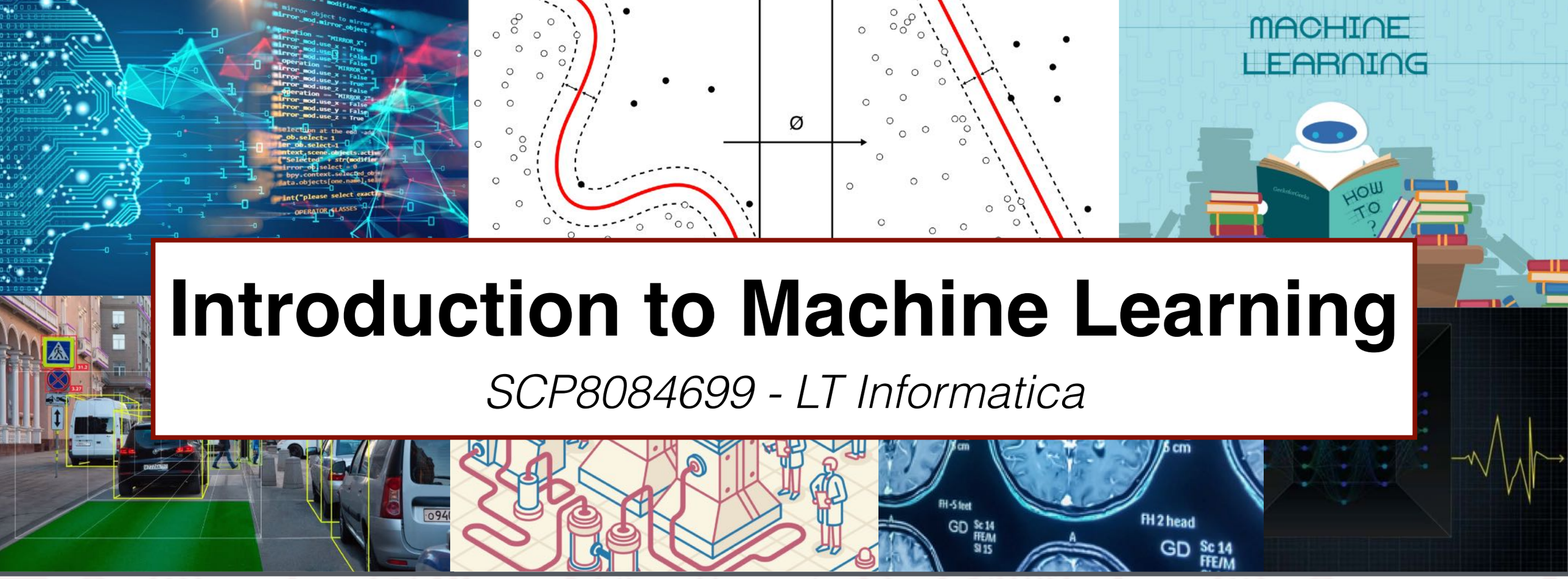


Introduction to Machine Learning

SCP8084699 - LT Informatica

Teaching machines to see: a quest to visual intelligence

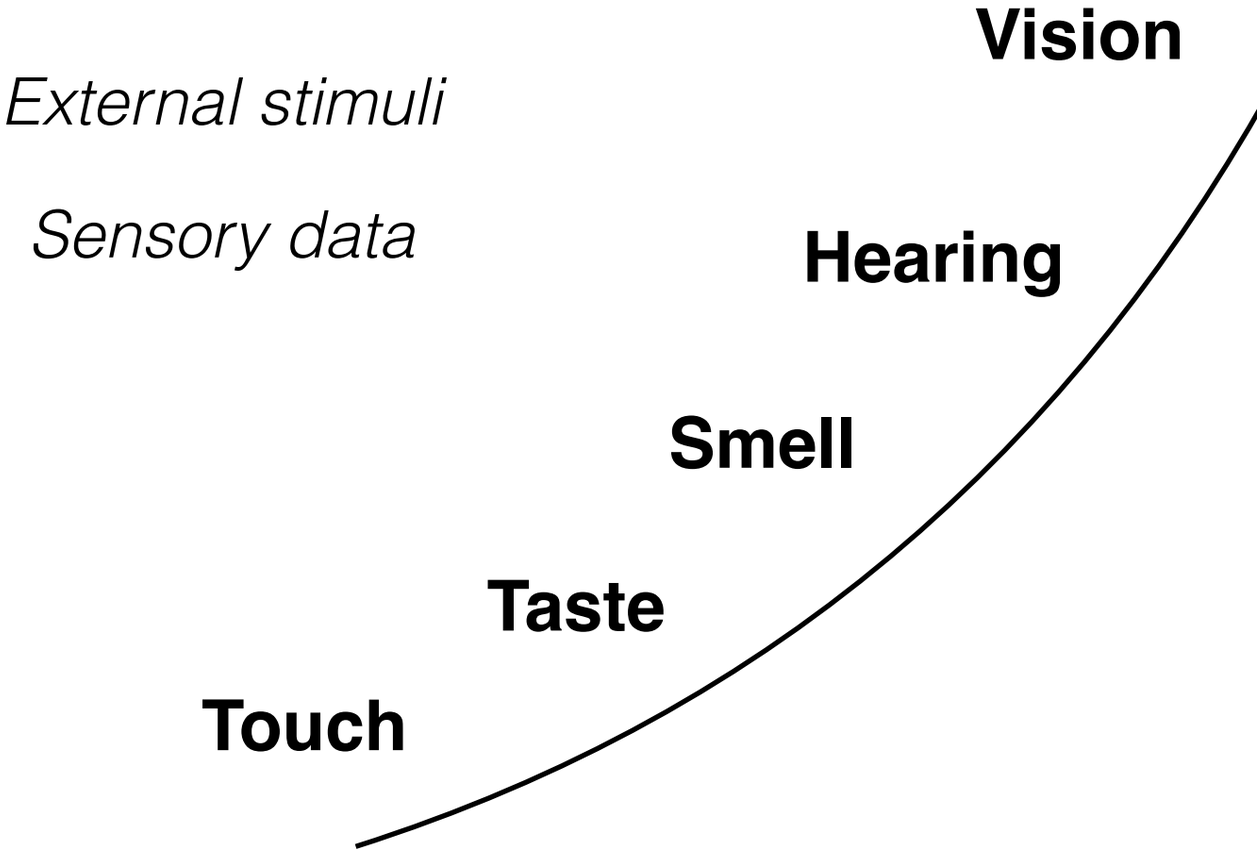
Prof. Lamberto Ballan

University of Padova, A.Y. 2021/22 **Jun 9, 2022**

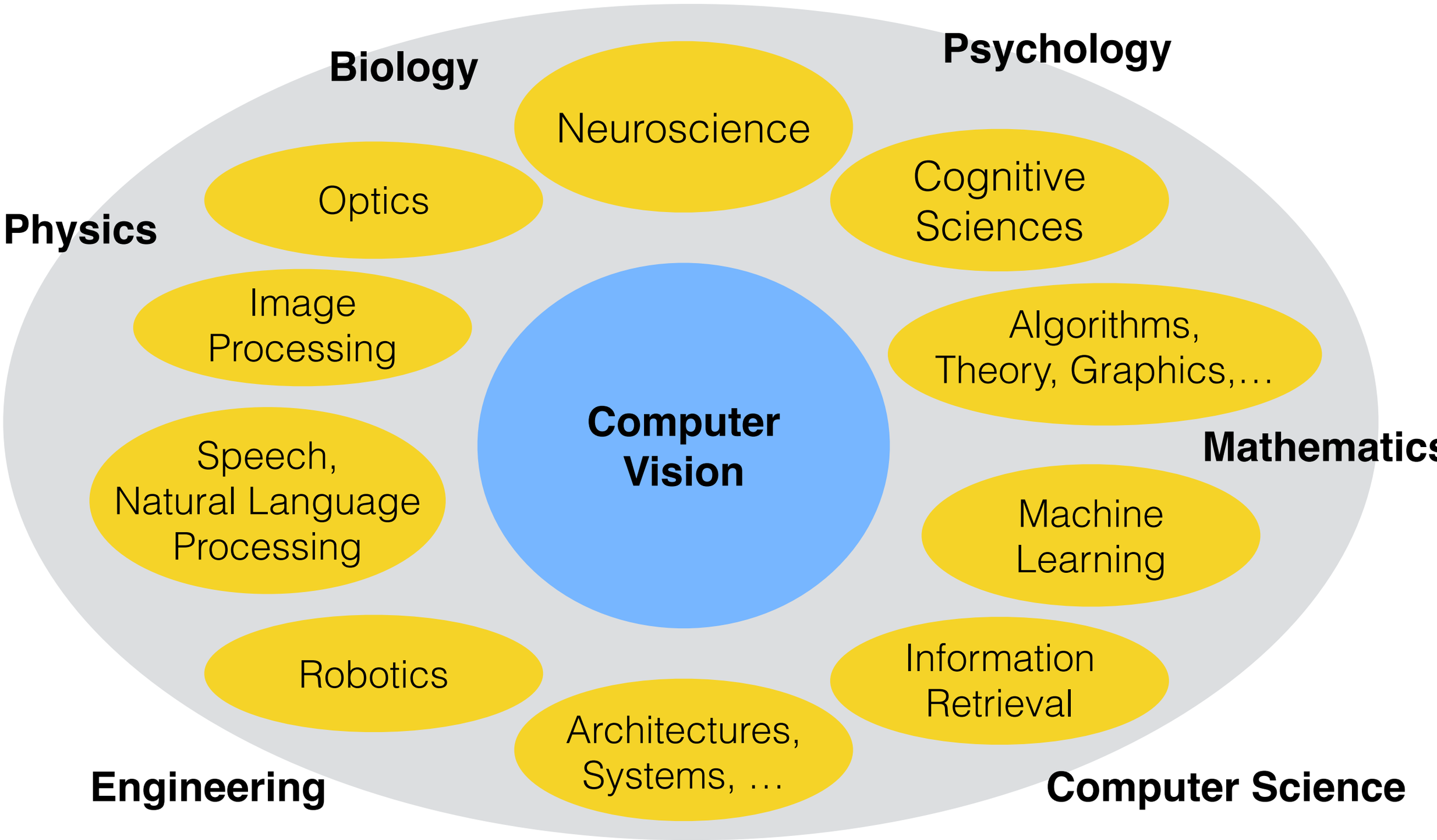
DIPARTIMENTO MATEMATICA
Department of Mathematics “Tullio Levi-Civita”

(Machine) Perception

- Perception is is the ability to capture, process, and make sense of the information that our *senses* receive



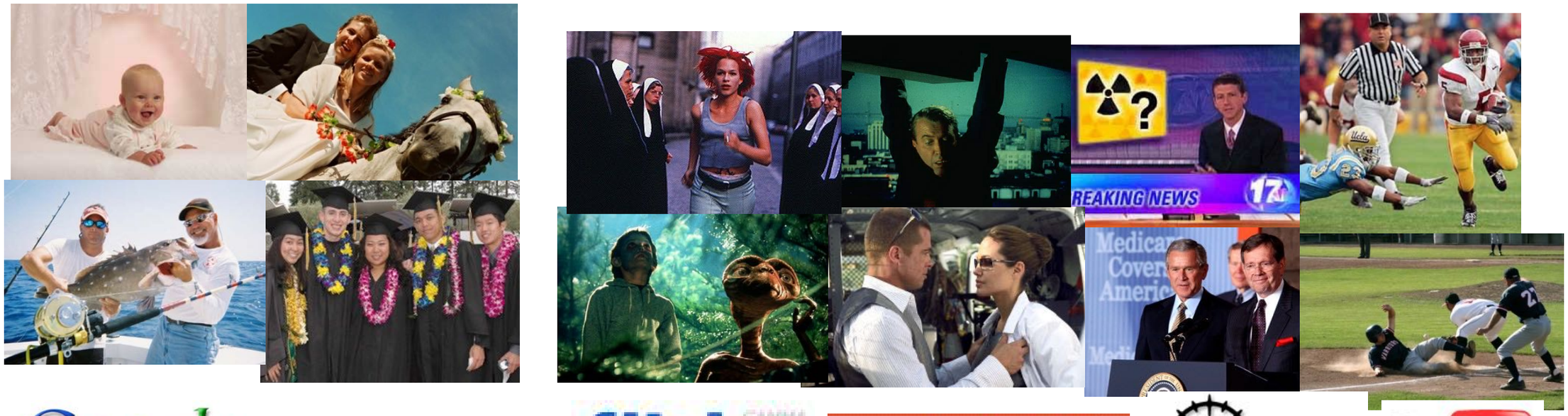
What is computer vision related to?



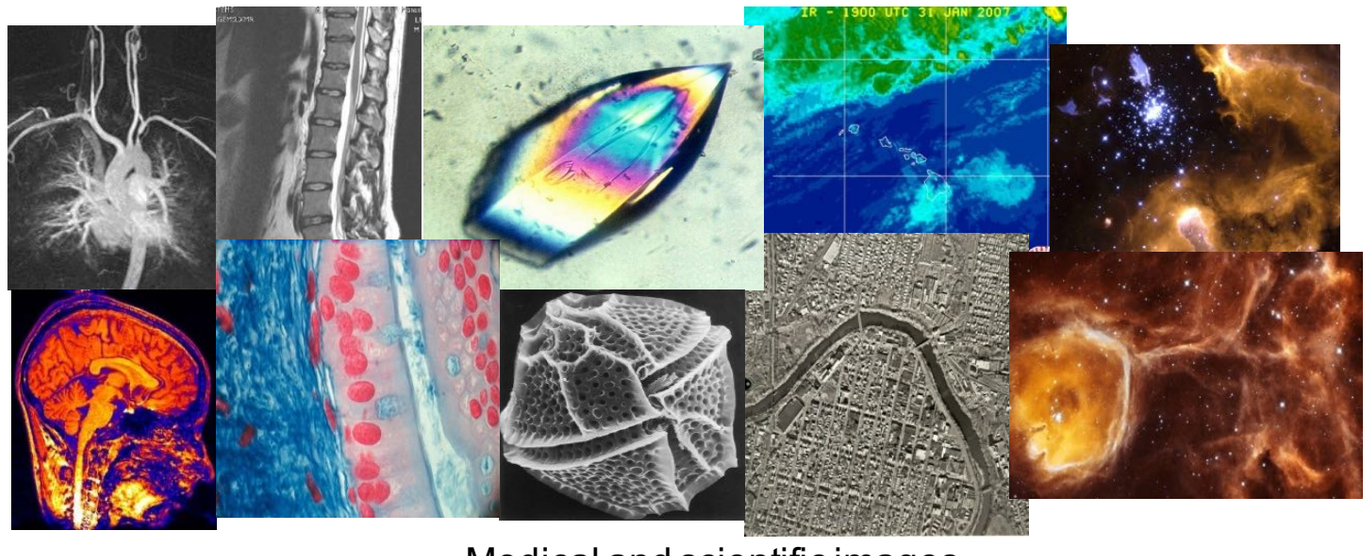
Slide credit: L. Fei-Fei

Why study computer vision?

- Big visual data: almost 90% of web data is visual



Surveillance and security



Medical and scientific images

Slide credit: L. Fei-Fei

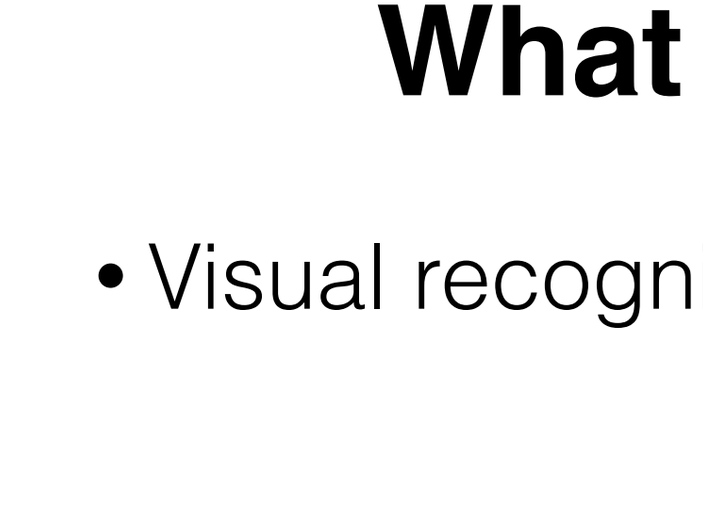
What is (computer) vision?

- Visual recognition is the holy grail of computer vision

Sensing device

Interpreting device

Image (or video)

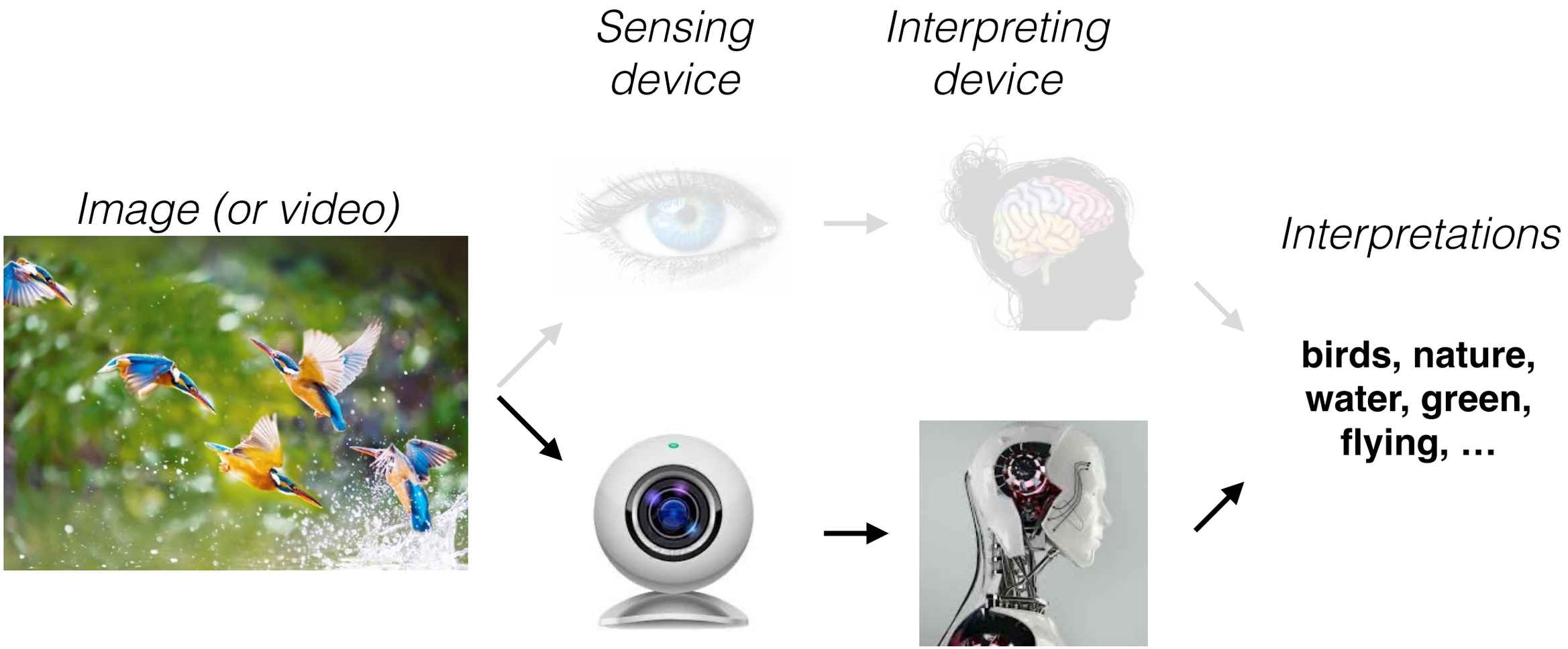


Interpretations

birds, nature, water, green, flying, …

What is computer vision?

