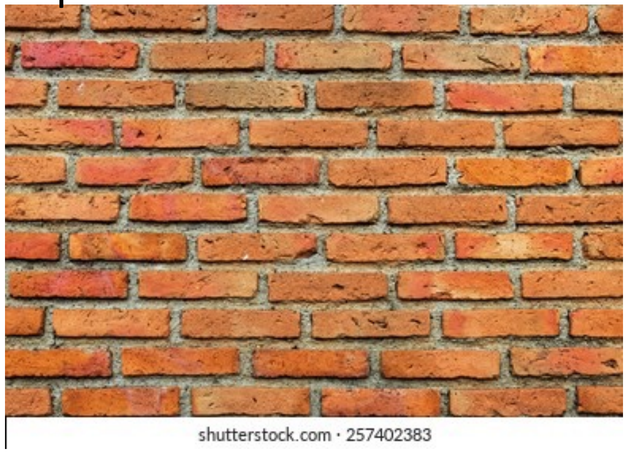
How to lay bricks?

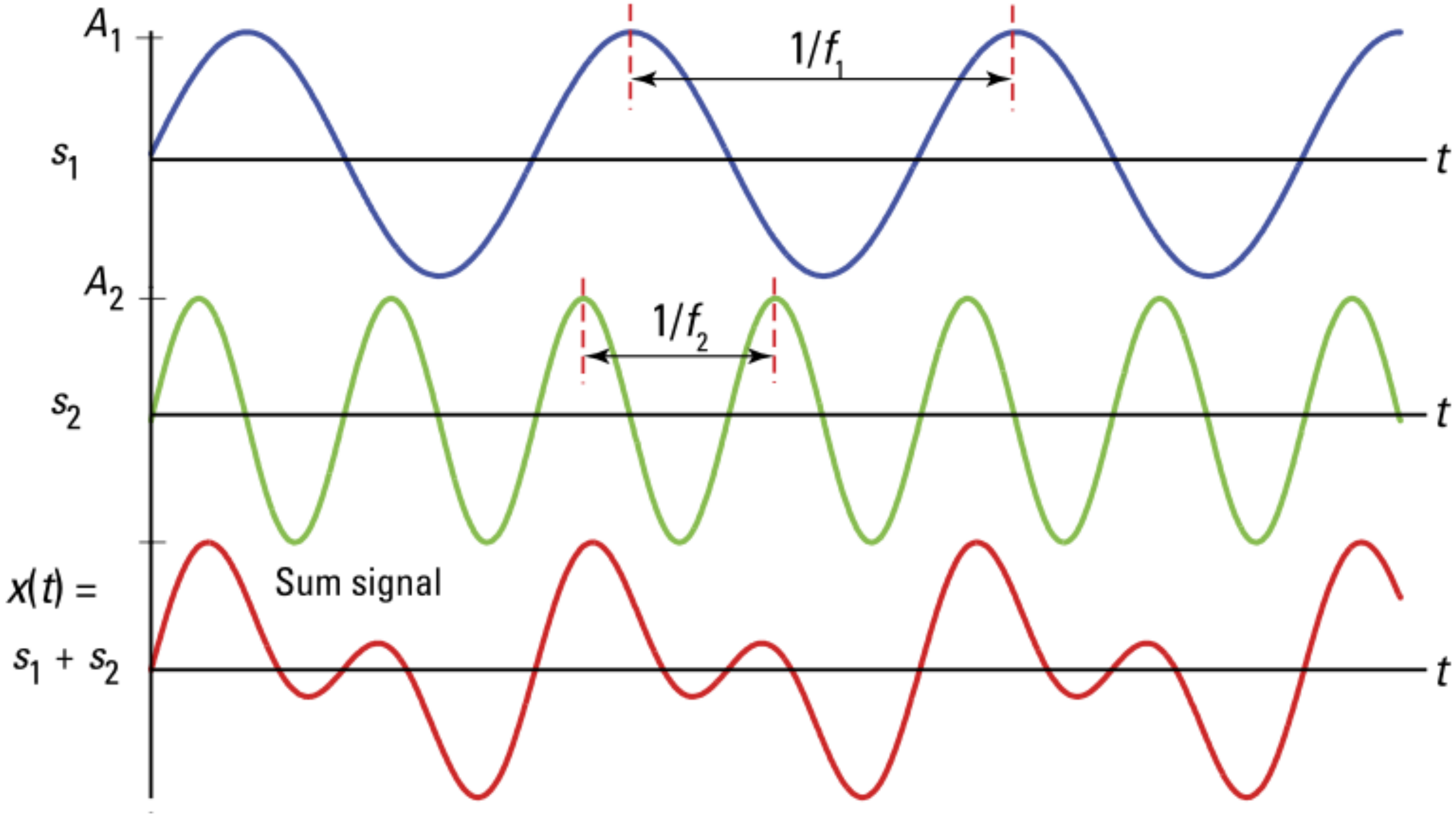
Option A

Option B

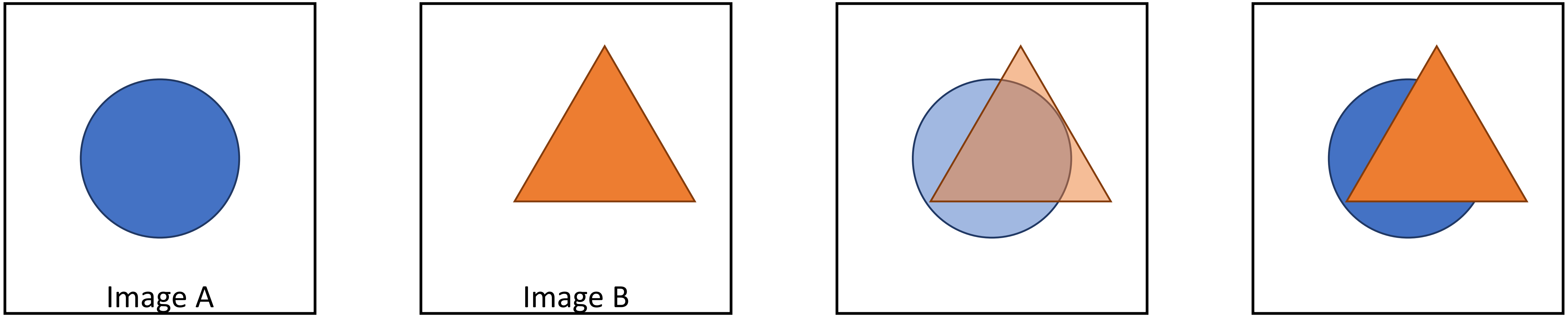


Non-linearities

Consider a simple addition of two sinusoids



Images are highly non-linear!



Linear combination A+B

Actual combination A+B

No non-linearities, what happens?

Without non-linearities

- $h_1 = W_1 x + b_1$
- $o = W_2 h_1 + b_2$

What's the problem here?

- $o = W_2 W_1 x + (W_2 b_1 + b_2)$
- $o = W^* x + b^*$

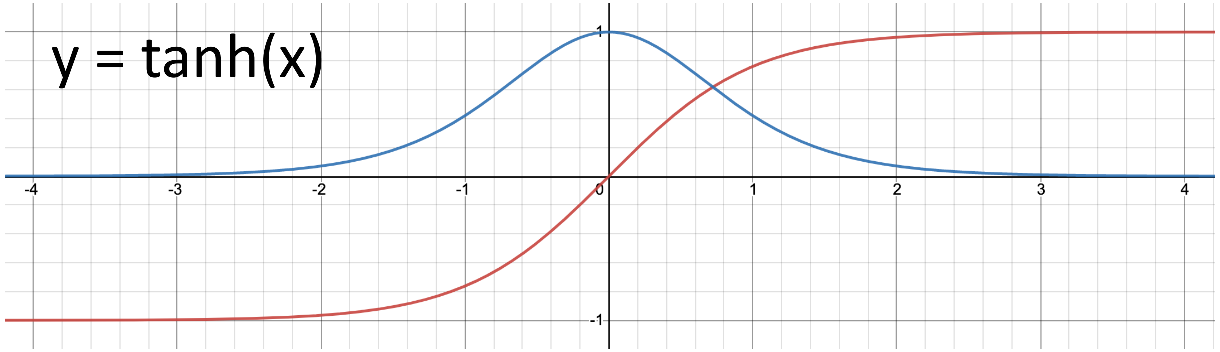
There is only one layer!

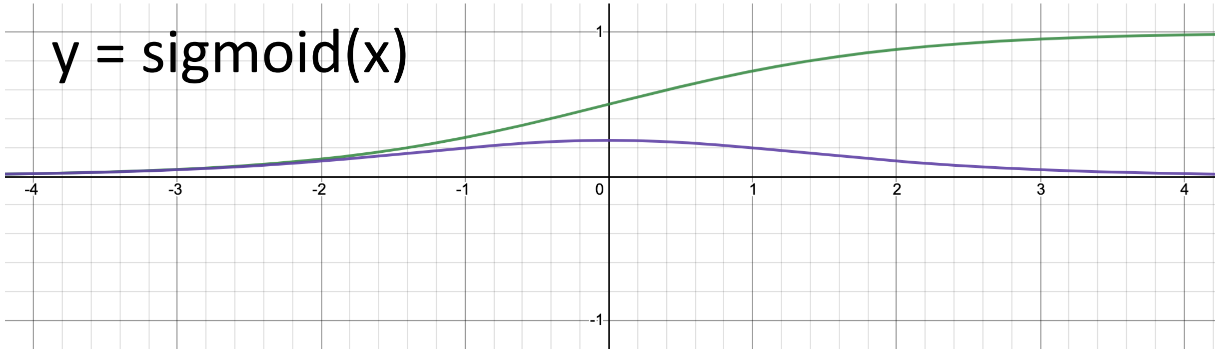
With non-linearities

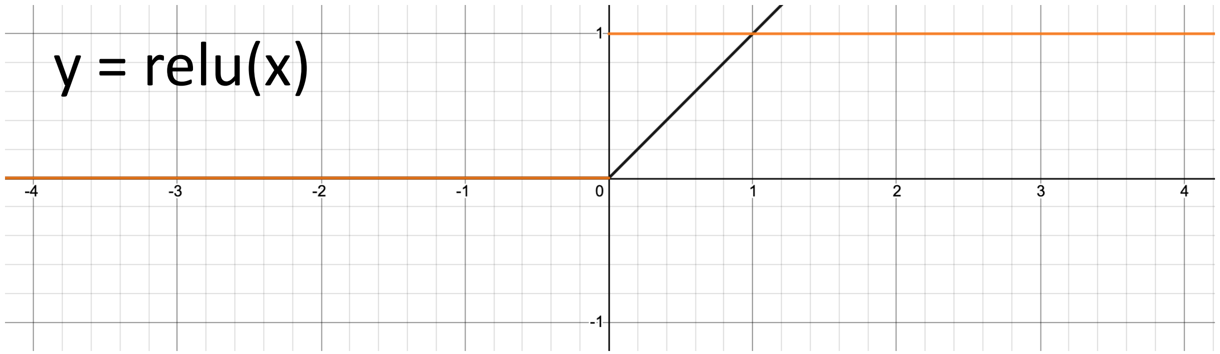
- $h_1 = \phi(W_1 x + b_1)$
- $o = W_2 h_1 + b_2$

There is meaning in having two layers now!

