EfficientNet:

Rethinking Model Scaling for Convolutional Neural Networks

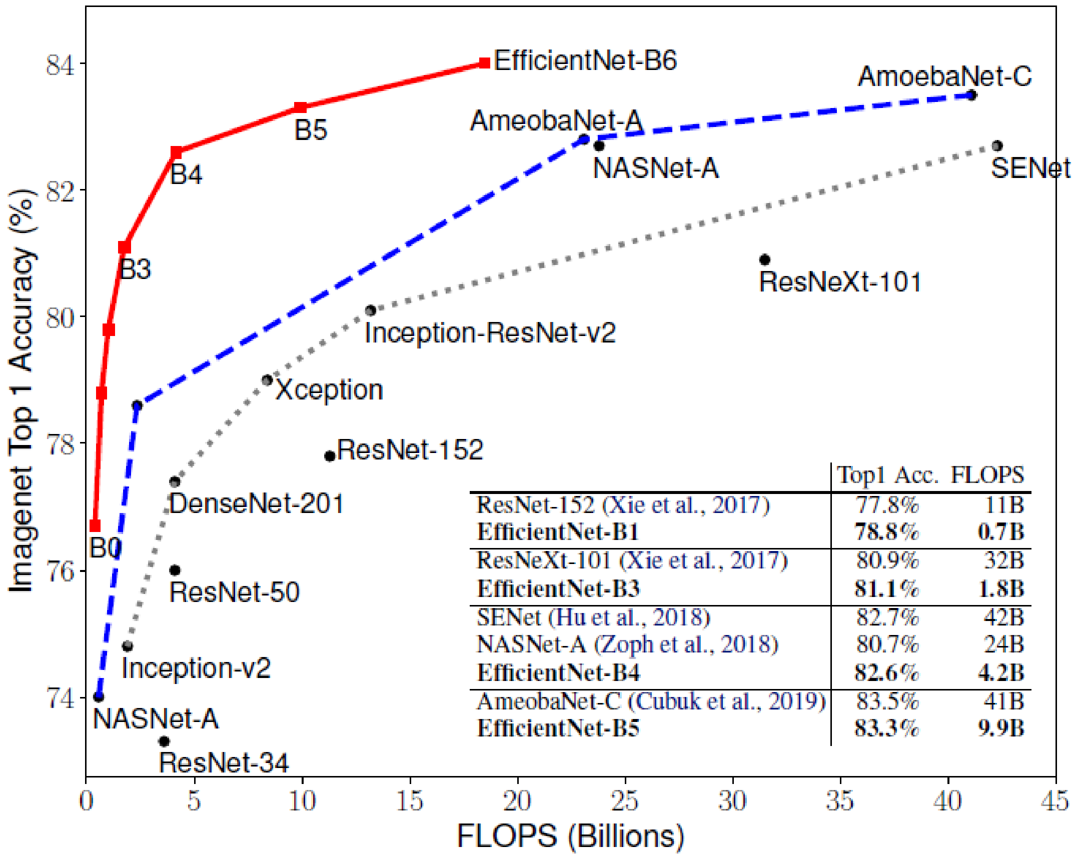
Mingxing Tan, et al., "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks", ICML 2019

References

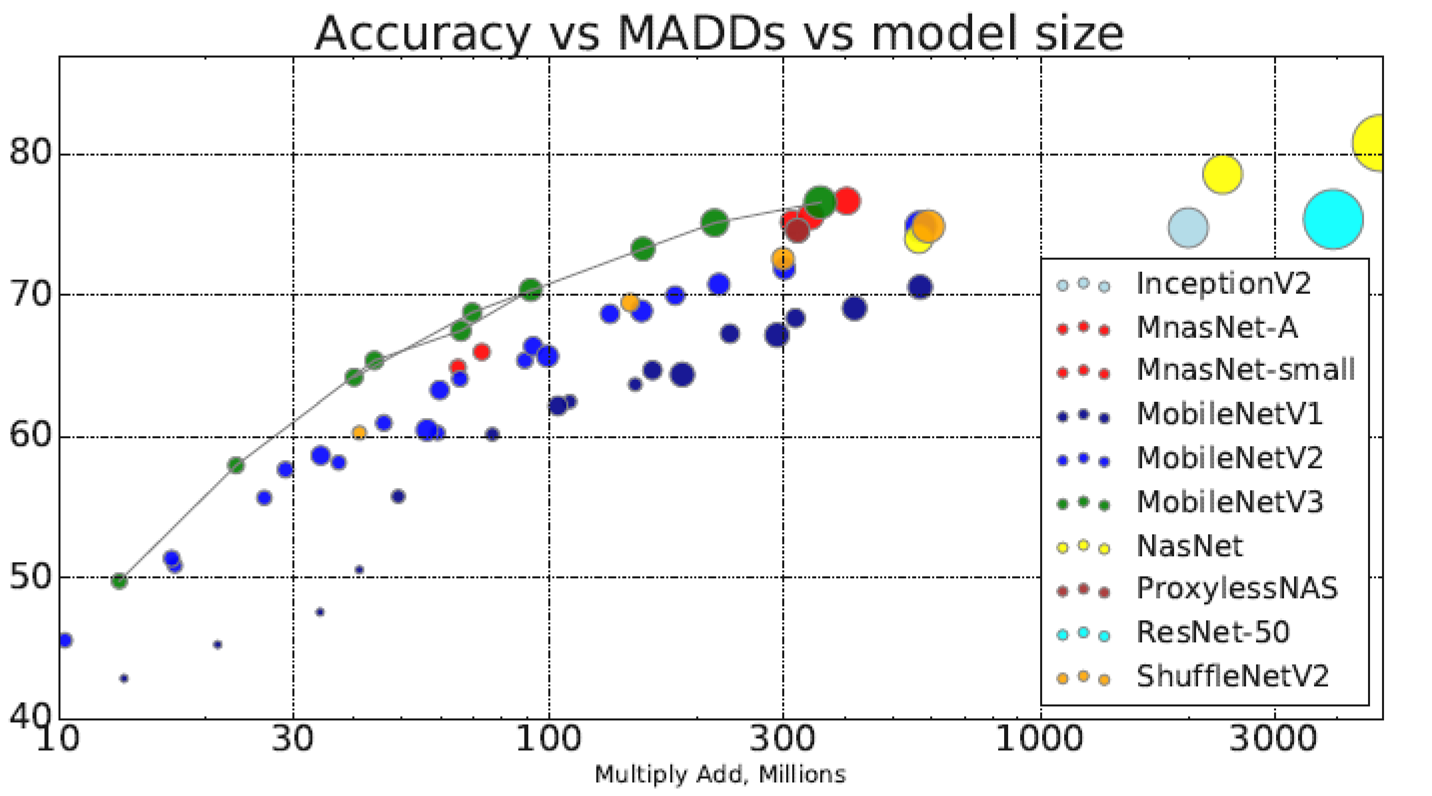
- Google AI Blog
  - https://ai.googleblog.com/2019/05/efficientnet-improving-accuracy-and.html
- Hoya012's Research Blog
  - https://hoya012.github.io/blog/EfficientNet-review/