- Visual recognition is the holy grail of computer vision



Visual perception: seeing is challenging

- Challenge: illumination

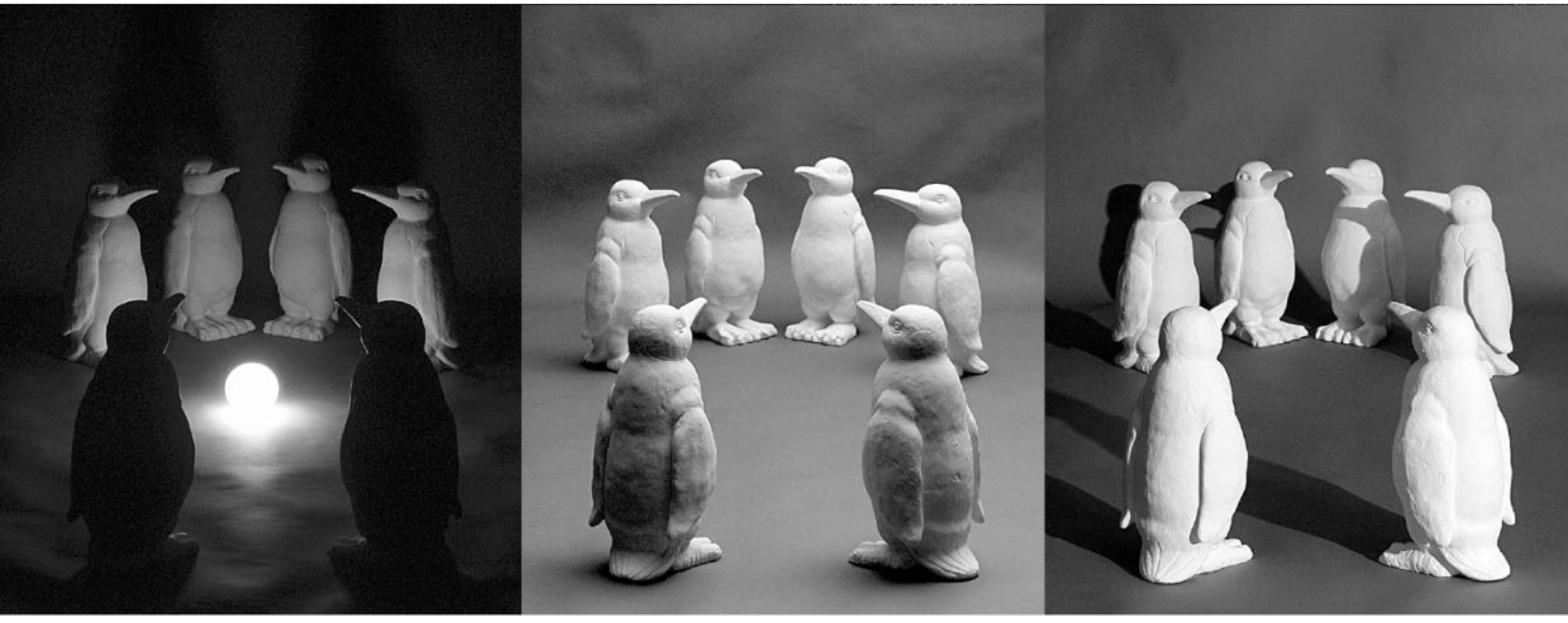


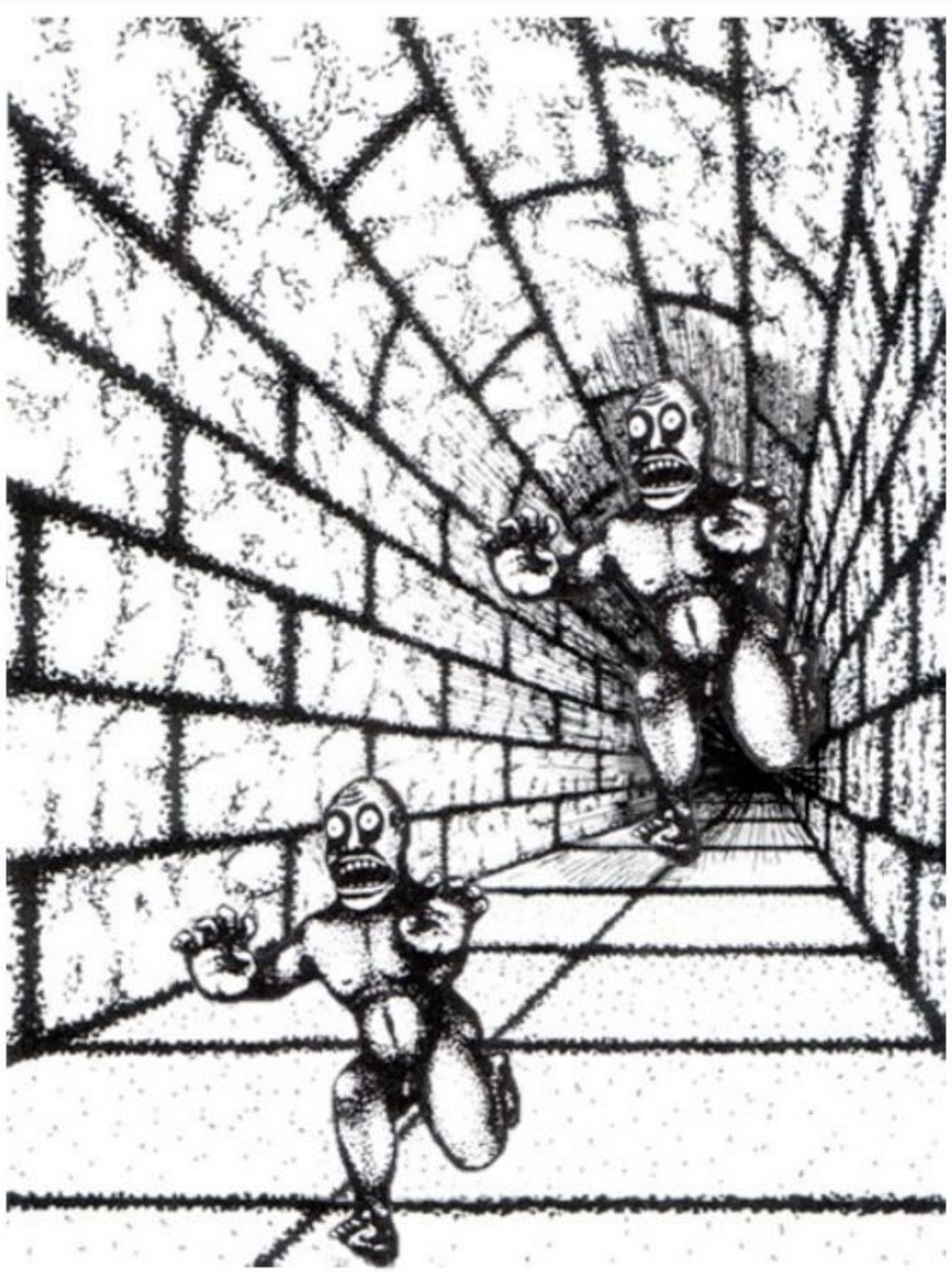
Image credit: J. Koenderink

Seeing is challenging

- Challenge: scale

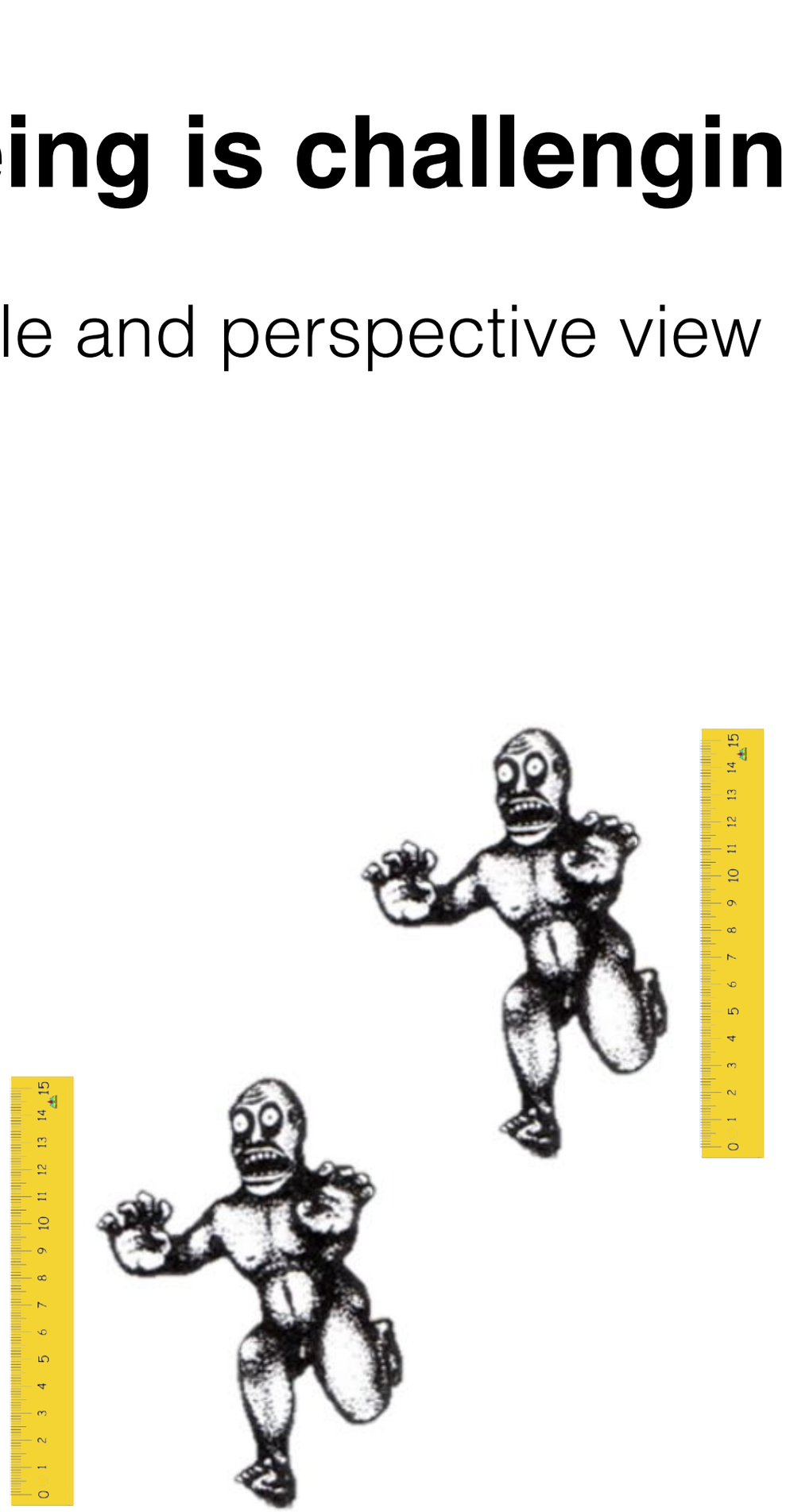
Seeing is challenging

- Challenge: scale and perspective view



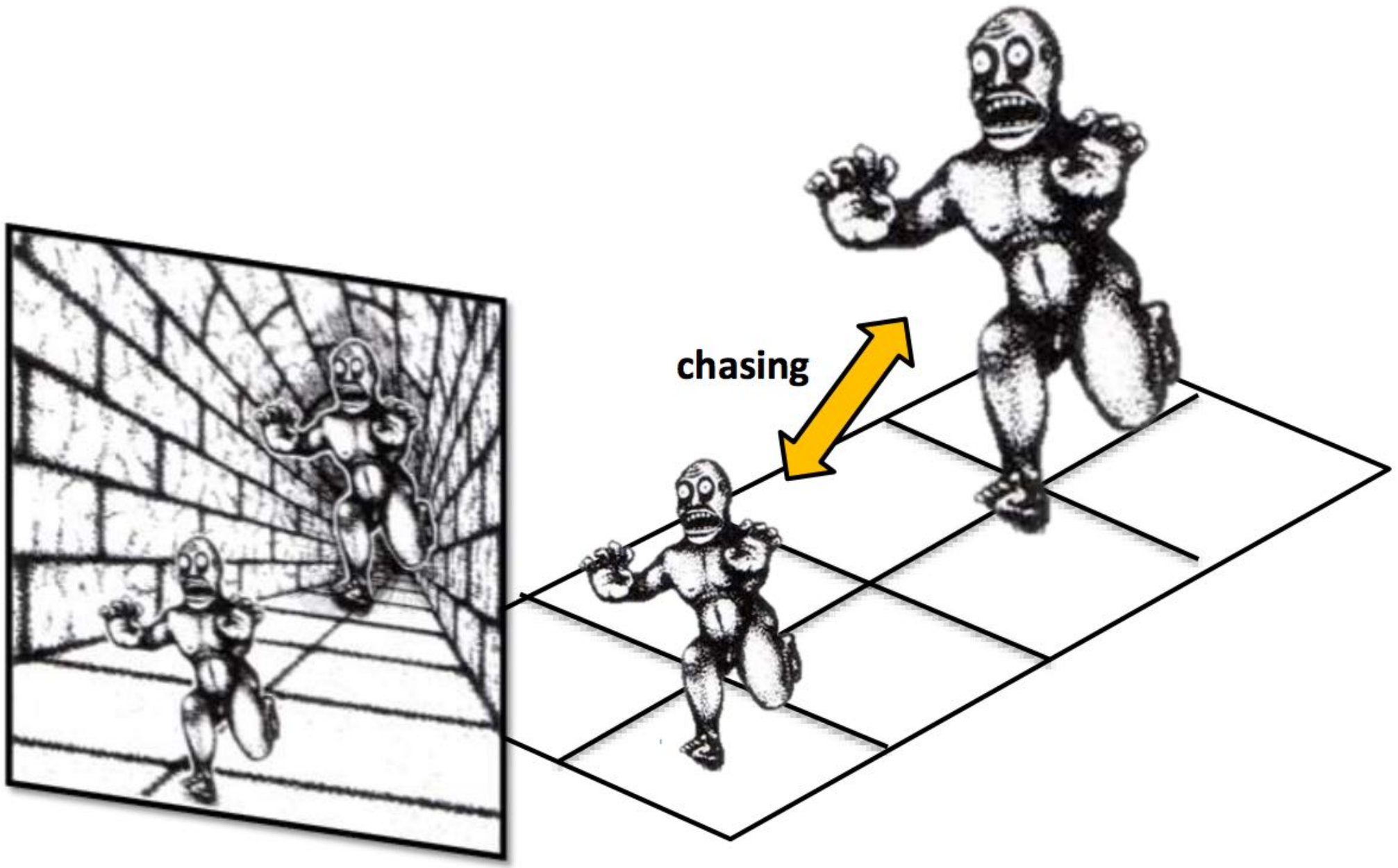
Seeing is challenging

- Challenge: scale and perspective view



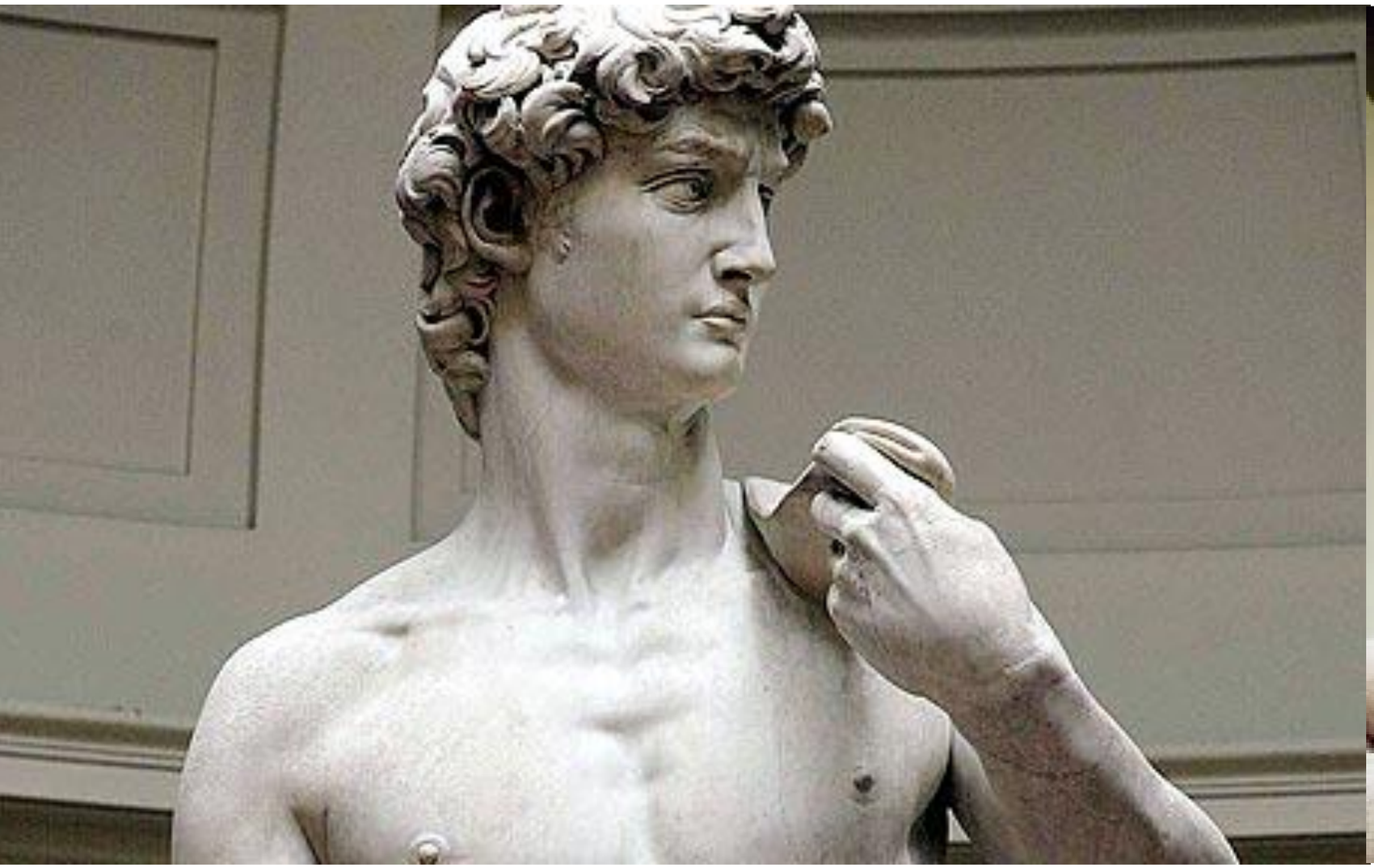
Seeing is challenging

- Challenge: scale and perspective view



Seeing is challenging

- Challenge: viewpoint



Michelangelo's David

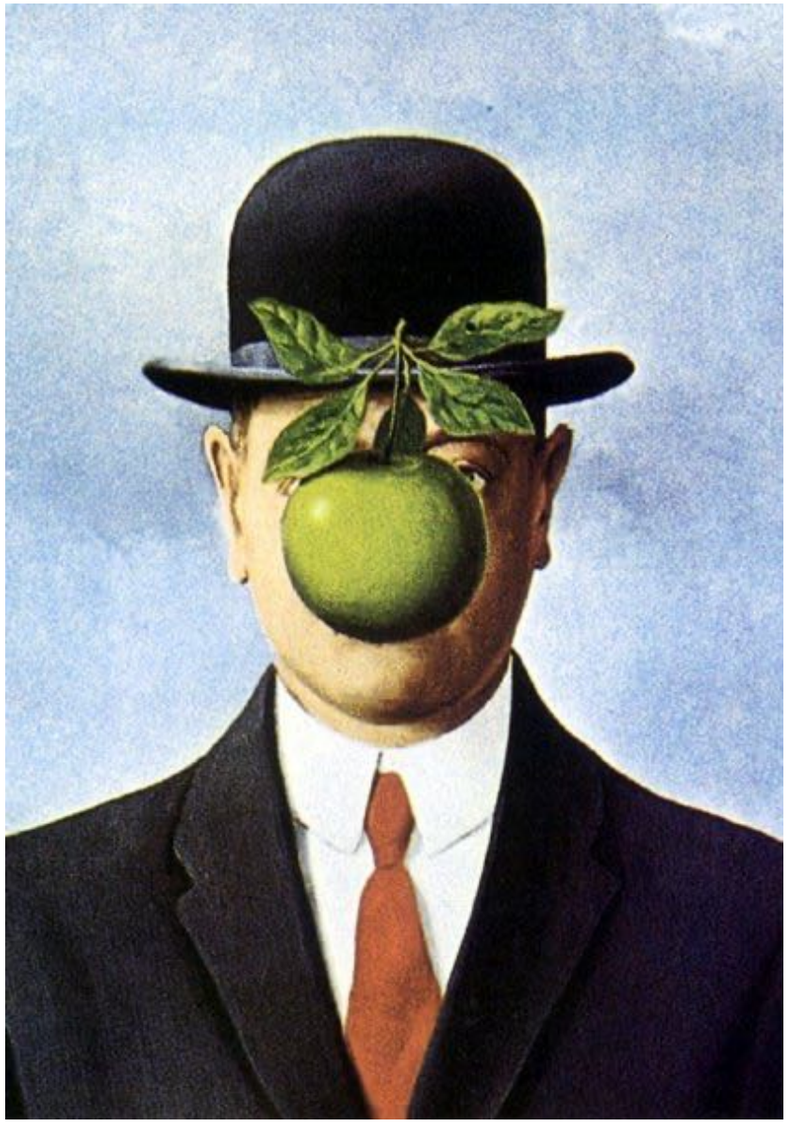
Seeing is challenging

- Challenge: deformation



Seeing is challenging

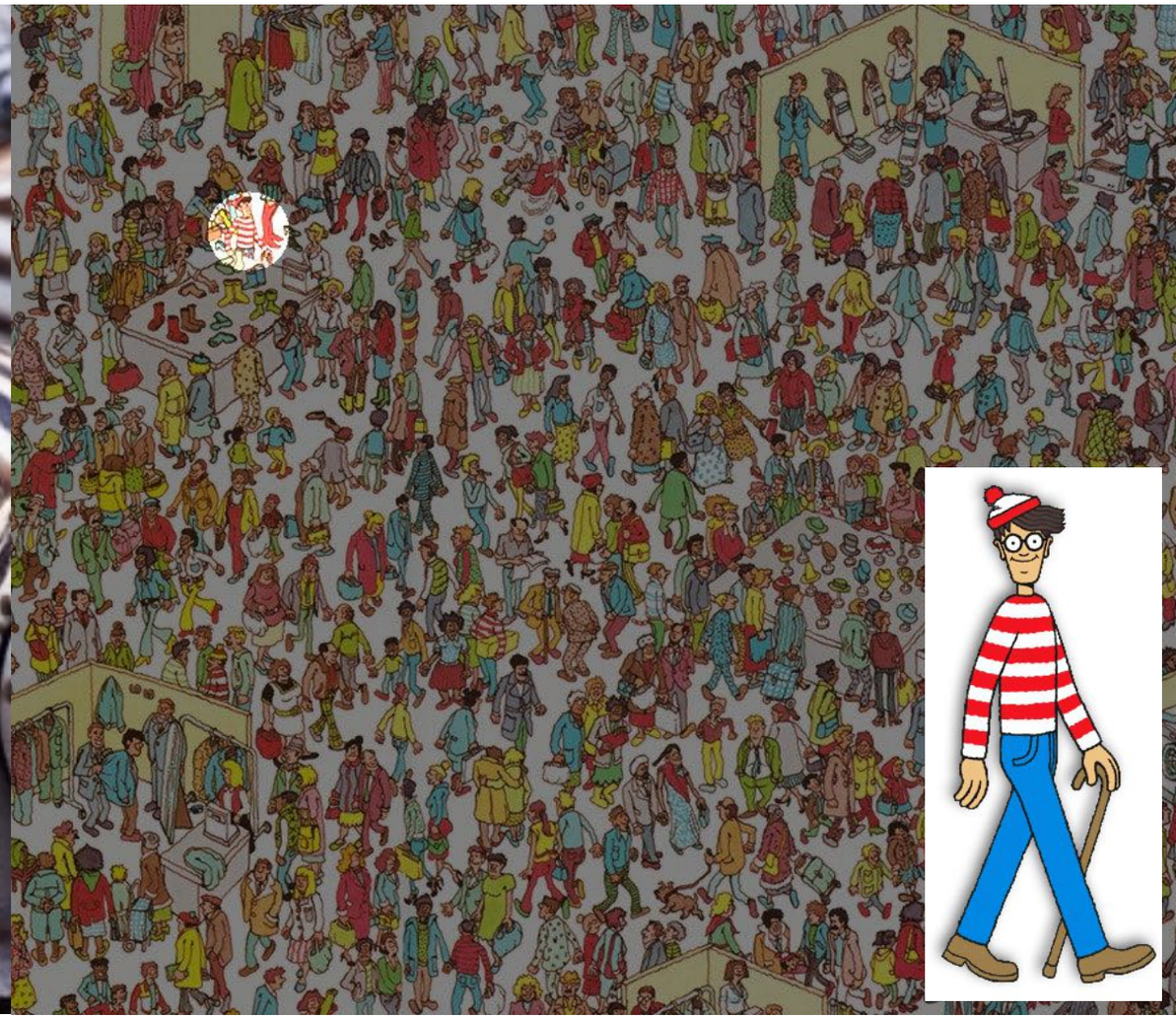
- Challenge: occlusion



Magritte's "The Son of Man"

Seeing is challenging

- Challenge: clutter



What is (computer) vision?

- Visual recognition is the holy grail of computer vision

Image (or video)

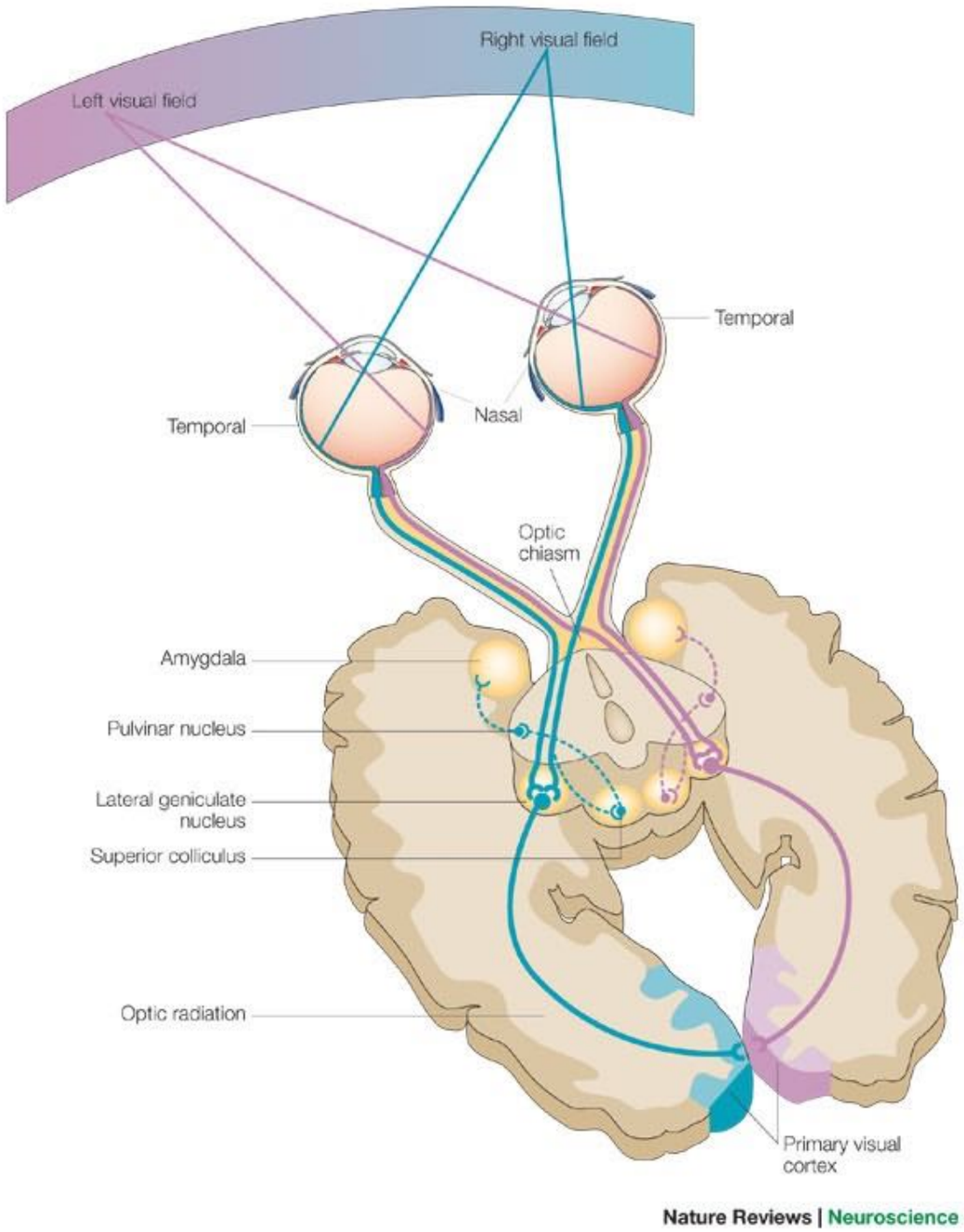
Sensing device

