Data Science 2

Convolution Neural Networks II

Ondřej Týbl

Charles University, Prague

Outline

1. Architectures

2. ResNet

3. EfficientNet

Architectures

Architectures

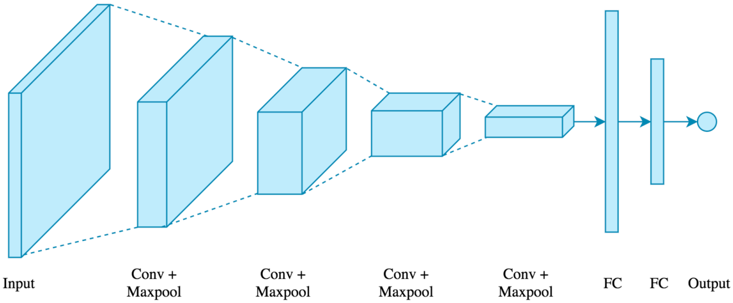


Figure 1: Basic convolution neural network architecture.

We repeat blocks of a) convolution, b) non-linear activation and optionally c) pooling. Number of channels is increasing – more global information uses higher dimensions. Finally, image is flattened and fully connected (FC) layers create an output vector.

ResNet

ResNet

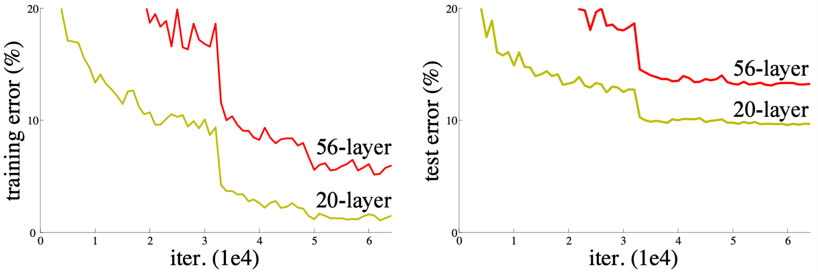


Figure 2: Train and test error on CIFAR 10 [4] for simple convolution neural networks with 20 and 56 convolution layers.

For simple networks such as from Fig. 1 we have

- test error gets worse with more layers – overfitting might explain,
- also train error gets worse with more layers – overfitting can not explain as the larger network contains the smaller one as a sub-network

ResNet

In [2] the problem from Fig. 2 is called *degradation* and the authors relate the problem to the *optimization difficulty*.

However, it is desirable to create deeper network for the purpose of increasing their *capacity*.

Idea: If the layers of the deeper network are difficult to be trained to an identity mapping we can make the layers to be a *difference from an identity mapping instead*.

ResNet

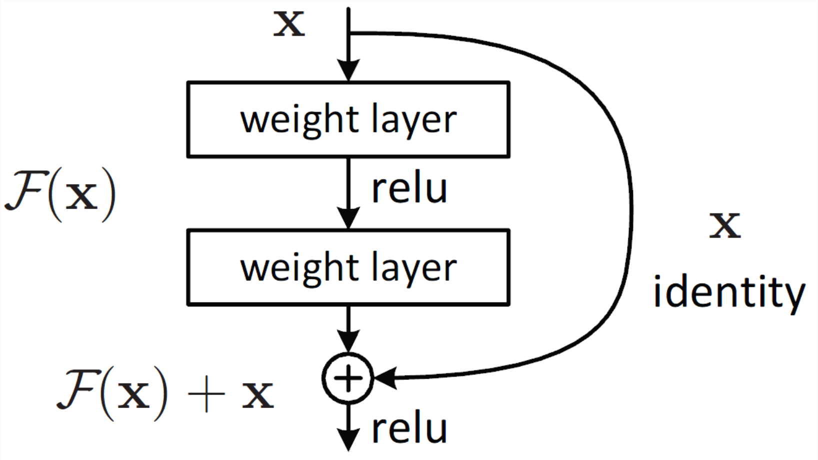


Figure 3: A residual building block from [2]. A sequence of convolution layers with activations and a residual connection which is added to the output of the last convolution. The last activation is applied *after* adding the residuals as suggested empirically.

