Deep Learning

Deep learning

Representation learning with a hierarchy of concepts

Those concepts are represented by layers in a neural network model

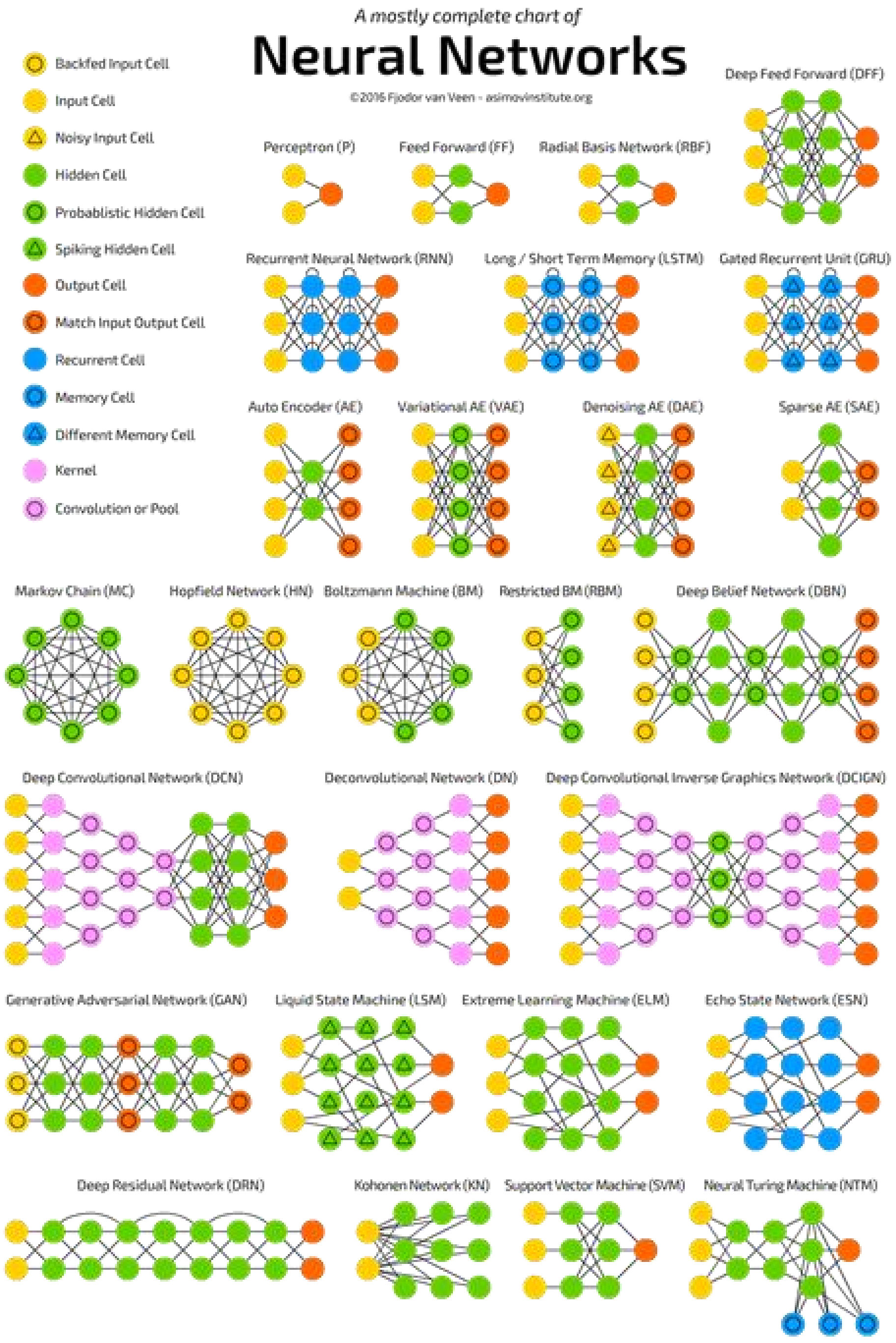
Types of Deep Learning Tools

Unsupervised models
- Autoencoders

Supervised models
- Image analysis:
    - Convolutional Neural Networks (CNNs)
    - …and Vision Transformers (ViTs)
- Text analysis and NLP
    - Transformers
- Timeseries analysis:
    - Transformers
    - Recurrent Neural Networks (RNNs)

Generative models
- Generative Adversarial Networks (GANs)
- Diffusion Models (e.g. DALL-E 2, Stable Diffusion)
- Generative Pre-trained Transformer (GPT)

Azimov Institute: http://www.asimovinstitute.org/neural-network-zoo/



Autoencoders

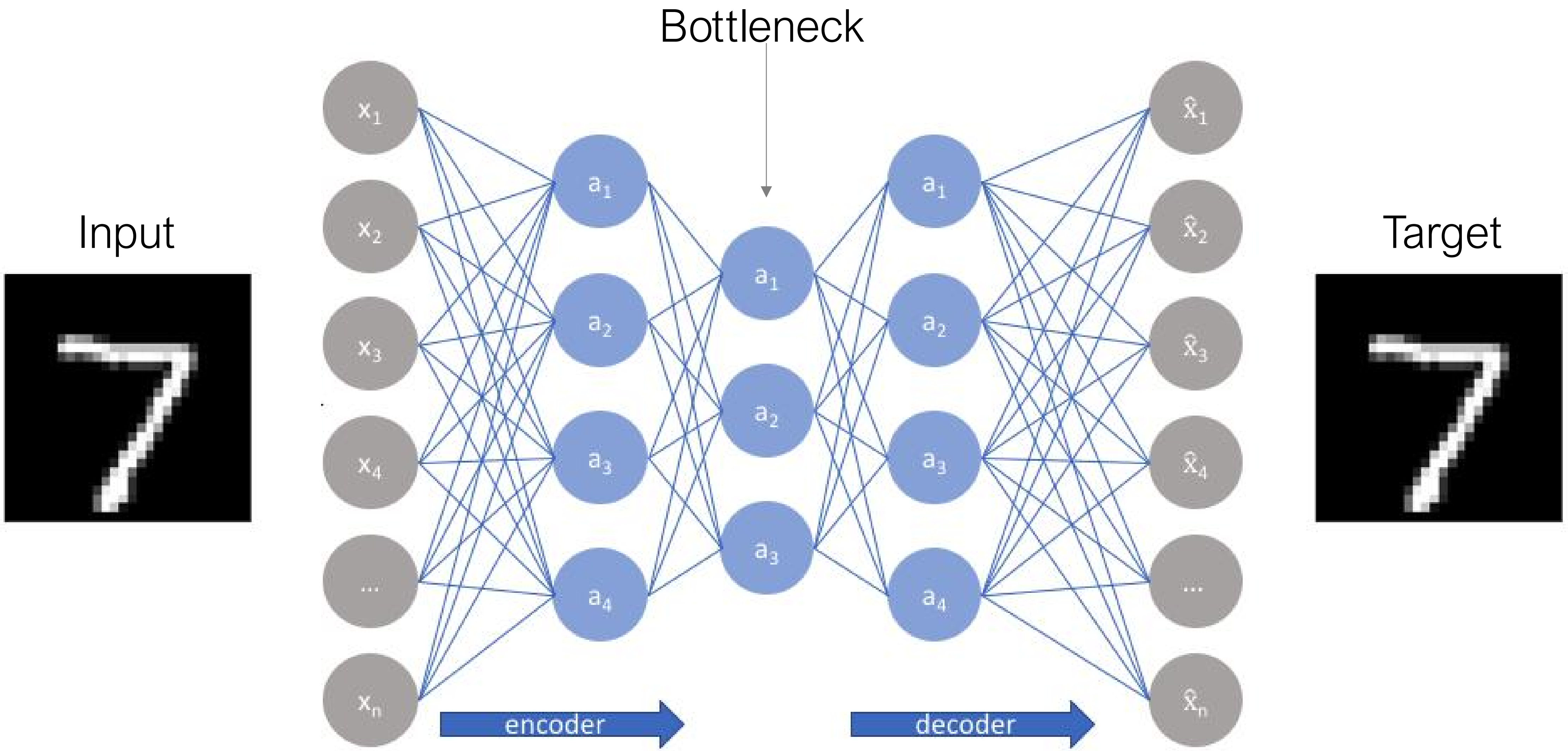
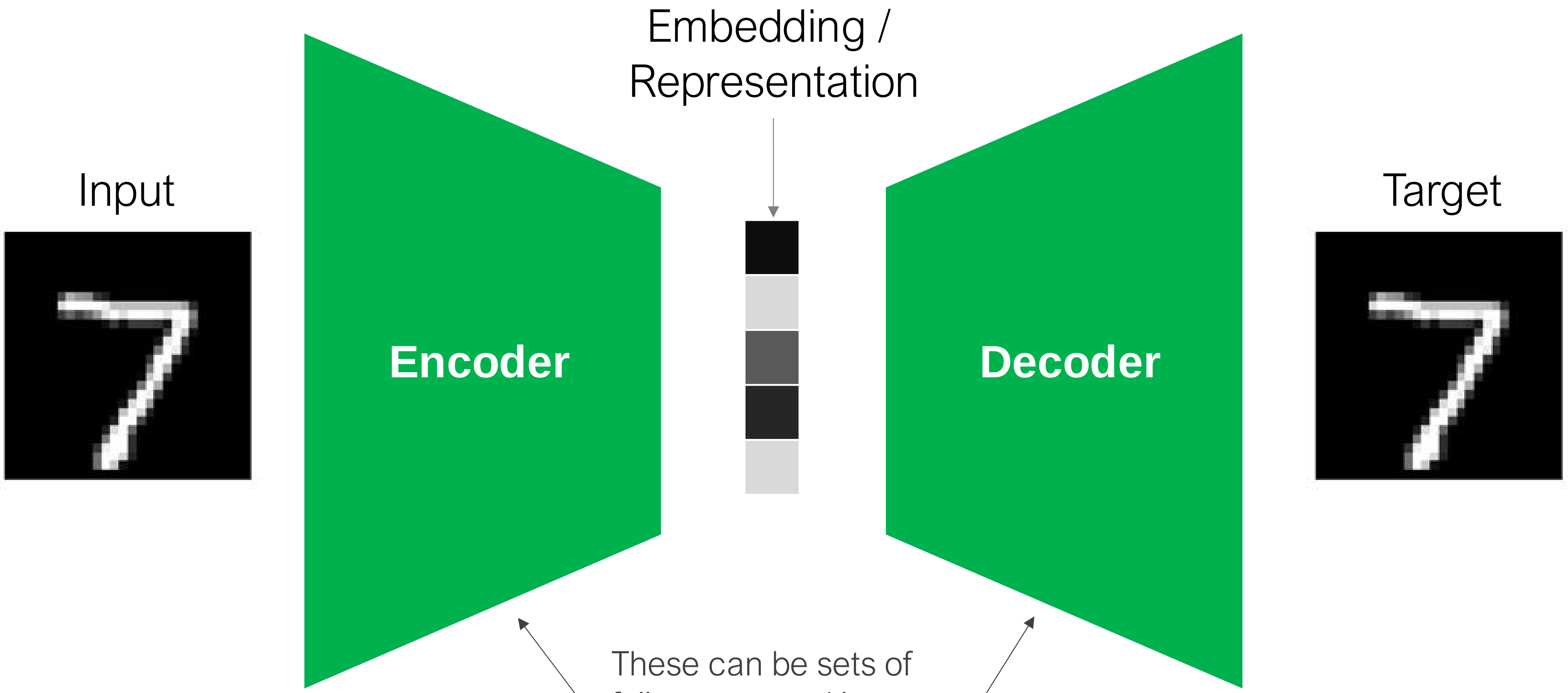


