EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks

Mingxing Tan & Quoc V. Le, Google Brain team (Google AI Research)
ICML 2019

mPP ML Journal Club
12/11/20

Mary Touranakou
ConvNets Model Scaling

• ConvNets acquire higher accuracy with bigger networks (scaling up), but that results to having more parameters and hardware overload (trade-off between accuracy & efficiency).

• There are different ways of performing scaling; a task that has not systematically been studied, and one that requires tedious manual tuning (which is expensive).
  o Depth (d): adding more layers for more complex features (ResNet-18 vs ResNet-200)
  o Width (w): adding more channels for fine-grained features (WideResNet, MobileNets)
  o Resolution (r): increasing input image size (more pixels) for more information

• The scaling process is usually based on human expertise/experience or on the application of good practices.
ConvNets Model Scaling

• Even if two or three dimensions are scaled that is done arbitrarily and/or manually → the design space is not extensively explored which seems to yield sub-optimal solutions in terms of accuracy and efficiency.

• Intuitively, e.g. it makes sense to scale the depth and width if the input image is bigger (scale all dimensions), but the relationship among the three dimensions has not been quantified before or systematically studied.

• Better accuracy needs to be gained though acquiring more efficiency.

• The authors wish to systematically and empirically study model efficiency for super large ConvNets that surpass SOTA accuracy. → they propose a novel way of model scaling.
Research Question

The paper focuses on redefining the process of model *scaling* and performing scaling in a systematic, empirical and quantifiable way.

• **Main Research Question:** Is there a principled method to scale up a ConvNet to achieve better accuracy and efficiency?

• Results show that it is critical to coordinate and balance all three dimensions, and that this can be done by uniformly scaling each one of them with constant ratio resolution to achieve better performance (in terms of accuracy AND efficiency). The authors call this method **compound scaling**.
Observation 1

• The optimal $d$, $w$, and $r$ are not independent, and their values change as resource constraints change. Traditionally single-dimension scaling is tested.

• Scaling up only one dimension can achieve better accuracy, but that accuracy saturates after ~80% for single-dimension scaling which highlights
Observation 2

- Scaling up a combination of dimensions, and actually balancing all three of them instead can achieve higher accuracy which might pave the path to new, optimal solutions for CNNs.

Width ($w$) comparison with different $d$ and $r$ values (under same FLOPS cost). Better accuracy is achieved when scaling all three dimensions.
Compound Scaling

- Compound scaling: uniformly scales network depth, width and resolution with a set of fixed scaling coefficients.
- For $2^\phi$ times more computational resources, increase network:

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

where $\alpha$, $\beta$, $\gamma$, constant coefficients (specify how to assign these extra resources) determined by a grid search in a baseline model, and $\phi$ is a user-specified coefficient that controls how many more resources are available for model scaling.
Compound Scaling

• FLOPS (how many operations are needed to run the network) are proportional to $d$, $w^2$, and $r^2$. Doubling network depth would double FLOPS but doubling the network width or resolution would increase FLOPS by four times.

• Scaling a CNN with (3) would approximately increase total FLOPS by $(\alpha \cdot \beta^2 \cdot \gamma^2)^\phi$. Authors constrain $\alpha \cdot \beta^2 \cdot \gamma^2 \approx 2$ such that for any new $\varphi$, the total FLOPS will approximately increase by $2^\phi$. 
Alternative ways of model scaling. Single-dimension scaling vs compound scaling.
Problem Formulation

• The maximization of model’s accuracy for any given resource constraints is formulated as an optimization problem:

\[
\text{max}_{d,w,r} \quad \text{Accuracy}(\mathcal{N}(d, w, r))
\]

\[
s.t. \quad \mathcal{N}(d, w, r) = \bigotimes_{i=1...s} \mathcal{F}_i^{d \cdot L_i}(X_{(r \cdot H_i, r \cdot W_i, w \cdot C_i)})
\]

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

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

(2)

• Expanding network length \((L_i)\), width \((C_i)\), and resolution \((H_i, W_i)\) with \(d\), \(w\), and \(r\) coefficients respectively without changing the layer architecture \(F_i\) predefined in the baseline network.
Compound Scaling: Evaluation on Existing ConvNets

Table 3. Scaling Up MobileNets 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%      |

Alternative ways of scaling existing ConvNets and the effect on Top-1 accuracy along with the use of FLOPS per scaled model. Compound scaling gives better accuracy than single-dimension scaling in all cases. However, we do see that existing models get a slight increase in the use of FLOPS with the higher accuracy they acquire.
NAS & EfficientNets

• The effectiveness of model scaling depends on the baseline network too (critical to start from a good baseline model).

