EfficientNet

Rethinking Model Scaling for Convolutional Neural Network,arXiv 2019

Image classification분야에서 매우 월등한 성능으로 큰 이목을 집중시킨 논문이다.

Abstract

CNN은 한정된 자원(하드웨어 자원을 말하는듯?)내에서 개발되어왔으며, 자원이 추가적으로 지원되는 한도 내에서 더 높은 정확도를 위해서 그 크기를 키워가는 방향으로 발전되어왔다. 이 논문에서는, model scaling에 대해 더 명확히 밝혀내기 위해 연구하게 되며, network의 depth, width, 그리고 resolution사이의 관계에 대한 균형을 맞춰야 더 나은 성능을 보인다는것을 체계적으로 밝혀낸다. 저자는 이 논문에서 모든 depth, width, resolution의 dimension들을 간단하면서도 높은 효율을 보이는 새로운 sacaling방법인 'compound coefficient'를 제안한며, MobileNet과 ResNet에 이 방법을 적용시켜봄으로써 효율성을 테스트한다.

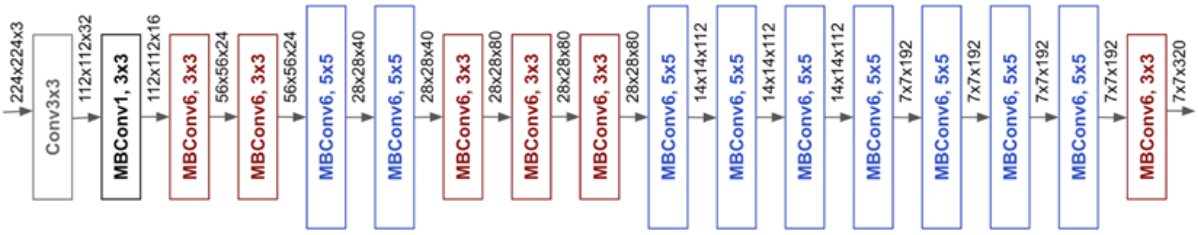
더 나아가, 저자는 'Neural Architecture Search(NAS, 강화학습 기반으로 최적의 network를 찾는 방법)'를 사용하여 baseline network를 설계하였으며 이 baeline network를 scale up 하여 가족 모델인 EfficientNet을 설계하였다. 특히, EfficientNet-B7은 ImageNet dataset에 대해 84.4%(top-1 acc)/ 97.1%(top-5 acc)를 얻었을 정도로 매우 좋은 성능을 보이는데, 이는 최신 ConvNet보다 8.4배 작으며, 6.1배 빠른 성능이다.

Scale Up

Recent Trends of CNNs

- Repeating Base Blocks

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



Scaling up ConvNets is widely used to achieve better accuracy.

- ResNet can be scaled from ResNet 18~ResNet 200 by using more layers.
- GPipe achieve 84.3% ImageNet top 1 accuracy by scaling up a baseline model 4 time larger.

What to Scale Up

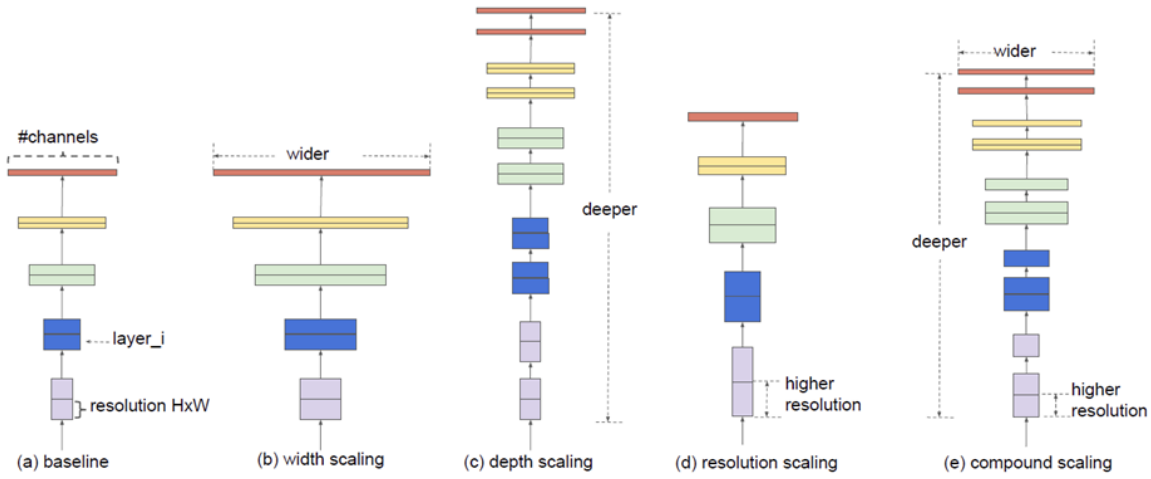
- #of Layers : Depth
- #of Channels : Width
- size of Input Images : Resolution