ResNet

Empirically, it has been observed that residual block helps to avoid a *vanishing gradient problem* as the block derivative with respect to the model parameters is close to an identity instead of zero.

Derivatives close to identity are useful as the chain rule for derivatives computation is a long sequence of multiplications for deep networks.

ResNet architecture was proposed with two specific residual connection blocks with 3×3 convolutions and strides 1 or 2.

ResNet

Overall, two types of blocks in ResNet:

- if stride $S = 1$ then no spatial reduction is present and the residual block is just a composition of two convolution layers with addition while number of channels is kept the same,
- if stride $S = 2$ then the first convolution reduces the spatial dimension and increases the number of channels, we can not therefore simply add the identity and a projection via 1×1 convolution to reduce spatial dimension and increase number of channels is applied instead of the identity

ResNet

Definition (Residual Connection in ResNet without Stride)

Assume a number of input channels $c \in \mathbb{N}$ and tensors

$$\begin{aligned} K_1, K_2 :& \{-1, \dots 1\}^2 \times \{0, \dots c-1\}^2 \mapsto \mathbb{R} \\ b_1, b_2 :& \{0, \dots, c-1\} \mapsto \mathbb{R}. \end{aligned}$$

Denote *BN* the batch normalization. The mapping tensor $I : \{0, \dots, h-1\} \times \{0, \dots, w-1\} \times \{0, \dots, c-1\} \mapsto \mathbb{R}$ to a tensor of the same dimension defined as

$$I \mapsto ReLU\left(BN\left(\mathcal{F}\left(I\right) + I\right)\right),$$

where $\mathcal{F}$ is a composition of convolution layers with kernels K_1, K_2 and biases b_1, b_2 is called a *residual connection in ResNet without stride.*

In this case the convolution layers do not change a tensor dimension and the addition $\mathcal{F}\left(I\right) + I$ is well defined.

ResNet

Definition (Residual Connection in ResNet with Stride)

Assume a number of input channels $c \in \mathbb{N}$ and tensors

$$\begin{aligned} K_1 :& \{-1, \ldots 1\}^2 \times \{0, \ldots, c-1\} \times \{0, \ldots, 2c-1\} \mapsto \mathbb{R} \\ K_2 :& \{-1, \ldots 1\}^2 \times \{0, \ldots, 2c-1\} \times \{0, \ldots, 2c-1\} \mapsto \mathbb{R} \\ P :& \{0\}^2 \times \{0, \ldots, c-1\} \times \{0, \ldots, 2c-1\} \mapsto \mathbb{R} \\ b_1, b_2 :& \{0, \ldots, 2c-1\} \mapsto \mathbb{R}. \end{aligned}$$

Denote BN the batch normalization. The mapping tensor $I : \{0, \ldots, h-1\} \times \{0, \ldots, w-1\} \times \{0, \ldots, c-1\} \mapsto \mathbb{R}$ to a tensor in $\{0, \ldots, h/2-1\} \times \{0, \ldots, w/2-1\} \times \{0, \ldots, 2c-1\} \mapsto \mathbb{R}$ as

$$I \mapsto ReLU\left(BN\left(\mathcal{F}\left(I\right) + P * I\right)\right),$$

where $\mathcal{F}$ is a composition of a convolution layer with kernel K_1, bias b_1 and stride 2 and a convolution layer with kernel K_2, bias b_2 and stride 1 is called a *residual connection in ResNet with stride.*

ResNet

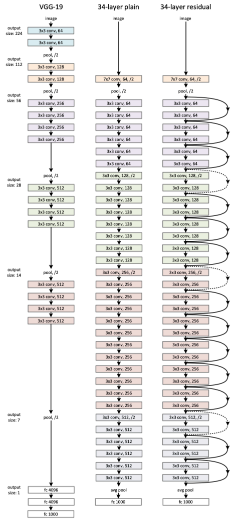


Figure 3. Example network architectures for ImageNet. **Left**: the VGG-19 model [41] (19.6 billion FLOPs) as a reference. **Middle**: a plain network with 34 parameter layers (3.6 billion FLOPs). **Right**: a residual network with 34 parameter layers (3.6 billion FLOPs). The dotted shortcuts increase dimensions. **Table 1** shows more details and other variants.

