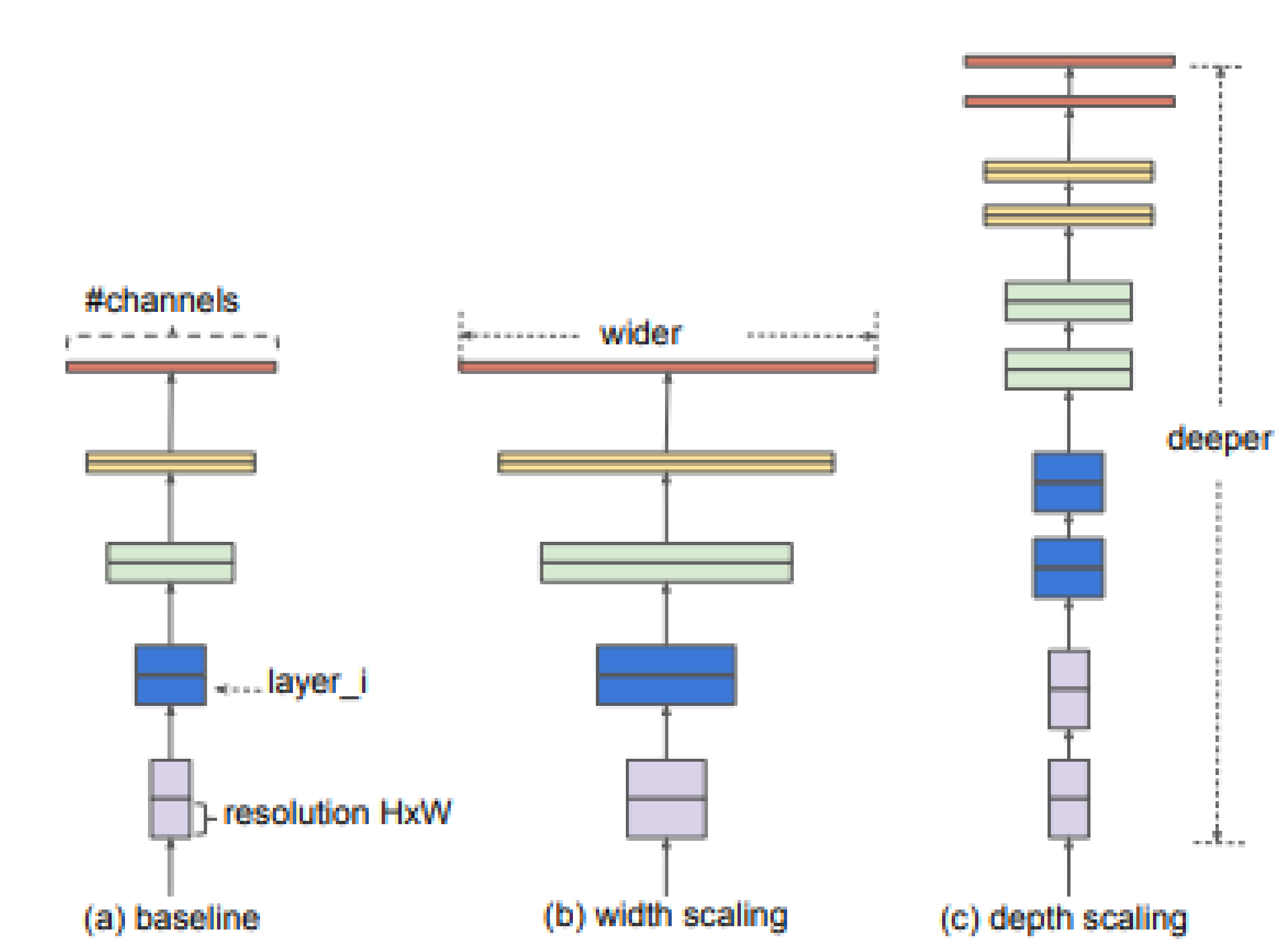


Layer Name	Output Size	ResNet-18
conv1	$112 \times 112 \times 64$	7×7, 64, stride 2
conv2_x	$56 \times 56 \times 64$	3×3 max pool, stride 2 $\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times 2$
conv3_x	$28 \times 28 \times 128$	$\begin{bmatrix} 3 \times 3, 128 \\ 3 \times 3, 128 \end{bmatrix} \times 2$
conv4_x	$14 \times 14 \times 256$	$\begin{bmatrix} 3 \times 3, 256 \\ 3 \times 3, 256 \end{bmatrix} \times 2$
conv5_x	$7 \times 7 \times 512$	$\begin{bmatrix} 3 \times 3, 512 \\ 3 \times 3, 512 \end{bmatrix} \times 2$
average pool	$1 \times 1 \times 512$	7×7 average pool
fully connected	1000	512×1000 fully connections
softmax	1000	