Two Steams After ResNet

Better accuracy vs Better efficiency



	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**



Intro.

- Scaling up ConvNets is widely used to achieve better accuracy.
  - ResNet can be scaled from ResNet-18 to ResNet-200 by using more layers.
  - GPipe achived 84.3% ImageNet top-1 accuracy by scaling up a baseline model 4 times larger.

- The most common way is to scale up ConvNets by their depth, width, or image resolution.
  - In previous work, it is common to scale only one of the three dimensions.
  - Though it is possible to scale up two or three dimensions arbitrarily, arbitrary scaling requires tedious manual tuning and still often yields sub-optimal accuracy and efficiency.

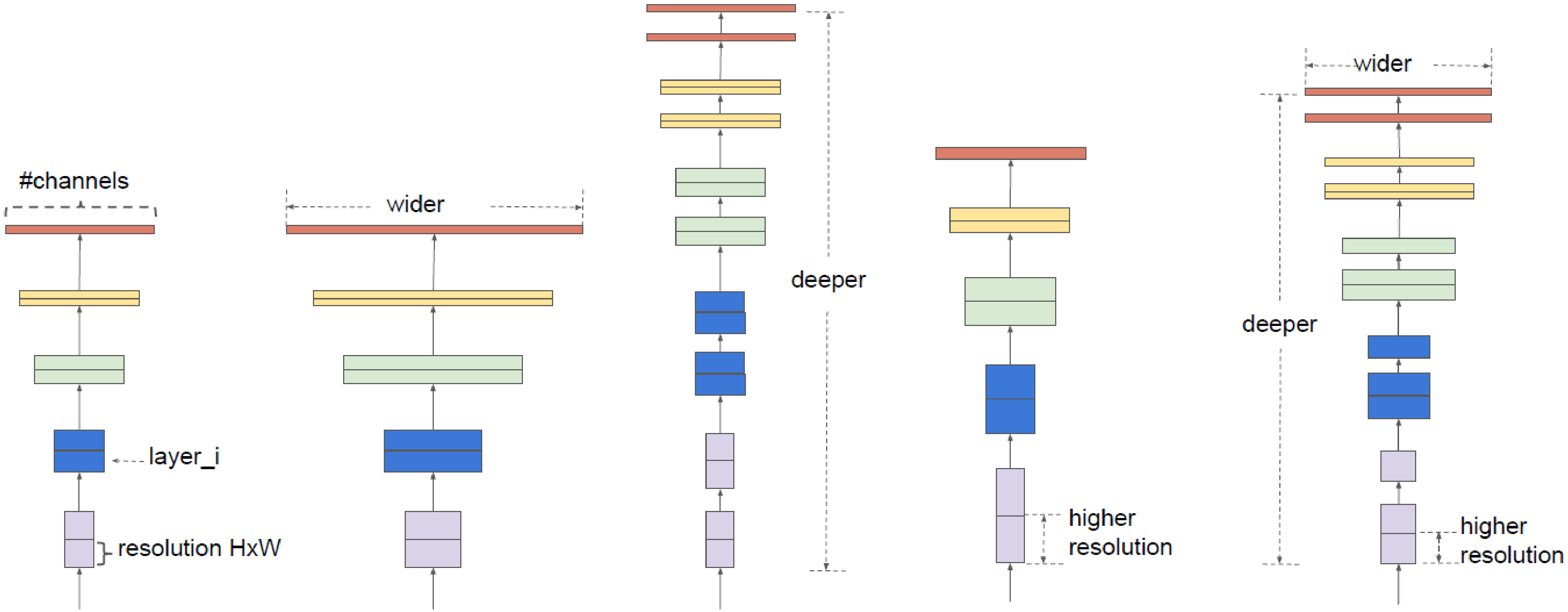
Intro.

- The authors want to study and rethink the process of scaling up ConvNets.
  - Q: **Is there a principled method to scale up ConvNets** that can achieve better accuracy and efficiency?

- 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.**

- Based on this observation, authors propose a **compound scaling methods**.

Compound Scaling



(a) baseline (b) width scaling (c) depth scaling (d) resolution scaling (e) compound scaling

Related Work – ConvNet Accuracy

- ConvNets have become increasingly more accurate by going bigger.
  - While the 2014 ImageNet winner GoogleNet (Szegedy et al., 2015) achieves 74.8% top-1 accuracy with about 6.8M parameters, the 2017 ImageNet winner SENet (Hu et al., 2018) achieves 82.7% top-1 accuracy with 145M parameters.
  - Recently, GPipe (Huang et al., 2018) further pushes the state-of-the-art ImageNet top-1 validation accuracy to 84.3% using 557M parameters.