Image from: https://www.jeremyjordan.me/autoencoders/

Autoencoders

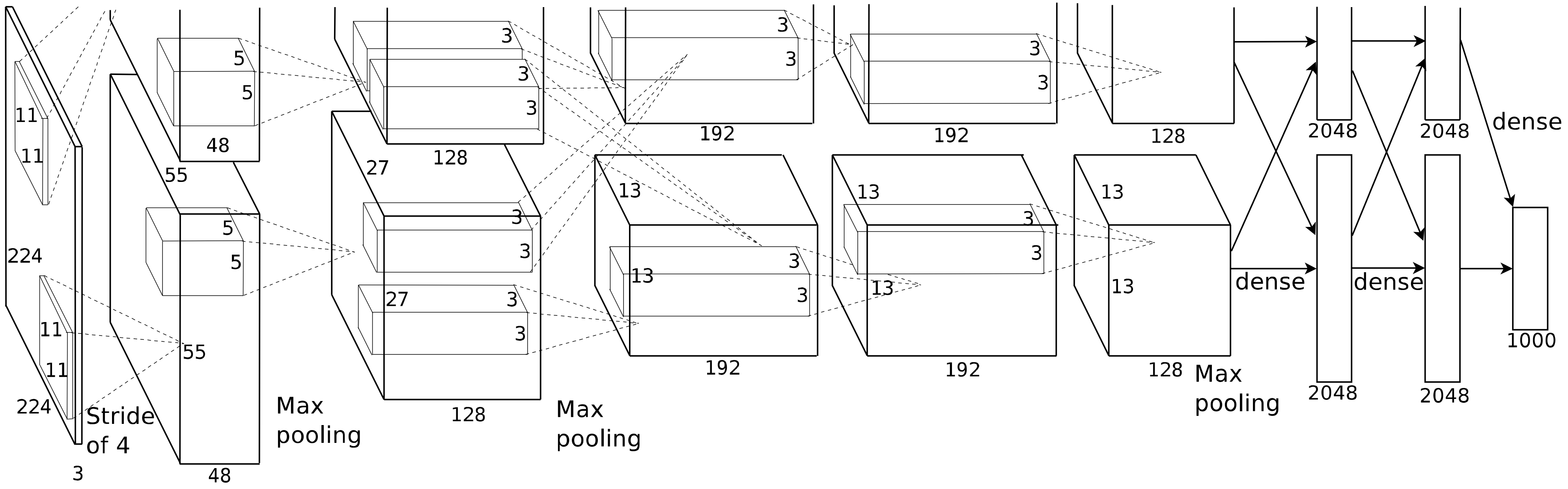
Our goal is often to develop a good **encoder** that represents our features well

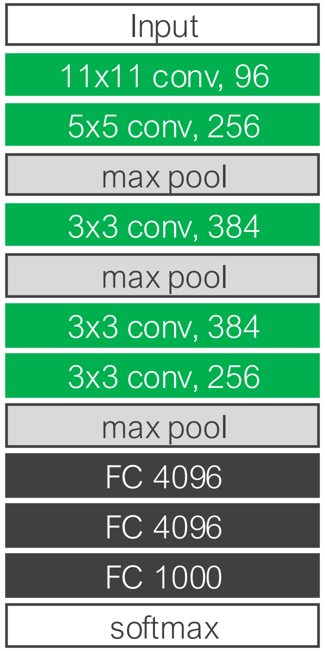


These can be sets of fully-connected layers or convolutional layers, etc.

Convolutional Neural Networks

AlexNet





Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. "Imagenet classification with deep convolutional neural networks." Advances in neural information processing systems. 2012.

Key

Input or output layer

Convolutional Layer

Fully Connected Layer

max pooling layer

Convolutional Neural Networks

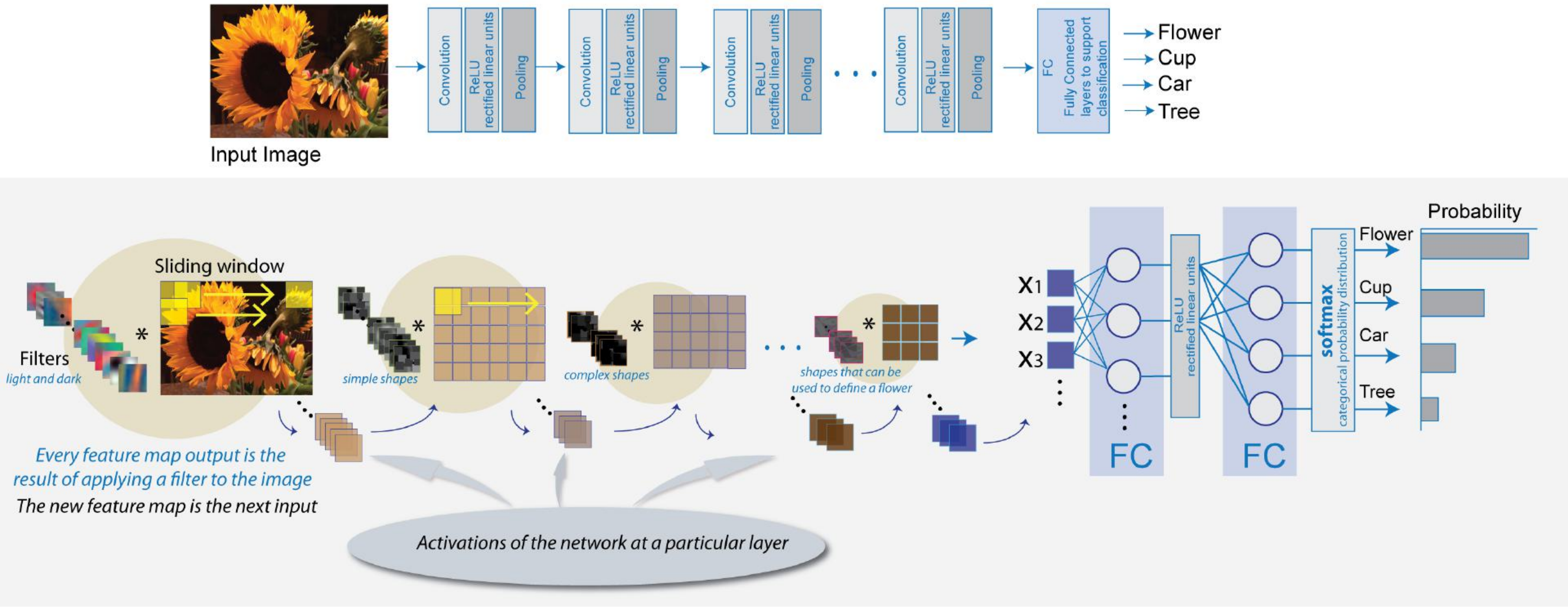
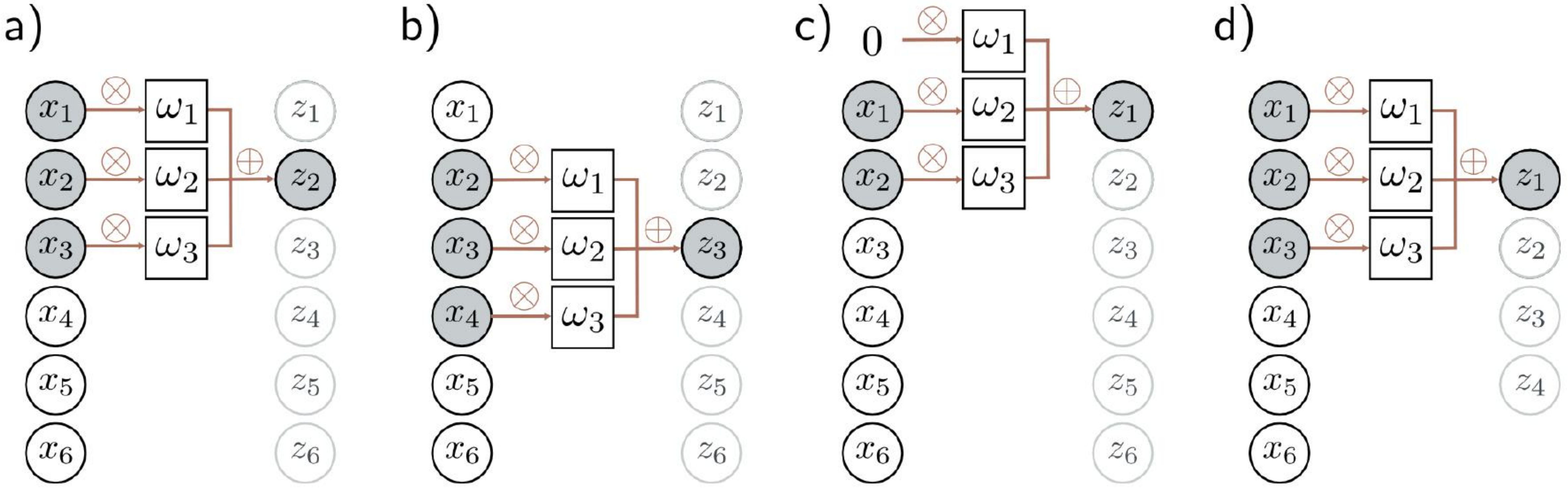


Image from the Mathworks

Convolution in 1 dimension



$z_2 = x_1 w_1 + x_2 w_2 + x_3 w_3$

$z_3 = x_2 w_1 + x_3 w_2 + x_4 w_3$

$z_1 = 0 w_1 + x_1 w_2 + x_2 w_3$

$z_1 = x_1 w_1 + x_2 w_2 + x_3 w_3$

Data: $\boldsymbol{x}$

1	2	5	1	4	2
0	2	3	2	0	0
4	5	5	9	8	1
6	3	4	2	3	1
0	1	9	8	7	2
2	3	5	5	5	6

$*$

Weights: $\boldsymbol{w}$

1	1	1
0	0	0
-1	-1	-1

$=$

Output: $\boldsymbol{x} * \boldsymbol{w}$

2D Convolution

Data: $\boldsymbol{x}$

1	2	5	1	4	2
0	2	3	2	0	0
4	5	5	9	8	1
6	3	4	2	3	1
0	1	9	8	7	2
2	3	5	5	5	6

$*$

Weights: $\boldsymbol{w}$

1	1	1
0	0	0
-1	-1	-1

$=$

Output: $\boldsymbol{x} * \boldsymbol{w}$

Computing one output value:

$$1 \cdot 1 + 1 \cdot 2 + 1 \cdot 5 +$$
$$0 \cdot 0 + 0 \cdot 2 + 0 \cdot 3 +$$
$$(-1) \cdot 4 + (-1) \cdot 5 + (-1) \cdot 5$$