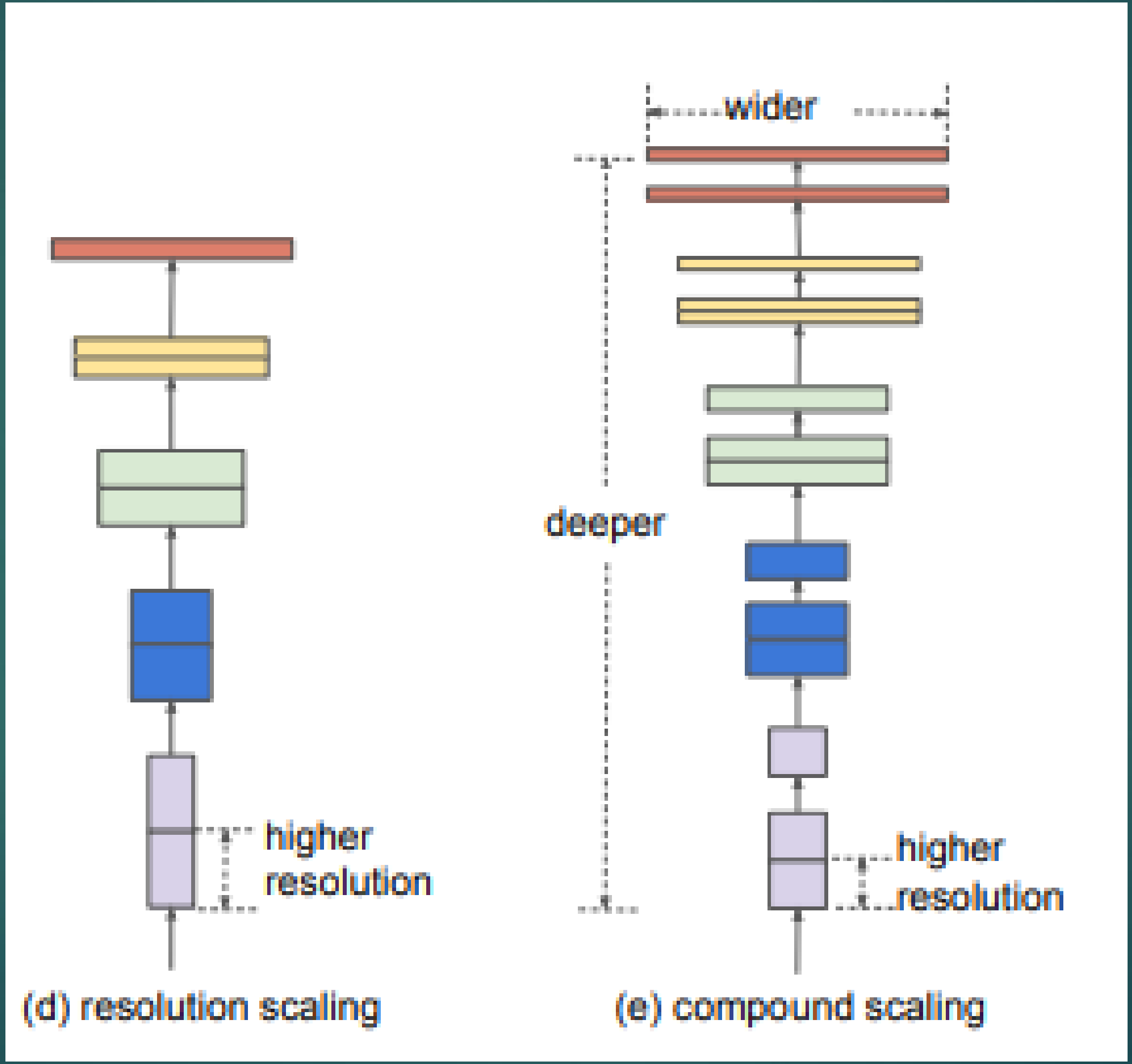
Simple Scaling?