ResNet

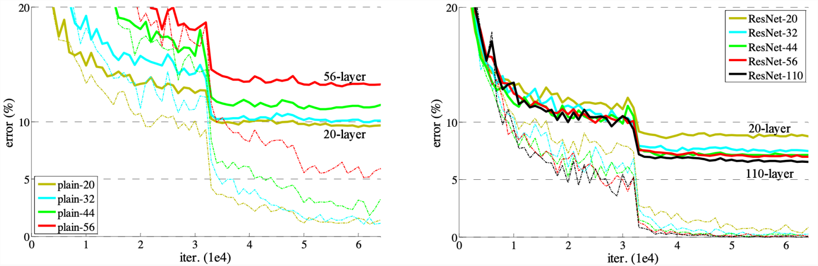
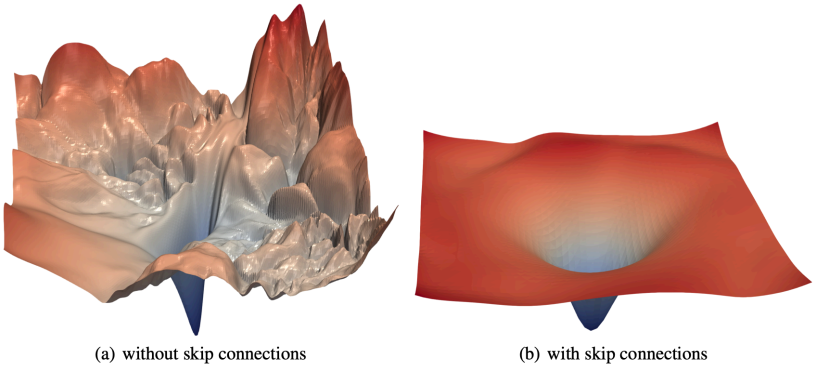


Figure 5: Train error on CIFAR 10 [4] compared for simple convolution networks (left) and networks with residual blocks for 20, 32, 44, 56 and 110 layers.

Adding residual (also called skip) connections causes the networks to scale the training error as expected – the higher the capacity the lower the error. If proper regularization is added also the test error decreases with increasing capacity.

ResNet

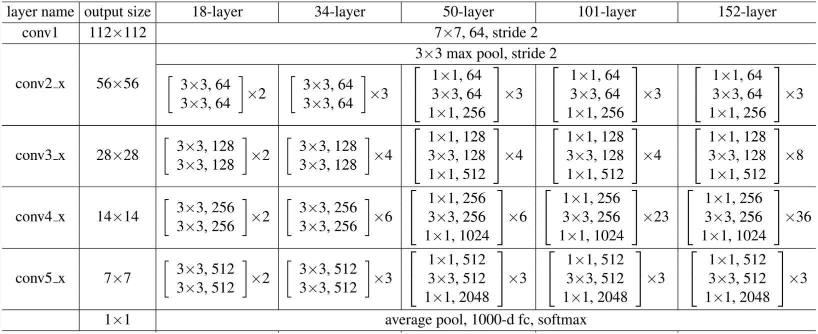


(a) without skip connections

(b) with skip connections

Figure 6: Train loss on CIFAR 10 [4] around the estimated minimizer as a function of two (randomly chosen) directions for ResNet56 with (left) and without (right) residual connections.