Simple non-linearities are often sufficient



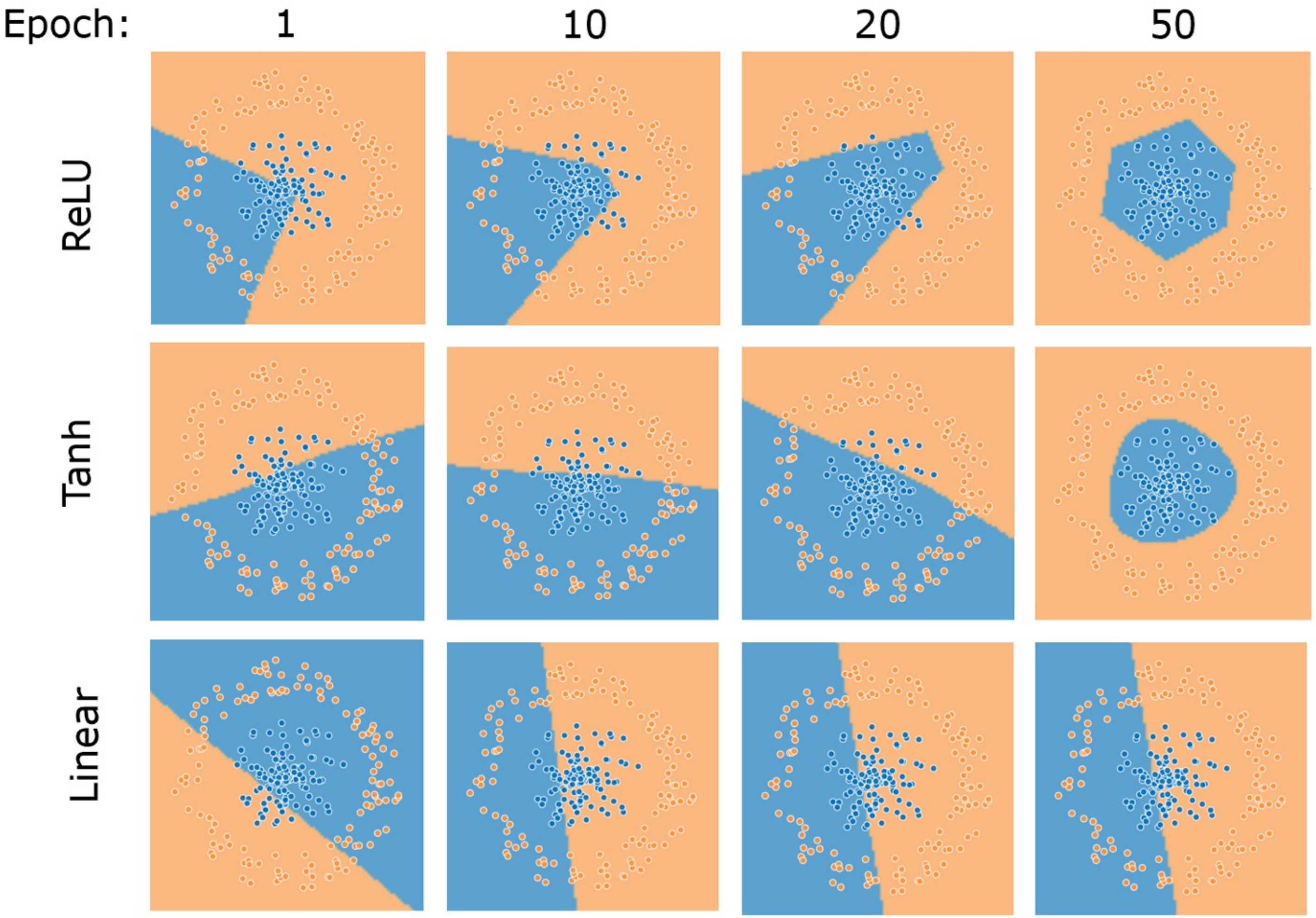




Graphs created using https://www.desmos.com/calculator

Rectified Linear Unit (ReLU): Why is it so popular?

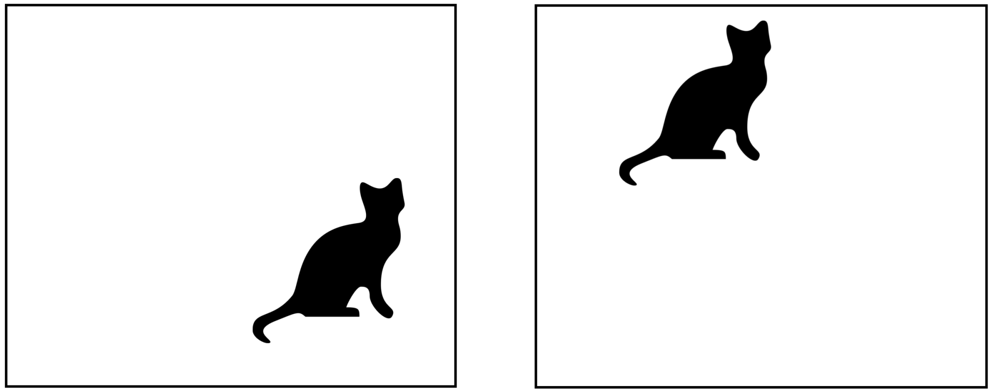
- Very fast
- Gradients
- Piece-wise linear approximation



Invariance

Remember invariance?

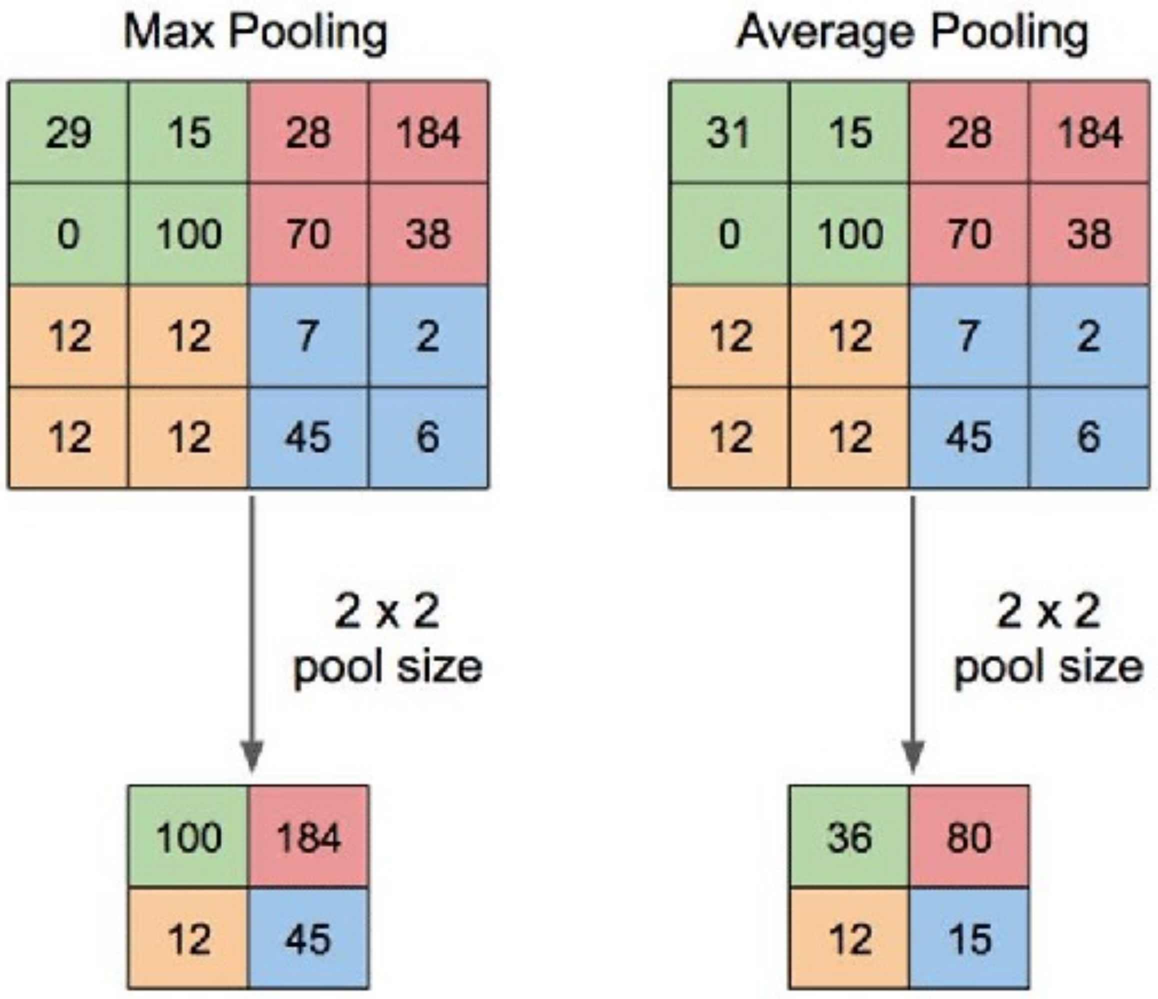
- <u>Quiz:</u> Are convolutions shift invariant or shift equivariant?
- Can we do something to obtain invariance?