Why to Scale Up

- If a network has more layers

    - We can capture richer and more complex features.
- If a network has more channels
    - We can have various patterns
- If an input image is bigger
    - We can use fine-grained patterns.
    - Early networks used 224x224, but these days use 299x299, 331x331 or 480x480(Gpipe)

Width, Depth, Resolution Scaling



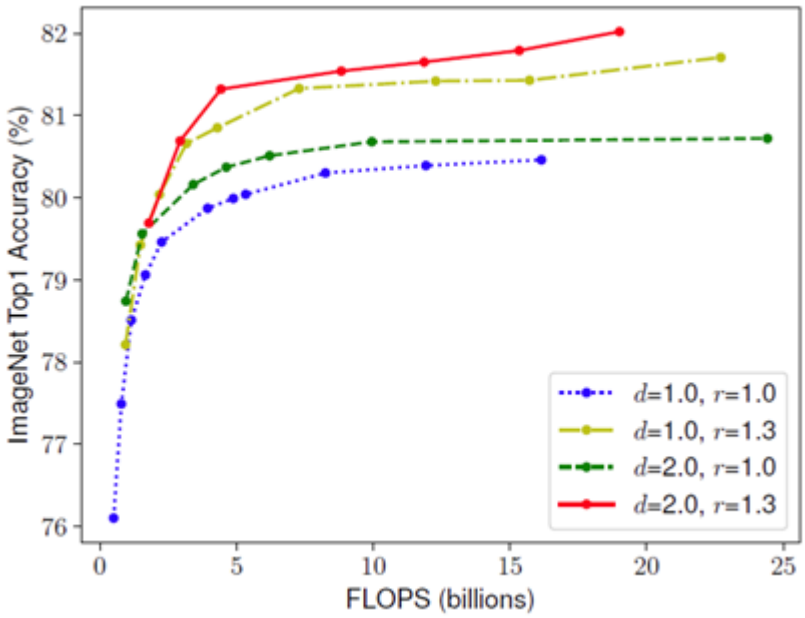
Difficulties

Saturation

- ResNet 1000 has similar accuracy as ResNet 101 even though it has much more layers.
- Hard to capture good features if networks are shallow even though it is wider.

It is critical to balance width, depth and resolution

- Scaling width w without changing depth(d=1.0) and resolution(r=1.0) results in quick saturation
- With deeper(d=2.0) and higher resolution(r=2.0), width scaling achieves much better accuracy under the same FLOPS cost.



Idea for Best Compound Scaling

1. Find out a good baseline model
2. Find out the golden ratio of width, depth and resolution for scaling
3. Scaling up each dimension of the baseline model keeping the golden ratio of width, depth and resolution

Formulation

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

s: stage,
F_i: operation of stage i,
L_i: repetition of F_i,
$X_{<H_i,W_i,C_i>}$: input

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

Compound Scaling

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

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

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

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

s: stage
F_i: operation of stage i,
L_i: repetition of F_i,
$X_{<H_i,W_i,C_i>}$: input

d: scale facor of depth
w: scale facor of width
r: scale facor of resolution

model scaling이란, 최선의 architecture를 찾는것에 집중하는 여타 ConvNet 디자인 방법과는 다르게, 기존의 baseline network에 대해 length, width, resolution등을 확장시키는것을 말한다. design space를 좁히기 위해, 저자는 모든 레이어들이 균등하게 scaling 되도록 하였으며, 한정된 resource 내에서 최고의 accuracy를 갖도록 하는 optimization 문제를 다룬다. 이를 식으로 나타내면 다음(Eq. 2)과 같다. F, L, H, W, C는 baseline network가 정해지면서 정해지며, w, d, r이 network 를 scaling하는데 사용되는 coefficient들이다.

