Riconoscimento di oggetti

17634 | VISIONE ARTIFICIALE

Cosa si intende per riconoscimento?

Gatto

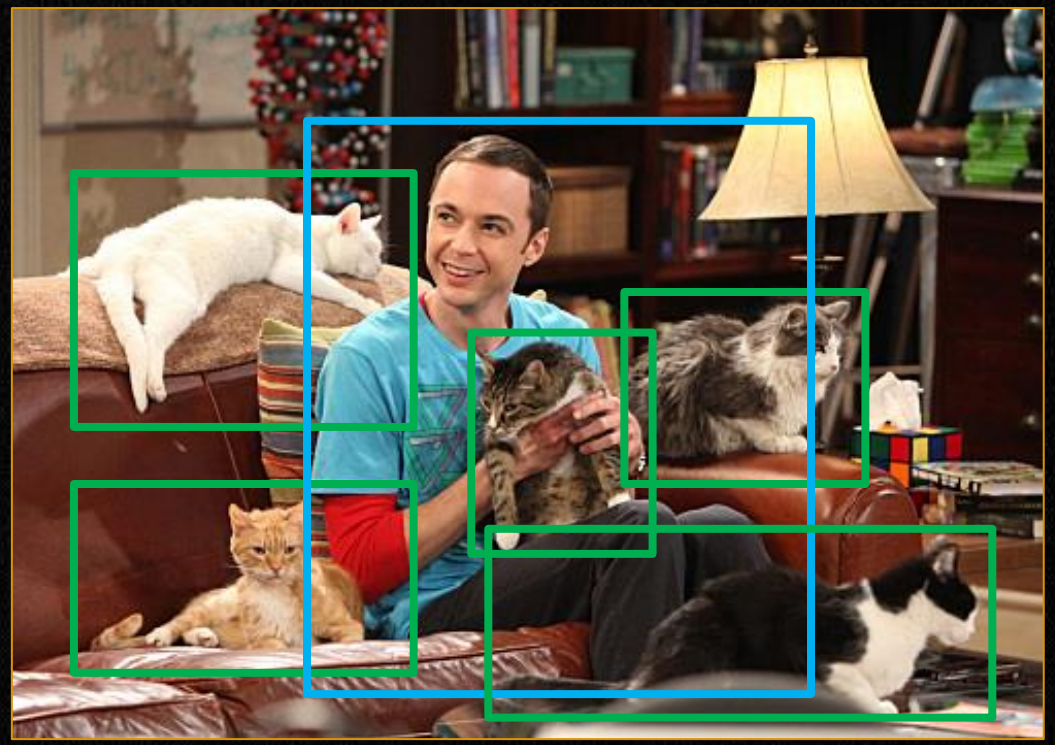
Classificazione

Cosa si intende per riconoscimento?

Gatto

Classificazione e localizzazione

Cosa si intende per riconoscimento?



Gatto, Persona

Object detection

Cosa si intende per riconoscimento?

Gatto, Persona

Instance segmentation

Cosa si intende per riconoscimento?

Una persona siede su un divano con cinque gatti

Descrizione della scena

Confronto diretto fra immagini

- Il confronto pixel-a-pixel di due immagini raramente può essere di qualche utilità, per molteplici ragioni, fra cui:
  - Differenze di traslazione, rotazione, scala e prospettiva
  - Deformazione e variabilità degli oggetti
  - Cambiamenti di illuminazione
  - Presenza di rumore nelle immagini e utilizzo di tecniche di acquisizione diverse

A

B

C

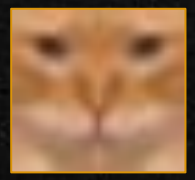
A, B e C hanno le stesse dimensioni e numero di canali (220x110x3):
A è più simile a B o a C?

```
print('||A-B|| =', round(np.sqrt(((A-B)**2).sum())))
print('||A-C|| =', round(np.sqrt(((A-C)**2).sum())))
```

```
||A-B|| = 2857.0
||A-C|| = 2795.0
```

Template matching

- Invece di confrontare direttamente due immagini, si definiscono uno più pattern modello (template): piccole immagini che vengono poi "cercate" all'interno dell'immagine, misurandone il grado di "somiglianza" (matching) in tutte le possibili posizioni.
  - La tecnica più semplice consiste nel confrontare i pixel del template con quelli dell'immagine, anche se in determinate applicazioni si possono ottenere migliori risultati confrontando altre caratteristiche, come i bordi o l'orientazione del gradiente.