ResNet



layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		$\begin{bmatrix}3\times3, 64\\3\times3, 64\end{bmatrix}\times2$	$\begin{bmatrix}3\times3, 64\\3\times3, 64\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 64\\3\times3, 64\\1\times1, 256\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 64\\3\times3, 64\\1\times1, 256\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 64\\3\times3, 64\\1\times1, 256\end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix}3\times3, 128\\3\times3, 128\end{bmatrix}\times2$	$\begin{bmatrix}3\times3, 128\\3\times3, 128\end{bmatrix}\times4$	$\begin{bmatrix}1\times1, 128\\3\times3, 128\\1\times1, 512\end{bmatrix}\times4$	$\begin{bmatrix}1\times1, 128\\3\times3, 128\\1\times1, 512\end{bmatrix}\times4$	$\begin{bmatrix}1\times1, 128\\3\times3, 128\\1\times1, 512\end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix}3\times3, 256\\3\times3, 256\end{bmatrix}\times2$	$\begin{bmatrix}3\times3, 256\\3\times3, 256\end{bmatrix}\times6$	$\begin{bmatrix}1\times1, 256\\3\times3, 256\\1\times1, 1024\end{bmatrix}\times6$	$\begin{bmatrix}1\times1, 256\\3\times3, 256\\1\times1, 1024\end{bmatrix}\times23$	$\begin{bmatrix}1\times1, 256\\3\times3, 256\\1\times1, 1024\end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix}3\times3, 512\\3\times3, 512\end{bmatrix}\times2$	$\begin{bmatrix}3\times3, 512\\3\times3, 512\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 512\\3\times3, 512\\1\times1, 2048\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 512\\3\times3, 512\\1\times1, 2048\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 512\\3\times3, 512\\1\times1, 2048\end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax				

Figure 7: ResNet architecture variants. Input image size is 224×224 as in ImageNet [1].

ResNet

	error	no. of parameters
ResNet18	-	11.2M
ResNet34	21.84%	23.1M
ResNet50	21.53%	23.5M
ResNet101	19.87%	42.5M
ResNet152	19.38%	58.2M

Table 1: ResNet variants with number of trainable parameters and test error on ImageNet [1] classification into 1000 classes from [2].

Test error for ResNet18 is missing; results have been improved since the original paper publication with using advanced optimizers and regularizations, see [7]

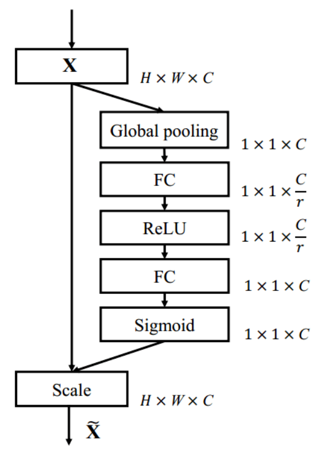
EfficientNet

EfficientNet

EfficientNet architecture enhances the ResNet architecture with residual connections with two key features

- *squeeze-and-excitation block* – channel importance depends on the input
- *separable convolution* – feature to lower the computation complexity

EfficientNet



Squeeze-and-excitation block [3].

- Block that keeps the dimensions unchanged
- Values in $(0, 1)$ are computed per channel (see the sigmoid activation) to calibrate channel importance. The input of the block is then calibrated by multiplication (scale box).
- After global pooling to produce C values only, a simple neural network with one hidden layer is used for the channel importance computation. The hidden layer is squeezed (see the r factor) to keep the parameter number low.

EfficientNet

The separable convolution will now be explained.

EfficientNet

Recall the convolution for edge detection. It is easily verified that we can decompose it

$$\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} * \begin{bmatrix} -1 & 0 & 1 \end{bmatrix} \tag{1}$$

While the representation on the left hand side of (1) has 9 parameters, the one on the right hand side has only 6.

Moreover, the convolution is associative

$$(K_1 * K_2) * K_3 = K_1 * (K_2 * K_3)\,,$$

we can simulate the edge detection convolution by a sequential application of convolutions with a smaller total number of parameters.

EfficientNet

Clearly, not every 3×3 convolution can be decomposed as above.

However, we shall try to construct our neural network by convolutions with such a property with the hope that it will still be general enough to express the data while the number of parameters gets lower.