- Assumption
    - All stages and layers share the scaling factors to reduce the search space

Flops of a CNN is proportional to d, W^2, r^2

- Doubling depth double FLOPS
- Doubling width or resolution increases FLOPS by four times

So, following constraints are added to searching for d, w and r

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

ϕ is a user specific parameter

파이는 얼마나 많은 resource를 사용할 지 에 대해 사용자가 정할 coefficient이며,알파,베타,감마가 small grid search방법으로 찾게될 변수들이다.

특히, Convolution operation의 FLOPS는 d, w^2, r^2 각각에 대해 비례하여 증감하는 성질을 갖고 있다. ConvNet의 FLOPS는 convolution operation이 지배적이므로, 위의 Eq.3에 의한 ConvNet의 FLOPS는 **(알파 * 베타^2 * 감마^2)^파이**에 비례하여 증감한다는것을 알 수 있다. Eq.3에서도 알 수 있듯, 알파 * 베타^2 * 감마^2값를 2로 제한시켰으므로, **총 FLOPS는 대략 2^파이**에 비례하여 증감한다.

Searching for Baseline Model

- By MNasNet

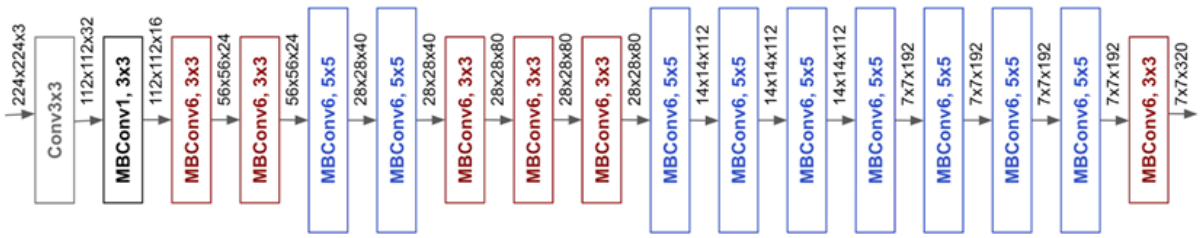
- a multi objective neural architecture search that optimizes both accuracy and FLOPS
- Optimization Goal:

$$ACC(m) \times [FLOPS(m)/T]^w$$

- The found baseline model will be scaled up for better accuracy with less resources.

Found Baseline Model

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



Golden Ratio and Model Scaling

Golden Ratio of α, β, γ by grid search

- Assuming twice more resources available,i.e., Φ = 1
- The best values are α=1.2, β=1.1, γ=1.15

Scaling the Baseline Model

- Choosing Φ considering available resources
- Scale the base model by

$$w = \alpha^{\phi},\ \ d = \beta^{\phi}, r = \gamma^{\phi}$$

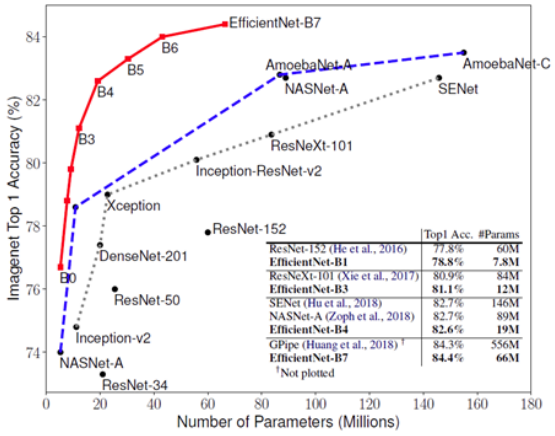
- They chose 7 different values for Φ, i.e., generated 7 different networks by scaling.

Performance

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).