Visione Artificiale - Raffaele Cappelli - raffaele.cappelli@unibo.it

Distanza fra immagine e template: SQDIFF

- Somma delle differenze al quadrato (in OpenCV CV_TM_SQDIFF):

$$\mathbf{R}[y,x] = \sum_{i=0}^{h-1}\sum_{j=0}^{w-1}(\mathbf{T}[i,j] - \mathbf{I}[y+i,x+j])^2$$

- Esempio (sul canale luminosità):

T

I

R

Distanza fra immagine e template: SQDIFF normalizzata

- Somma delle differenze al quadrato normalizzata (in OpenCV CV_TM_SQDIFF_NORMED):

$$\mathbf{R}[y,x] = \frac{\sum_{i=0}^{h-1}\sum_{j=0}^{w-1}(\mathbf{T}[i,j] - \mathbf{I}[y+i,x+j])^2}{\sqrt{\left(\sum_{i=0}^{h-1}\sum_{j=0}^{w-1}(\mathbf{T}[i,j])^2\right)\cdot\left(\sum_{i=0}^{h-1}\sum_{j=0}^{w-1}(\mathbf{I}[y+i,x+j])^2\right)}}$$

- Esempio (sul canale luminosità):

T

I

R

Distanza fra immagine e template: Correlazione

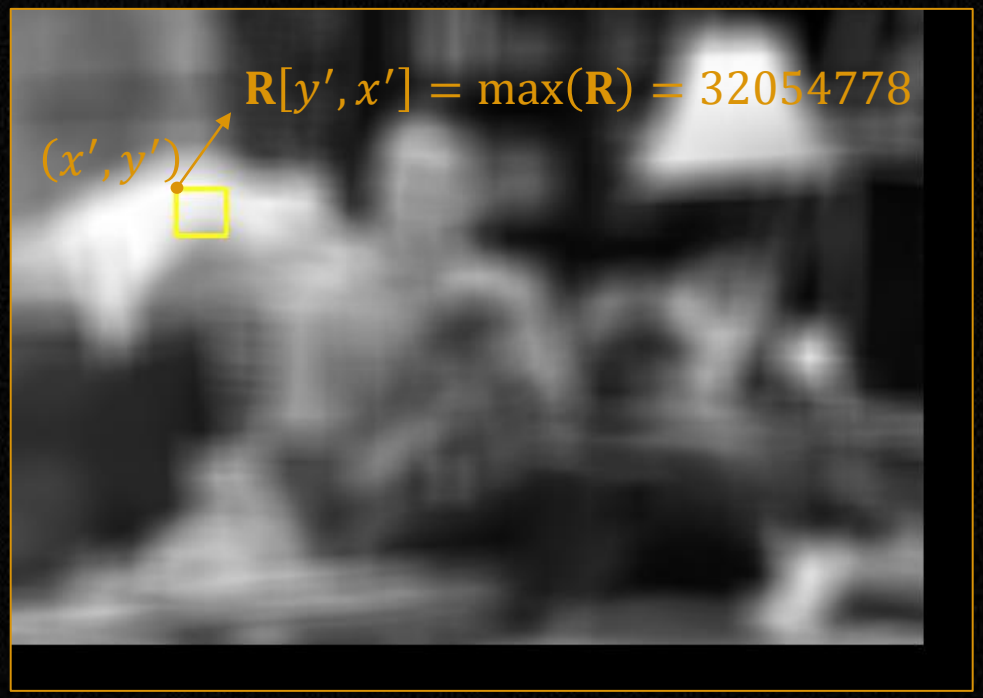
- Correlazione (in OpenCV CV_TM_CCORR):

$$\mathbf{R}[y,x] = \sum_{i=0}^{h-1}\sum_{j=0}^{w-1}(\mathbf{T}[i,j]\cdot\mathbf{I}[y+i,x+j])$$

- Esempio (sul canale luminosità):