EfficientNet

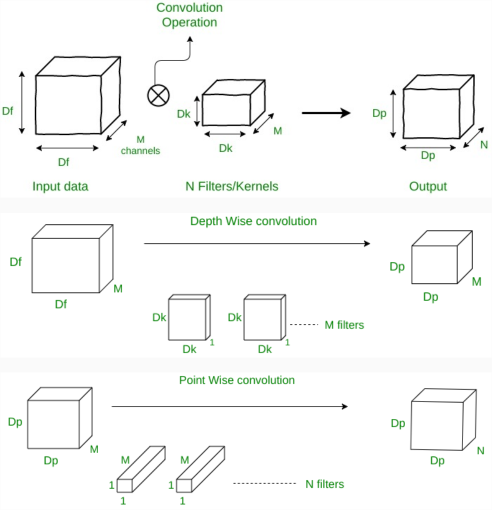


Figure 8: Schema of classical convolution (up) compared to depth wise and point wise convolutions (middle and down).

EfficientNet

Definition (Depth Wise Convolution)

A convolution with kernel[1] $K : \{-(Dk-1)/2, \ldots, (Dk-1)/2\}^2 \times \{0, \ldots, c-1\}^2 \mapsto \mathbb{R}$ with the diagonal structure

$$K_{\cdot,\cdot,c_1,c_2} \equiv 0, \quad c_1 \neq c_2$$

is called a *depth wise convolution*.

The diagonal structure implies that spatial features are combined per channel separately.

The number of multiplications for the image $Df \times Df$:

$$c \times Dk^2 \times Df^2$$

[1] We use square $Dk \times Dk$ convolutions for simplicity.

EfficientNet

Definition (Point-Wise Convolution)

A convolution with 1×1 kernel, i.e. a kernel of the form $K : \{0\}^2 \times \{0, \ldots, c_{input} - 1\} \times \{0, \ldots, c_{output} - 1\} \mapsto \mathbb{R}$, is called *point-wise convolution*.

For each pixel separately, all channels are combined.

The number of multiplications for the image $Dp \times Dp$:

$$c_{input} \times Dp^2 \times c_{output}$$

EfficientNet

Definition (Separable Convolution)

A convolution which is a composition of depth-wise and point-wise convolution is called *separable convolution*.

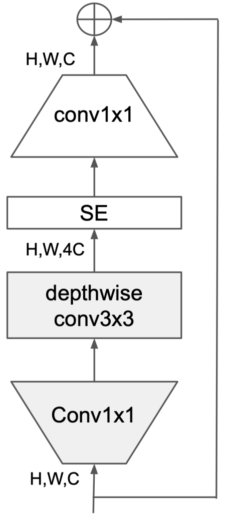
From above, the number of multiplications for separable convolution for $Df \times Df$ image is

$$c_{input} \times Dk^2 \times Df^2 + c_{input} \times Df^2 \times c_{output}$$

When the number of channels c_{input}, c_{output} is dominant to Dk, Df which is the usual case, a general convolution has much more multiplications:

$$c_{input} \times Dk^2 \times Df^2 \times c_{output}$$

EfficientNet

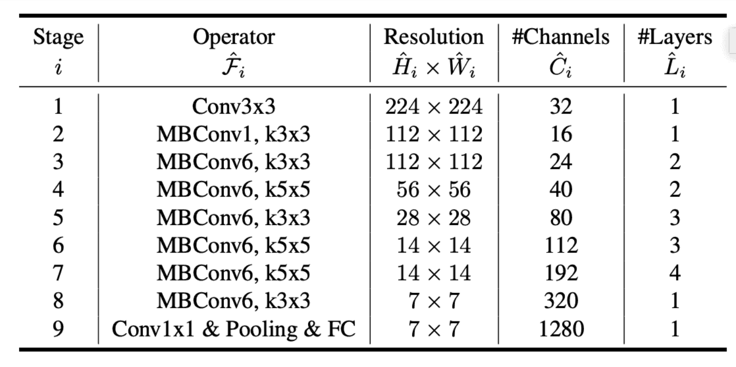


Mobile Inverted Bottleneck (MBConv) [5].