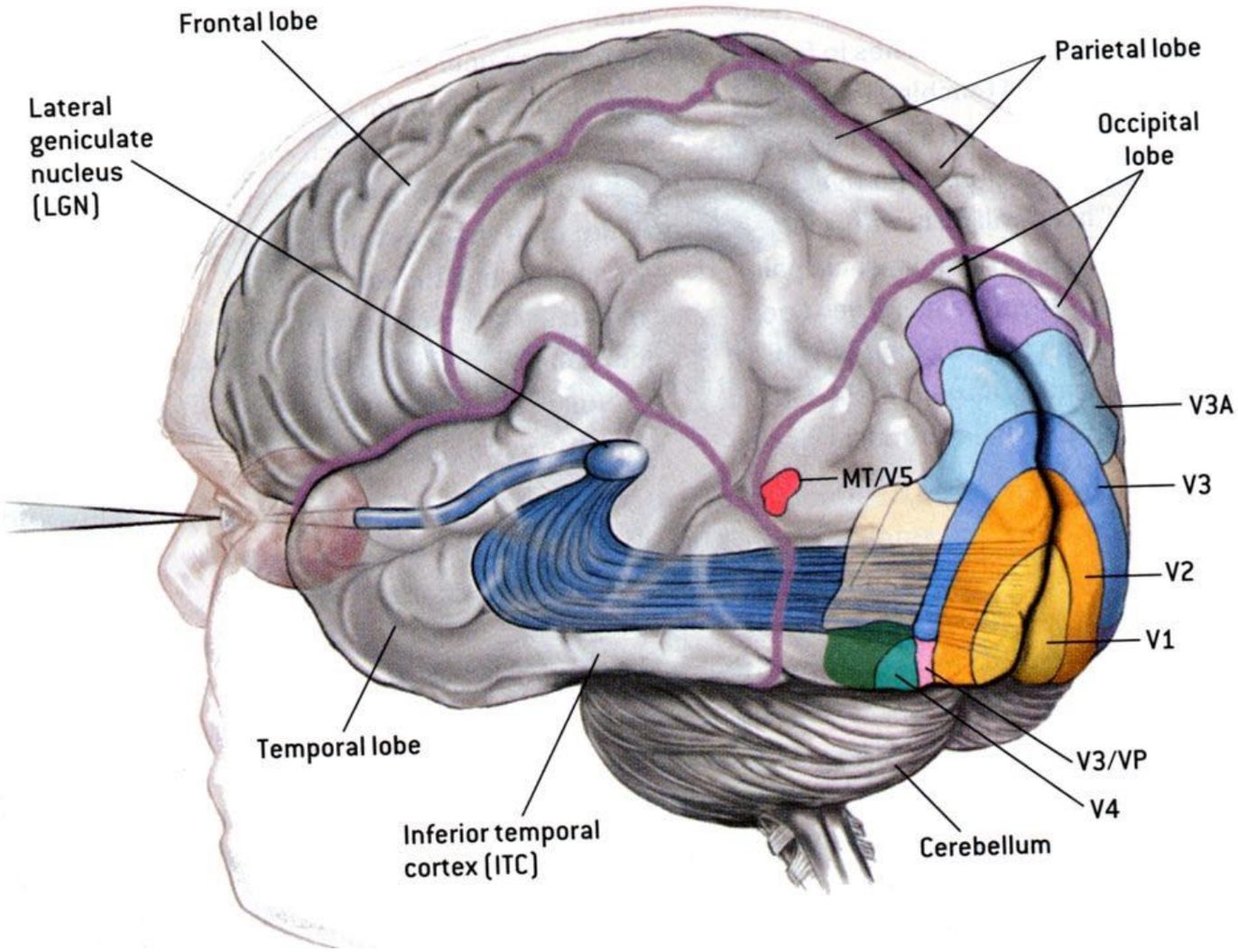
Interpreting device

Interpretations

birds, nature, water, green, flying, …

Human (mammalian) vision

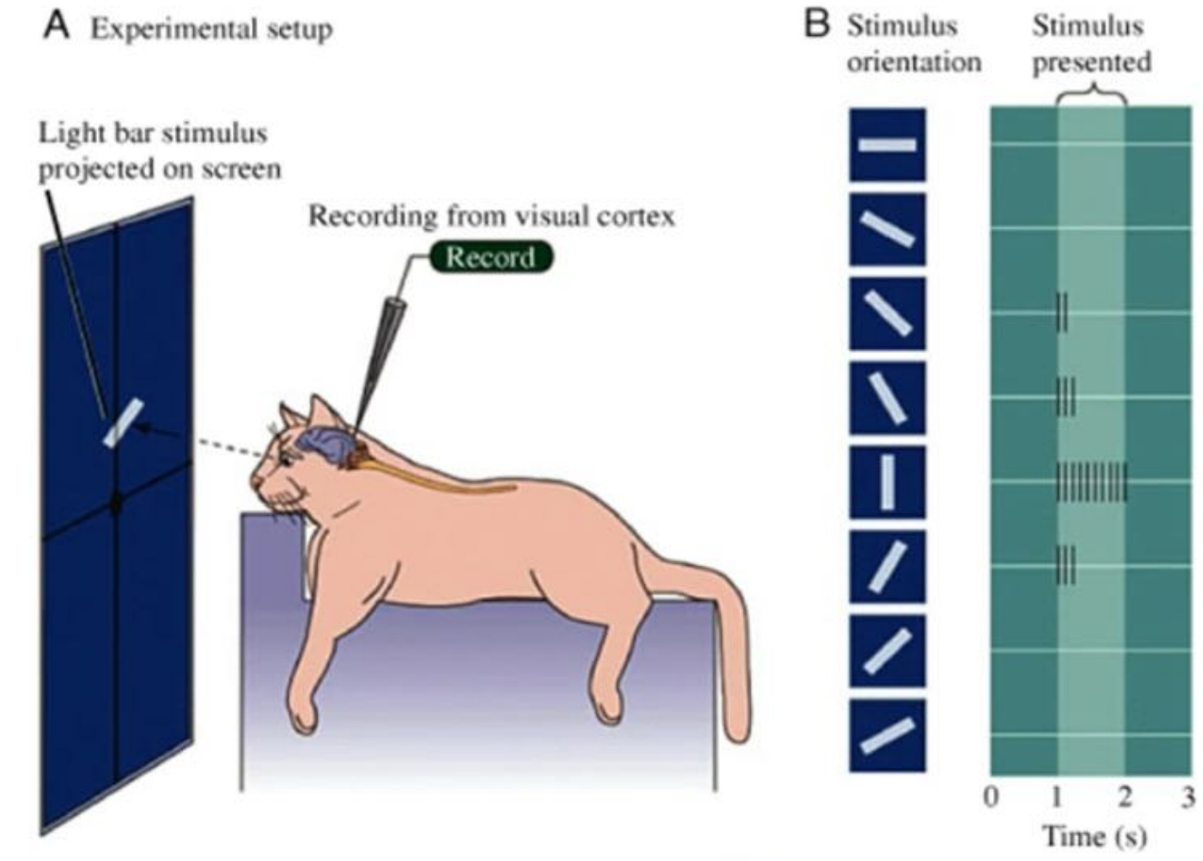


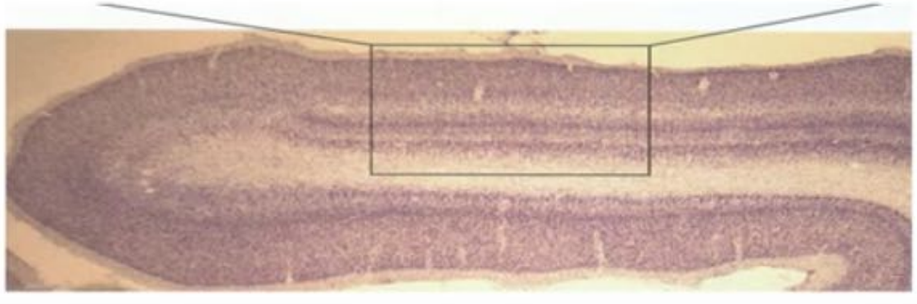


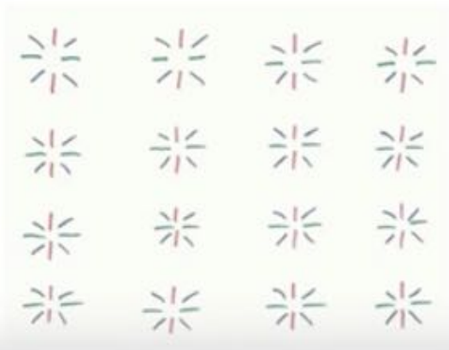
Functional specialization:
V1: primary visual cortex
V4: color, V3/VP: recognition
MT/V5: motion

Human (mammalian) vision

- 1981: *Hubel* & *Wiesel* won the Nobel in medicine







Hubel and Wiesel, *Receptive fields of single neurones in the cat's striate cortex*, J. Physiol 1959

https://www.youtube.com/watch?v=8VdFf3egwfg&t=18s

The goal of computer vision

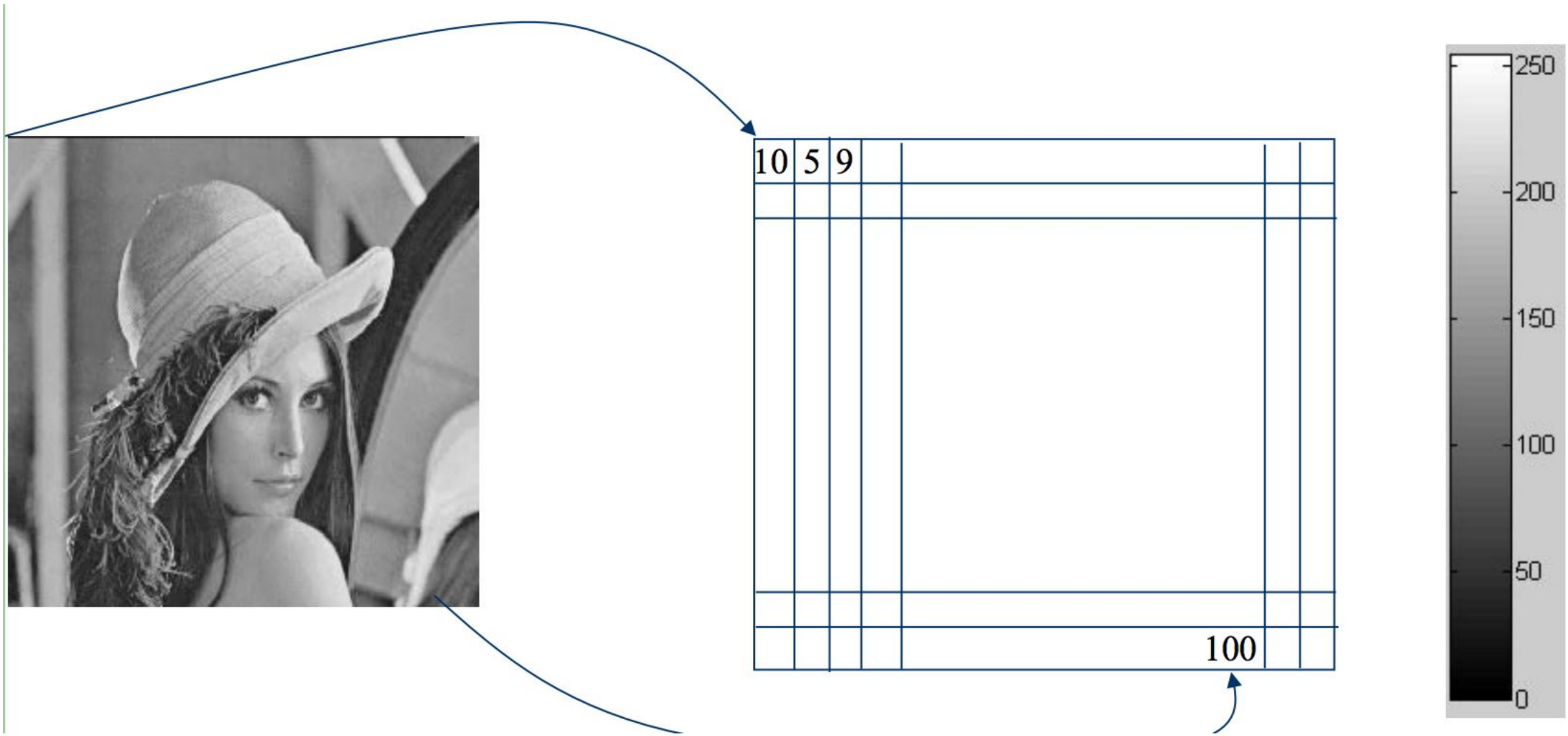
- To bridge the gap between pixels and meaning

What we see

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

What a computer sees

Grayscale image representation



Color image representation

B [0,…,255]

G [0,…,255]

R [0,…,255]

Images are sampled and quantized

- An images contains discrete number of pixels
- Let's see a simple example
  - Pixel value: color (thee values in RGB, HSV, Lab)

