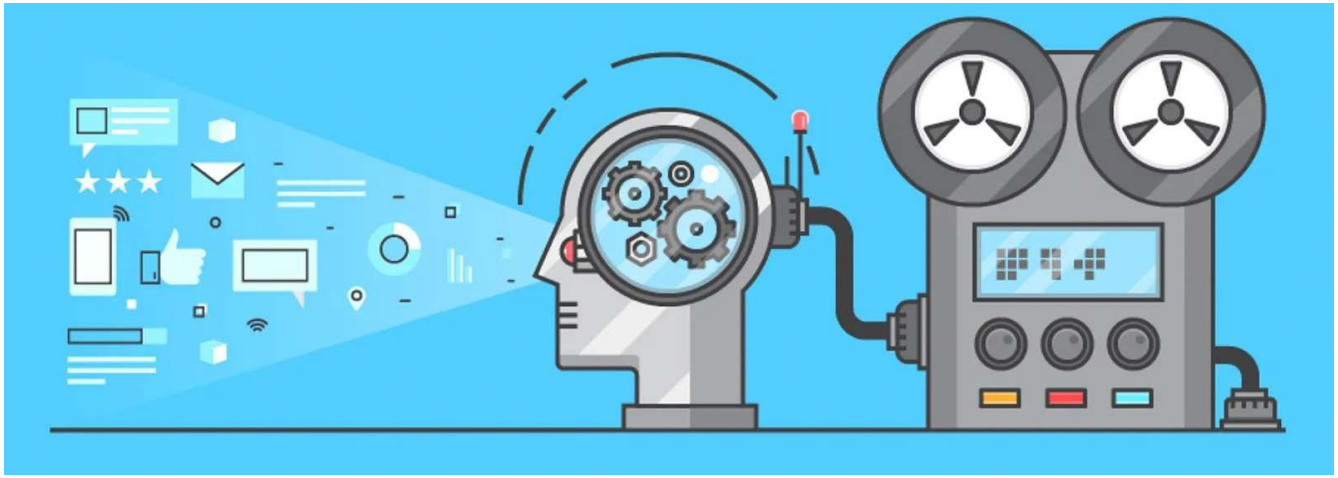
Deep Learning and Neural Networks

Topic 5:

Convolutional Neural Networks

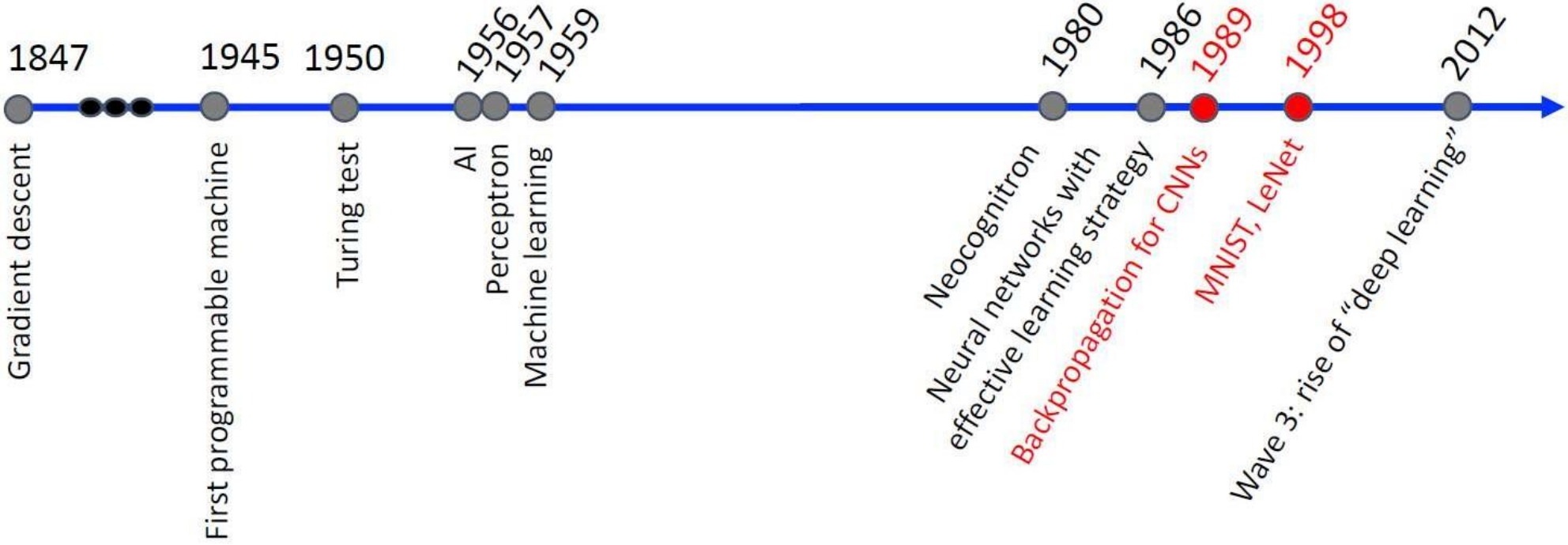


Ricardo Abel Espinosa Loera, McS

Researcher in DL & Computer Vision

BACKGROUND
Neural Networks for images

Historical Context: Inspiration




1
Background

Which are the different tasks in visual object recognition?

Image from [Fei Fei Li, 2011]

classification

Does this image contain a car? [yes/no]

detection

Does this image contains a car? [yes? where]

detection+

What objects does this image contain? [where]

segmentation

What is the exact location? [object contour]

semantic interpretation

Semantic and geometric attributes estimation

Which are the different tasks in visual object recognition?

Image from [Fei Fei Li, 2011]

classification

Does this image contain a car? [yes/no]

detection

Does this image contains a car? [yes? where]

detection+

What objects does this image contain? [where]

segmentation

What is the exact location? [object contour]

semantic interpretation

Semantic and geometric attributes estimation

Which are the different tasks in visual object recognition?

Image from [Fei Fei Li, 2011]

classification

Does this image contains a car? [yes/no]

detection

Does this image contain a car? [yes? where]

detection+

What objects does this image contain? [where]

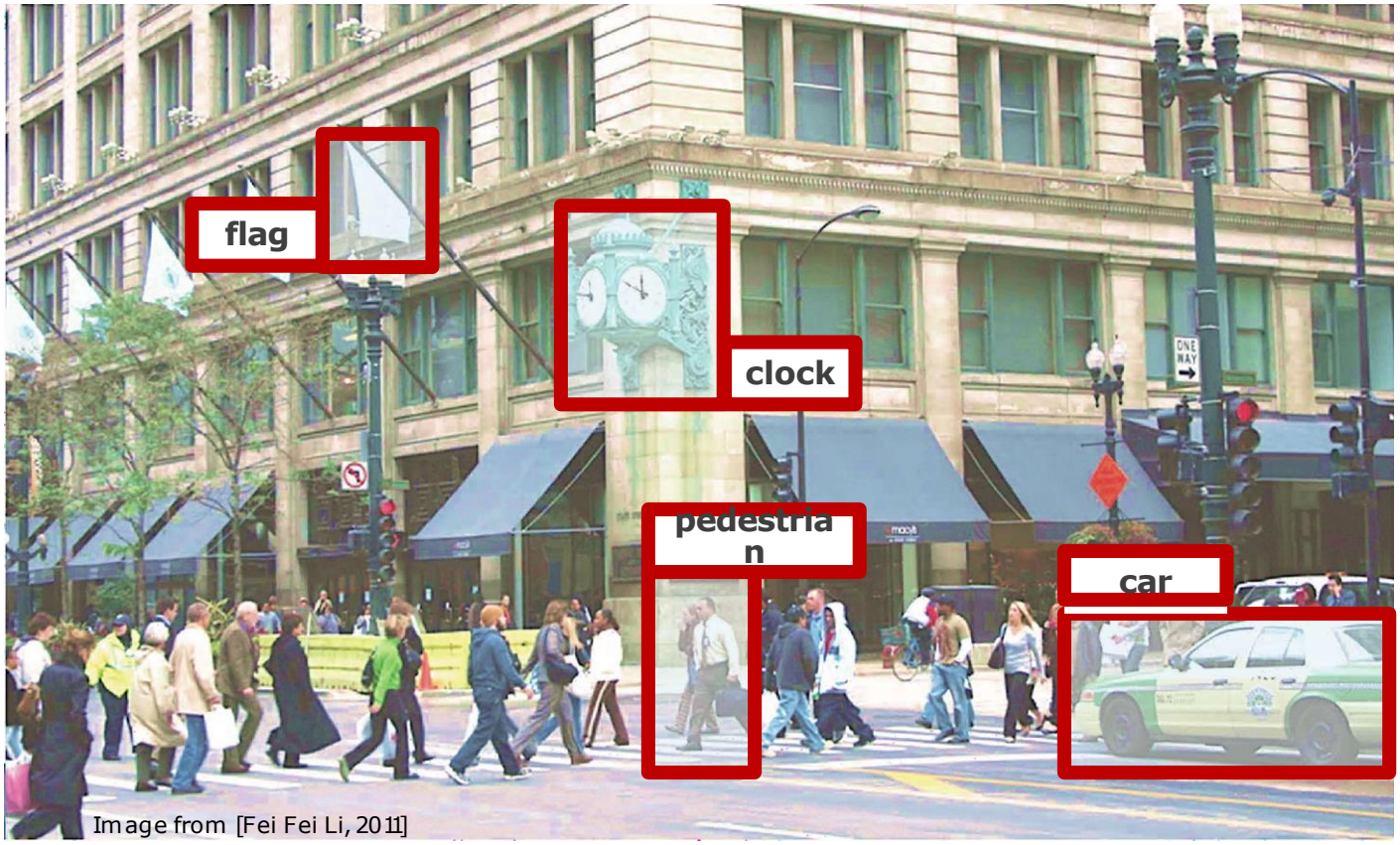
segmentation

What is the exact location? [object contour]

semantic interpretation

Semantic and geometric attributes estimation

Which are the different tasks in visual object recognition?



classification
Does this image contain a car? [yes/no]

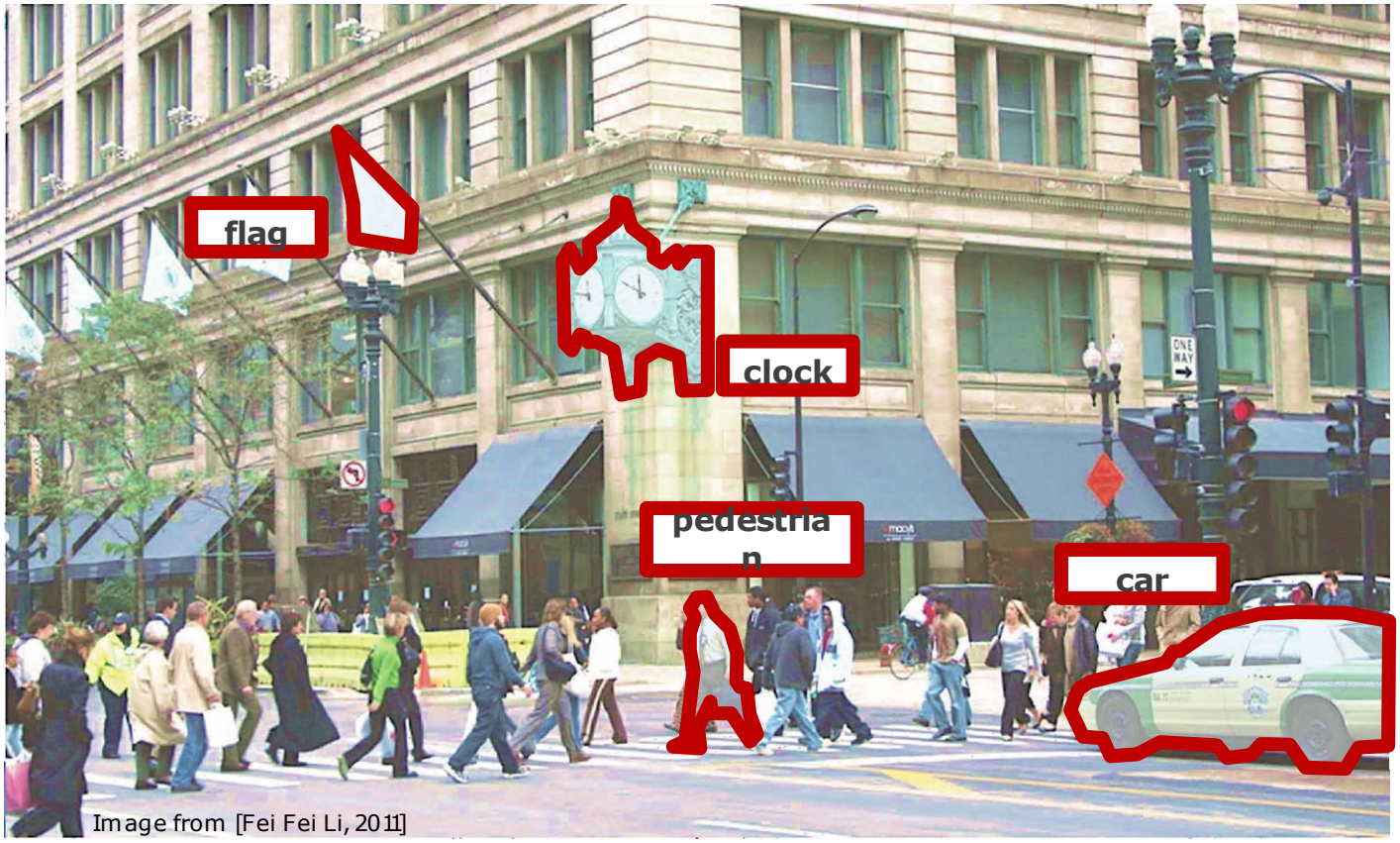
detection
Does this image contains a car? [yes? where]

detection+
What objects does this image contain? [where]

segmentation
What is the exact location? [object contour]

semantic interpretation
Semantic and geometric attributes estimation

Which are the different tasks in visual object recognition?



classification
Does this image contain a car? [yes/no]

detection
Does this image contains a car? [yes? where]

detection+
What objects does this image contain? [where]

segmentation
What is the exact location? [object contour]

semantic interpretation
Semantic and geometric attributes estimation

Which are the different tasks in visual object recognition?

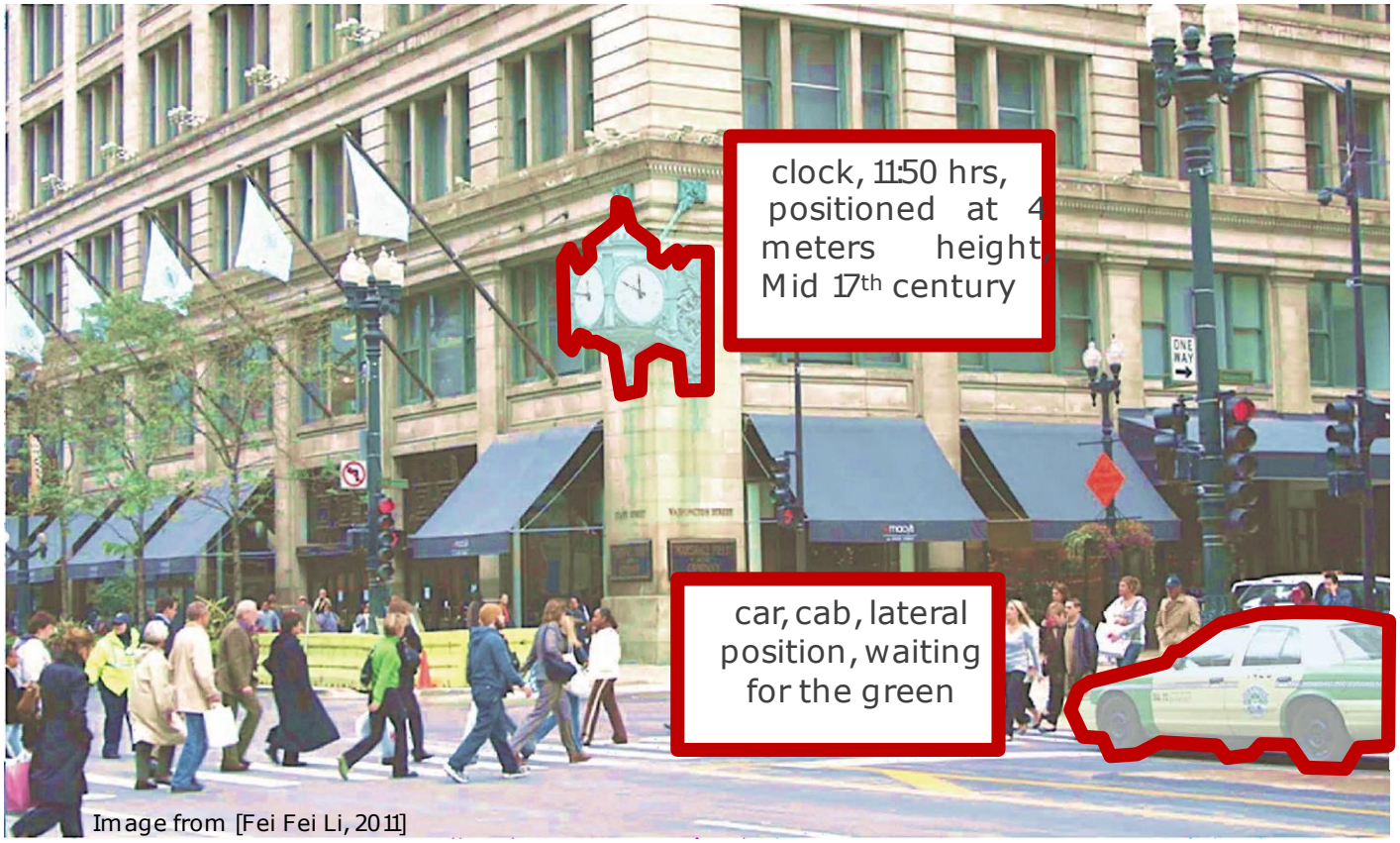


Image from [Fei Fei Li, 2011]

classification
Does this image contain a car? [yes/no]

detection
Does this image contains a car? [yes? where]

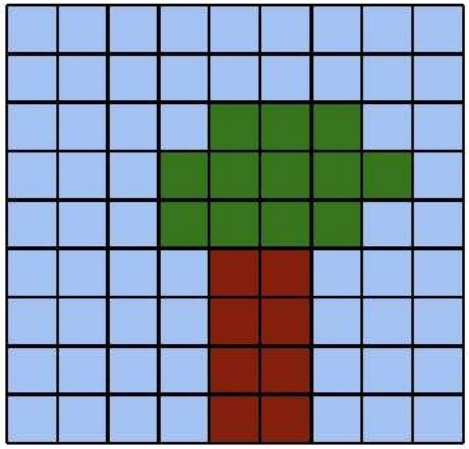
detection+
What objects does this image contain? [where]

segmentation
What is the exact location? [object contour]

semantic interpretation
Semantic and geometric attributes estimation

BACKGROUND

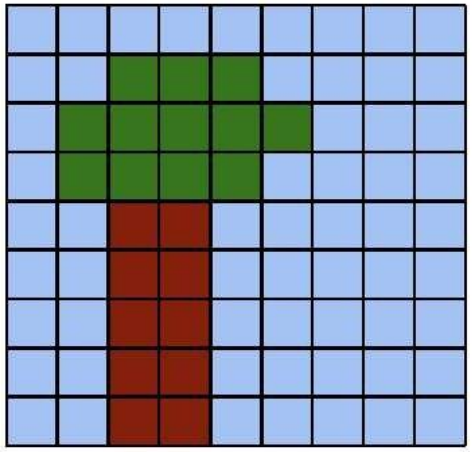
Neural Networks for images



A digital image is a 2D **grid of pixels**.

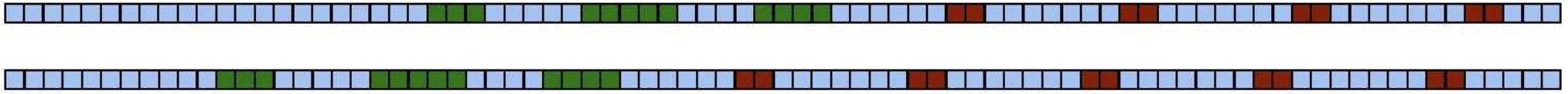
A neural network expects a **vector of numbers** as input.

BACKGROUND
Neural Networks for images



A digital image is a 2D **grid of pixels.**

A neural network expects a **vector of numbers** as input.



BACKGROUND

Locality and traslation invariance

Locality: nearby pixels are more strongly correlated

Translation invariance: meaningful patterns can occur anywhere in the image

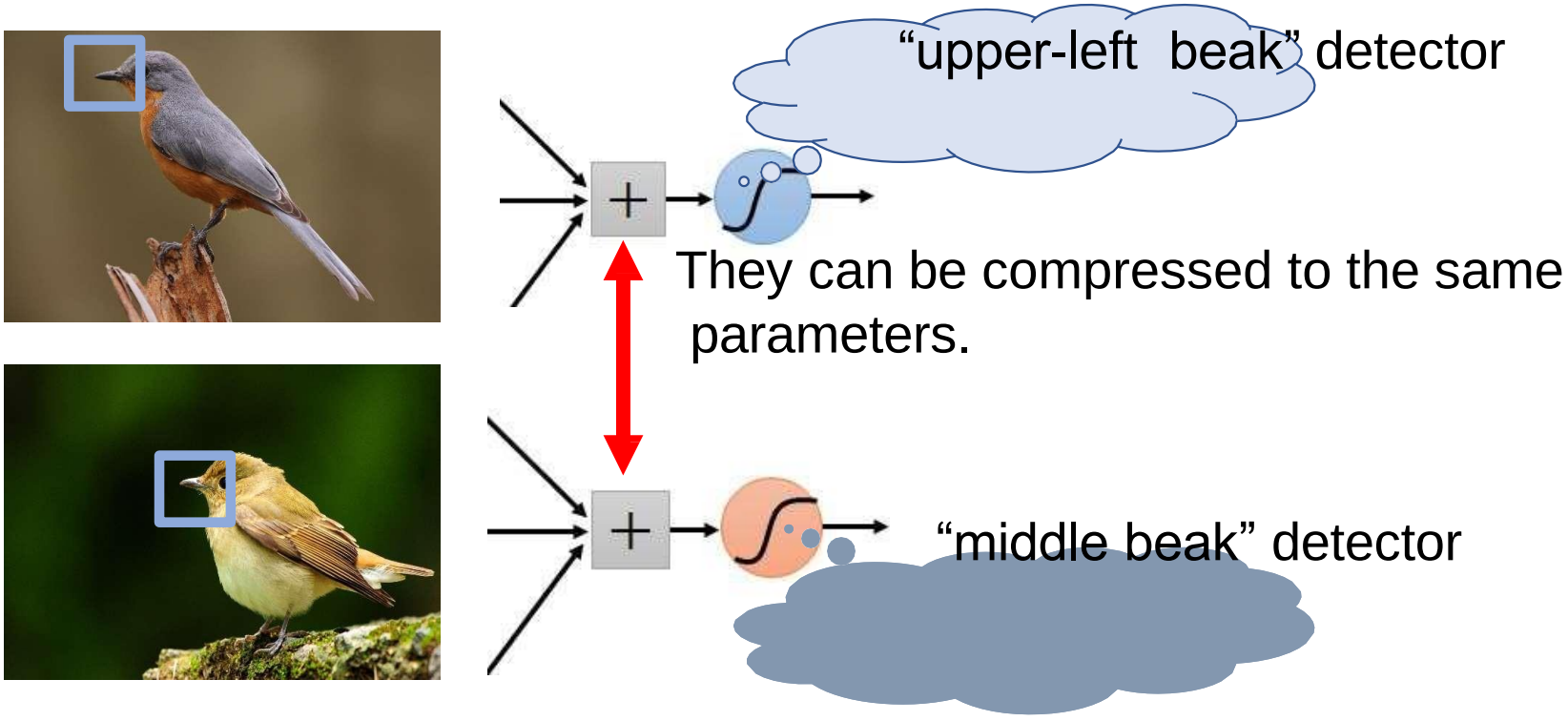
BACKGROUND
Locality and traslation invariance

- Some patterns are much smaller than the whole image

Can represent a small region with fewer parameters

BACKGROUND
Locality and traslation invariance

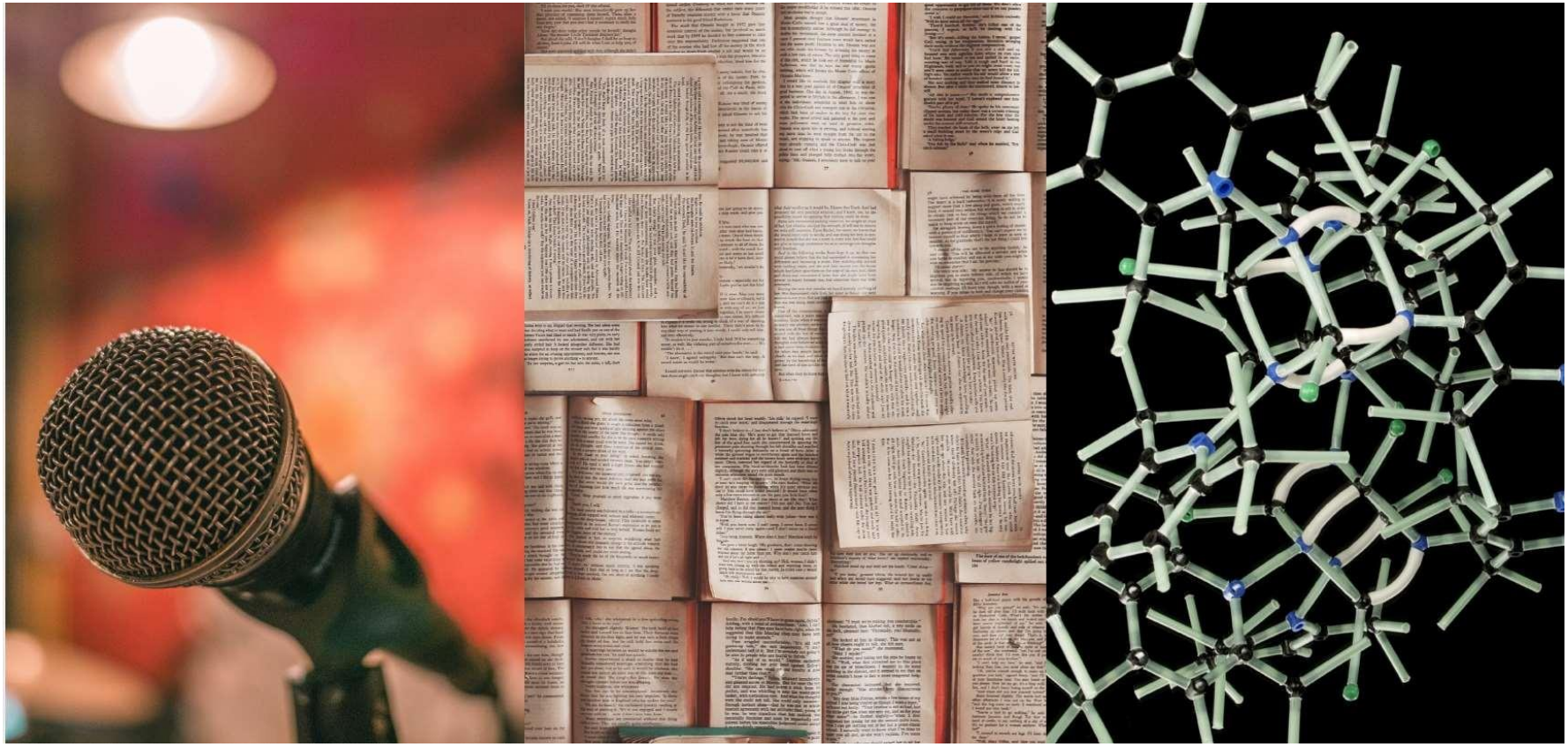
What about training a lot of such “small” detectors and each detector must “move around”.



BACKGROUND
Locality and traslation invariance

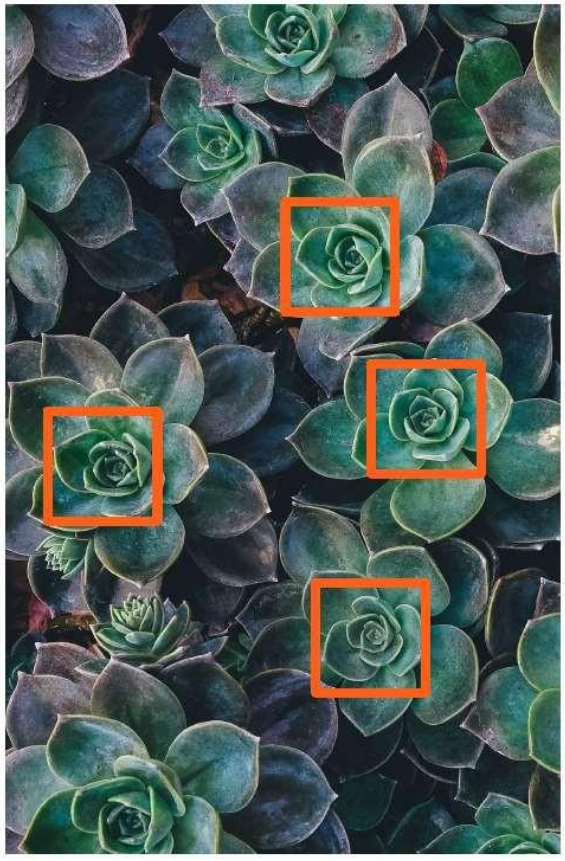
BACKGROUND

Locality and traslation invariance



BACKGROUND

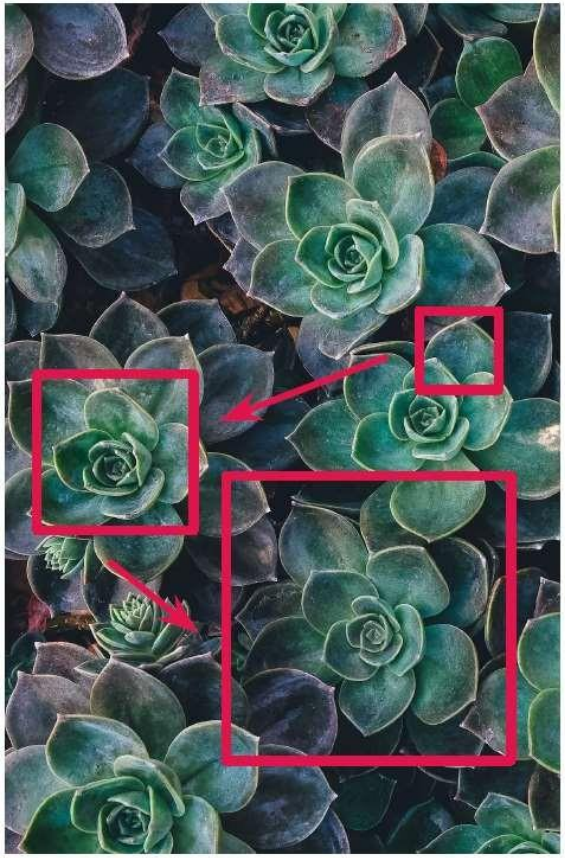
Taking Advantage of topological structure



Weight sharing: use the same network parameters to detect local patterns at many locations in the image

BACKGROUND

Taking Advantage of topological structure



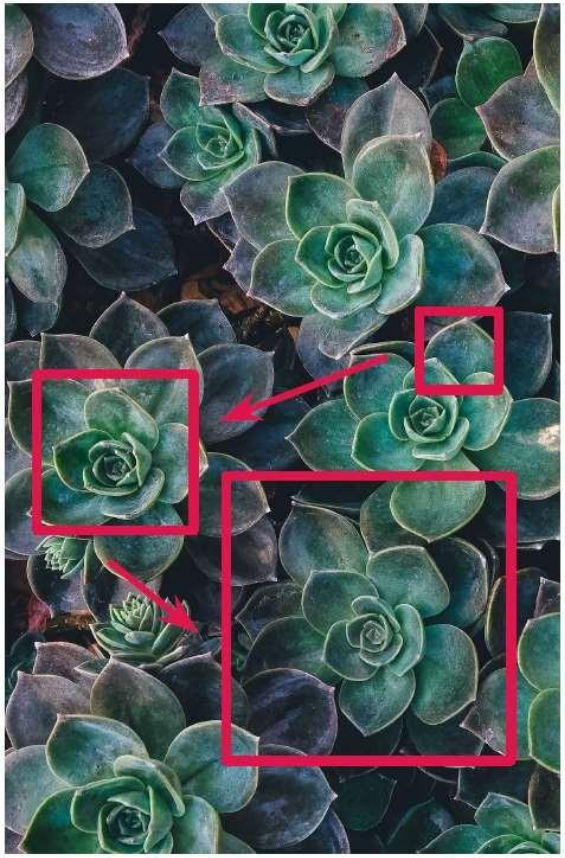
Weight sharing: use the same network parameters to detect local patterns at many locations in the image

Hierarchy: local low-level features are composed into larger, more abstract features

edges and textures → object parts → objects

BACKGROUND

Taking Advantage of topological structure



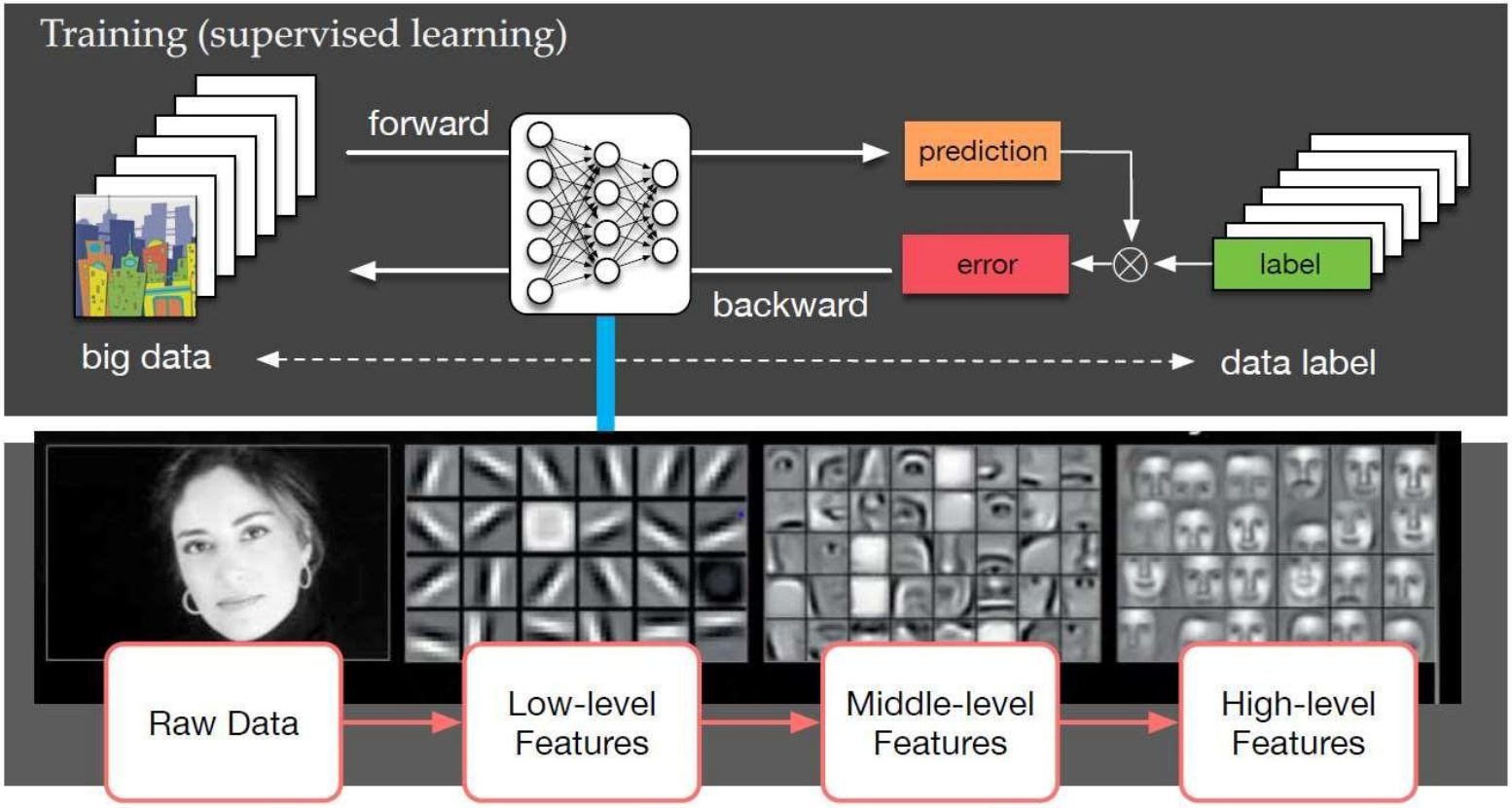
Weight sharing: use the same network parameters to detect local patterns at many locations in the image

Hierarchy: local low-level features are composed into larger, more abstract features

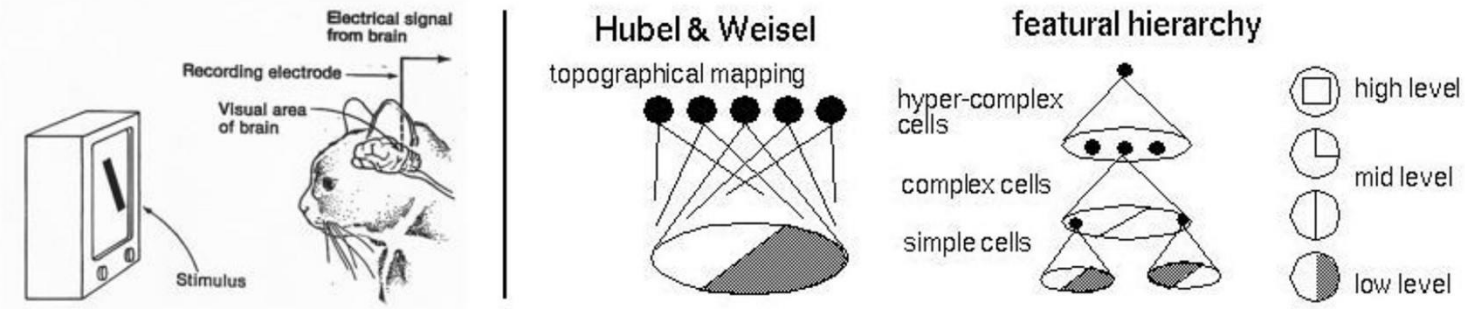
edges and textures → object parts → objects

BACKGROUND

Taking Advantage of topological structure



BACKGROUND
experiments of Hubel and Wiesel

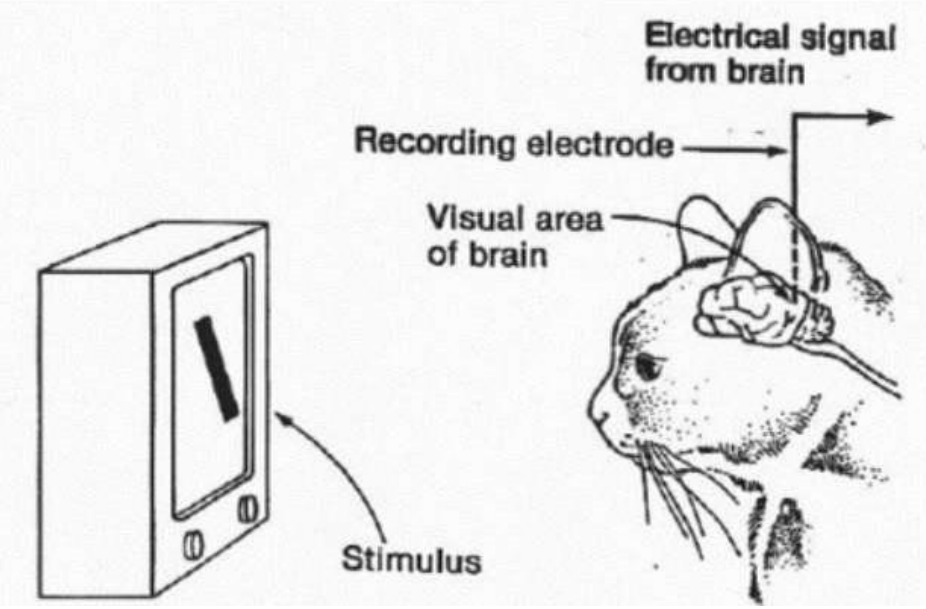


D.H. Hubel and T.N. Wiesel (1962) Receptive fields, binocular interaction and functional architecture in the cat's visual cortex *J Physiol.* 160(1): 106154.2.