- Imagine you have a “cat kernel”, and you get some hits. Then what?

Max vs. Average Pooling

- Is one better than the other?
- When and why?



Pooling by striding

- Input = 32 x 128 x **11 x 11**
- Kernel = 128x128 x 3x3, padding = 1x1, stride = 2x2
- Output = 32 x 128 x **6 x 6**

Normalization

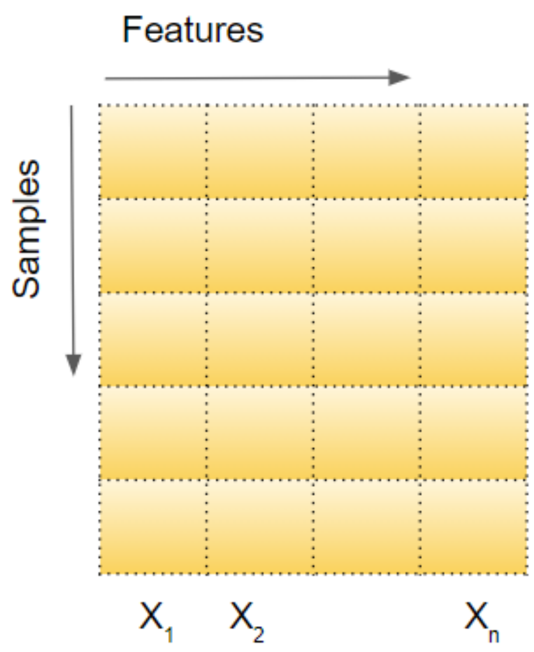
arXiv
https://arxiv.org › cs
8

Batch Normalization: Accelerating Deep Network Training ...

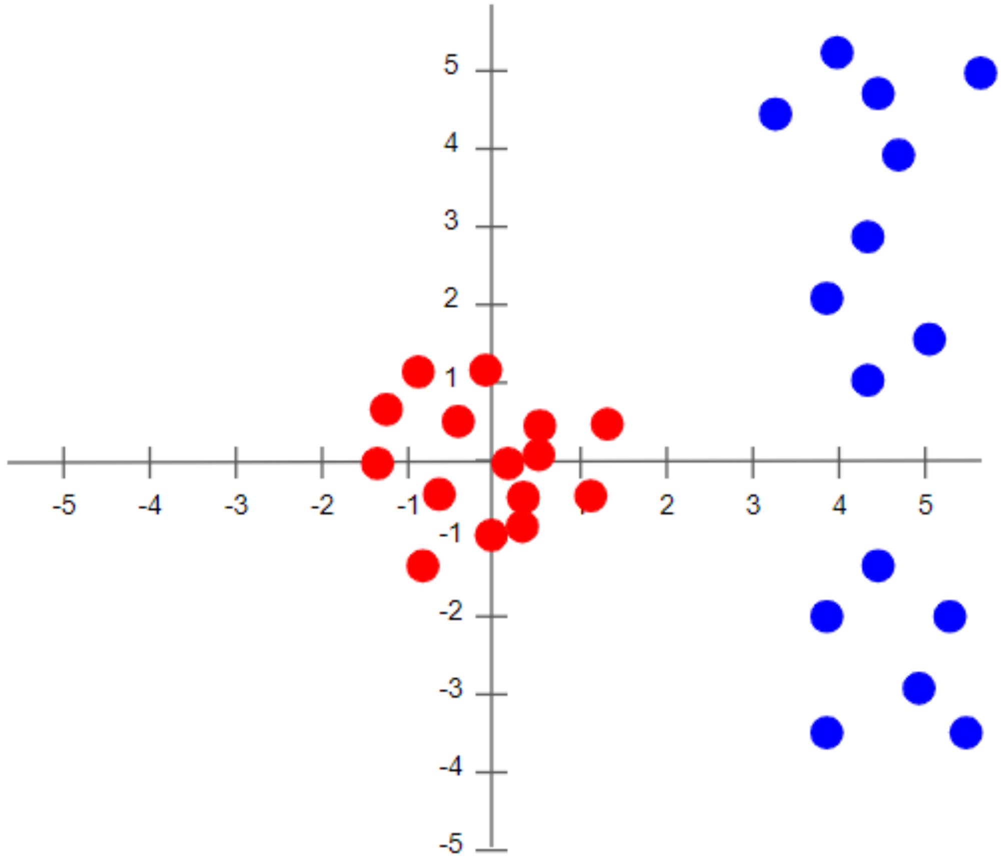
by S Ioffe · 2015 · Cited by 48175 — **Batch Normalization** allows us to use much higher learning rates and be less careful about initialization. It also acts as a regularizer, ...

https://arxiv.org/abs/1502.03167

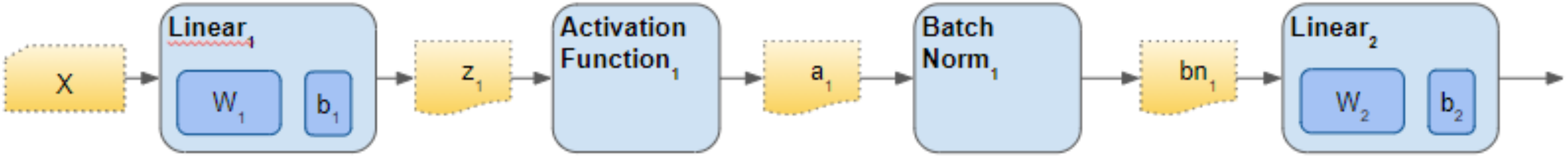
Why normalize input data?

