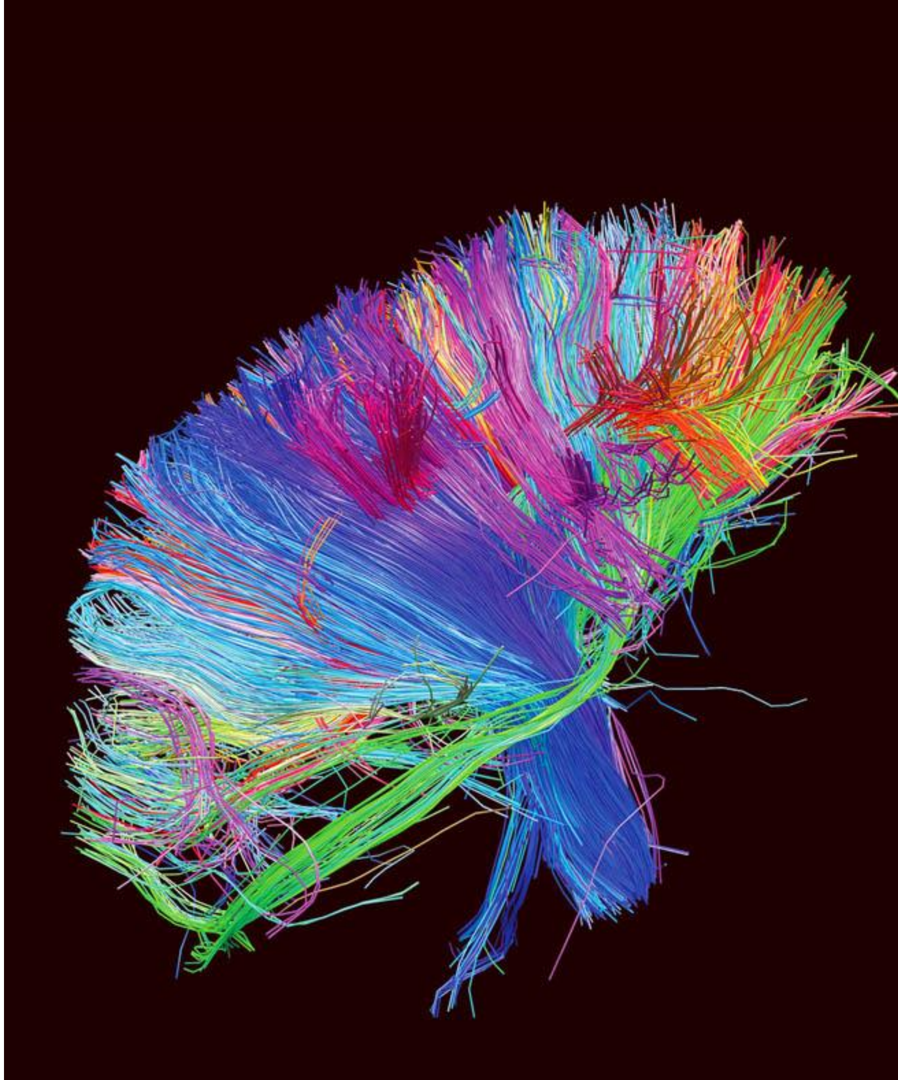
Multi-object recognition in images

Marie BRUNET CARTEAUX



Business Services

Context

Images : mathematical objects

→ can be processed as such

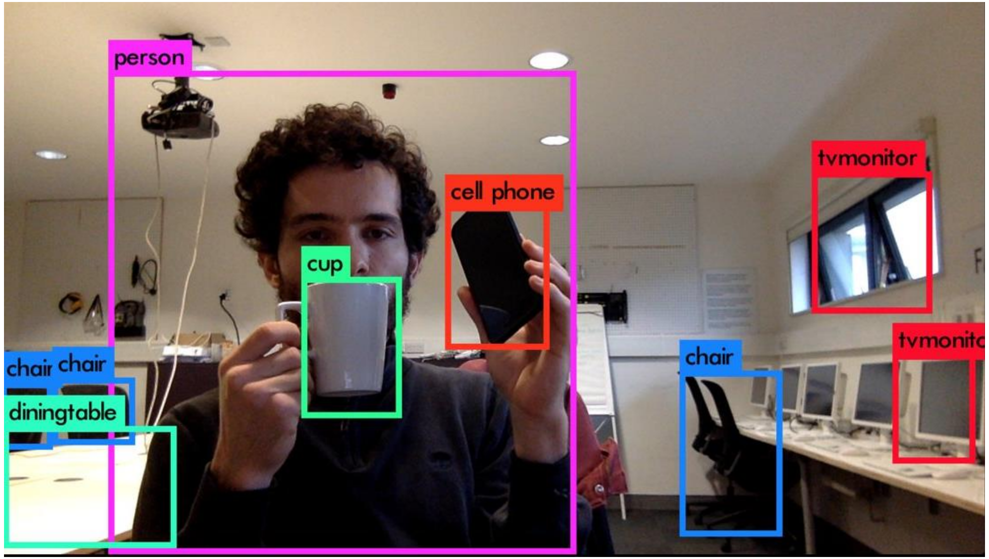
Each pixel p is associated with a frequency f such that f ∈ [0; 255]

Multi-object recognition : result on many mathematical operations processed on images

Context

Multi-object recognition : identify where are interesting features in images

Image processing-based segmentation



Deep learning-based detection

Context

Multi-object recognition opens up a wide field of possibilities

Detect people and abandoned luggage in public transportation

Spot free lots in parkings

Detect free lots in warehouses

Design brief

Goal: detect people and abandoned luggage in public transportation

→ Associated constraints:

- **Light exposure, contrast & camera position**
- **Execution period**
- **Mandated execution latency**
- **Initialization needs**
- **Hardware at disposal**
- **Data anonymization & user protection**

Design brief - Constraints

- Light exposure, contrast & camera position

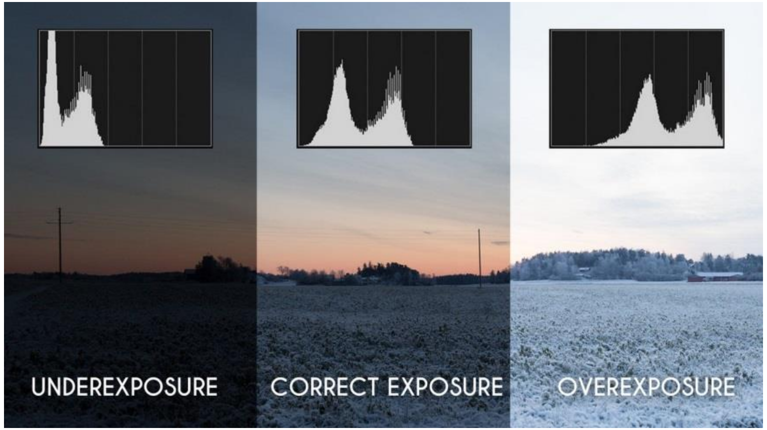
Those can vary depending on the time of the day or the client