2D Convolution

2D Convolution

Data: $\boldsymbol{x}$

1	2	5	1	4	2
0	2	3	2	0	0
4	5	5	9	8	1
6	3	4	2	3	1
0	1	9	8	7	2
2	3	5	5	5	6

$*$

Weights: $\boldsymbol{w}$

1	1	1
0	0	0
-1	-1	-1

$=$

Output: $\boldsymbol{x} * \boldsymbol{w}$

-6			

Computing one output value:

$$1 \cdot 1 + 1 \cdot 2 + 1 \cdot 5 +$$
$$0 \cdot 0 + 0 \cdot 2 + 0 \cdot 3 +$$
$$(-1) \cdot 4 + (-1) \cdot 5 + (-1) \cdot 5 = \mathbf{-6}$$

Data: $\boldsymbol{X}$

1	2	5	1	4	2
0	2	3	2	0	0
4	5	5	9	8	1
6	3	4	2	3	1
0	1	9	8	7	2
2	3	5	5	5	6

$*$

Weights: $\boldsymbol{w}$

1	1	1
0	0	0
-1	-1	-1

$=$

Output: $\boldsymbol{X} * \boldsymbol{w}$

-6	**-11**		

Computing one output value:

$$1 \cdot 2 + 1 \cdot 5 + 1 \cdot 1 + 0 \cdot 2 + 0 \cdot 3 + 0 \cdot 2 + (-1) \cdot 5 + (-1) \cdot 5 + (-1) \cdot 9 = \mathbf{-11}$$

2D Convolution

Data: $\boldsymbol{X}$

1	2	5	1	4	2
0	2	3	2	0	0
4	5	5	9	8	1
6	3	4	2	3	1
0	1	9	8	7	2
2	3	5	5	5	6

$*$

Weights: $\boldsymbol{w}$

1	1	1
0	0	0
-1	-1	-1

=

Output: $\boldsymbol{X} * \boldsymbol{w}$

-6	**-11**	**-12**	

Computing one output value:

$$
\begin{aligned}
&1 \cdot 5 + 1 \cdot 1 + 1 \cdot 4 + \\
&0 \cdot 3 + 0 \cdot 2 + 0 \cdot 0 + \\
&(-1) \cdot 5 + (-1) \cdot 9 + (-1) \cdot 8 = \mathbf{-12}
\end{aligned}
$$

2D Convolution

Data: $\boldsymbol{X}$

1	2	5	1	4	2
0	2	3	2	0	0
4	5	5	9	8	1
6	3	4	2	3	1
0	1	9	8	7	2
2	3	5	5	5	6

$*$

Weights: $\boldsymbol{w}$

1	1	1
0	0	0
-1	-1	-1

$=$

Output: $\boldsymbol{X} * \boldsymbol{w}$

-6	**-11**	**-12**	**-11**

Computing one output value:

$$1 \cdot 1 + 1 \cdot 4 + 1 \cdot 2 +$$
$$0 \cdot 2 + 0 \cdot 0 + 0 \cdot 0 +$$
$$(-1) \cdot 9 + (-1) \cdot 8 + (-1) \cdot 1 = \mathbf{-11}$$

2D Convolution

2D Convolution

Data: $\boldsymbol{X}$

1	2	5	1	4	2
0	2	3	2	0	0
4	5	5	9	8	1
6	3	4	2	3	1
0	1	9	8	7	2
2	3	5	5	5	6

$*$

Weights: $\boldsymbol{w}$

1	1	1
0	0	0
-1	-1	-1

$=$

Output: $\boldsymbol{X} * \boldsymbol{w}$

-6	**-11**	**-12**	**-11**
-7			

Computing one output value:

$$1 \cdot 0 + 1 \cdot 2 + 1 \cdot 3 + 0 \cdot 4 + 0 \cdot 5 + 0 \cdot 5 + (-1) \cdot 6 + (-1) \cdot 3 + (-1) \cdot 4 = \mathbf{-7}$$

Data: $\boldsymbol{X}$

1	2	5	1	4	2
0	2	3	2	0	0
4	5	5	9	8	1
6	3	4	2	3	1
0	1	9	8	7	2
2	3	5	5	5	6

6 x 6

$*$

Weights: $\boldsymbol{w}$

1	1	1
0	0	0
-1	-1	-1

3 x 3

=

Output: $\boldsymbol{X} * \boldsymbol{w}$

-6	**-11**	**-12**	**-11**
-7	**-2**	**-2**	**-4**
4	**1**	**-2**	**1**
3	**-4**	**-6**	**-10**

4 x 4

2D Convolution

What features do layers respond to?

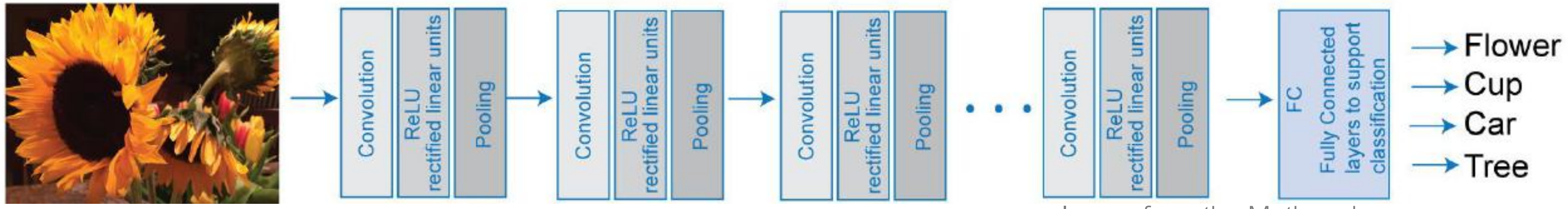
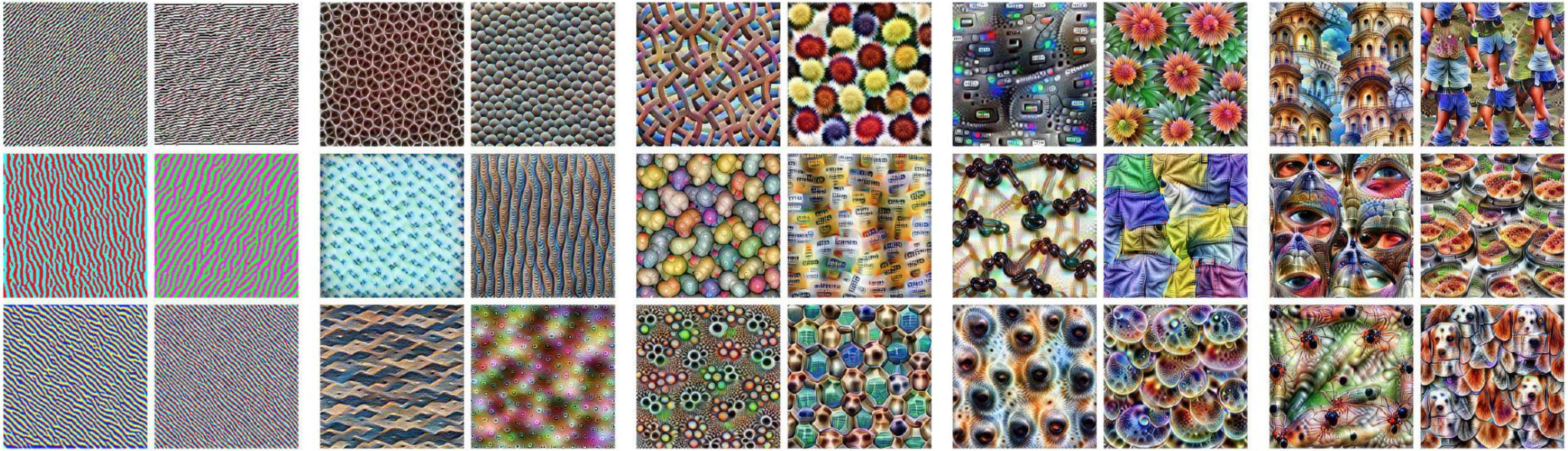


Image from the Mathworks



Edges (layer conv2d0) **Textures** (layer mixed3a) **Patterns** (layer mixed4a) **Parts** (layers mixed4b & mixed4c) **Objects** (layers mixed4d & mixed4e)

Olah et al, 2017: https://distill.pub/2017/feature-visualization/

Features

Dataset Examples show us what neurons respond to in practice

Optimization isolates the causes of behavior from mere correlations. A neuron may not be detecting what you initially thought.

Baseball—or stripes? *mixed4a, Unit 6*	Animal faces—or snouts? *mixed4a, Unit 240*	Clouds—or fluffiness? *mixed4a, Unit 453*	Buildings—or sky? *mixed4a, Unit 492*