BACKGROUND
experiments of Hubel and Wiesel

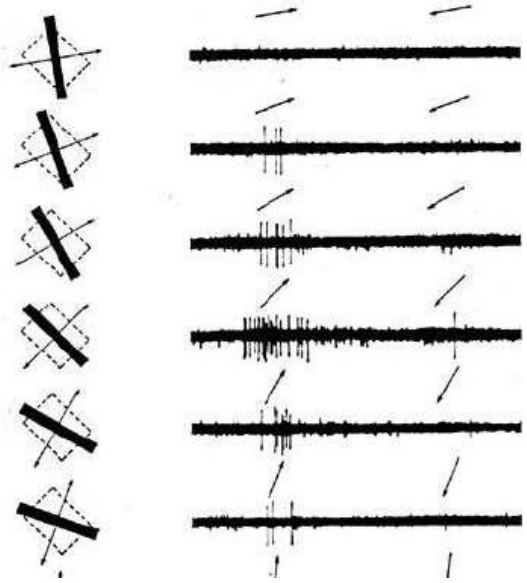
BACKGROUND
experiments of Hubel and Wiesel

Experiment Set-up:

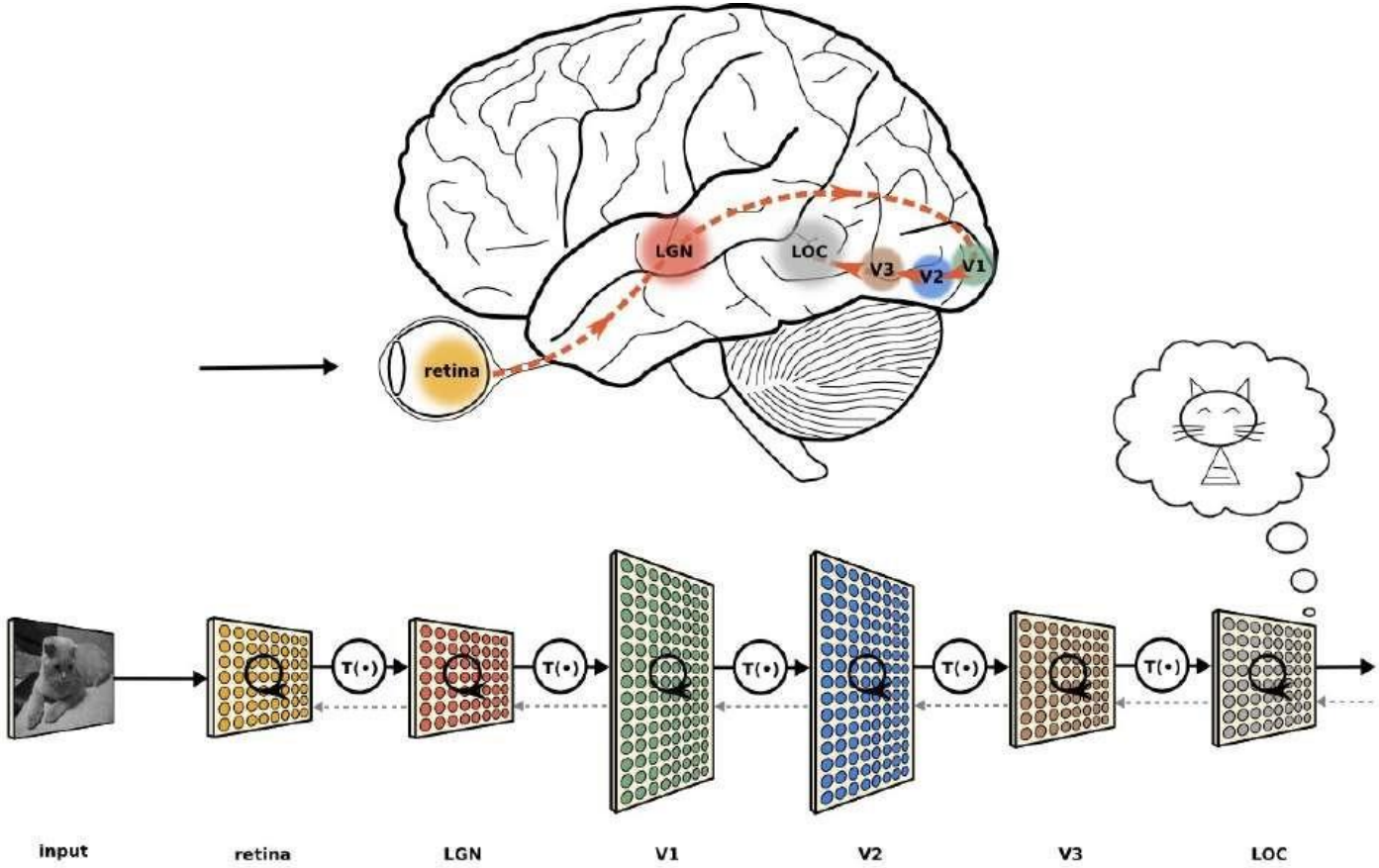


Key Finding: initial neurons responded strongly only when light was shown in certain orientations

V1 physiology:
direction
selectivity

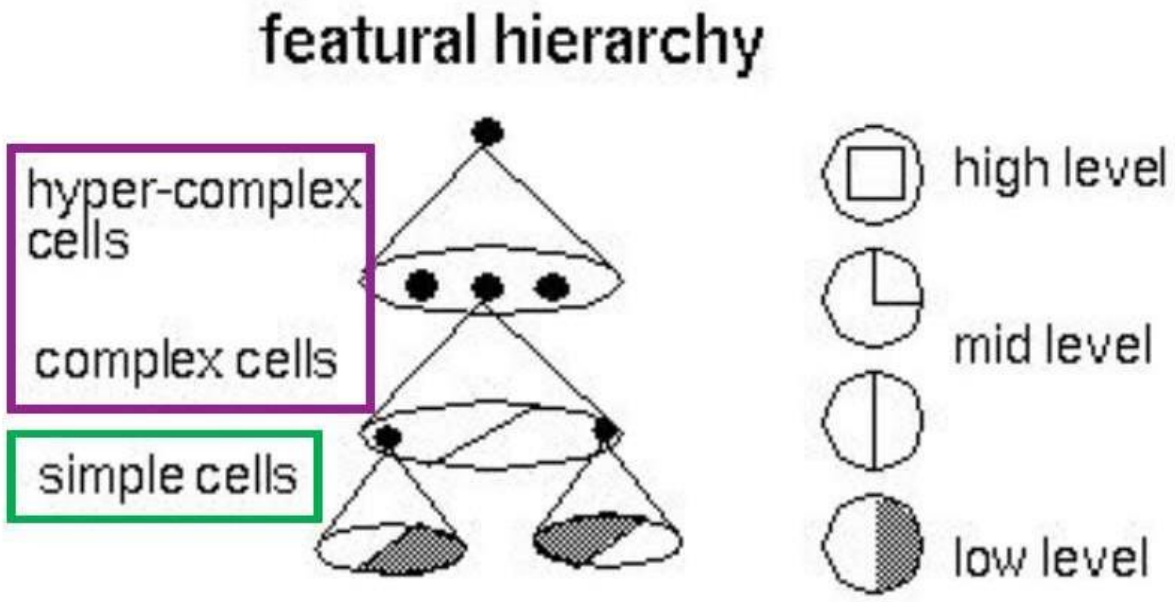


BACKGROUND
experiments of Hubel and Wiesel

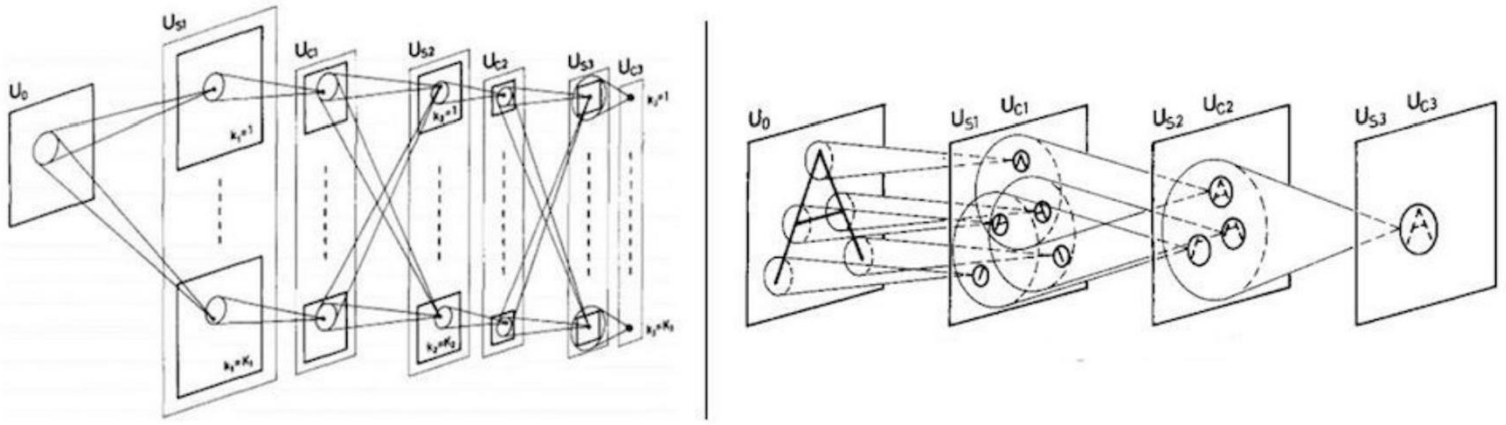


BACKGROUND
experiments of Hubel and Wiesel

Key Idea: cells are organized as a hierarchy of feature detectors, with higher level features responding to patterns of activation in lower level cells

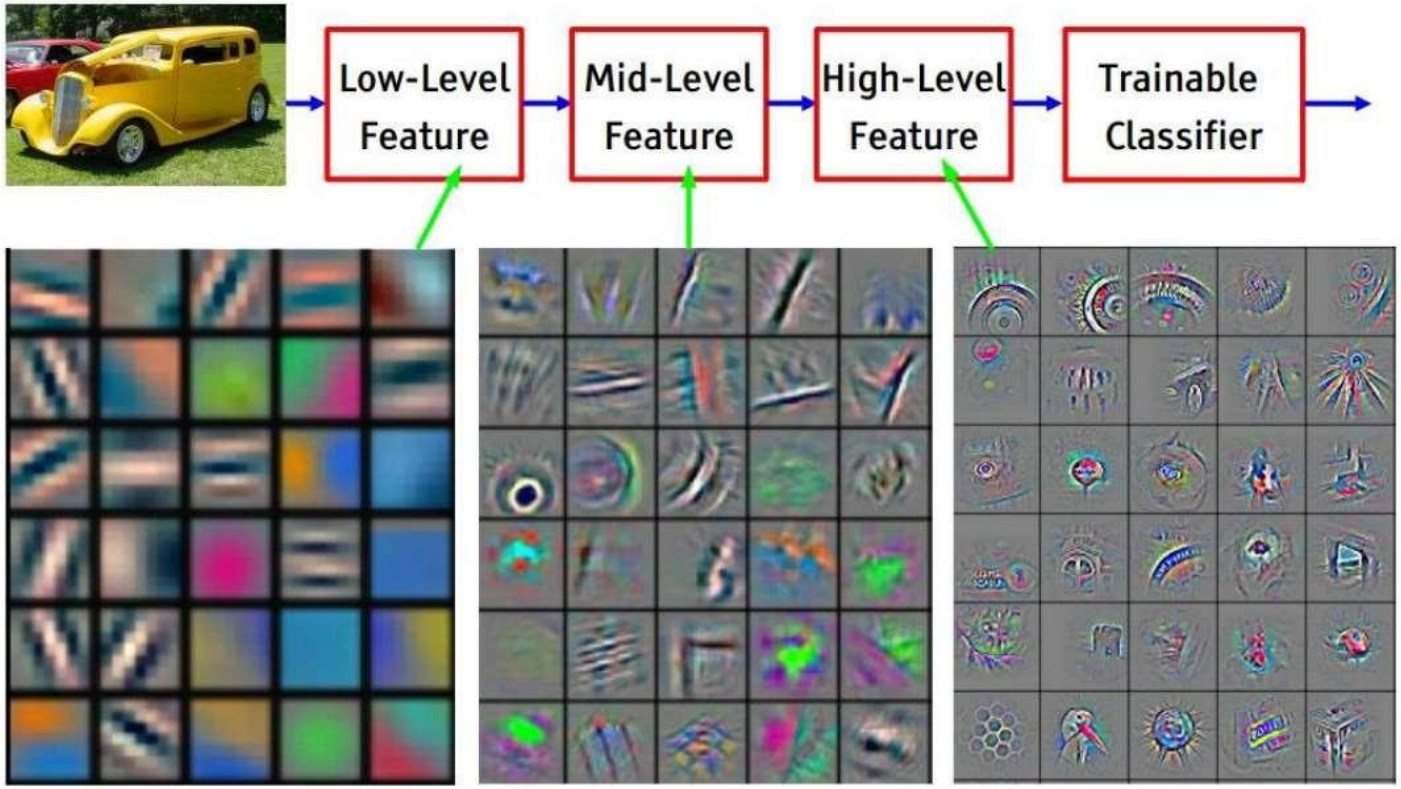


BACKGROUND
neocognitron



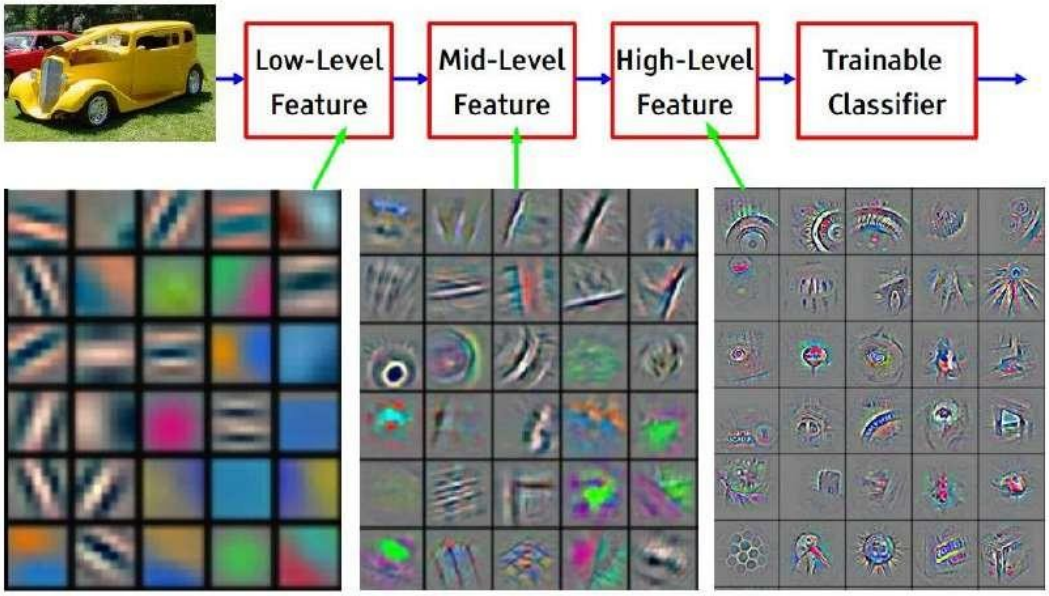
K. Fukushima (1980) Neocognitron: A Self-organizing Neural Network Model for a Mechanism of Pattern Recognition Unaffected by Shift in Position. *Biol. Cybernetics* 36, 193–202

BACKGROUND
LeNet

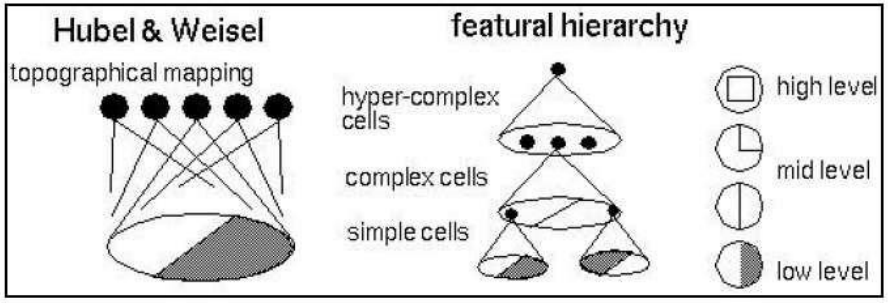


Feature visualization of convolutional net trained on ImageNet from [Zeiler & Fergus 2013]

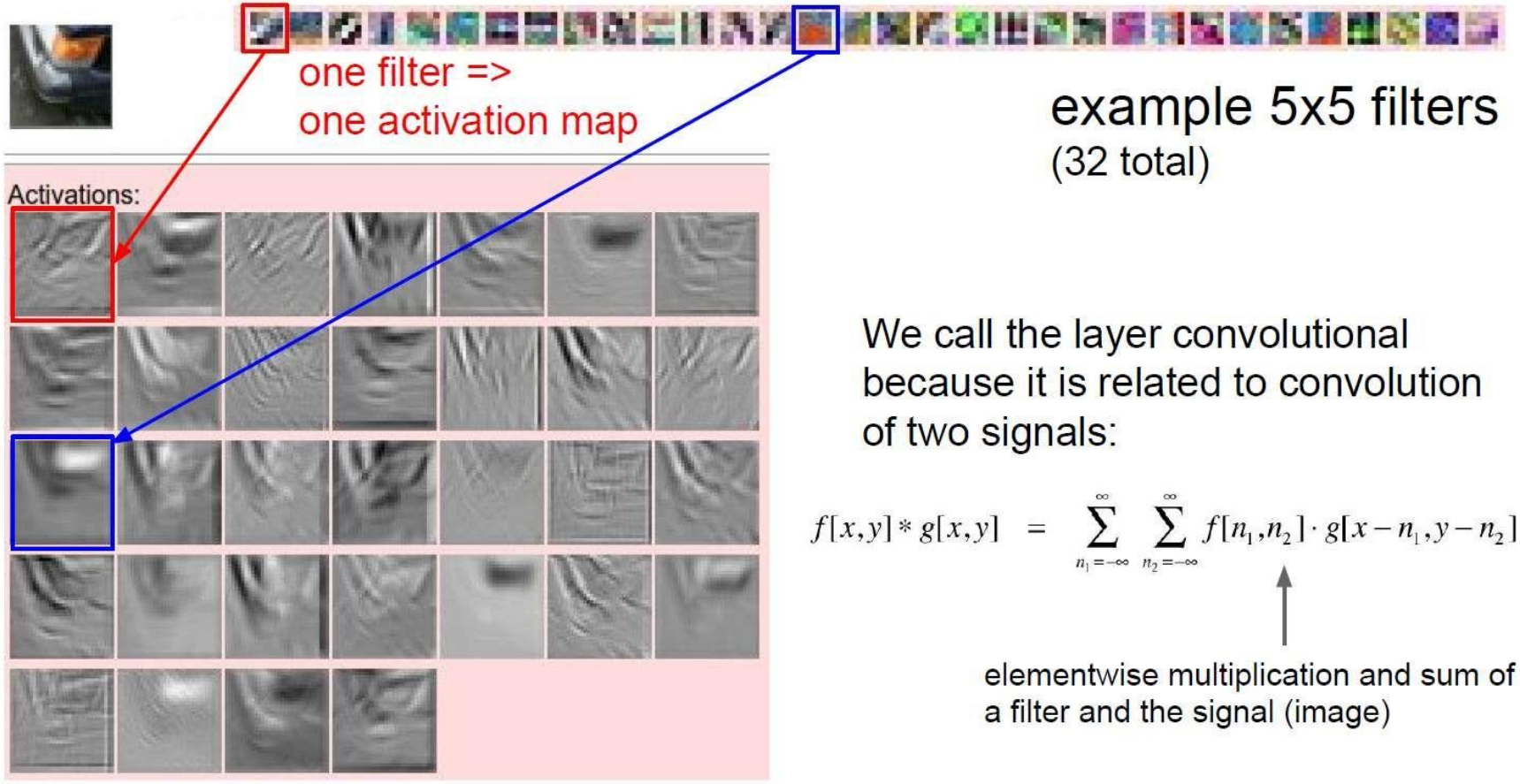
BACKGROUND
LeNet



Feature visualization of convolutional net trained on ImageNet from [Zeiler & Fergus 2013]



BACKGROUND
LeNet



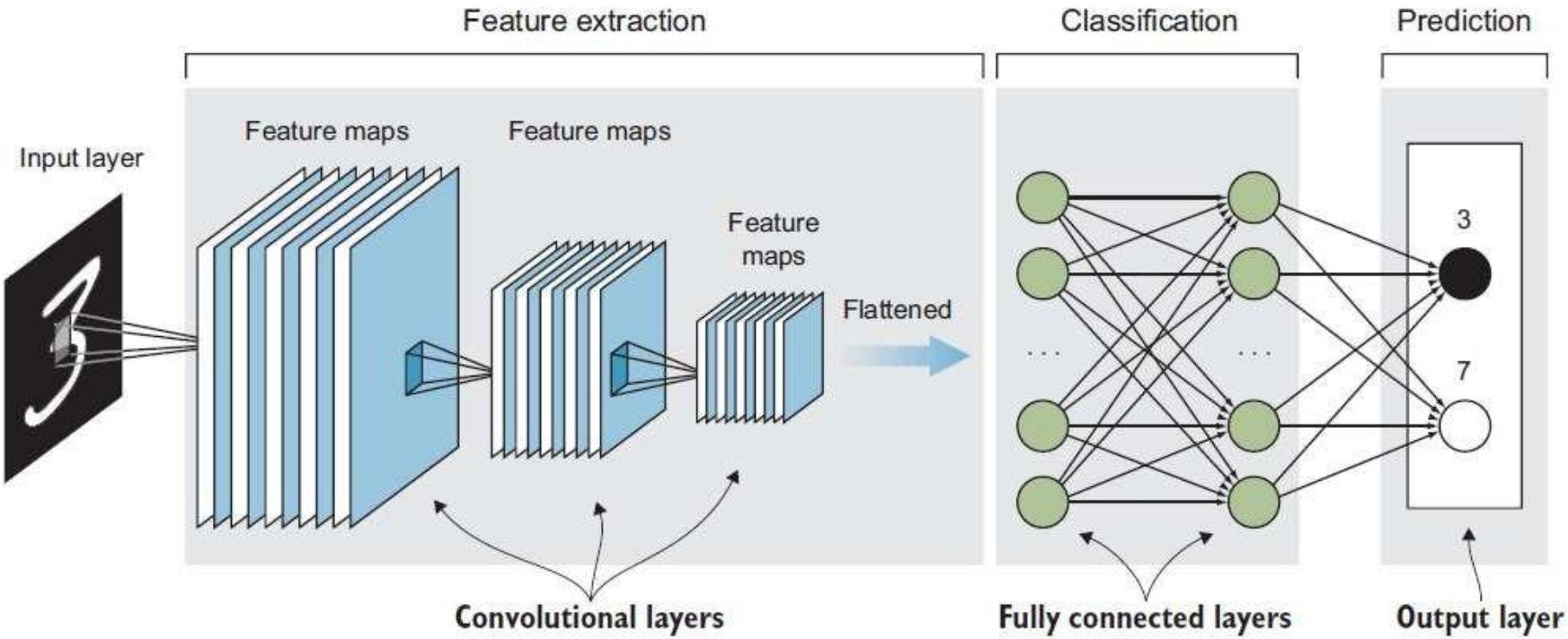
example 5x5 filters
(32 total)

We call the layer convolutional because it is related to convolution of two signals:

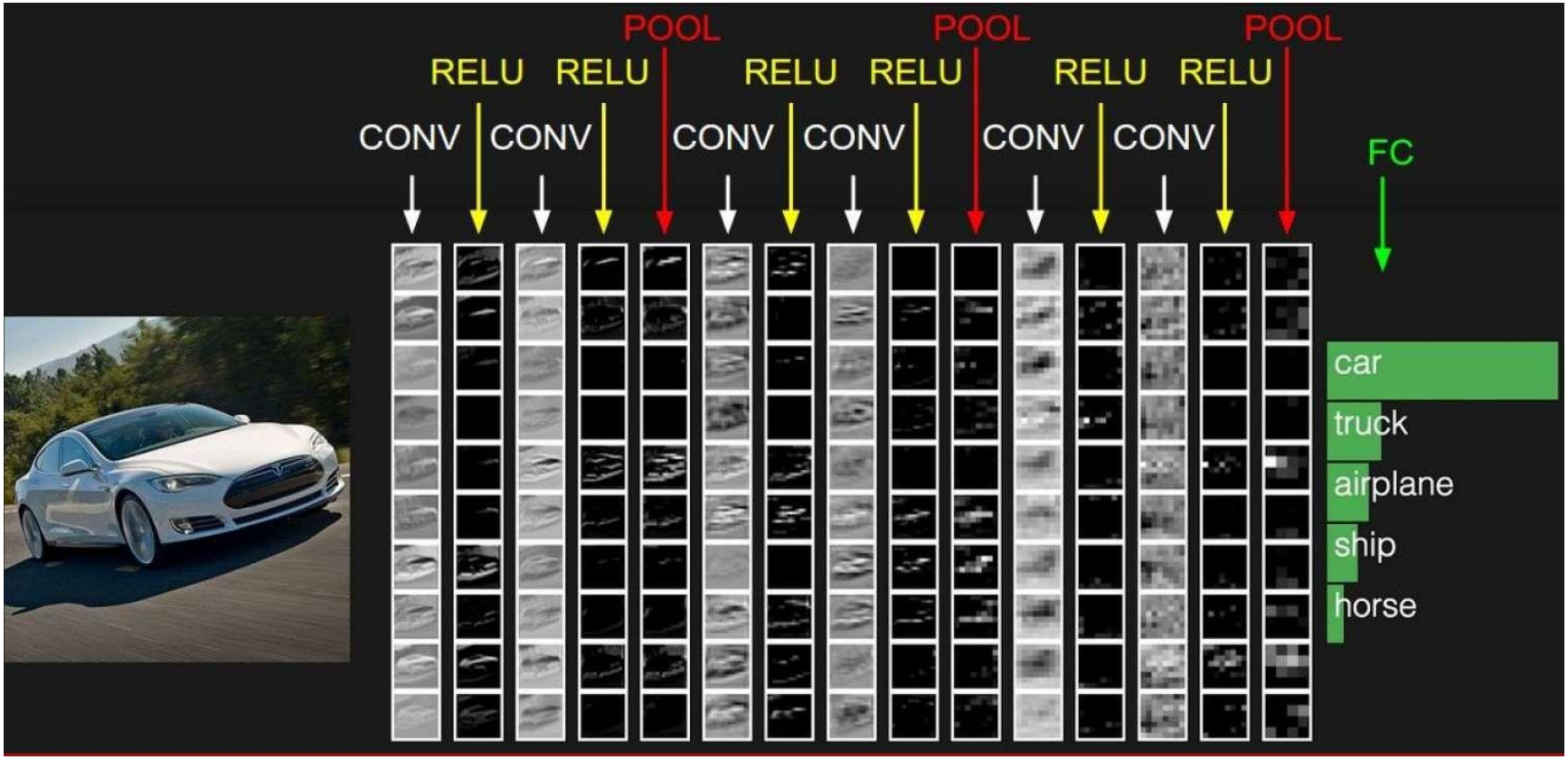
$$f[x,y] * g[x,y] = \sum_{n_1=-\infty}^{\infty} \sum_{n_2=-\infty}^{\infty} f[n_1,n_2] \cdot g[x-n_1, y-n_2]$$

elementwise multiplication and sum of a filter and the signal (image)

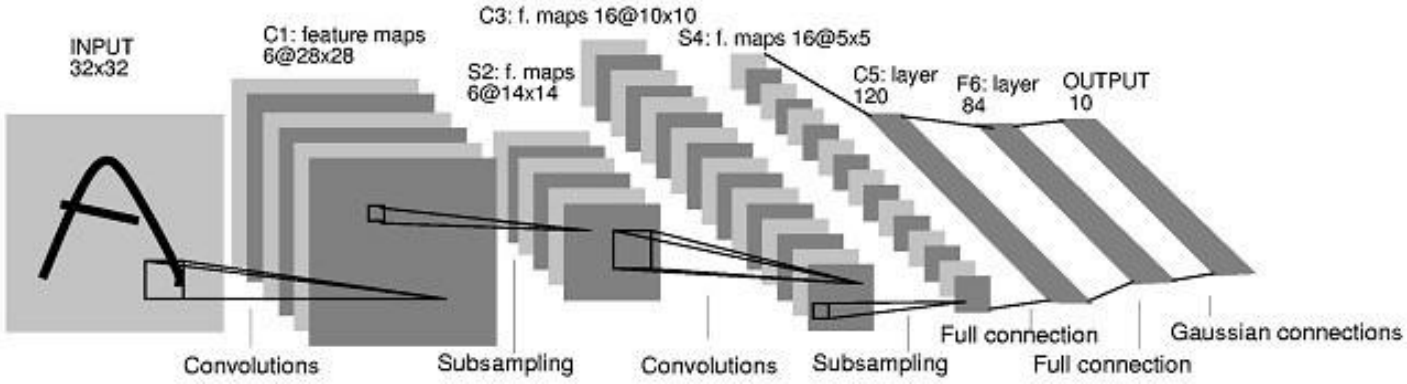
BACKGROUND
LeNet



BACKGROUND
LeNet



BACKGROUND
LeNet



Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner (1998) Gradient-based learning applied to document recognition, *Proc. IEEE* 86(11): 22782324

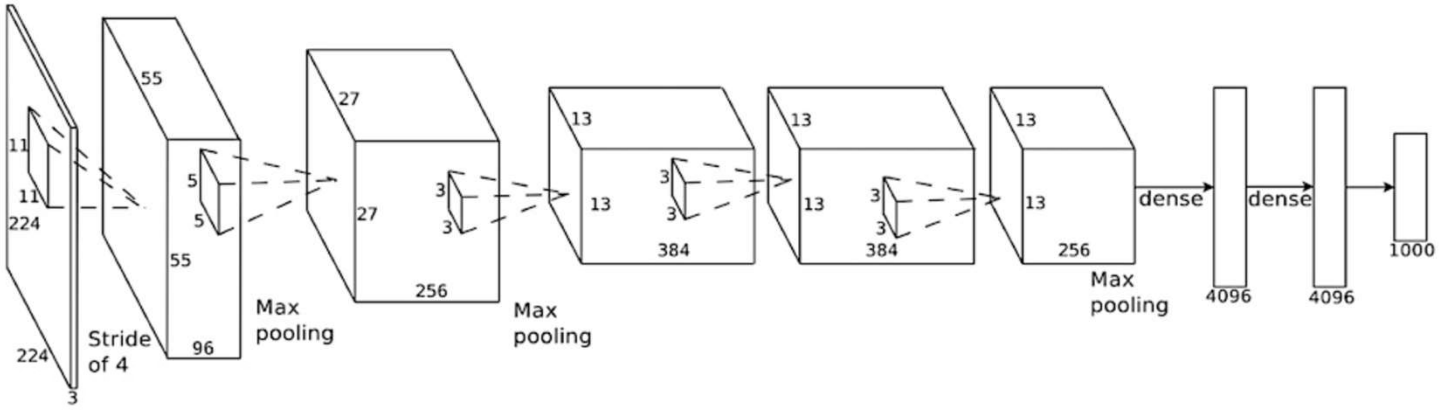
BACKGROUND
LeNet

LeNet-5 model summary

Layer (type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 28, 28, 6)	156
average_pooling2d_1 (Average	(None, 14, 14, 6)	0
conv2d_2 (Conv2D)	(None, 10, 10, 16)	2416
average_pooling2d_2 (Average	(None, 5, 5, 16)	0
conv2d_3 (Conv2D)	(None, 1, 1, 120)	48120
flatten_1 (Flatten)	(None, 120)	0
dense_1 (Dense)	(None, 84)	10164
dense_2 (Dense)	(None, 10)	850

Total params: 61,706
Trainable params: 61,706
Non-trainable params: 0

BACKGROUND
AlexNet



A. Krizhevsky, I. Sutskever, G. E. Hinton (2012) ImageNet Classification with Deep Convolutional Neural Networks. Advances in Neural Information Processing Systems 25

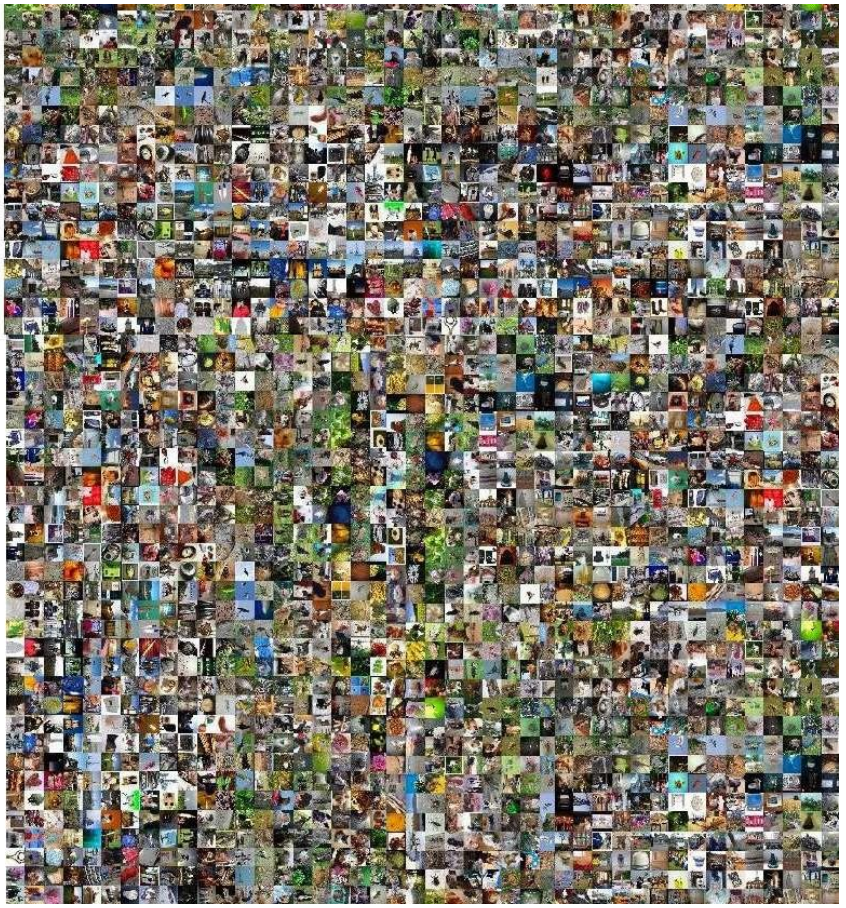
BACKGROUND
Data Drives Research

The ImageNet challenge

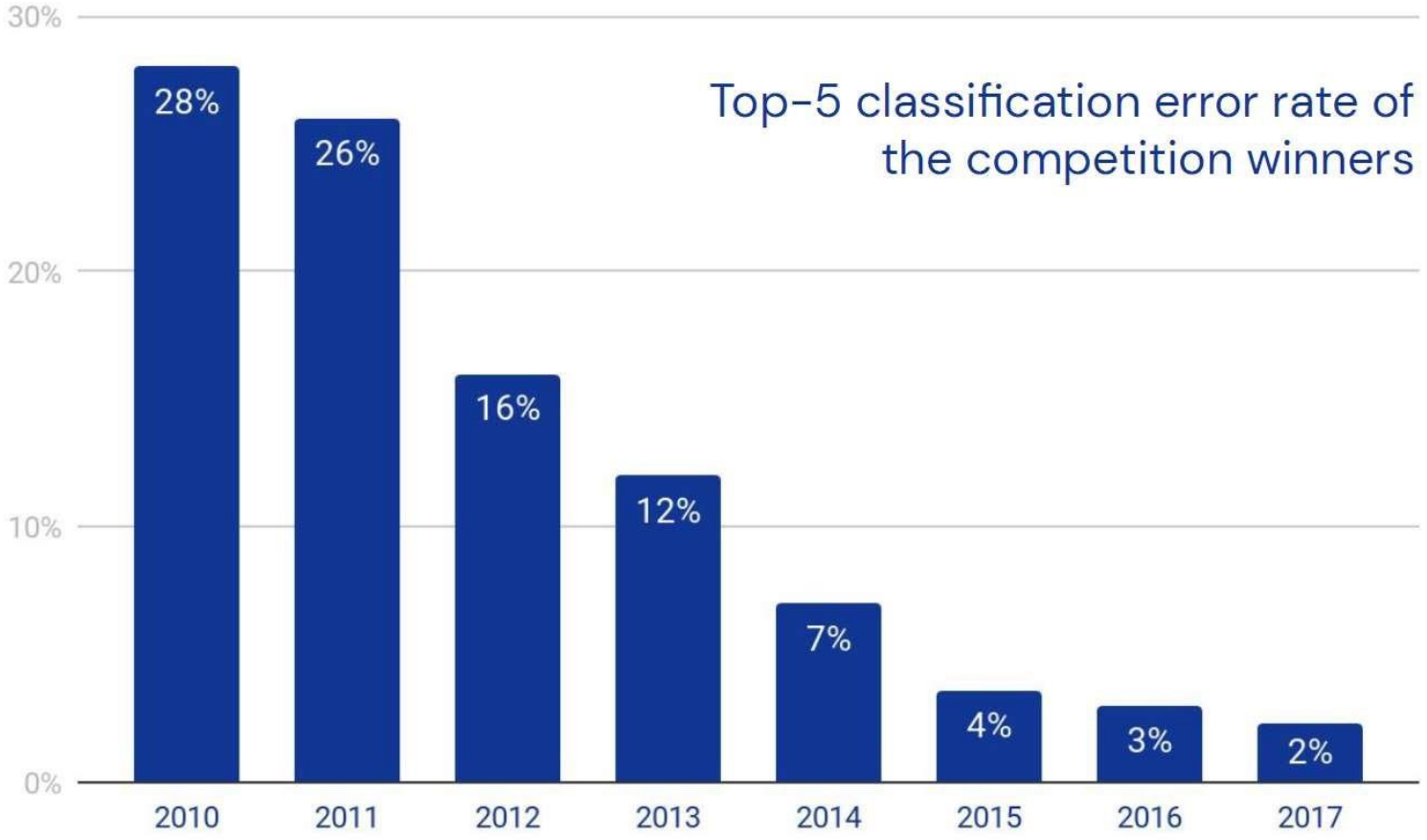
- Major computer vision benchmark
- Ran from 2010 to 2017
- 1.4M images, 1000 classes
- Image classification

Want to learn more?