- Execution period

The solution shall be as fast as possible

- Mandated execution cadency

The solution shall not waste useless data & resources



Design brief - Constraints

- Initialization needs

There should be none

- Hardware at disposal

No embedded CPU/GPU → remote computing

- Data anonymization & user privacy

Alignment with the GDPR requirements

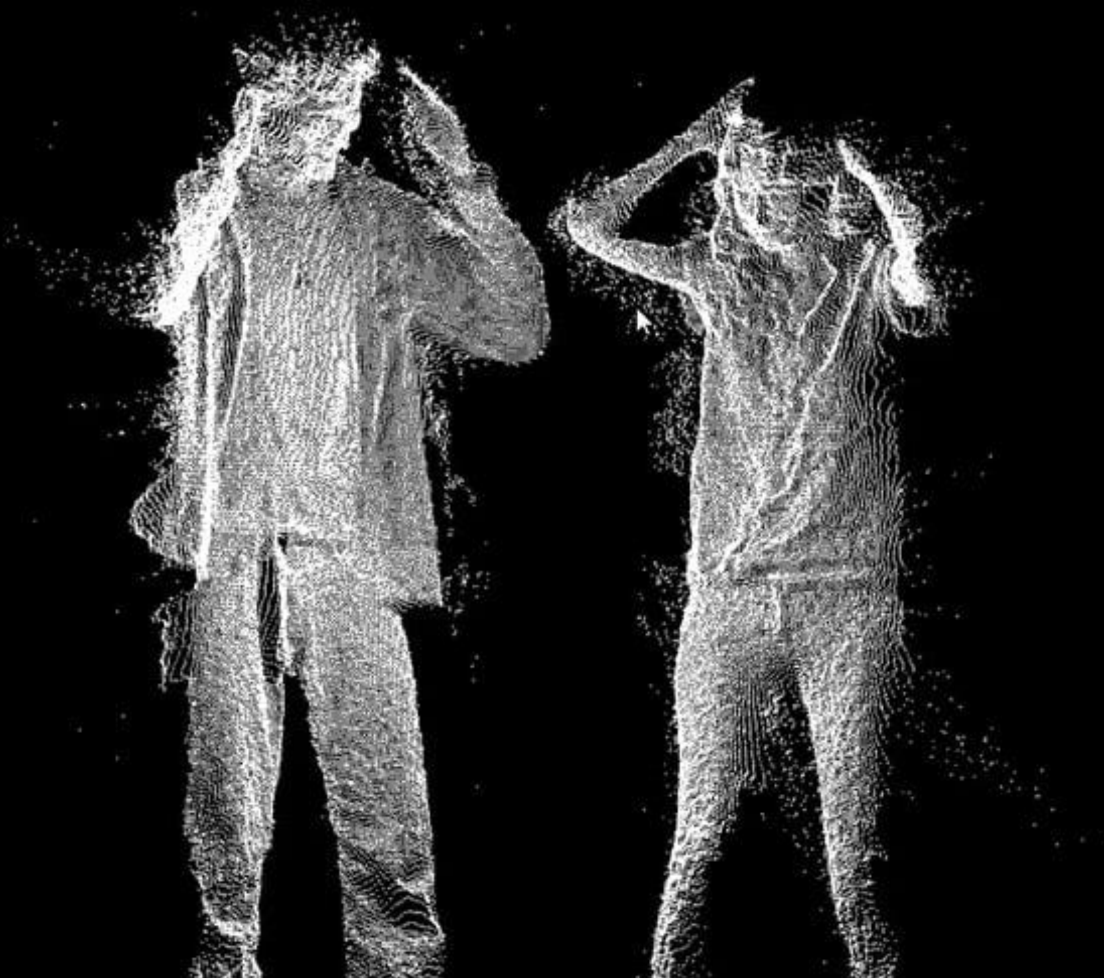
Objective

1. Overview existing techniques in both:
    - Image processing
    - Deep learning

2. Understand the need for auxiliary processing
    - Before launching the algorithm
    - And after, for better accuracy

3. Learn to measure detection accuracy

Image processing



Introduction to image processing

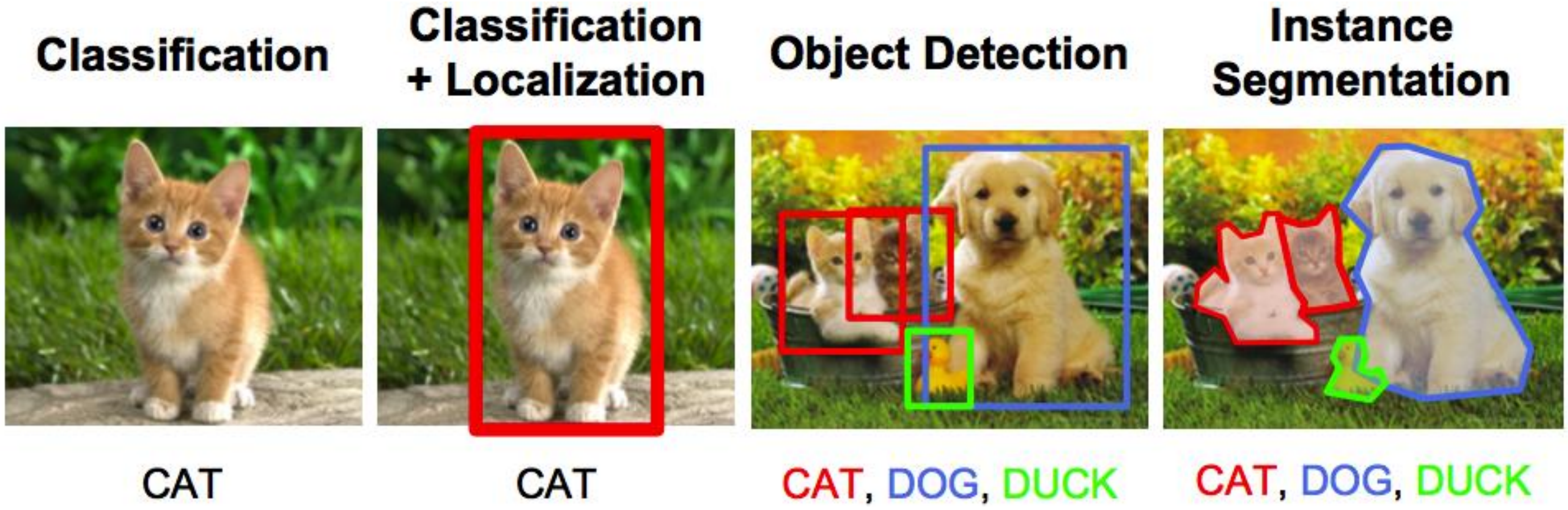
Any method/algorithm aiming to using or modifying images

Strengths of IP:

- Fast
- Easy to implement
- Low hardware requirements

Semantic Instance Segmentation

IP-based techniques perform semantic segmentation

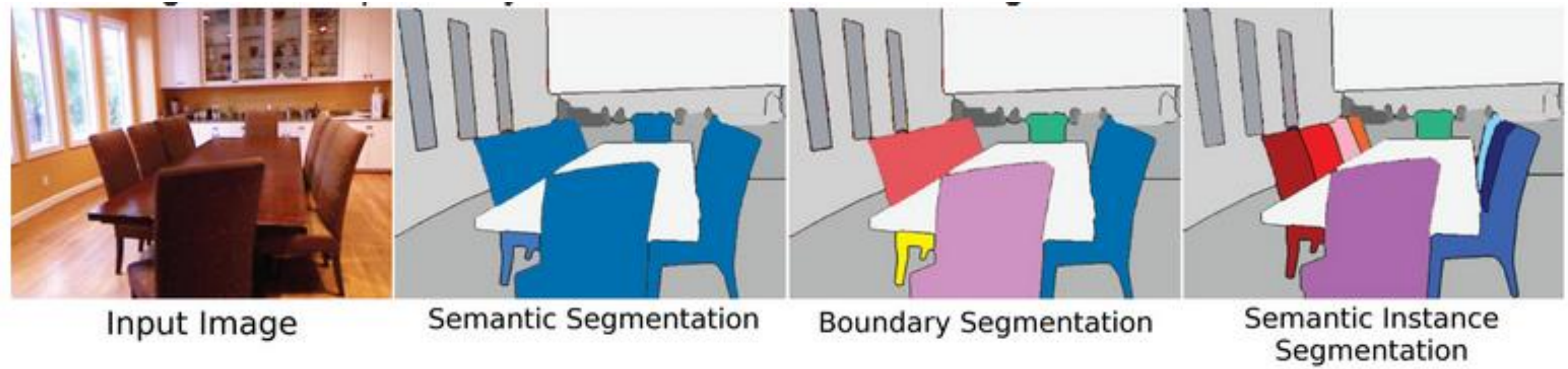


Machine learning-based techniques

IP-based techniques

Semantic Instance Segmentation

IP-based techniques perform semantic segmentation

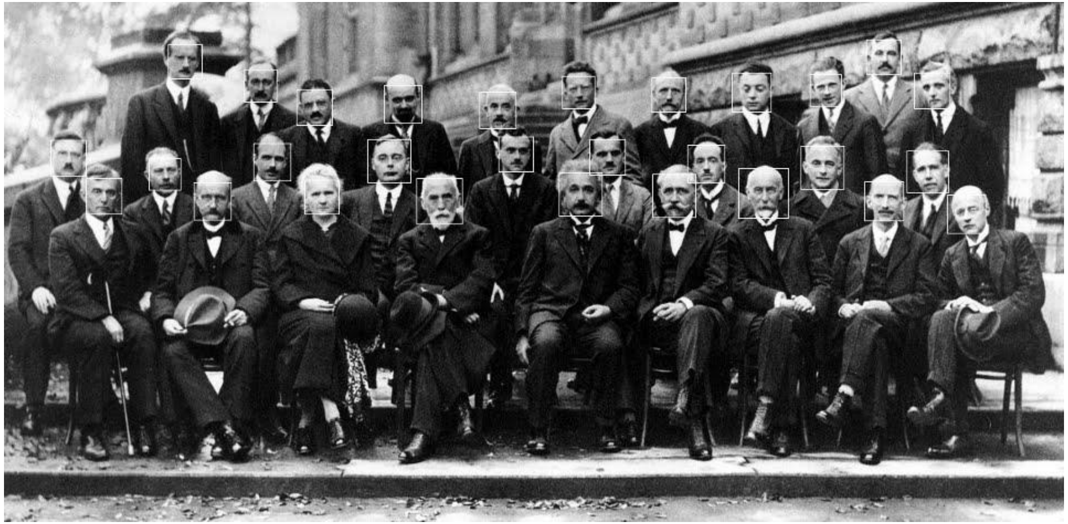


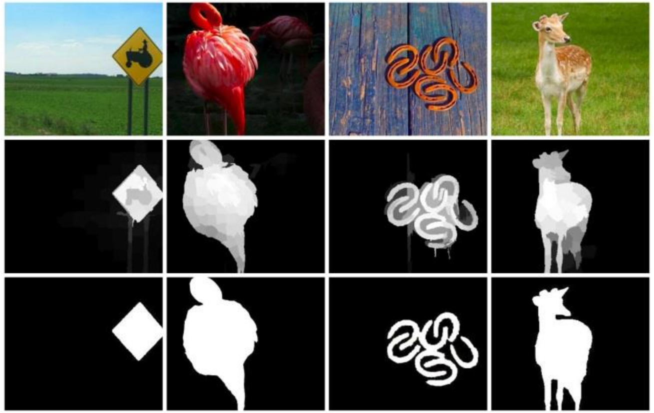
State-of-the-art

Few algorithms ready-to-use

Two famous techniques:

- Viola-Jones →
- Saliency map ↓





Elementary blocks

- Edge detection
- Noise reduction
- Morphological mathematics
- Template matching
- Hough transformation
- Histogram equalization

Edge detection

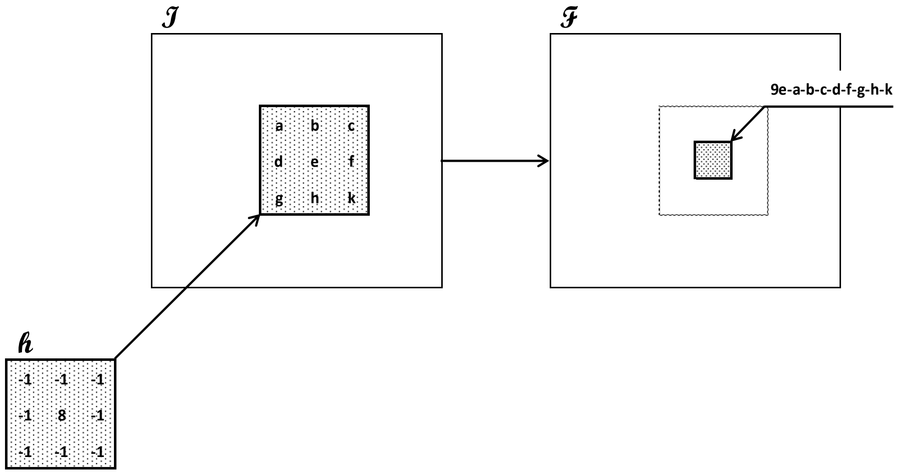
$$Image : \begin{cases} Low\ frequencies: slow\ variations\ (uniform\ areas) \\ High\ frequencies: fast\ variations\ (edges, noise) \end{cases}$$

Contour extraction → high-pass filters

Enhance edges AND noise: must reduce noise before extracting contours

$$F(x,y) = \sum_{a,b} h(a,b) \cdot I(x+a, y+b)$$

$$h = \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$



Edge detection – Canny

> 1986: multi-stage algorithm

Grayscale Input image

Remove noise

Gradient magnitude

Non-maximum suppression

Thresholding

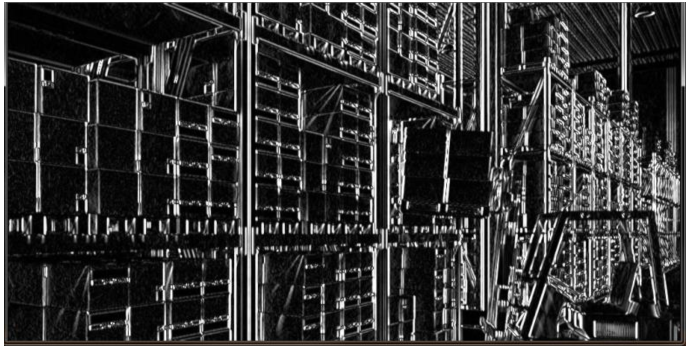
Binary edges map

Edge detection – Sobel

> 1968: computes approximative gradient of the image intensity function

Based on derivatives: give indications about horizontal & vertical changes

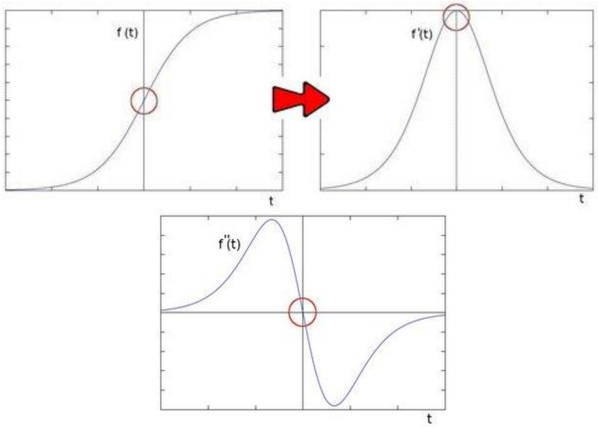
→



Edge detection – Laplace

> XVIIIth century: calculus of second derivatives

Tracks jumps in pixels intensities (⇔ zero in 2nd derivative)



→


Original
Sobel
Canny
Laplace

Other edge detection algorithms (less famous)

- Deriche
- Prewitt
- Robert
- ...

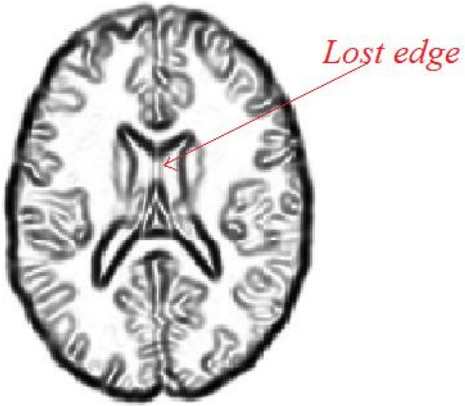
Noise reduction

- Noise has undesired effects thereafter (e.g. false positives edges appearing)

- Many noise-reduction techniques: depend on the use-case & image characteristics

- Smoothing is widely used to reduce noise

Noise image



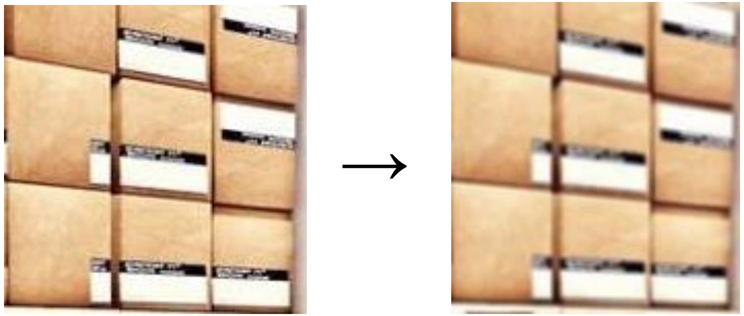
Edge detection

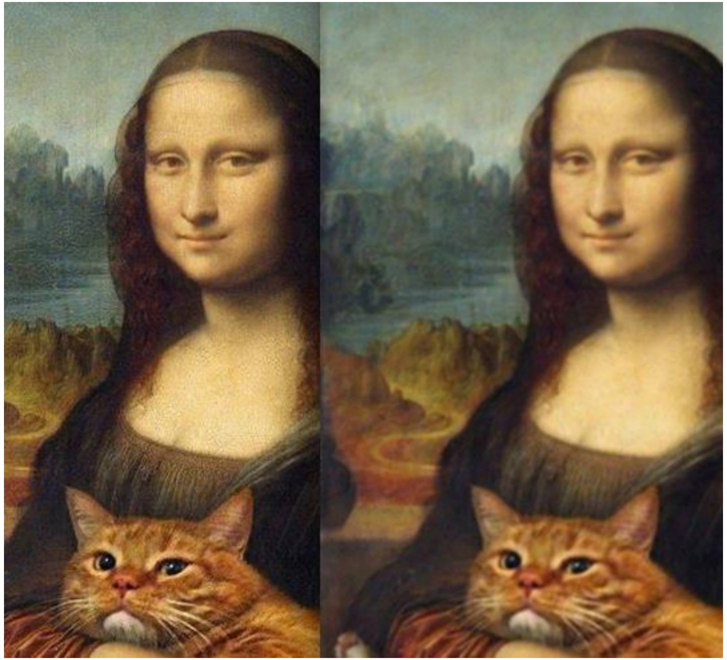
Noise reduction - Blurring

Simplest smoothing algorithm: « normalizing box »