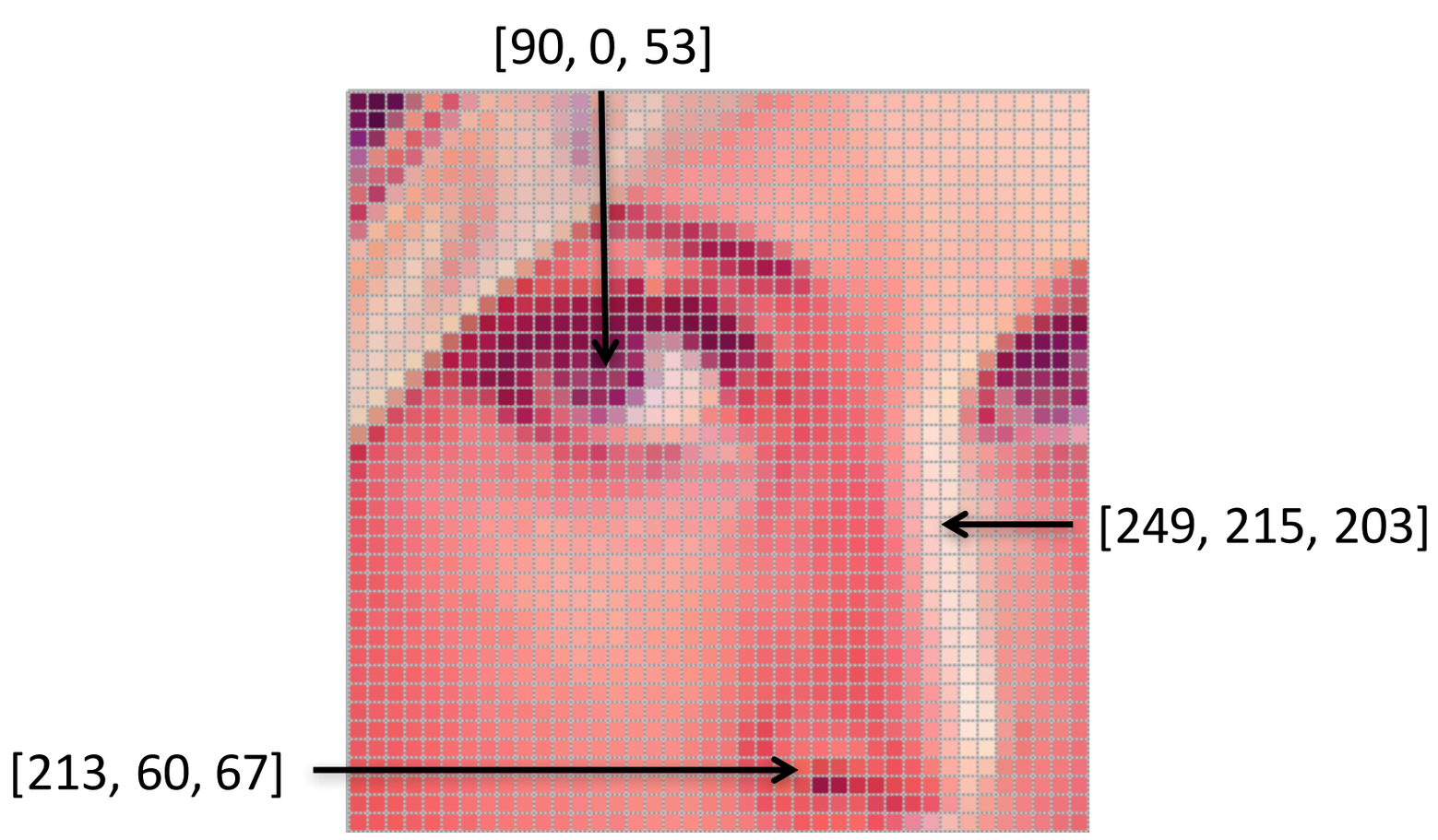


Image histograms

- Histograms are very simple image representations
  - They capture the global distribution of gray (or RGB) levels in a given image

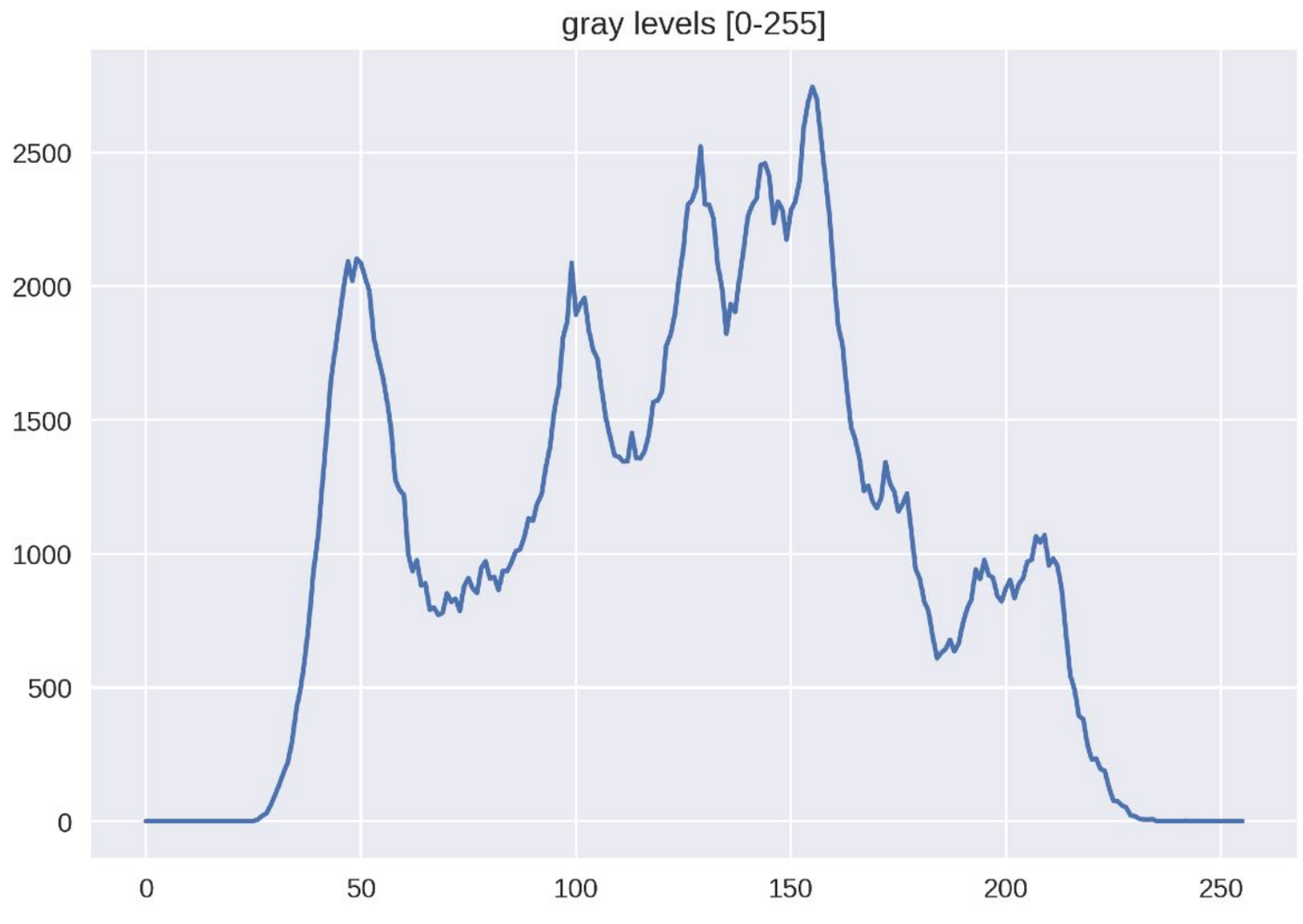


Image histograms

- Histograms are very simple image representations
  - They capture the global distribution of gray (or RGB) levels in a given image

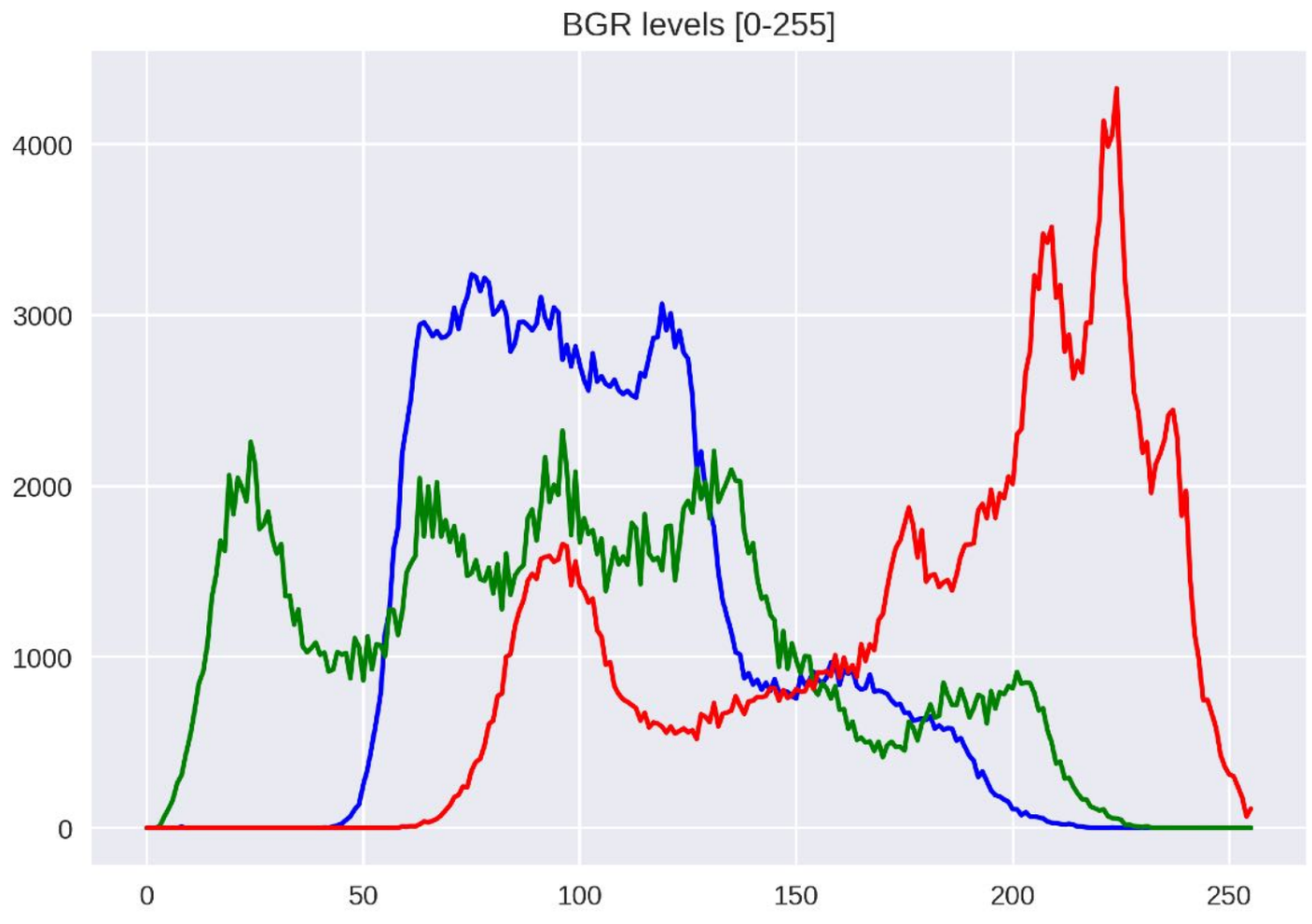
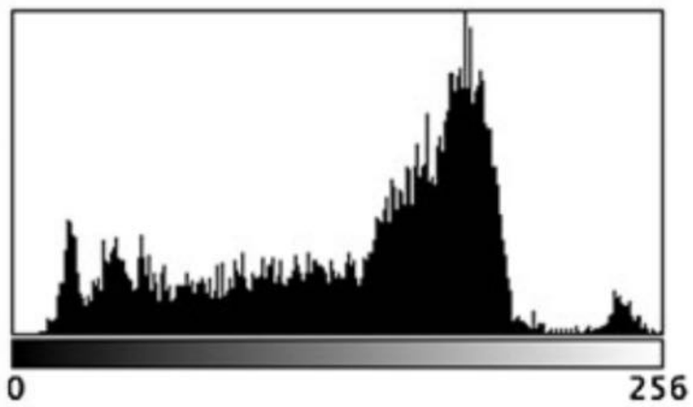
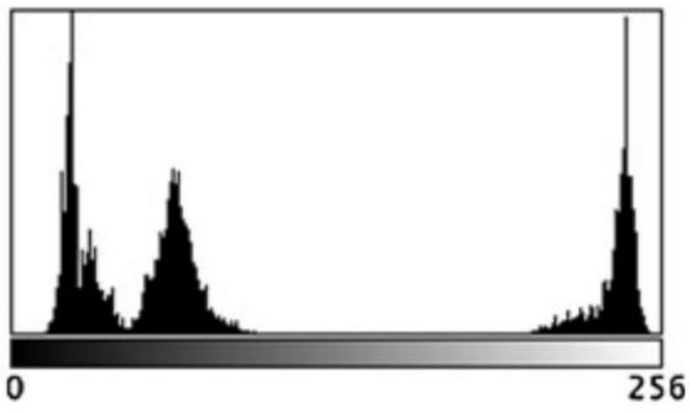


Image histograms



Count: 10192
Mean: 133.711
StdDev: 55.391

Min: 9
Max: 255
Mode: 178 (180)



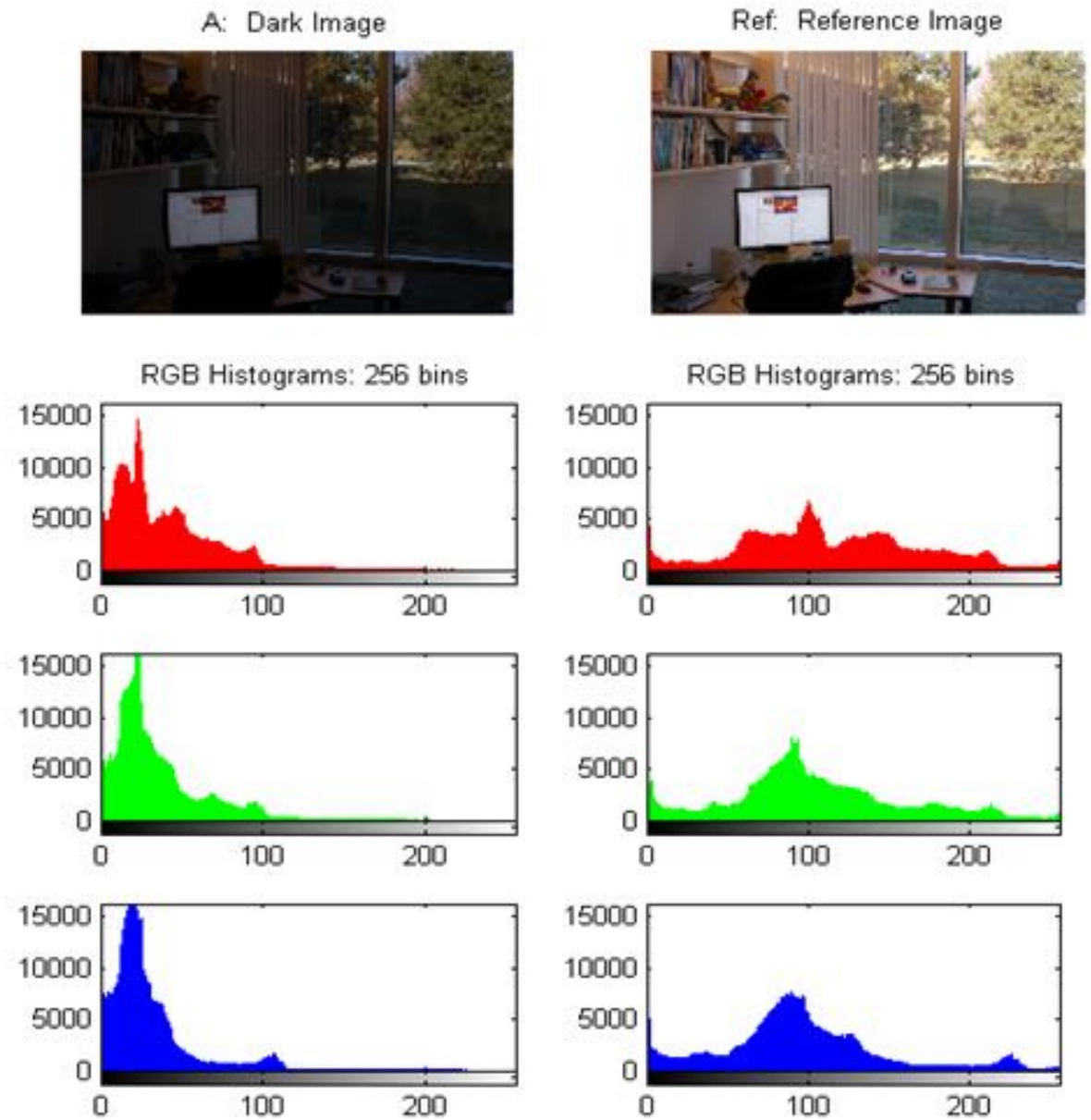
Count: 10192
Mean: 104.637
StdDev: 89.862

Min: 11
Max: 254
Mode: 23 (440)

Slide credit: M. Shah

Image histograms

- A use case: comparing two images (not robust)



Images as functions

- Formally, an image is a function f from $\mathbf{R}^2$ to $\mathbf{R}^M$:
  - $f(x, y)$ gives the **intensity** at position (x, y)
  - it is defined over a rectangle with a finite range:

$$f: [a, b] \times [a, b] \rightarrow [0, 255]$$

Domain support — range

  - A color image:

$$f(x, y) = \begin{bmatrix} r(x, y) \\ g(x, y) \\ b(x, y) \end{bmatrix}$$

Filters (linear systems)

- **Filtering:** forming a new image whose pixel values are transformed from original pixel values
- Which are the goals?
  - **extract** useful information from images
  - **transform** images into another domain where we can modify/enhance image properties

Filters (linear systems)

de-noising example

Salt and pepper noise

super-resolution example

in-painting example

Filters (linear systems)

- We define a filter (system) as a unit that converts an input function $f[n, m]$ into an output (response) function $g[n, m]$, where (n, m) are the independent variables