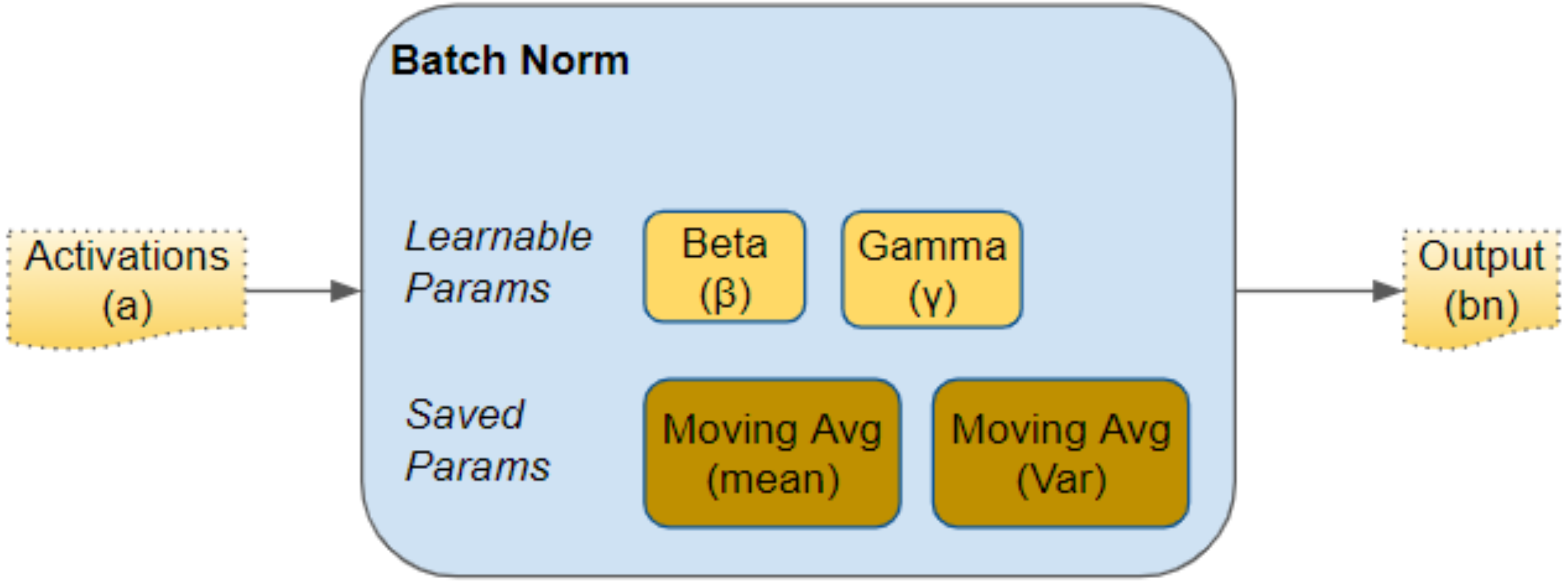


$$X_i = \frac{X_i - Mean_i}{StdDev_i}$$



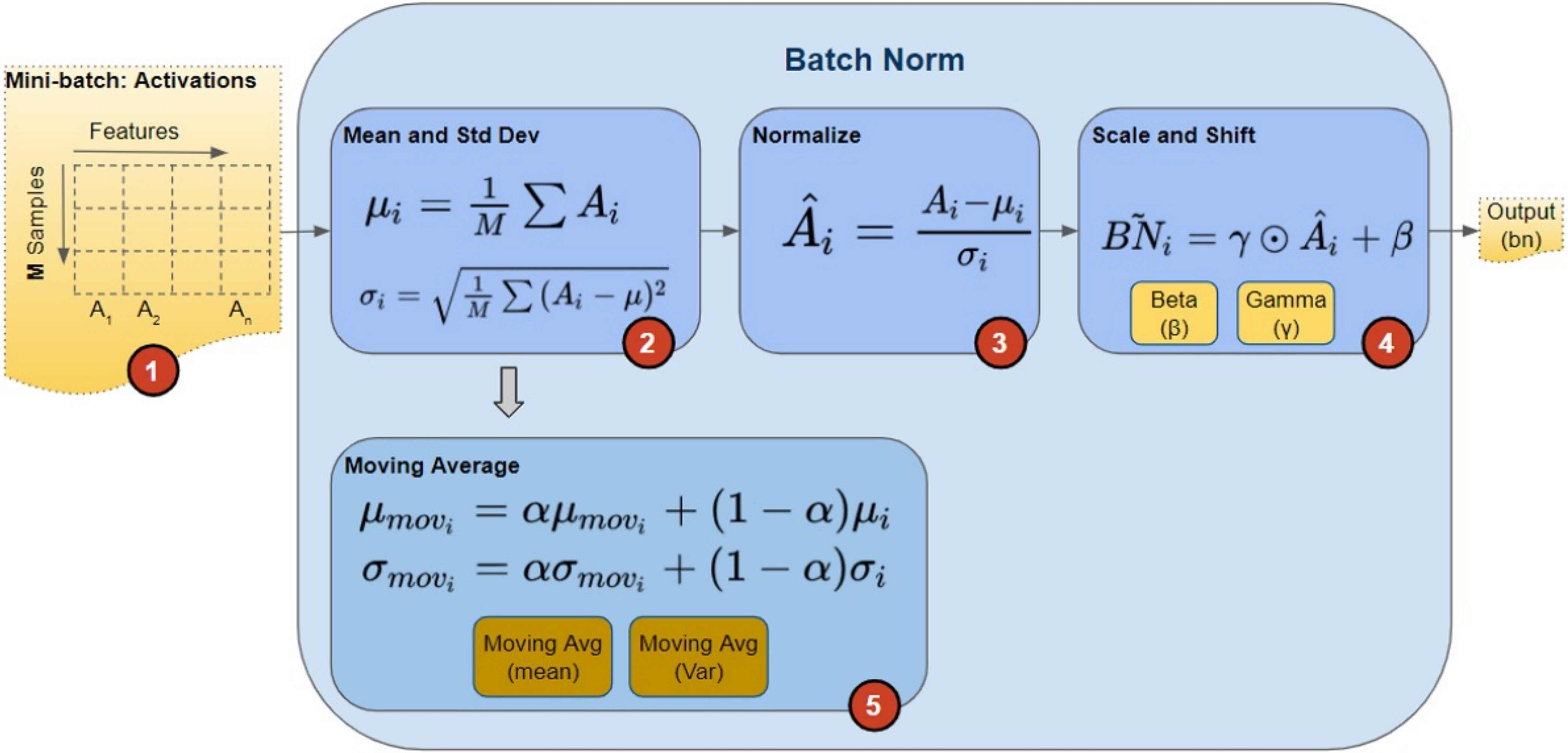
With BatchNorm





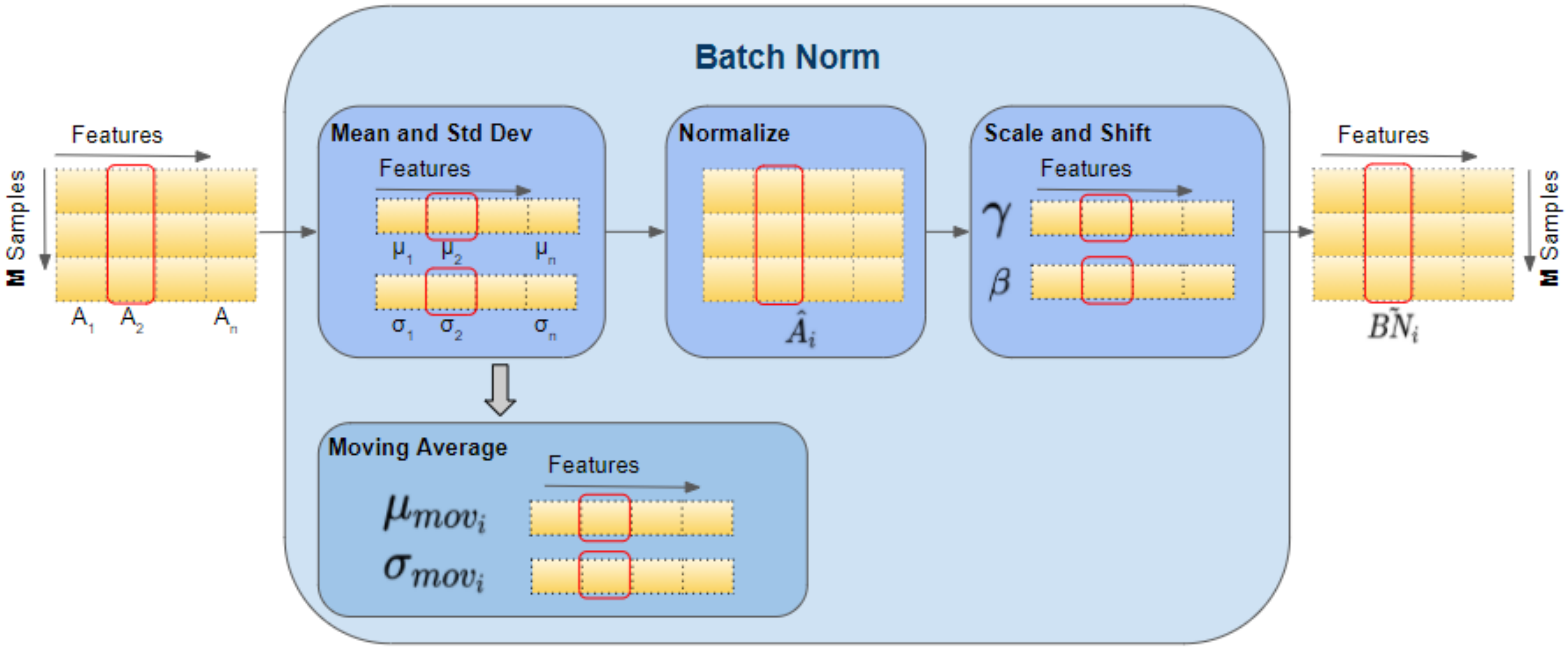
Source: https://towardsdatascience.com/batch-norm-explained-visually-how-it-works-and-why-neural-networks-need-it-b18919692739

Not just normalize, also scale and shift

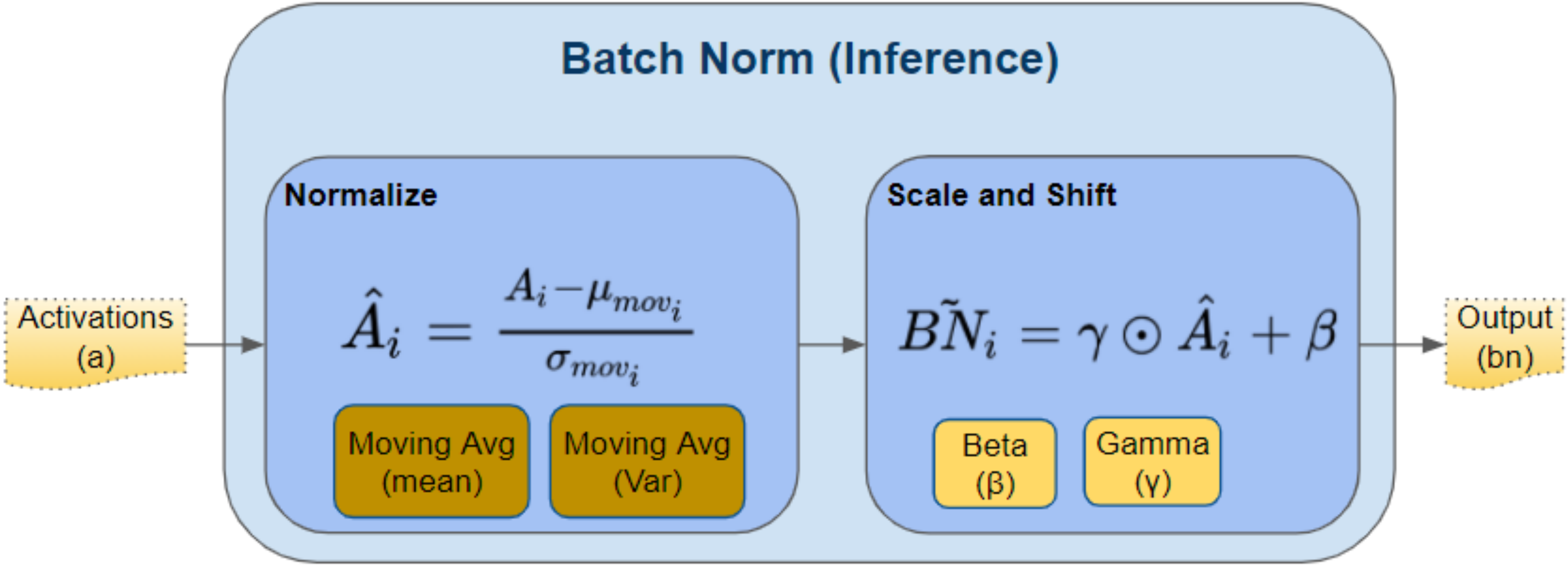


Source: https://towardsdatascience.com/batch-norm-explained-visually-how-it-works-and-why-neural-networks-need-it-b18919692739

How does it work exactly?



BatchNorm during inference



All you need is BatchNorm!?