- Separable convolution (the first two boxes) that increases the number of channels with low number of parameters (here with an expansion ratio $r = 4$
- Followed by squeeze-and-excitation block
- Finally, 1×1 convolution is applied to get to the input number of channels
- Residual connection at the points with low number of channels[a].

[a] When the channel count is high, the expansion step dominates, making the residual connection less useful

EfficientNet



Stage	Operator	Resolution	#Channels	#Layers
i	$\hat{\mathcal{F}}_i$	$\hat{H}_i \times \hat{W}_i$	$\hat{C}_i$	$\hat{L}_i$
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

Figure 9: EfficientNet architecture [6]. Each stage i consists of a stacking identical blocks $\hat{L}_i$ times. The input image size is 224×224 as in ImageNet [1].

Here, MBConvS, k$N \times N$ stands for a Mobile Inverted Bottleneck Block with expansion ratio S and kernel size N enhanced with a squeeze and excitation block.

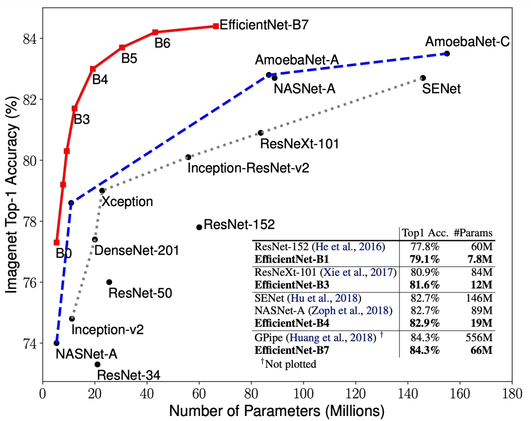
EfficientNet

The EfficientNet architecture does not follow the usual rule when the number of channels is increased by a factor of 2 as the specific parameters have been found via a hyperparameter optimization.

The architecture from Figure 9 is the base network. Authors proposed other size variants via scaling of the network width W(channel count), depth D (number of layers) and input resolution R: to double the computation complexity, one should increase W by a factor 1.1, D by 1.2 and R by 1.15. The desired complexity is obtained due to

$$1.1^2 \times 1.2 \times 1.15^2 \sim 2$$

EfficientNet



	Top1 Acc.	#Params
ResNet-152 (He et al., 2016)	77.8%	60M
EfficientNet-B1	**79.1%**	**7.8M**
ResNeXt-101 (Xie et al., 2017)	80.9%	84M
EfficientNet-B3	**81.6%**	**12M**
SENet (Hu et al., 2018)	82.7%	146M
NASNet-A (Zoph et al., 2018)	82.7%	89M
EfficientNet-B4	**82.9%**	**19M**
GPipe (Huang et al., 2018) †	84.3%	556M
EfficientNet-B7	**84.3%**	**66M**

†Not plotted

Figure 10: EfficientNet and ResNet variants compared on ImageNet [1].

The EfficientNet architecture is consistently better than ResNet and other previous networks for different size variants.

References i

J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei.
Imagenet: A large-scale hierarchical image database.
In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.

K. He, X. Zhang, S. Ren, and J. Sun.
Deep residual learning for image recognition.
In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

J. Hu, L. Shen, and G. Sun.
Squeeze-and-excitation networks.
In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7132–7141, 2018.

References ii

A. Krizhevsky, G. Hinton, et al.
Learning multiple layers of features from tiny images.
2009.

M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen.
Mobilenetv2: Inverted residuals and linear bottlenecks.
In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018.

M. Tan and Q. Le.
Efficientnet: Rethinking model scaling for convolutional neural networks.
In *International conference on machine learning*, pages 6105–6114. PMLR, 2019.

References iii

R. Wightman, H. Touvron, and H. Jégou.
Resnet strikes back: An improved training procedure in timm.
arXiv preprint arXiv:2110.00476, 2021.