- Although higher accuracy is critical for many applications, we have already hit the hardware memory limit, and thus **further accuracy gain needs better efficiency.**

Related Work – ConvNet Efficiency

- Deep ConvNets are often over-parameterized.
  - Model compression is a common way to reduce model size by trading accuracy for efficiency.
  - it is also common to handcraft efficient mobile-size ConvNets, such as SqueezeNets, MobileNets, and ShuffleNets.
  - Recently, neural architecture search becomes increasingly popular in designing efficient mobile-size ConvNets such as MNasNet.

- However, **it is unclear how to apply these techniques for larger models that have much larger design space and much more expensive tuning cost.**

Related Work – Model Scaling

- There are many ways to scale a ConvNet for different resource constraints
  - ResNet can be scaled down (e.g., ResNet-18) or up (e.g.,ResNet-200) by adjusting network depth (#layers).
  - WideResNet and MobileNets can be scaled by network width (#channels).
  - It is also well-recognized that bigger input image size will help accuracy with the overhead of more FLOPS.
- The Network depth and width are both important for ConvNets expressive power, it still remains an open question of how to effectively scale a ConvNet to achieve better efficiency and accuracy.

Problem Formulation

We can define ConvNets as:

input tensor

spatial dimension

channel dimension

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

stage

F_i is repeated L_i times in stage i

$$\mathcal{N} = \mathcal{F}_k \odot \dots \odot \mathcal{F}_1 \odot \mathcal{F}_1(X_1) = \bigodot_{j=1\dots k} \mathcal{F}_j(X_1)$$

Problem Formulation

- Unlike regular ConvNet designs that mostly focus on finding the best layer architecture F_i, model scaling tries to expand the network length (L_i), width (C_i), and/or resolution ($H_i;W_i$) without changing F_i predefined in the baseline network.

- By fixing F_i, model scaling simplifies the design problem for new resource constraints, but it still remains a large design space to explore different $L_i;C_i;H_i;W_i$ for each layer.

Problem Formulation

- In order to further reduce the design space, the authors restrict that **all layers must be scaled uniformly with constant ratio.**

$$\max_{d,w,r} \quad Accuracy\big(\mathcal{N}(d,w,r)\big)$$

$$s.t. \quad \mathcal{N}(\underline{d,w,r}) = \bigodot_{i=1\ldots s} \hat{\mathcal{F}}_i^{d\cdot\hat{L}_i}\big(X_{\langle r\cdot\hat{H}_i,\, r\cdot\hat{W}_i,\, w\cdot\hat{C}_i\rangle}\big)$$

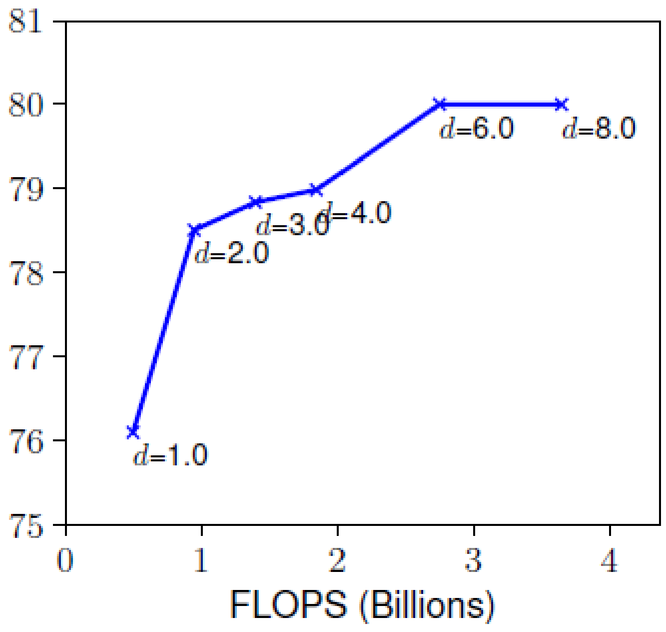
$$\text{Memory}(\mathcal{N}) \leq \text{target\_memory}$$

$$\text{FLOPS}(\mathcal{N}) \leq \text{target\_flops}$$

coefficients for scaling network width, depth and resolution

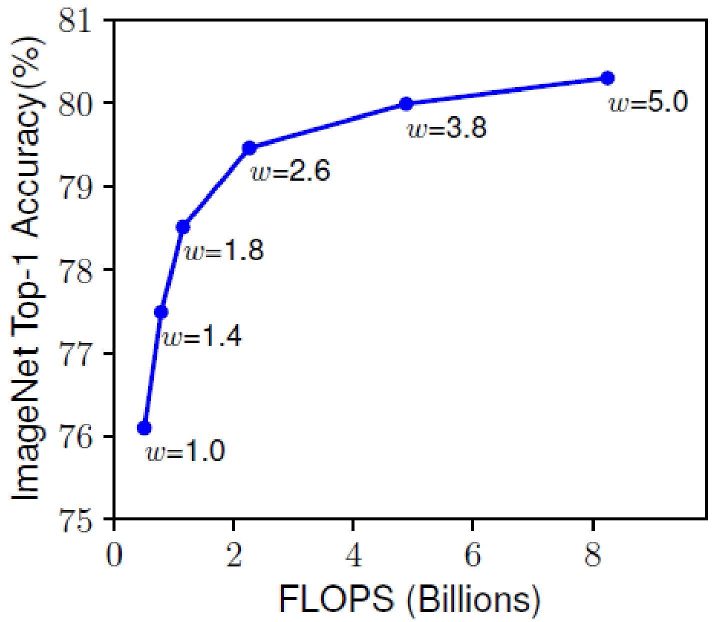
Scaling Dimensions – Depth

- The intuition is that deeper ConvNet can capture richer and more complex features, and generalize well on new tasks.
- However, the accuracy gain of very deep network diminishes.
  - For example, ResNet-1000 has similar accuracy as ResNet-101 even though it has much more layers.