T

I



R

Distanza fra immagine e template: Correlazione normalizzata

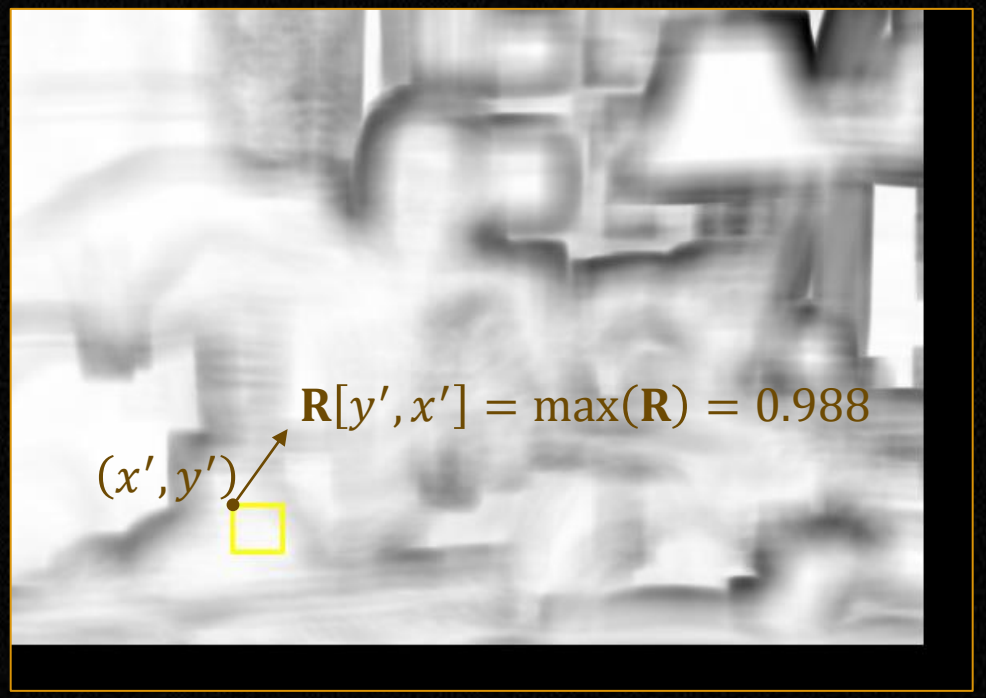
- Correlazione normalizzata (in OpenCV CV_TM_CCORR_NORMED):

$$\mathbf{R}[y,x] = \frac{\sum_{i=0}^{h-1}\sum_{j=0}^{w-1}(\mathbf{T}[i,j]\cdot\mathbf{I}[y+i,x+j])}{\sqrt{\left(\sum_{i=0}^{h-1}\sum_{j=0}^{w-1}(\mathbf{T}[i,j])^2\right)\cdot\left(\sum_{i=0}^{h-1}\sum_{j=0}^{w-1}(\mathbf{I}[y+i,x+j])^2\right)}}$$

- Esempio (sul canale luminosità):

T

I



R

Distanza fra immagine e template: Coefficiente di correlazione

- Coefficiente di correlazione (in OpenCV CV_TM_CCOEFF):

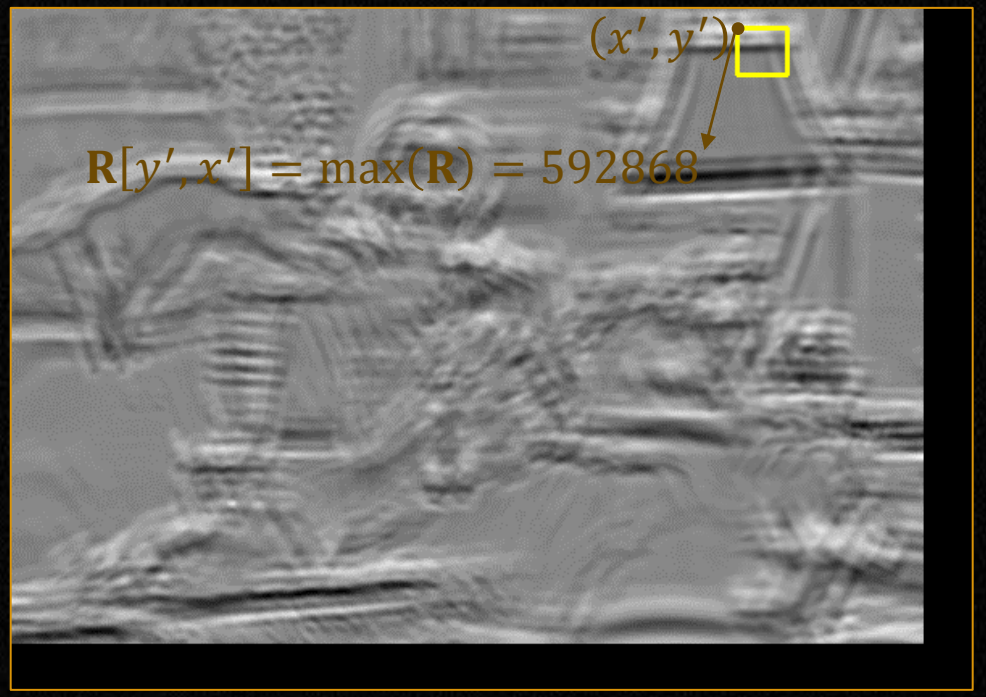
$$\mathbf{R}[y,x] = \sum_{i=0}^{h-1}\sum_{j=0}^{w-1}\left(\mathbf{T}'[i,j]\cdot\mathbf{I}'_{x,y}[y+i,x+j]\right)$$