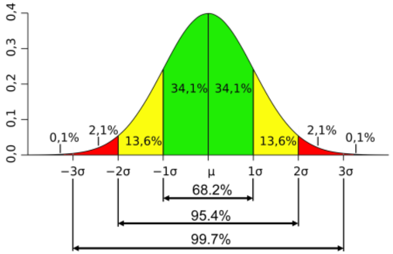
$$h = \frac{1}{3*3} \cdot \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Powerful noise remover but also smoothes edges



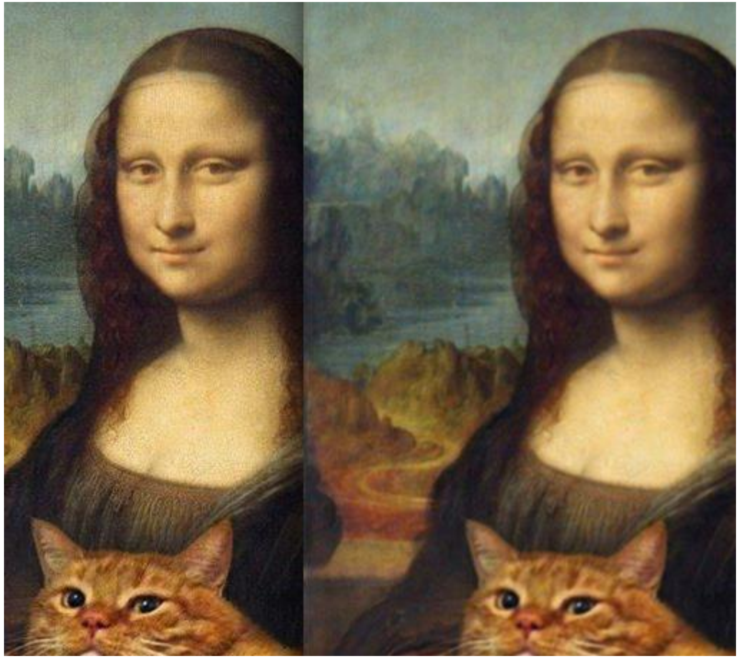


Noise reduction - Gaussian smoothing

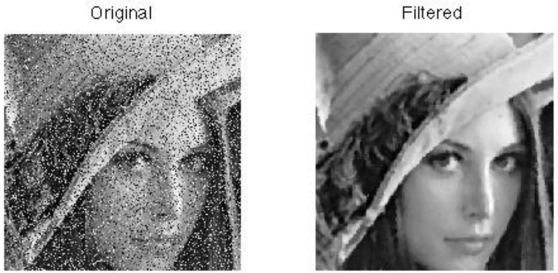


Most useful but rather slow

→

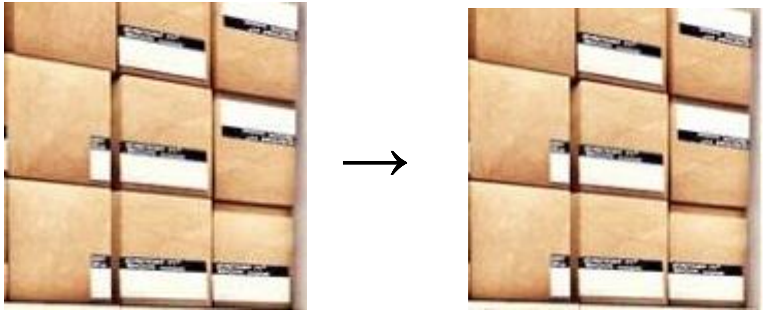


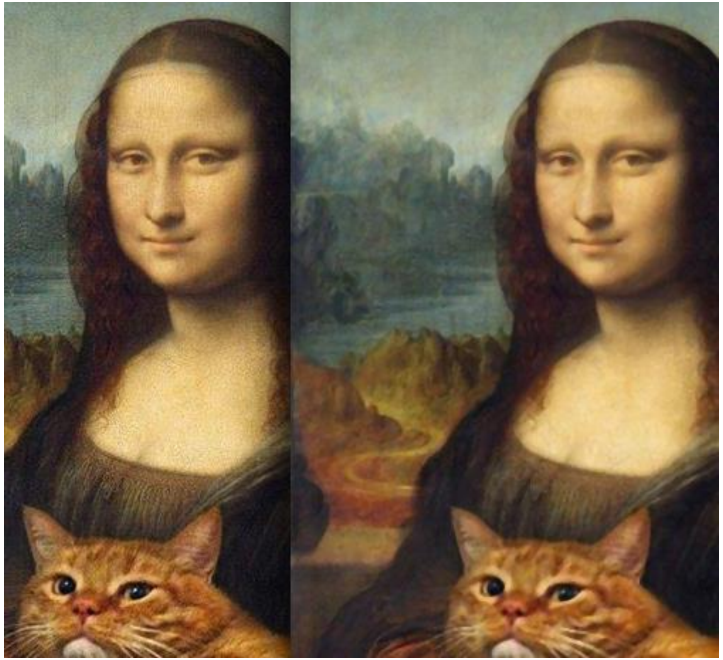
Noise reduction - Median smoothing



Replace each pixel w/ the median of its neighbours

Very powerful on « salt-n-pepper » noise

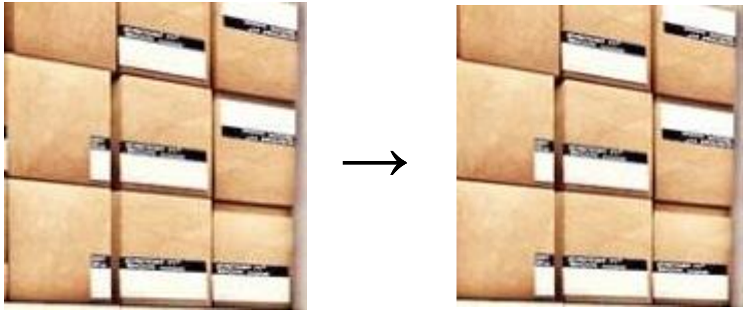


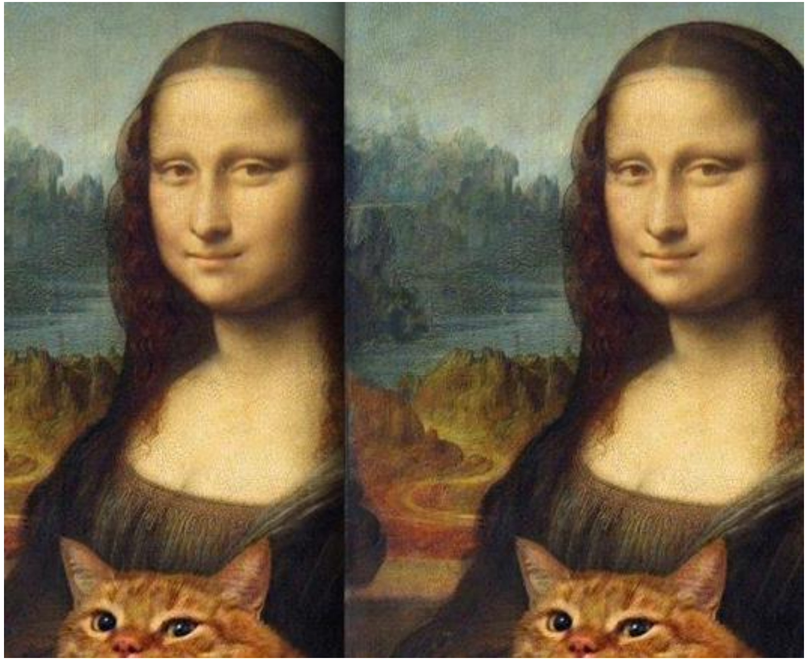


Noise reduction - Bilateral smoothing

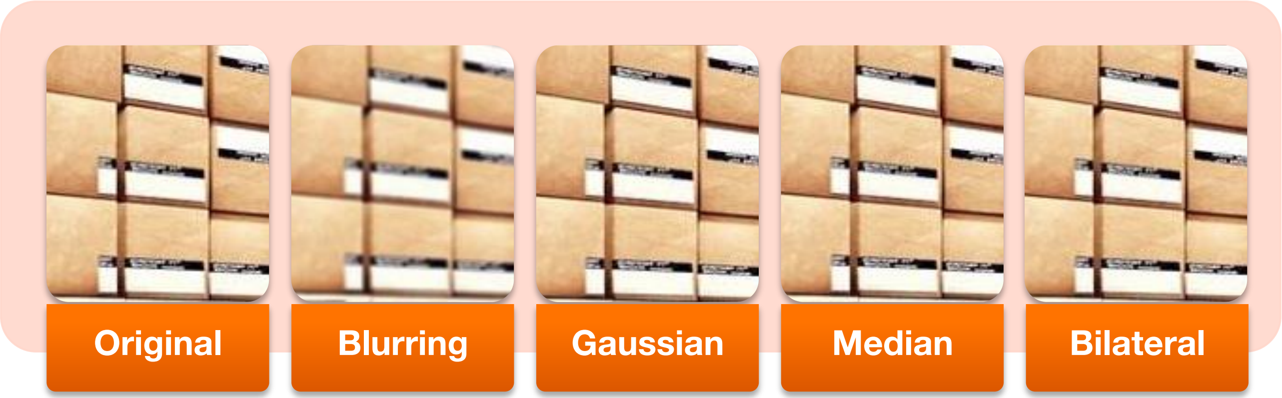
Prevents edges smoothing while reducing noise