- At issue: it is possible to scale two or three dimensions arbitrarily, arbitrary scaling requires tedious manual tuning and still often yields sub-optimal accuracy and efficiency.

Scaling


#channels
wider
deeper
layer_i
resolution HxW
(a) baseline
(b) width scaling
(c) depth scaling

Scaling (cont.)



(d) resolution scaling

(e) compound scaling

Finding

- Empirical study shows that it is critical to balance all dimensions of network width/depth/resolution, and surprisingly such balance can be achieved by simply scaling each of them with constant ratio.
- If you want to use 2N times more computational resources, then we can simply increase the network depth by αN , width by βN , and image size by γN , where α, β, γ are constant coefficients determined by a small grid search on the original small model.

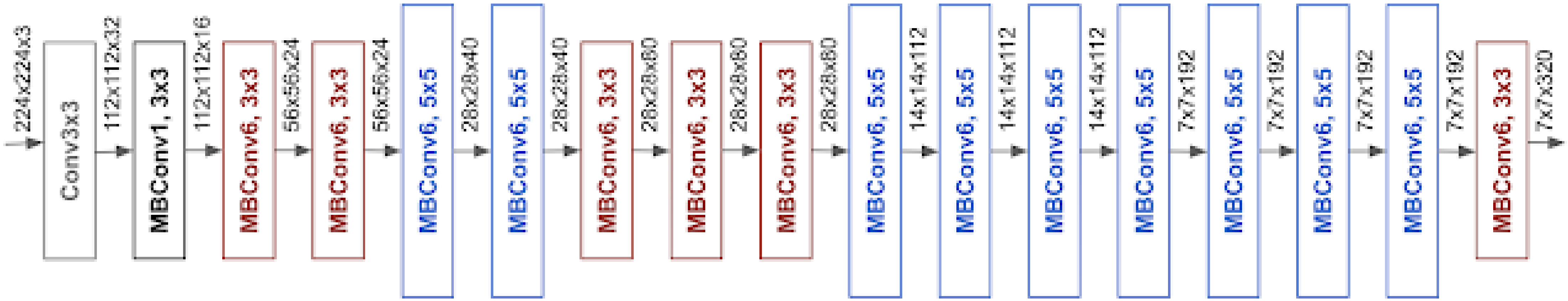
ConvNet – Related Work

- Increasing Accuracy:
  - Recently, GPipe (Huang et al., 2018) further pushes the state-of-the-art ImageNet top-1 validation accuracy to 84.3% using 557M parameters: it is so big that it can only be trained with anspecialized pipeline parallelism library by partitioning thennetwork and spreading each part to a different accelerator.
- Increasing Efficiency:
  - Model compression is used to reduce the computational power necessary to instance on mobile devices.

Compound Model Scaling

- ConvNet layers are often partitioned into multiple stages and all layers in each stage share the same architecture, this allows for some simplification in the expression of the model.
- In order to further reduce the design space, we restrict that all layers must be scaled uniformly with constant ratio
- Target... to maximize the model accuracy for any given resource constraints, which can be formulated as an optimization problem

EfficientNet Model



The architecture for our baseline network EfficientNet-B0 is simple and clean, making it easier to scale and generalize.

Function

- A ConvNet Layer i can be defined as a function: Yi = Fi(Xi), where Fi is the operator, Yi is output tensor, Xi is input tensor, with tensor shape hHi , Wi , Cii 1 , where Hi and Wi are spatial dimension and Ci is the channel dimension.
- Where F Li i denotes layer Fi is repeated Li times in stage i, hHi , Wi , Cii denotes the shape of input tensor X of layer i

$$\mathcal{N} = \bigodot_{i=1...s} \mathcal{F}_i^{L_i}\left(X_{\langle H_i, W_i, C_i\rangle}\right)$$

Model Scaling

- Unlike regular ConvNet designs that mostly focus on finding the best layer architecture Fi, model scaling tries to expand the network length (Li), width (Ci), and/or resolution (Hi, Wi) without changing Fi predefined in the baseline network.
- By fixing Fi, model scaling simplifies the design problem for new resource constraints, but it still remains a large design space to explore different Li, Ci, Hi, Wi for each layer. In order to further reduce the design space, we restrict that all layers must be scaled uniformly with constant ratio.