- In the case of images, (n, m) represents the **spatial position** in the image

$$f[n, m] \rightarrow \boxed{\text{System } \mathcal{S}} \rightarrow g[n, m]$$

Filter example: moving average

- 2D moving average over a 3×3 window of neighborhood (this is also referred as to *box filter*)

$$g[n,m] = \frac{1}{9} \sum_{k=n-1}^{n+1} \sum_{l=m-1}^{m+1} f[k,l]$$

$$= \frac{1}{9} \sum_{k=-1}^{1} \sum_{l=-1}^{1} f[n-k, m-l]$$

h

$\frac{1}{9}$

1	1	1
1	1	1
1	1	1

Image Kernel

$$(f * h)[n,m] = \sum_{k,l} f[k,l]\, h[n-k, m-l]$$

Convolution

Filter example: moving average

$F[x, y]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$G[x, y]$

	0								

$$(f * h)[n, m] = \sum_{k, l} f[k, l]\, h[n\text{-}k, m\text{-}l]$$

Convolution

Filter example: moving average

$F[x, y]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$G[x, y]$

	0	10							

$$(f * h)[n, m] = \sum_{k,\, l} f\,[k, l]\; h[n\text{-}k, m\text{-}l]$$

Convolution

Filter example: moving average

$F[x, y]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$G[x, y]$

	0	10	20						

$$(f * h)[n, m] = \sum_{k, l} f[k, l]\, h[n\text{-}k, m\text{-}l]$$

Convolution

Filter example: moving average

$F[x, y]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$G[x, y]$

	0	10	20	30					

$$(f * h)[n, m] = \sum_{k, l} f[k, l]\, h[n\text{-}k, m\text{-}l]$$

Convolution

Filter example: moving average

$F[x, y]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$G[x, y]$

	0	10	20	30	30				

$$(f * h)[n, m] = \sum_{k,\, l} f\,[k, l]\; h[n\text{-}k, m\text{-}l]$$

Convolution

Filter example: moving average

$F[x, y]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

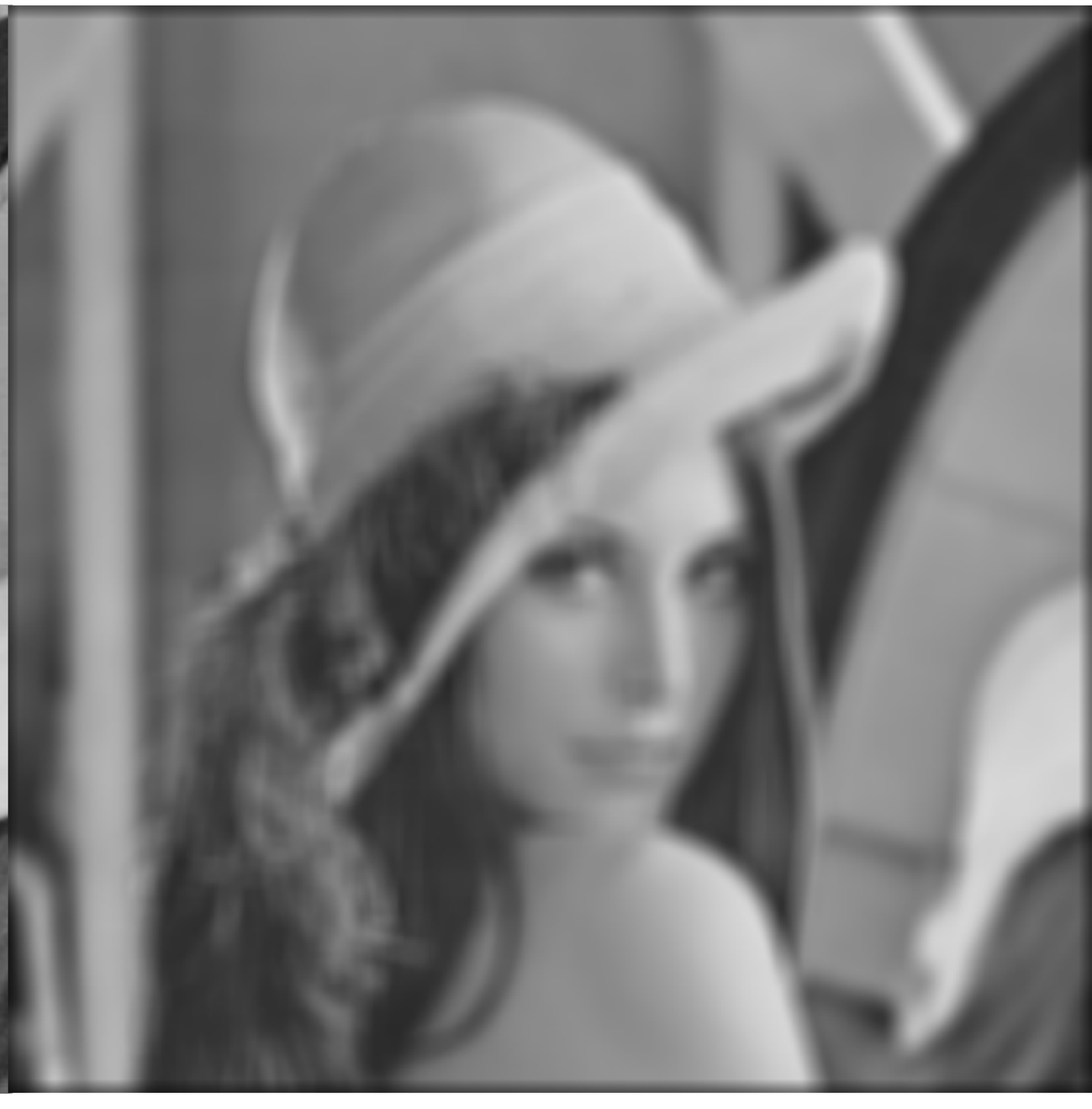
$G[x, y]$

	0	10	20	30	30	30	20	10	
	0	20	40	60	60	60	40	20	
	0	30	60	90	90	90	60	30	
	0	30	50	80	80	90	60	30	
	0	30	50	80	80	90	60	30	
	0	20	30	50	50	60	40	20	
	10	20	30	30	30	30	20	10	
	10	10	10	0	0	0	0	0	

$$(f * h)[n, m] = \sum_{k, l} f[k, l]\, h[n-k, m-l]$$

Convolution

Filter example: moving average



Filter example: moving average

- In summary, this filter:
  - Replaces each pixel with an average of its neighbors
  - Achieves smoothing effect (i.e. it removes sharp features)

$$h$$

$$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Image Kernel

Check also this interactive demo: http://setosa.io/ev/image-kernels/

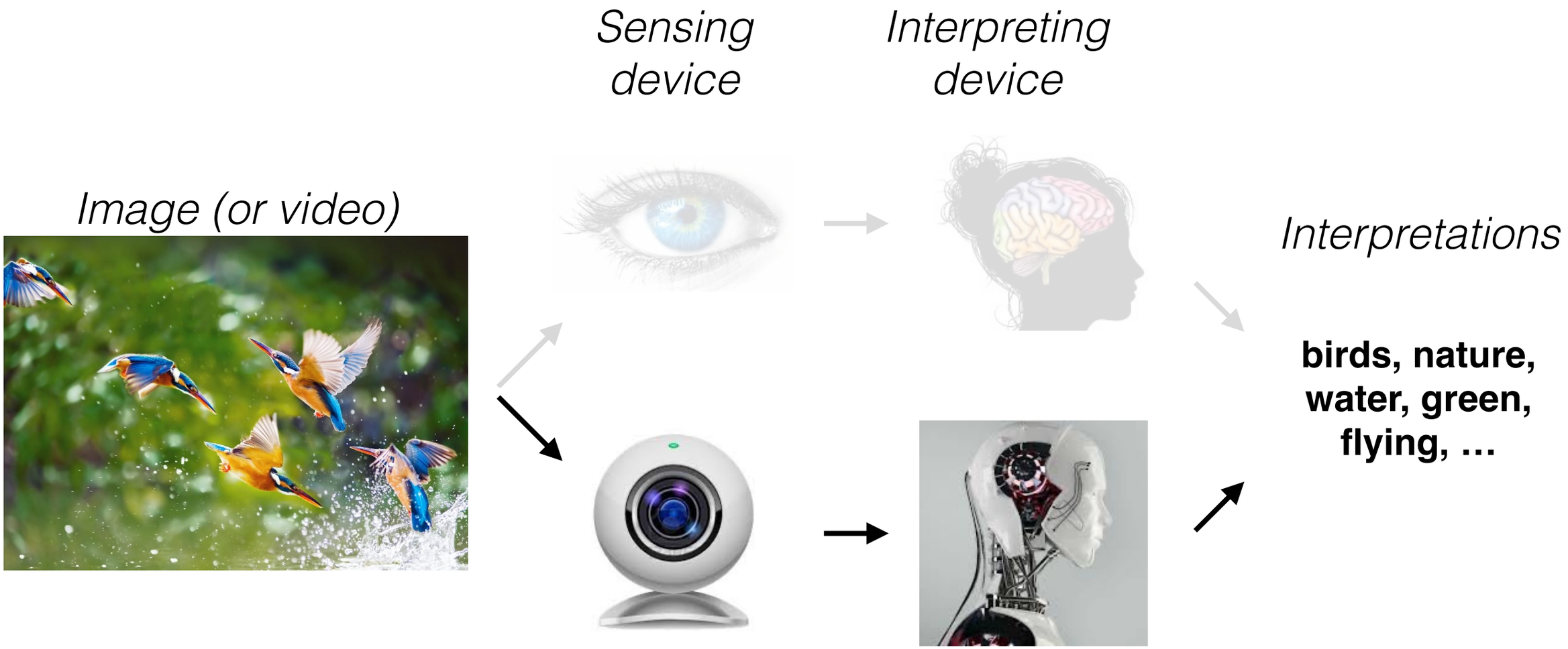
Image filtering: main idea

- Compute a function of the *local neighborhood* at each pixel in the image
  - The function is specified by a “filter” saying how to combine values from neighbors

- Applications of image filtering:
  - extract information (edges, corners, blobs, …)
  - detect patterns (template matching)
  - de-noising, super-resolution, in-painting

What is computer vision?

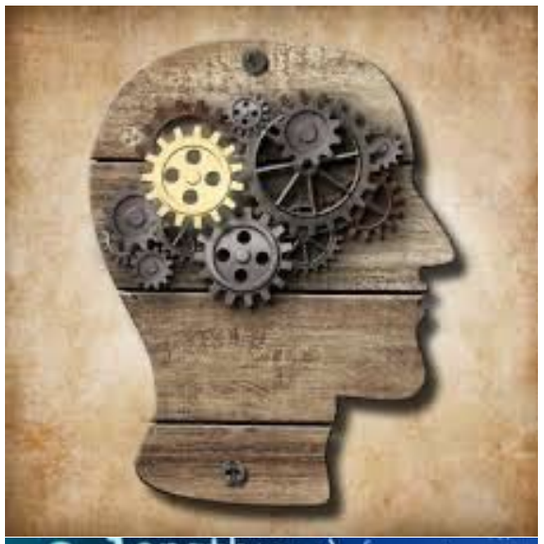
- Visual recognition is the holy grail of computer vision



The goal of computer vision

- Q: What objects are in the image?

Input



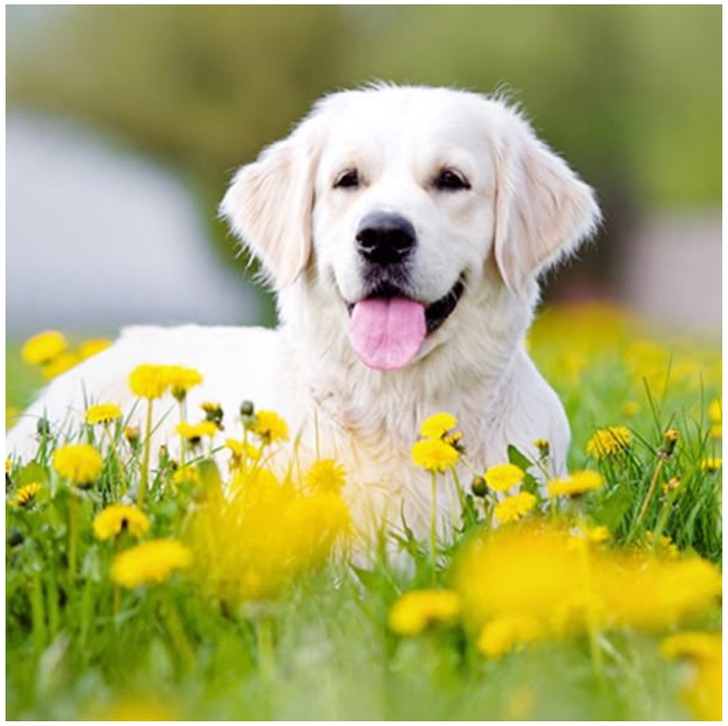
Output

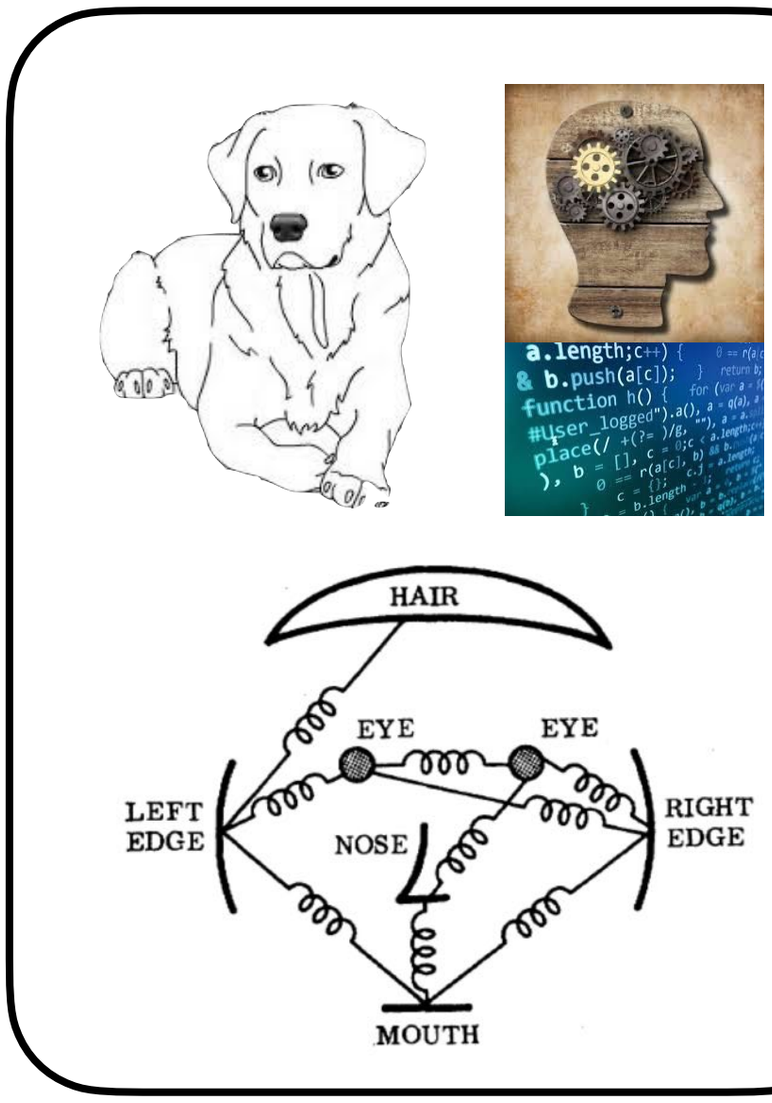
“dog”

The goal of computer vision

- Visual object recognition: “the old way”

Input





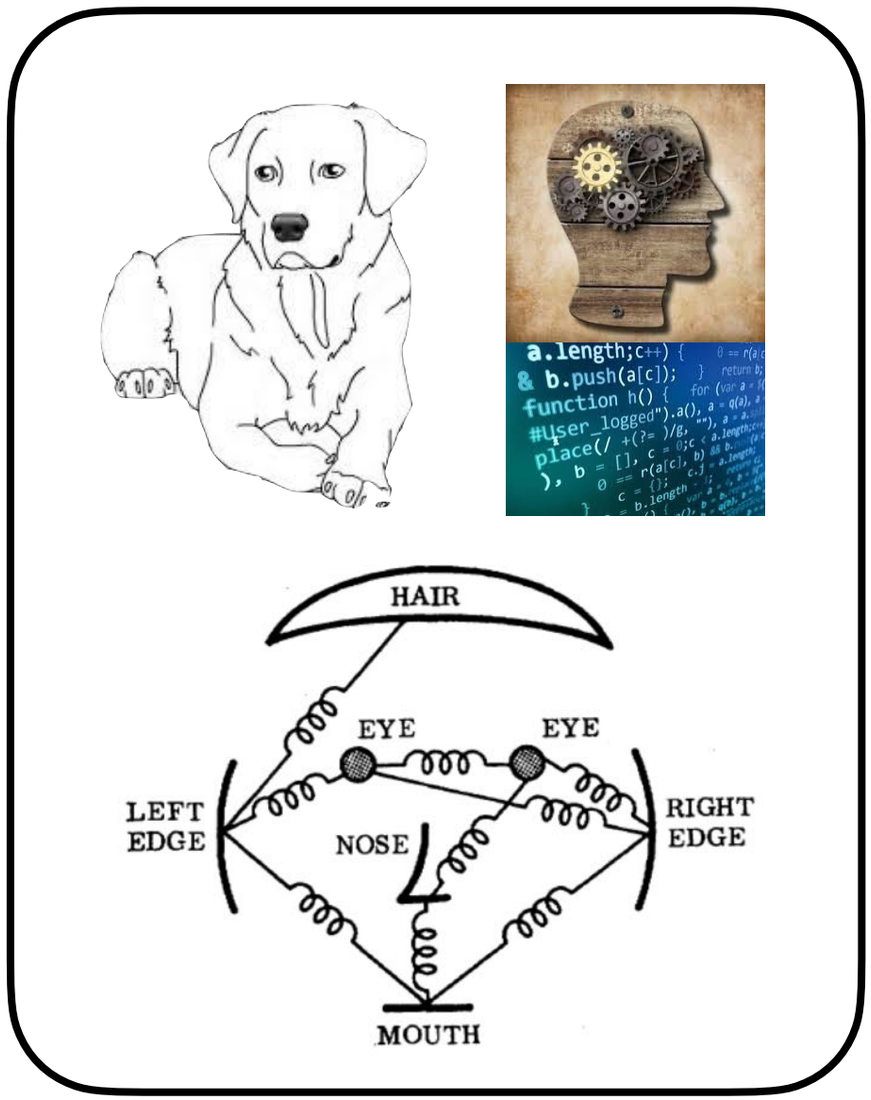
Output

“dog”

Exemplar-based object recognition

- It worked quite reasonably well, but…

Input



Output

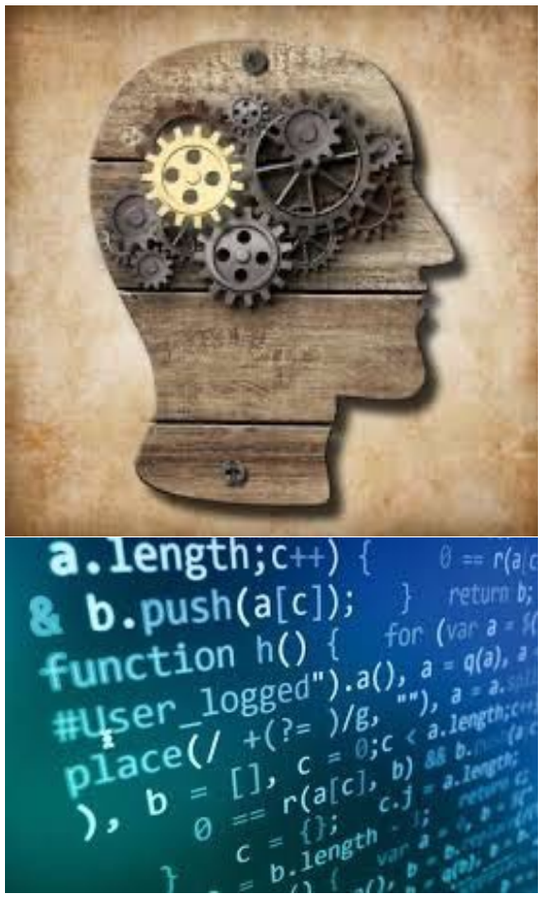
“dog”

Teaching machines to see

- The Machine Learning (data-driven) paradigm:

Input

Machine Learning



Output

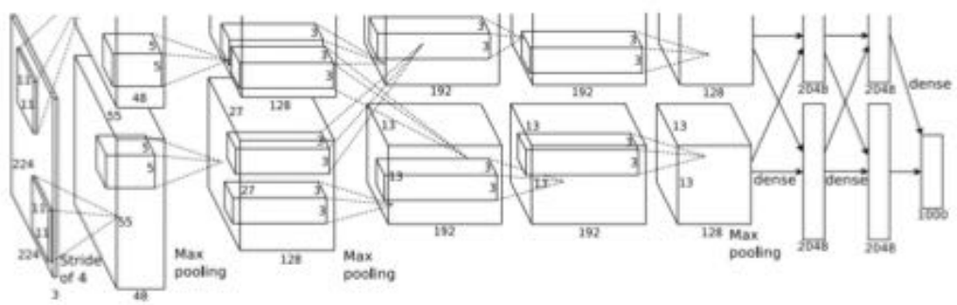
“car”
“dog”
“cat”
“person”
“tree”
“chair”
“bike”
“sky”
“computer”
…