• The paper addresses the technique of network architecture search (NAS) to develop a new baseline model (**EfficientNet**), and scale it up to get a family of models called EfficientNets. Experiments show that EfficientNets can run faster, with fewer parameters, and achieve better accuracy.

• NAS becomes popular in designing mobile-size ConvNets and optimizes both accuracy and FLOPS (efficiency) by automating the lookup of highly performant network architectures (systematically exploring the network architecture space to approach an optimal solution).
EfficientNet & Compound Scaling

• Step 1: Search on the small baseline network (EfficientNet-B0); fixing $\varphi = 1$, perform grid search and find the best values for the coefficients $\alpha, \beta, \gamma$ under the predefined constraint (3). Optimal values are $\alpha = 1.2, \beta = 1.1, \gamma = 1.15$.

• Step 2: keeping $\alpha, \beta, \gamma$ fixed (same scaling coefficients used in all family models) and scaling up the baseline network for different values of $\varphi$. 
EfficientNet & Compound Scaling

Scaling up EfficientNet-B0 with different methods. More accuracy at the cost of more FLOPS with single-dimension scaling. Compound scaling improves further accuracy up to 2.5%.
EfficientNet & Compound Scaling

Scaling up EfficientNet-B0 with different methods – Visual results. We can see that increase in depth results to richer/more complex features, increase in width to a more fine-grained representation e.g. of a macaron, while the increase in resolution captures better accuracy. The compound scaling method performs better in both cases.
Compound Scaling: Evaluation on EfficientNets

EfficientNet reduces by an order of magnitude the FLOPS used → up to 16x FLOPS reduction.

| Model                        | Top-1 Acc. | Top-5 Acc. | #Params | Ratio-to-EfficientNet | #FLOPs  | Ratio-to-EfficientNet |
|------------------------------|------------|------------|---------|------------------------|---------|-----------------------|
| EfficientNet-B0              | 77.1%      | 93.3%      | 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              | 79.1%      | 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 reduces by an order of magnitude the model parameters → up to 8.4x parameter reduction.
Compound Scaling: Evaluation on EfficientNets

Model size vs ImageNet accuracy. EfficientNet significantly outperforms other ConvNets.

EfficientNet outperforms GPipe (new SOTA) being 8.4x smaller & 6.1x faster (37B FLOPS).

EfficientNet using 18x fewer FLOPS outperforms ResNetXt-101 for similar accuracy.
## Compound Scaling: Evaluation of EfficientNets on Transfer Learning Datasets

The scaled EfficientNets achieve new SOTA in 5 out of 8 datasets **with an order of magnitude fewer parameters.**

| Model          | 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) | 1Gpipe | 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) | 1Gpipe | 94.8% - | EfficientNet-B7 | 94.7% - |
| Flowers        | Inception-v4 | 98.5% | 41M | EfficientNet-B5 | 98.5% | 28M (1.5x) | 1DAT | 94.8% | - | EfficientNet-B7 | 94.7% - |
| FGVC Aircraft  | Inception-v4 | 90.9% | 41M | EfficientNet-B3 | 90.7% | 10M (4.1x) | DAT | 97.7% | - | EfficientNet-B7 | 98.8% - |
| Oxford-IIIT Pets | ResNet-152 | 94.5% | 58M | EfficientNet-B4 | 94.8% | 17M (5.6x) | DAT | 92.9% | - | EfficientNet-B7 | 92.9% - |
| Food-101       | Inception-v4 | 90.8% | 41M | EfficientNet-B4 | 91.5% | 17M (2.4x) | GPipe | 95.9% | 556M | EfficientNet-B6 | 95.4% | 41M (14x) |
|                | Geo-Mean | (4.7x) | | | | (9.6x) | |