Olah et al, 2017: https://distill.pub/2017/feature-visualization/

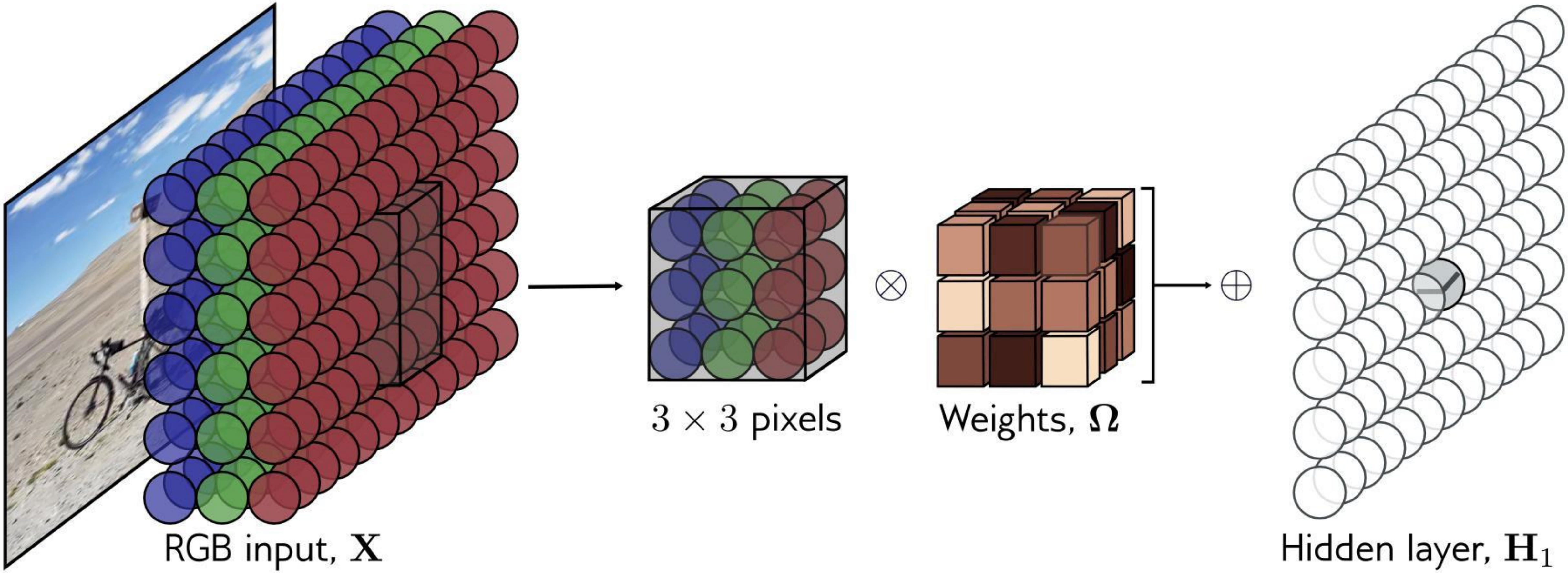
Resources on Visualization of Features

Feature visualization: https://distill.pub/2017/feature-visualization/

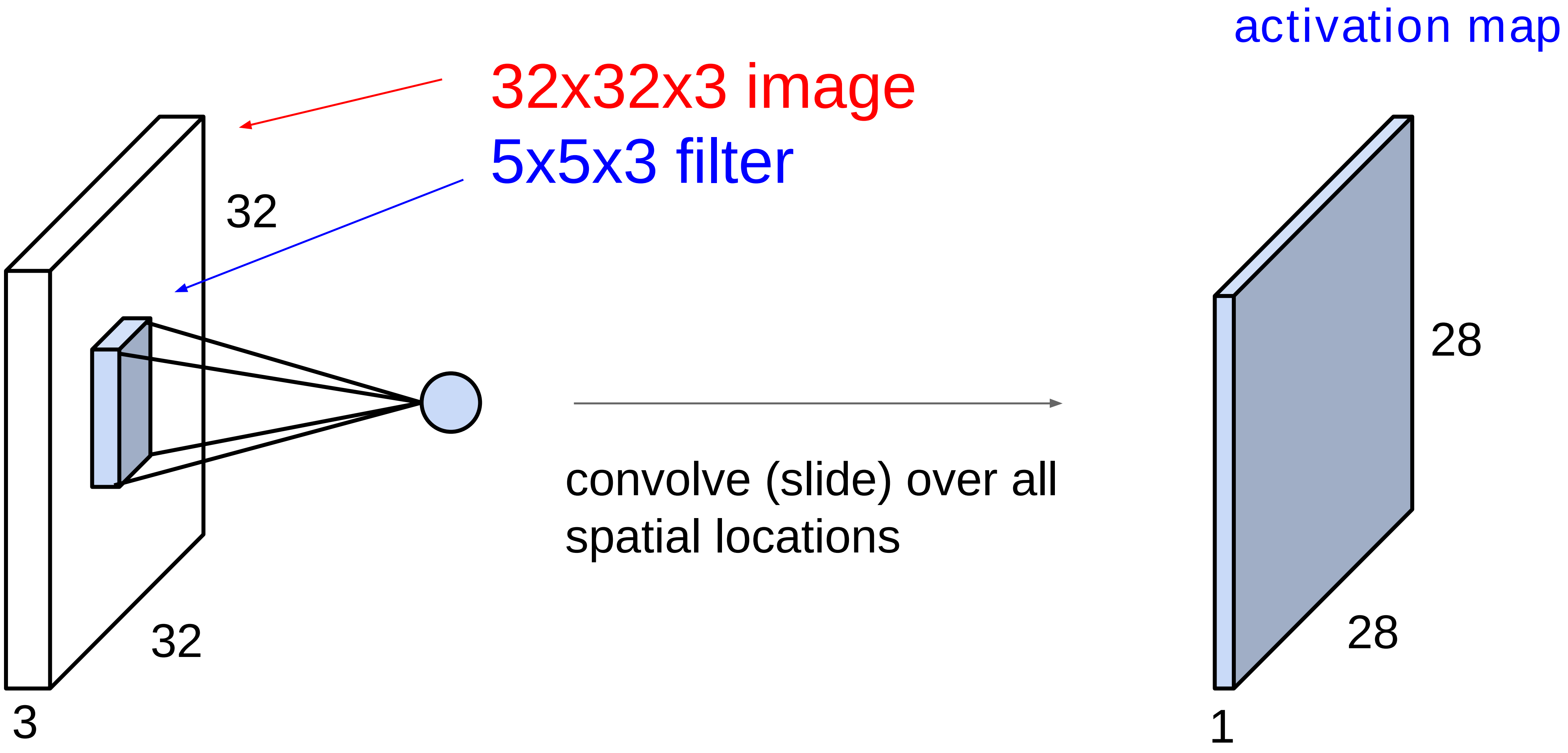
Building blocks of interpretability: https://distill.pub/2018/building-blocks/

Activation Activation Atlases: https://distill.pub/2019/activation-atlas/

Convolution Example

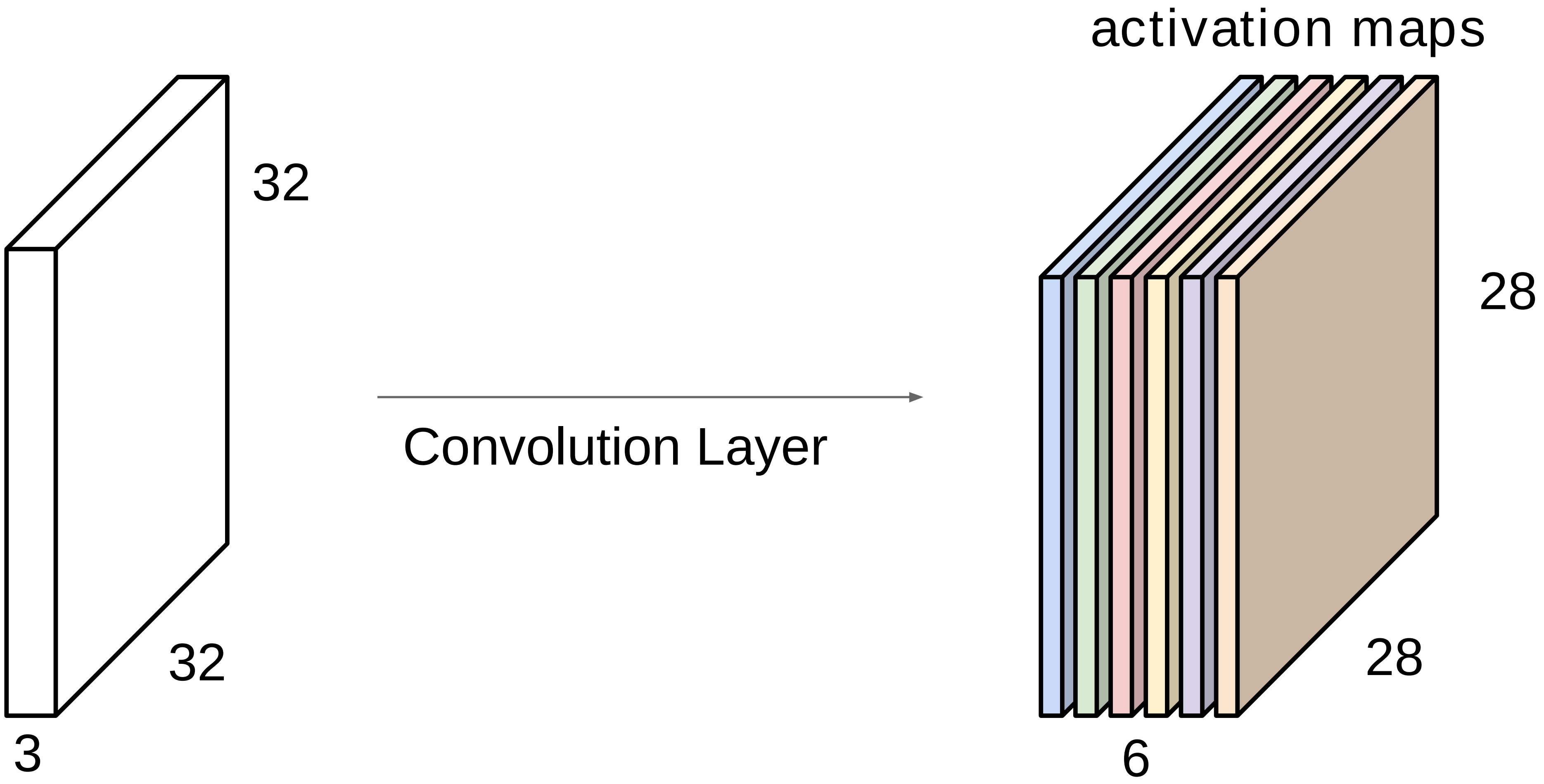


Convolution Layer



From Fei-Fei Li, Justin Johnson, and Serena Young. CS231n, 2018

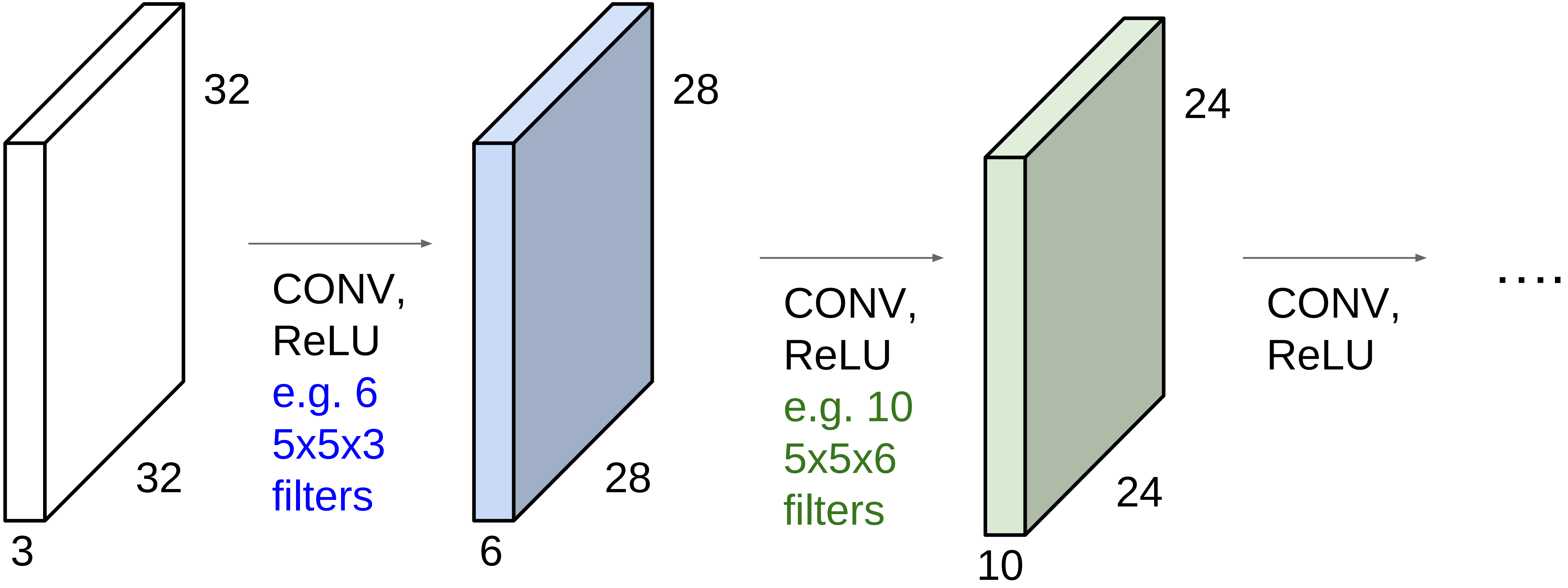
For example, if we had 6 5x5 filters, we'll get 6 separate activation maps:



We stack these up to get a "new image" of size 28x28x6!