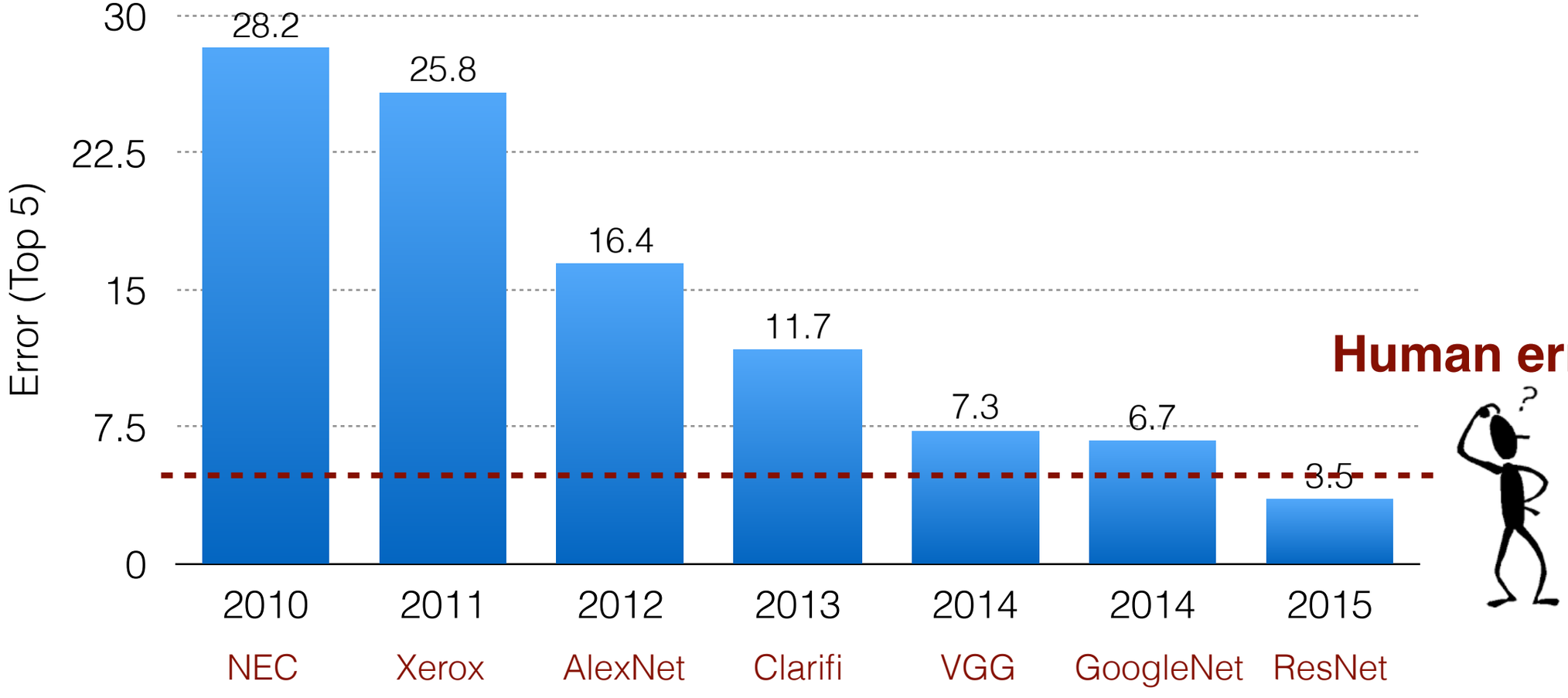
The rise of Deep Learning

https://www.youtube.com/watch?v=40riCqvRoMs

The rise of Deep Learning

- ImageNet-LSVRC (classification) results over the years



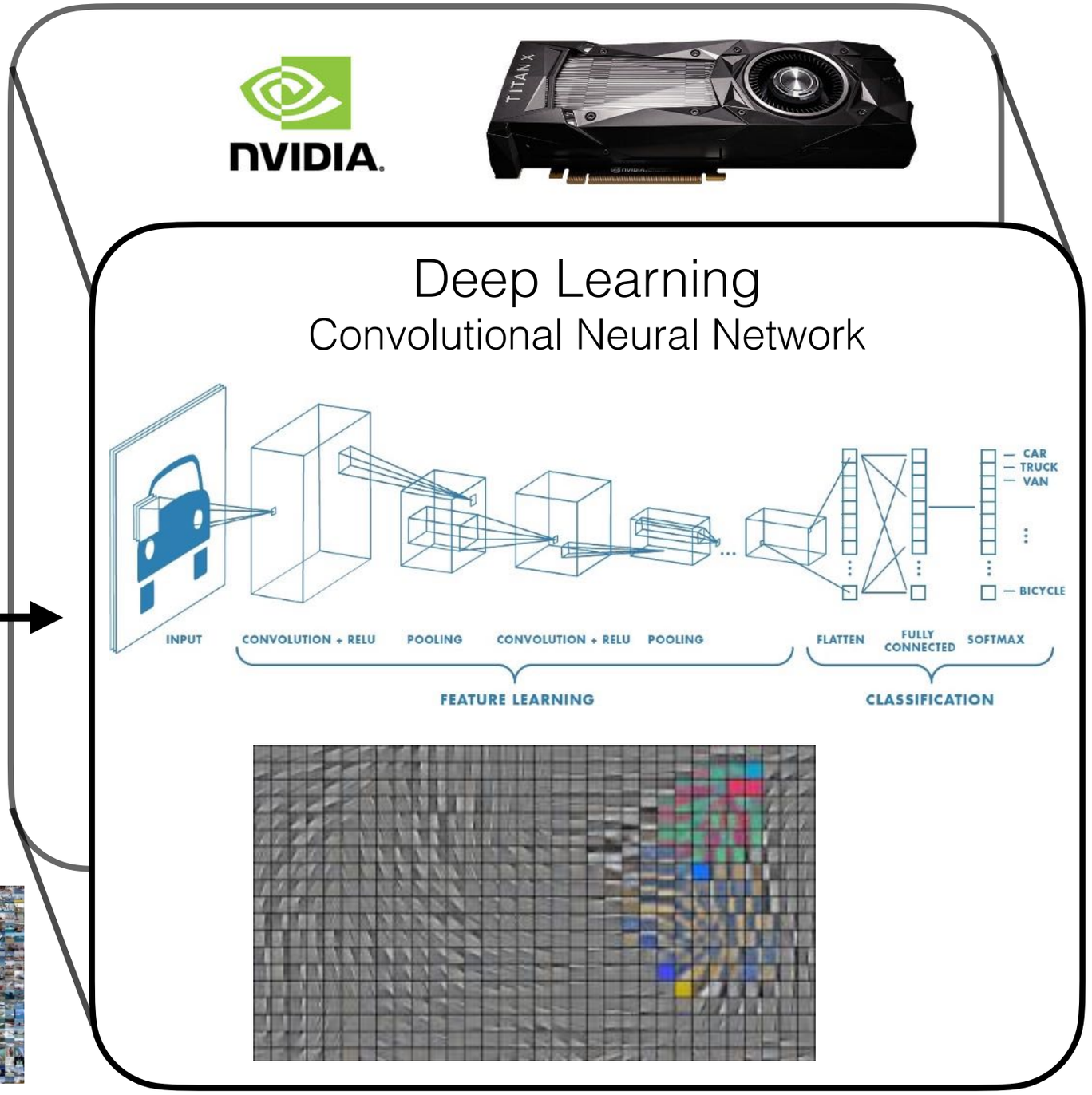


Teaching machines to see (in the DL era)

Input

...

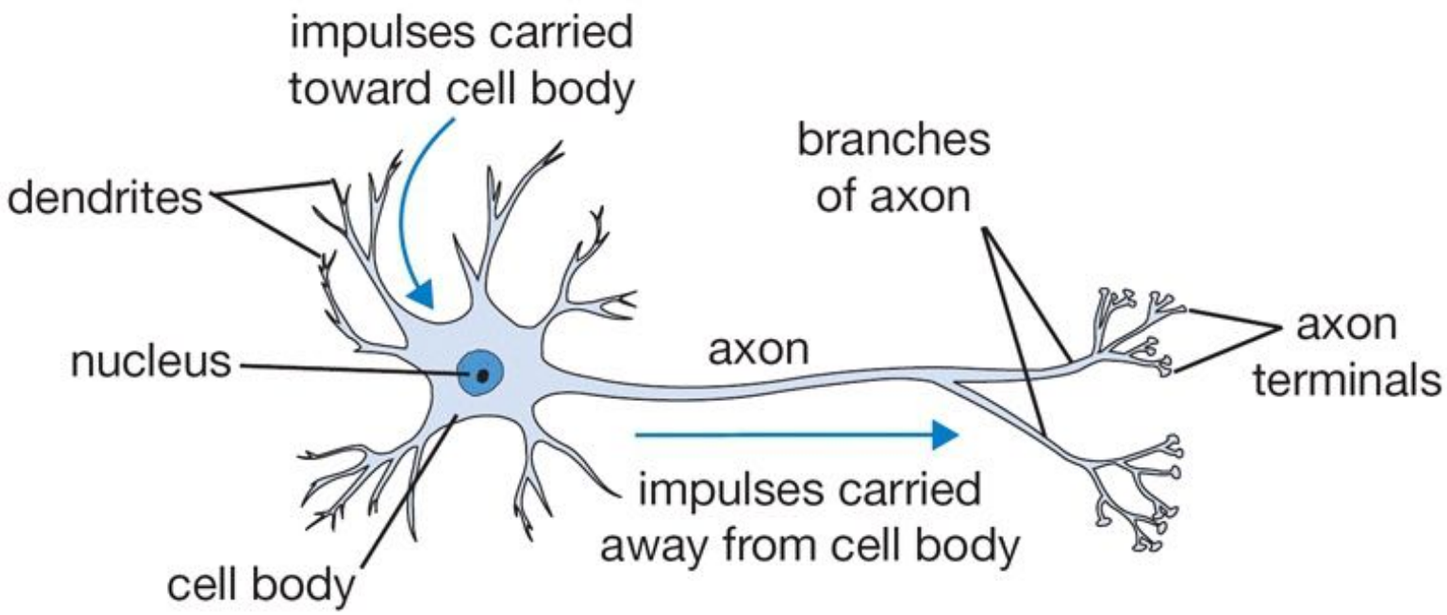
15M images, 22K classes



Output

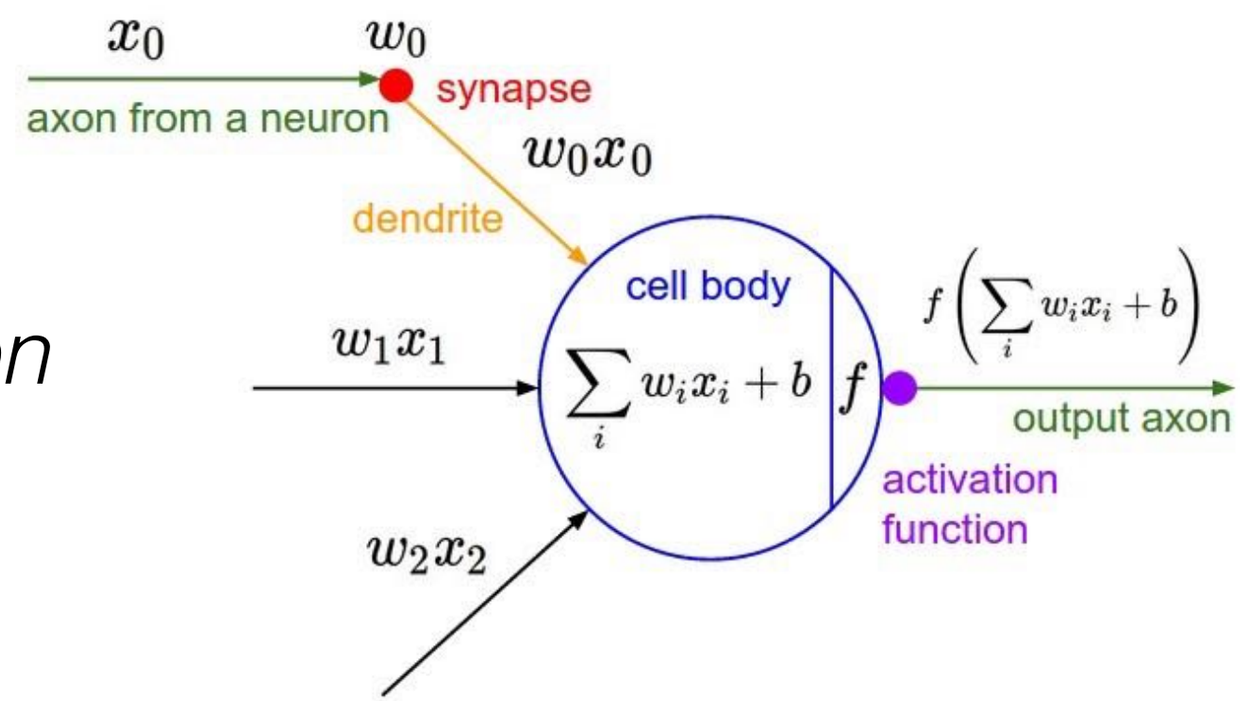
"car"
"dog"
"cat"
"person"
"tree"
"chair"
"Bike"
"sky"
"computer"
...

Artificial Neural Networks



Biological Neuron

A simple Artificial Neuron (perceptron)



e.g. sigmoid activation

$$\sigma(x) = \frac{1}{1+e^{-x}}$$

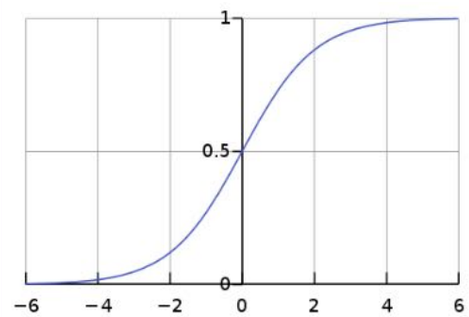
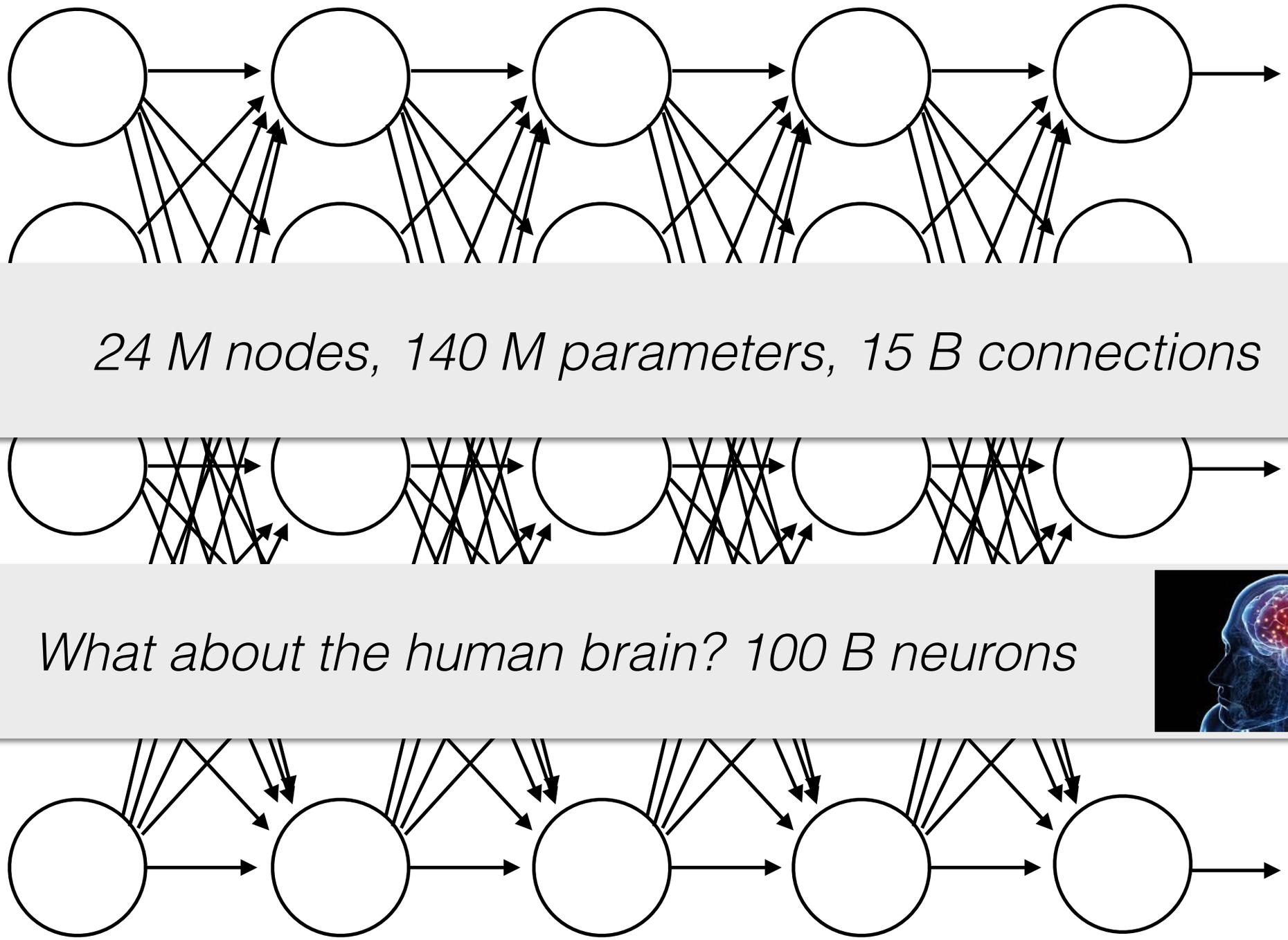


Image credit: Fei-Fei, Karpathy, Johnson

Deep Networks



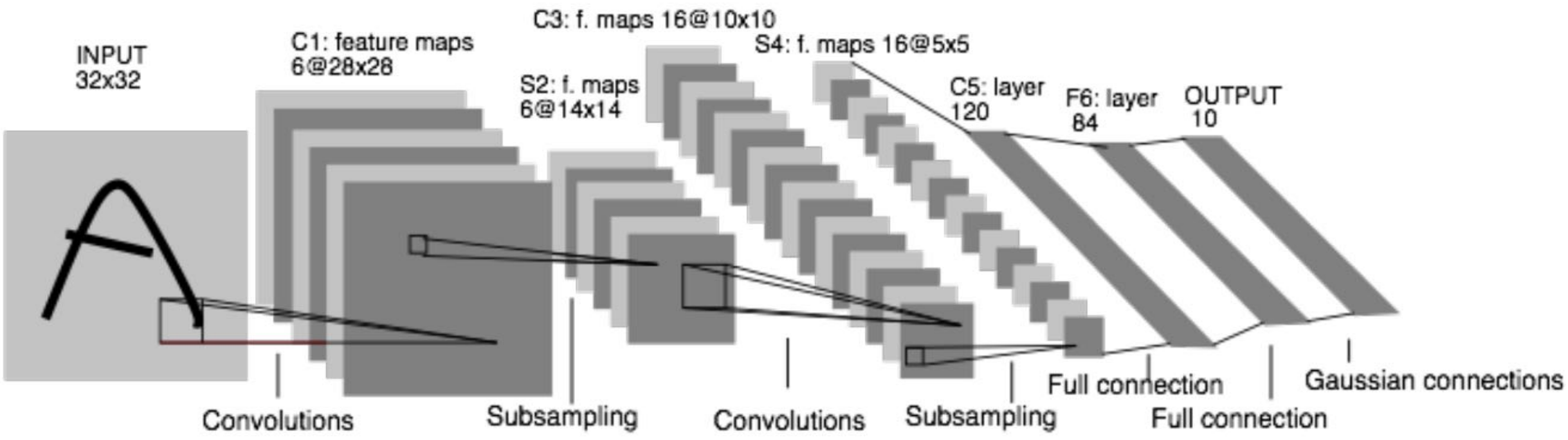
24 M nodes, 140 M parameters, 15 B connections

What about the human brain? 100 B neurons

Convolutional Neural Networks

- CNNs: the assumption that the inputs are images allows us to encode certain local properties into the architecture

LeNet-5



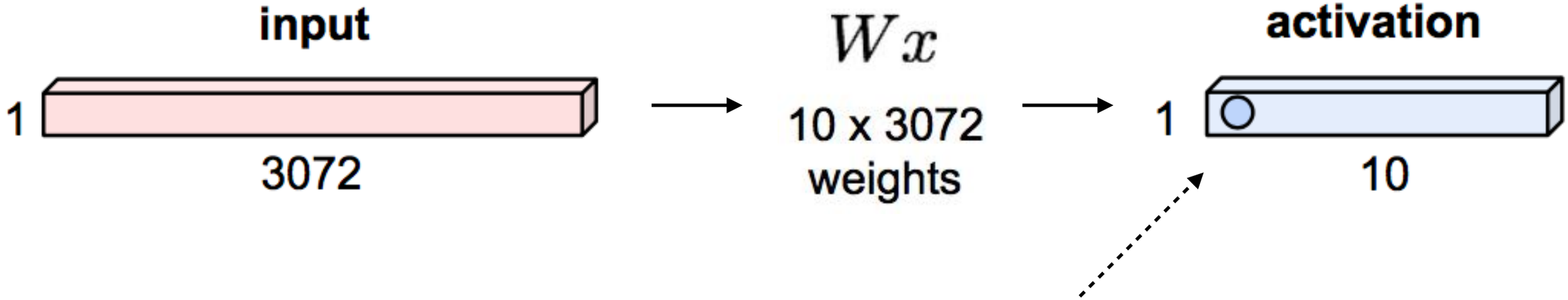
Y.LeCun, L.Bottou, Y.Bengio, P.Haffner, “Gradient-based learning applied to document recognition”, Proc.IEEE 1998

Convolutional Neural Networks

- ConvNet (CNN): a sequence of layers where each layer transforms one volume of activations to another through a *differentiable* function
- We have three main types of layers:
  - Convolutional Layer
  - Pooling Layer
  - Fully Connected Layer (exactly as in “regular” ANN)
- We stack these layers to form a full CNN architecture