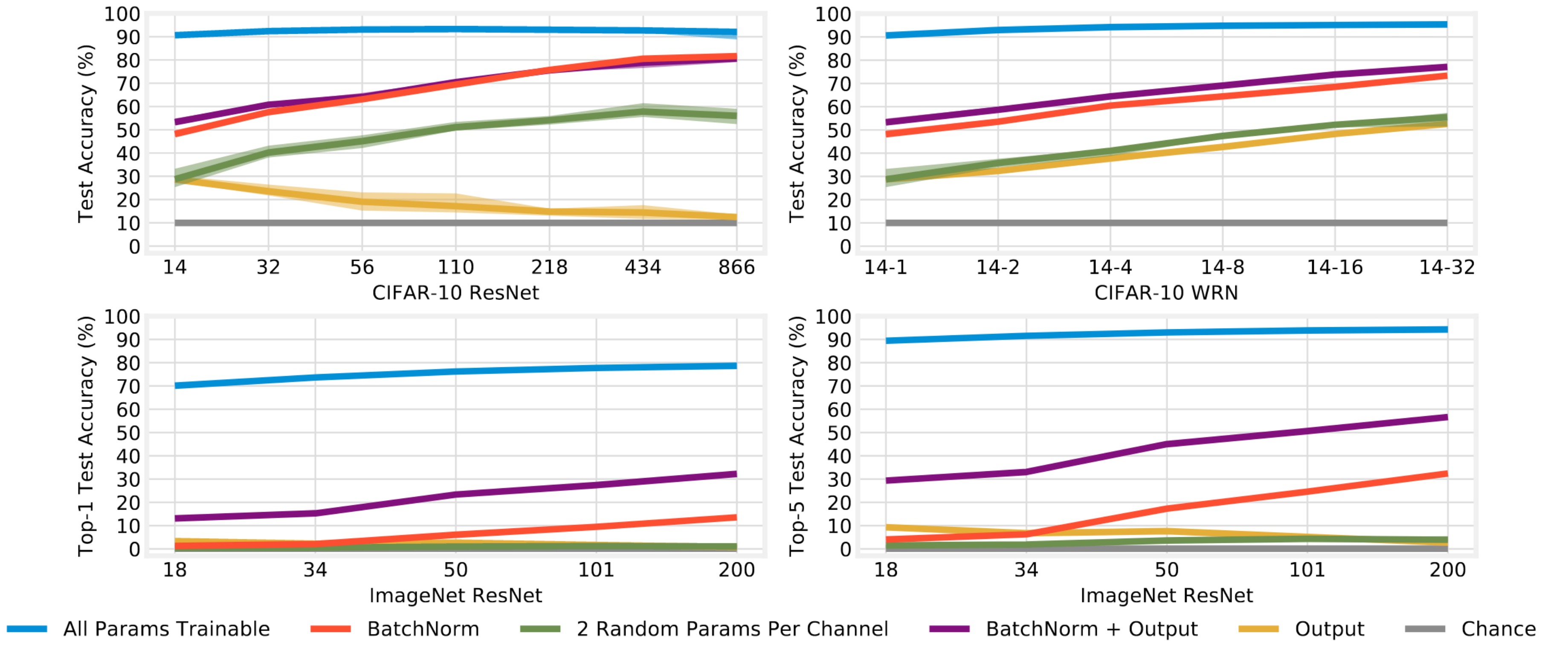


Figure 2: Accuracy of ResNets for CIFAR-10 (top left, deep; top right, wide) and ImageNet (bottom left, top-1 accuracy; bottom right, top-5 accuracy) with different sets of parameters trainable.

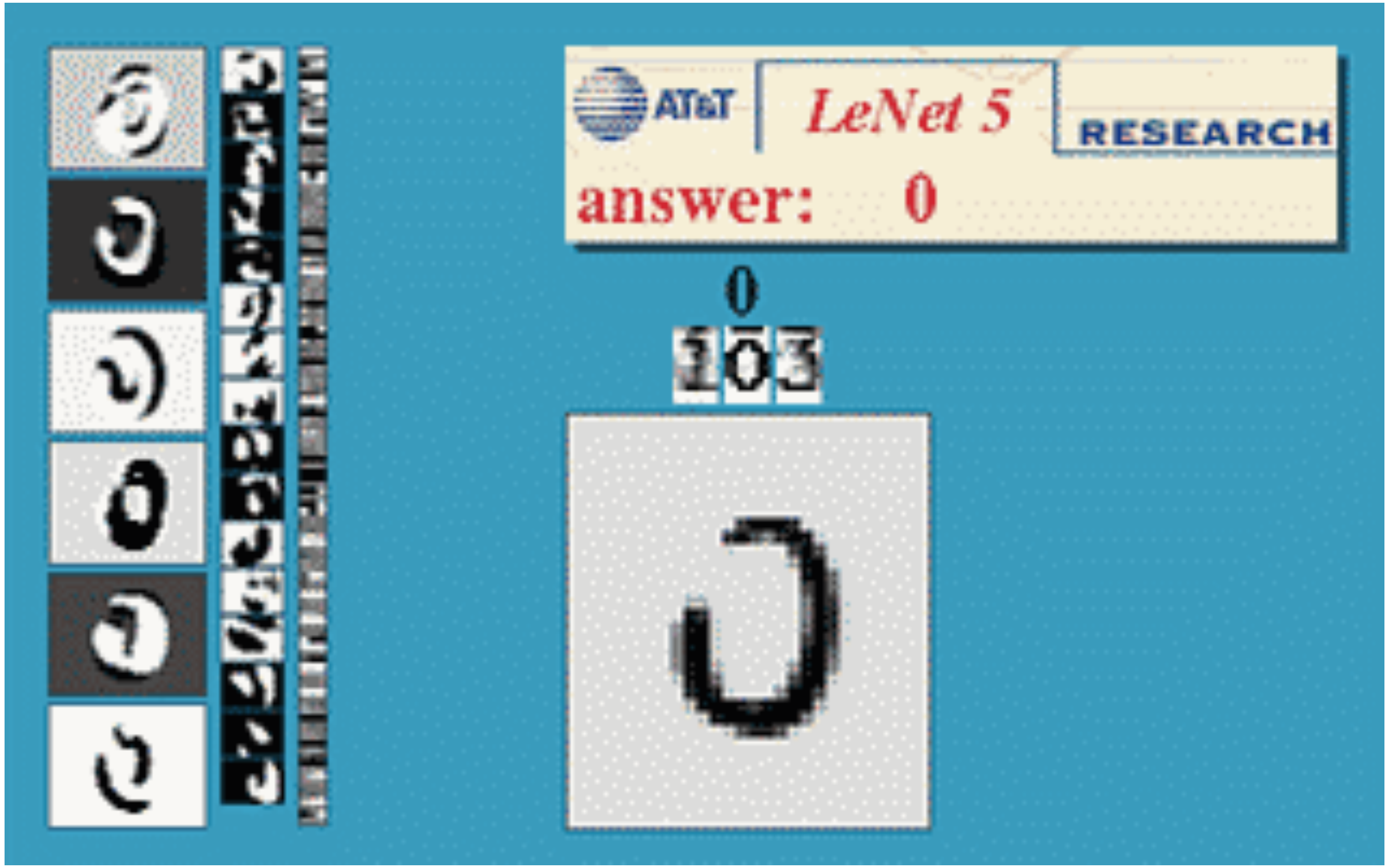
Training BatchNorm and Only BatchNorm: On the Expressive Power of Random Features in CNNs https://arxiv.org/abs/2003.00152, ICLR 2021

Parameter Initialization

- Let’s go simple. Set all weights and bias to 0. What happens?
- Choose wisely
- (or don’t choose at all; let PyTorch / Tensorflow do their thing; but check at least once that they are doing a reasonable thing)
- Small random Gaussian or uniform distribution
- Kaiming or Xavier is used typically in modern networks
- https://pytorch.org/docs/stable/nn.init.html

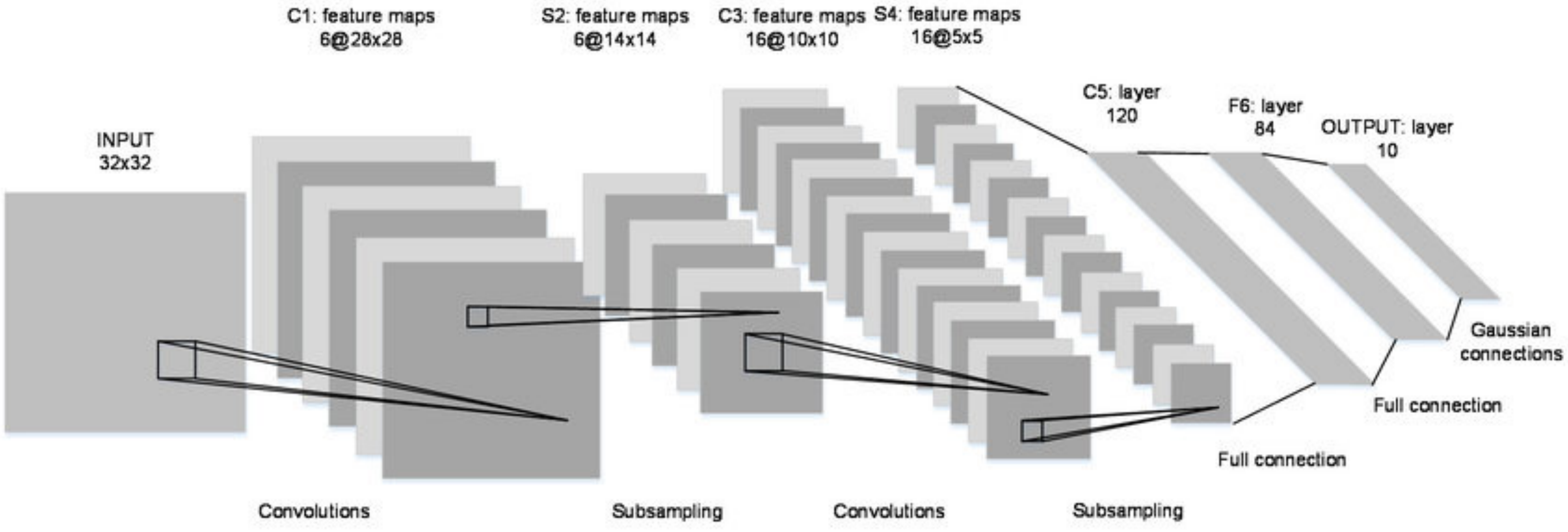
Stacking Lego blocks

LeNet (1989)