Consider weighed neighbors (Gaussian + difference of intensity)





Noise reduction - Performance comparison



Smoothing filter	Execution time (µs)
Blur	1873
Gaussian	1508
Median	1118
Bilateral	251032

Morphological mathematics

- Set of operations based on shapes

- Apply a structuring element to an input image

- Usually used to isolate elements

Morphological mathematics – Dilation & Erosion

Dilation

- Grow image regions based on structuring elements

- Suppress holes & gaps

- Thicken edges

Erosion

- Shrinks image regions (similarly to dilation)

- Thin elements

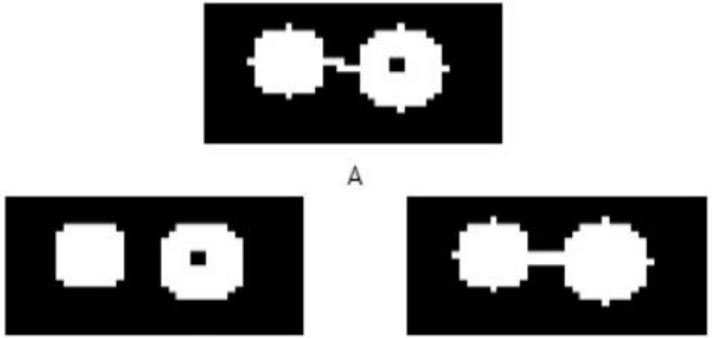
- Remove small components & noise

Morphological mathematics – Opening & Closing

Opening: Erosion then dilation

Closing: Dilation then Erosion

Useful for getting rid of small artefacts or open/close holes in shapes

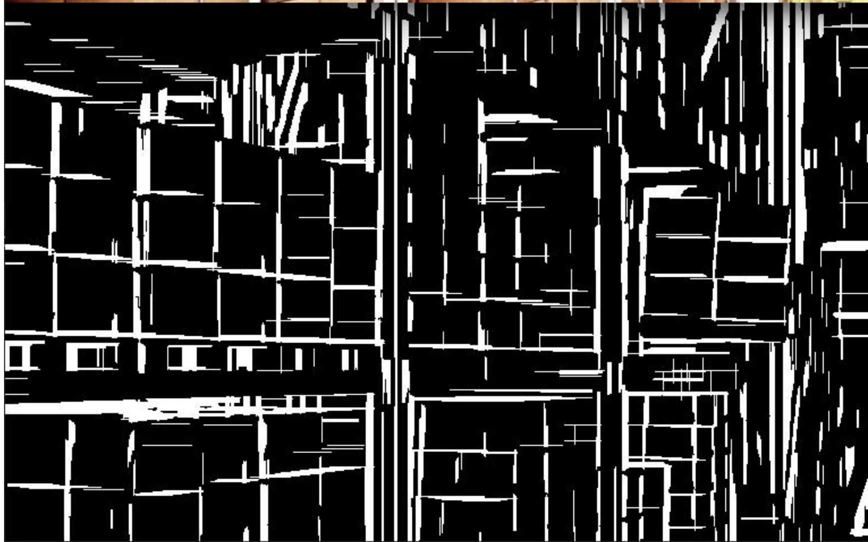


opening of A
→removal of small protrusions, thin connections, ...

closing of A
→ removal of holes

Morphological mathematics – Lines extraction

Use erosion & dilation operations with line-shaped structuring elements



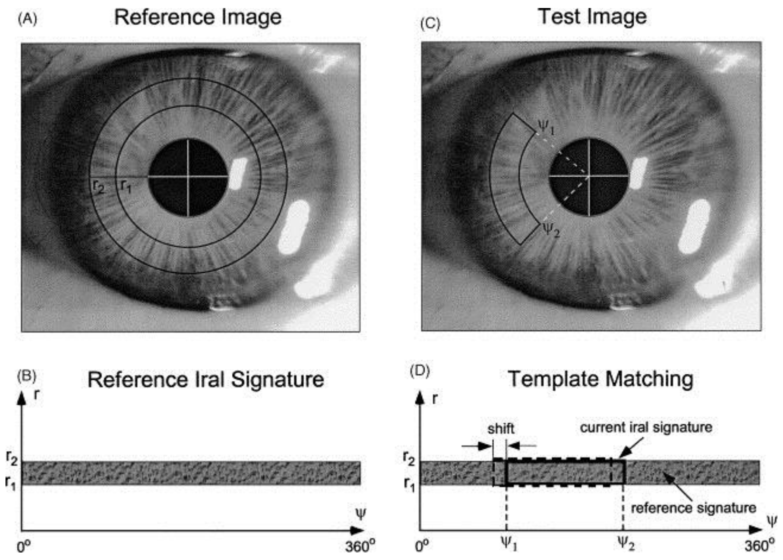
Template matching

Find areas of an image that match a template image

Needs: input image + patch
→ patch is moved on the image (sliding)

∀ location, metrics say if the patch is similar to the image area

Actual location: those with the highest matching probability



Hough transformation

Widely-used method for detecting geometrical patterns in images

Switch from Euclidian- to Polar coordinates frame

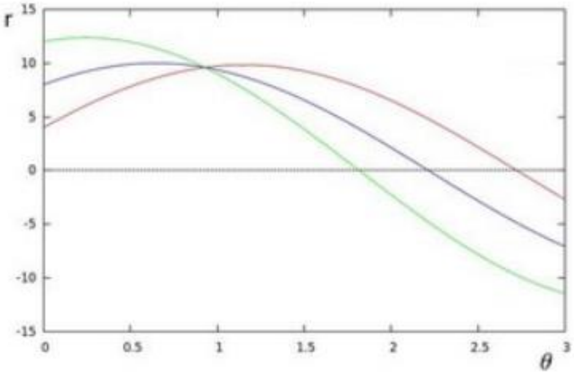
Lines correspondences appear in Polar frame