Fully Connected layer

- input: 32x32x3 image -> *stretch to* 3072x1

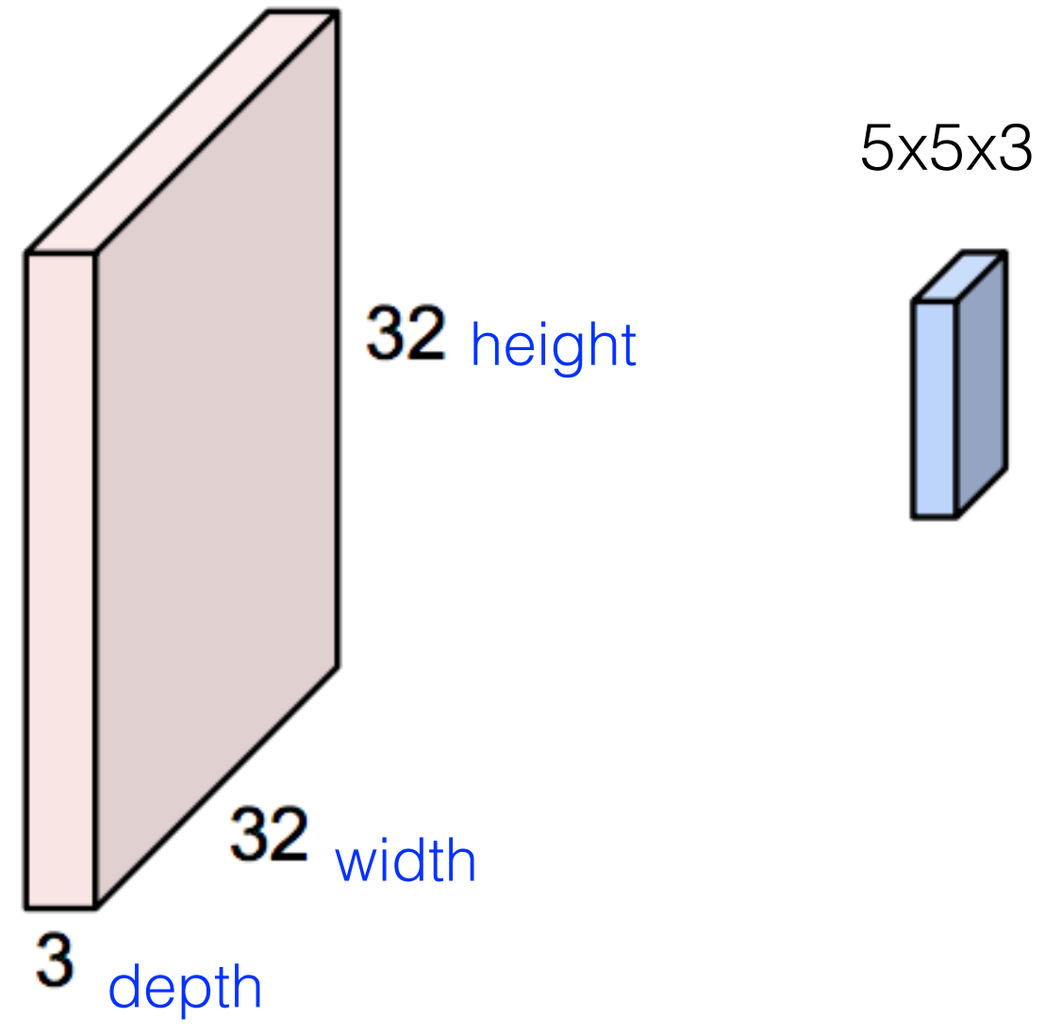


activation: *1 number, i.e. the result of taking a dot product between a row of W and the input (a 3072-d dot product)*

Slide credit: Fei-Fei, Karpathy, Johnson

Convolution layer

- input: 32x32x3 image -> *preserve spatial structure*



Convolve the filter with the image,
i.e. “slide over the image spatially,
computing dot products”

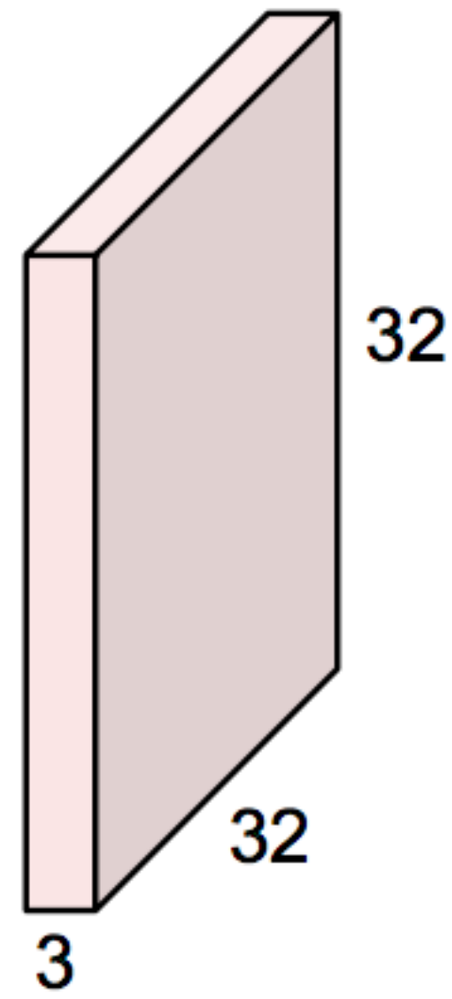
Slide credit: Fei-Fei, Karpathy, Johnson

Convolution layer

- input: 32x32x3 image -> *preserve spatial structure*

32x32x**3** image

Note: filters always extend the full depth of the input volume



5x5x**3** filter

Convolve the filter with the image, i.e. “slide over the image spatially, computing dot products”

Slide credit: Fei-Fei, Karpathy, Johnson

Image filtering and convolution

- Image filtering example: Gaussian smoothing

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

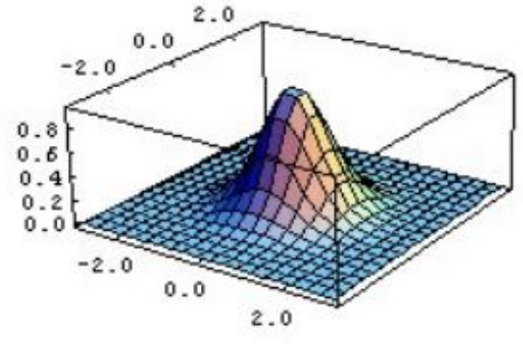
$F[x, y]$

$$\frac{1}{16}\begin{array}{|c|c|c|}\hline 1 & 2 & 1 \\ \hline 2 & 4 & 2 \\ \hline 1 & 2 & 1 \\ \hline\end{array}$$

$H[x, y]$

This kernel is an approximation of a 2D Gaussian function

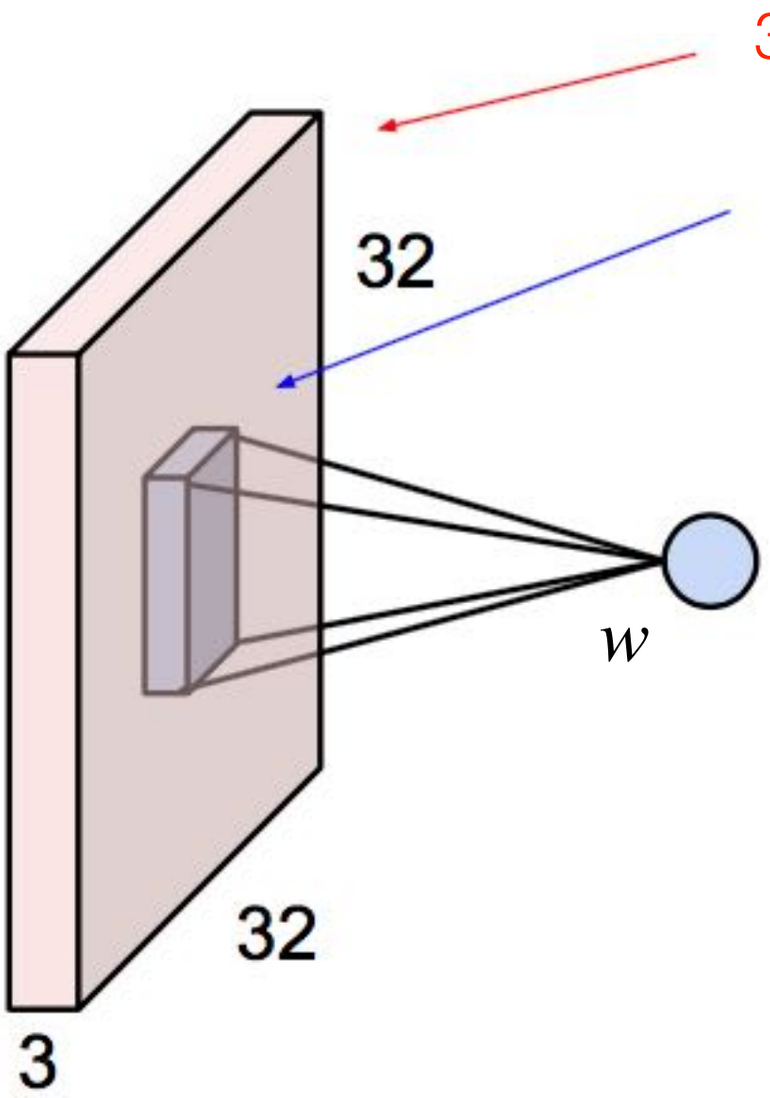
$$h(u,v) = \frac{1}{2\pi\sigma^2} e^{-\frac{u^2+v^2}{\sigma^2}}$$



Convolution: $$(f * h)[n, m] = \sum_{k,\, l} f[k, l]\, h[n\text{-}k, m\text{-}l]$$

Convolution layer

- input: 32x32x3 image -> *preserve spatial structure*



activation: *1 number, i.e. the result of taking a dot product between the filter and a 5x5x3 chunk of the image*

*(i.e. 5*5*3=75-d dot product + bias)*

$$w^T x + b$$

Slide credit: Fei-Fei, Karpathy, Johnson

Convolutional Neural Networks

- Visualization of a simple ConvNet architecture:

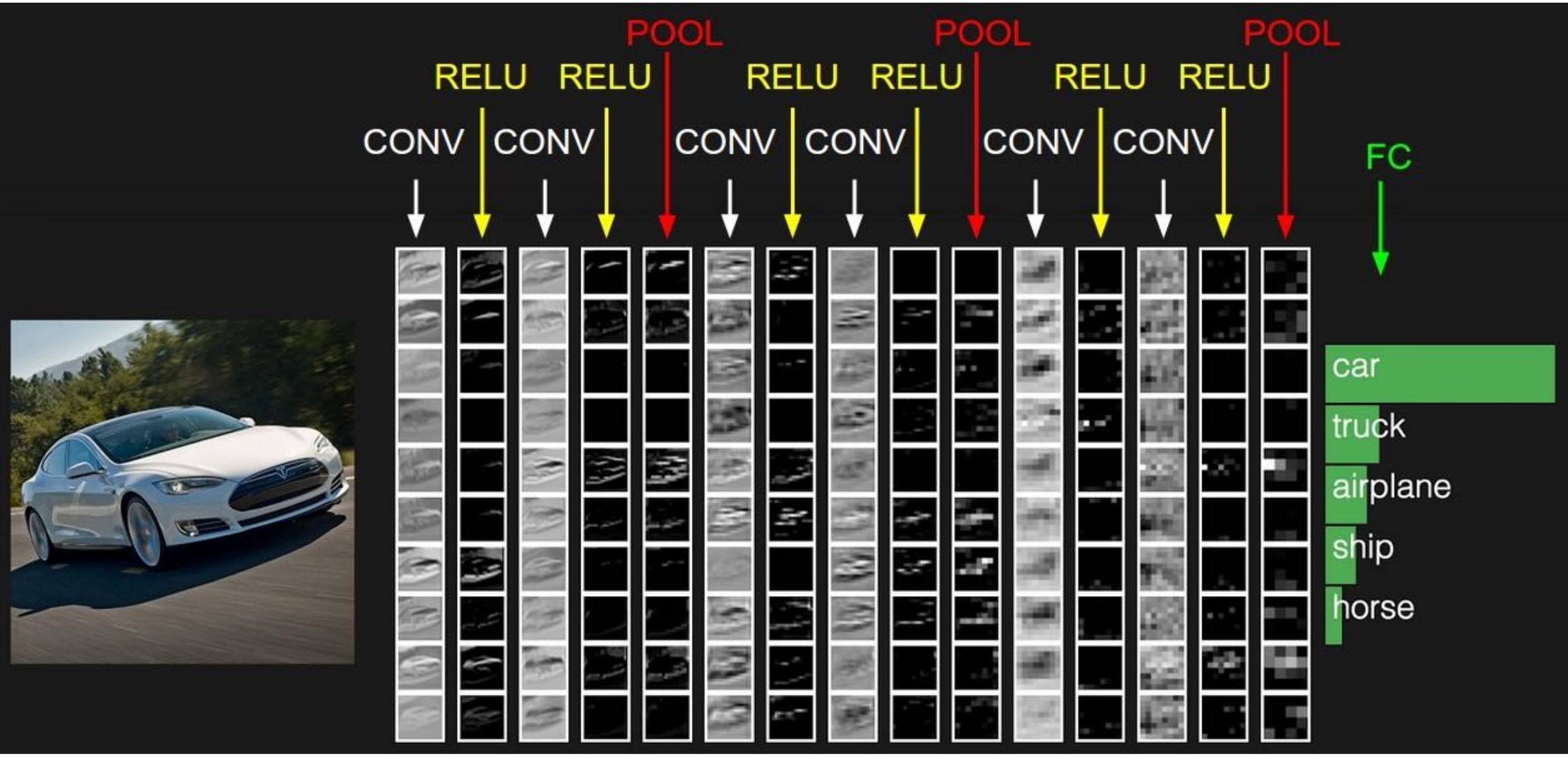
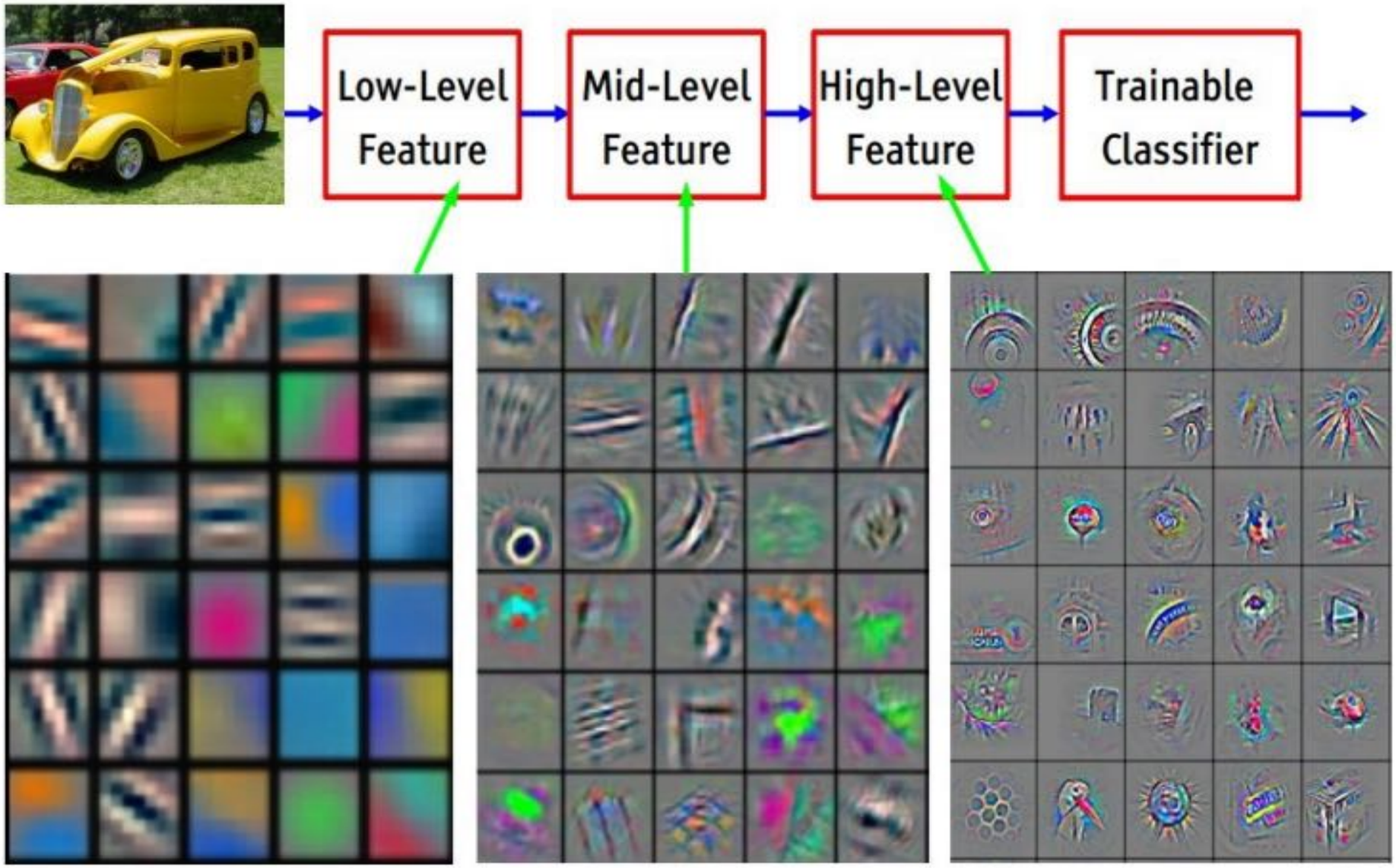


Image credit: Fei-Fei, Karpathy, Johnson

Convolutional Neural Networks

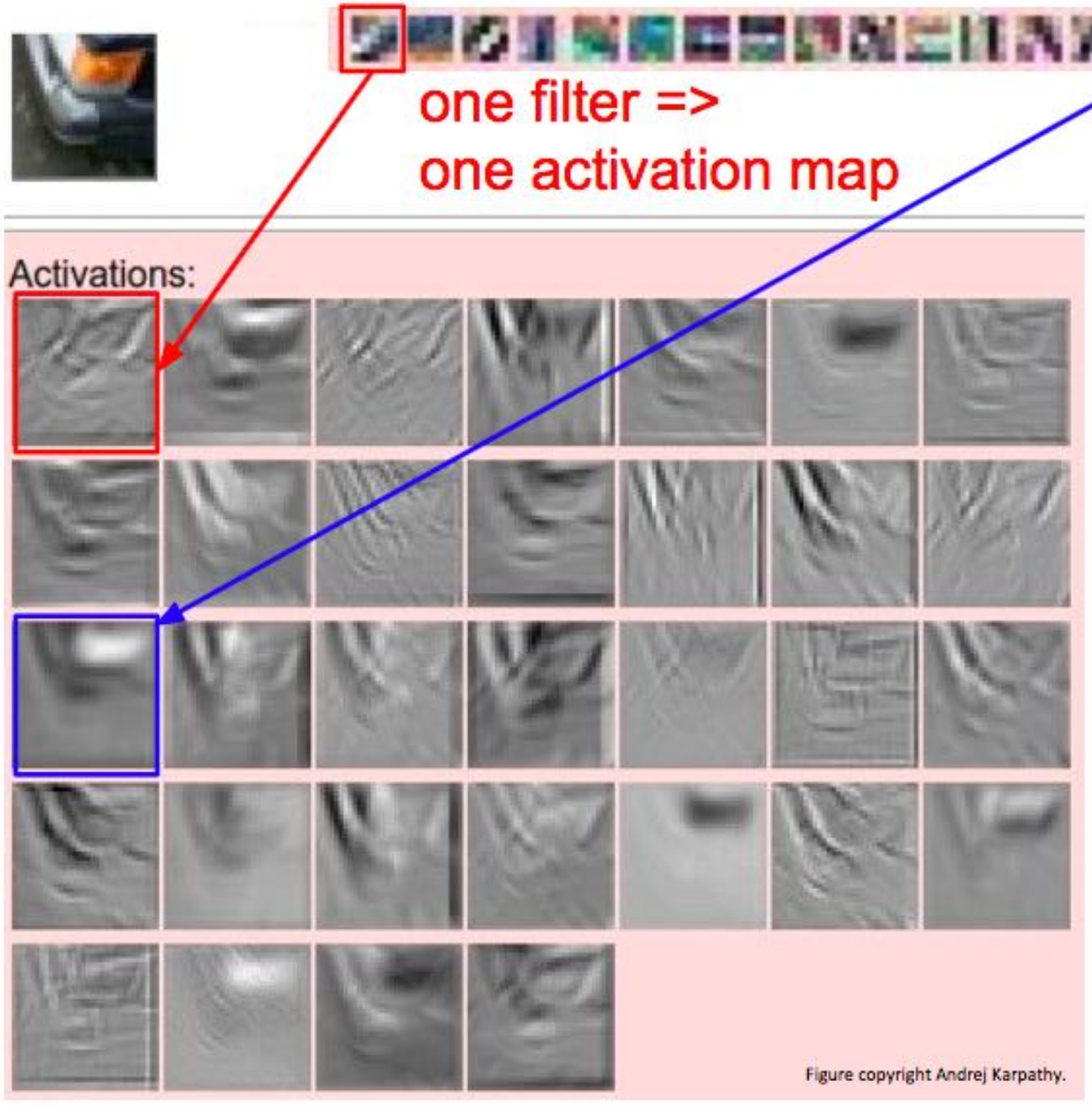
- Preview: feature visualization of CNN (trained on ImageNet)