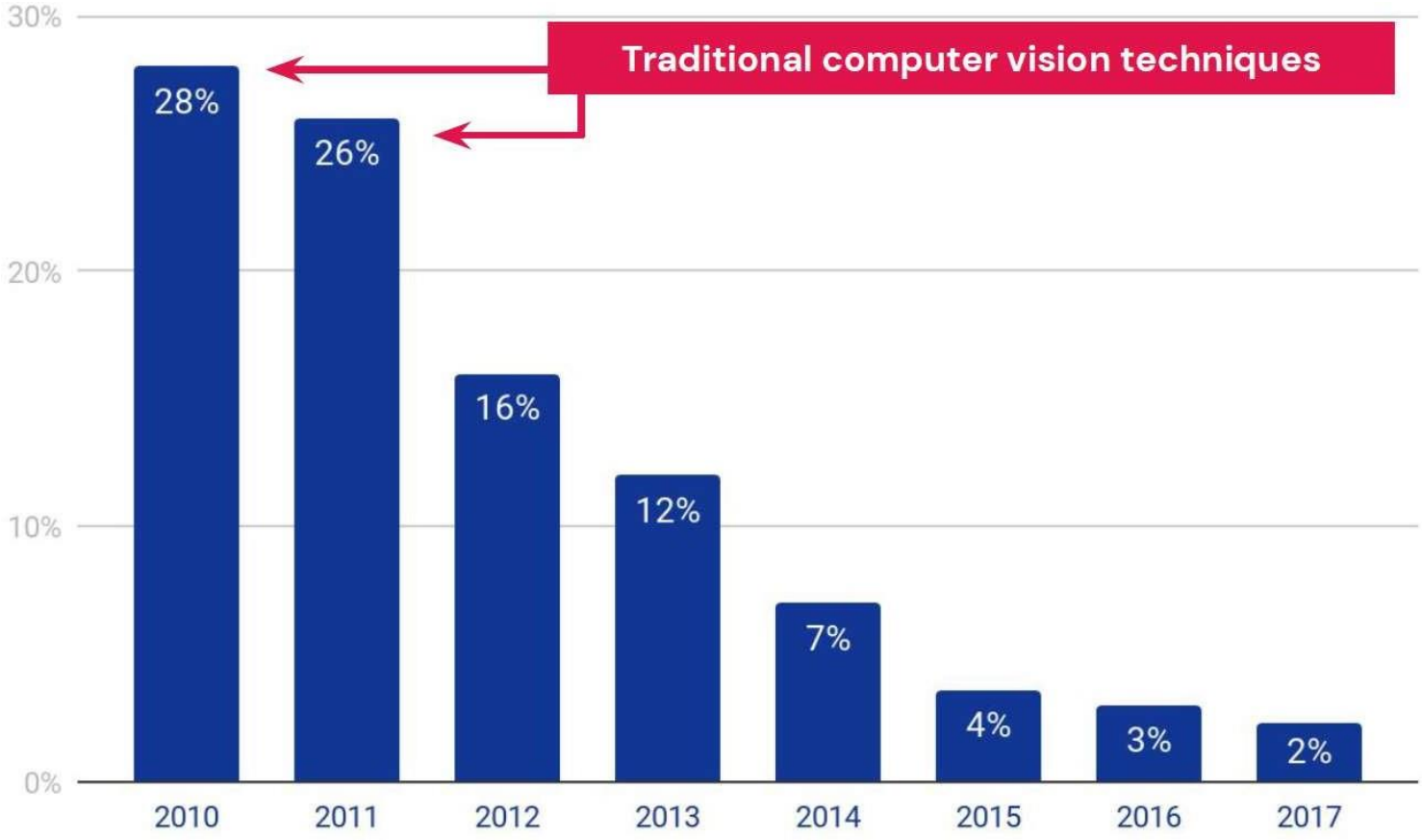
Russakovsky, Olga et al. ***ImageNet Large Scale Visual Recognition Challenge*** International Journal of Computer Vision 115.3 (**2015**)



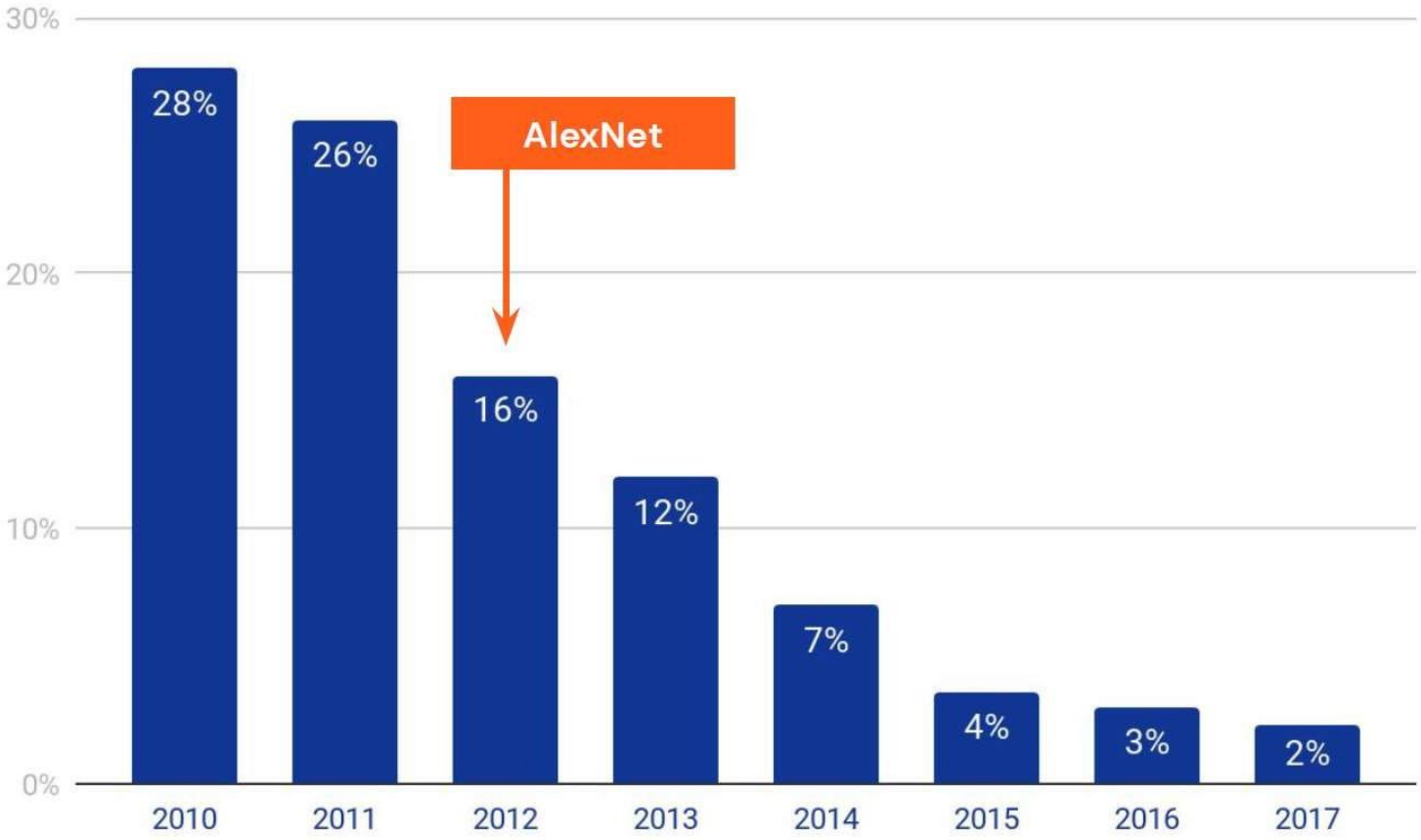
BACKGROUND
Data Drives Research



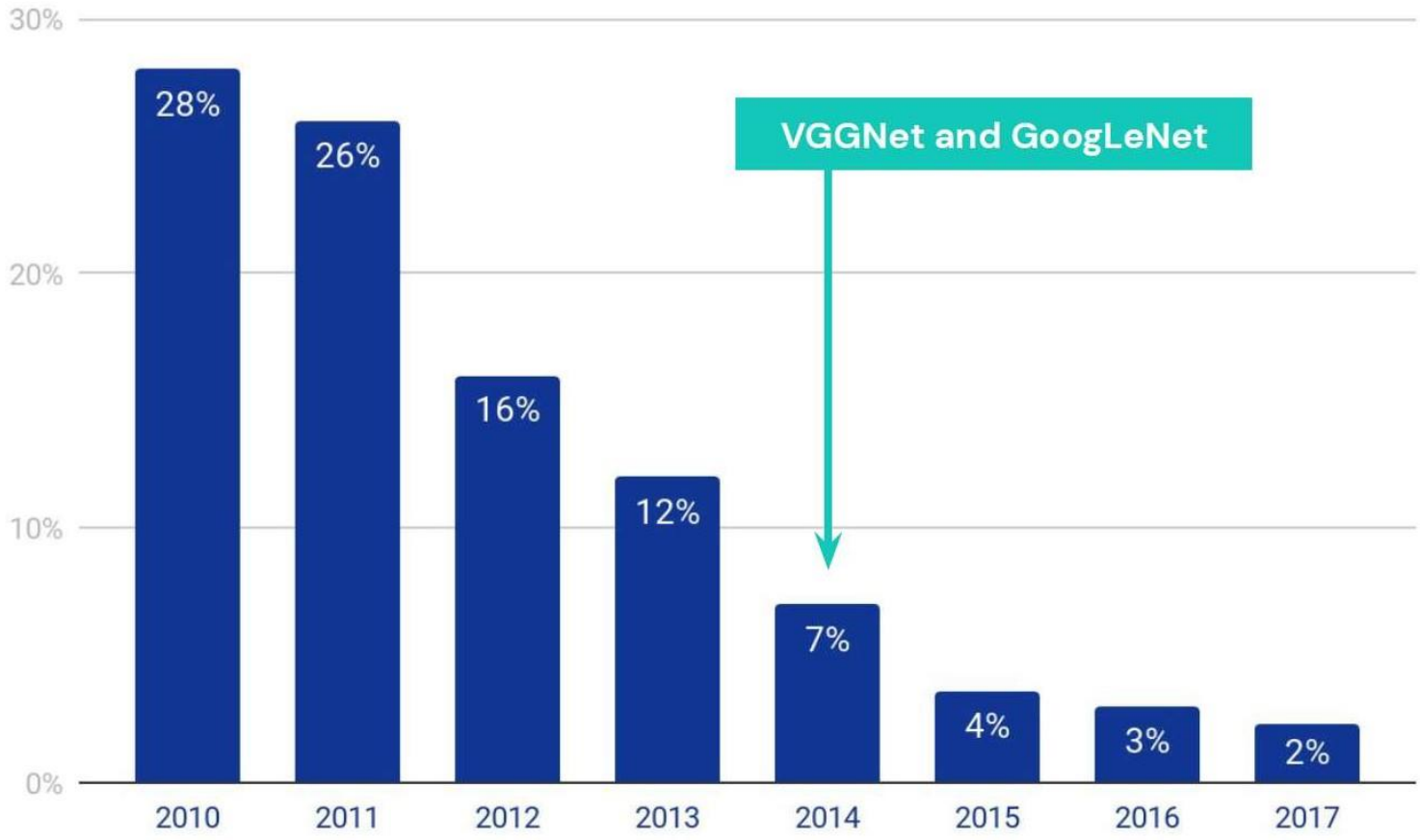
BACKGROUND
Data Drives Research



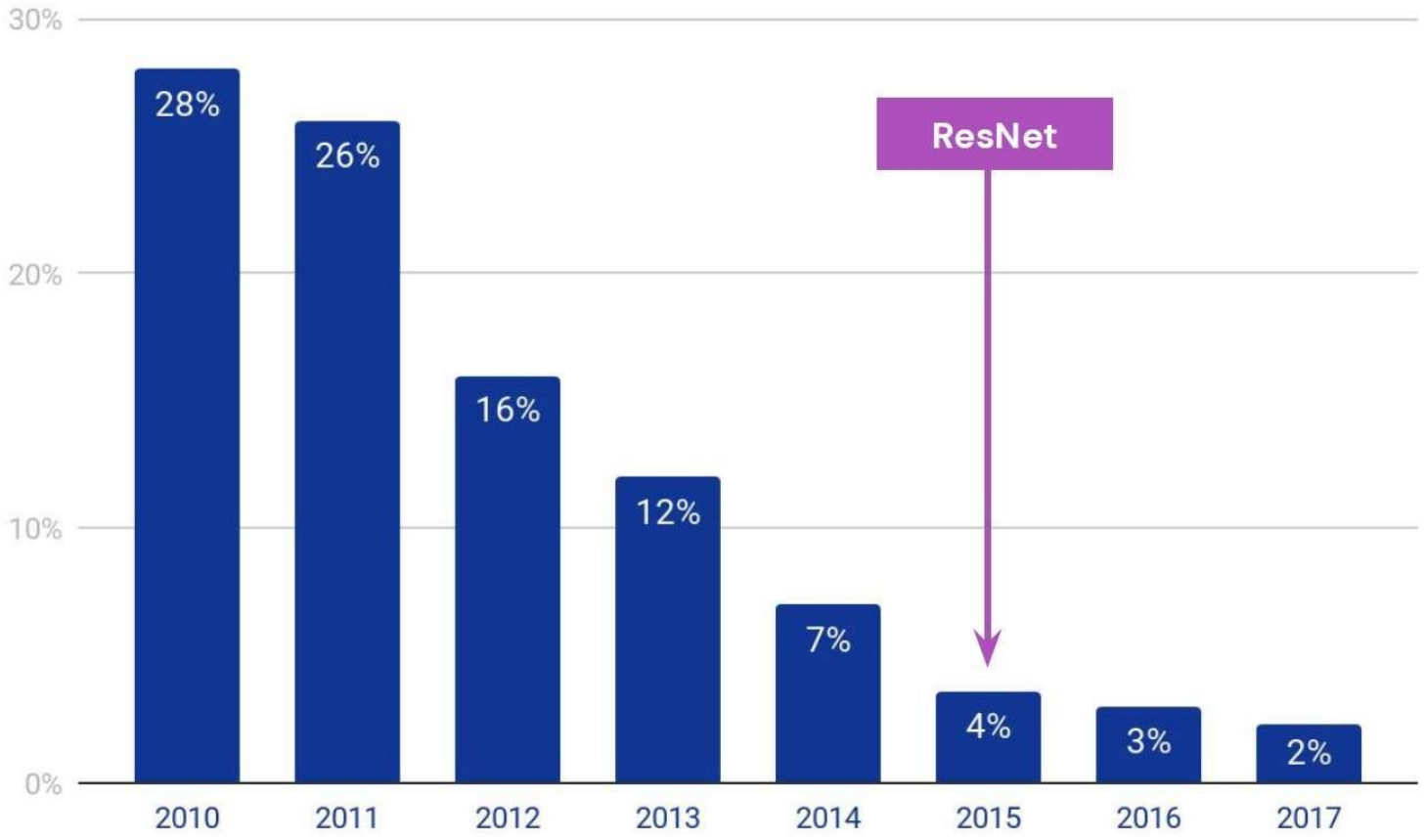
BACKGROUND
Data Drives Research



BACKGROUND
Data Drives Research



BACKGROUND
Data Drives Research

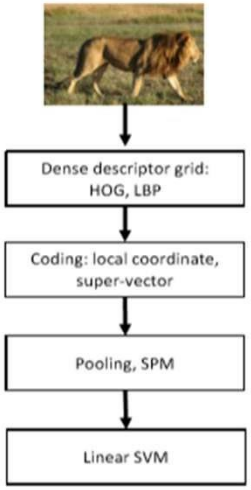


BACKGROUND
Data Drives Research

IMAGENET

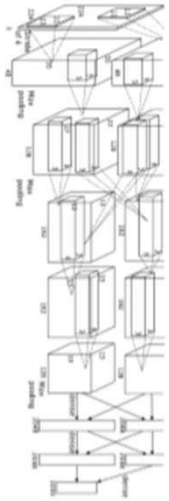
Year 2010

NEC-UIUC



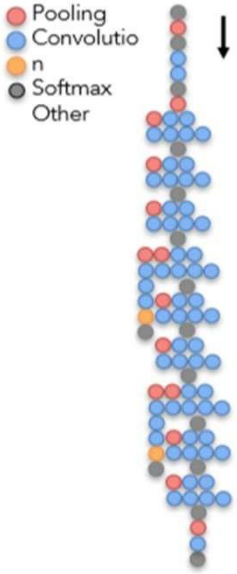
Year 2012

SuperVision

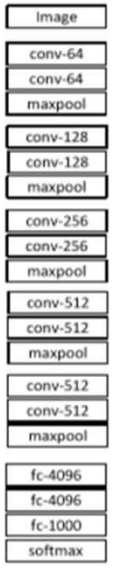


Year 2014

GoogLeNet

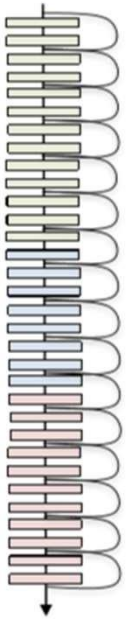


VGG

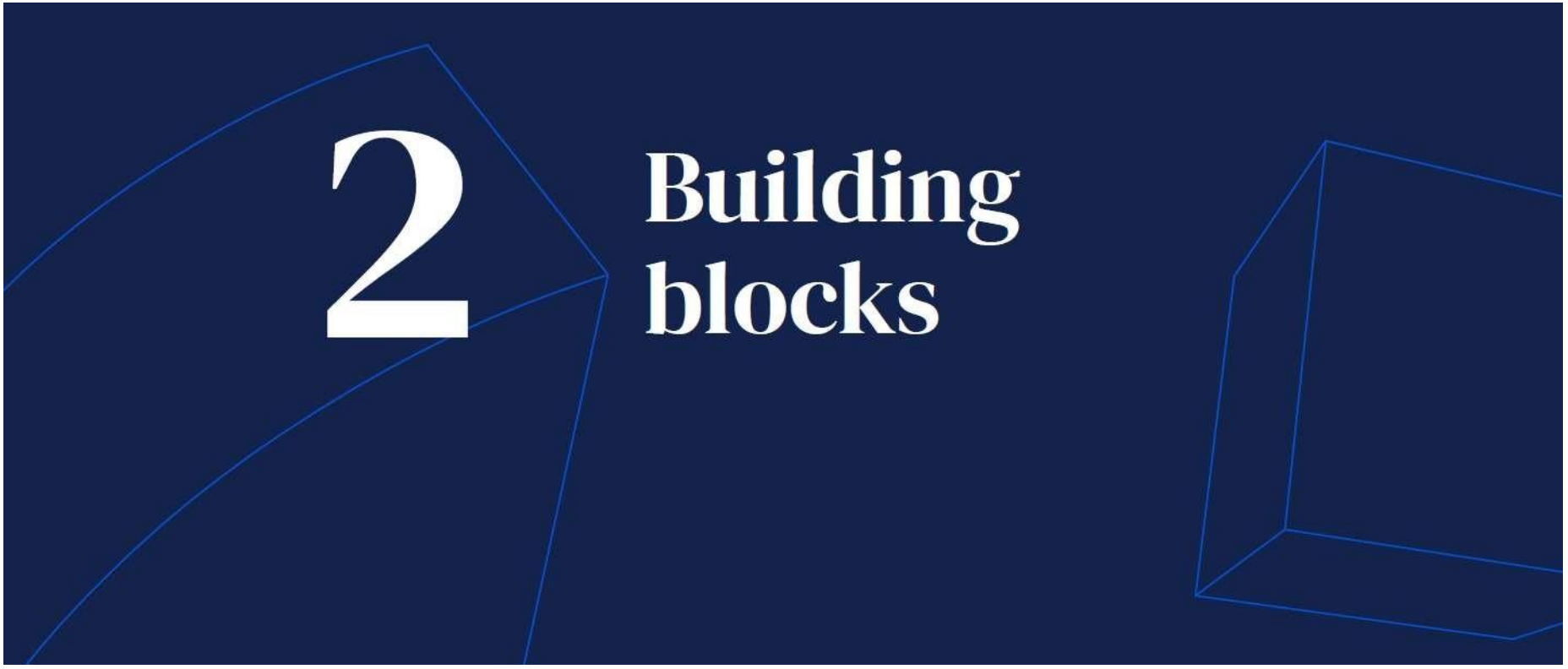


Year 2015

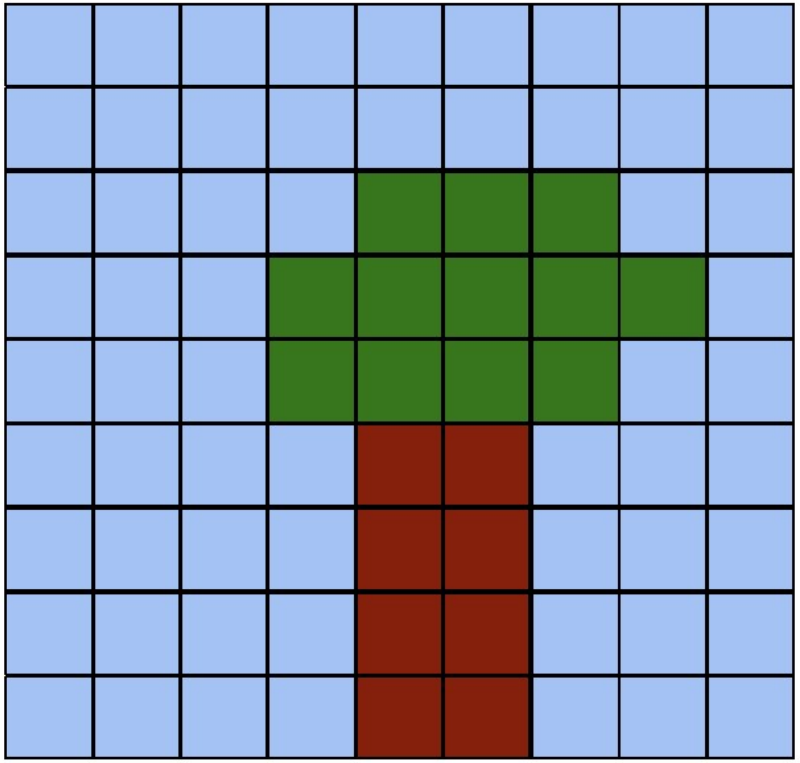
MSRA



Convolutional Neural Networks
Building Blocks

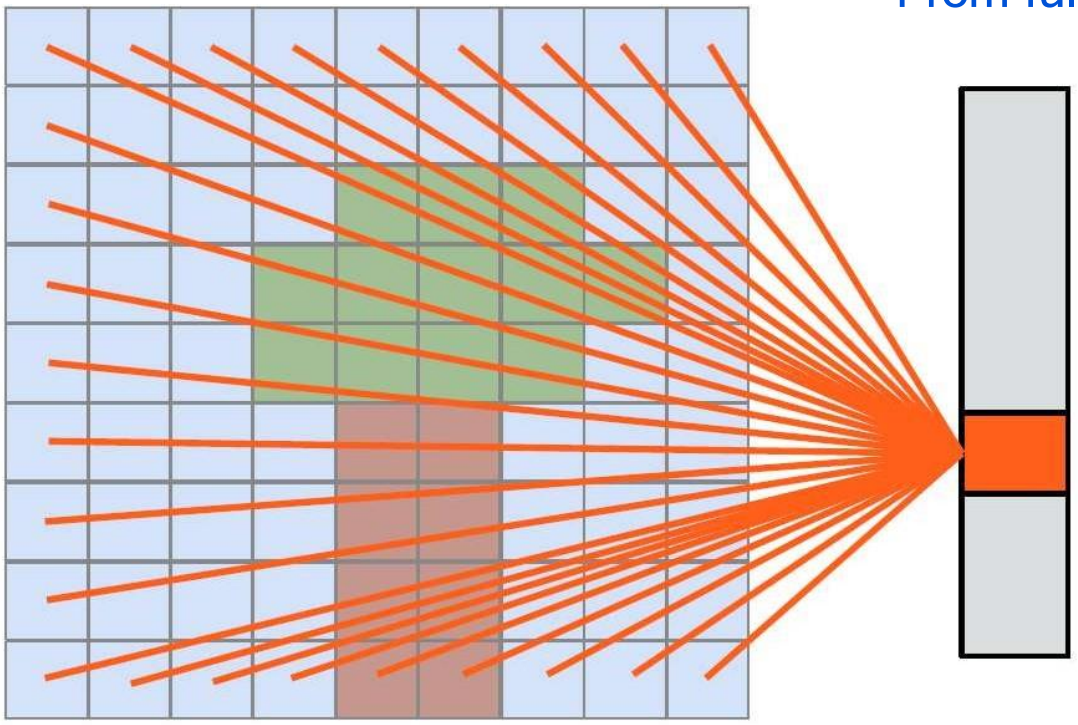


Convolutional Neural Networks
Building Blocks



Convolutional Neural Networks
Building Blocks

From fully connected to locally connected

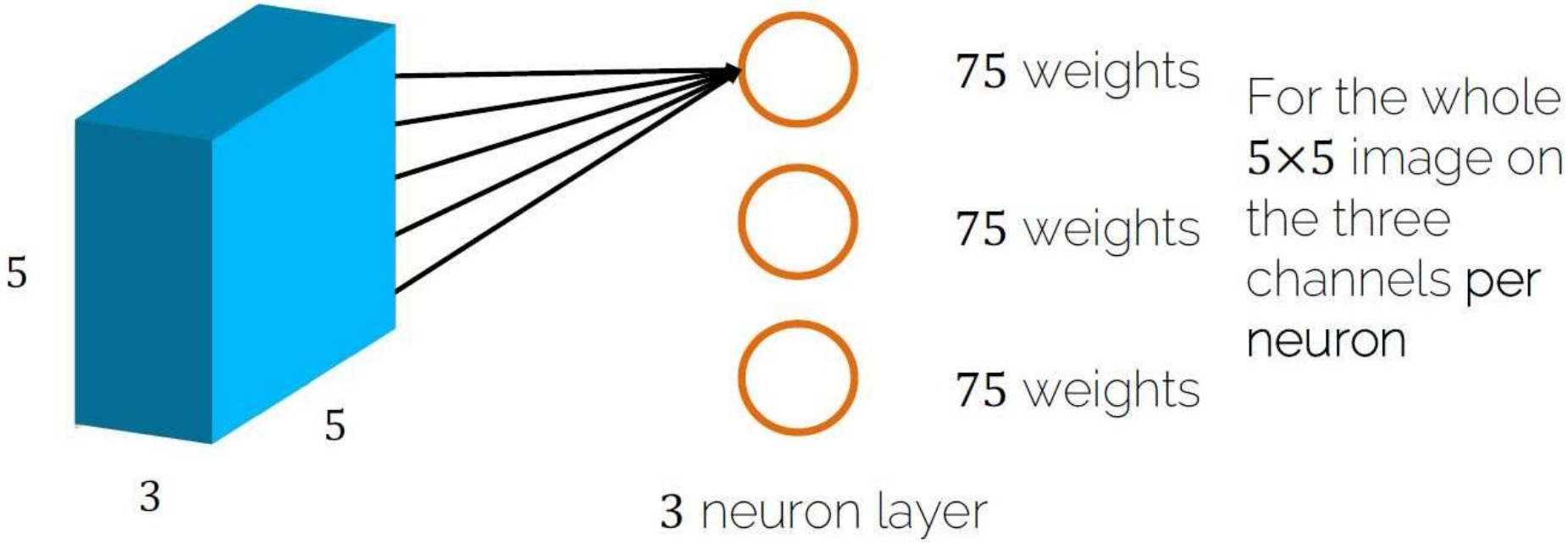


fully-connected unit

$$y = \sum_{i \in \text{image}} \mathbf{w}_i \mathbf{x}_i + b$$

Problems using FC Layers on Images

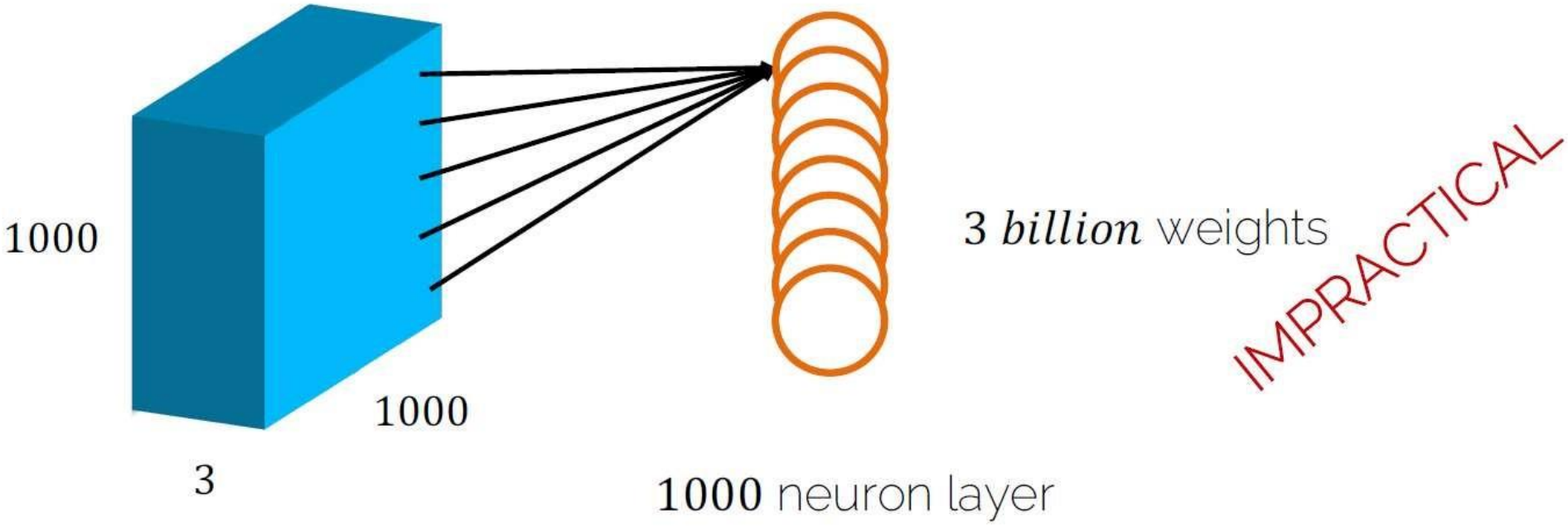
- How to process a tiny image with FC layers



Convolutional Neural Networks
Building Blocks

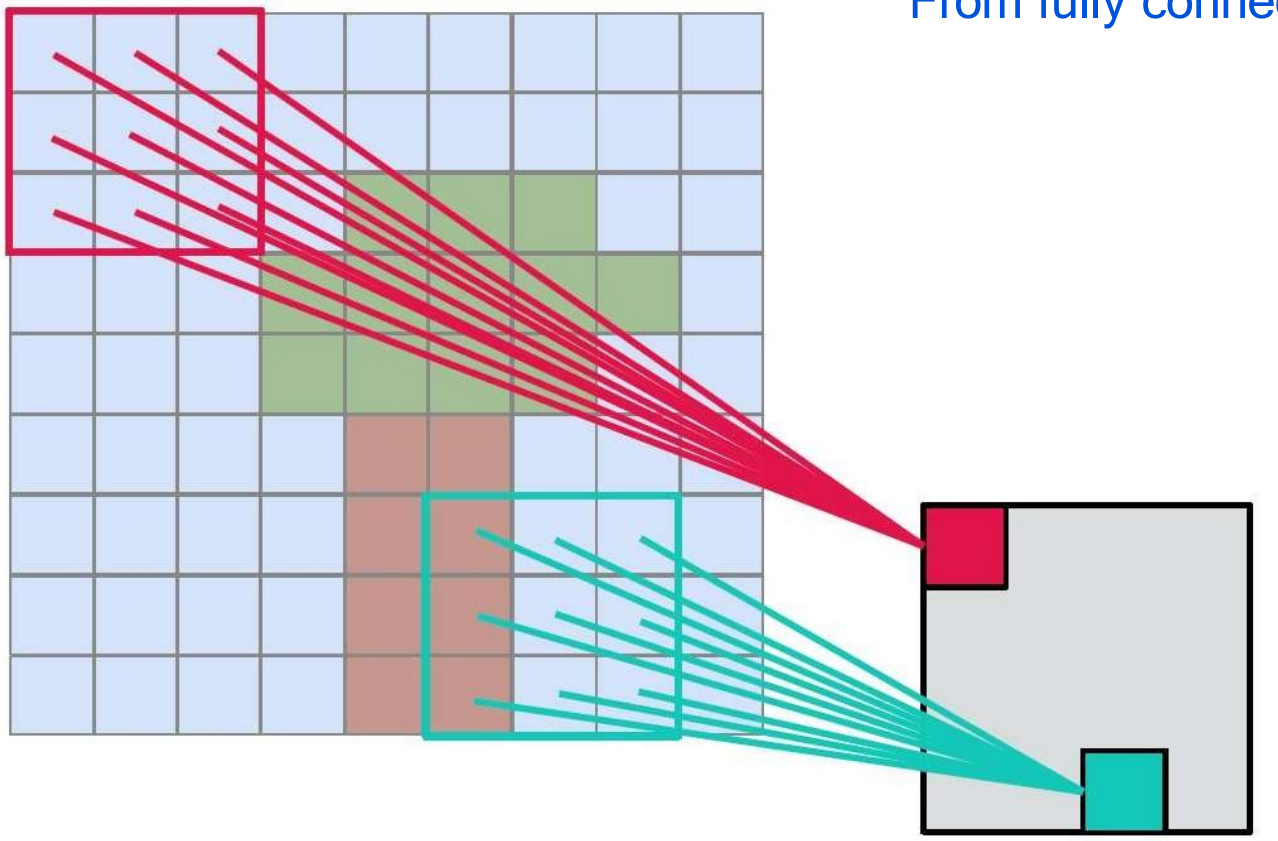
Problems using FC Layers on Images

- How to process a tiny image with FC layers



Convolutional Neural Networks
Building Blocks

From fully connected to locally connected

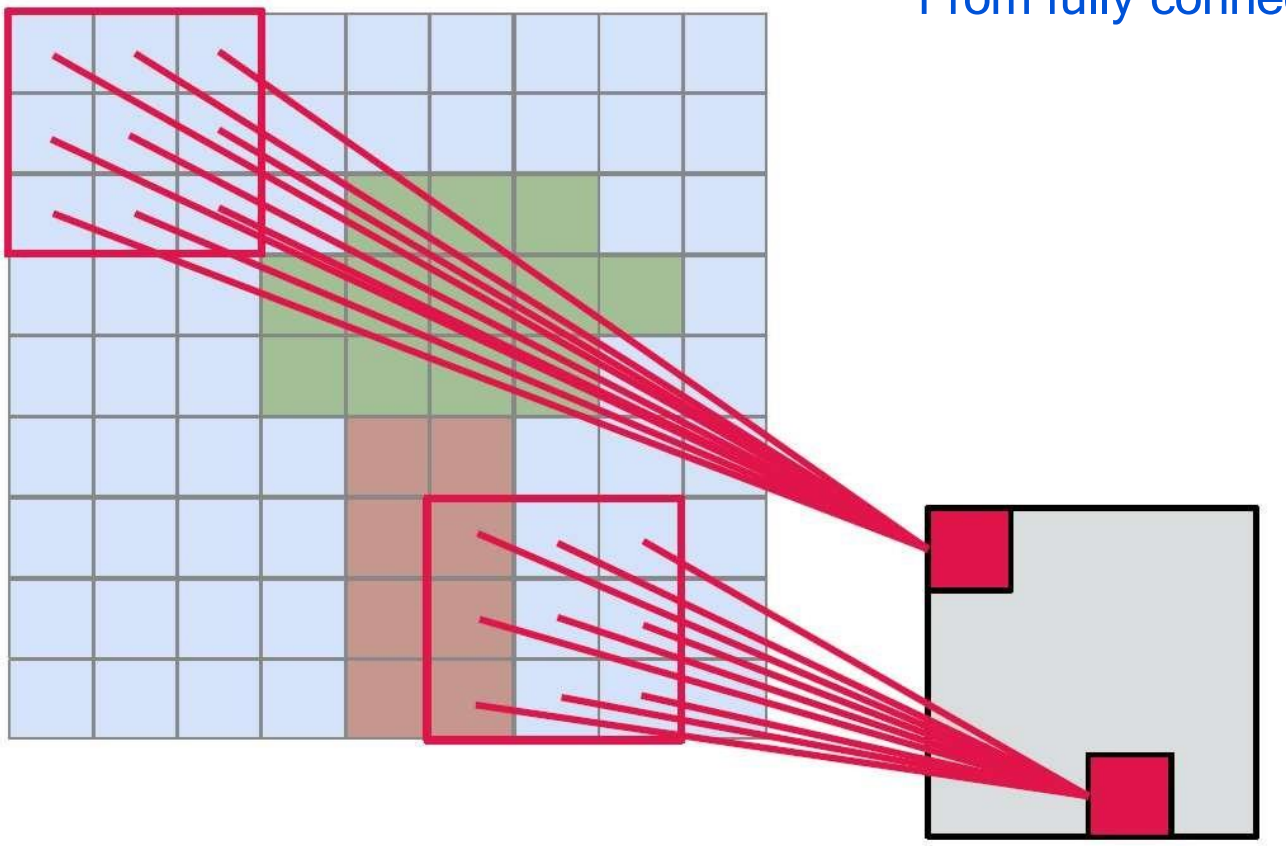


$$y = \sum_{i \in 3 \times 3} \mathbf{w}_i \mathbf{x}_i + b$$

locally-connected units
3×3 receptive field

Convolutional Neural Networks
Building Blocks

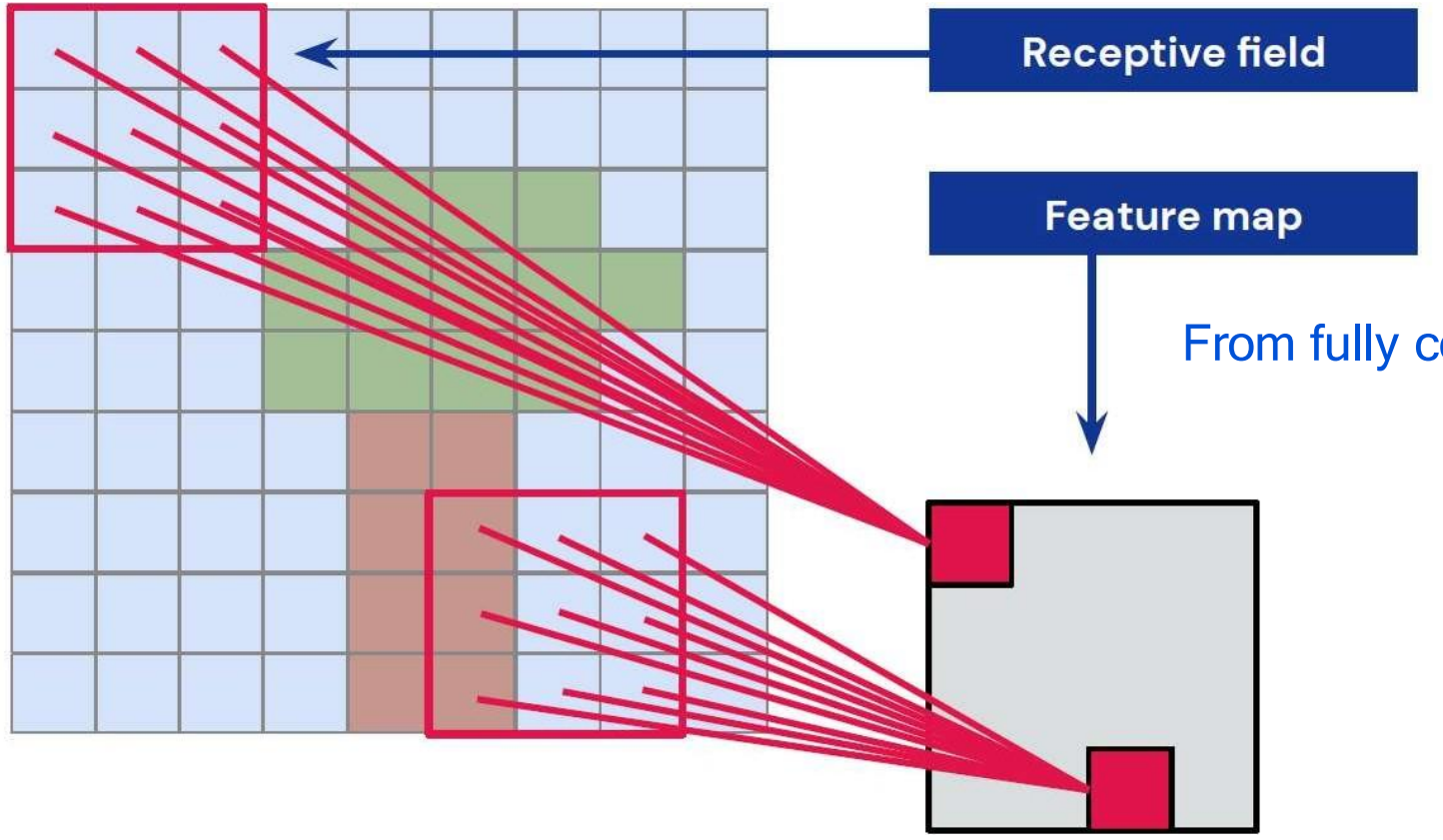
From fully connected to convolutional



$$y = \mathbf{w} * \mathbf{x} + b$$

convolutional units
3×3 receptive field

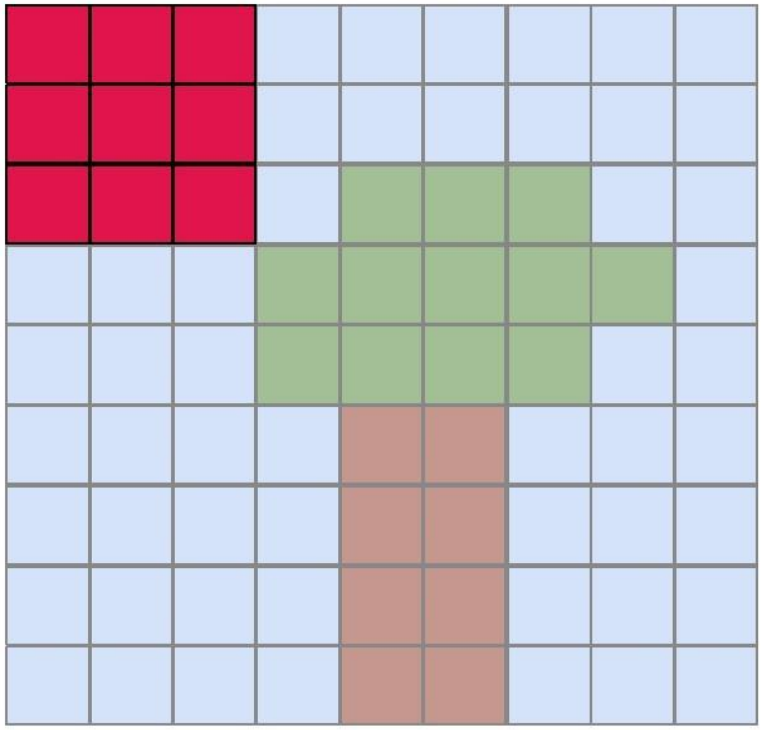
Convolutional Neural Networks
Building Blocks