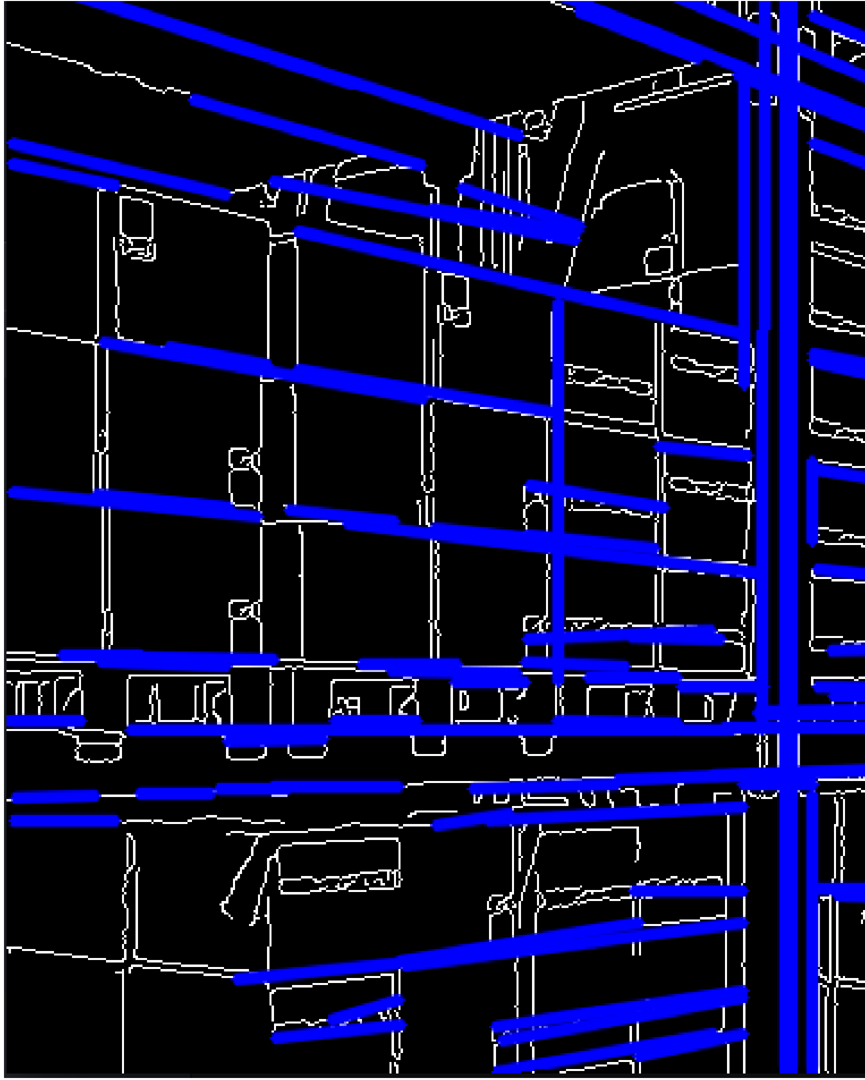
Hough transformation

Every point (x_0,y_0) is expressed as $r_\theta = x_0 cos\theta + y_0 sin\theta$

→ Gives a sinusoid called family of lines

2 families of lines intersect: these 2 points belong to the same line in the Euclidian frame



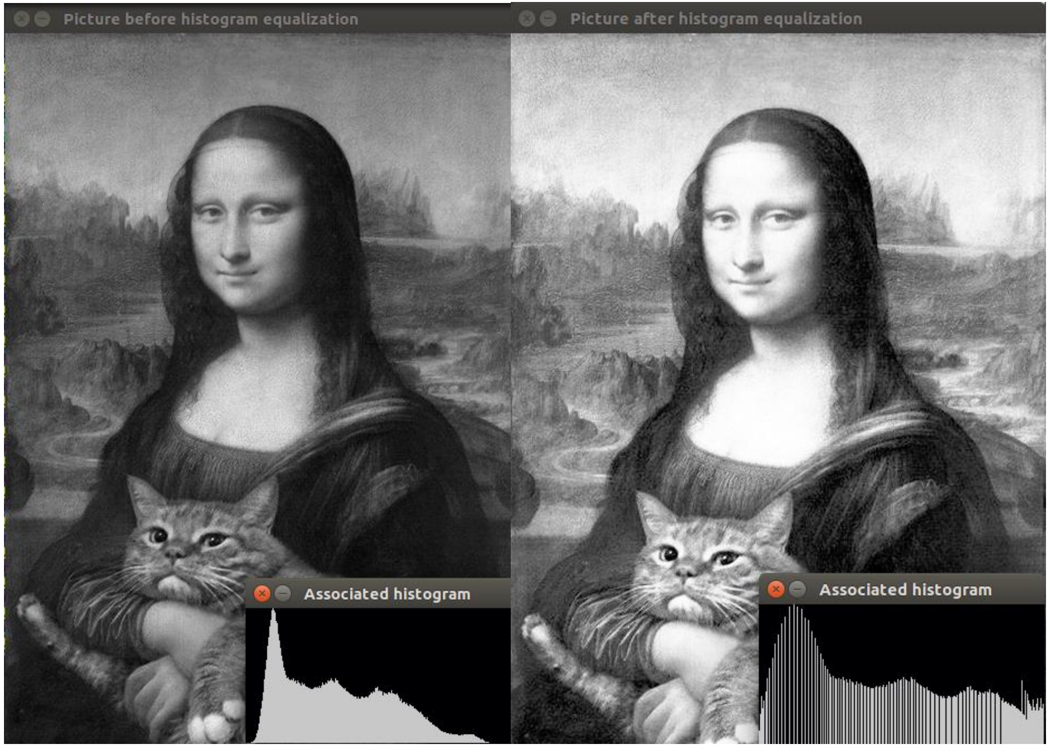


Histogram equalization

Histogram: graphical representation of intensity distribution in an image

Equalization: broaden the histogram

Better contrast



Open-source frameworks

ImageMagick

OpenCV

Boost.GIL
Generic Image Library

Simd

libvips
A fast image processing library with low memory needs.

And many more …

In a nutshell

IP-based technique: a mix of all of the abovementioned

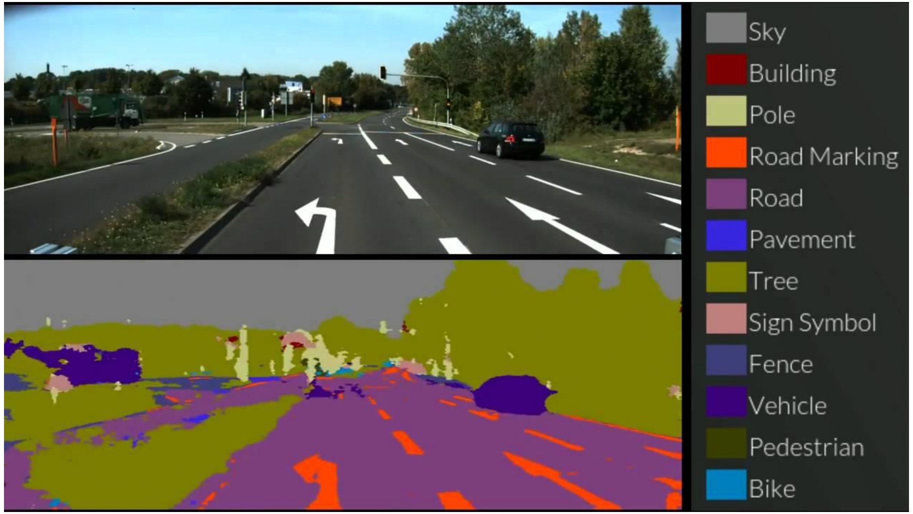
Drawbacks of IP-based detection:

- **Fine tuning of parameters is needed**
- **Hard to build a generic algorithm**
- **Difficult to qualify « good performance »**

Deep learning

Power of machine learning

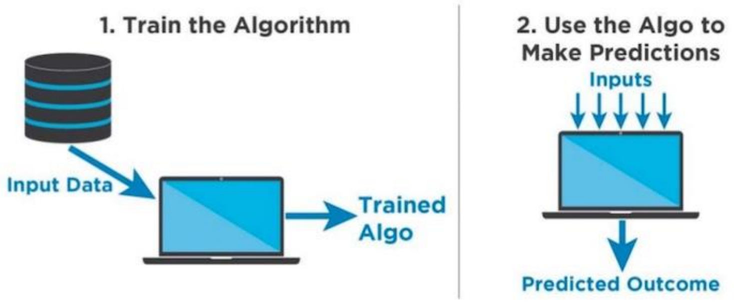
Instant translation



Self-driven cars

Gaming IAs

Principle of machine learning



1. Feed the algo with labeled data

→ It will find a causal relationship btw input & result

$$Y = f(X) + \epsilon$$

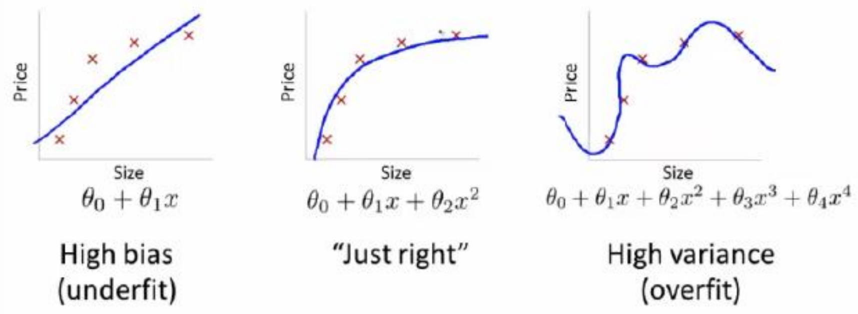
X (input) = years of higher education
Y (output) = annual income
f = function describing the relationship between X and Y
ε (epsilon) = random error term (positive or negative) with mean zero