From Fei-Fei Li, Justin Johnson, and Serena Young. CS231n, 2018

Preview: ConvNet is a sequence of Convolution Layers, interspersed with activation functions

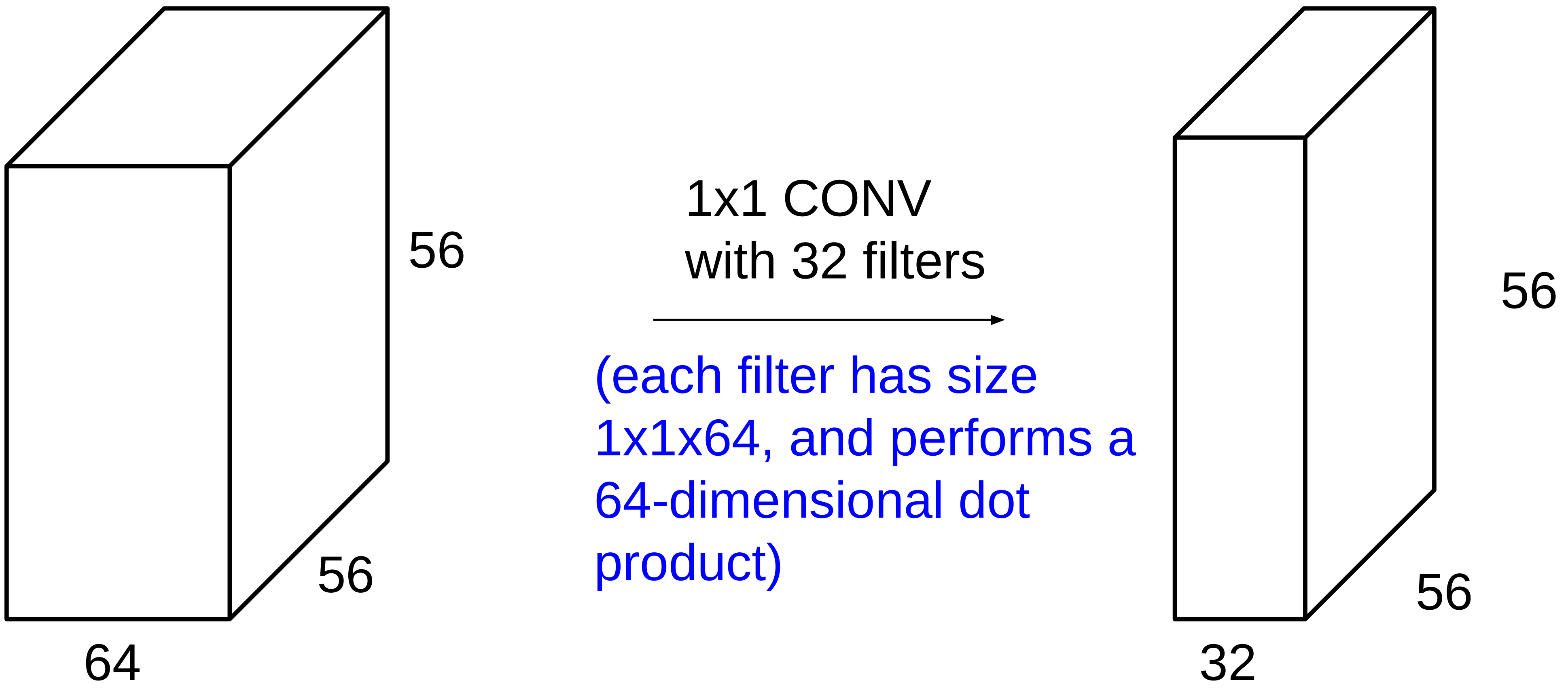


Parameters = (5*5*3)*6 = 450

(5*5*6)*10 = 1,500

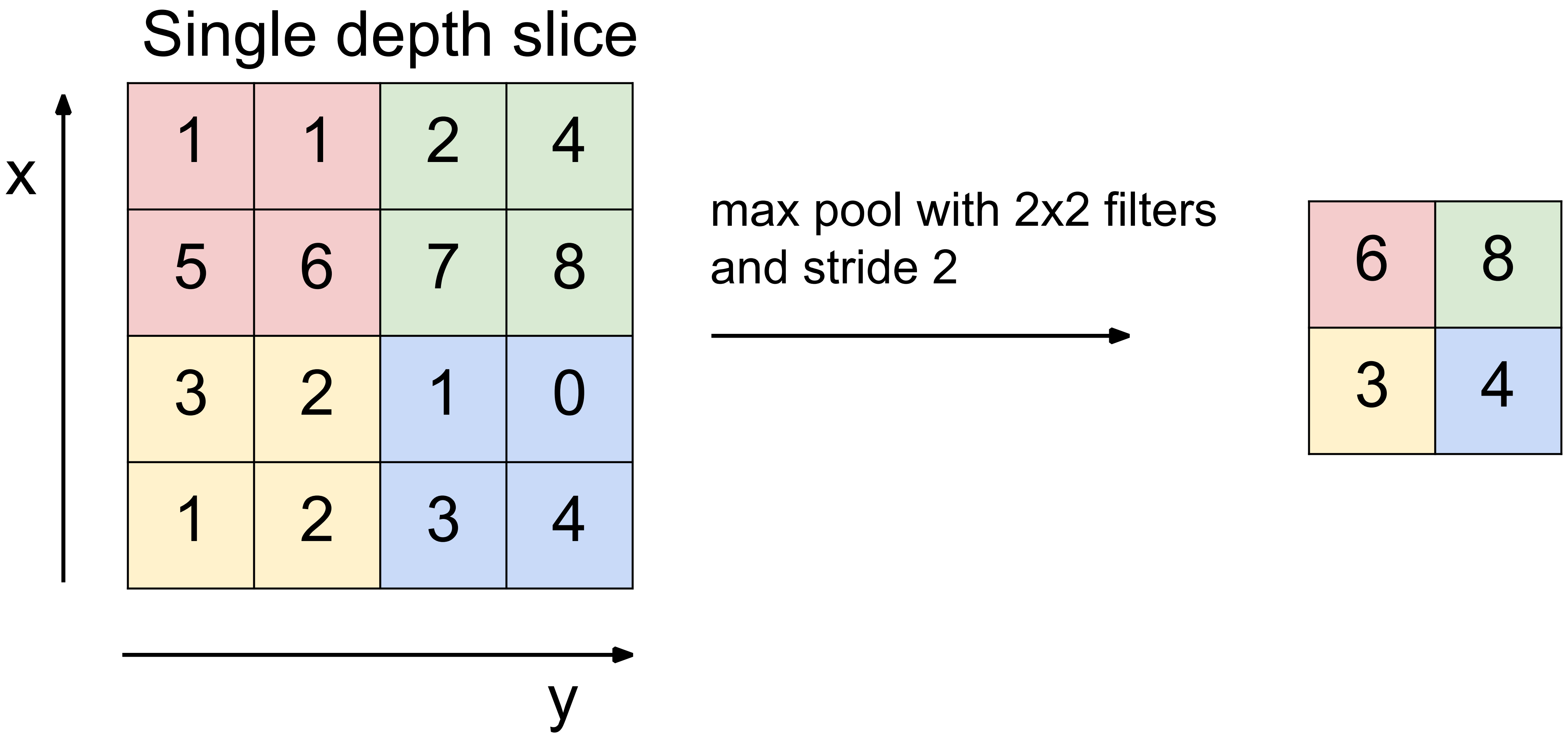
From Fei-Fei Li, Justin Johnson, and Serena Young. CS231n, 2018