con $\mathbf{T}'[i,j] = \mathbf{T}[i,j] - \frac{\sum_{i'=0}^{h-1}\sum_{j'=0}^{w-1}\mathbf{T}[i',j']}{w\cdot h}$ e $\mathbf{I}'[i,j] = \mathrm{I}[i,j] - \frac{\sum_{i'=0}^{h-1}\sum_{j'=0}^{w-1}\mathbf{I}[y+i',x+j']}{w\cdot h}$

- Esempio (sul canale luminosità):

T

I



R

Distanza fra immagine e template: CCOEFF normalizzato

- Coefficiente di correlazione normalizzato (in OpenCV CV_TM_CCOEFF_NORMED):

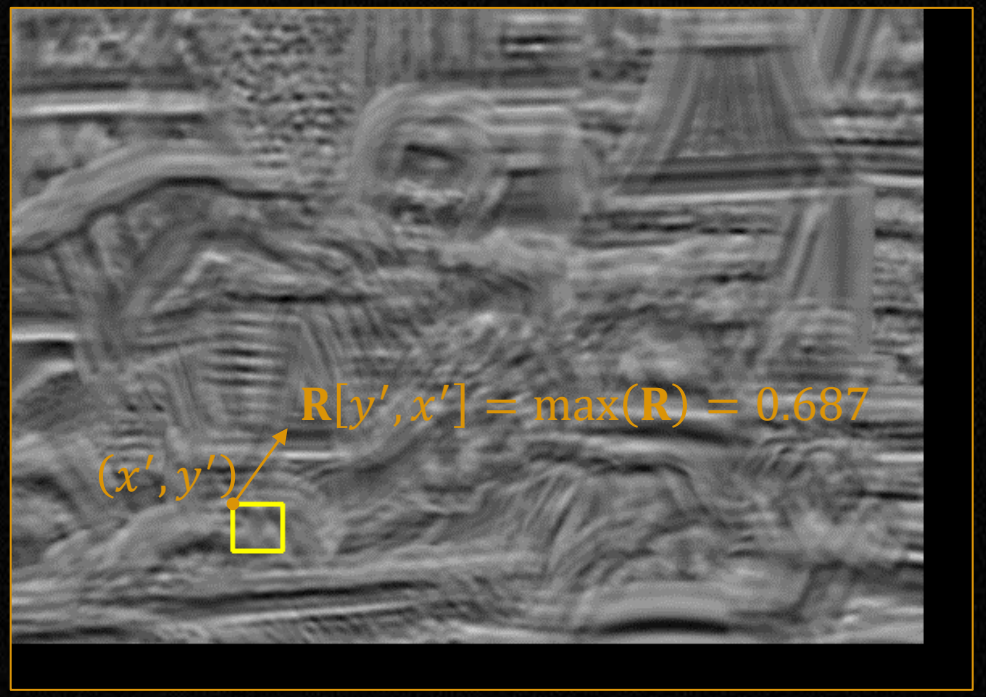
$$\mathbf{R}[y,x] = \frac{\sum_{i=0}^{h-1}\sum_{j=0}^{w-1}\left(\mathbf{T}'[i,j]\cdot\mathbf{I}'_{x,y}[y+i,x+j]\right)}{\sqrt{\left(\sum_{i=0}^{h-1}\sum_{j=0}^{w-1}(\mathbf{T}'[i,j])^2\right)\cdot\left(\sum_{i=0}^{h-1}\sum_{j=0}^{w-1}\left(\mathbf{I}'_{x,y}[y+i,x+j]\right)^2\right)}}$$

con $\mathbf{T}'[i,j] = \mathbf{T}[i,j] - \frac{\sum_{i'=0}^{h-1}\sum_{j'=0}^{w-1}\mathbf{T}[i',j']}{w\cdot h}$ e $\mathbf{I}'[i,j] = \mathrm{I}[i,j] - \frac{\sum_{i'=0}^{h-1}\sum_{j'=0}^{w-1}\mathbf{I}[y+i',x+j']}{w\cdot h}$