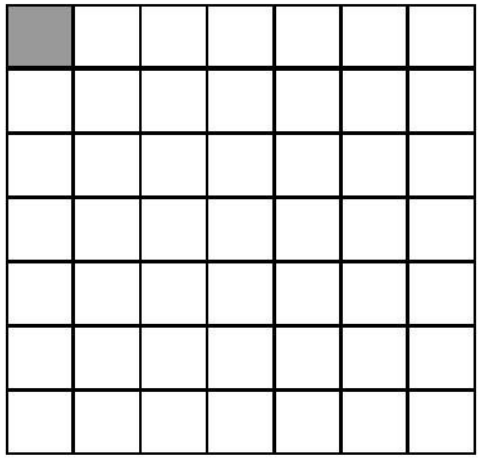


From fully connected to convolutional

Convolutional Neural Networks

Convolution Operation

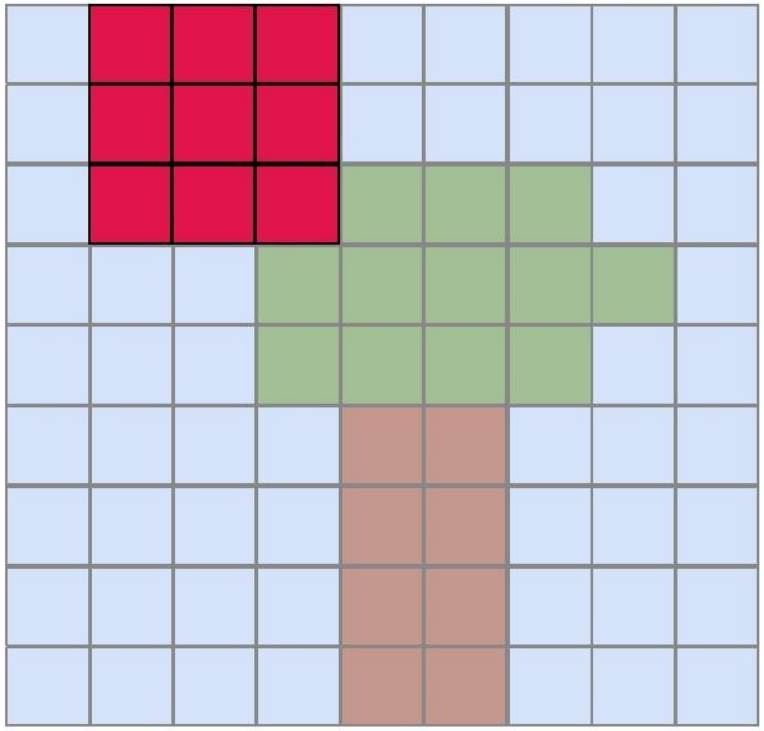


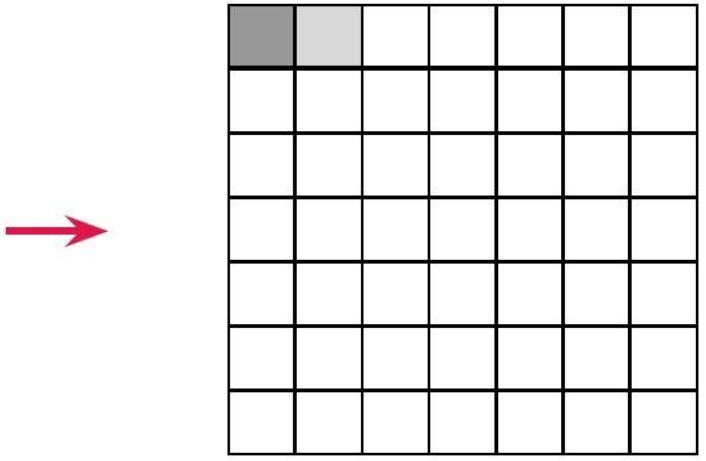


The **kernel** slides across the image and produces an output value at each position

Convolutional Neural Networks

Convolution Operation

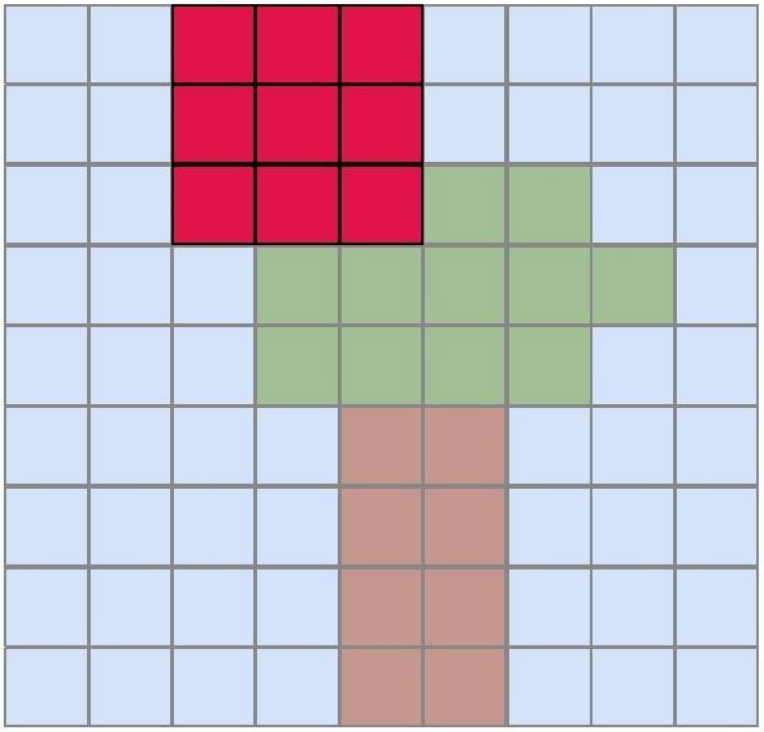


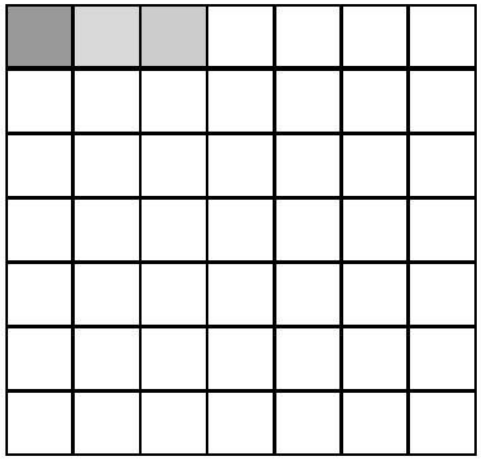


The **kernel** slides across the image and produces an output value at each position

Convolutional Neural Networks

Convolution Operation

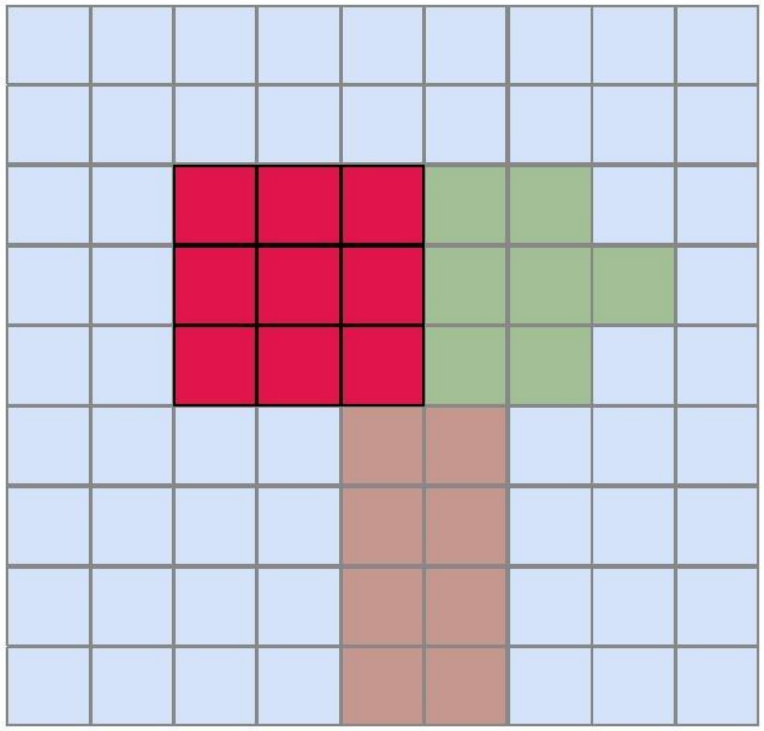


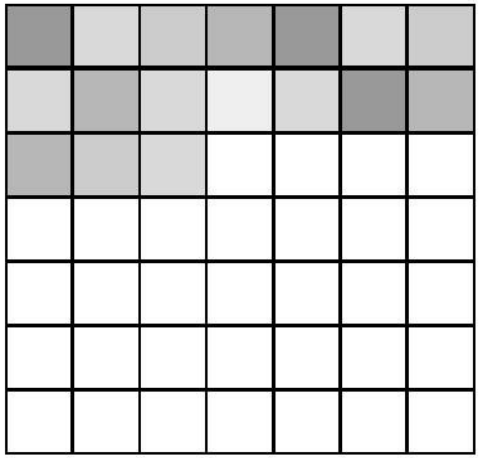


The **kernel** slides across the image and produces an output value at each position

Convolutional Neural Networks

Convolution Operation

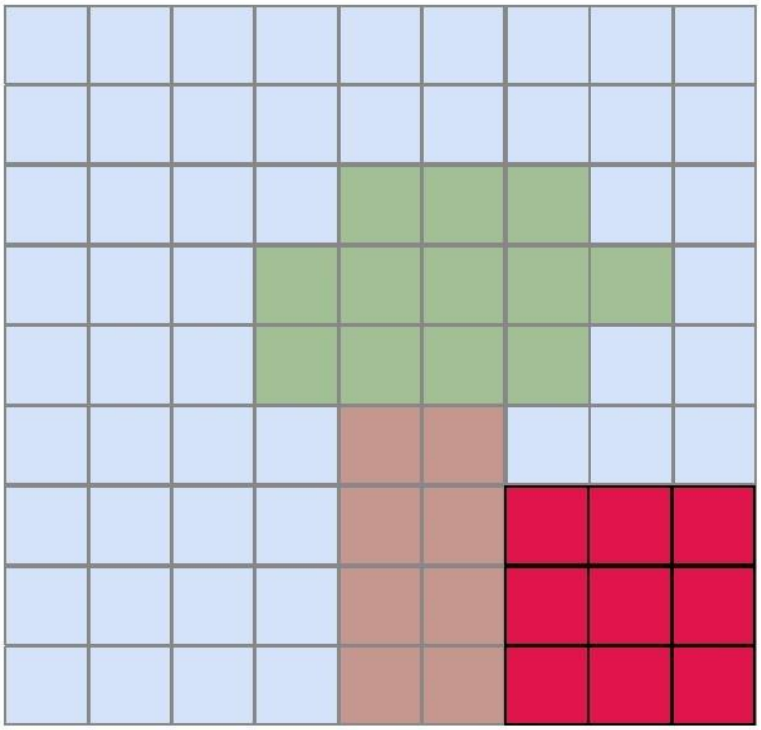


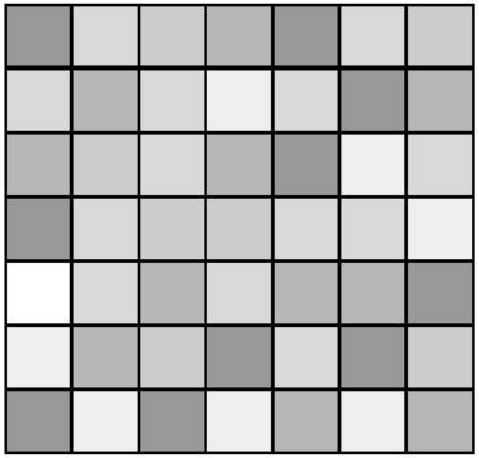


The **kernel** slides across the image and produces an output value at each position

Convolutional Neural Networks

Convolution Operation

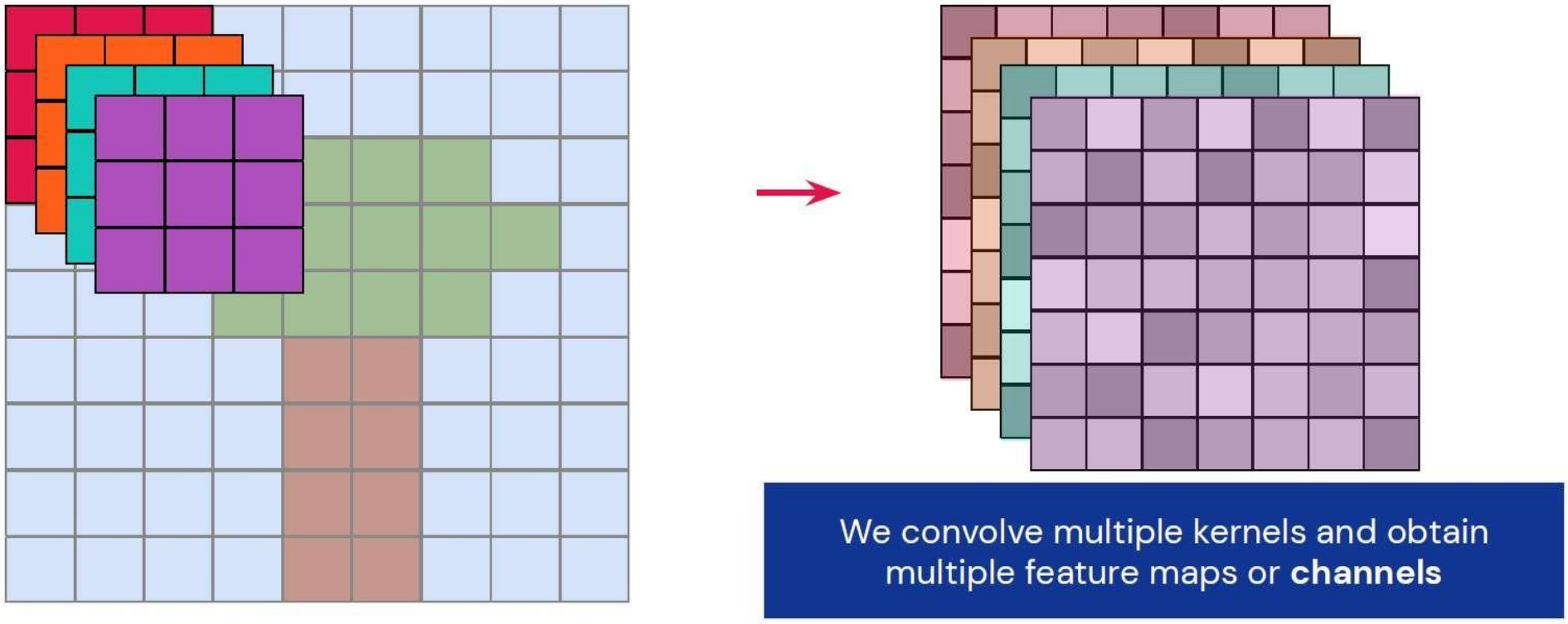




The **kernel** slides across the image and produces an output value at each position

Convolutional Neural Networks

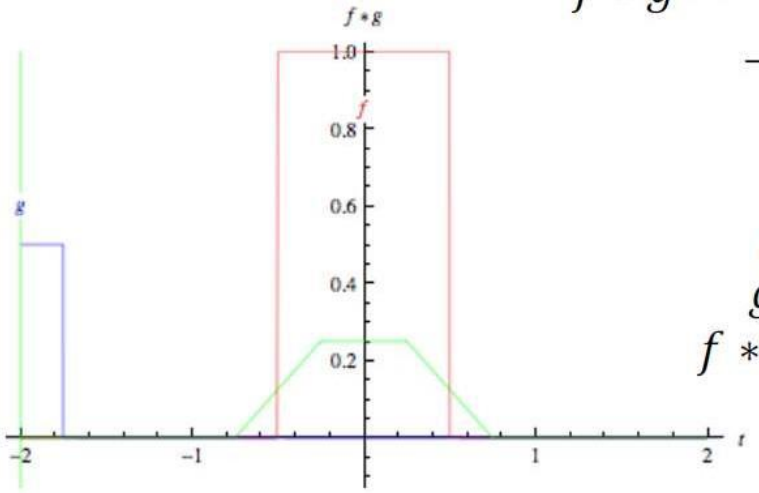
Convolution Operation



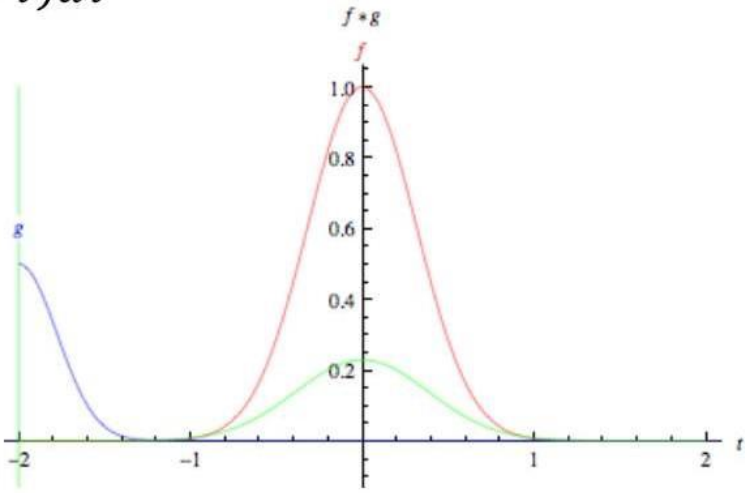
We convolve multiple kernels and obtain multiple feature maps or **channels**

Convolutional Neural Networks
Convolution Operation

$$f * g = \int_{-\infty}^{\infty} f(\tau)g(t-\tau)d\tau$$



f = red
g = blue
$f * g$ = green



Convolution of two box functions

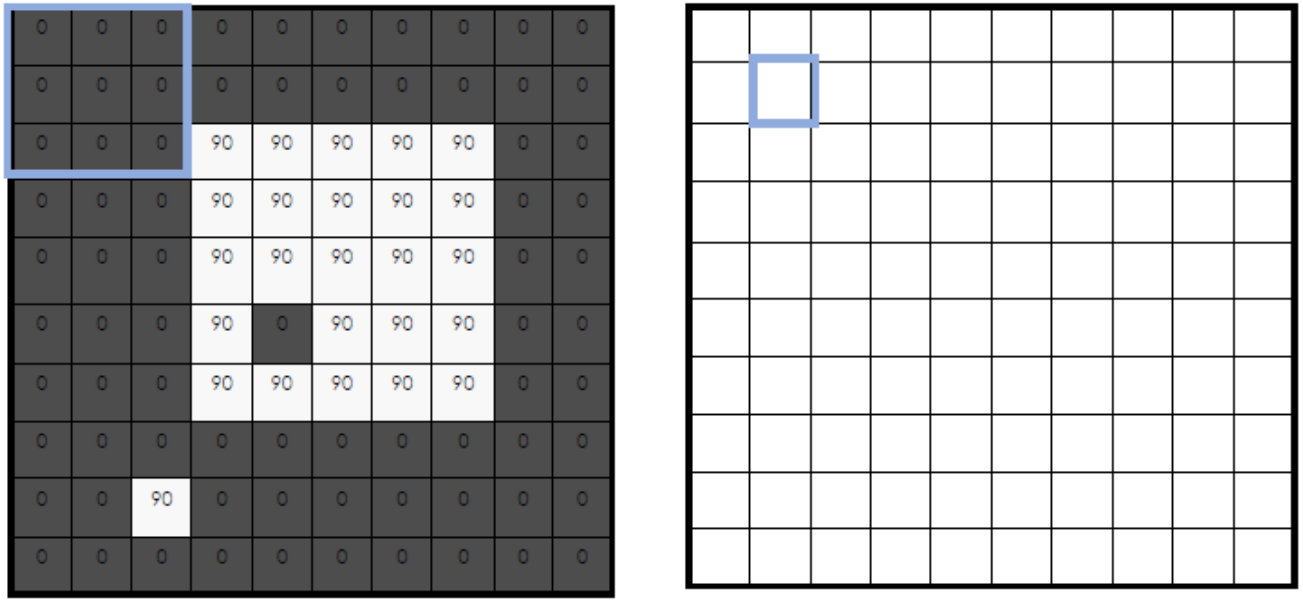
Convolution of two Gaussians

Application of a filter to a function
- The 'smaller' one is typically called the filter kernel

Convolutional Neural Networks

Convolution Operation

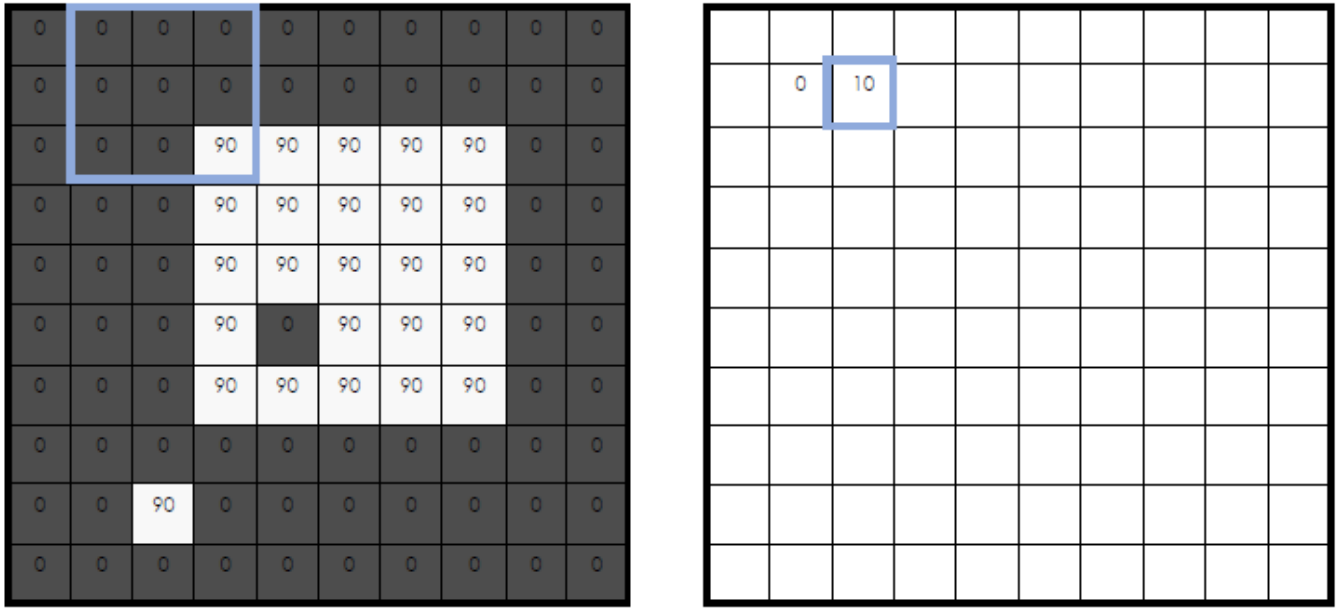
$$h[m,n] = \sum_{k,l} f[k,l]\, I[m+k, n+l]$$



Convolutional Neural Networks

Convolution Operation

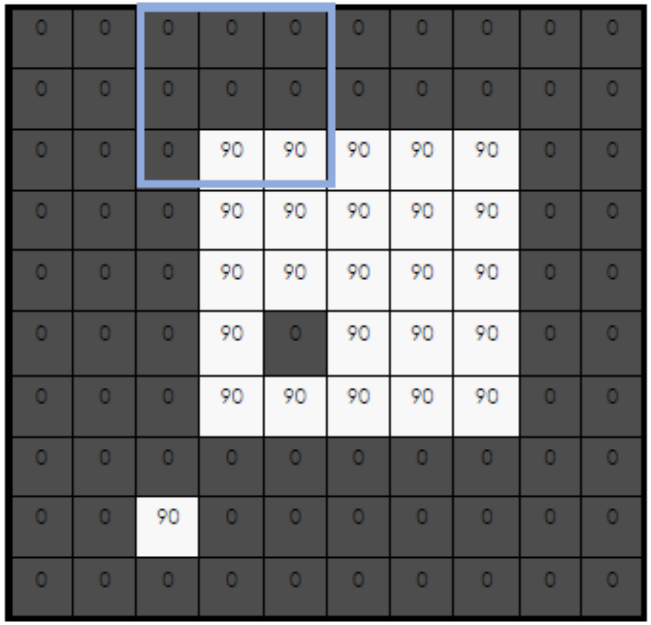
$$h[m,n] = \sum_{k,l} f[k,l]\, I[m+k, n+l]$$

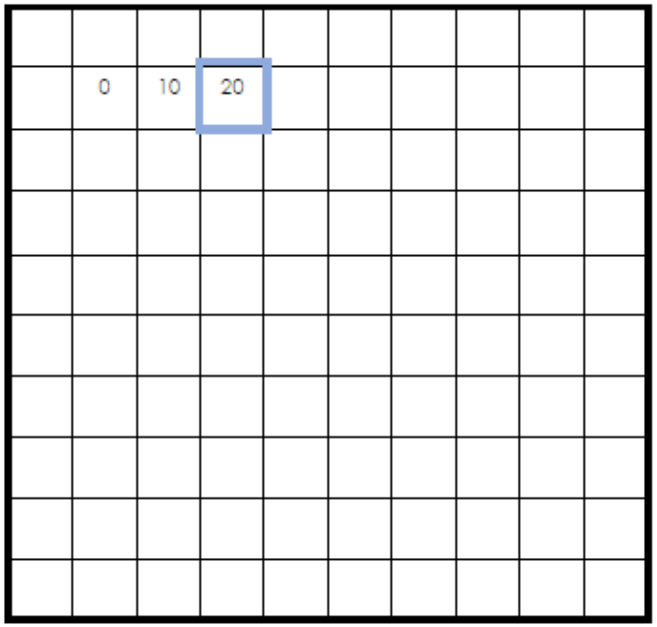


Convolutional Neural Networks

Convolution Operation

$$h[m,n] = \sum_{k,l} f[k,l]\, I[m+k, n+l]$$

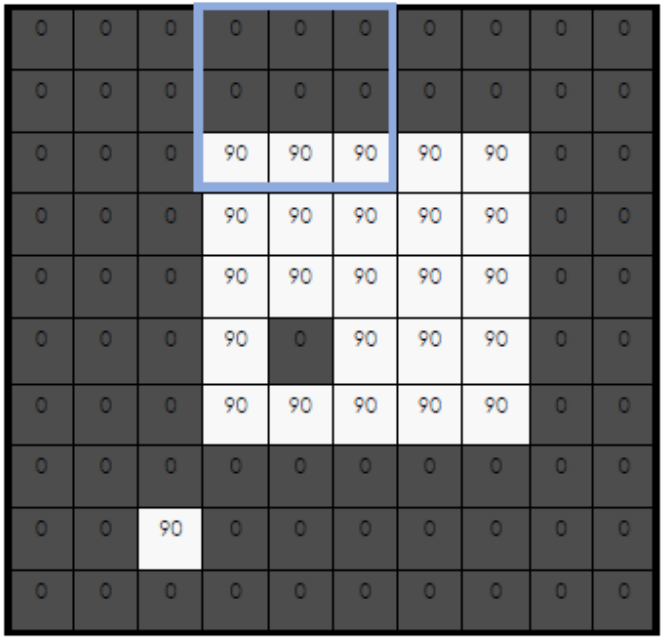


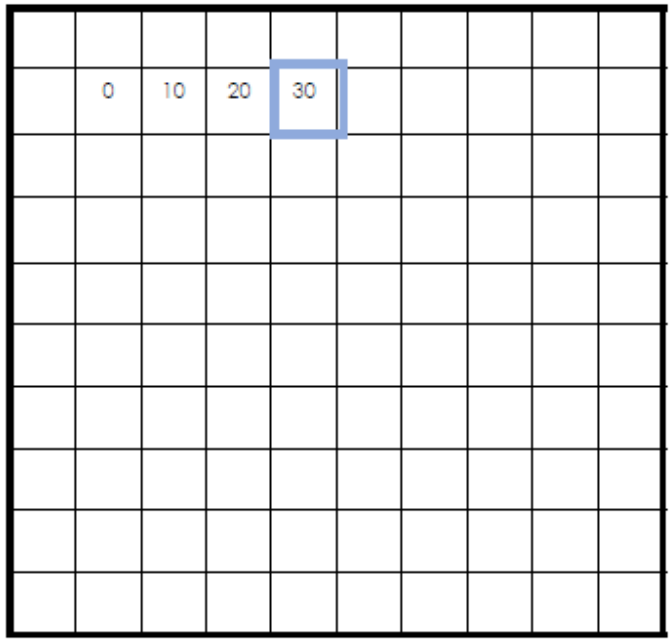


Convolutional Neural Networks

Convolution Operation

$$h[m,n] = \sum_{k,l} f[k,l]\, I[m+k, n+l]$$

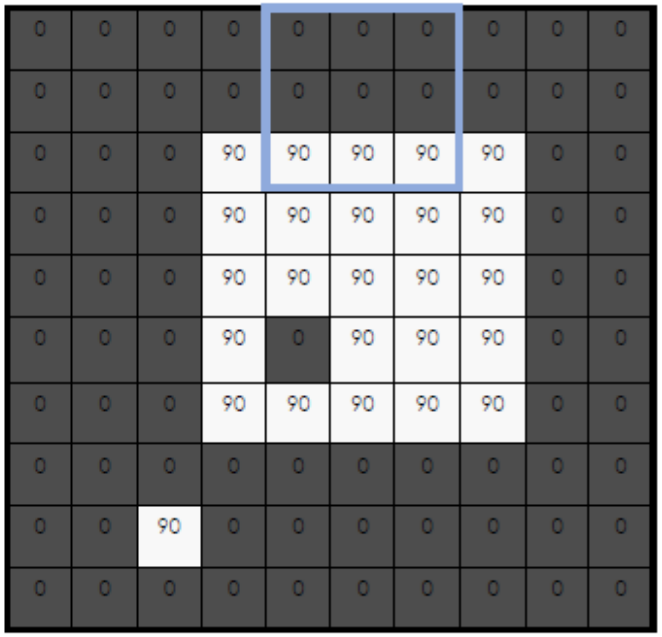


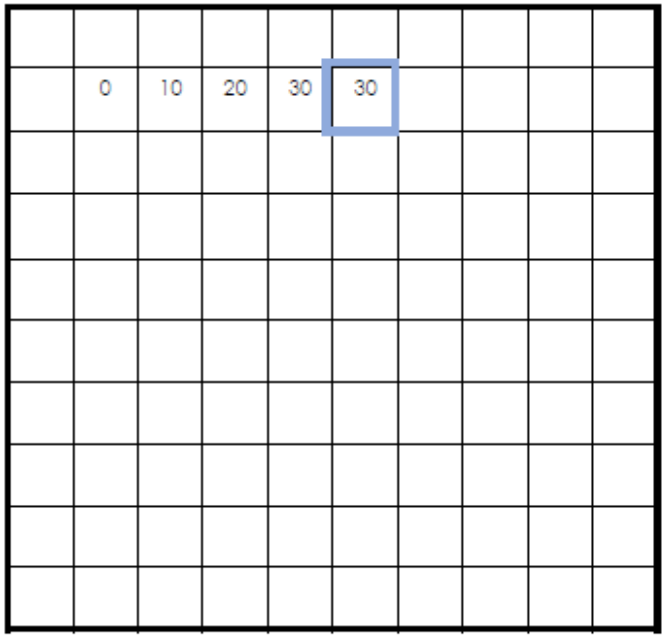


Convolutional Neural Networks

Convolution Operation

$$h[m,n] = \sum_{k,l} f[k,l]\, I[m+k, n+l]$$

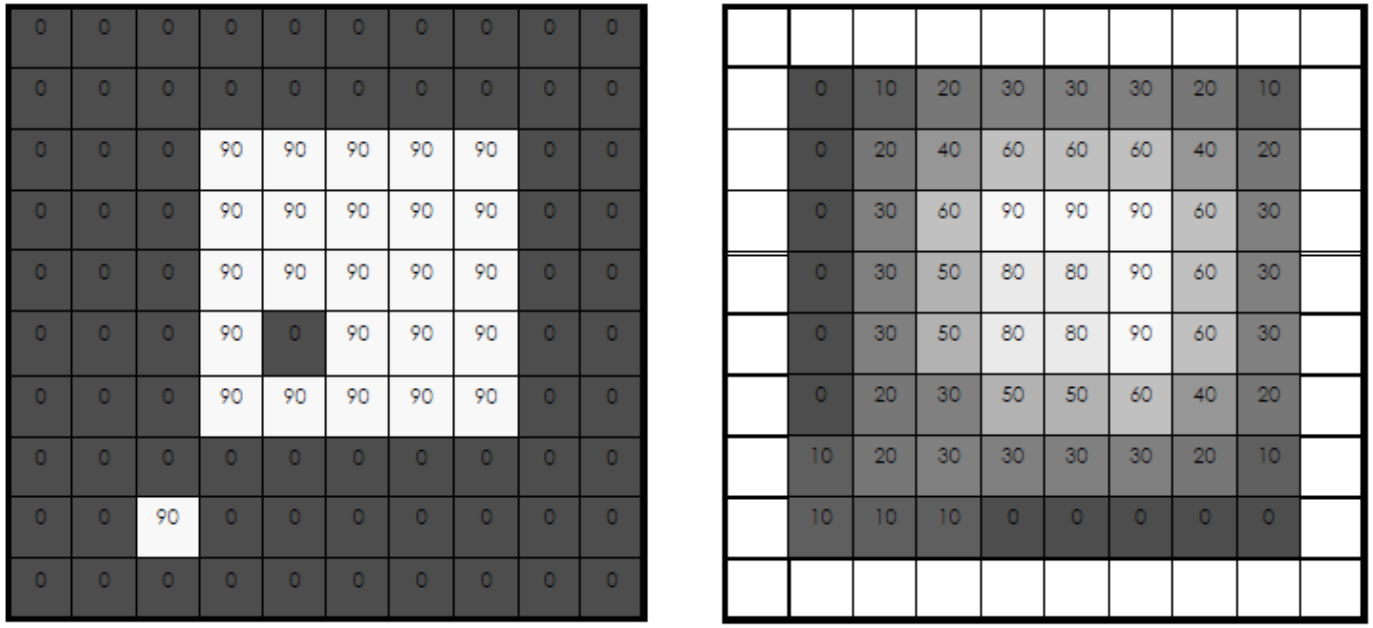




Convolutional Neural Networks

Convolution Operation

$$h[m,n] = \sum_{k,l} f[k,l]\, I[m+k, n+l]$$



Convolutional Neural Networks

Convolution Operation

Image filtering

The “kernels” compute a function of the sorrounding pxels

$$h[m,n] = \sum_{k,l} f[k,l]\, I[m+k, n+l]$$

Very important in image-processing applications

Improve of enhance properties of the image

Noise, sizes changes, contrast, etc

Information extraction

Texture, lines, keypoints, etc. Pattern detection and mateching:

Find correspondences between templates

“Gradient Filter”

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

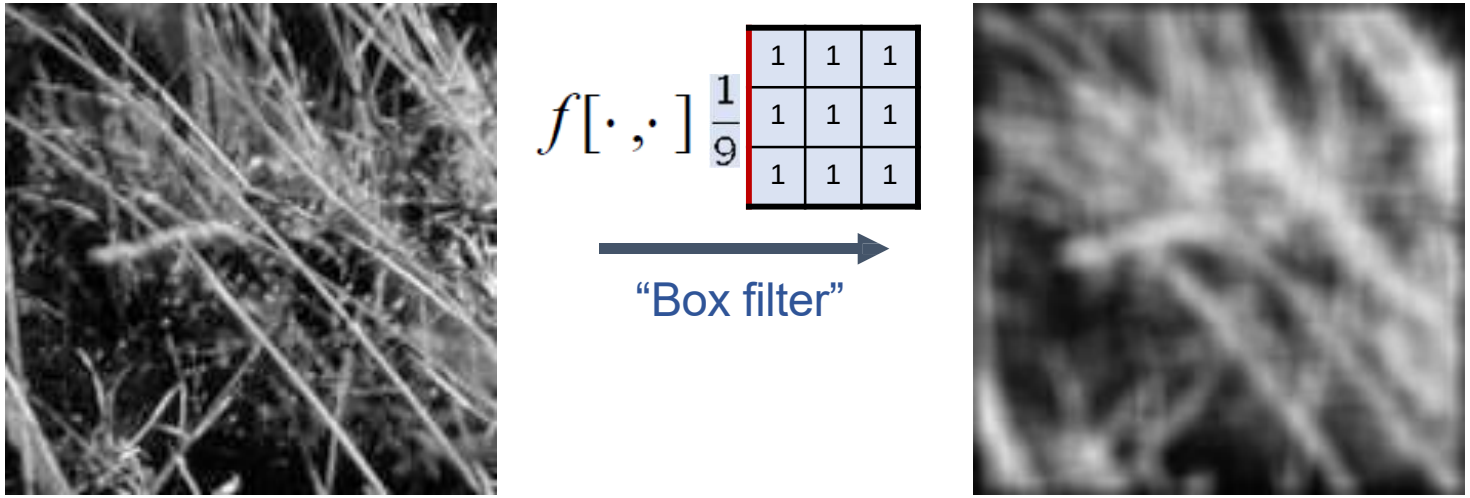
“Sobel Filter”

1	0	-1
2	0	-2
1	0	-1

Convolutional Neural Networks
Convolution Operation

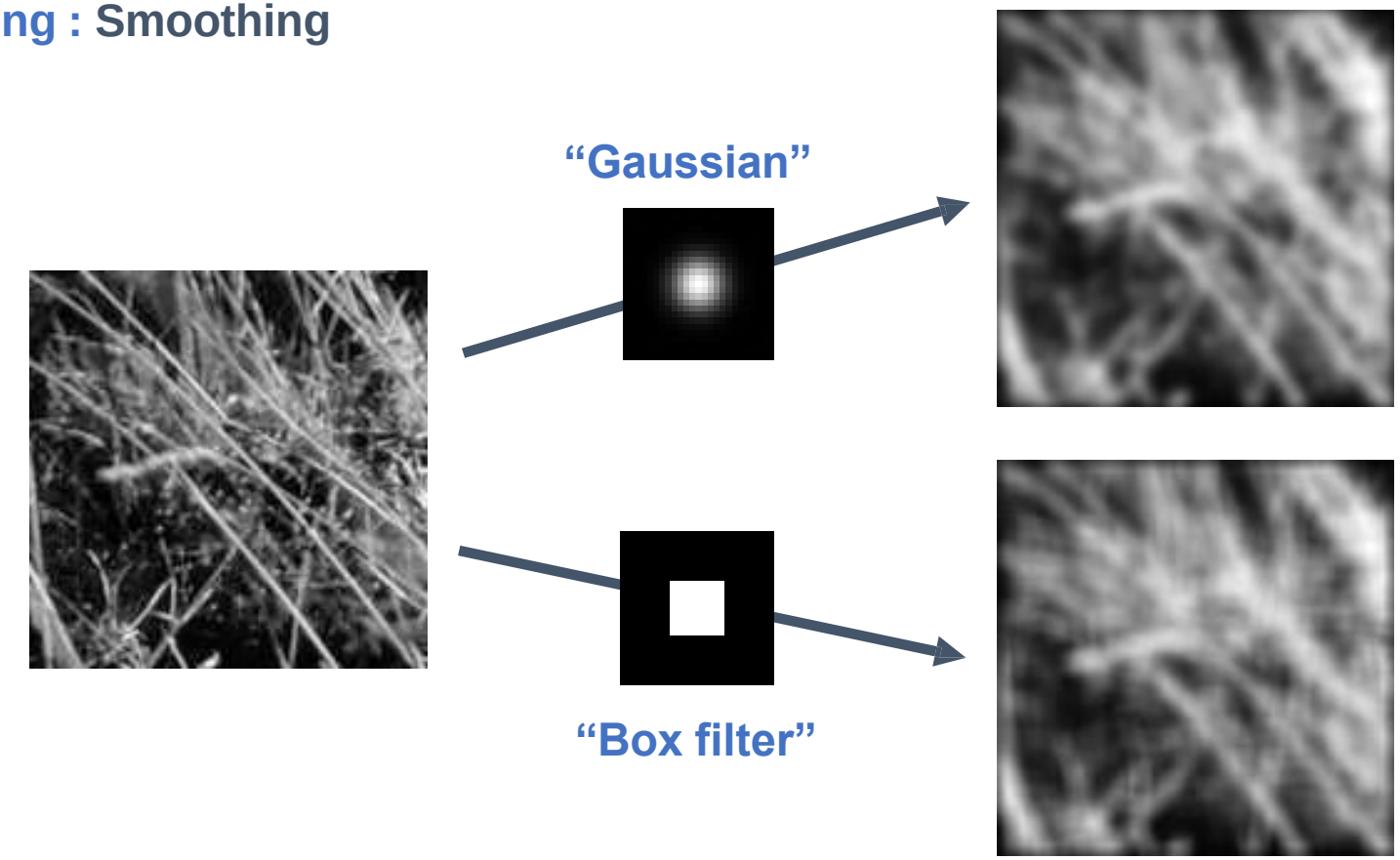
Image filtering : **Smoothing**

- This “kernel” replaces each pixel with the means of its neighbors
- It yields a smoothing effect (low-pass filter)
- A type of image processing highly used in computer vision