Observations

- Observation 1 – Scaling up any dimension of network width, depth, or resolution improves accuracy, but the accuracy gain diminishes for bigger models.
- Observation 2 – In order to pursue better accuracy and efficiency, it is critical to balance all dimensions of network width, depth, and resolution during ConvNet scaling

Approach

- Approach targets and optimizes FLOPS over latency.
- It is possible to achieve even better performance by searching for a, β, γ directly around a large model, but the search cost becomes prohibitively more expensive on larger models. Our method solves this issue by only doing search once on the small baseline network (step 1), and then use the same scaling coefficients for all other models (step 2)

Scale and Target

- The first step in the compound scaling method is to perform a grid search to find the relationship between different scaling dimensions of the baseline network under a fixed resource constraint (e.g., 2x more FLOPS).This determines the appropriate scaling coefficient for each of the dimensions mentioned above. Then apply those coefficients to scale up the baseline network to the desired target model size or computational budget.

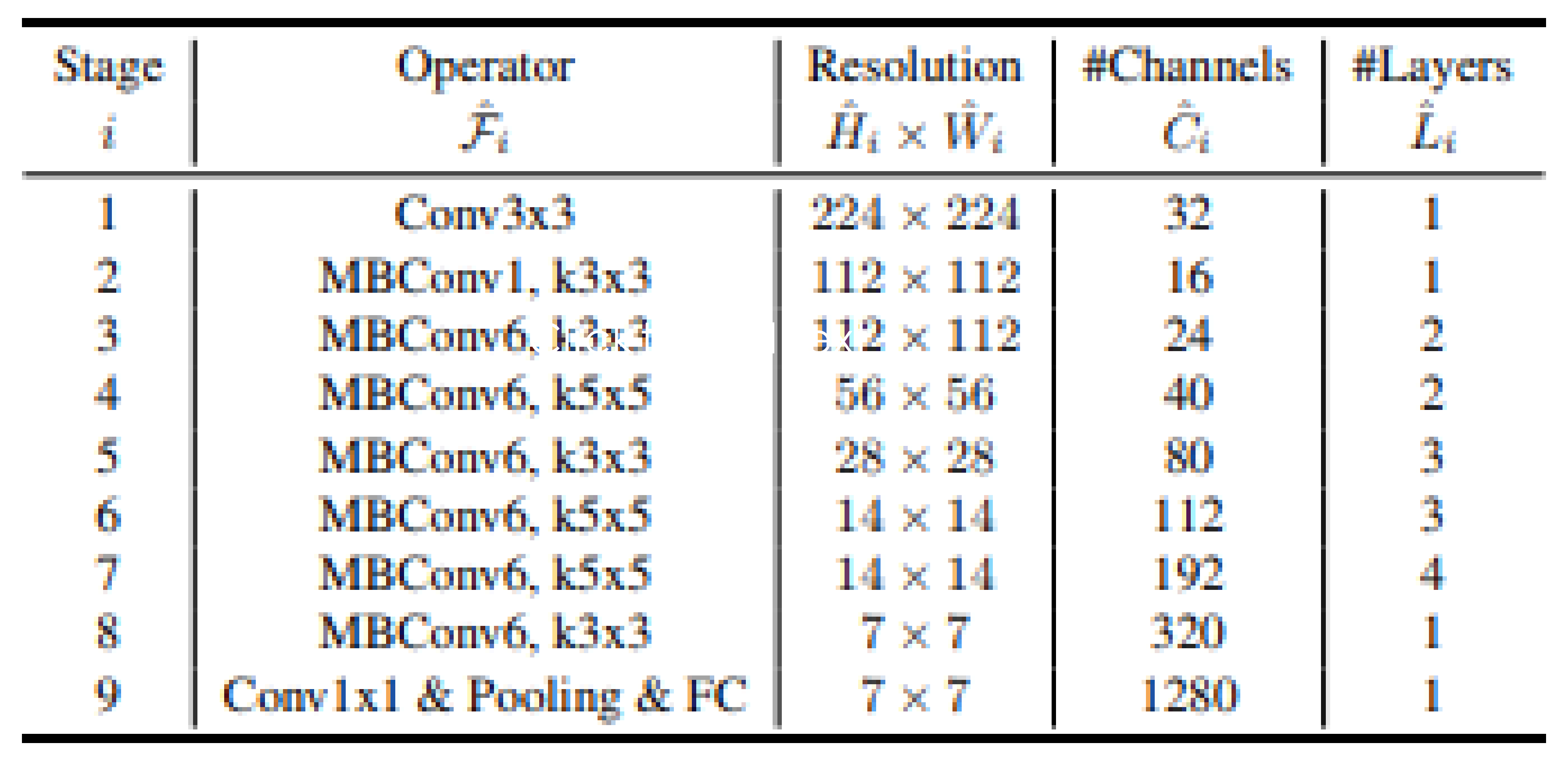
Compound Scaling Method

- Compound coefficient φ to uniformly scales network width, depth, and resolution in a principled way:
- depth: $d = a \varphi$
- width: $w = \beta \varphi$
- resolution: $r = \gamma \varphi$
  - s.t. $a \cdot \beta 2 \cdot \gamma 2 \approx 2$
  - $a \geq 1, \beta \geq 1, \gamma \geq 1$

- Compound coefficient φ to uniformly scales network width, depth, and resolution in a principled way