- Esempio (sul canale luminosità):

T

I



R

Template matching in OpenCV

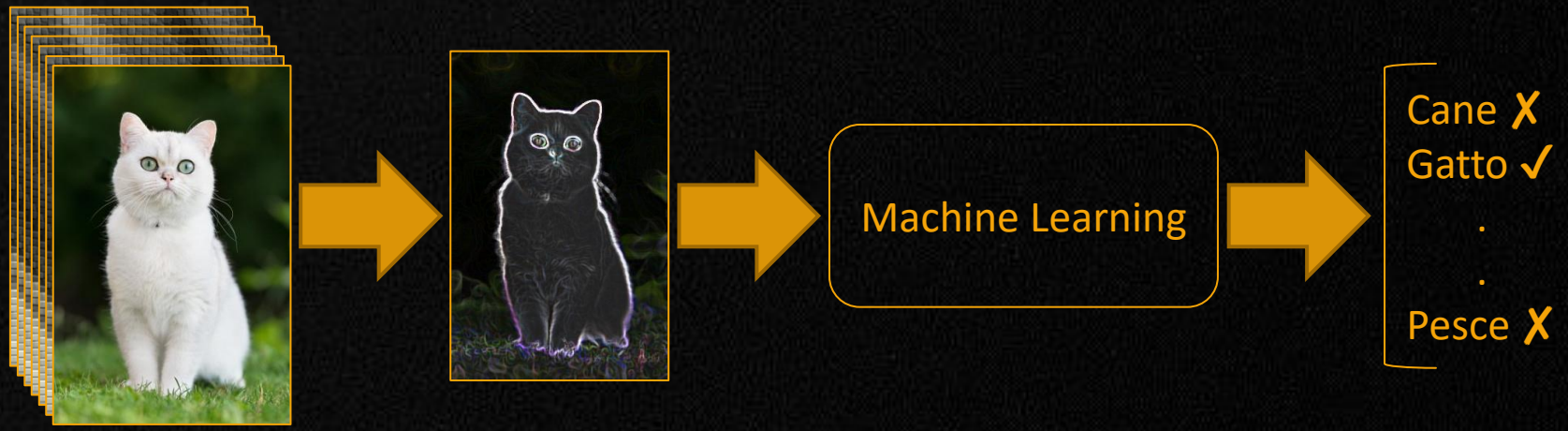

```
I = cv.imread('esempi/sheldon.jpg', cv.IMREAD_GRAYSCALE)
T = cv.imread('esempi/cat-template.png', cv.IMREAD_GRAYSCALE)
h, w = T.shape
metodi = ['cv.TM_SQDIFF', 'cv.TM_SQDIFF_NORMED',
          'cv.TM_CCORR', 'cv.TM_CCORR_NORMED',
          'cv.TM_CCOEFF', 'cv.TM_CCOEFF_NORMED']

@interact(metodo=metodi)
def test_templateMatching(metodo):
    metodo = eval(metodo)
    # Confronta T con tutte le posizioni (x,y) di I, risultato in R
    R = cv.matchTemplate(I, T, metodo)
    # Cerca il minimo e il massimo di R
    r_min, r_max, pos_min, pos_max = cv.minMaxLoc(R)
    if metodo in [cv.TM_SQDIFF, cv.TM_SQDIFF_NORMED]:
        pos, val = pos_min, r_min
    else:
        pos, val = pos_max, r_max
    # Disegna un rettangolo nella miglior posizione trovata
    res = cv.rectangle(I.copy(), pos, (pos[0]+w, pos[1]+h), (0,255,255), 2)
```

Estrazione caratteristiche (feature) e riconoscimento

- Un modo per ridurre la quantità di informazioni di un'immagine, rappresentando le parti più interessanti sotto forma di vettori numerici compatti.
- Una volta che le feature sono state estratte, possono essere utilizzate per individuare e riconoscere oggetti con varie tecniche.