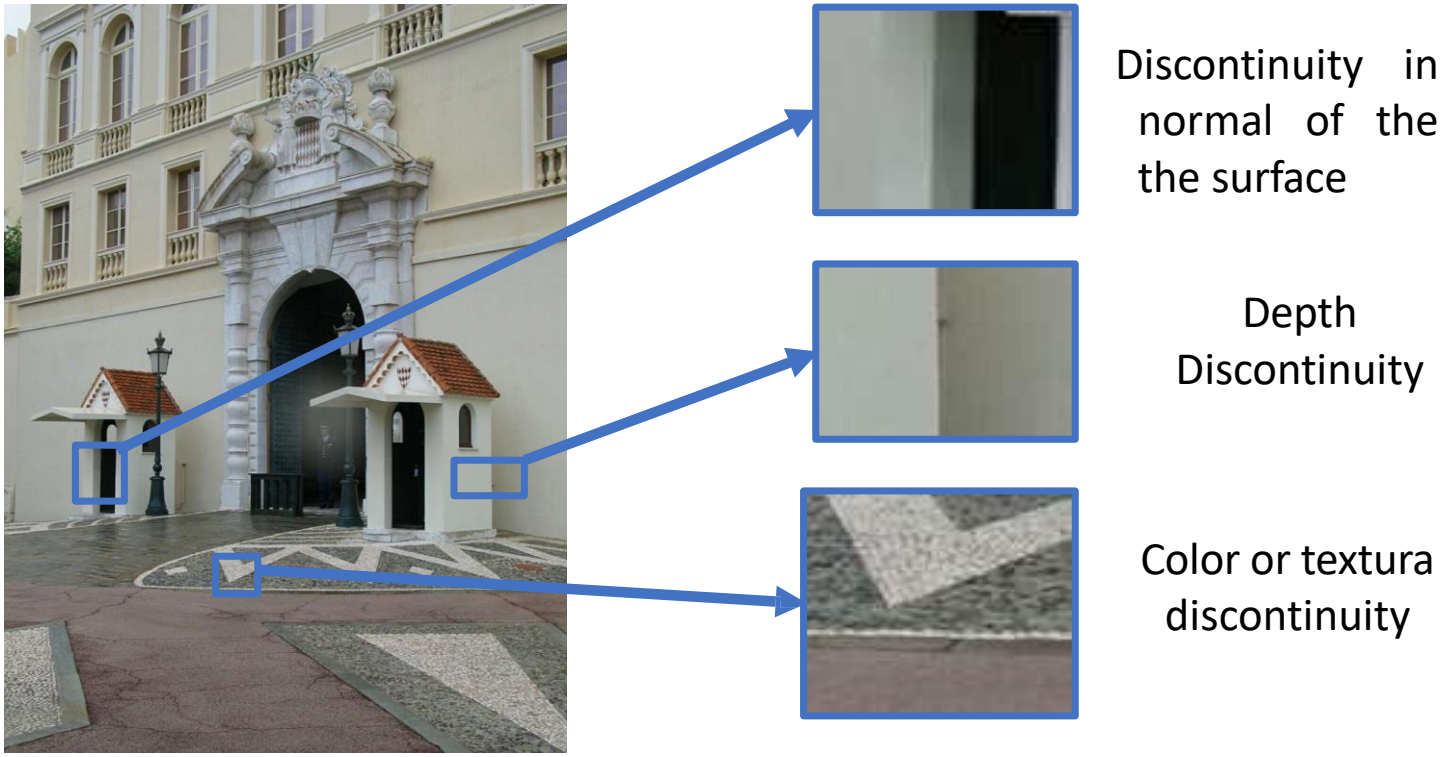
Convolutional Neural Networks
Convolution Operation

Image filtering : Smoothing



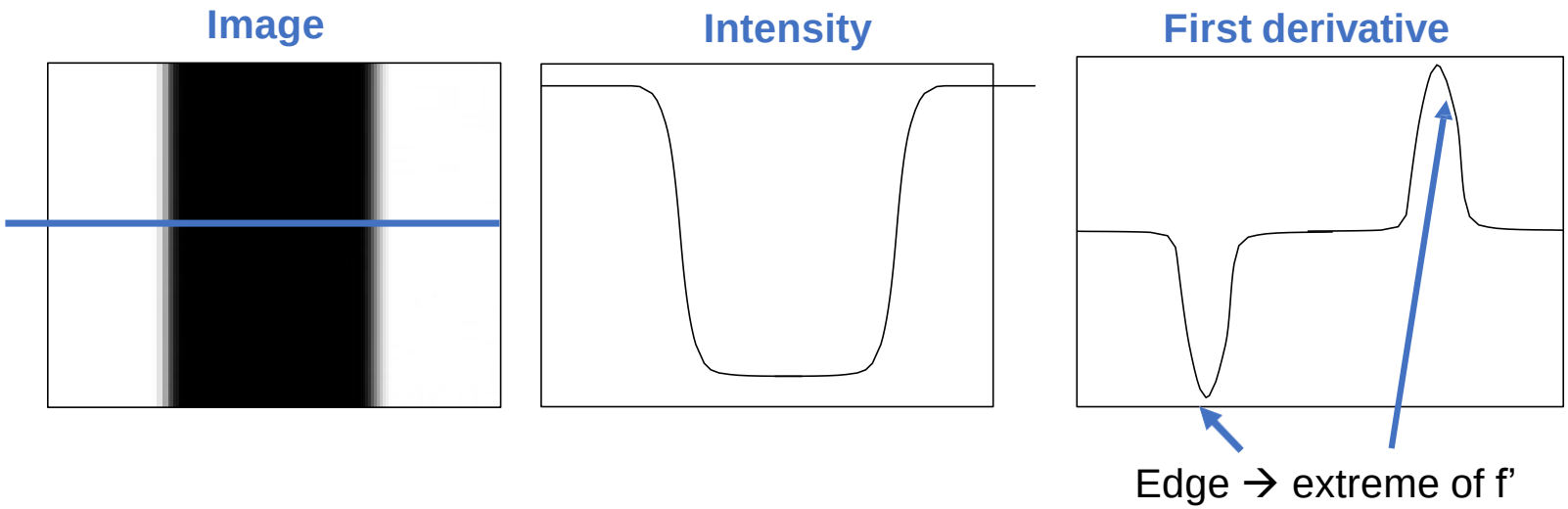
Convolutional Neural Networks
Convolution Operation

Image filtering : Edge Detection



Convolutional Neural Networks
Convolution Operation

$$\frac{df}{dx} = \lim_{\Delta x \to 0} \frac{f(x) - f(x - \Delta x)}{\Delta x} = f'(x) = f_x$$



Convolutional Neural Networks

Convolution Operation inside

$$\frac{df}{dx} = \lim_{\Delta x \to 0} \frac{f(x) - f(x - \Delta x)}{\Delta x} = f'(x)$$

$$\frac{df}{dx} = \frac{f(x) - f(x-1)}{1} = f'(x)$$

$$\frac{df}{dx} = f(x) - f(x-1) = f'(x)$$

Discrete derivative in 1D

Convolutional Neural Networks

Convolution Operation

Backward

$$\frac{df}{dx} = f(x) - f(x-1) = f'(x)$$

Forward

$$\frac{df}{dx} = f(x) - f(x+1) = f'(x)$$

Central

$$\frac{df}{dx} = f(x+1) - f(x-1) = f'(x)$$

Discrete derivative in 1D

Convolutional Neural Networks

Convolution Operation

Discrete Derivative in 2D

Given function → $f(x,y)$

Gradient Vector →
$$\nabla f(x,y) = \begin{bmatrix} \frac{\partial f(x,y)}{\partial x} \\ \frac{\partial f(x,y)}{\partial y} \end{bmatrix} = \begin{bmatrix} f_x \\ f_y \end{bmatrix}$$

Gradient Magnitude →
$$\left|\nabla f(x,y)\right| = \sqrt{f_x^2 + f_y^2}$$

Gradient Orientation →
$$\theta = \tan^{-1}\frac{f_x}{f_y}$$

Convolutional Neural Networks
Convolution Operation

Backward filter: $f(x) - f(x-1) = f'(x)$ [0 1 -1]

Forward: $f(x) - f(x+1) = f'(x)$ [-1 1 0]

Central: $f(x+1) - f(x-1) = f'(x)$ [1 0 -1]

Types of discrete derivatives in 1D

Convolutional Neural Networks

Convolution Operation

-1	0	
0	1	

0	-1	
1	0	

Robert's cross-gradient operators

+1	0	-1
+1	0	-1
+1	0	-1

+1	+1	+1
0	0	0
-1	-1	-1

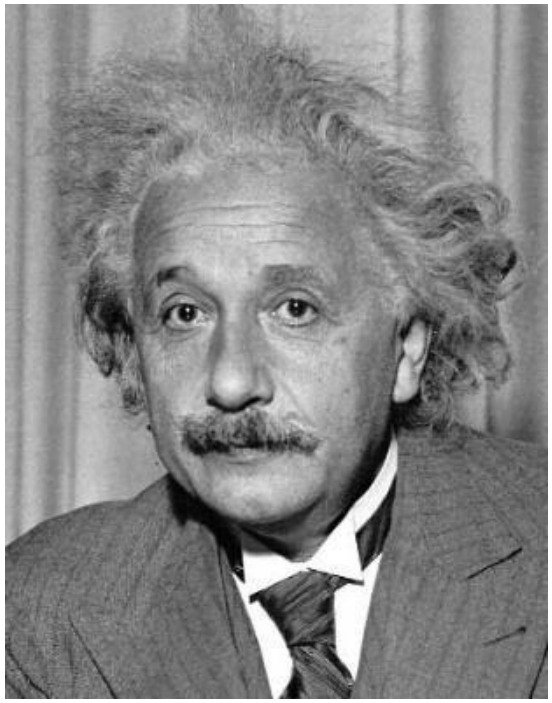
Prewitt Gradient Operators

-1	0	1
-2	0	2
-1	0	1

-1	-2	-1
0	0	0
1	2	1

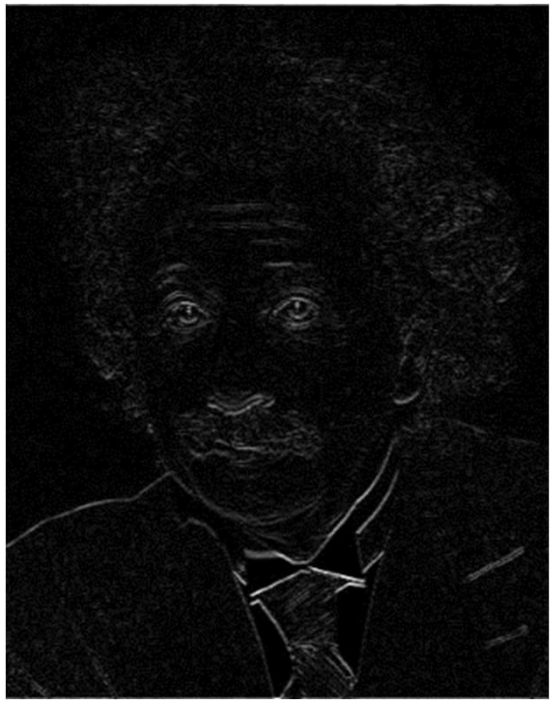
Sobel Gradient Operators

Convolutional Neural Networks
Convolution Operation



Sobel

1	2	1
0	0	0
-1	-2	-1



Convolutional Neural Networks
Convolution Operation

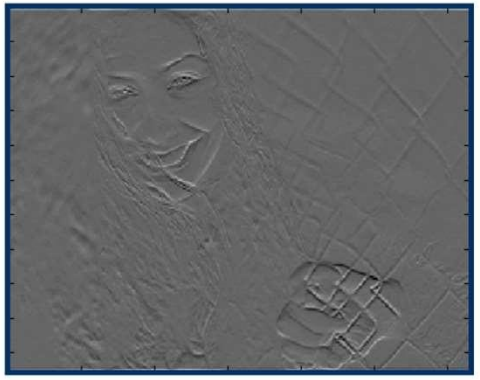
Detección de líneas

Horizontal Lines

Discrete Derivative in 2D

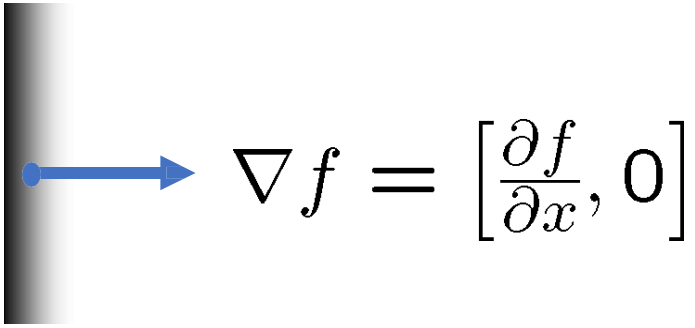
$$\frac{1}{3}\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$$

$$\frac{1}{3}\begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$$

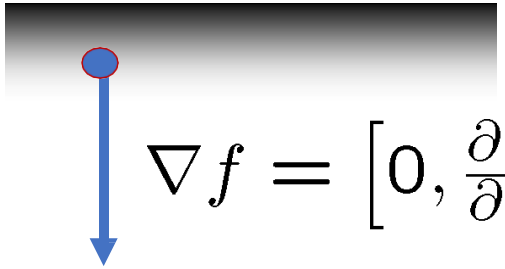


Convolutional Neural Networks
Convolution Operation

The **gradient** of an image $\rightarrow$ $\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$



$\nabla f = \left[\frac{\partial f}{\partial x}, 0\right]$



$\nabla f = \left[0, \frac{\partial f}{\partial y}\right]$

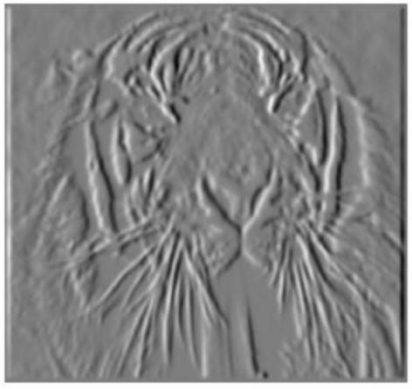
The **vector of the gradient** points in the direction of the major increment of the intensity of the input function

The **"strength of the line"** is given by the **magnitude of the gradient**

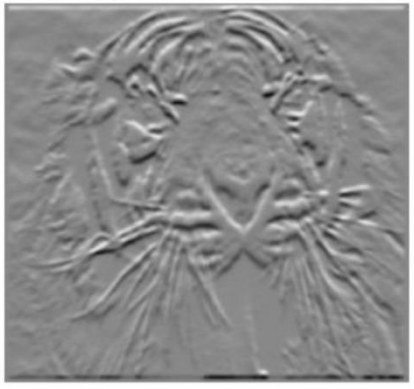
Convolutional Neural Networks
Convolution Operation

Original Image

Gradient Magnitued



Gradient Direction x

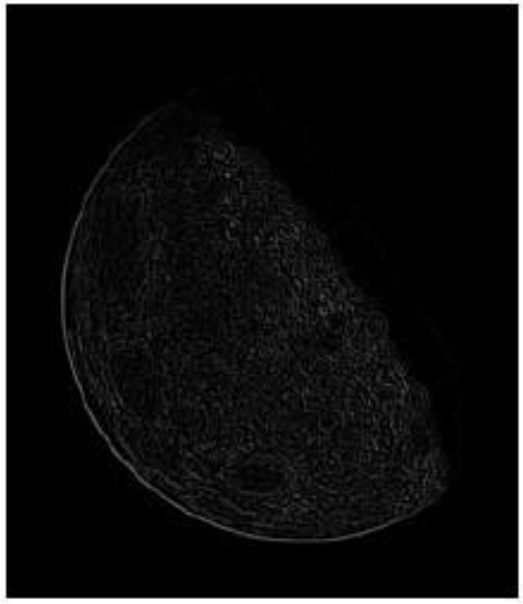


Gradient Direction y

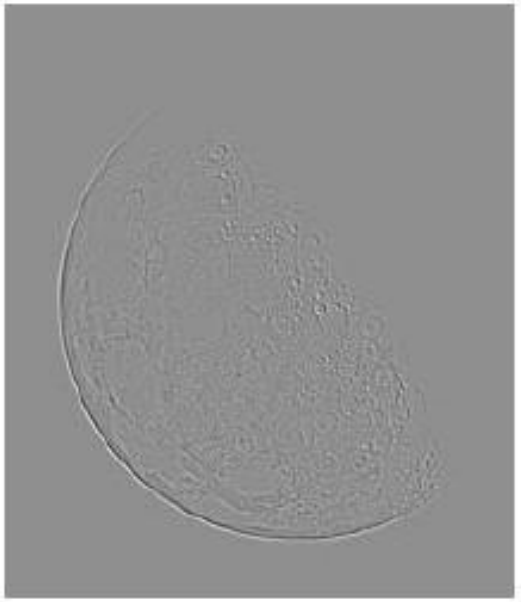
Convolutional Neural Networks

Convolution Operation

Orinal Image



Laplacian Non scaled



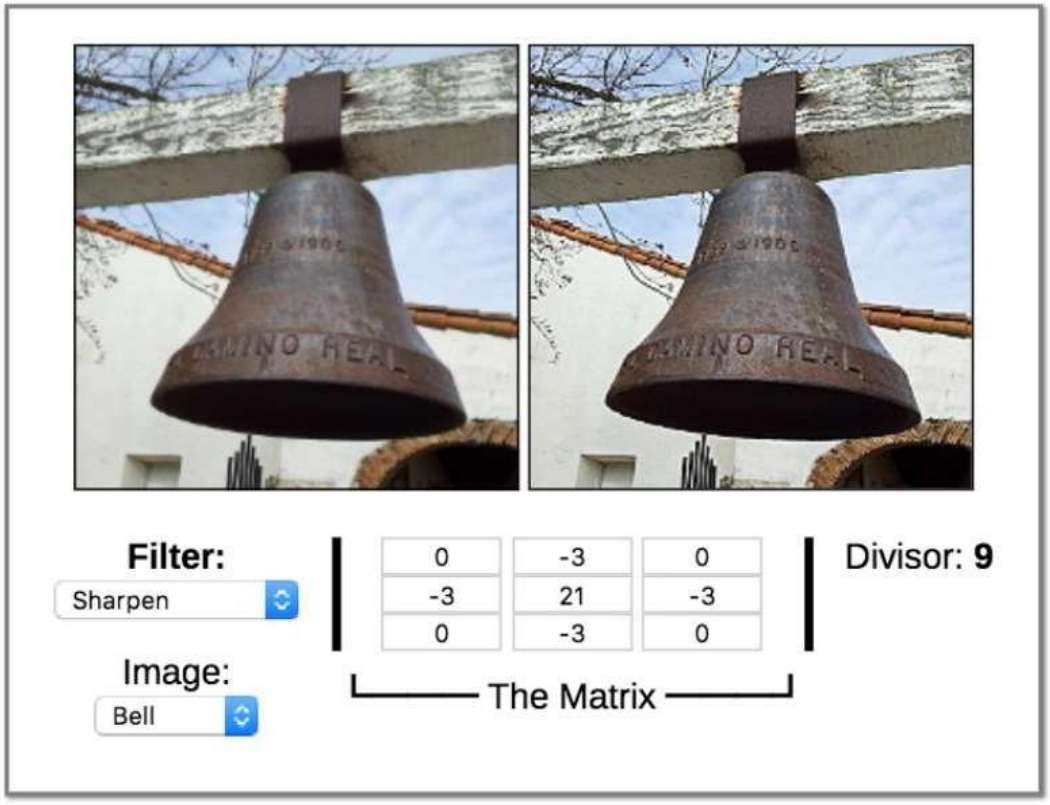
Laplacian Scaled

Convolutional Neural Networks
Convolution Operation

0	1	0
1	-4	1
0	1	0

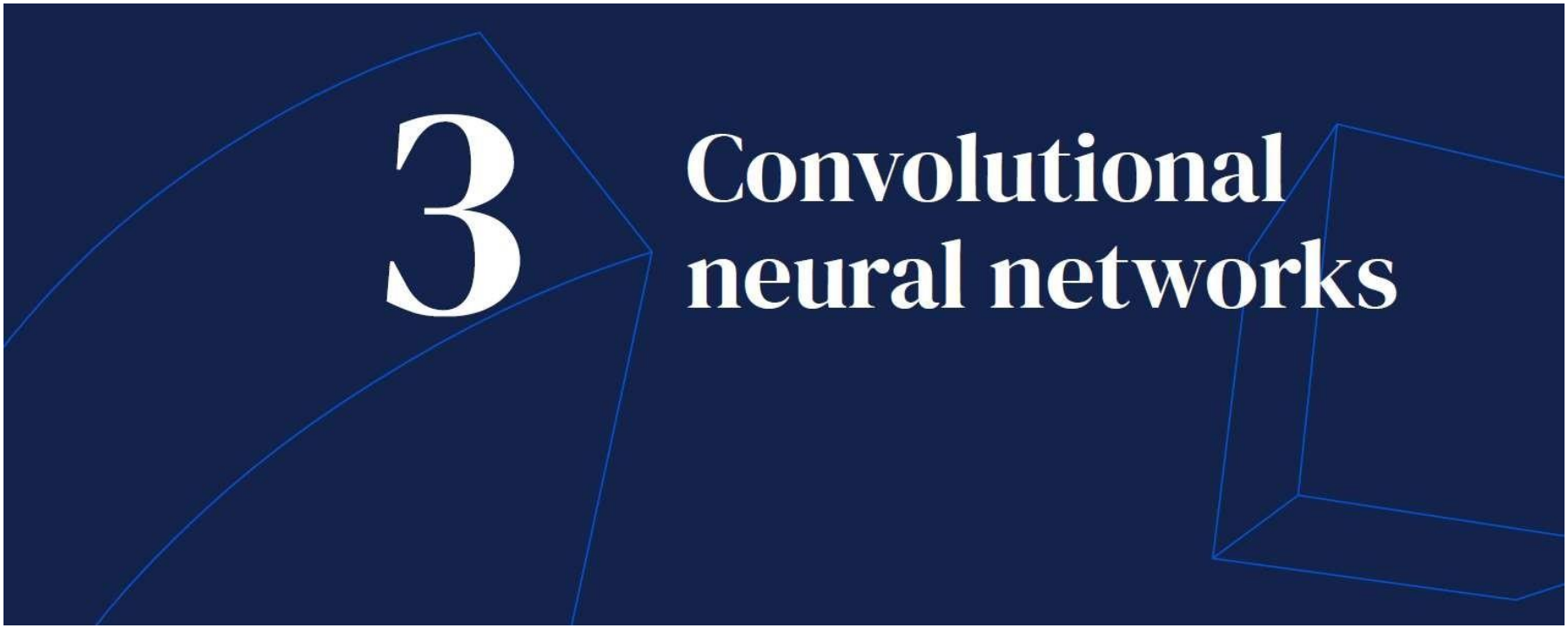
1	1	1
1	-8	1
1	1	1

Convolutional Neural Networks
Convolution Operation



Demo: http://beej.us/blog/data/convolution-image-processing/

Convolutional Neural Networks
Convolution Operation



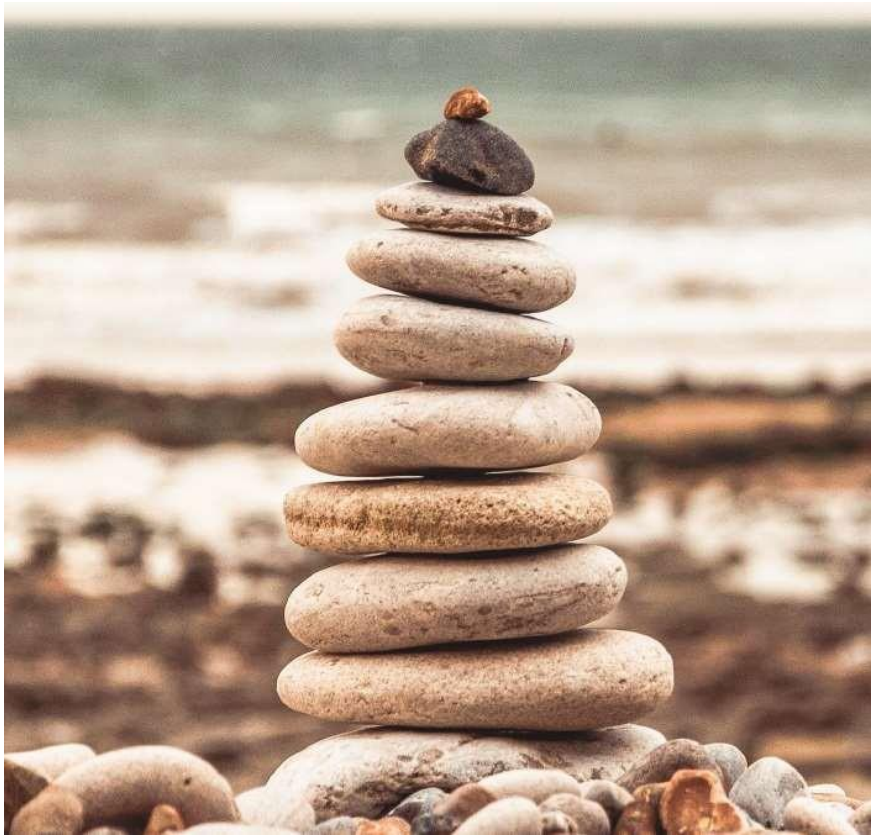
Convolutional Neural Networks
Convolution Operation

Stacking the building blocks

CNNs or ConvNets

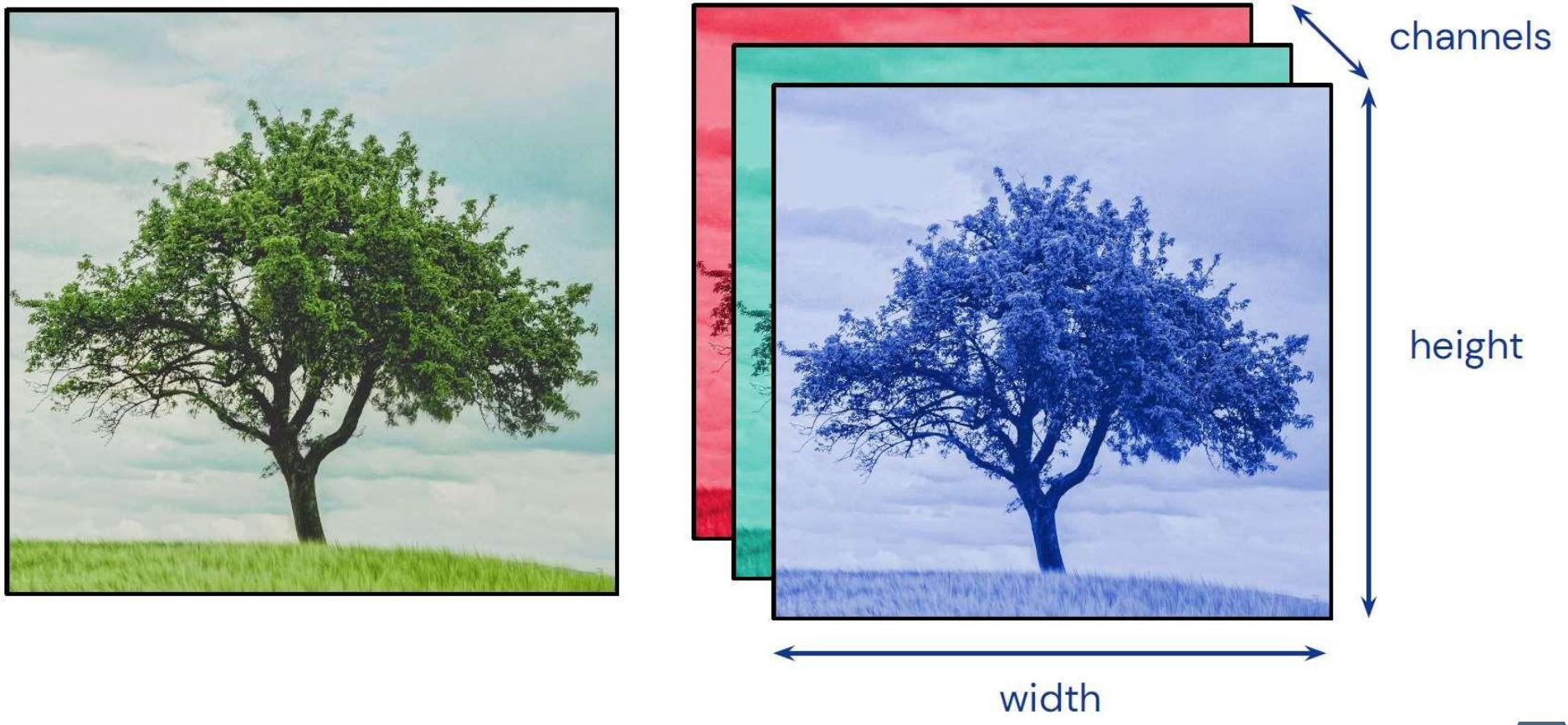
Up to to 100s of layers

Alternate convolutions and pooling to create a hierarchy



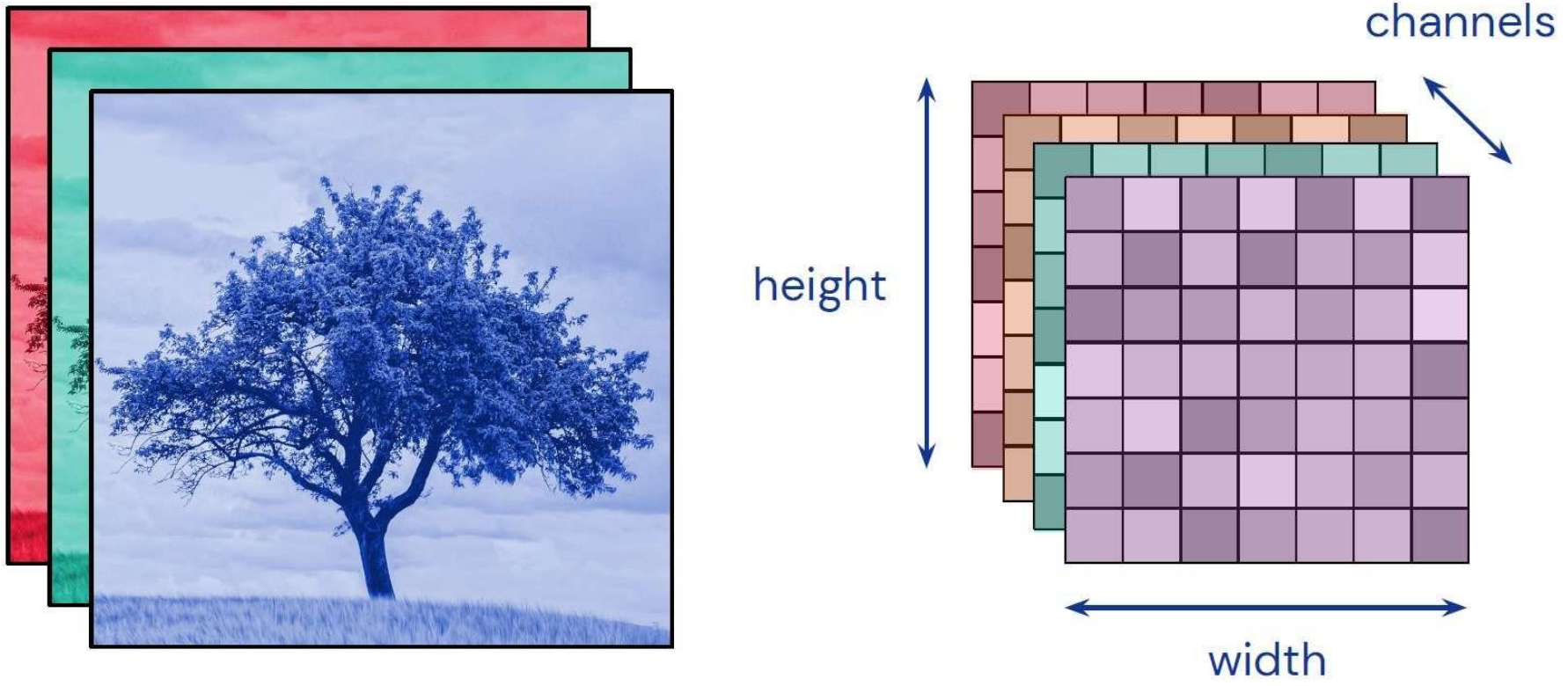
Convolutional Neural Networks
Convolution Operation

Inputs and outputs are tensors

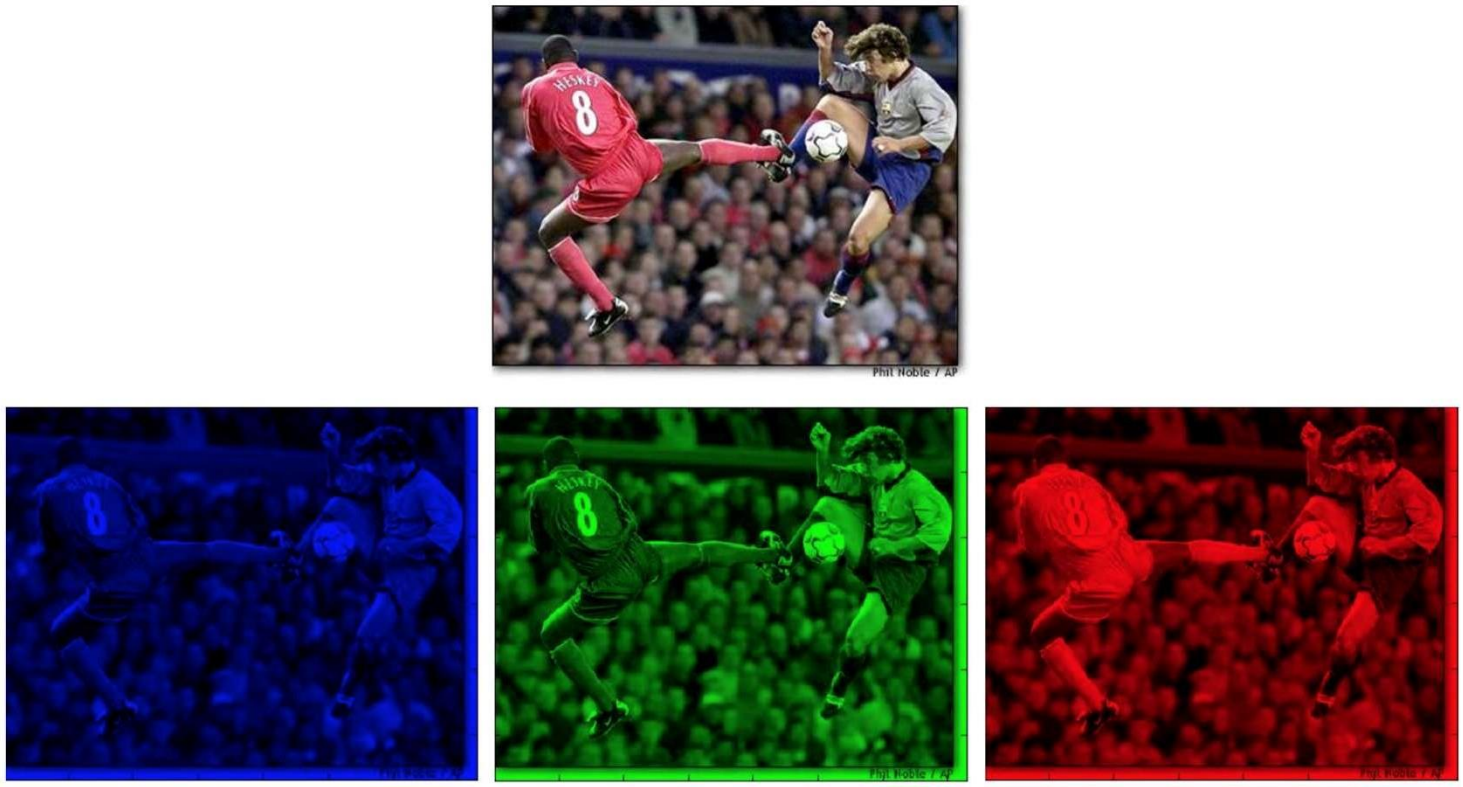


Convolutional Neural Networks
Convolution Operation

Inputs and outputs are tensors



Convolutional Neural Networks
Convolution Operation



Convolutional Neural Networks
Convolution Operation

An image contains a discrete number of pixels

Example:

Value of the pixels

"grayscale"

(or "intensity"):

[0,255]

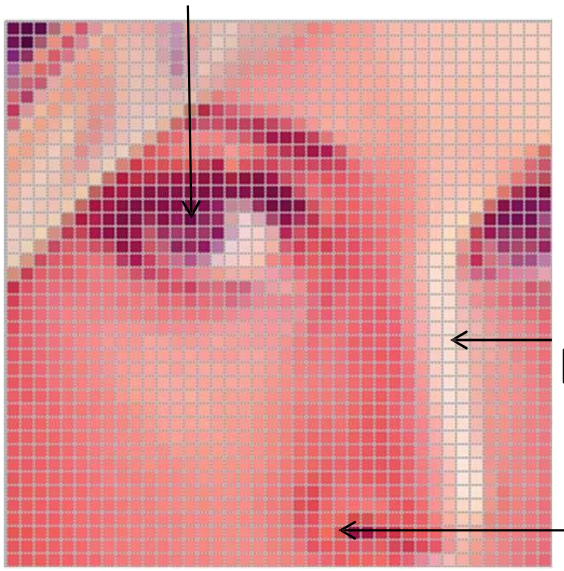
"color"

RGB: [R, G, B]

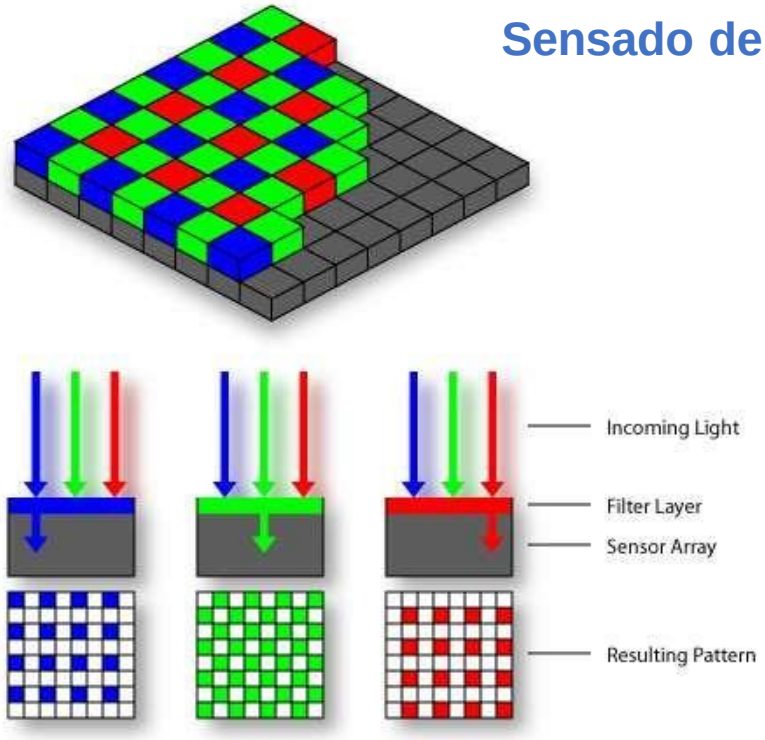
Lab: [L, a, b]

HSV: [H, S, V]