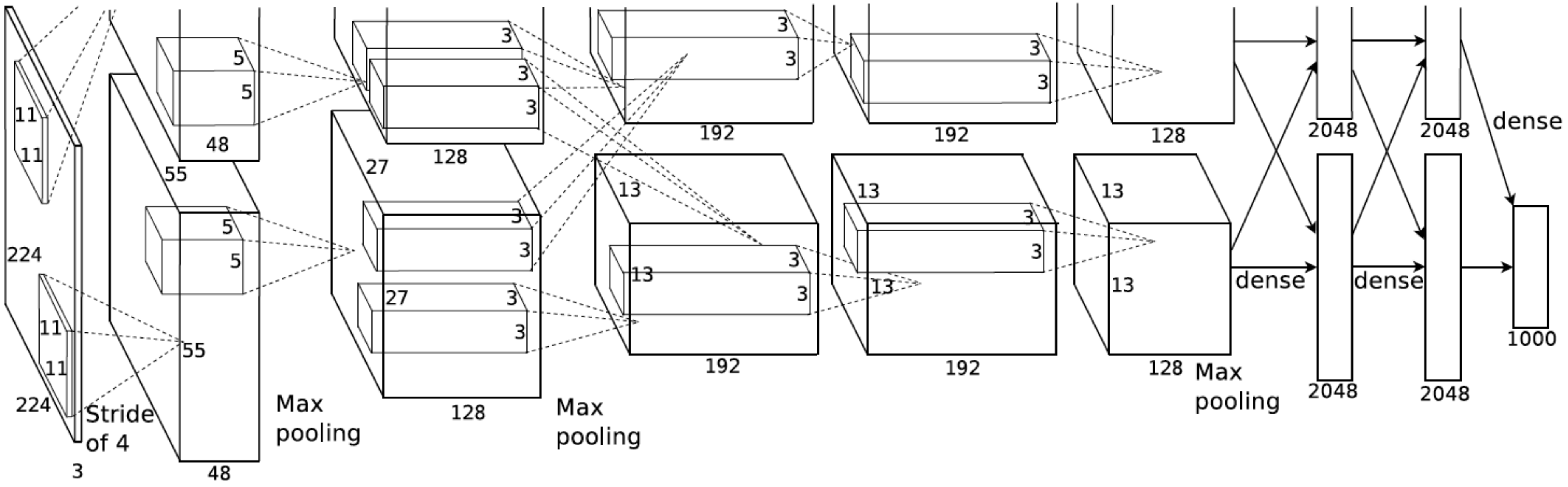
Source: http://yann.lecun.com/exdb/lenet/index.html

LeNet



C1: feature maps
6@28x28
S2: feature maps
6@14x14
C3: feature maps
16@10x10
S4: feature maps
16@5x5
C5: layer
120
F6: layer
84
OUTPUT: layer
10
INPUT
32x32
Gaussian
connections
Full connection
Full connection
Convolutions
Subsampling
Convolutions
Subsampling

AlexNet (2012)



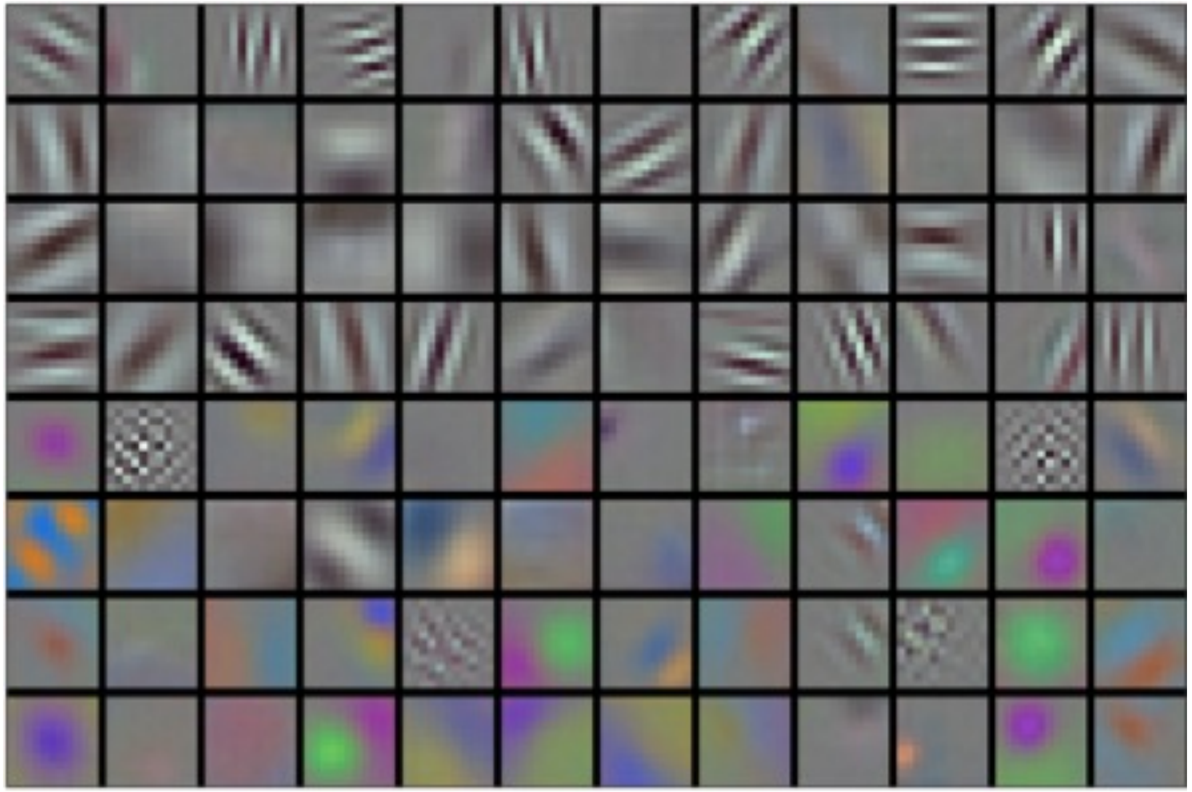
Neural Information Processing Systems
https://papers.nips.cc › paper › 4824-imagenet-classifi...

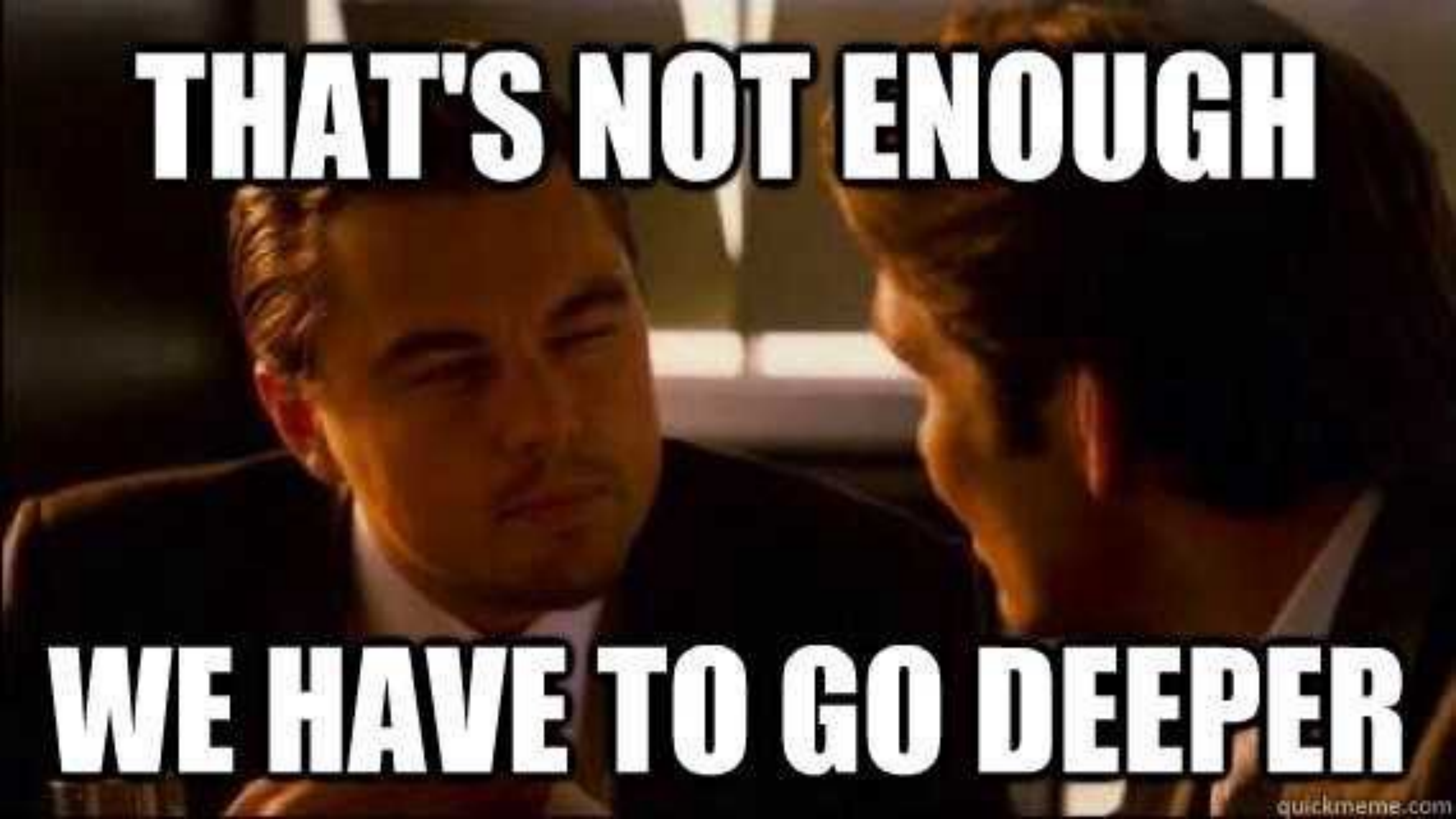
ImageNet Classification with Deep Convolutional Neural ...

by A Krizhevsky · 2012 · Cited by 118625 — We trained a large, **deep convolutional neural** network to **classify** the 1.3 million high-resolution images in the LSVRC-2010 **ImageNet** traini…

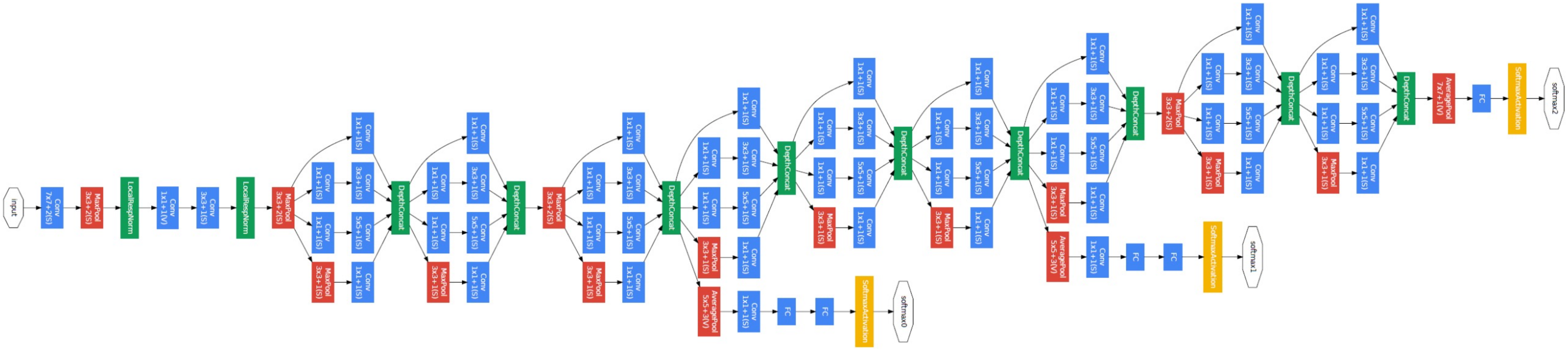
AlexNet

- Different size conv kernels
- Linear layers at the end
- Max pooling in between
- Data augmentation!
- 2 GPUs before
  `torch.nn.parallel.DataParallel`