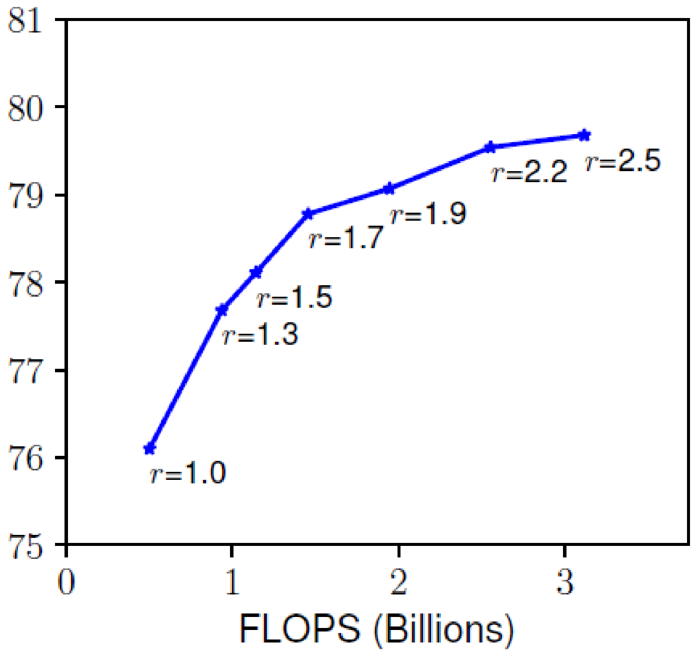
Scaling Dimensions – Width

- Scaling network width is commonly used for small size models.
- As discussed in WideResNet, wider networks tend to be able to capture more fine-grained features and are easier to train.
- However, extremely wide but shallow networks tend to have difficulties in capturing higher level features.
- And the accuracy quickly saturates when networks become much wider with larger w.



Scaling Dimensions – Resolution

- With higher resolution input images, ConvNets can potentially capture more fine-grained patterns.
  - Starting from 224x224 in early ConvNets, modern ConvNets tend to use 299x299 or 331x331 for better accuracy. Recently, GPipe achieves state-of-the-art ImageNet accuracy with 480x480 resolution.
- Higher resolutions improve accuracy, but the accuracy gain diminishes for very high resolutions.



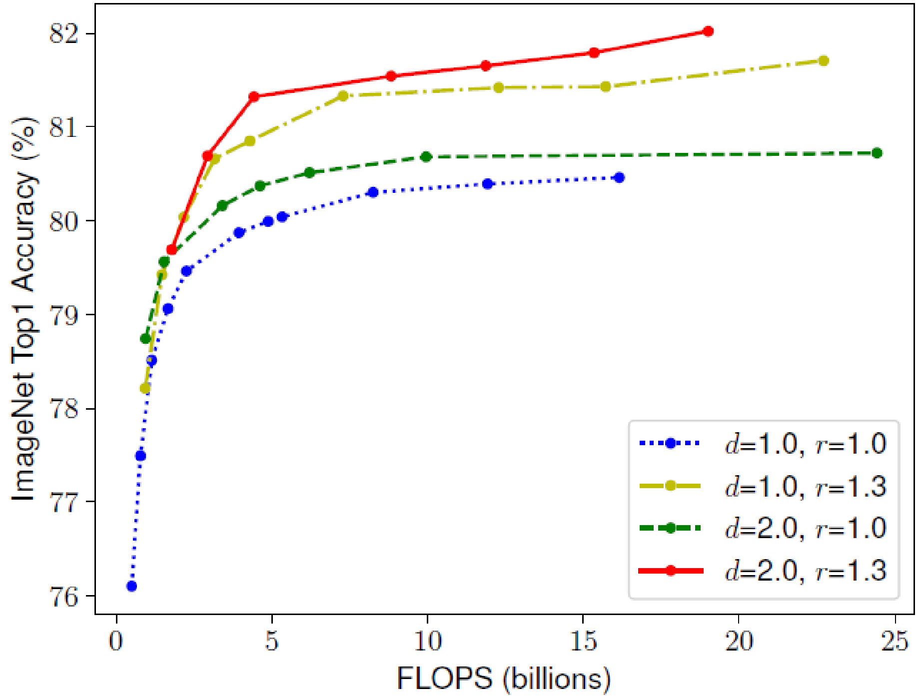
Scaling Dimensions

Observation 1

Scaling up any dimension of network width, depth, or resolution improves accuracy, but the accuracy gain diminishes for bigger models.

Compound Scaling

- Intuitively, the compound scaling method makes sense because if the input image is bigger, then the network needs more layers to increase the receptive field and more channels to capture more fine-grained patterns on the bigger image.
- If we only scale network width w without changing depth (d=1.0) and resolution (r=1.0), the accuracy saturates quickly.
- With deeper (d=2.0) and higher resolution (r=2.0), width scaling achieves much better accuracy under the same FLOPS cost.



Compound Scaling

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.

Compound Scaling Method

$$\begin{aligned}
\text{depth: } d &= \alpha^{\phi} \\
\text{width: } w &= \beta^{\phi} \\
\text{resolution: } r &= \gamma^{\phi} \\
\text{s.t. } \alpha \cdot \beta^2 \cdot \gamma^2 &\approx 2 \\
\alpha \geq 1, \beta \geq 1, \gamma &\geq 1
\end{aligned}$$

- α, β, γ are constants that can be determined by a small grid search.
- Intuitively, ϕ is a user-specified coefficient that controls how many more resources are available for model scaling.

Compound Scaling Method

$$\begin{aligned} \text{depth: } & d = \alpha^\phi \\ \text{width: } & w = \beta^\phi \\ \text{resolution: } & r = \gamma^\phi \\ \text{s.t. } & \alpha \cdot \beta^2 \cdot \gamma^2 \approx 2 \\ & \alpha \geq 1, \beta \geq 1, \gamma \geq 1 \end{aligned}$$

- Notably, the FLOPS of a regular convolution op is proportional to d, w^2, r^2.
  - Doubling network depth will double FLOPS, but doubling network width or resolution will increase FLOPS by four times. Since convolution ops usually dominate the computation cost in ConvNets, scaling a ConvNet with above equation will approximately increase total FLOPS by $(\alpha \cdot \beta^2 \cdot \gamma^2)^\phi$

- In this paper, total FLOPs approximately increase by 2^ϕ

EfficientNet Architecture

- Inspired by MNasNet, the authors develop our baseline network by leveraging a multi-objective neural architecture search that optimizes both accuracy and FLOPS.