Images are
Sampled and Quantizied



Convolutional Neural Networks
Convolution Operation

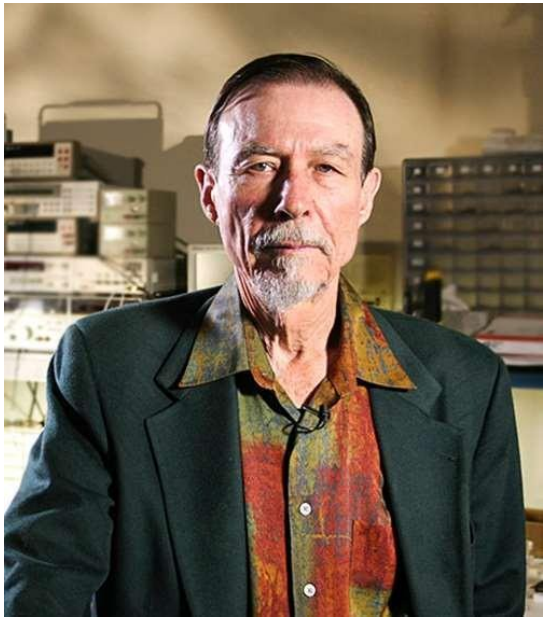


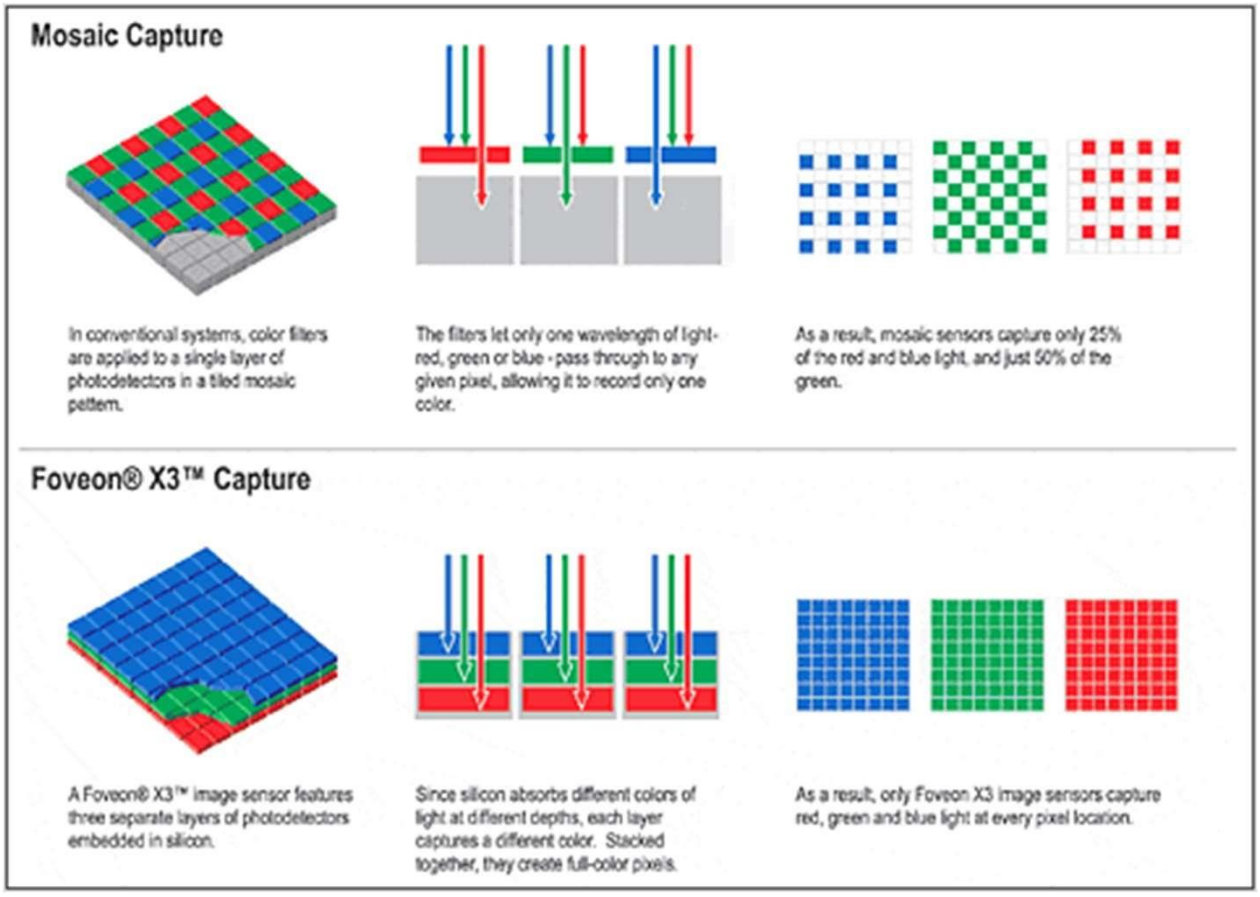
Sensado de colores practico: arreglo de Bayer

Estimar valores R, G, B en celda G a parir de pixeles aledaños

Convolutional Neural Networks
Convolution Operation

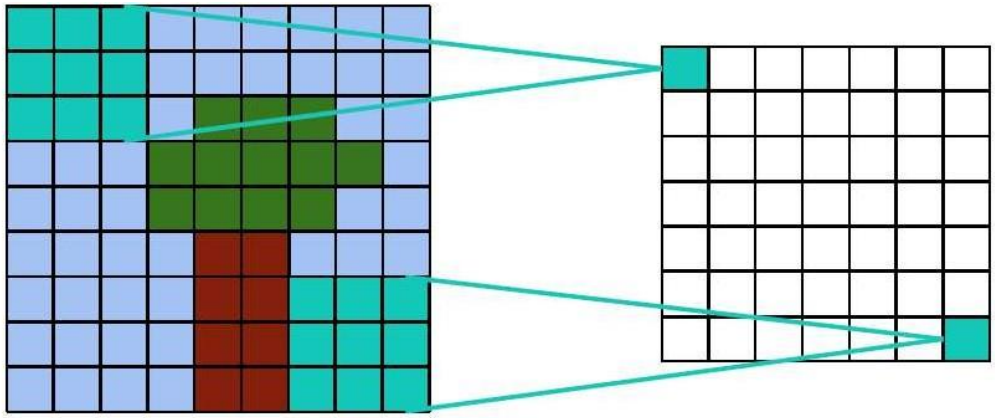
Carver Mead





Convolutional Neural Networks

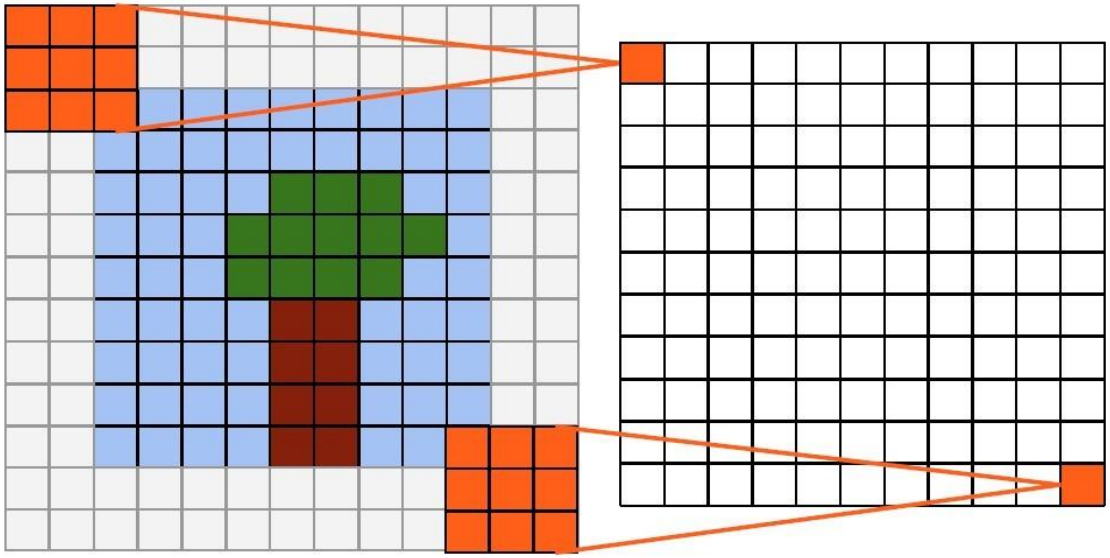
Convolution Operation: Variants



Valid convolution: output size = input size − kernel size + 1

Convolutional Neural Networks

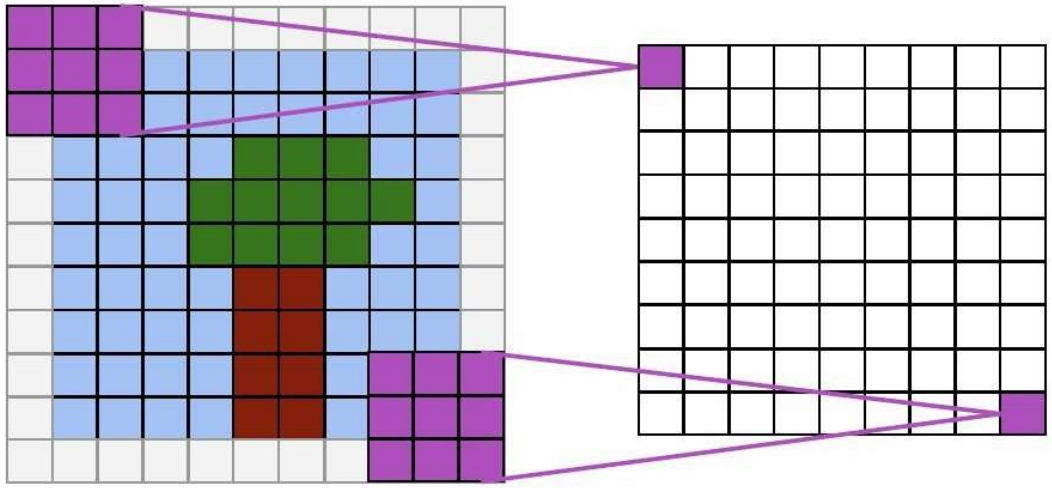
Convolution Operation: Variants



Full convolution: output size = input size + kernel size − 1

Convolutional Neural Networks

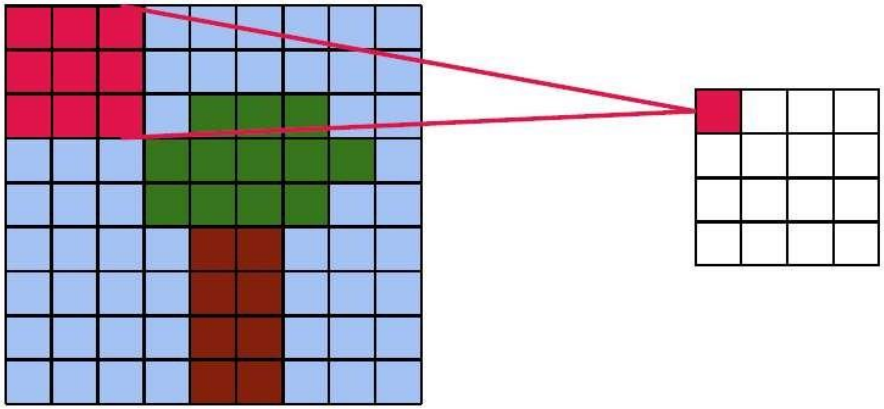
Convolution Operation: Variants



Same convolution: output size = input size

Convolutional Neural Networks

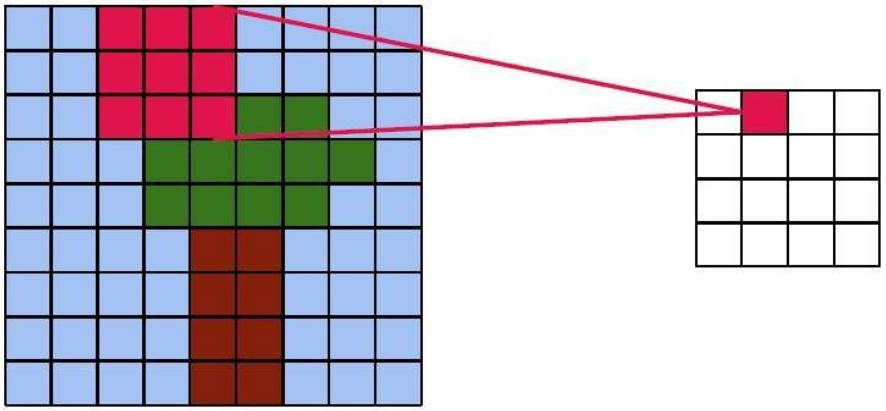
Convolution Operation: Variants



Strided convolution: kernel slides along the image with a step > 1

Convolutional Neural Networks

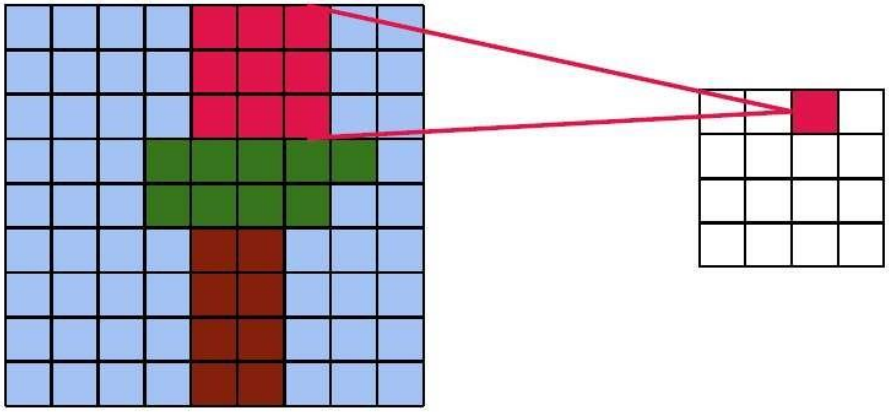
Convolution Operation: Variants



Strided convolution: kernel slides along the image with a step > 1

Convolutional Neural Networks

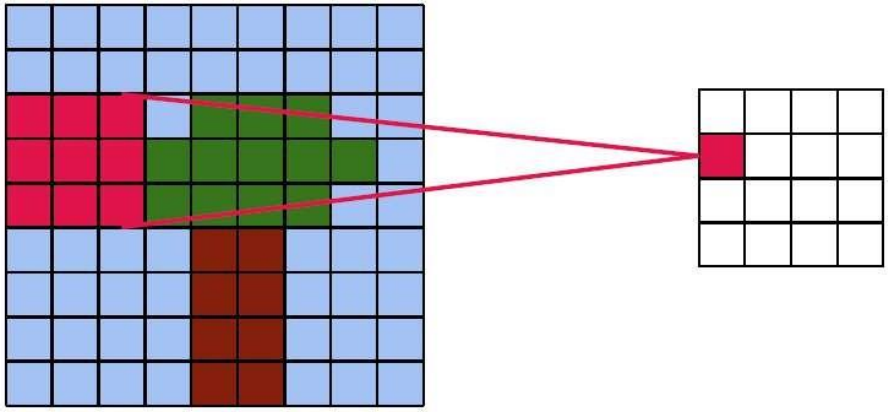
Convolution Operation: Variants



Strided convolution: kernel slides along the image with a step > 1

Convolutional Neural Networks

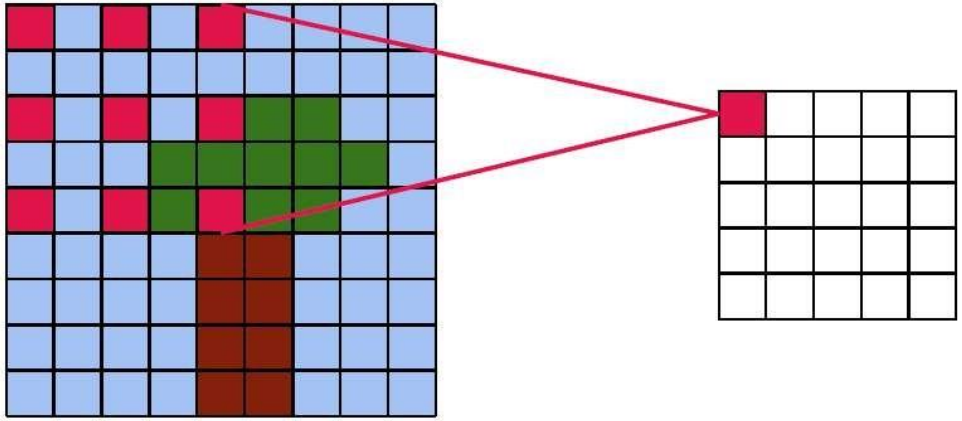
Convolution Operation: Variants



Strided convolution: kernel slides along the image with a step > 1

Convolutional Neural Networks

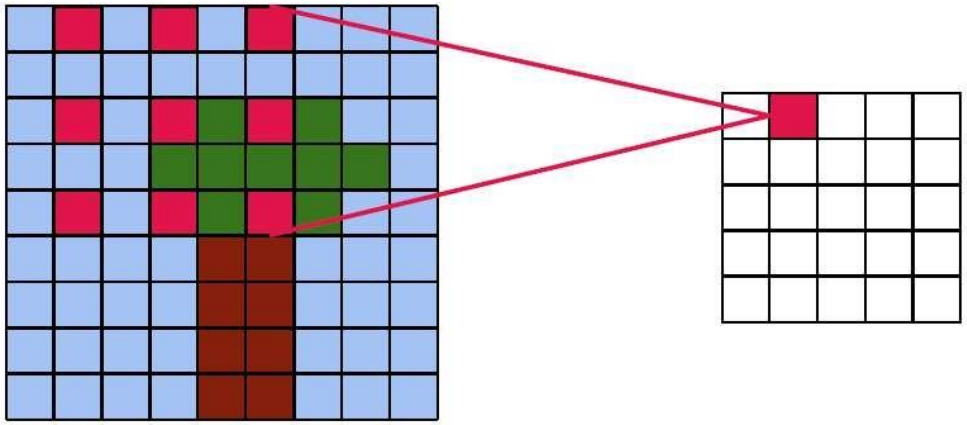
Convolution Operation: Variants



Dilated convolution: kernel is spread out, step > 1 between kernel elements

Convolutional Neural Networks

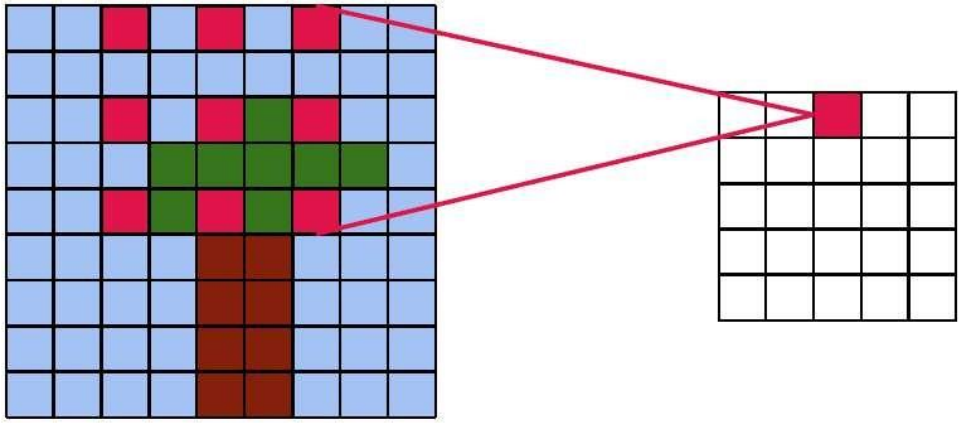
Convolution Operation: Variants



Dilated convolution: kernel is spread out, step > 1 between kernel elements

Convolutional Neural Networks

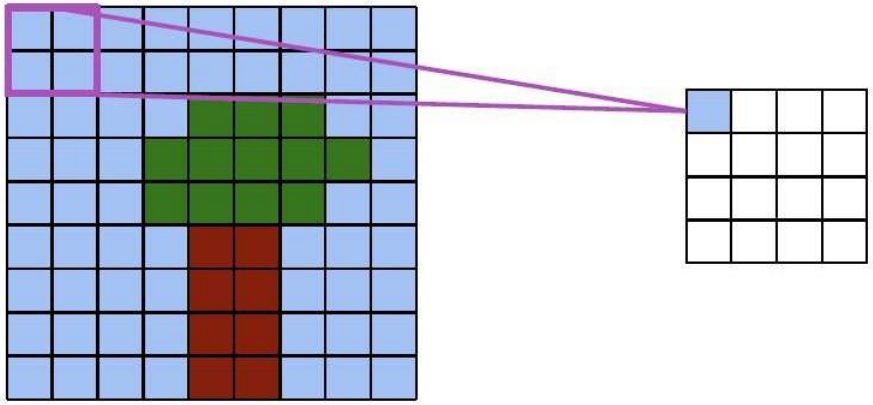
Convolution Operation: Variants



Dilated convolution: kernel is spread out, step > 1 between kernel elements

Convolutional Neural Networks

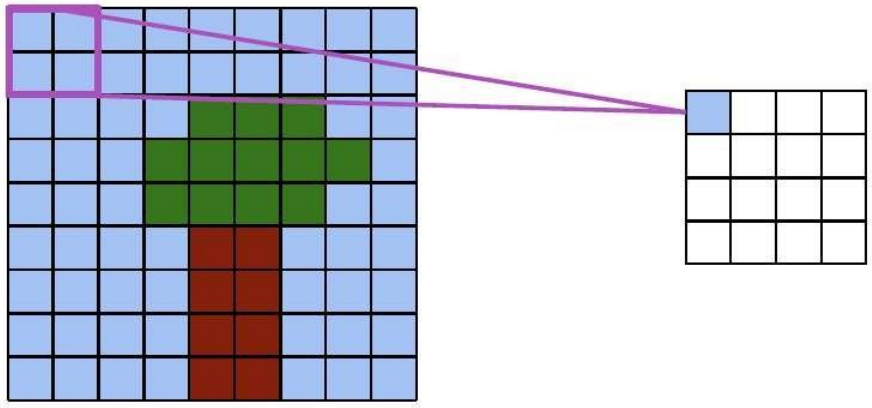
Convolution Operation: Variants



Pooling: compute mean or max over small windows to reduce resolution

Convolutional Neural Networks

Convolution Operation: Variants

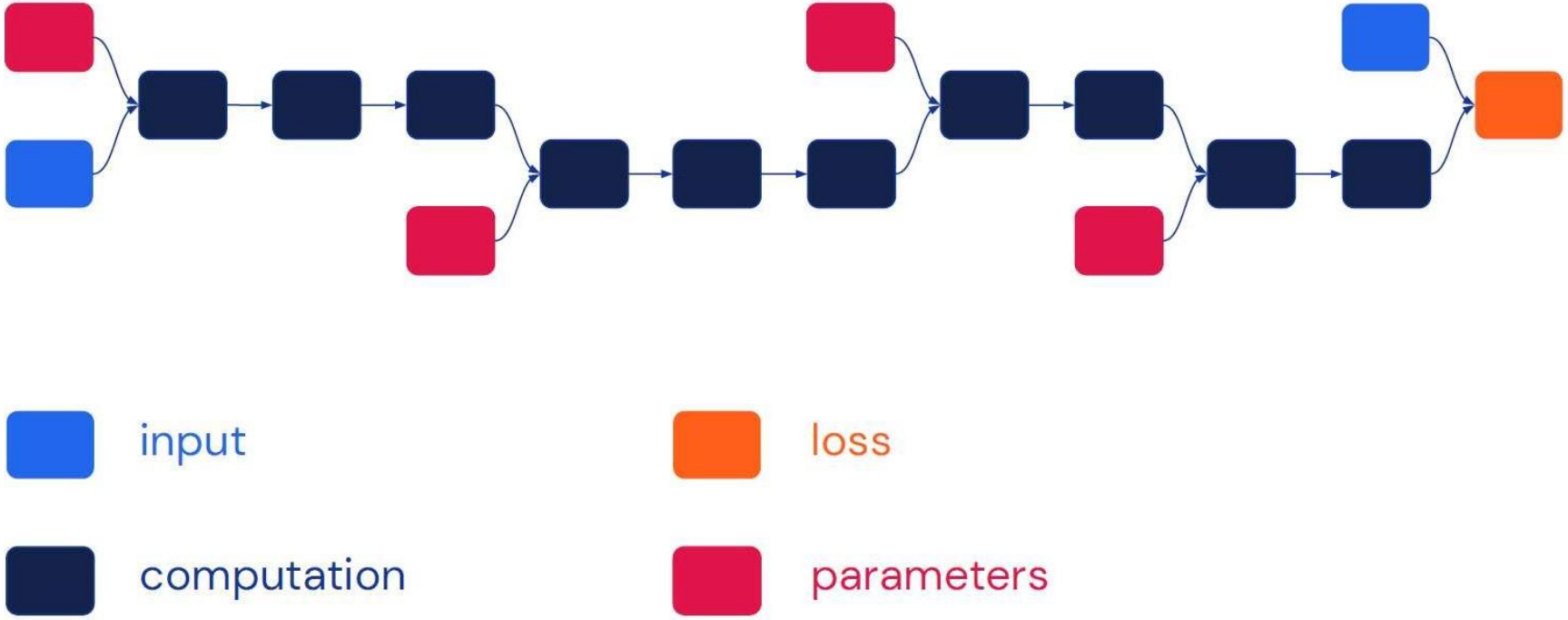


Pooling: compute mean or max over small windows to reduce resolution

Convolutional Neural Networks

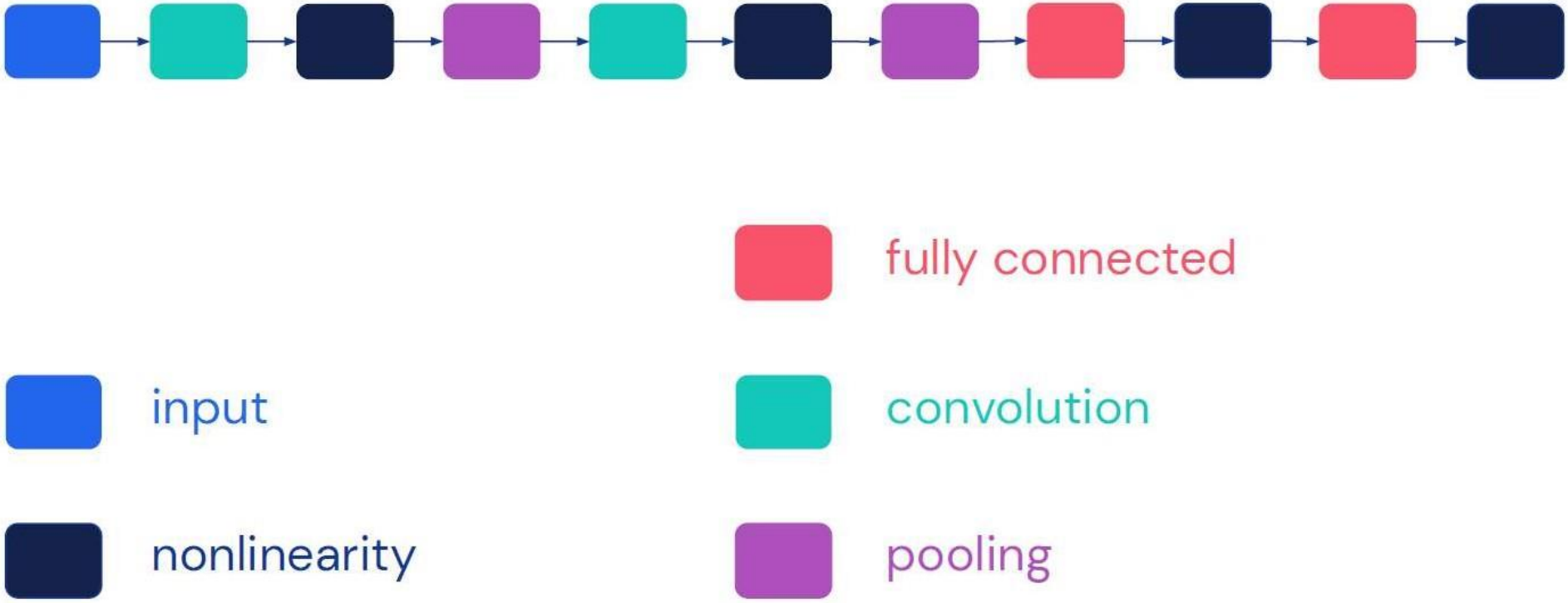
Stacking the building blocks

Neural networks as computational graphs



Convolutional Neural Networks

Stacking the building blocks



Convolutional Neural Networks

Some Case Studies: Le-Net5 (1998)

Convolutional Neural Networks

Some Case Studies: Le-Net5 (1998)

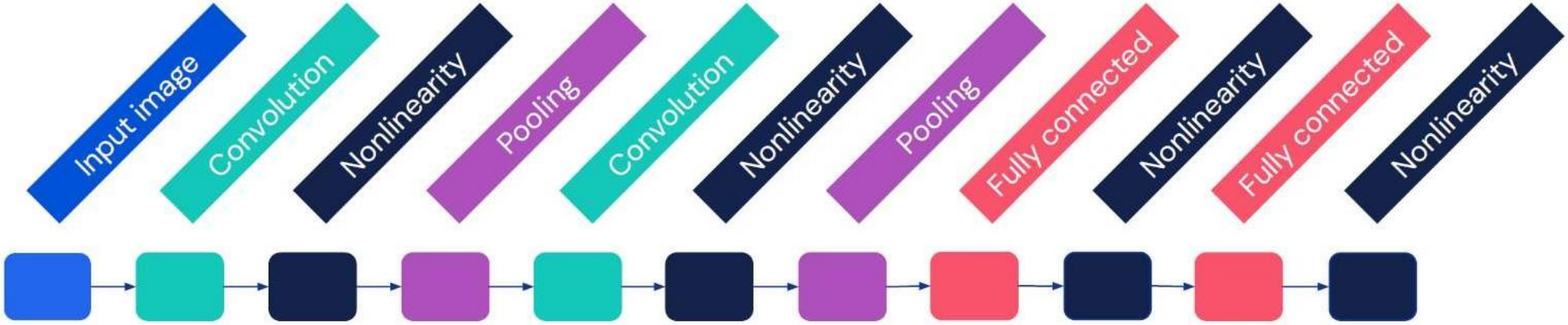
Architecture of **LeNet-5**, a convnet for handwritten digit recognition

Want to learn more?

Lecun, Y.; Bottou, L.; Bengio, Y.; Haffner, P.
Gradient-based learning applied to document recognition
Proceedings of the IEEE 86(11) **(1998)**

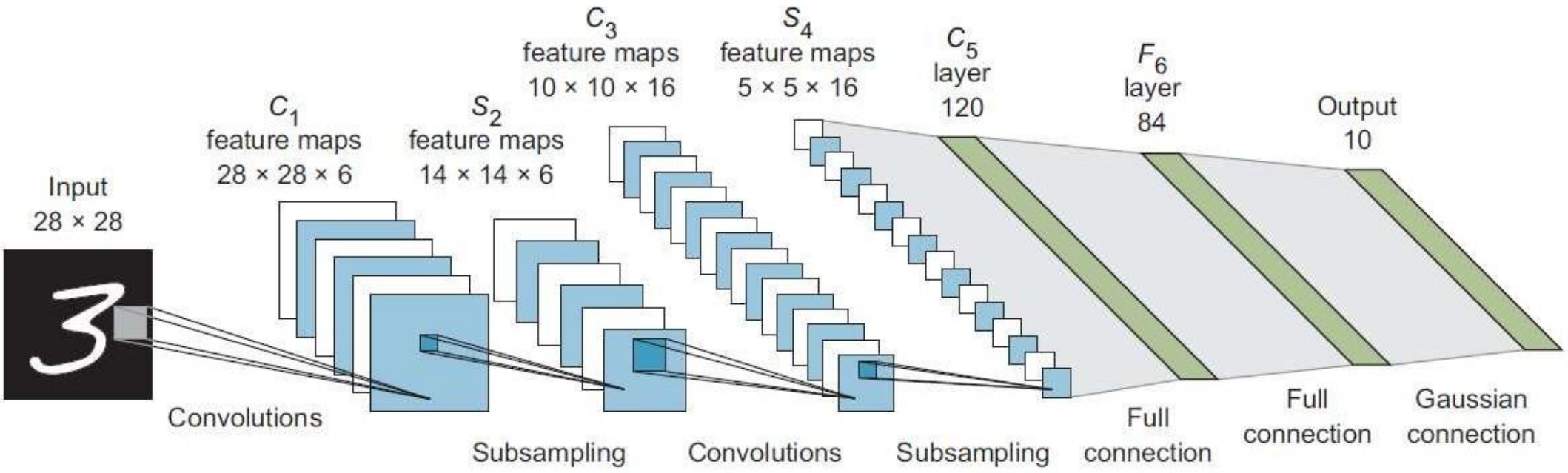
Convolutional Neural Networks

Some Case Studies: Le-Net5 (1998)



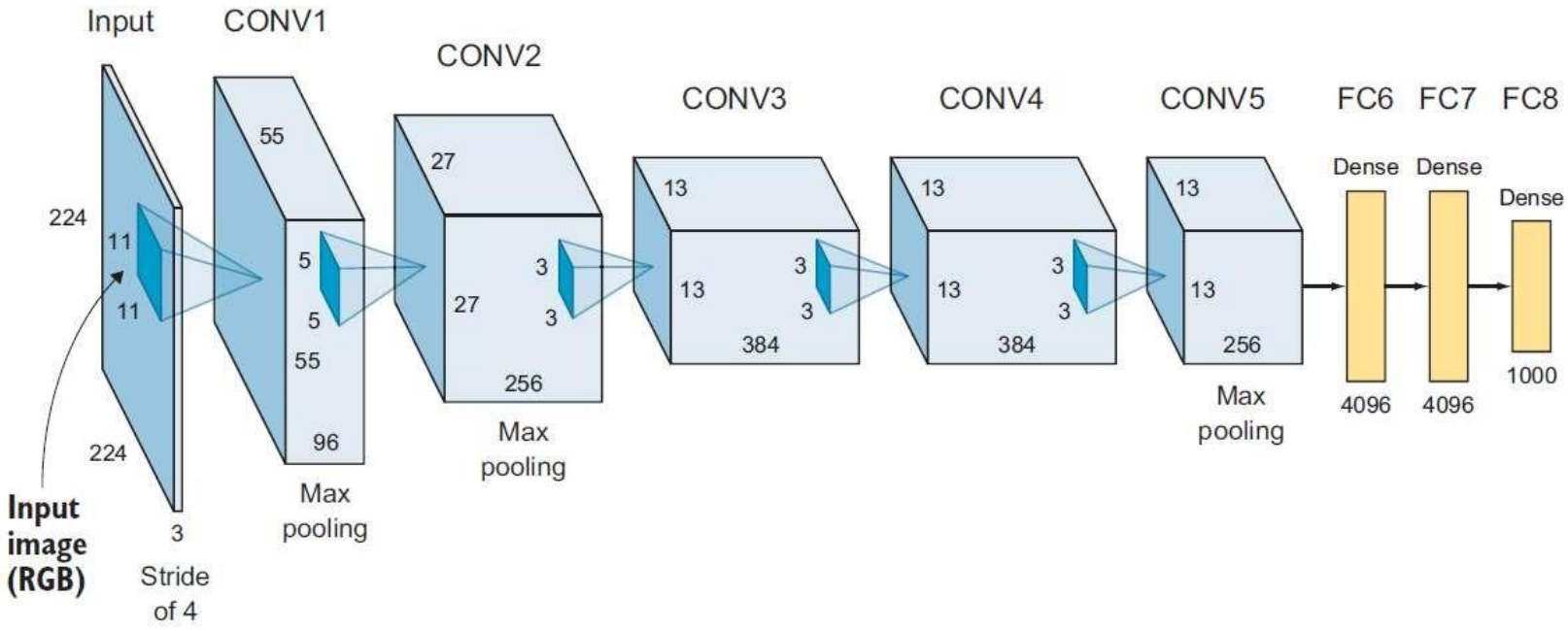
Convolutional Neural Networks

Some Case Studies: Le-Net5 (1998)



Convolutional Neural Networks

Some Case Studies: AlexNet (2012)



Want to learn more?

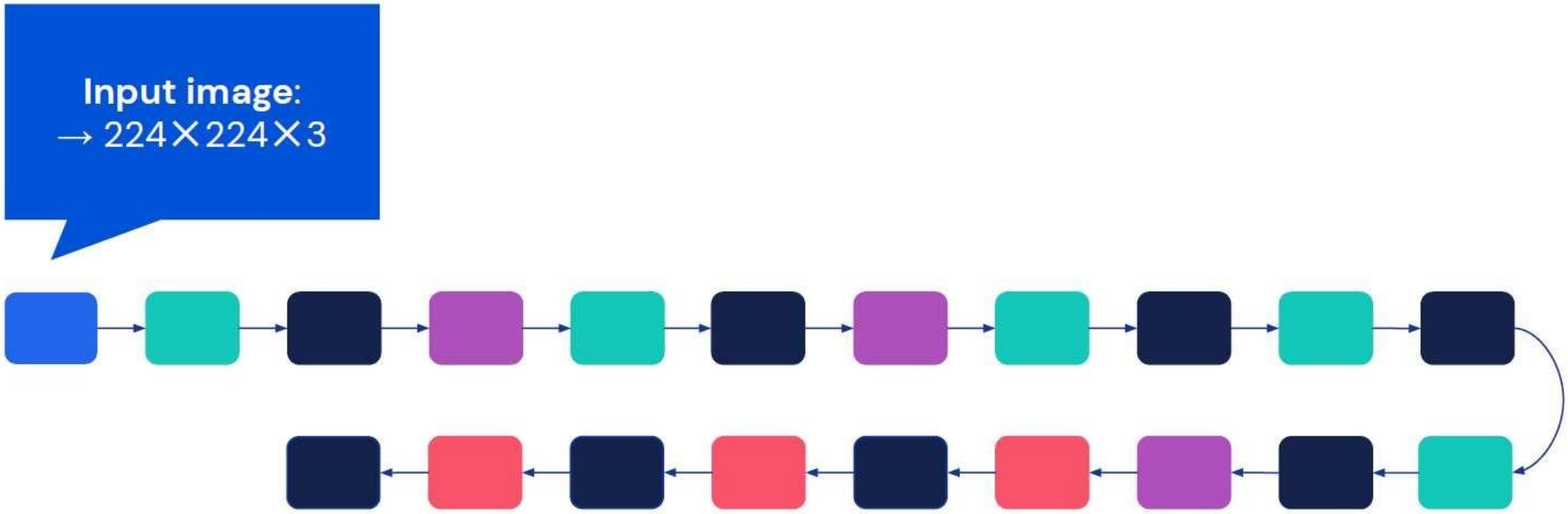
Krizhevsky, A.; Sutskever, I.; Hinton, G.E.
ImageNet classification with deep convolutional neural networks
Neural Information Processing Systems (2012)

Architecture: 8 layers, ReLU, dropout, weight decay

Infrastructure: large dataset, trained 6 days on 2 GPUs

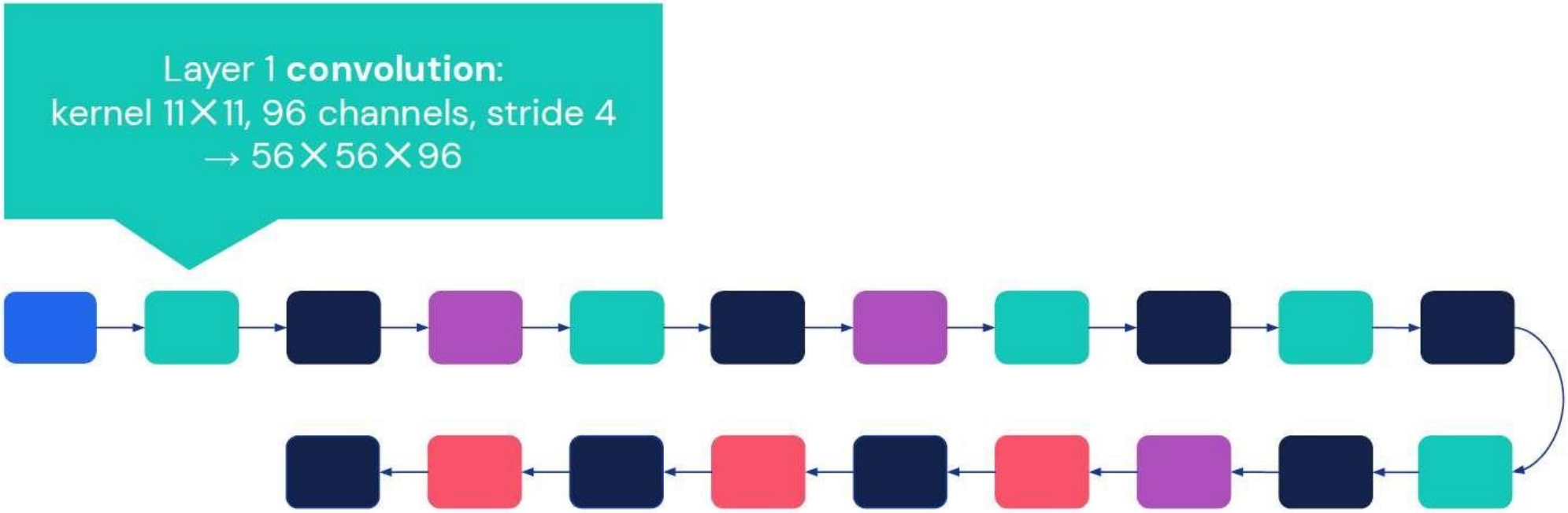
Convolutional Neural Networks

Some Case Studies: AlexNet (2012)