1. GPipe (Huang et al., 2018) trains giant models with specialized pipeline parallelism library.
2. DAT denotes domain adaptive transfer learning (Ngiam et al., 2018). Here we only compare ImageNet-based transfer learning results.
3. 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).
Recap

• Rethinking CNN model scaling → a systematic, quantifiable approach.
• Paper’s approach: balancing all three dimensions can be done by scaling each one with constant ratio → compound scaling. Compound scaling uniformly scales depth, width and resolution with a set of fixed scaling coefficients.
• Traditional approaches use scaling methods to improve accuracy at the cost of extra FLOPS, while compound scaling can achieve state-of-the-art accuracy while maximizing efficiency at the same time (by minimizing the use of hardware resources used). Not necessarily trading accuracy over efficiency!
• Compound scaling & EfficientNet work well with transfer learning datasets and give state-of-the-art accuracy in 5 out of 8 datasets while using 9.6x fewer parameters.
• EfficientNets seem to consistently achieve better accuracy within an order of magnitude fewer parameters than existing models use.
Potential interest for mPP

• We are using and looking into CNN models.
• We have either an enormous amount of data to deal with or resource constraints (performance needs to be improved under constrained budget)
• Manual tuning is tedious, time-consuming, and cannot guarantee that better results will yield within our time-constraints.
• We can make sure that before abandoning a model that is not working well enough, we have optimized its dimensions in the most optimal way to reach and uncover its full potential.
• A similar concept could be applied to other types of models potentially (?).
• Your ideas/opinion?
Links

• https://arxiv.org/abs/1905.11946
• https://github.com/tensorflow/tpu/tree/master/models/official/efficientnet
• https://towardsdatascience.com/neural-architecture-search-nas-the-future-of-deep-learning-c99356351136
Backup
## Baseline Network EfficientNet

| Stage $i$ | Operator          | Resolution  | #Channels $\hat{C}_i$ | #Layers $\hat{L}_i$ |
|-----------|-------------------|-------------|-----------------------|---------------------|
| 1         | Conv3x3           | $224 \times 224$ | 32                    | 1                   |
| 2         | MBConv1, k3x3     | $112 \times 112$ | 16                    | 1                   |
| 3         | MBConv6, k3x3     | $112 \times 112$ | 24                    | 2                   |
| 4         | MBConv6, k5x5     | $56 \times 56$   | 40                    | 2                   |
| 5         | MBConv6, k3x3     | $28 \times 28$   | 80                    | 3                   |
| 6         | MBConv6, k5x5     | $14 \times 14$   | 112                   | 3                   |
| 7         | MBConv6, k5x5     | $14 \times 14$   | 192                   | 4                   |
| 8         | MBConv6, k3x3     | $7 \times 7$     | 320                   | 1                   |
| 9         | Conv1x1 & Pooling & FC | $7 \times 7$     | 1280                  | 1                   |
## Compound Scaling: Evaluation on EfficientNets

| Model                  | Top-1 Acc. | Top-5 Acc. | #Params | Ratio-to-EfficientNet | #FLOPs | Ratio-to-EfficientNet |
|------------------------|------------|------------|---------|-----------------------|--------|-----------------------|
| EfficientNet-B0        | 77.1%      | 93.3%      | 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        | 79.1%      | 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        | 80.1%      | 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        | 81.6%      | 95.7%      | 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        | 82.9%      | 96.4%      | 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        | 83.6%      | 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%      | 96.8%      | 43M     | 1x                    | 19B    | 1x                    |
| EfficientNet-B7        | 84.3%      | 97.0%      | 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).
Compound Scaling: Evaluation on EfficientNets

• These gain in overall performance regards to
  • Better architecture (NAS)
  • Better scaling (compound scaling)
  • Better training settings customized for EfficientNet (*unclear what does that mean*)
Thank you!