THAT'S NOT ENOUGH
WE HAVE TO GO DEEPER
quickmeme.com

GoogLeNet (2014)

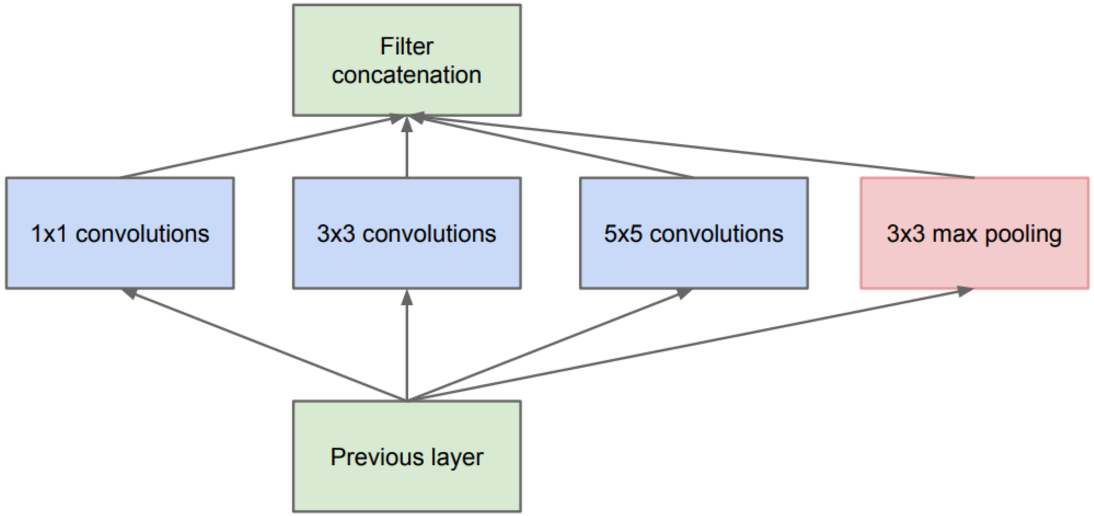


arXiv
https://arxiv.org › cs

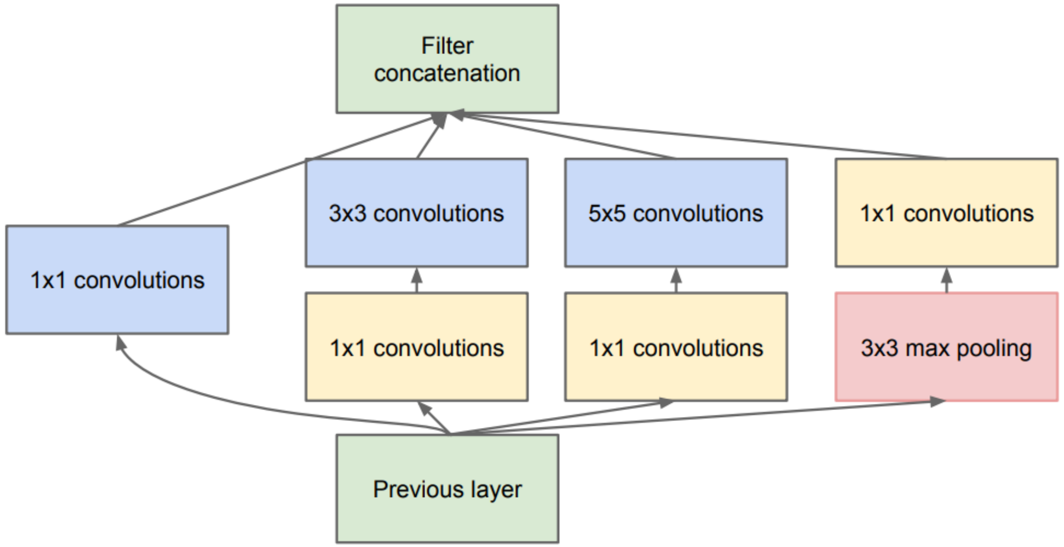
[1409.4842] Going Deeper with Convolutions

by C Szegedy · 2014 · Cited by 51715 — One particular incarnation used in our submission for ILSVRC 2014 is called **GoogLeNet**, a 22 layers deep network, the quality of which is ...

Inception Module



(a) Inception module, naïve version



(b) Inception module with dimension reductions

A Simple Guide to the Versions of the Inception Network
https://towardsdatascience.com/a-simple-guide-to-the-versions-of-the-inception-network-7fc52b863202

1x1 Convolution

- What's happening here?
- Input image: $B \times C_{in} \times H_{in} \times W_{in}$
  - Multi-channel processing C_{in}
  - Input data is a batch of B samples
- Convolution filter
  - Weight parameters: $C_{out} \times C_{in} \times K_H = 1 \times K_W = 1$
- Essentially in each spatial cell, C_{in} is converted to C_{out}

VGG-Net (2014)

- All convolutions with a 3x3 kernel
- All max-pooling layers are 2x2 kernel
- Linear layers at the end
- Plug and play in Caffe

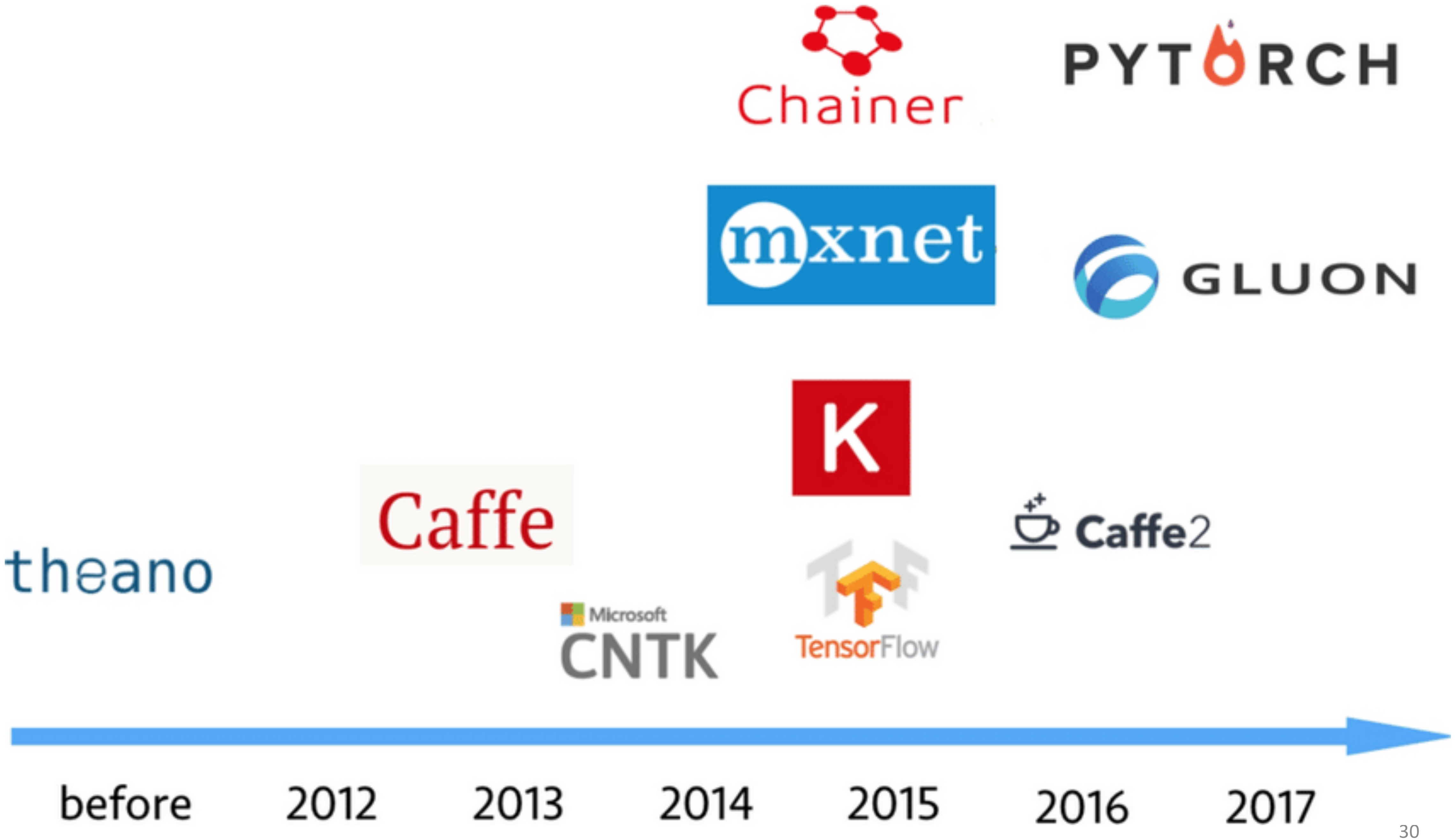
ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