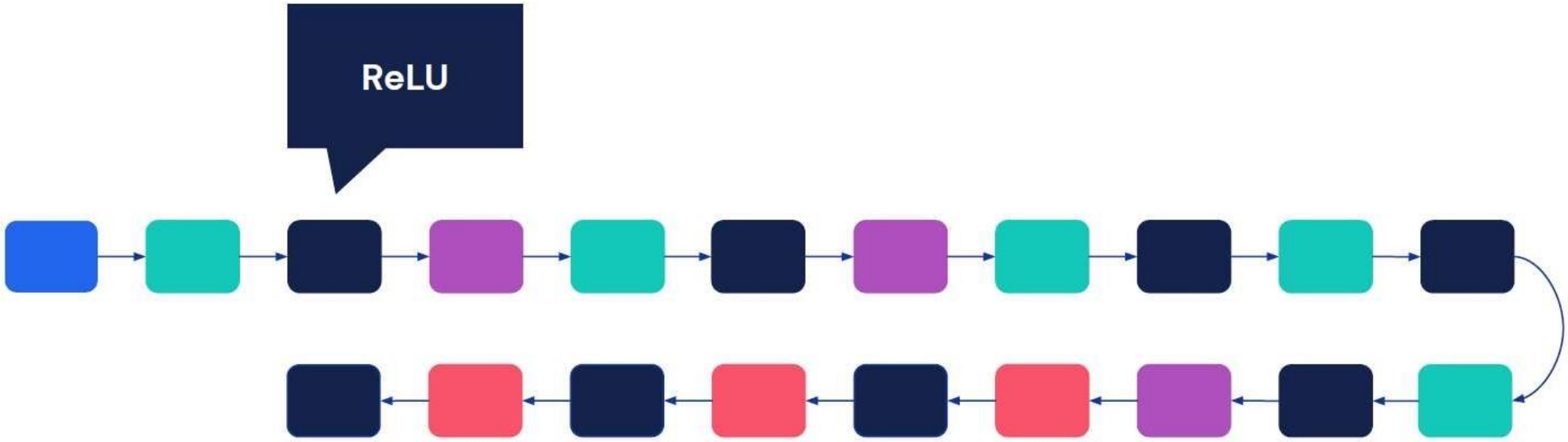
Convolutional Neural Networks

Some Case Studies: AlexNet (2012)



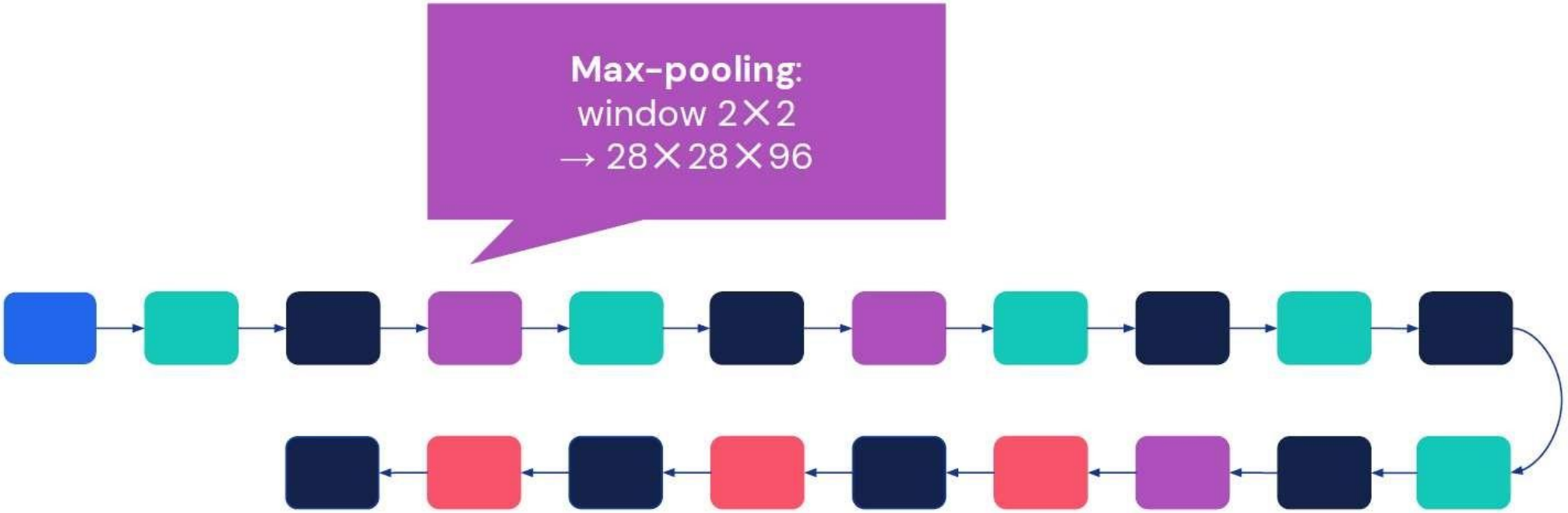
Convolutional Neural Networks

Some Case Studies:AlexNet (2012)



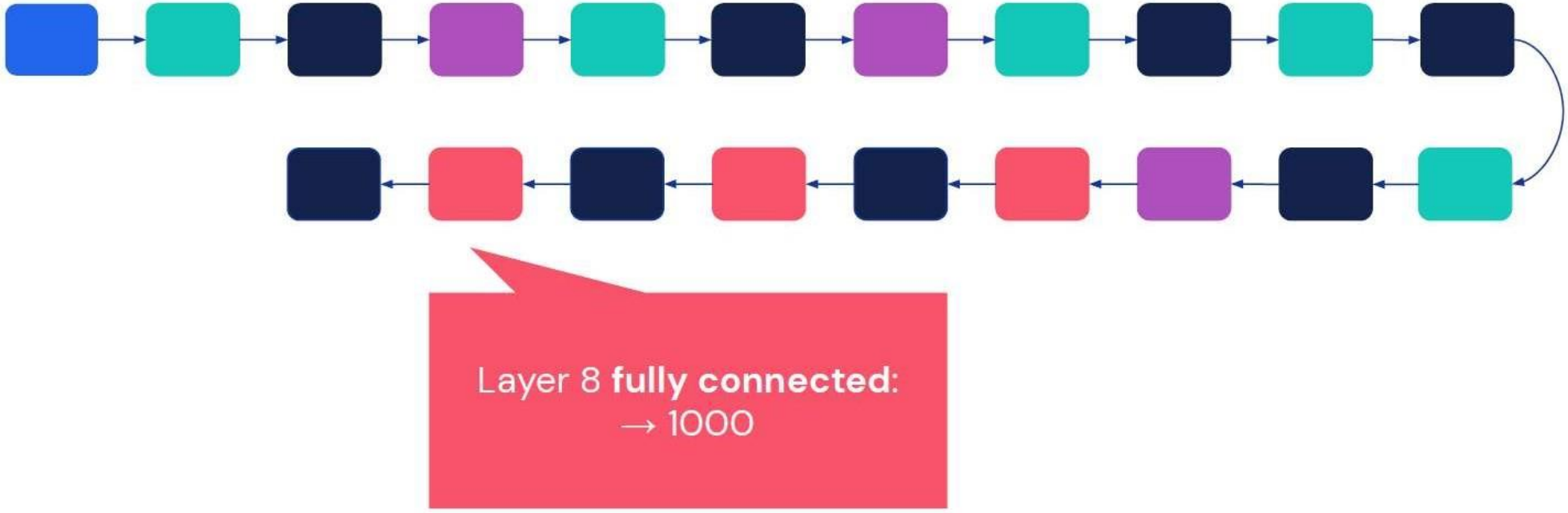
Convolutional Neural Networks

Some Case Studies: AlexNet (2012)



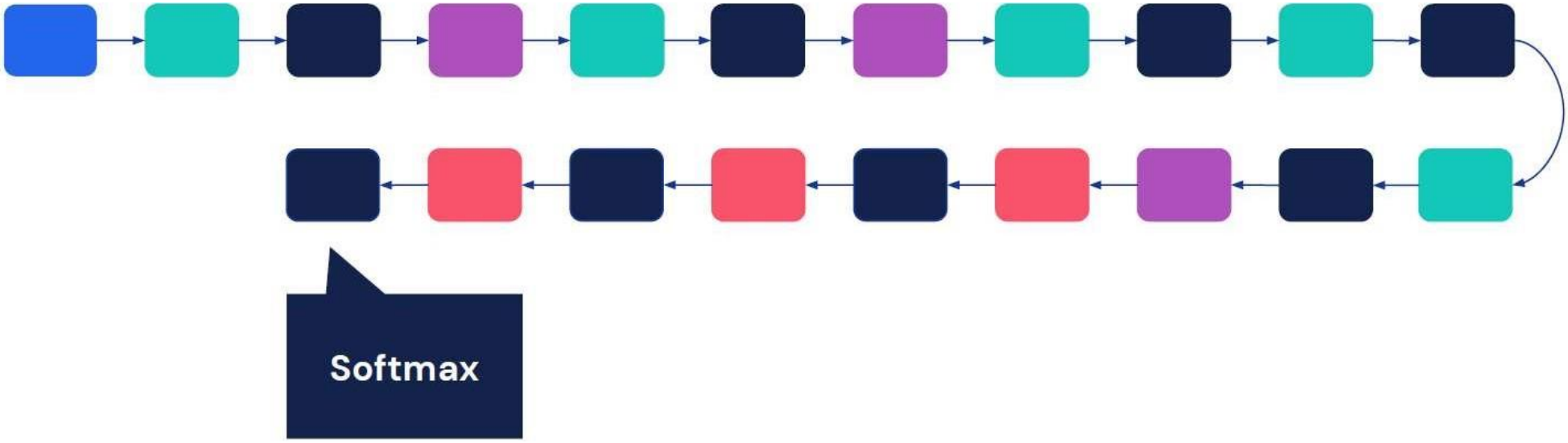
Convolutional Neural Networks

Some Case Studies: AlexNet (2012)



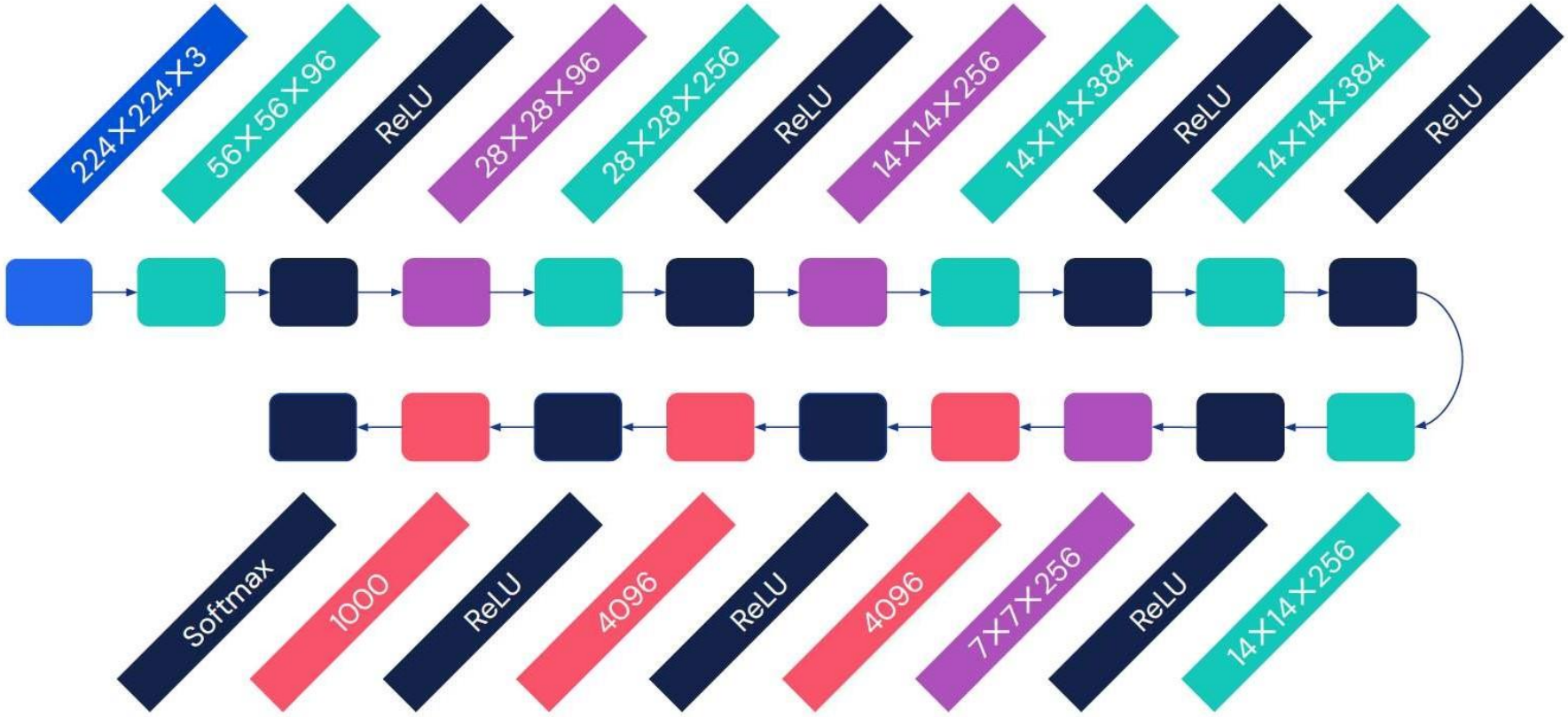
Convolutional Neural Networks

Some Case Studies:AlexNet (2012)



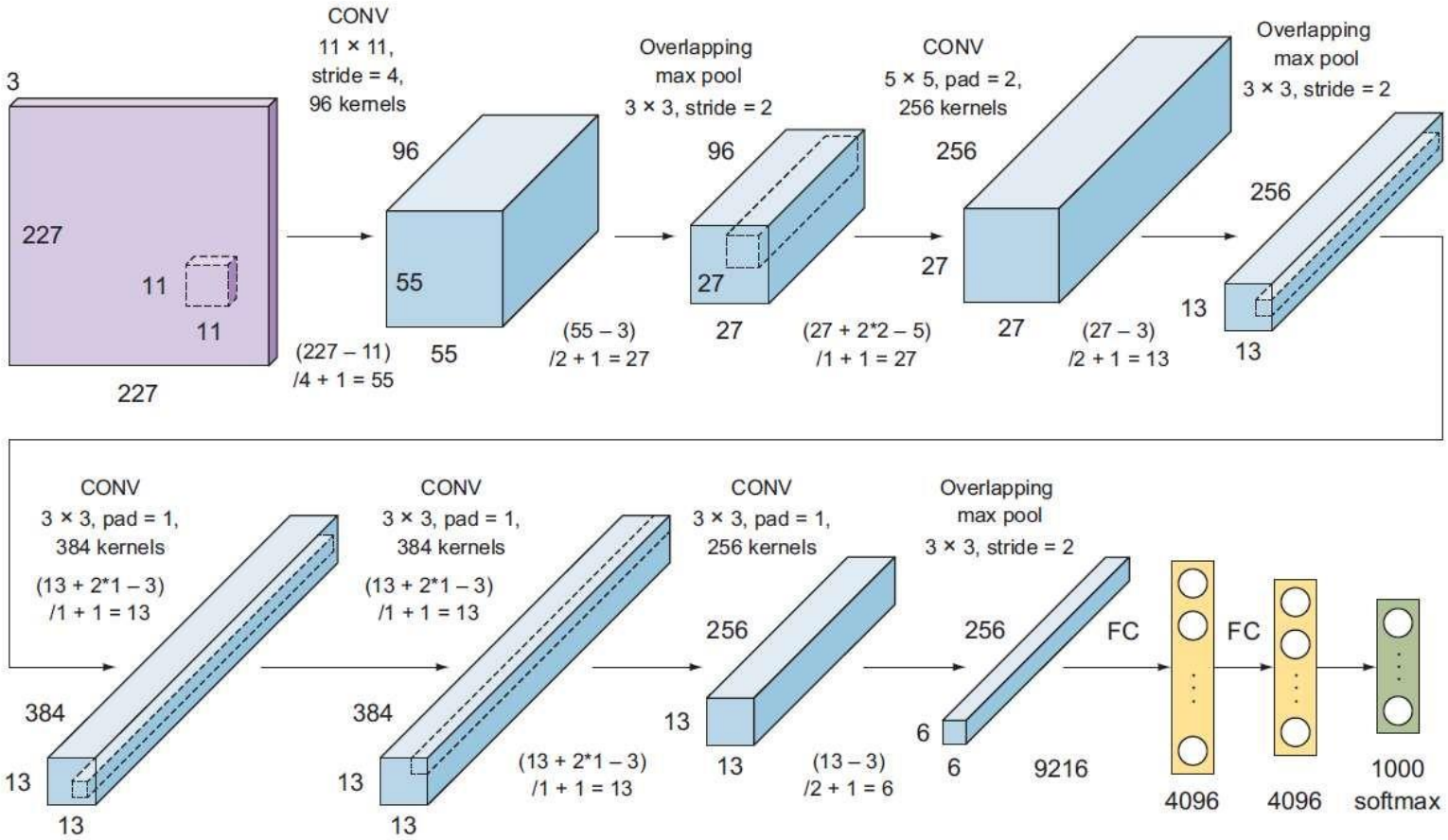
Convolutional Neural Networks

Some Case Studies: AlexNet (2012)



Convolutional Neural Networks

Some Case Studies:AlexNet (2012)



Convolutional Neural Networks

Some Case Studies: AlexNet (2012)

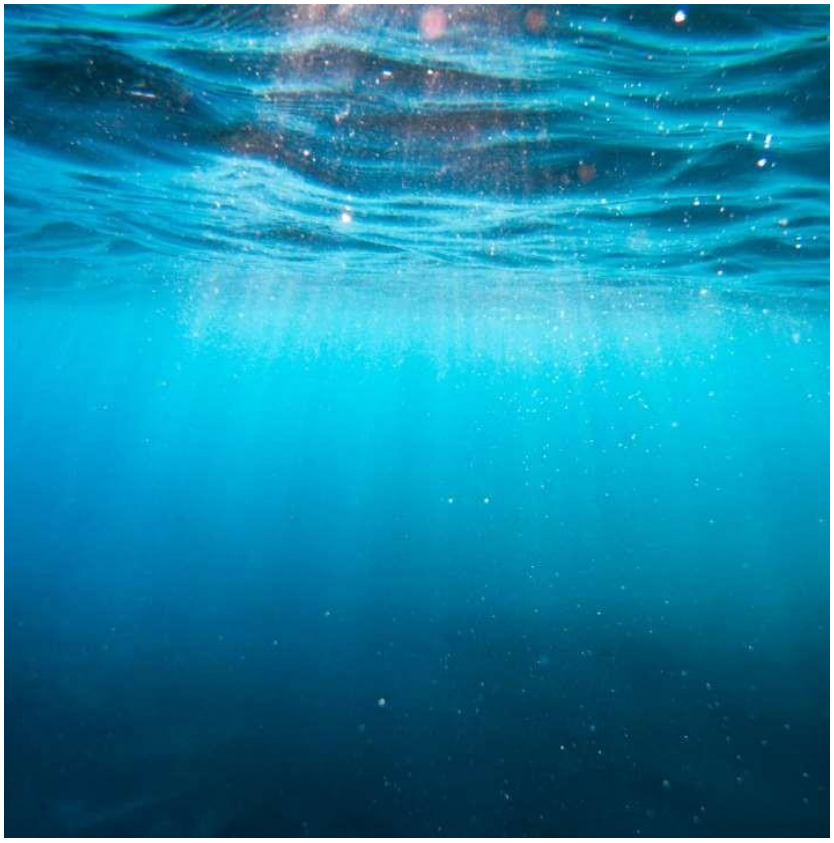
Stacking the building blocks

Each layer is a linear classifier by itself

More layers: more non-linearities

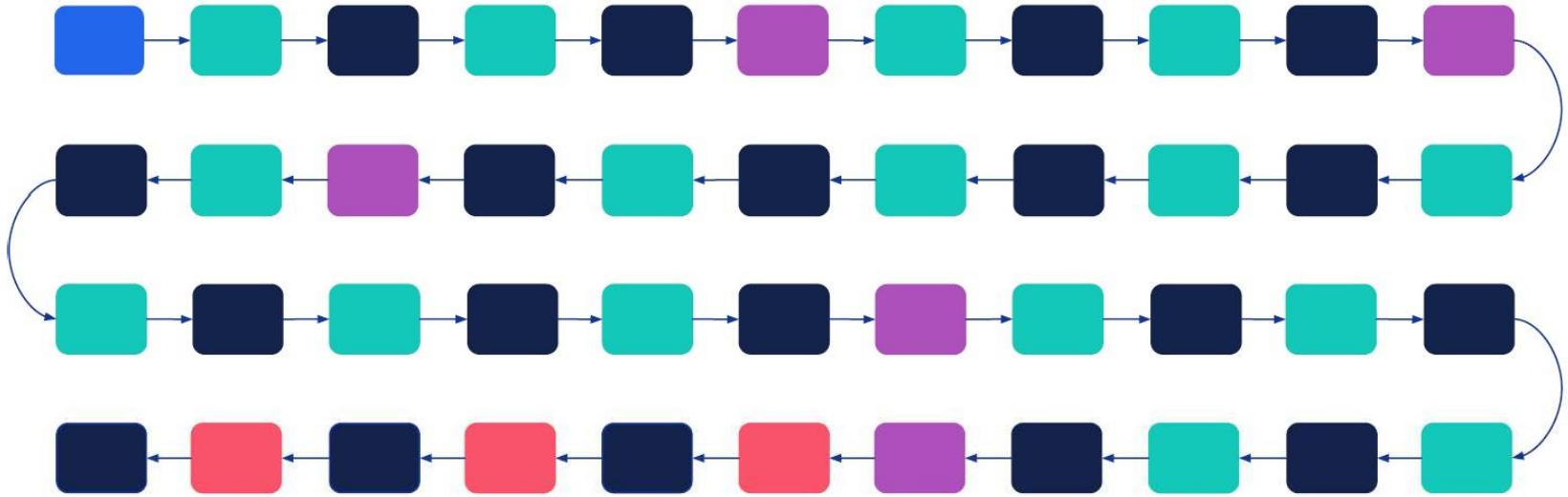
What limits the number of layers of ConvNets?

Computation, Optimization, Dissection



Convolutional Neural Networks

Some Case Studies: VGGNet (2014)



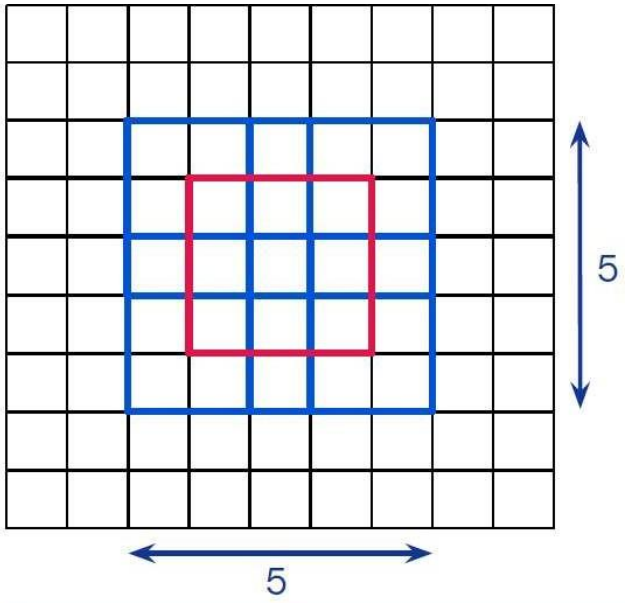
Want to learn more?

Simonyan, K.; Zisserman, A.
Very deep convolutional networks for large-scale image recognition International Conference on Learning Representations (**2015**)

Stack many convolutional layers before pooling
Use "same" convolutions to avoid resolution reduction

Convolutional Neural Networks

Some Case Studies:VGGNet (2014)



Stacking 3×3 kernels

1st 3×3 conv. layer

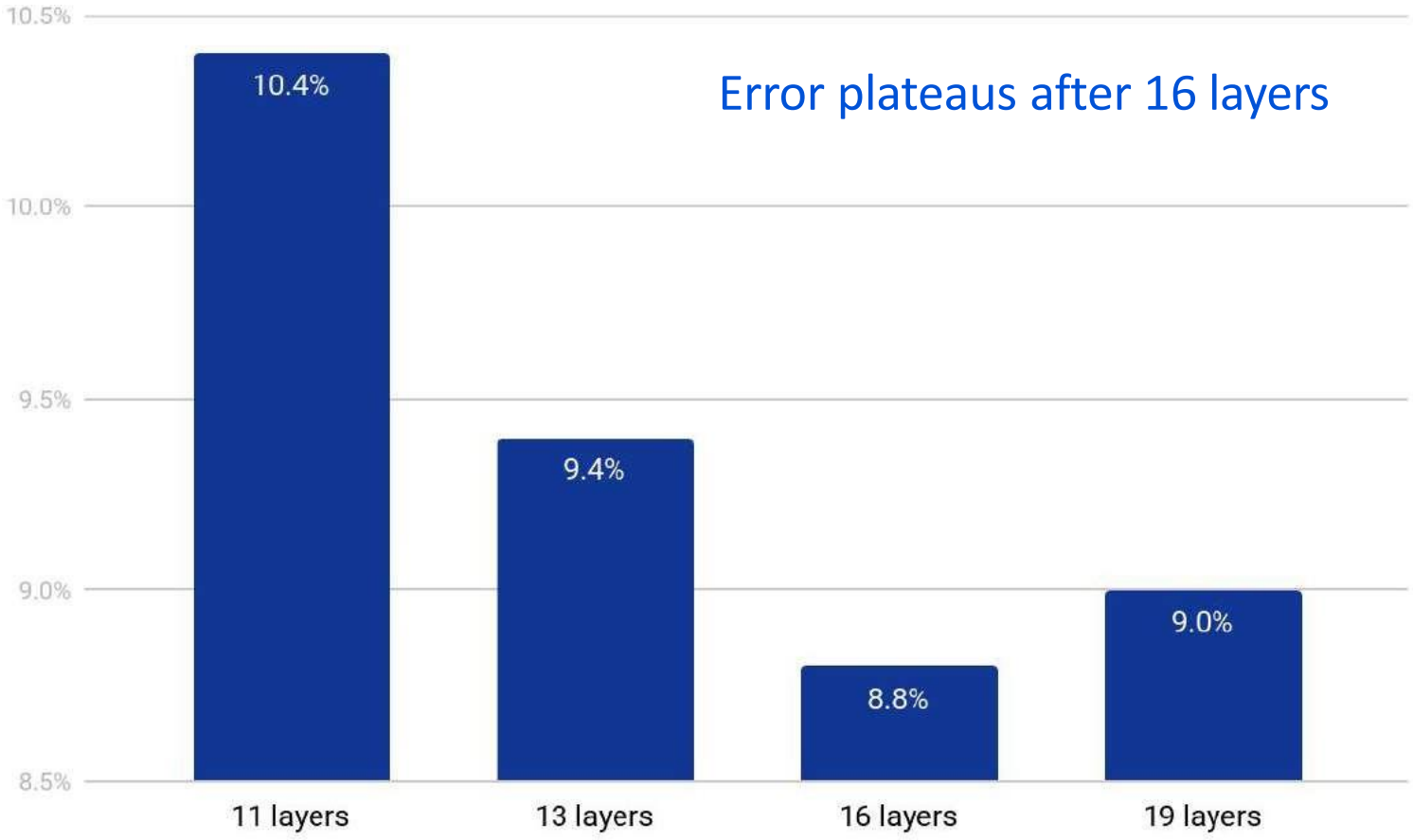
2nd 3×3 conv. layer

Architecture: up to 19 layers, 3×3 kernels only, "same" convolutions

Infrastructure: trained for 2-3 weeks on 4 GPUs (data parallelism)

Convolutional Neural Networks

Some Case Studies: VGGNet (2014)



Convolutional Neural Networks

Some Case Studies: VGGNet (2014)

Challenges of depth

Computational Complexity
Optimisation difficulties

Improving optimisation

Careful Initialisation
Sophisticated Optimizers
Normalization Layers
Network Design

Convolutional Neural Networks

Some Case Studies: GoogleLenet (2014)

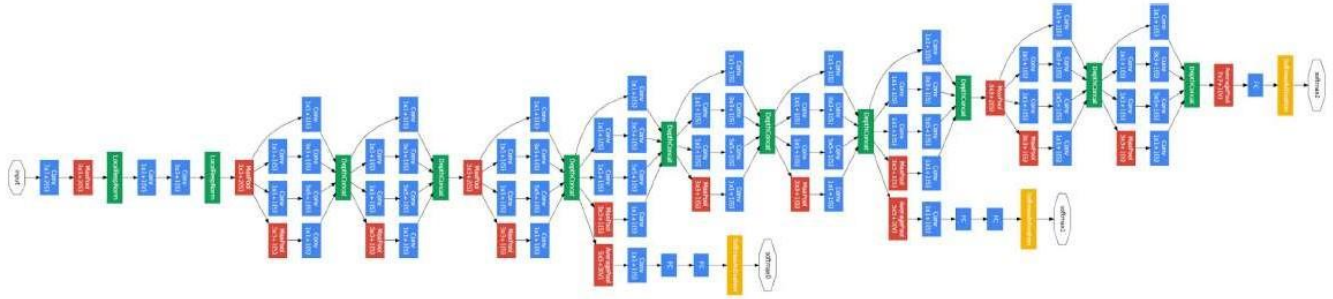
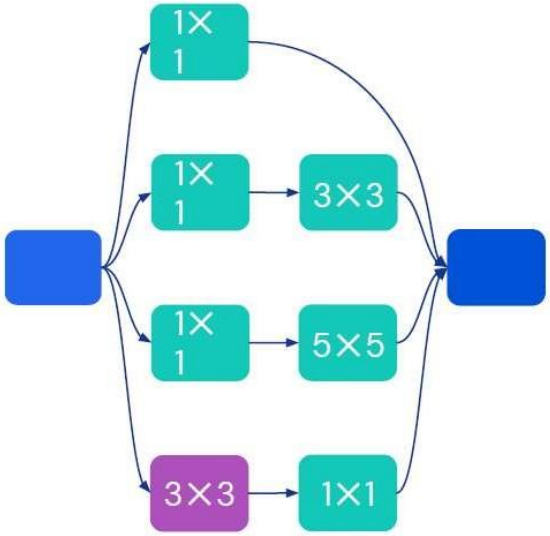


Figure from Szegedy et al. (2015)

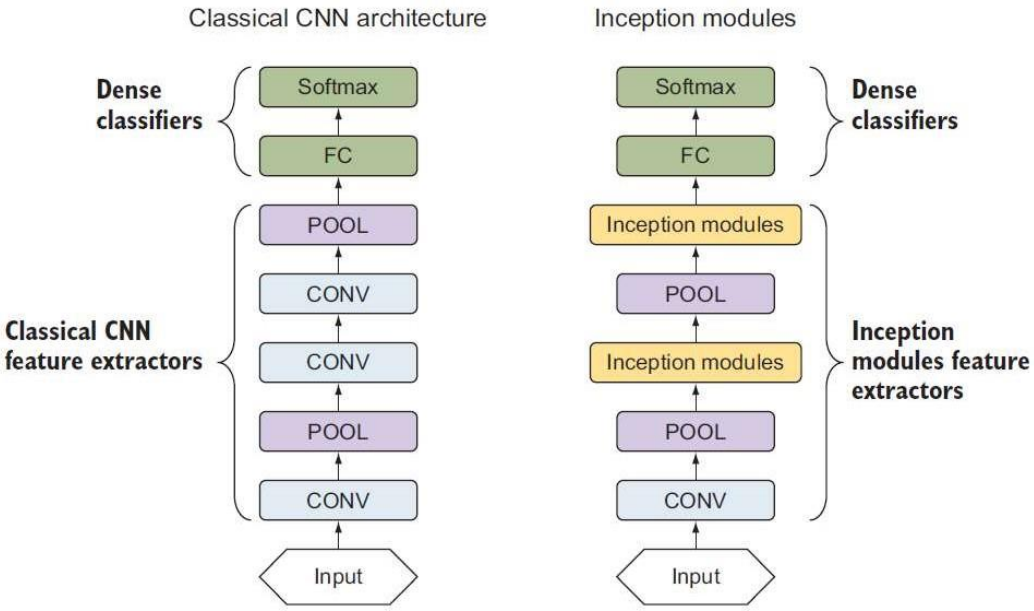


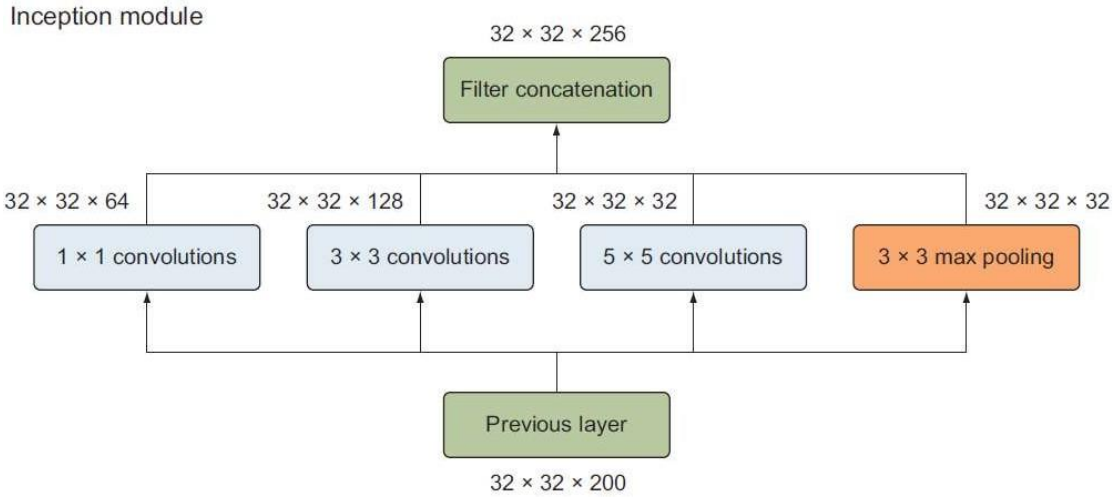
Want to learn more?

Szegedy, C. et al.
Going deeper with convolutions IEEE conference on computer vision and pattern recognition (**2015**)

Convolutional Neural Networks

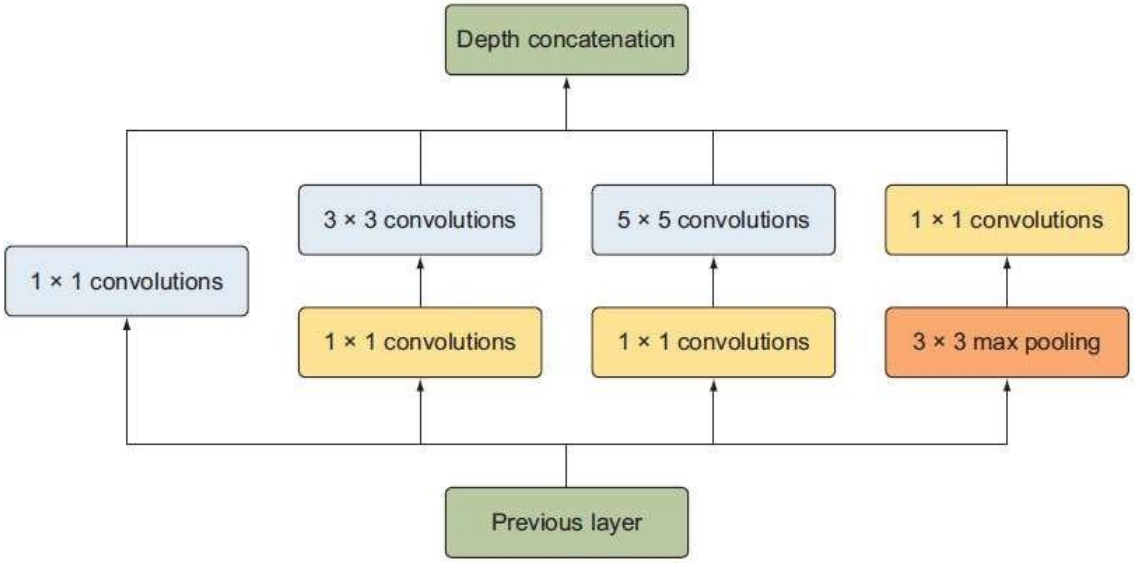
Some Case Studies:GoogleLenet (2014)

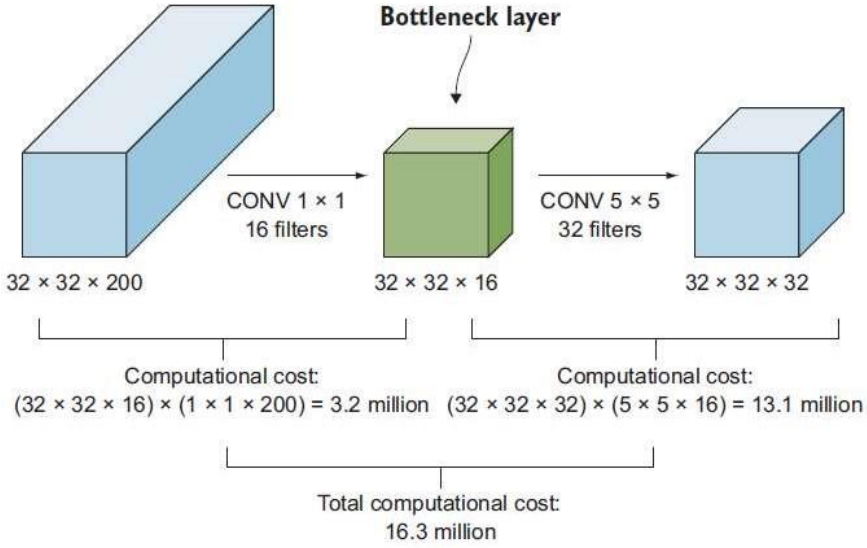




Convolutional Neural Networks

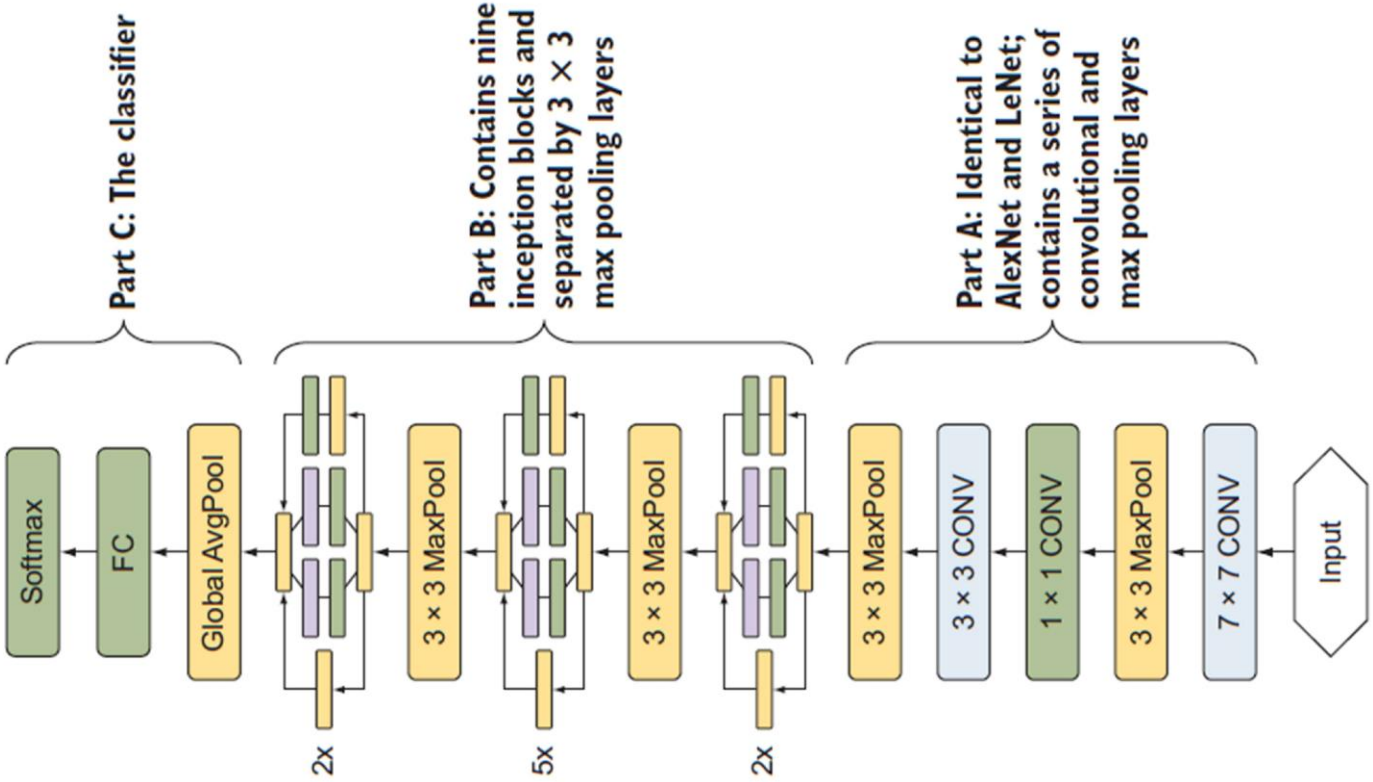
Some Case Studies:GoogleLenet (2014)





Convolutional Neural Networks

Some Case Studies: GoogleLenet (2014)



Convolutional Neural Networks

Some Case Studies:GoogleLenet (2014)

Batch Normalization

Input: Values of x over a mini-batch: $\mathcal{B} = \{x_{1...m}\}$;
Parameters to be learned: γ, β
Output: $\{y_i = \mathrm{BN}_{\gamma,\beta}(x_i)\}$

$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \text{// mini-batch mean}$$

$$\sigma^2_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_{\mathcal{B}})^2 \qquad \text{// mini-batch variance}$$

$$\widehat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma^2_{\mathcal{B}} + \epsilon}} \qquad \text{// normalize}$$

$$y_i \leftarrow \gamma \widehat{x}_i + \beta \equiv \mathrm{BN}_{\gamma,\beta}(x_i) \qquad \text{// scale and shift}$$

Figure from Ioffe et al. (2015)

Want to learn more?