1 x 1 Convolution Explained



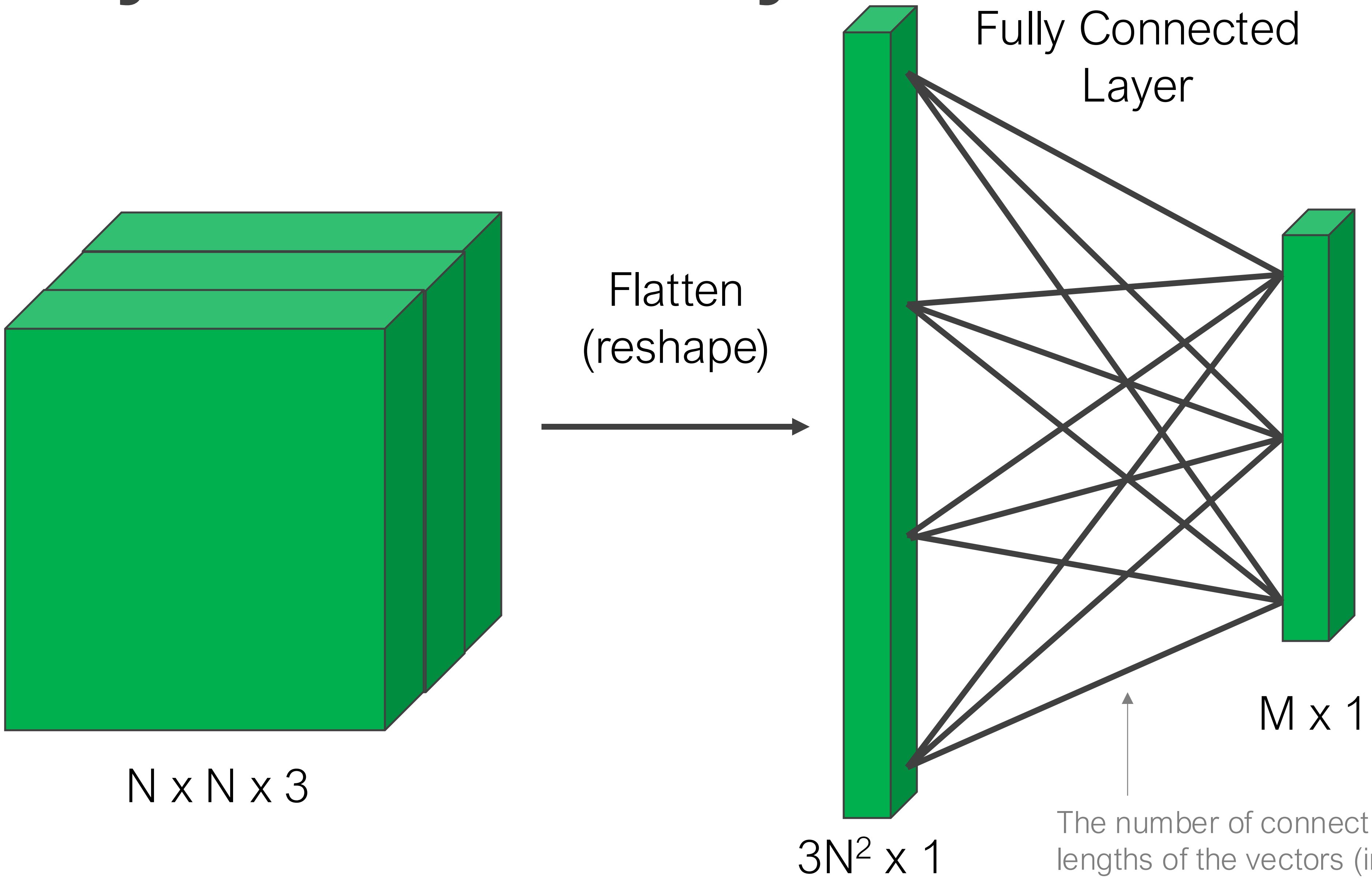
From Fei-Fei Li, Justin Johnson, and Serena Young. CS231n, 2018

Max Pooling



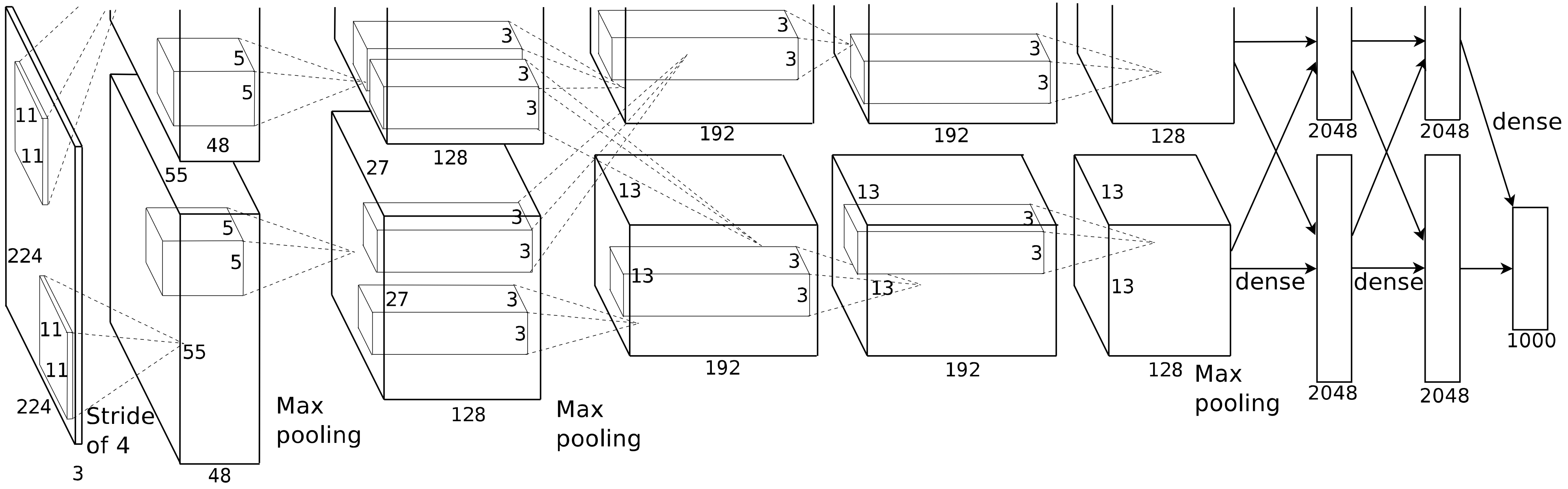
From Fei-Fei Li, Justin Johnson, and Serena Young. CS231n, 2018

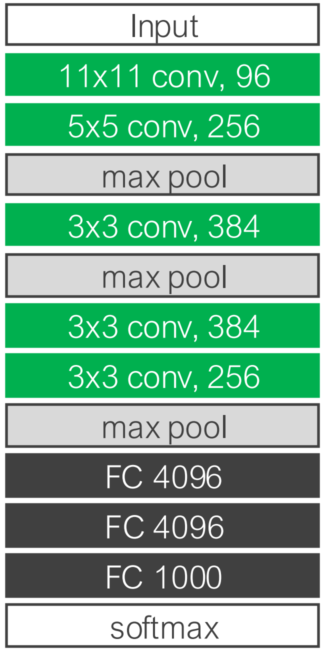
Fully Connected Layer



The number of connections is the product of the lengths of the vectors (in this case $3N^2M$)

AlexNet





Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. "Imagenet classification with deep convolutional neural networks." Advances in neural information processing systems. 2012.

Key

- Input or output layer
- Convolutional Layer
- Fully Connected Layer
- max pooling layer

CNN Architectures

AlexNet (2012)

- Input
- 11x11 conv, 96
- 5x5 conv, 256
- max pool
- 3x3 conv, 384
- max pool
- 3x3 conv, 384
- 3x3 conv, 256
- max pool
- FC 4096
- FC 4096
- FC 1000
- softmax

VGG16 (2014)

- Input
- 3x3 conv, 64
- 3x3 conv, 64
- max pool
- 3x3 conv, 128
- 3x3 conv, 128
- max pool
- 3x3 conv, 256
- 3x3 conv, 256
- 3x3 conv, 256
- max pool
- 3x3 conv, 512
- 3x3 conv, 512
- 3x3 conv, 512
- max pool
- 3x3 conv, 512
- 3x3 conv, 512
- 3x3 conv, 512
- max pool
- FC 4096
- FC 4096
- FC 1000
- softmax

VGG19 (2014)

- Input
- 3x3 conv, 64
- 3x3 conv, 64
- max pool
- 3x3 conv, 128
- 3x3 conv, 128
- max pool
- 3x3 conv, 256
- 3x3 conv, 256
- 3x3 conv, 256
- 3x3 conv, 256
- max pool
- 3x3 conv, 512
- 3x3 conv, 512
- 3x3 conv, 512
- 3x3 conv, 512
- max pool
- 3x3 conv, 512
- 3x3 conv, 512
- 3x3 conv, 512
- 3x3 conv, 512
- max pool
- FC 4096
- FC 4096
- FC 1000
- softmax

Note: an activation function is applied to the output of each layer

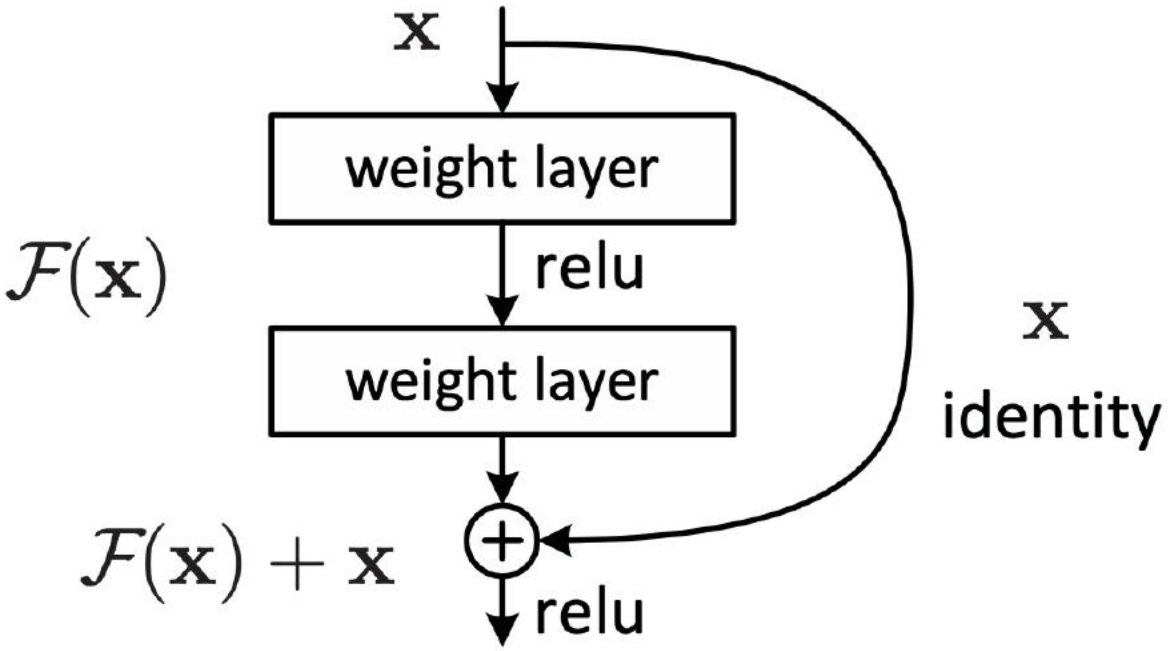
Fewer layers, larger filters

Key

- Input or output layer
- Convolutional Layer
- Fully Connected Layer
- max pooling layer