arXiv
https://arxiv.org › cs

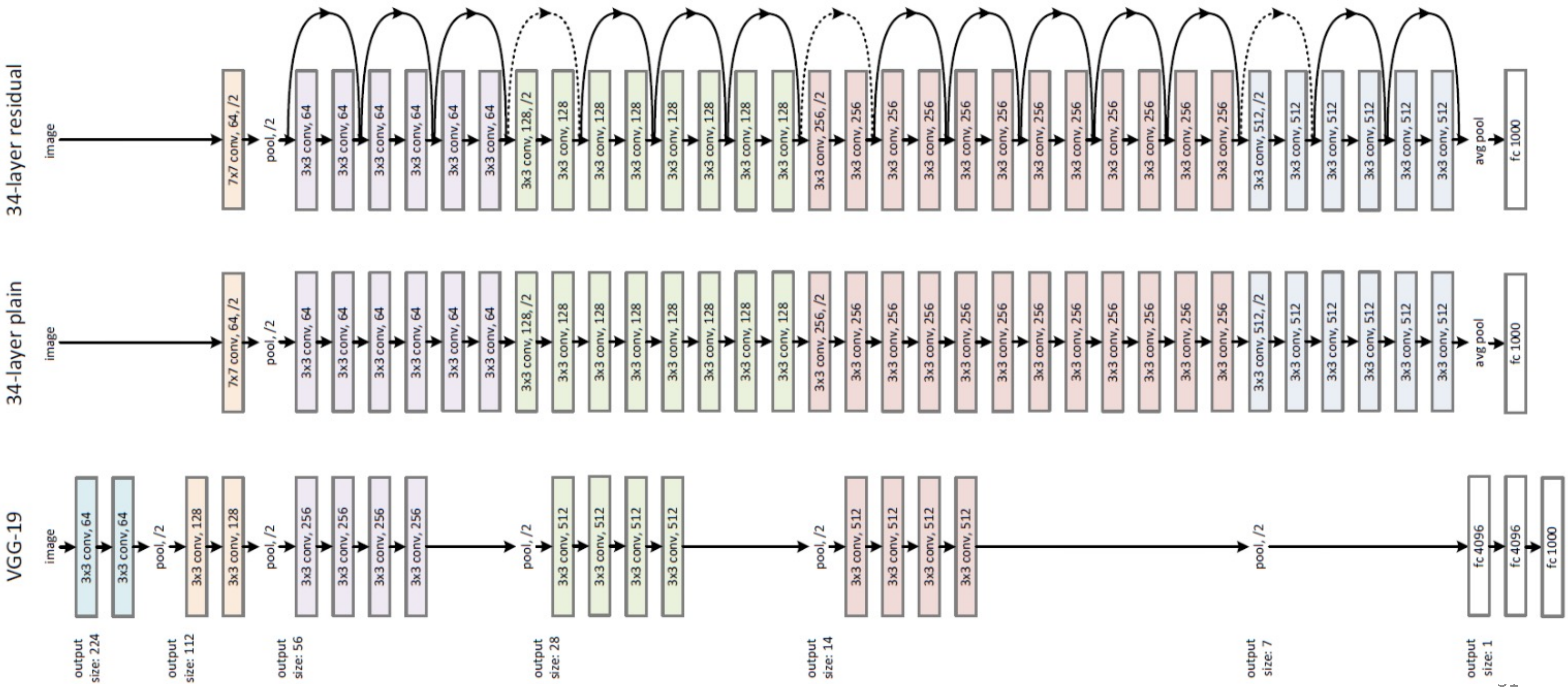
8

Very Deep Convolutional Networks for Large-Scale Image ...

by K Simonyan · 2014 · Cited by 105438 — In this work we investigate the effect of the **convolutional network** depth on its accuracy in the **large-scale image recognition** setting.…


Chainer
PYTORCH
mxnet
GLUON
K
Caffe
Caffe2
theano
Microsoft
CNTK
TensorFlow
before
2012
2013
2014
2015
2016
2017

ResNet (2015)


34-layer residual
34-layer plain
VGG-19
image
7x7 conv, 64, /2
pool, /2
3x3 conv, 64
3x3 conv, 128, /2
3x3 conv, 128
3x3 conv, 256, /2
3x3 conv, 256
3x3 conv, 512, /2
3x3 conv, 512
avg pool
fc 1000
3x3 conv, 64
3x3 conv, 128
3x3 conv, 256
3x3 conv, 512
fc 4096
output size: 224
output size: 112
output size: 56
output size: 28
output size: 14
output size: 7
output size: 1

Residual connections are very powerful!

- Solve the problems of vanishing gradients

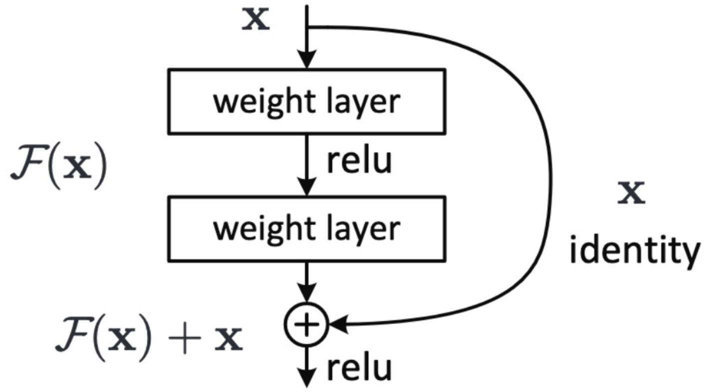
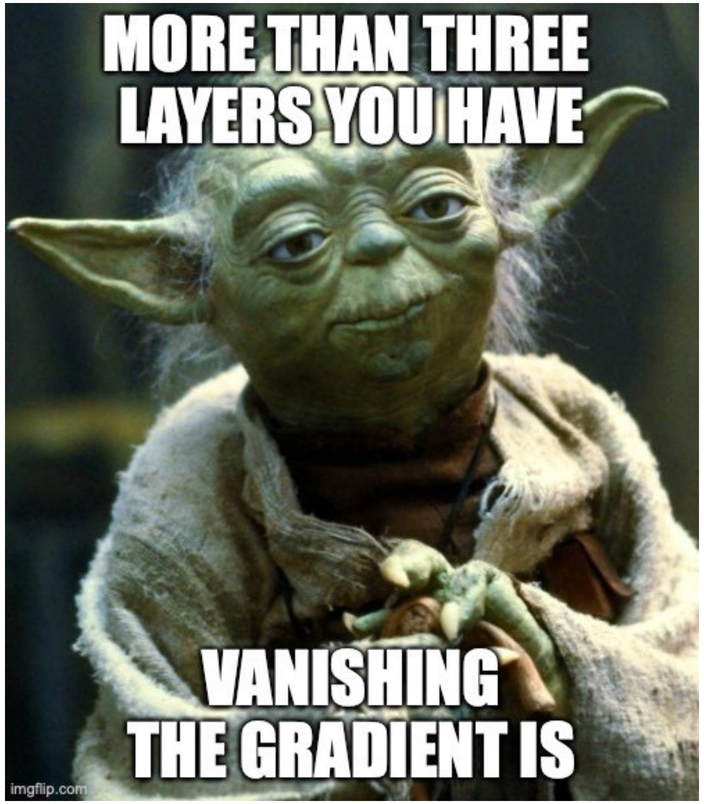


Figure 2. Residual learning: a building block.



Ok, maybe not exactly at 3, but you get the point

arXiv
https://arxiv.org › cs

8

[1512.03385] Deep Residual Learning for Image Recognition

by K He · 2015 · Cited by 172159 — We present a **residual learning** framework to ease the training of **networks** that are substantially **deeper** than those used previously.

Efficacy of residual connections

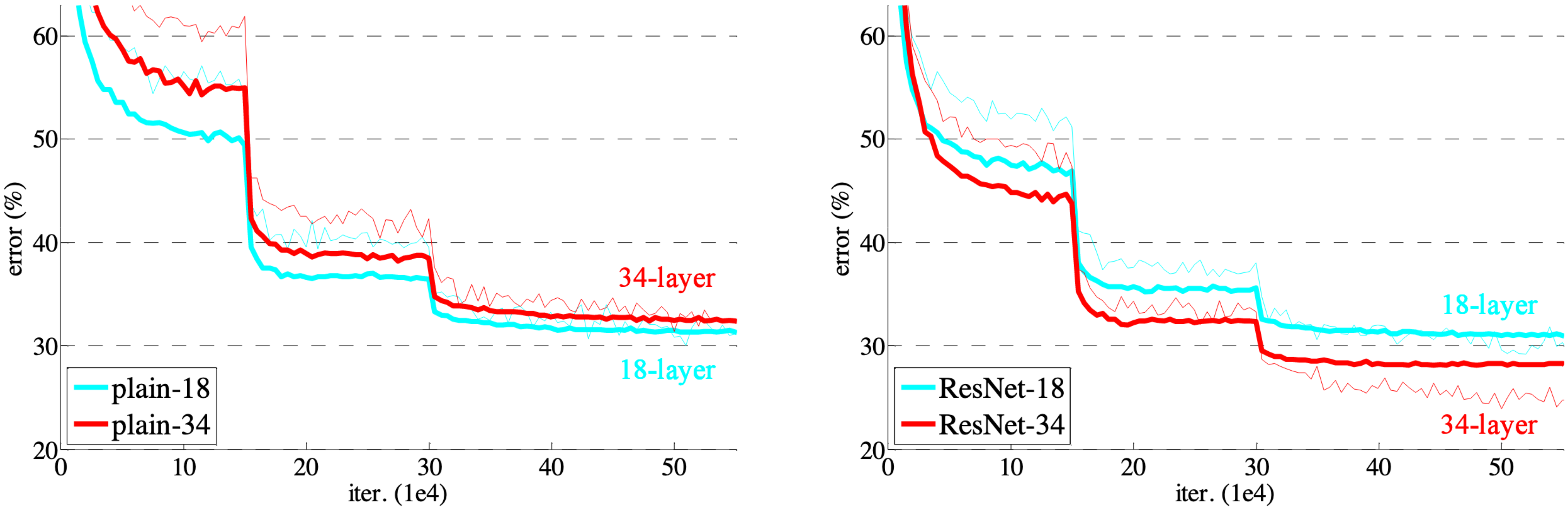


Figure 4. Training on **ImageNet**. Thin curves denote training error, and bold curves denote validation error of the center crops. Left: plain networks of 18 and 34 layers. Right: ResNets of 18 and 34 layers. In this plot, the residual networks have no extra parameter compared to their plain counterparts.

Source: https://arxiv.org/abs/1512.03385

Resnet modules

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				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9