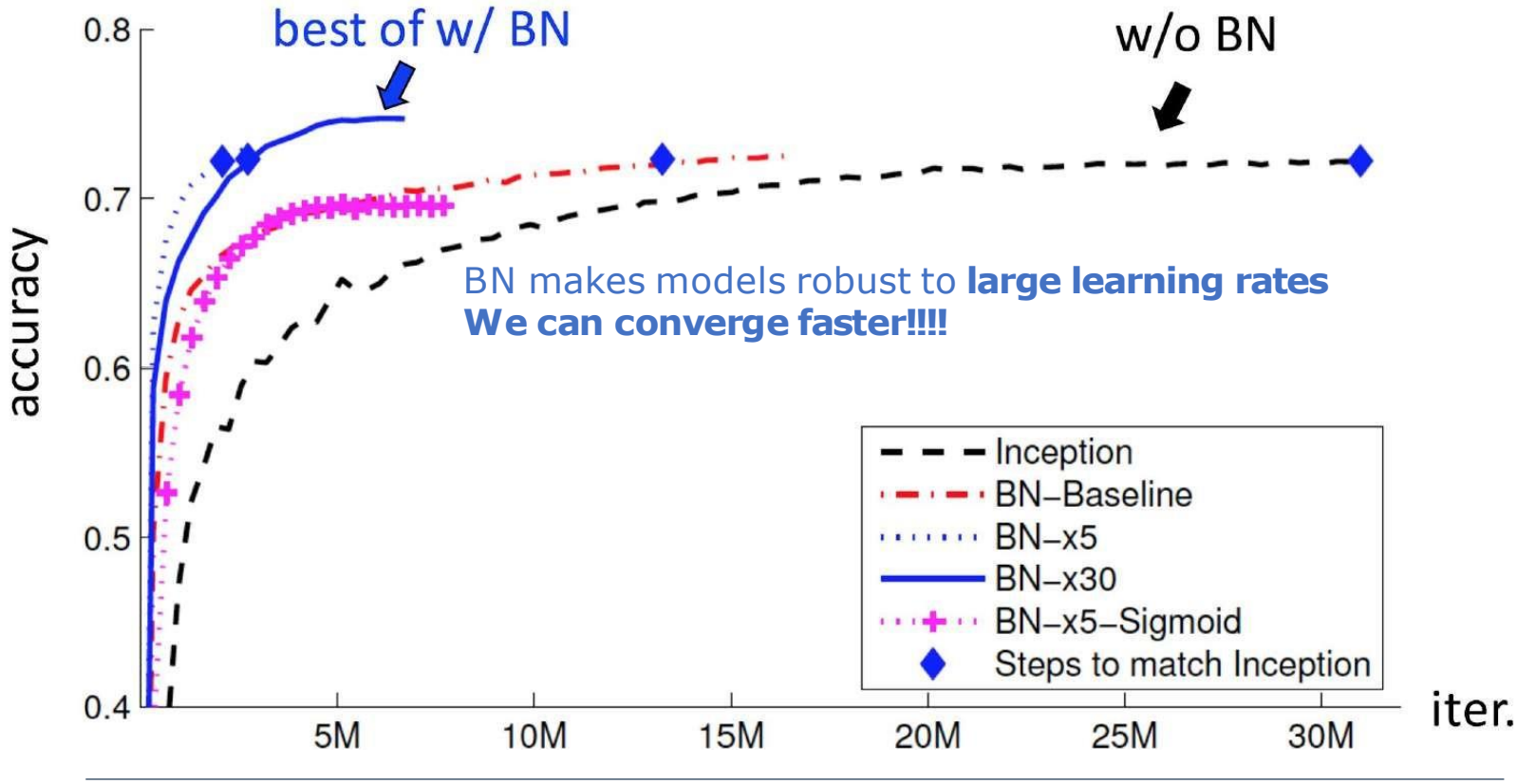
Ioffe, S.; Szegedy, C. ***Batch normalization: Accelerating deep network training by reducing internal covariate shift*** International conference on machine learning (**2015**)

Reduces sensitivity to **initialisation**
Introduces **Noise** and makes robust to **large learning rates**
Introduces stochasticity and acts as a **regulariser**

Convolutional Neural Networks

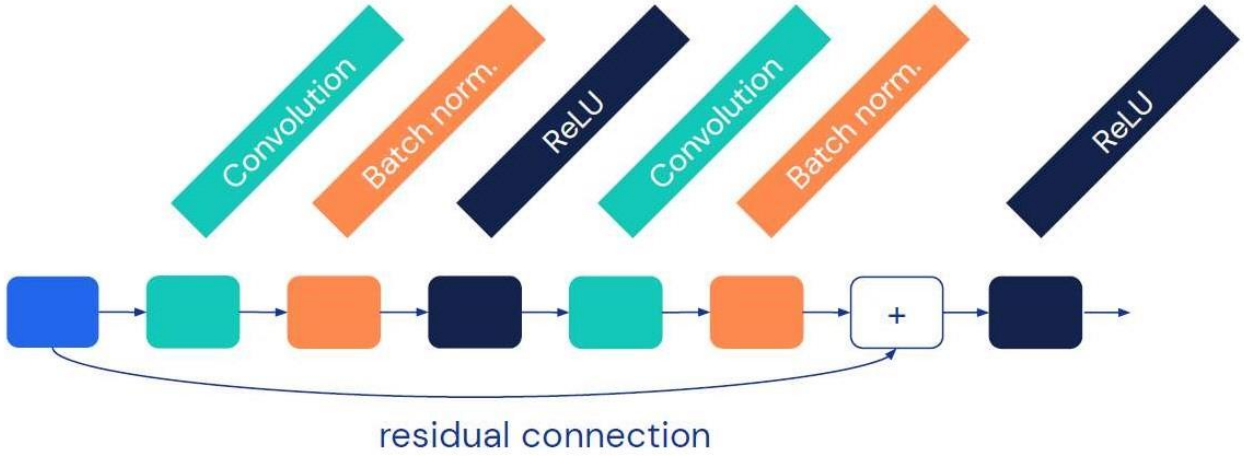
Some Case Studies: GoogleLenet (2014)

Batch Normalization



Convolutional Neural Networks

Some Case Studies: ResNet (2015)



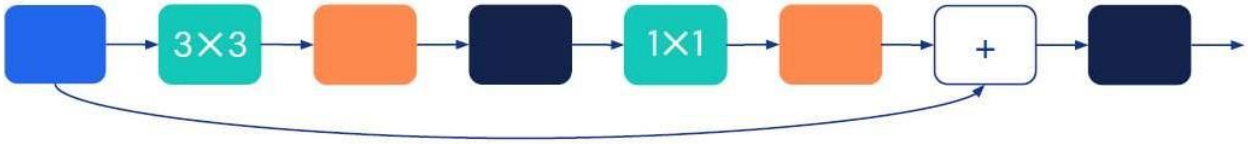
Want to learn more?

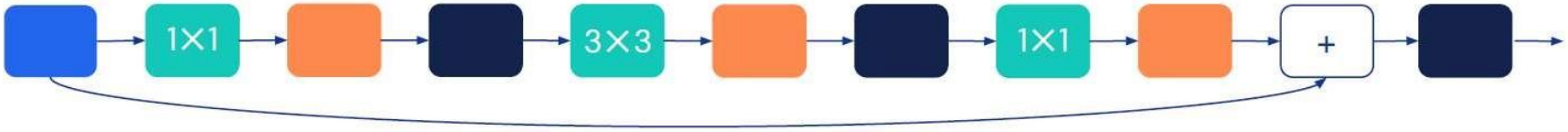
He, K. et al.
Deep residual learning for image recognition
IEEE conference on computer vision and pattern recognition **(2016)**

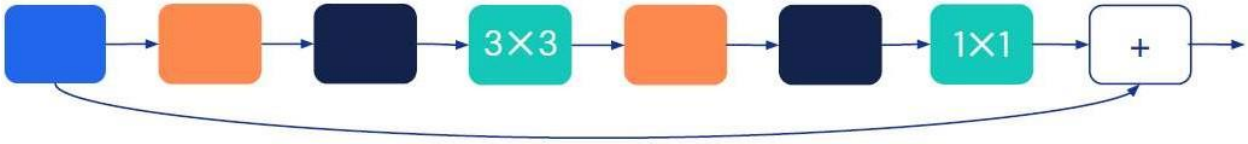
Residual connections facilitate training deeper networks

Convolutional Neural Networks

Some Case Studies: ResNet (2015)







Want to learn more?

He, K. et al.
Identity mappings in deep residual networks
European conference on computer vision
(2016)

ResNet V2 (bottom) avoids all nonlinearities in the residual pathway

Convolutional Neural Networks

Some Case Studies:ResNet (2015)

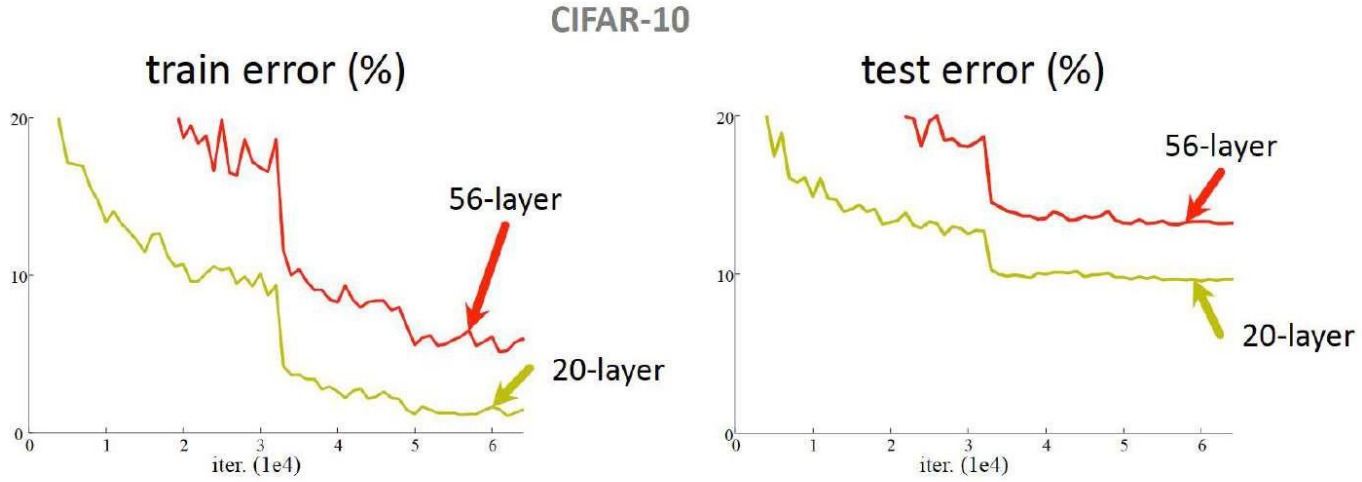
layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$
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

Table from He et al. (2015)

Convolutional Neural Networks

Some Case Studies:ResNet (2015)

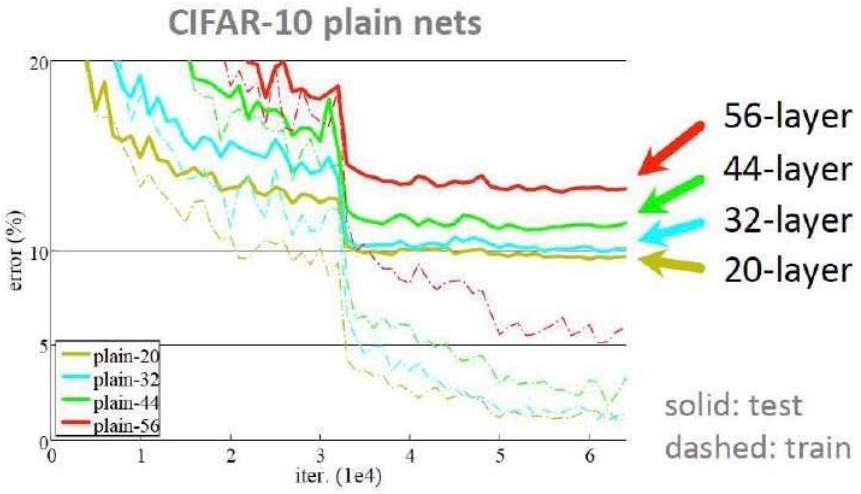
Simply stacking layers?

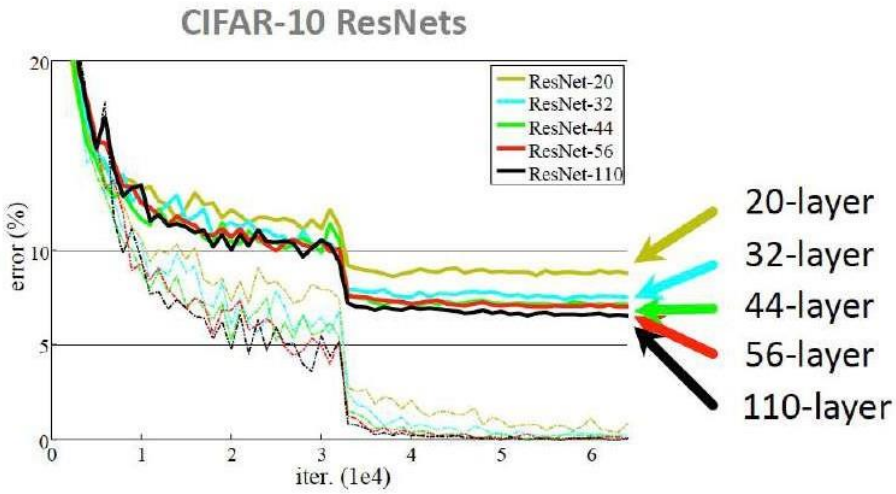


- Plain nets: stacking 3x3 conv layers
- 56-layer net has **higher training error and test error** than 20-layer net
- A deeper model should not have higher training error

Convolutional Neural Networks

Some Case Studies:ResNet (2015)

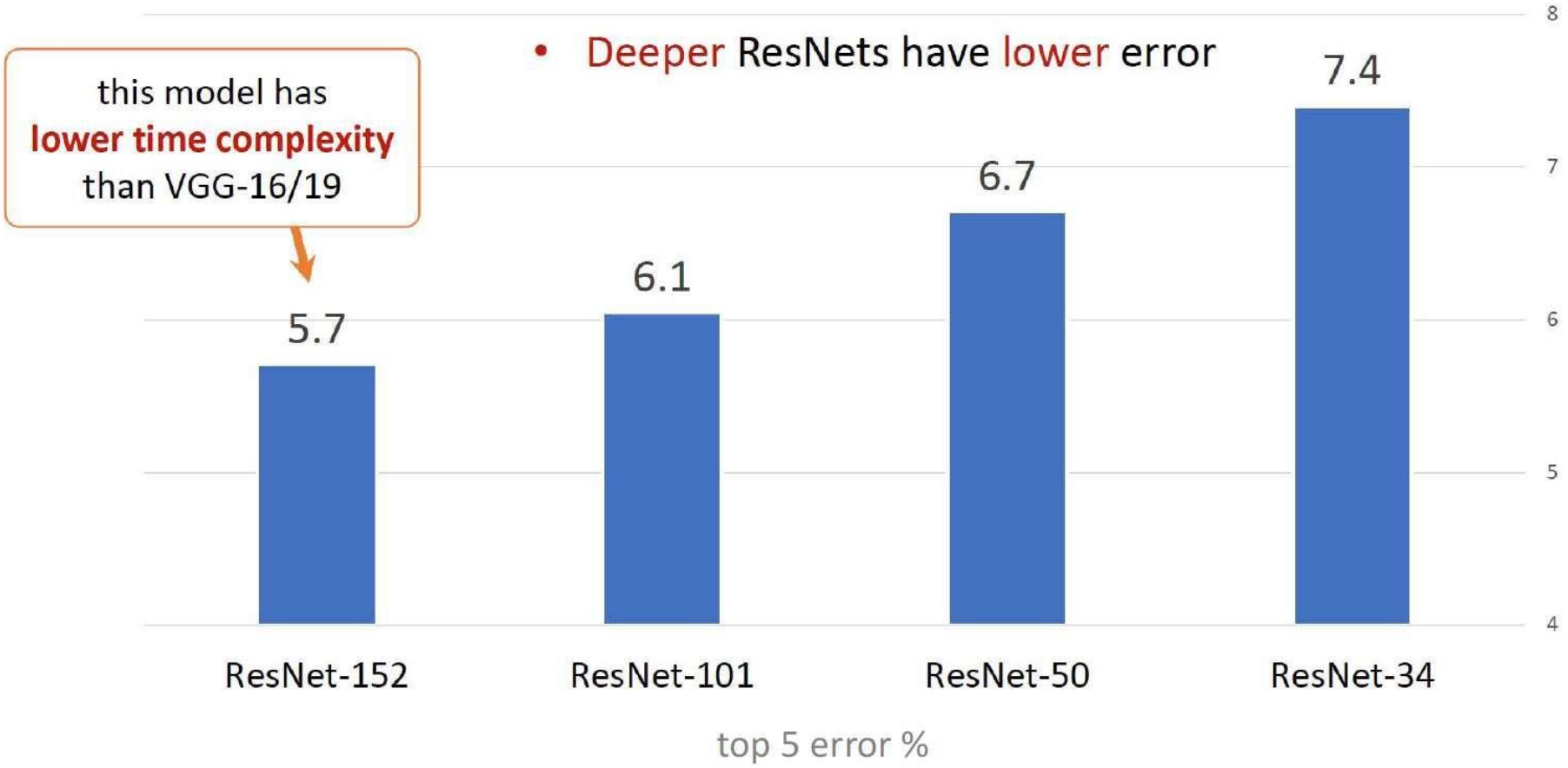




- Deep ResNets can be trained easier
- Deeper ResNets have lower training error, and also lower test error

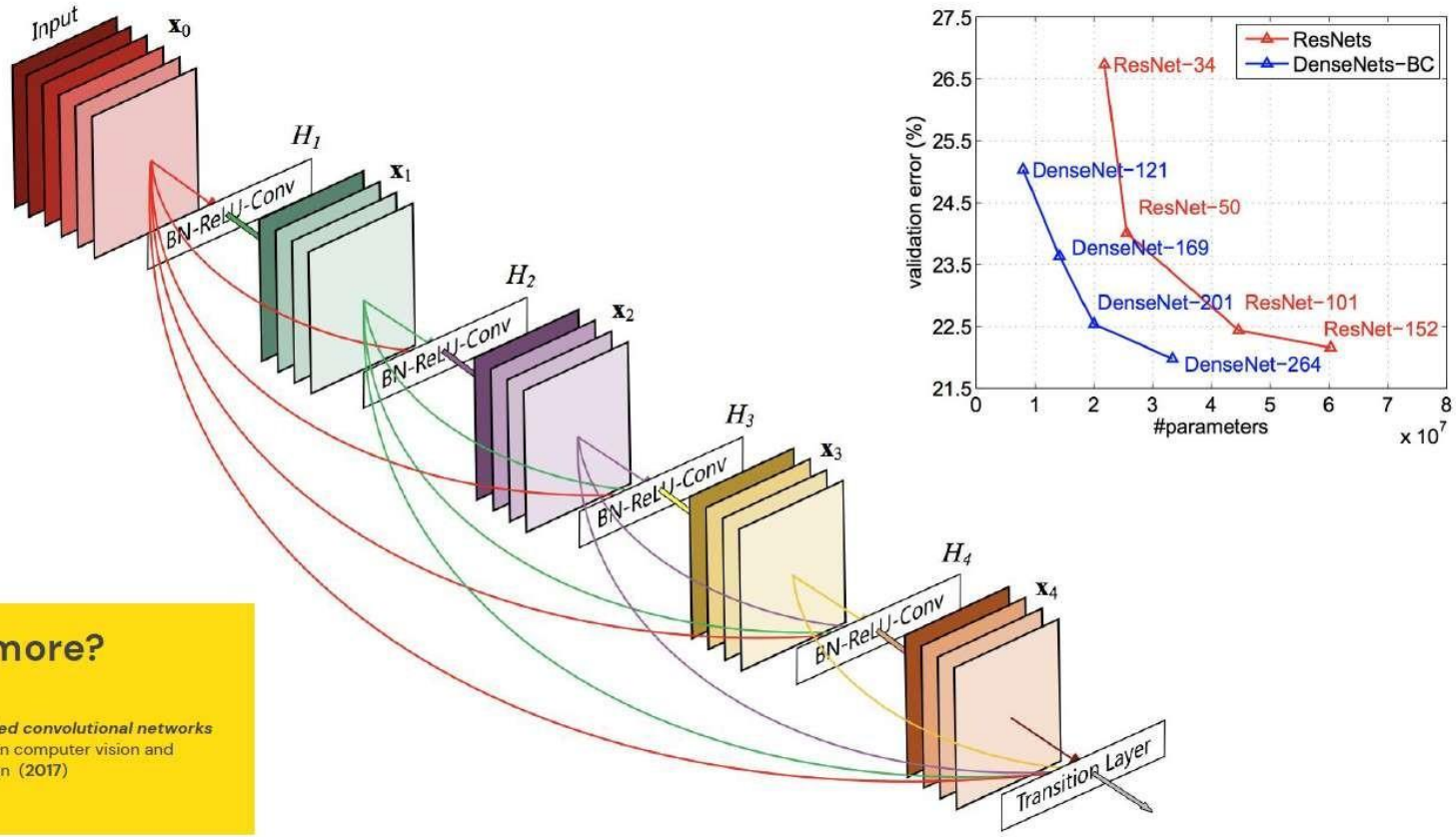
Convolutional Neural Networks

Some Case Studies: ResNet (2015)



Convolutional Neural Networks

Some Case Studies: DenseNet (2016)

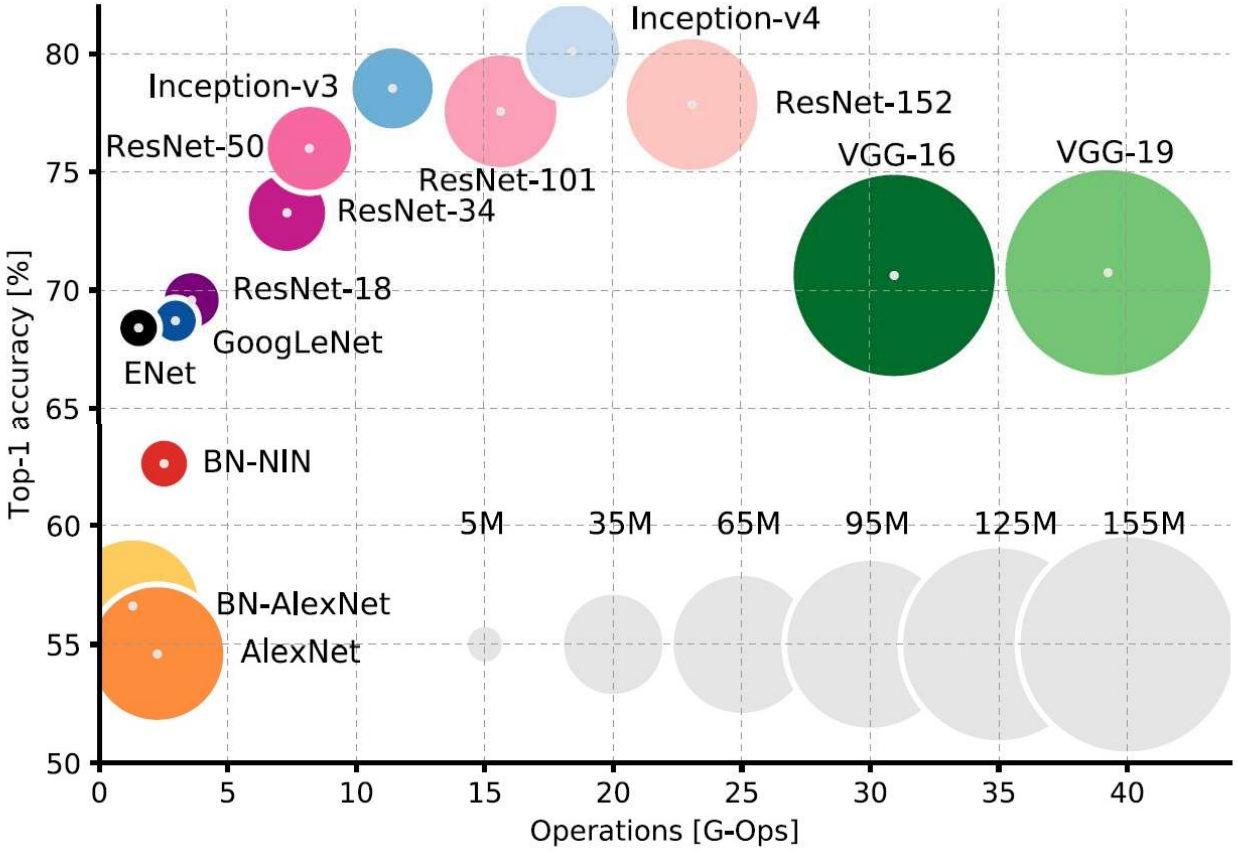


Want to learn more?

Huang, G. et al.
Densely connected convolutional networks
IEEE conference on computer vision and pattern recognition **(2017)**

Figures from Huang et al. (2015)

Convolutional Neural Networks
Summary

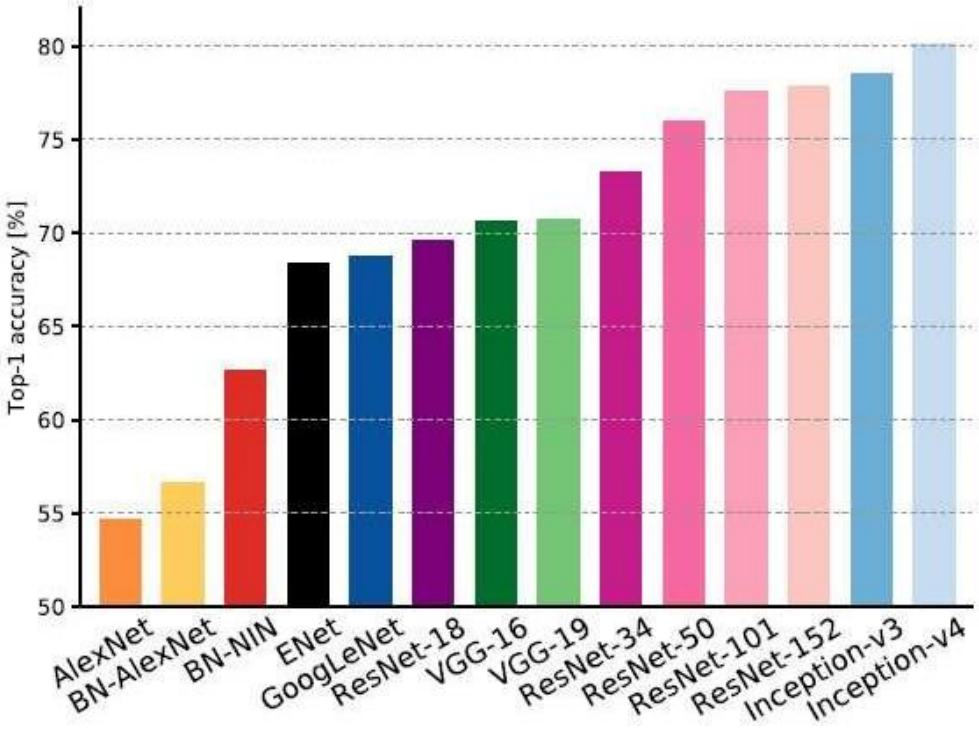


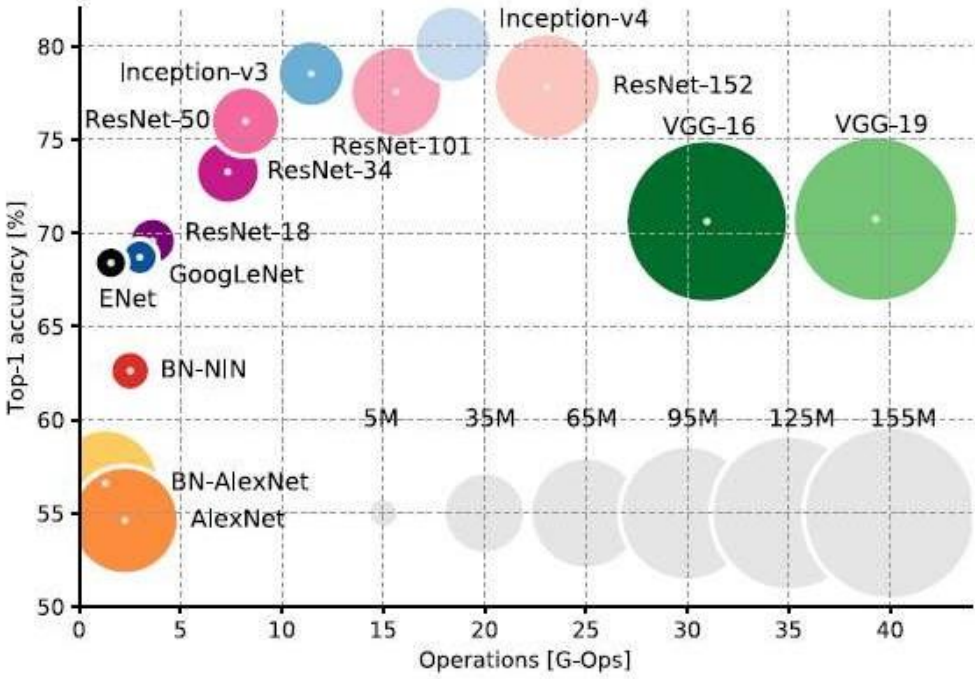
Canziani, A., Paszke, A., & Culurciello, E. (2016).

An analysis of deep neural network models for practical applications.

arXiv preprint arXiv:1605.07678.

Convolutional Neural Networks
Summary





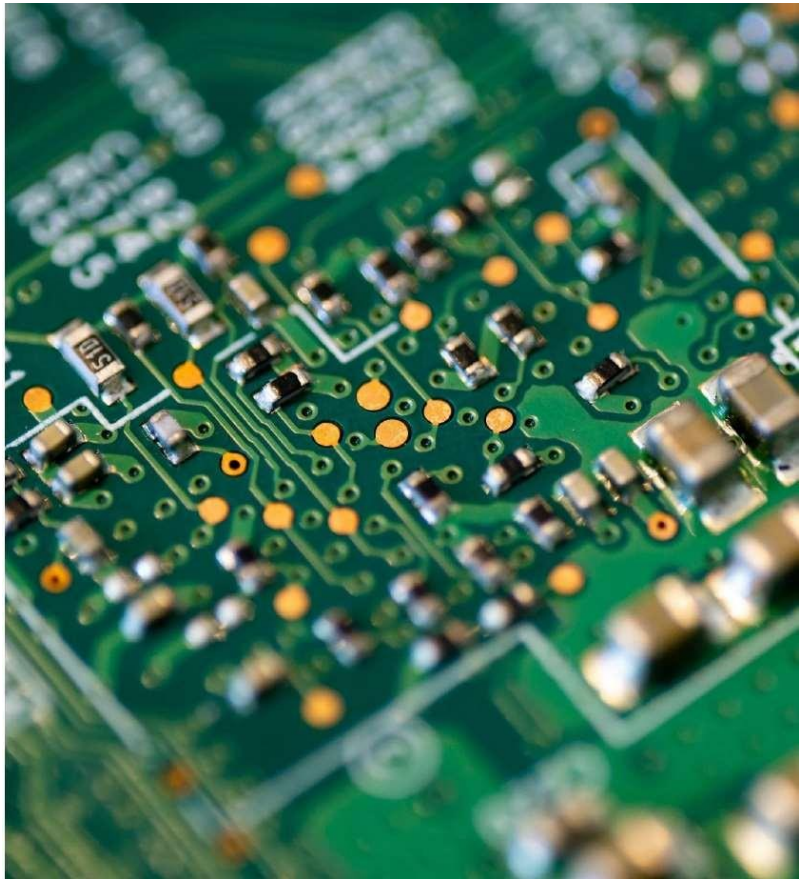
Convolutional Neural Networks
Reducing Complexity

Reducing Complexity

Depthwise Convolutions Separable Convolutions Inverted BottleNecks (MobileNet, MNASNet, EfficientNet)

Improving resource usage

Quantized Neural Networks
Binarized Neural Networks



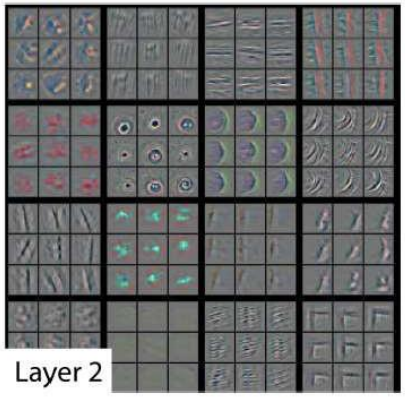
Convolutional Neural Networks
Data Augmentation

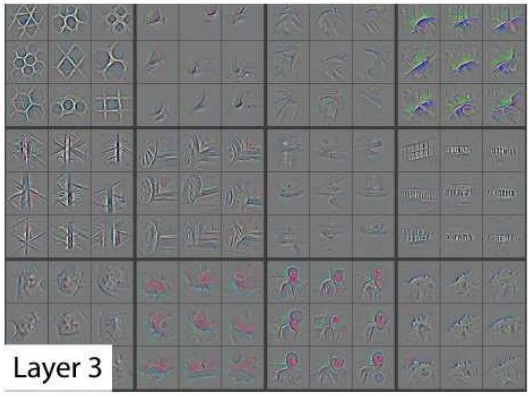
By design, convnets are only robust against **translation**

Data augmentation makes them robust against other transformations: rotation, scaling, shearing, warping, ...

Convolutional Neural Networks

Visualizing what a convnet learns





Figures from Zeiler et al. (2014)

Want to learn more?

Zeiler, M.D.; Fergus, R.
Visualizing and understanding convolutional networks *European conference on computer vision* **(2014)**

Convolutional Neural Networks

Visualizing what a convnet learns

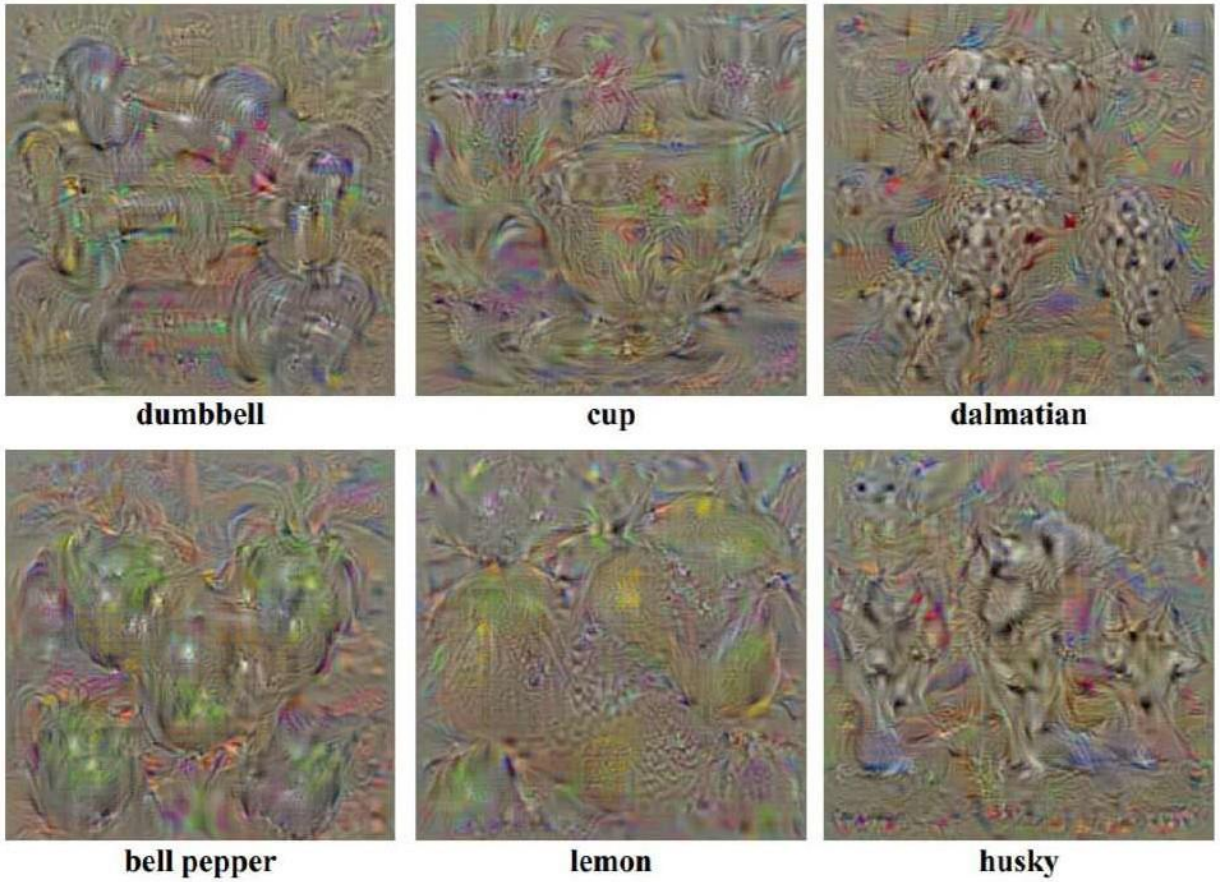


Figure from Simonyan et al. (2013)

Convolutional Neural Networks

Visualizing what a convnet learns

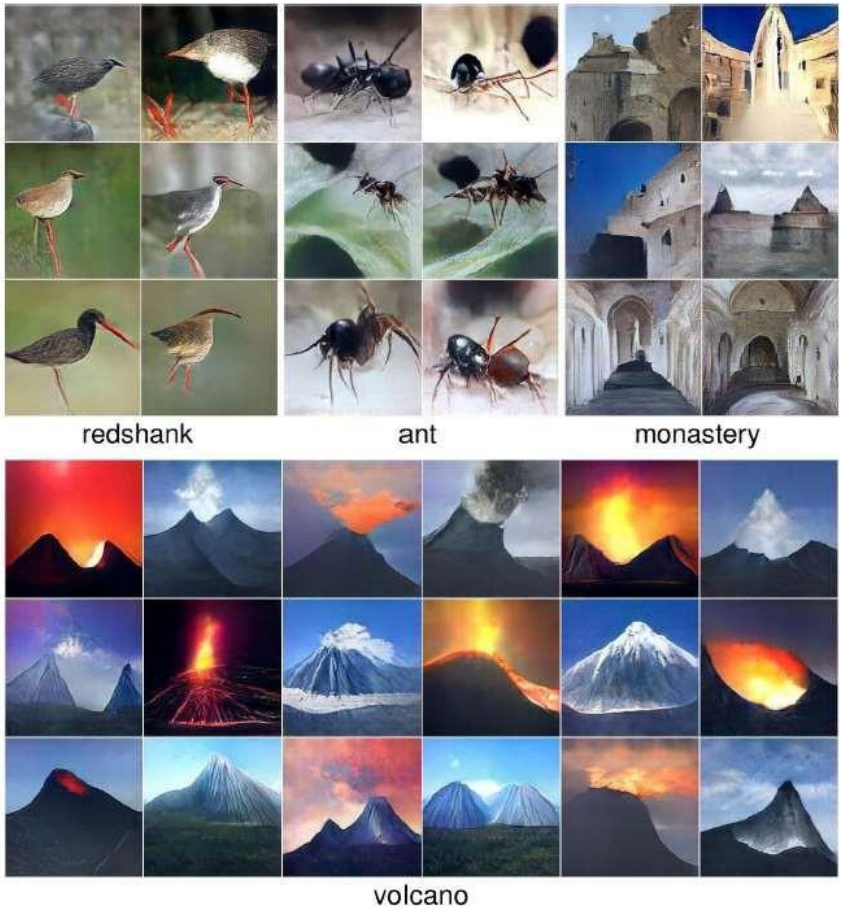


Figure from Nguyen et al. (2016)

Convolutional Neural Networks

Visualizing what a convnet learns

https://distill.pub/2017/feature-visualization/ by **Chris Olah**, **Alexander Mordvintsev** and **Ludwig Schubert**

Feature Visualization

How neural networks build up their understanding of images

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

Feature visualization allows us to see how GoogLeNet [1], trained on the ImageNet [2] dataset, builds up its understanding of images over many layers. Visualizations of all channels are available in the appendix.

AUTHORS
Chris Olah
Alexander Mordvintsev
Ludwig Schubert

AFFILIATIONS
Google Brain Team
Google Research
Google Brain Team

PUBLISHED
Nov. 7, 2017

DOI
10.23915/distill.00007