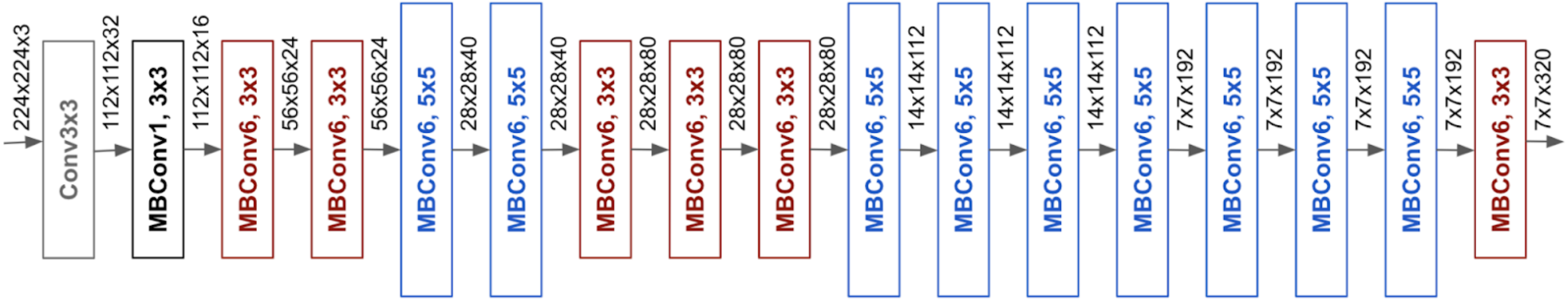
Target FLOPS

- Optimization Goal : $ACC(m) \times [FLOPS(m)/T]^w$ where w=-0.07

- Latency is not included in the optimization goal since they are not targeting any specific hardware device.

EfficientNet-B0 Baseline Network

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	28×28	112	3
7	MBConv6, k5x5	14×14	192	4
8	MBConv6, k3x3	7×7	320	1
9	Conv1x1 & Pooling & FC	7×7	1280	1



EfficientNet-B1 to B7

- Step 1:

We first fix ϕ = 1, assuming twice more resources available and do a small grid search of α, β, γ.

The best values for EfficientNet-B0 are α=1.2, β=1.1, γ=1.15.

$$\alpha \cdot \beta^2 \cdot \gamma^2 \approx 2$$

- Step 2:

We then fix α, β, γ as constants and scale up baseline network with different ϕ to obtain EfficientNet-B1 to B7.

Scaling Up MobileNets and ResNets

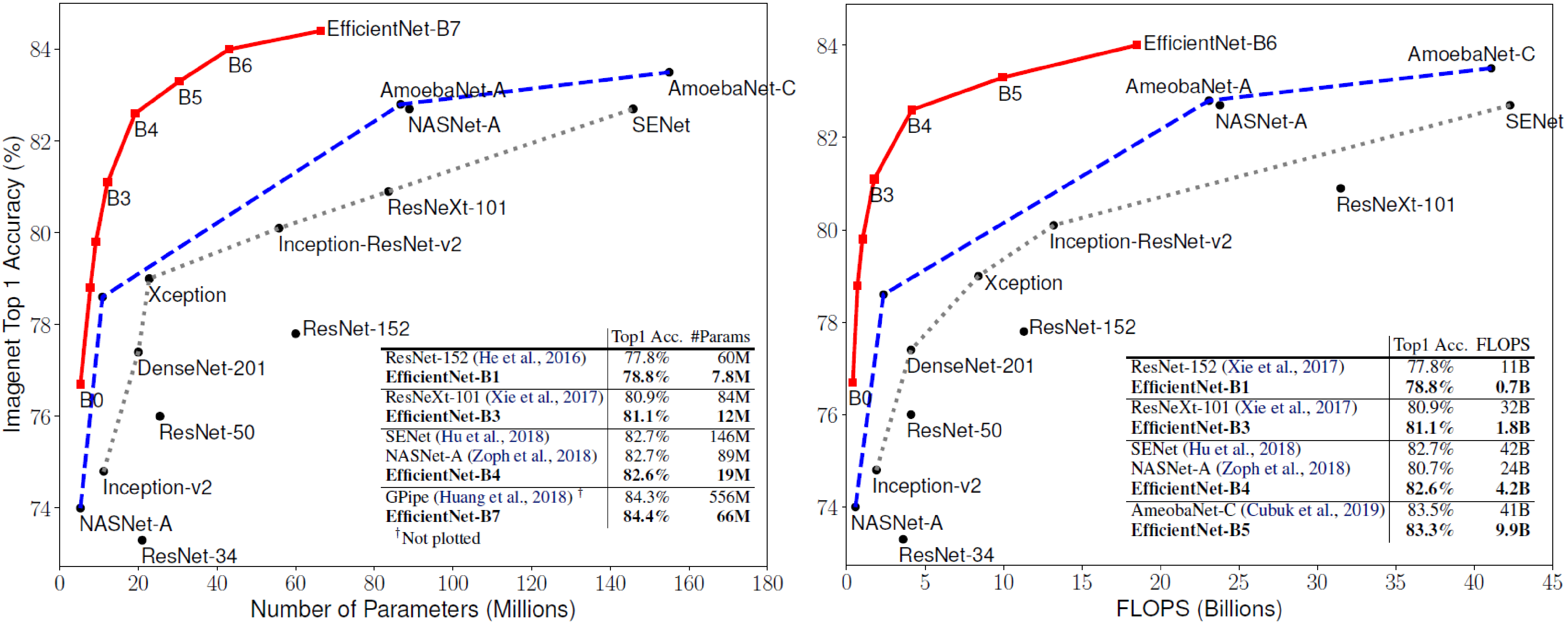
Model	FLOPS	Top-1 Acc.
Baseline MobileNetV1 (Howard et al., 2017)	0.6B	70.6%
Scale MobileNetV1 by width (w=2)	2.2B	74.2%
Scale MobileNetV1 by resolution (r=2)	2.2B	72.7%
compound scale (d=1.4, w=1.2, r=1.3)	**2.3B**	**75.6%**
Baseline MobileNetV2 (Sandler et al., 2018)	0.3B	72.0%
Scale MobileNetV2 by depth (d=4)	1.2B	76.8%
Scale MobileNetV2 by width (w=2)	1.1B	76.4%
Scale MobileNetV2 by resolution (r=2)	1.2B	74.8%
MobileNetV2 compound scale	**1.3B**	**77.4%**
Baseline ResNet-50 (He et al., 2016)	4.1B	76.0%
Scale ResNet-50 by depth (d=4)	16.2B	78.1%
Scale ResNet-50 by width (w=2)	14.7B	77.7%
Scale ResNet-50 by resolution (r=2)	16.4B	77.5%
ResNet-50 compound scale	**16.7B**	**78.8%**