EfficientNet-B0 baseline network

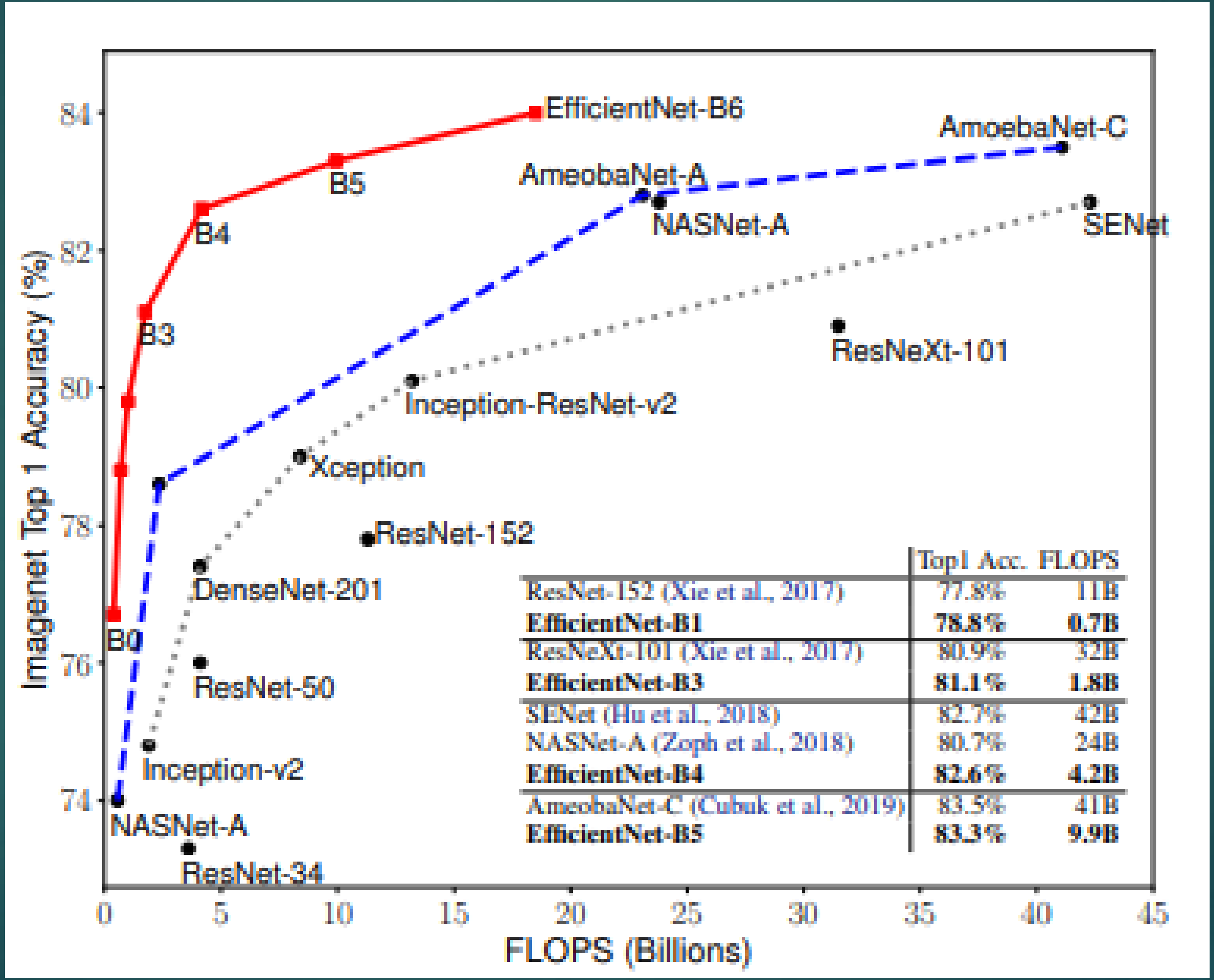
- The FLOPS target is 400M



Stage i	Operator $\hat{\mathcal{F}}_i$	Resolution $\hat{H}_i \times \hat{W}_i$	#Channels $\hat{C}_i$	#Layers $\hat{L}_i$
1	Conv3x3	224 × 224	32	1
2	MBConv1, k3x3	112 × 112	16	1
3	MBConv6, k3x3	112 × 112	24	2
4	MBConv6, k5x5	56 × 56	40	2
5	MBConv6, k3x3	28 × 28	80	3