EfficientNe-BΦ



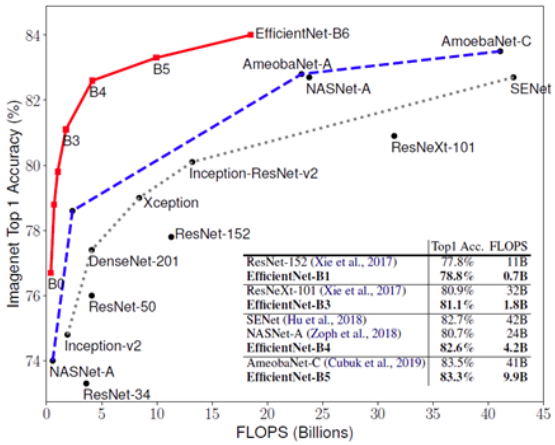


Table 4. **Inference Latency Comparison** – Latency is measured with batch size 1 on a single core of Intel Xeon CPU E5-2690.

	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**

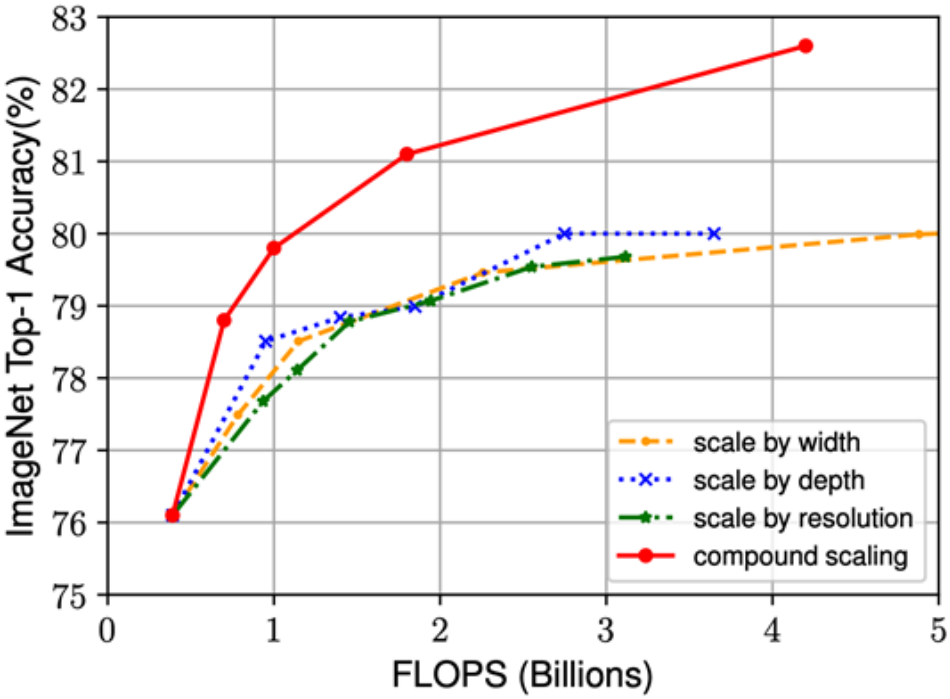


Figure 8. **Scaling Up EfficientNet-B0 with Different Methods.**

scaling Comparison

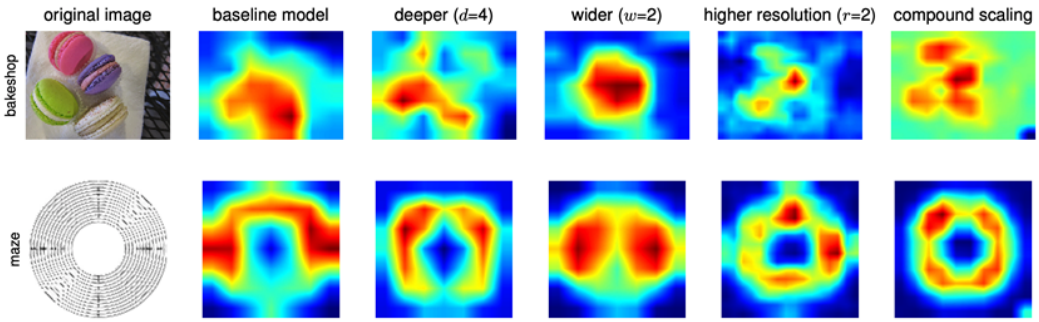


Figure 7. **Class Activation Map (CAM) (Zhou et al., 2016) for Different Models in Table 7 -** Our compound scaling method allows the scaled model (last column) to focus on more relevant regions with more object details. Model details are in Table 7.

They have applied the same approach to scale MobileNet and ResNet

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%**