ImageNet Results for EfficientNet

Model	Top-1 Acc.	Top-5 Acc.	#Params	Ratio-to-EfficientNet	#FLOPS	Ratio-to-EfficientNet
EfficientNet-B0	**76.3%**	**93.2%**	**5.3M**	**1x**	**0.39B**	**1x**
ResNet-50 (He et al., 2016)	76.0%	93.0%	26M	4.9x	4.1B	11x
DenseNet-169 (Huang et al., 2017)	76.2%	93.2%	14M	2.6x	3.5B	8.9x
EfficientNet-B1	**78.8%**	**94.4%**	**7.8M**	**1x**	**0.70B**	**1x**
ResNet-152 (He et al., 2016)	77.8%	93.8%	60M	7.6x	11B	16x
DenseNet-264 (Huang et al., 2017)	77.9%	93.9%	34M	4.3x	6.0B	8.6x
Inception-v3 (Szegedy et al., 2016)	78.8%	94.4%	24M	3.0x	5.7B	8.1x
Xception (Chollet, 2017)	79.0%	94.5%	23M	3.0x	8.4B	12x
EfficientNet-B2	**79.8%**	**94.9%**	**9.2M**	**1x**	**1.0B**	**1x**
Inception-v4 (Szegedy et al., 2017)	80.0%	95.0%	48M	5.2x	13B	13x
Inception-resnet-v2 (Szegedy et al., 2017)	80.1%	95.1%	56M	6.1x	13B	13x
EfficientNet-B3	**81.1%**	**95.5%**	**12M**	**1x**	**1.8B**	**1x**
ResNeXt-101 (Xie et al., 2017)	80.9%	95.6%	84M	7.0x	32B	18x
PolyNet (Zhang et al., 2017)	81.3%	95.8%	92M	7.7x	35B	19x
EfficientNet-B4	**82.6%**	**96.3%**	**19M**	**1x**	**4.2B**	**1x**
SENet (Hu et al., 2018)	82.7%	96.2%	146M	7.7x	42B	10x
NASNet-A (Zoph et al., 2018)	82.7%	96.2%	89M	4.7x	24B	5.7x
AmoebaNet-A (Real et al., 2019)	82.8%	96.1%	87M	4.6x	23B	5.5x
PNASNet (Liu et al., 2018)	82.9%	96.2%	86M	4.5x	23B	6.0x
EfficientNet-B5	**83.3%**	**96.7%**	**30M**	**1x**	**9.9B**	**1x**
AmoebaNet-C (Cubuk et al., 2019)	83.5%	96.5%	155M	5.2x	41B	4.1x
EfficientNet-B6	**84.0%**	**96.9%**	**43M**	**1x**	**19B**	**1x**
EfficientNet-B7	**84.4%**	**97.1%**	**66M**	**1x**	**37B**	**1x**
GPipe (Huang et al., 2018)	84.3%	97.0%	557M	8.4x	-	-

We omit ensemble and multi-crop models (Hu et al., 2018), or models pretrained on 3.5B Instagram images (Mahajan et al., 2018).

ImageNet Results for EfficientNet



	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

	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**

Inference Latency Comparison

Table 4. **Inference Latency Comparison** – Latency is measured with batch size 1 on a single core of Intel Xeon CPU E5-2690.

	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 Results for EfficientNets

Table 5. **EfficientNet Performance Results on Transfer Learning Datasets**. Our scaled EfficientNet models achieve new state-of-the-art accuracy for 5 out of 8 datasets, with 9.6x fewer parameters on average.