6	MBConv6, k5x5	14 × 14	112	3
7	MBConv6, k5x5	14 × 14	192	4
8	MBConv6, k3x3	7 × 7	320	1
9	Conv1x1 & Pooling & FC	7 × 7	1280	1

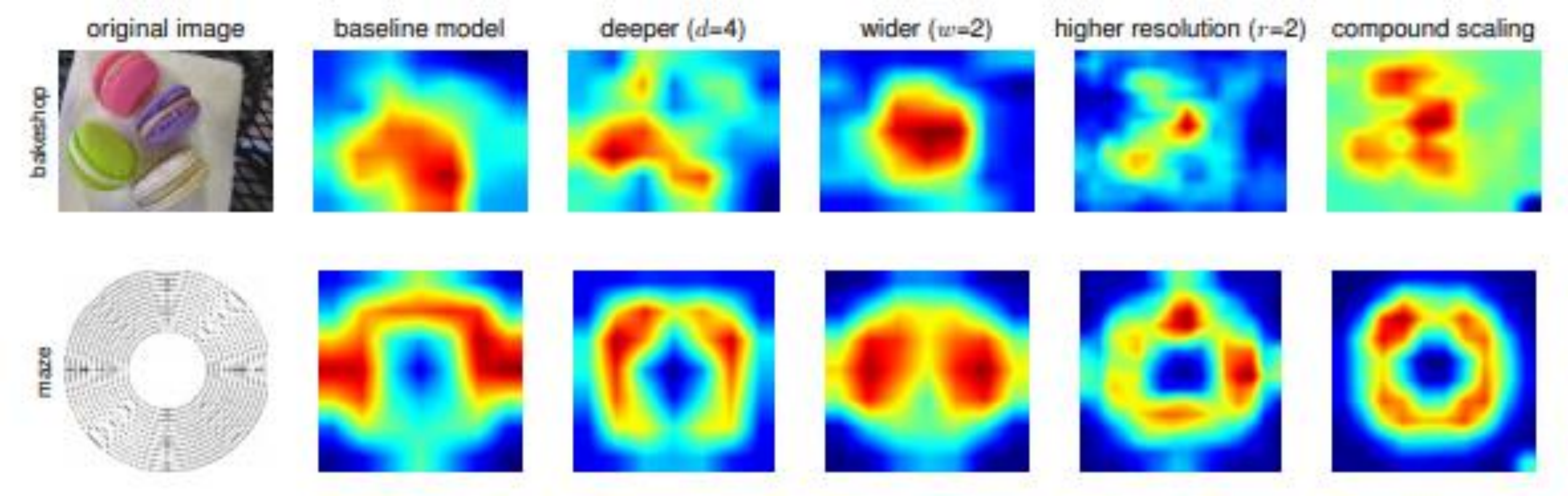
The main building block is mobile inverted bottleneck MBConv (Sandler et al., 2018; Tan et al., 2019), to which we also add squeeze-and-excitation optimization (Hu et al., 2018)