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

Slide credit: Y. LeCun

Convolutional Neural Networks



example 5x5 filters
(32 total)

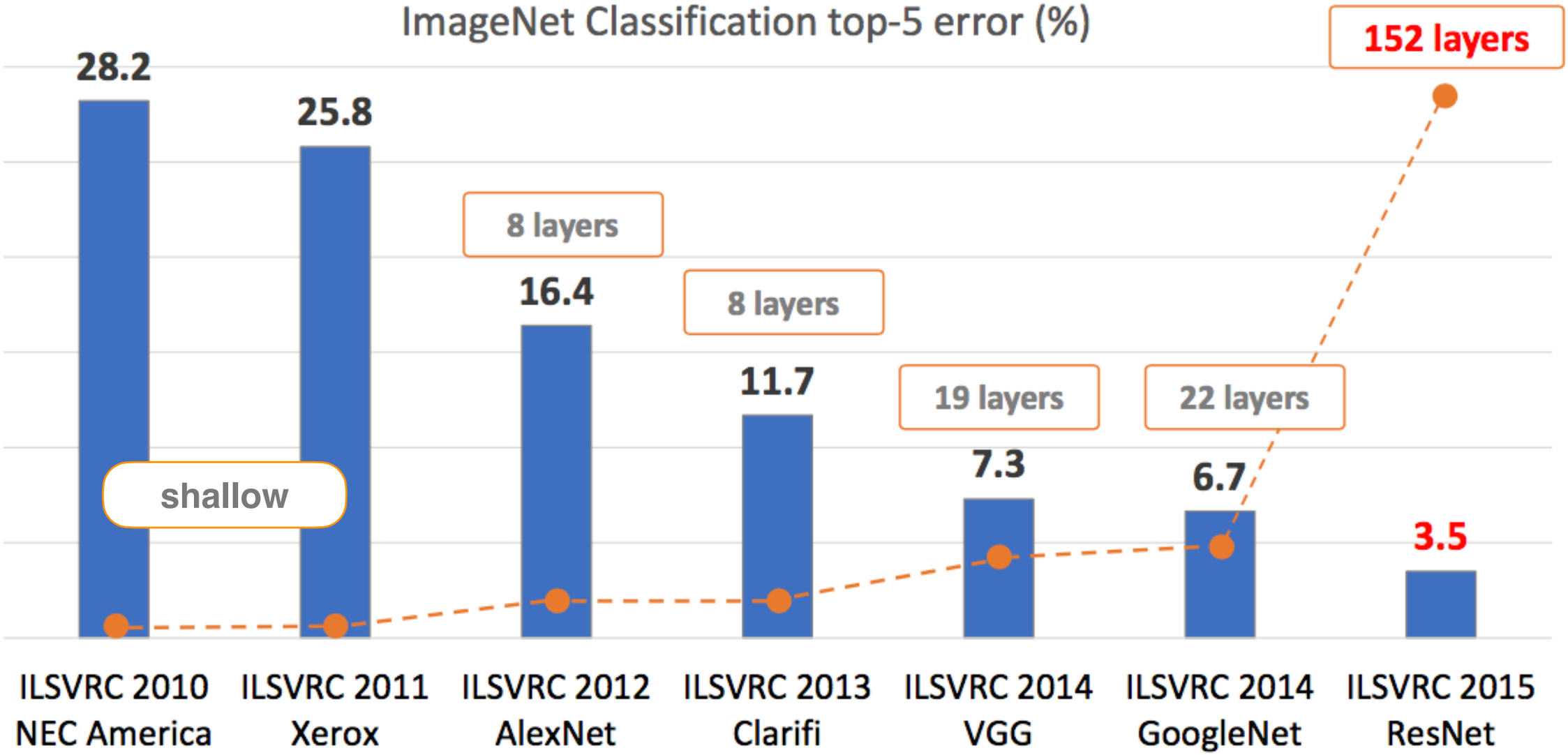
We call the layer convolutional because it is related to convolution of two signals:

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

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

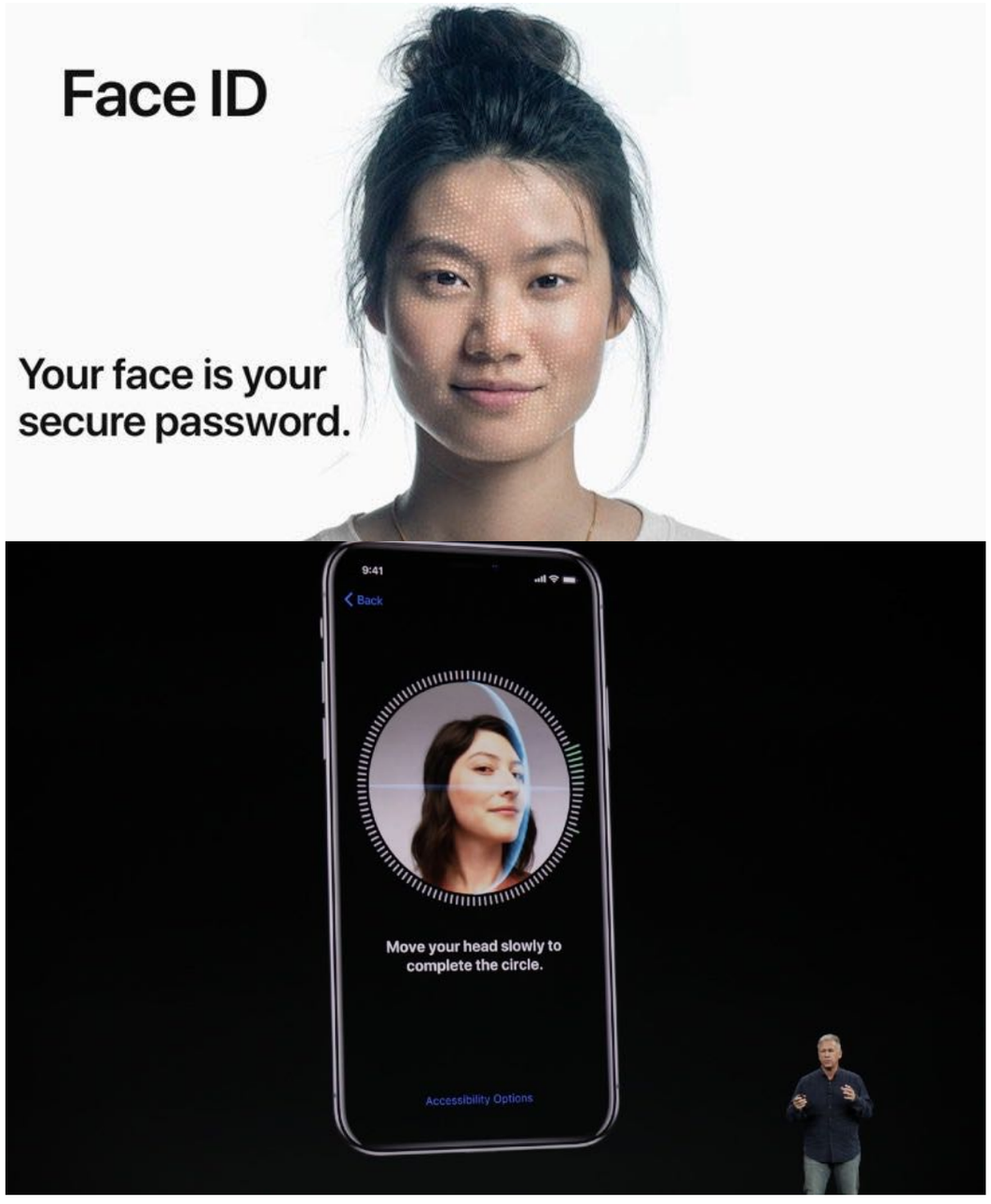
The rise of Deep Learning

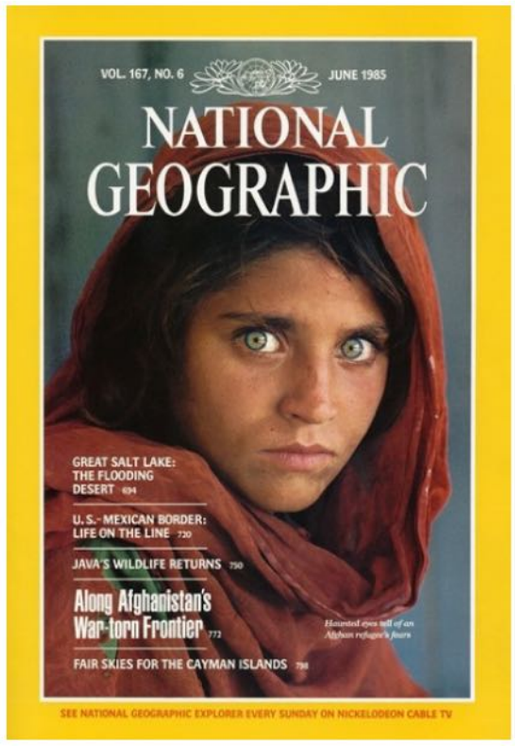
- Revolution of depth: how deep is deep?

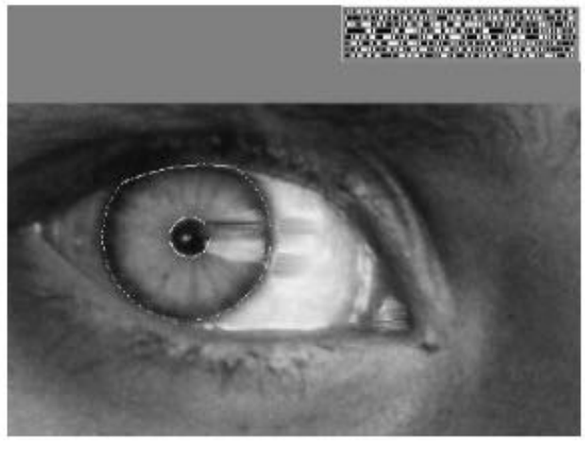


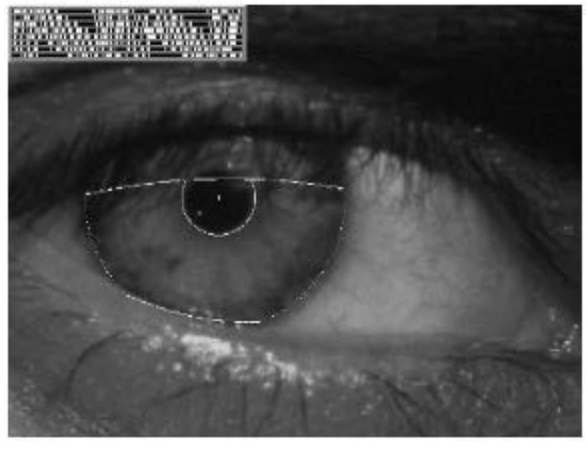
Deep Networks are everywhere

Face recognition, Biometrics





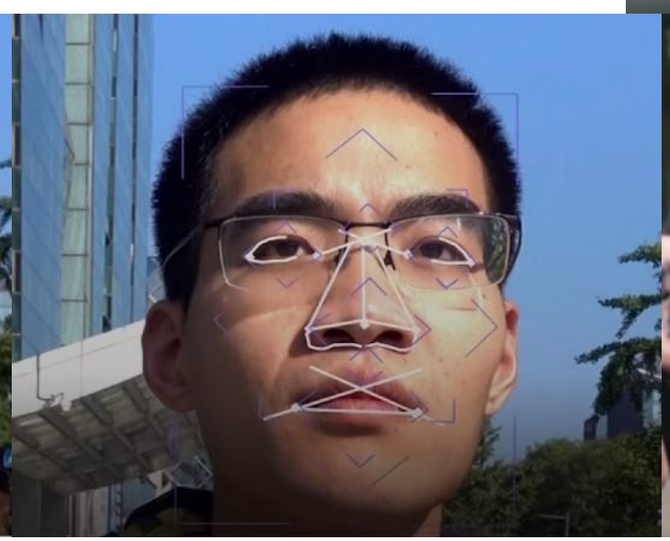


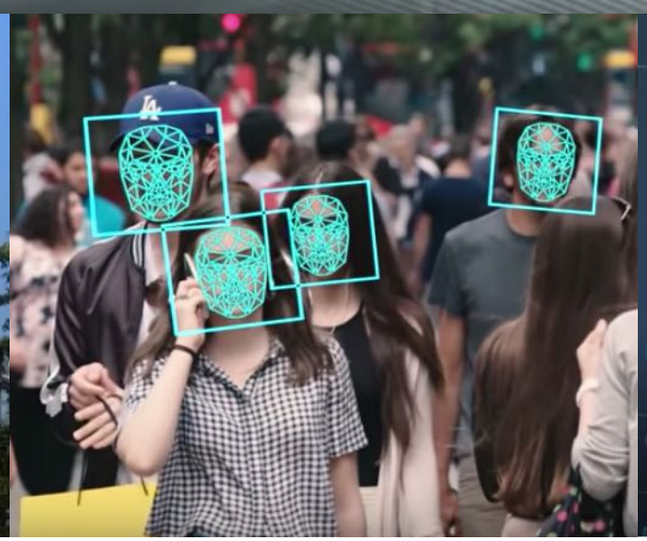


https://www.youtube.com/watch?v=nT_PXjLol_8

Deep Networks are everywhere

City-scale surveillance, Security

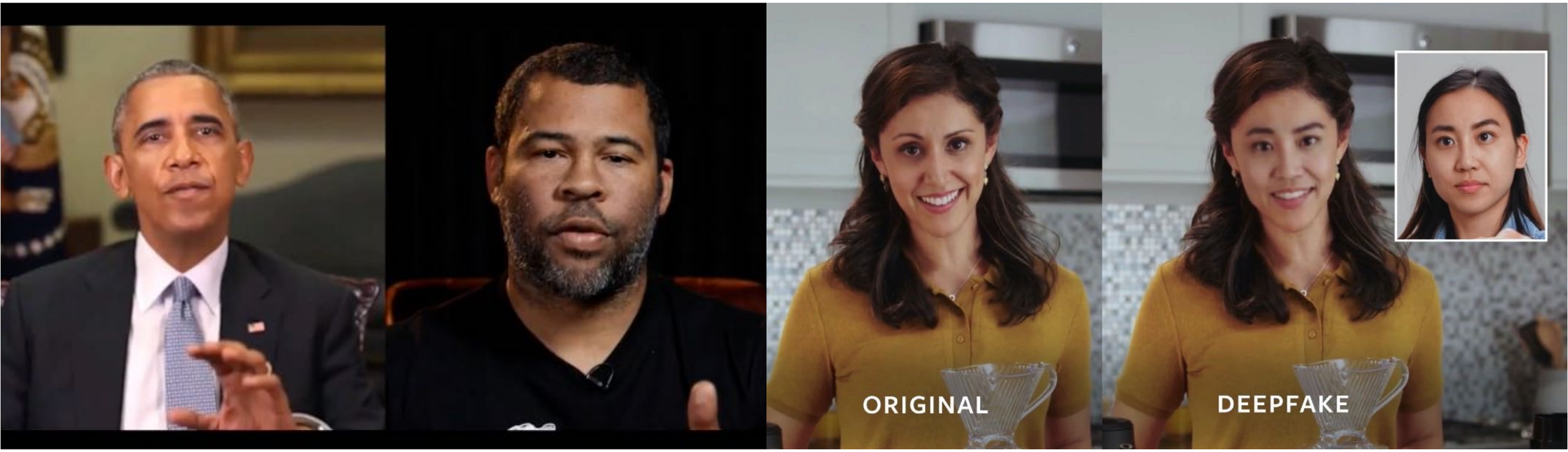




https://www.youtube.com/watch?v=lH2gMNrUuEY

Deep Networks are everywhere

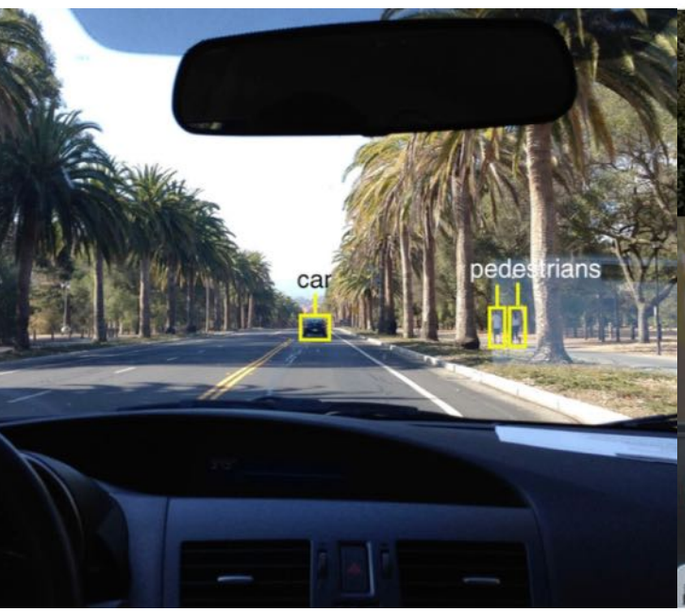
AI synthesized media content, “Deep fakes”

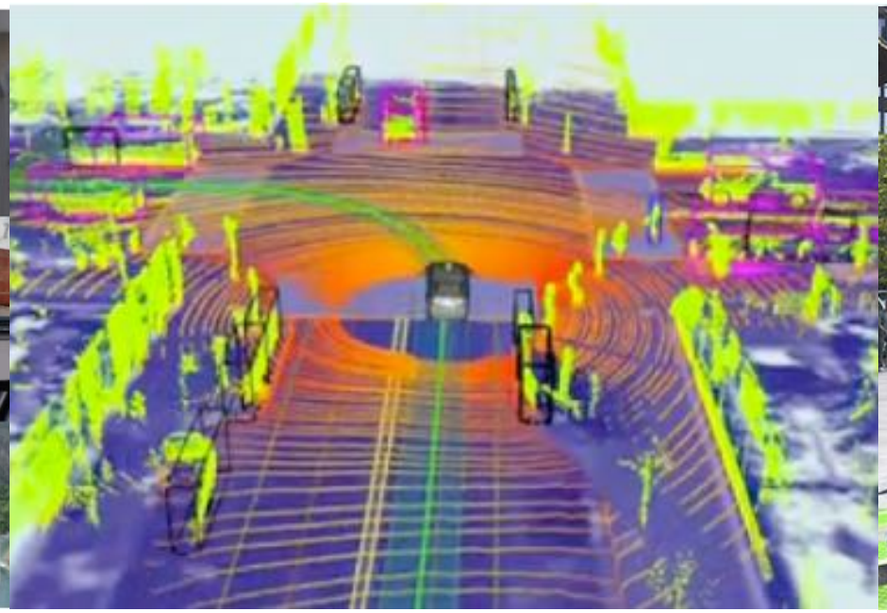


https://www.youtube.com/watch?v=gLoI9hAX9dw

Deep Networks are everywhere

Intelligent transportation, Self-driving cars

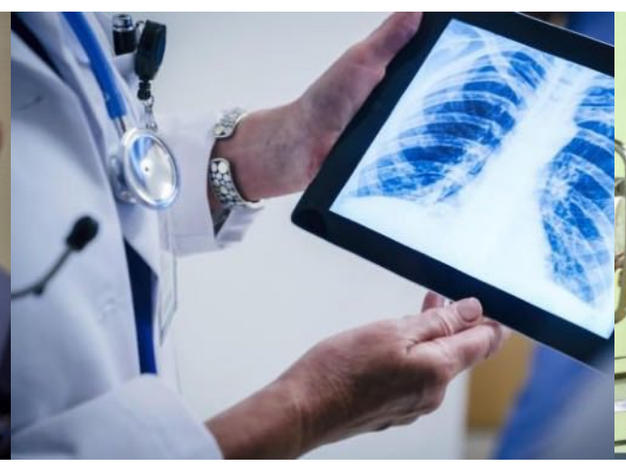




https://www.youtube.com/watch?v=1W9q5SjaJTc

Deep Networks are everywhere

Ambient intelligence, Mobility, Retails and Hospitals



Contact

- **Office:** Torre Archimede 6CD, room 622
- **Office hours** (ricevimento)**:** Monday 9:00-11:00

✉ lamberto.ballan@unipd.it

🏠 http://www.lambertoballan.net

🏠 http://vimp.math.unipd.it

@ twitter.com/lambertoballan