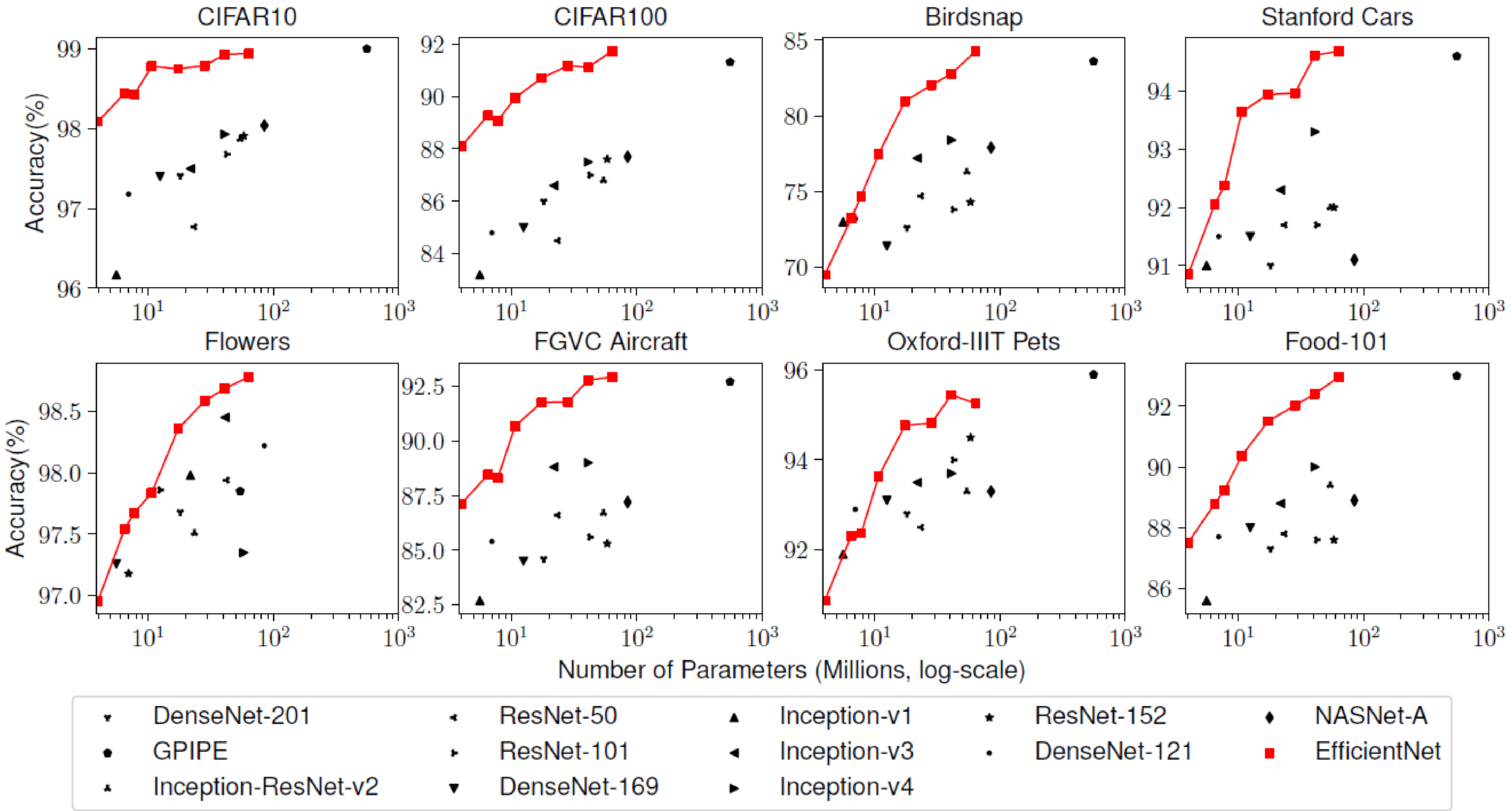
	Comparison to best public-available results						Comparison to best reported results					
	Model	Acc.	#Param	Our Model	Acc.	#Param(ratio)	Model	Acc.	#Param	Our Model	Acc.	#Param(ratio)
CIFAR-10	NASNet-A	98.0%	85M	EfficientNet-B0	98.1%	4M (21x)	†Gpipe	**99.0%**	556M	EfficientNet-B7	98.9%	64M (8.7x)
CIFAR-100	NASNet-A	87.5%	85M	EfficientNet-B0	88.1%	4M (21x)	Gpipe	91.3%	556M	EfficientNet-B7	**91.7%**	64M (8.7x)
Birdsnap	Inception-v4	81.8%	41M	EfficientNet-B5	82.0%	28M (1.5x)	GPipe	83.6%	556M	EfficientNet-B7	**84.3%**	64M (8.7x)
Stanford Cars	Inception-v4	93.4%	41M	EfficientNet-B3	93.6%	10M (4.1x)	‡DAT	**94.8%**	-	EfficientNet-B7	94.7%	-
Flowers	Inception-v4	98.5%	41M	EfficientNet-B5	98.5%	28M (1.5x)	DAT	97.7%	-	EfficientNet-B7	**98.8%**	-
FGVC Aircraft	Inception-v4	90.9%	41M	EfficientNet-B3	90.7%	10M (4.1x)	DAT	92.9%	-	EfficientNet-B7	**92.9%**	-
Oxford-IIIT Pets	ResNet-152	94.5%	58M	EfficientNet-B4	94.8%	17M (5.6x)	GPipe	**95.9%**	556M	EfficientNet-B6	95.4%	41M (14x)
Food-101	Inception-v4	90.8%	41M	EfficientNet-B4	91.5%	17M (2.4x)	GPipe	93.0%	556M	EfficientNet-B7	**93.0%**	64M (8.7x)
Geo-Mean						**(4.7x)**						**(9.6x)**

†GPipe (Huang et al., 2018) trains giant models with specialized pipeline parallelism library.
‡DAT denotes domain adaptive transfer learning (Ngiam et al., 2018). Here we only compare ImageNet-based transfer learning results.
Transfer accuracy and #params for NASNet (Zoph et al., 2018), Inception-v4 (Szegedy et al., 2017), ResNet-152 (He et al., 2016) are from (Kornblith et al., 2019).

Dataset	Train Size	Test Size	#Classes
CIFAR-10 (Krizhevsky & Hinton, 2009)	50,000	10,000	10
CIFAR-100 (Krizhevsky & Hinton, 2009)	50,000	10,000	100
Birdsnap (Berg et al., 2014)	47,386	2,443	500
Stanford Cars (Krause et al., 2013)	8,144	8,041	196
Flowers (Nilsback & Zisserman, 2008)	2,040	6,149	102
FGVC Aircraft (Maji et al., 2013)	6,667	3,333	100
Oxford-IIIT Pets (Parkhi et al., 2012)	3,680	3,369	37
Food-101 (Bossard et al., 2014)	75,750	25,250	101