FLOPS vs. ImageNet Accuracy

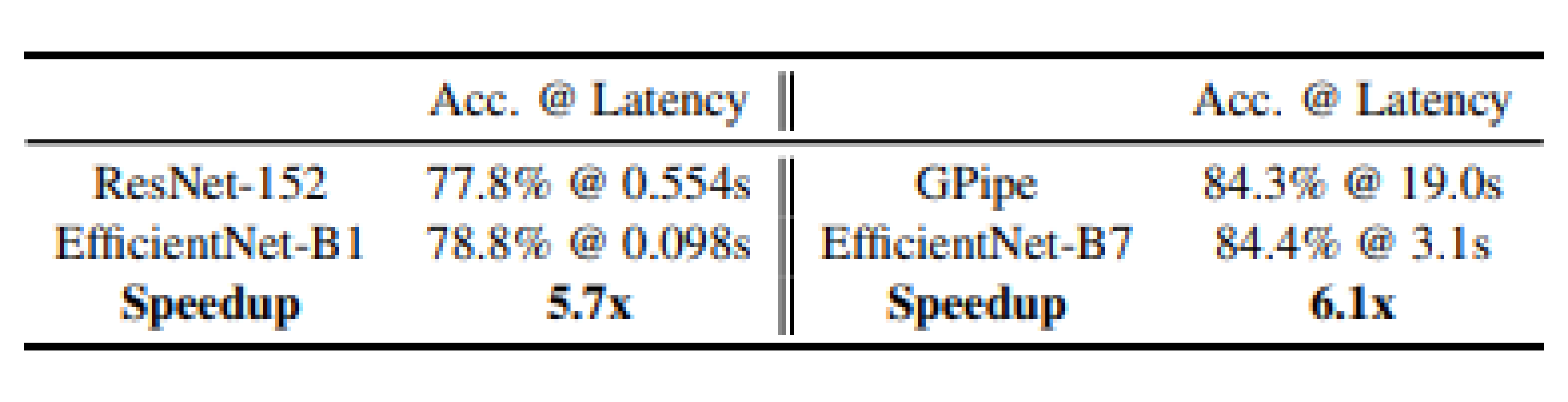


	Top1 Acc.	FLOPS
ResNet-152 (Xie et al., 2017)	77.8%	11B
EfficientNet-B1	**78.8%**	**0.7B**
ResNeXt-101 (Xie et al., 2017)	80.9%	32B
EfficientNet-B3	**81.1%**	**1.8B**
SENet (Hu et al., 2018)	82.7%	42B
NASNet-A (Zoph et al., 2018)	80.7%	24B
EfficientNet-B4	**82.6%**	**4.2B**
AmeobaNet-C (Cubuk et al., 2019)	83.5%	41B
EfficientNet-B5	**83.3%**	**9.9B**

Class Activation Map



Inference Latency



	Acc. @ Latency		Acc. @ Latency
ResNet-152	77.8% @ 0.554s	GPipe	84.3% @ 19.0s
EfficientNet-B1	78.8% @ 0.098s	EfficientNet-B7	84.4% @ 3.1s
Speedup	**5.7x**	**Speedup**	**6.1x**

Transfer Learning Efficiency

- The scaled EfficientNet models achieve new state-of-theart accuracy for 5 out of 8 datasets, with 9.6x fewer parameters on average.
- The EfficientNet-B3 achieves higher accuracy than ResNeXt101 (Xie et al., 2017) using 18x fewer FLOPS.

Training Parameters

- We train our EfficientNet models on ImageNet using similar settings as (Tan et al., 2019): RMSProp optimizer with decay 0.9 and momentum 0.9; batch norm momentum 0.99; weight decay 1e-5; initial learning rate 0.256 that decays by 0.97 every 2.4 epochs. We also use swish activation (Ramachandran et al., 2018; Elfwing et al., 2018), fixed AutoAugment policy (Cubuk et al., 2019), and stochastic depth (Huang et al., 2016) with drop connect ratio 0.2. As commonly known that bigger models need more regularization, we linearly increase dropout (Srivastava et al., 2014) ratio from 0.2 for EfficientNet-B0 to 0.5 for EfficientNet-B7.