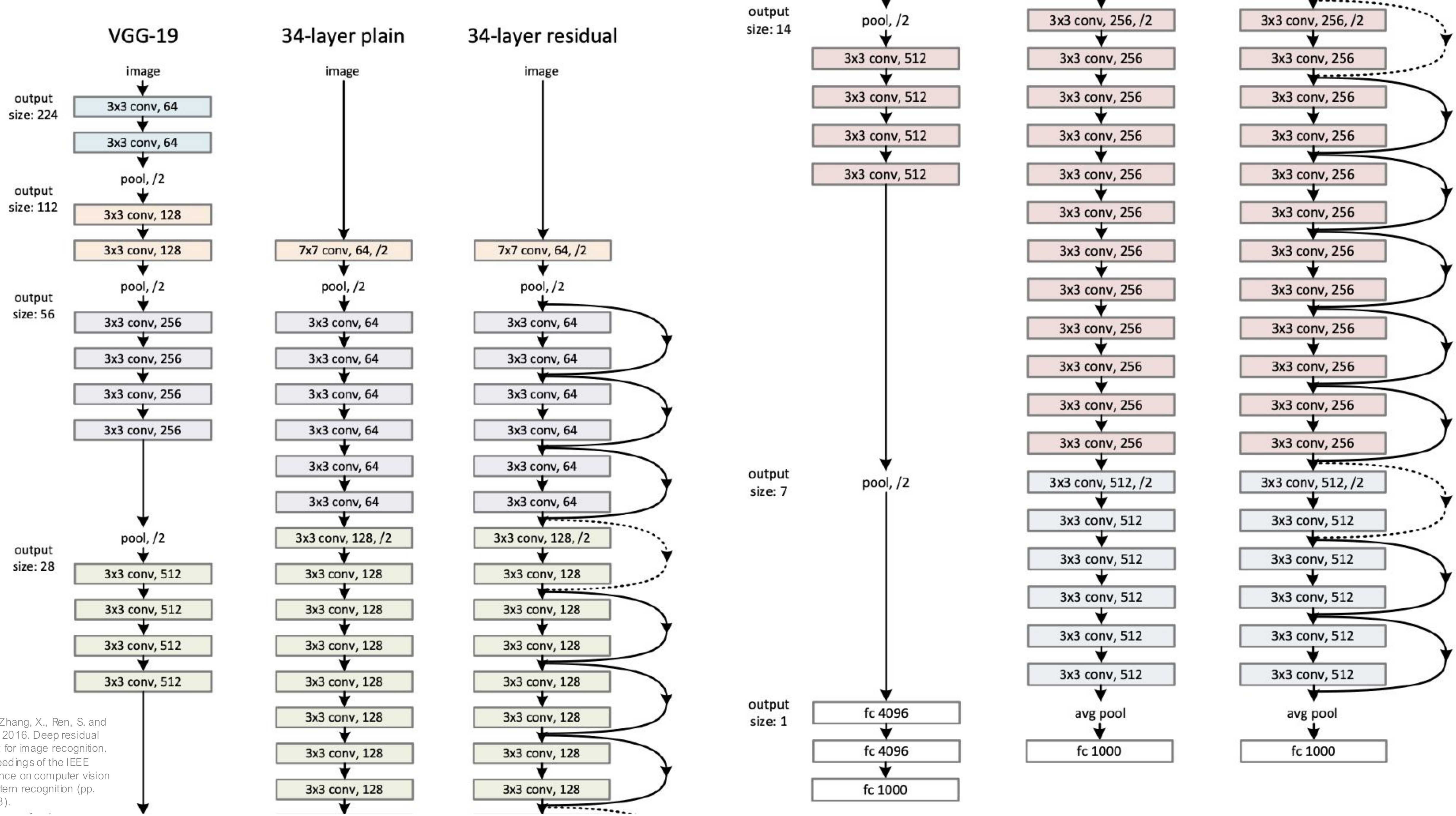
Adapted from Fei-Fei Li, Justin Johnson, and Serena Young. CS231n, 2018

Residual Networks (ResNet)



Skip Connection enable faster convergence, more effectively backpropate the error signal (avoiding vanishing gradients)

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



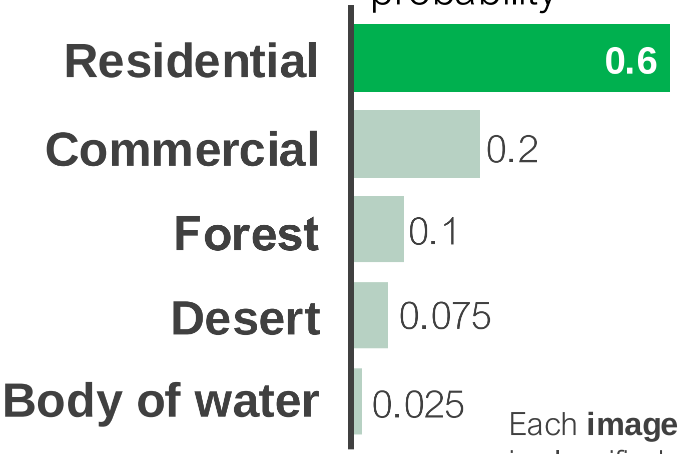
He, K., Zhang, X., Ren, S. and Sun, J., 2016. Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778).


Scene
classification
Object
detection
Image
segmentation
Confidence or
probability
Residential
0.6
Commercial
0.2
Forest
0.1
Desert
0.075
Body of water
0.025
Each **image (scene)** is classified
Object type & probability
Building 0.8
Building 0.7
Building 0.9
Building 0.4
Building 0.6
Car 0.85
Probability
Building
0.7
Road
0.2
Plant
0.06
Water
0.04
Each **pixel** is classified

Scene classification

AlexNet, VGG, GoogLeNet, ResNet, Inception, DenseNet, SqueezeNet, EfficientNet

Confidence or probability



Each **image (scene)** is classified

Object detection

Faster/Fast/R-CNN
Mask R-CNN
YOLO
Single Shot Detector (SSD)
RetinaNet

Object type & probability

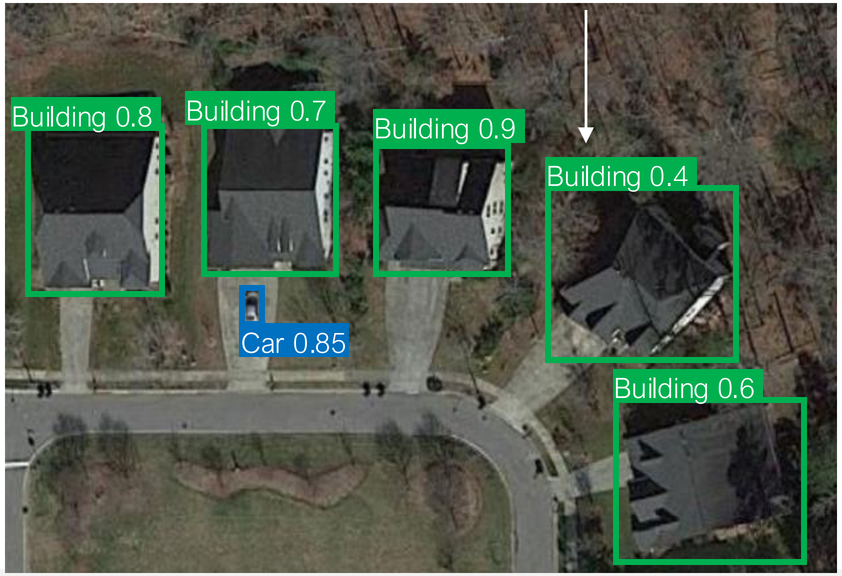


Image segmentation

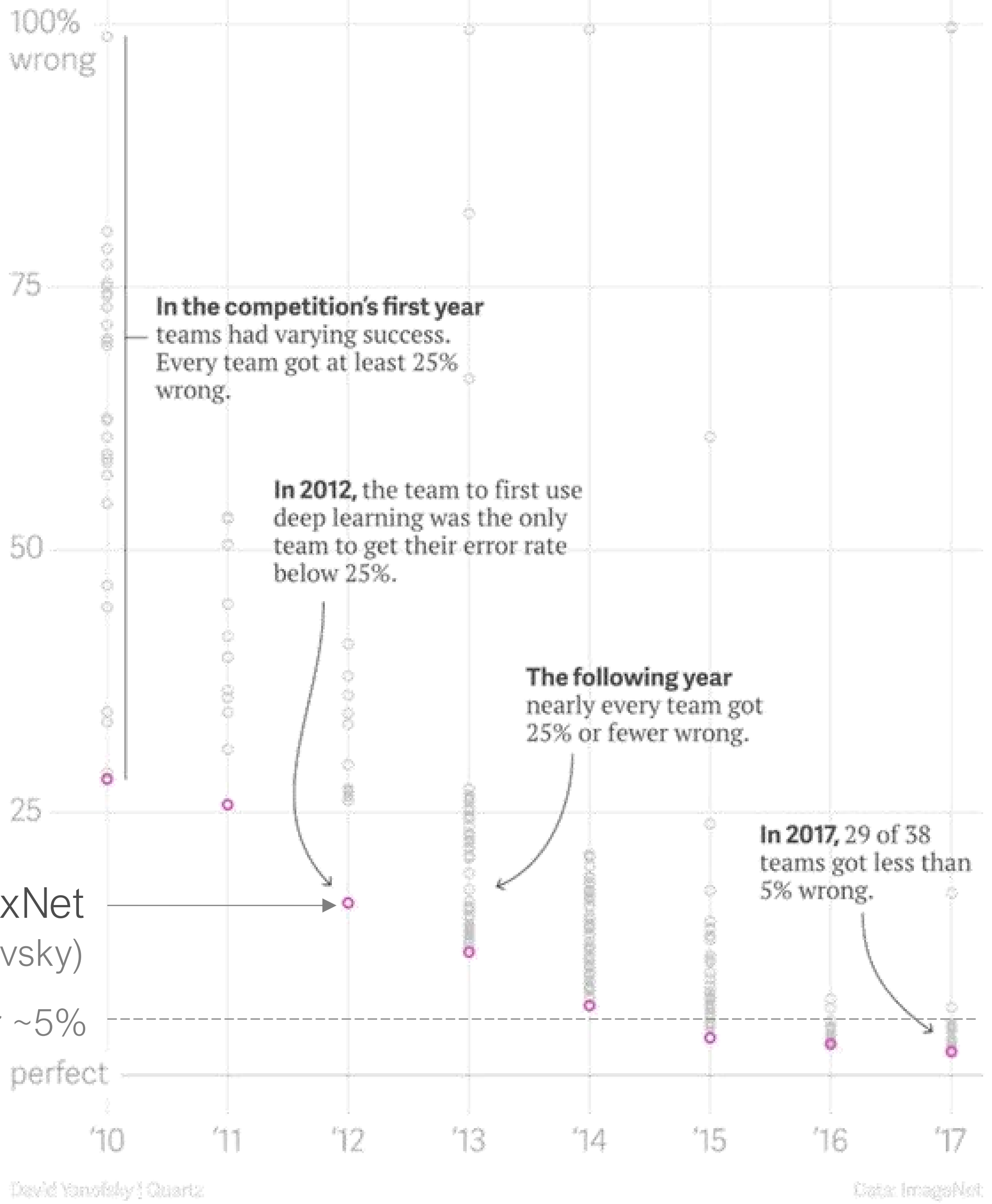
U-Net (2015)
SegNet (2016)
DeepLab (2017)
FCN (2016)

ImageNet Large Scale Visual Recognition Challenge (ILSVRC)

Fei-Fei Li et al. 2010 ([link]())