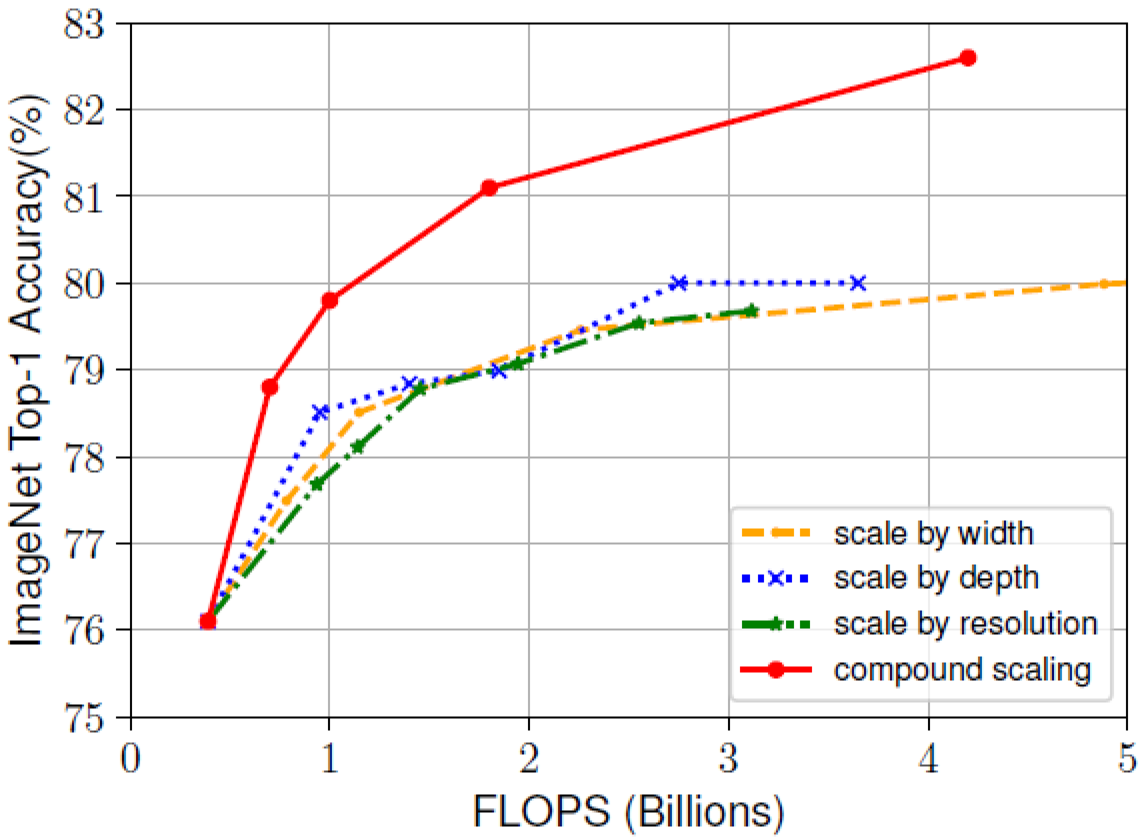
<Transfer Learning Datasets>

Transfer Learning Results for EfficientNets

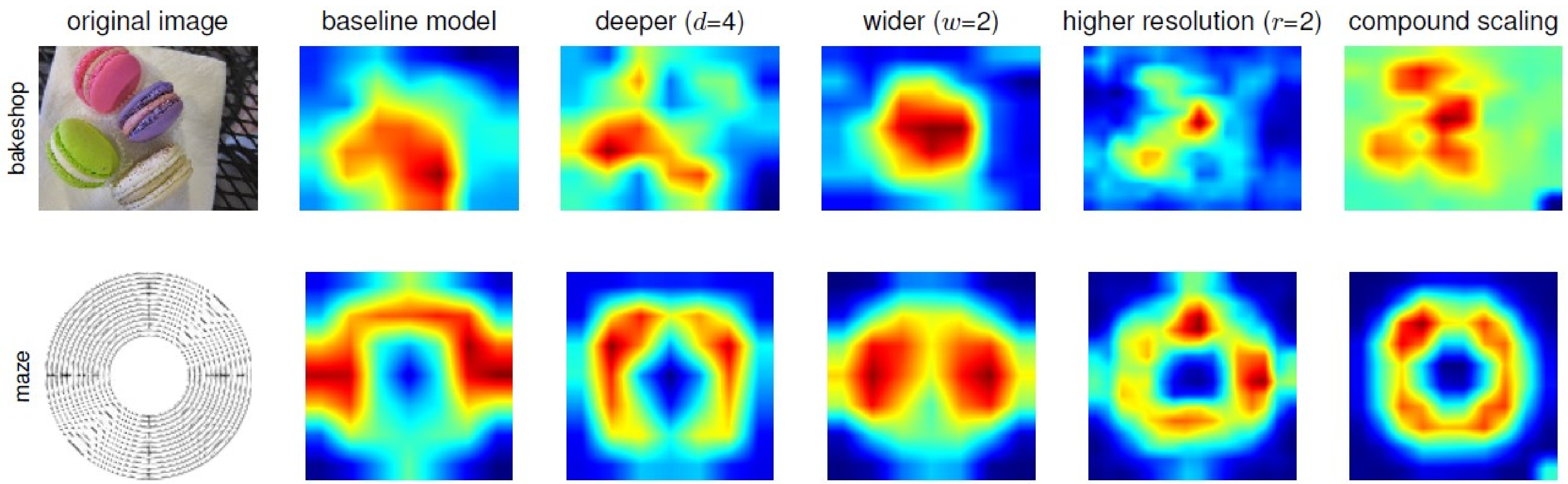


Discussion

- Disentangling the contribution of proposed scaling method from the EfficientNet architecture.



Class Activation Maps



Model	FLOPS	Top-1 Acc.
Baseline model (EfficientNet-B0)	0.4B	76.3%
Scale model by depth (d=4)	1.8B	79.0%
Scale model by width (w=2)	1.8B	78.9%
Scale model by resolution (r=2)	1.9B	79.1%
Compound Scale (d=1.4, w=1.2, r=1.3)	**1.8B**	**81.1%**