2. Give the algo new, unlabeled data

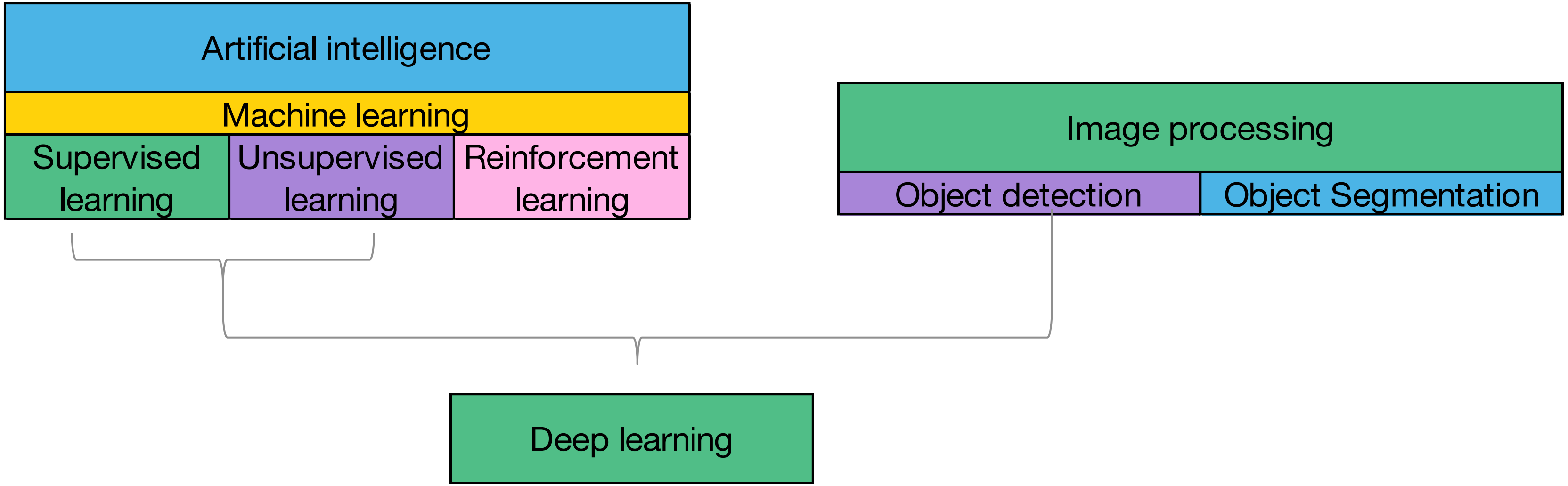
→ It will compute the causal relationship's result

Caution: avoid high bias/variance

→ It causes under/overfit of the algorithm



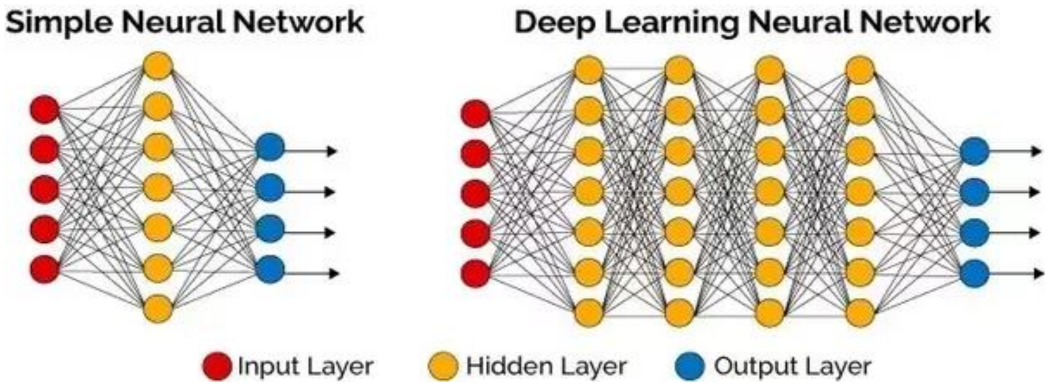
Introduction to deep learning

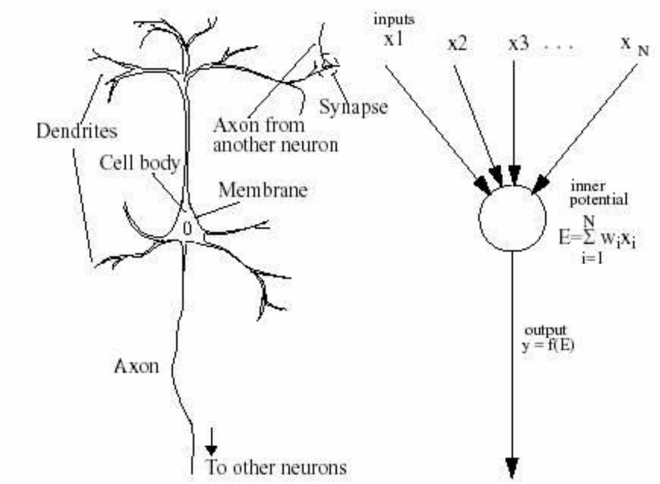


Why deep learning?

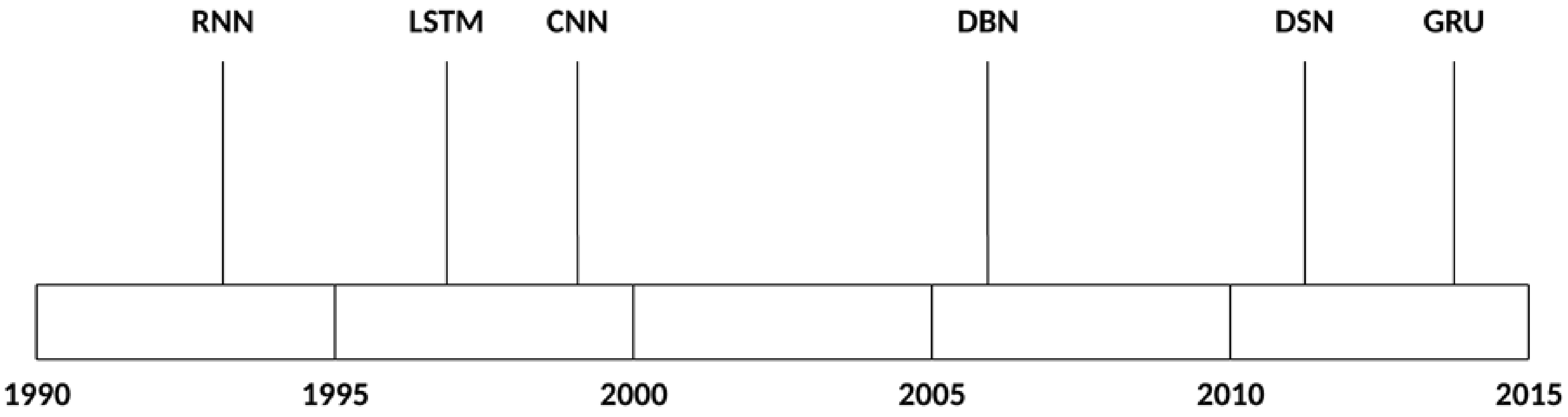
Machine learning is not enough for images
→ Pixels: heavy vectors w/ huge amount of data
→ Feature recognition: abstract process

Need layers of abstraction





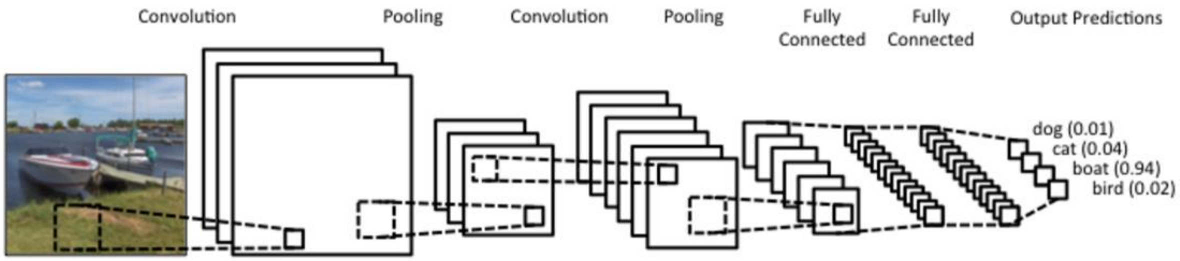
DL models for image processing

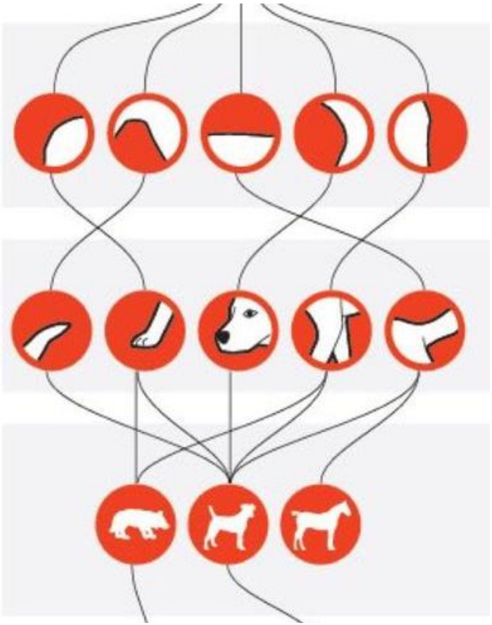


2 « dedicated » to image processing:

- CNN
- DBN

Convolutional Neural Networks (CNNs)





Inspired by animal visual cortex

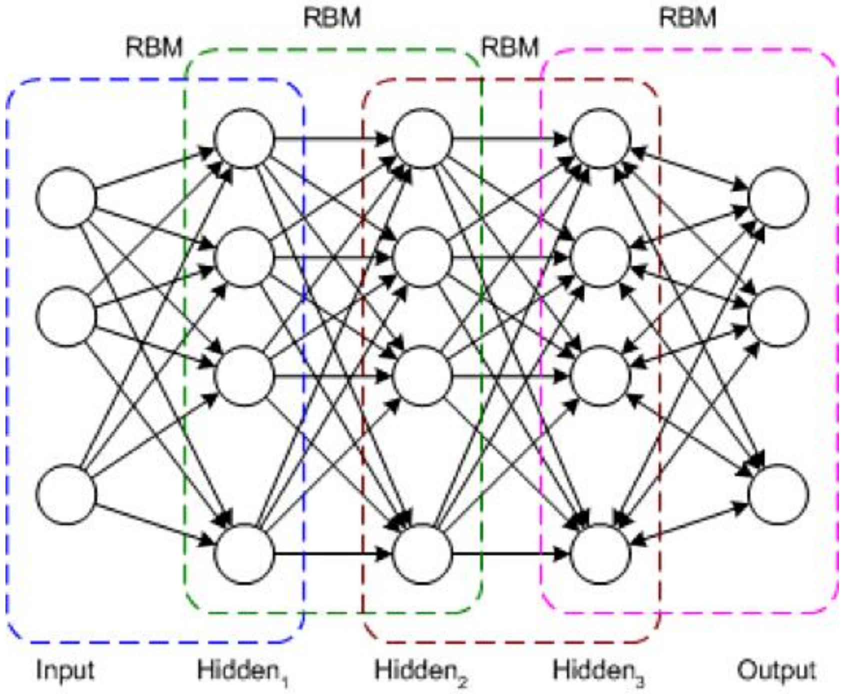
Early layers recognize fine features (e.g. edges)
Later layers recombine them into high-level attributes
Final step: classification

Deep Belief Networks (DBNs)

Stack of RBMs

2 steps: unsupervised pretraining & supervised fine-tuning

Full net training using gradient descent/back-propagation



Open-source frameworks

TensorFlow

Caffe

Caffe2

DL4J

And many more …

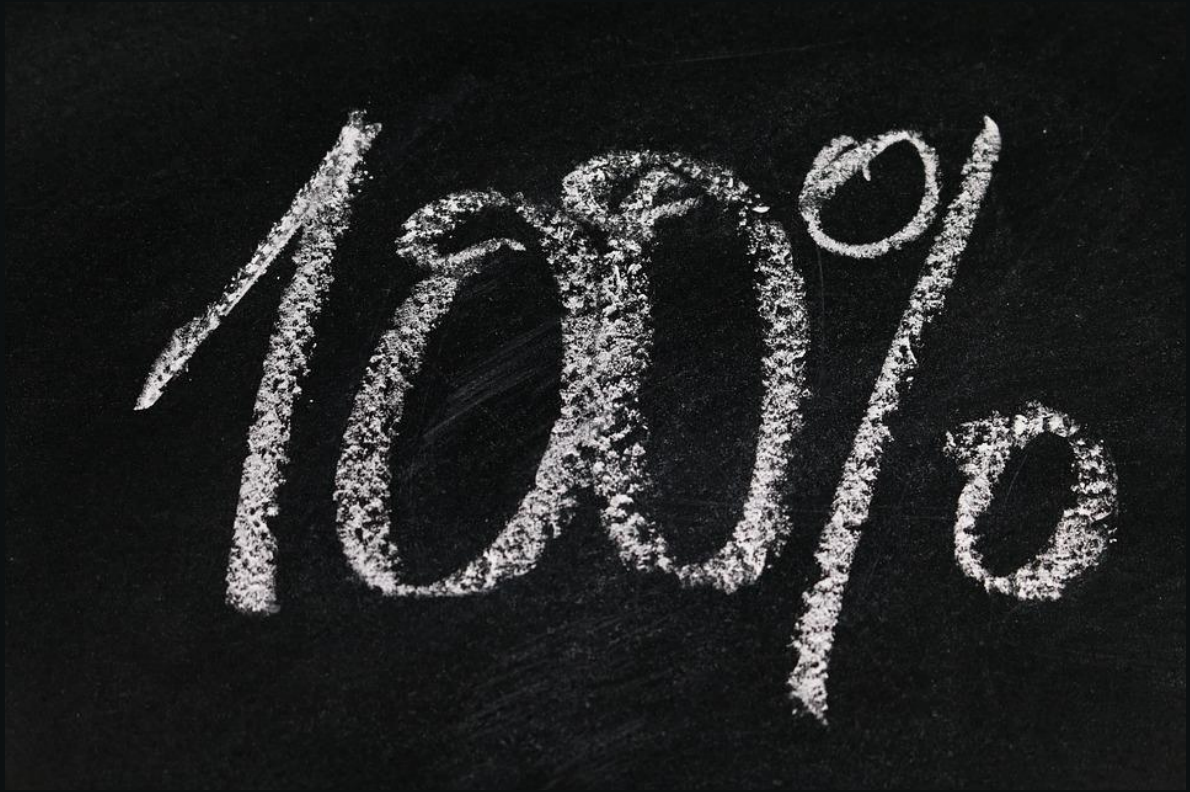
In a nutshell

Deep learning is very powerful & fast once trained

Training requires powerful hardware (GPUs, cloud computing …) and time

Additional image processing is needed to normalize input images

Auxiliary processing



Pre- & postprocessing

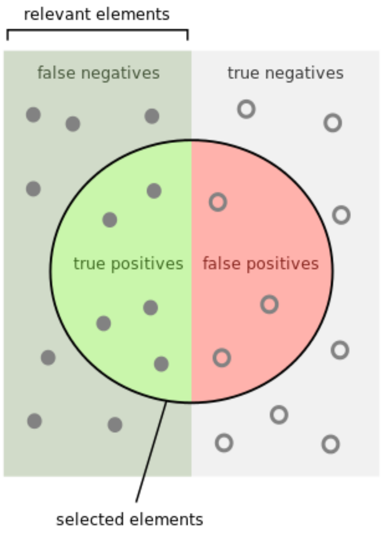
For both techniques, images shall be normalized before

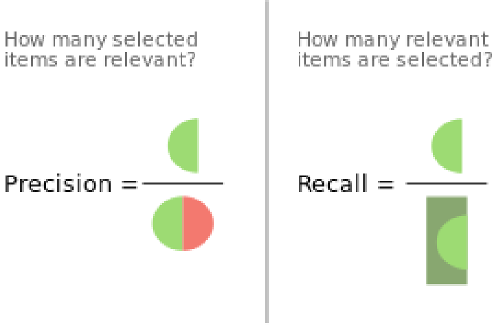
Preprocessing includes:

- Up/Downsampling
- Resolution change
- Noise reduction
- Light/contrast equalization

Postprocessing smoothes the obtained segmentation

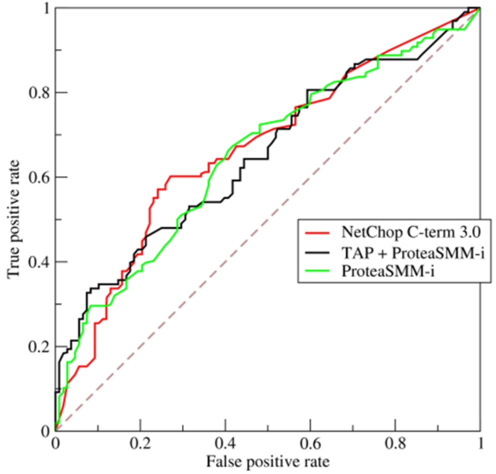
Performance metrics





Assess the segmentation/detection accuracy

- **Precision-Recall**
- **F-Measure (DSC)**
- **Receiver Operating Characteristics**
- **AUC**
- **Mean Absolute Error**



Shall be combined to offer an objective accuracy measure

Conclusion

Image processing	Deep learning
Complex parameters tuning	Few variables to adjust
Few resources needed	Uncertain hardware requirements
Varying computational time	Ultrafast
Uncertain segmentation accuracy	Reliable object recognition

- Auxiliary processing is needed by both technologies
- One shall be careful when trying to assess an algorithm's performance

Thank you

marie.brunet@orange.com

orange™ Business Services