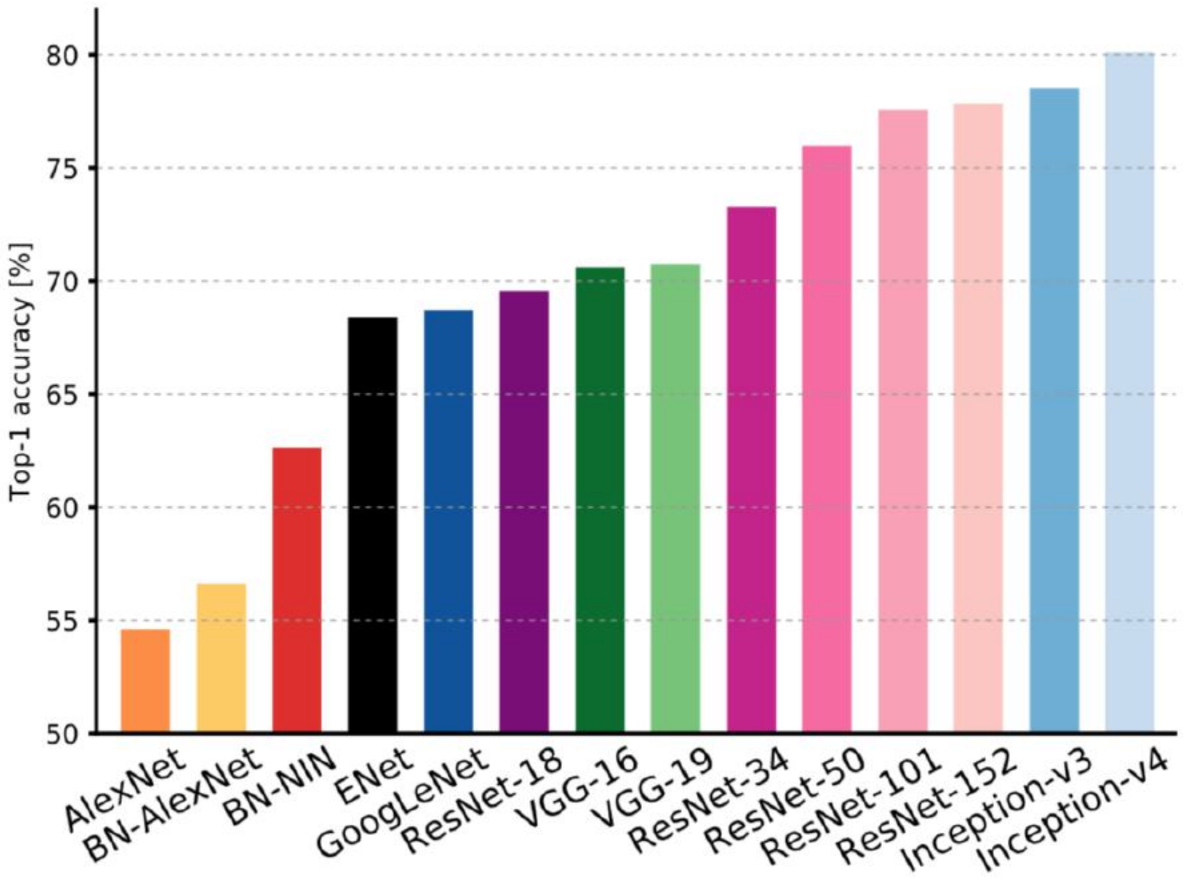
Competition at:
Conference on Computer Vision and Pattern Recognition (CVPR)

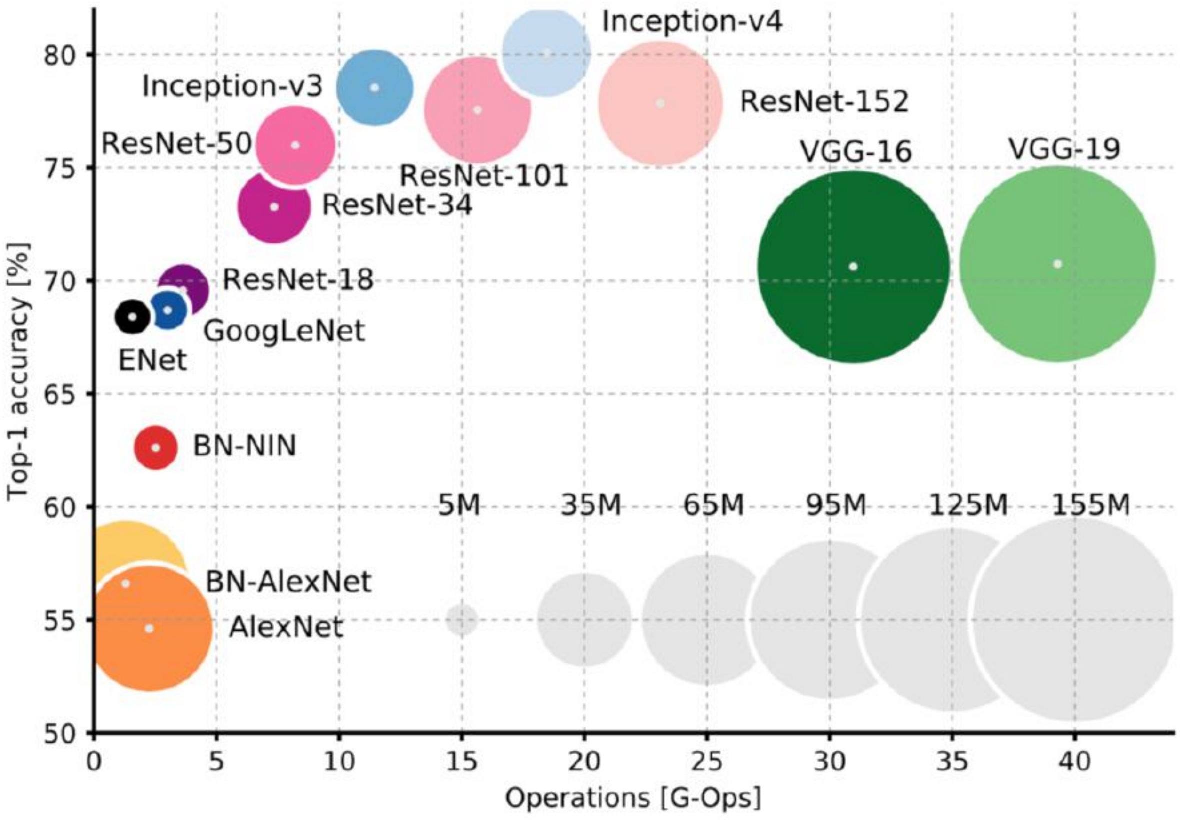
USED FOR MODEL PRETRAINING



Source: Quartz, [link]()

Deep Learning Models Compared





Models compared for ImageNet

Many of these models are available through Keras (link)

A. Canziani, E. Culurciello and A. Paszke, "Evaluation of neural network architectures for embedded systems," *2017 IEEE International Symposium on Circuits and Systems (ISCAS)*, Baltimore, MD, 2017, pp. 1-4.

Deep learning frameworks

Tensorflow ([link]())
Framework for implementing graphical models, such as neural networks

Keras ([link]())
Wrapper for Tensorflow to make coding easier: higher level and excellent API

PyTorch ([link]())
Framework for implementing graphical models, such as neural networks

KERAS DEMO

Generative Adversarial Networks

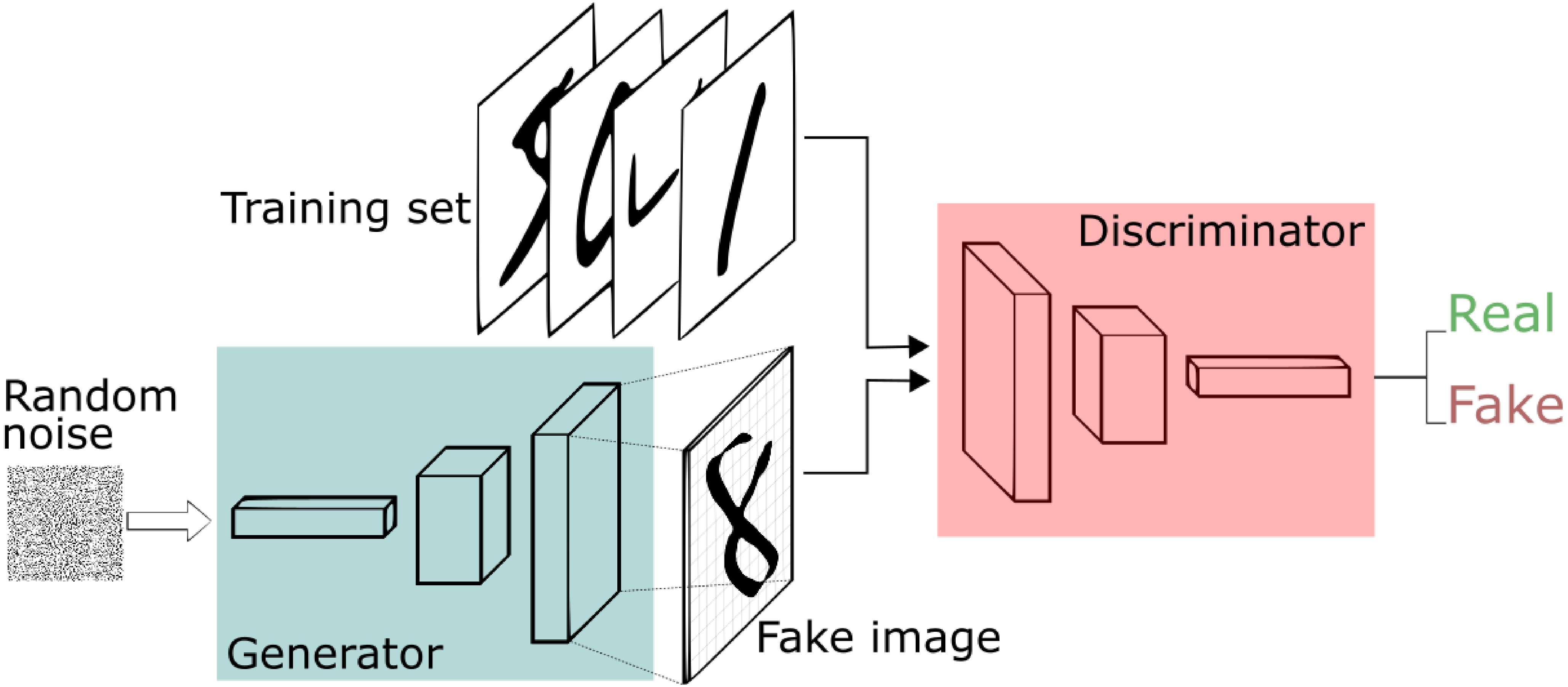


Image from: https://skymind.ai/wiki/generative-adversarial-network-gan