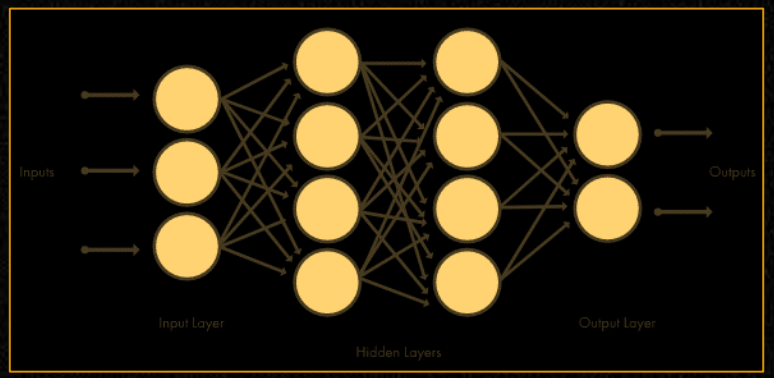
Machine learning

- Un sistema di machine learning (apprendimento automatico) durante la fase di «training» (addestramento), apprende a partire da una serie esempi.
- Successivamente, in fase di inference, è in grado di generalizzare e gestire anche nuovi dati nello stesso dominio applicativo.



Deep learning

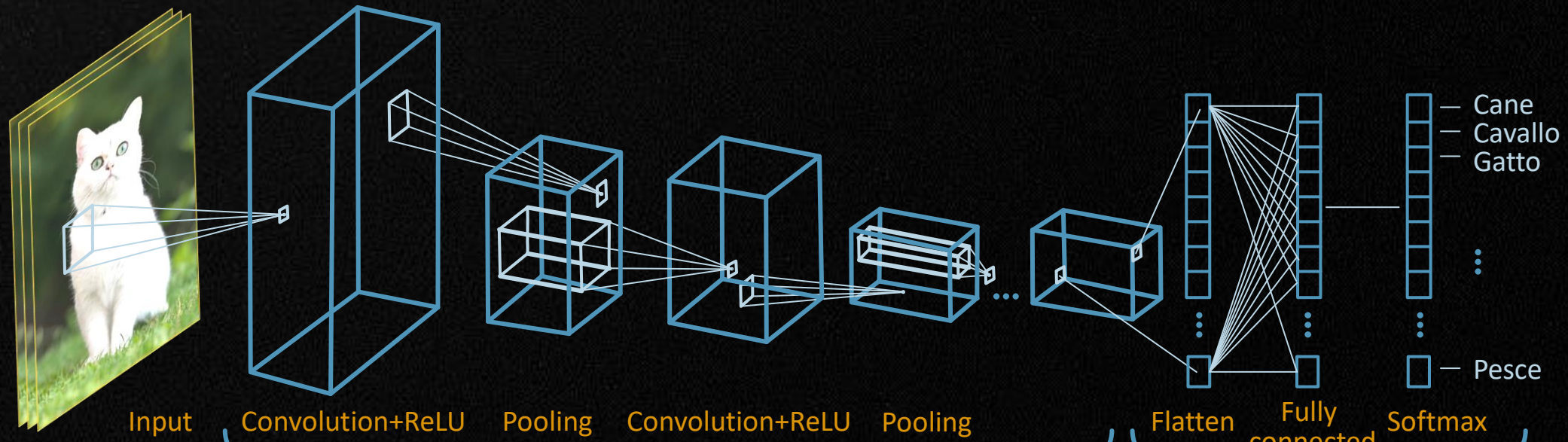
- Una tiplogia di machine learning in cui *sia la scelta delle feature che la classificazione viene appresa direttamente dagli esempi*.
- L'addestramento avviene utilizzando un grande set di dati etichettati e reti neurali «profonde», ossia contenenti un numero elevato di livelli hidden (decine o centinaia).
- L'addestramento può richiedere molto tempo (giorni o settimane). L'utilizzo di una o più GPU consente di velocizzare il processo.



Visione Artificiale - Raffaele Cappelli - raffaele.cappelli@unibo.it

Convolutional Neural Network (CNN)

- Reti neurali profonde specializzate per immagini e video: sono fra le tecniche di deep learning più utilizzate.
- A ciascuna immagine, a diverse risoluzioni, vengono applicati filtri, il cui output viene utilizzato come input per il livello seguente. I filtri possono catturare inizialmente feature molto semplici, come la luminosità e i bordi, per assumere via via forme più complesse, fino a definire l'oggetto da riconoscere.
- Utilizzano connessioni locali e condivisione dei pesi per ridurre drasticamente numero di parametri.



Utilizzare una semplice CNN con OpenCV

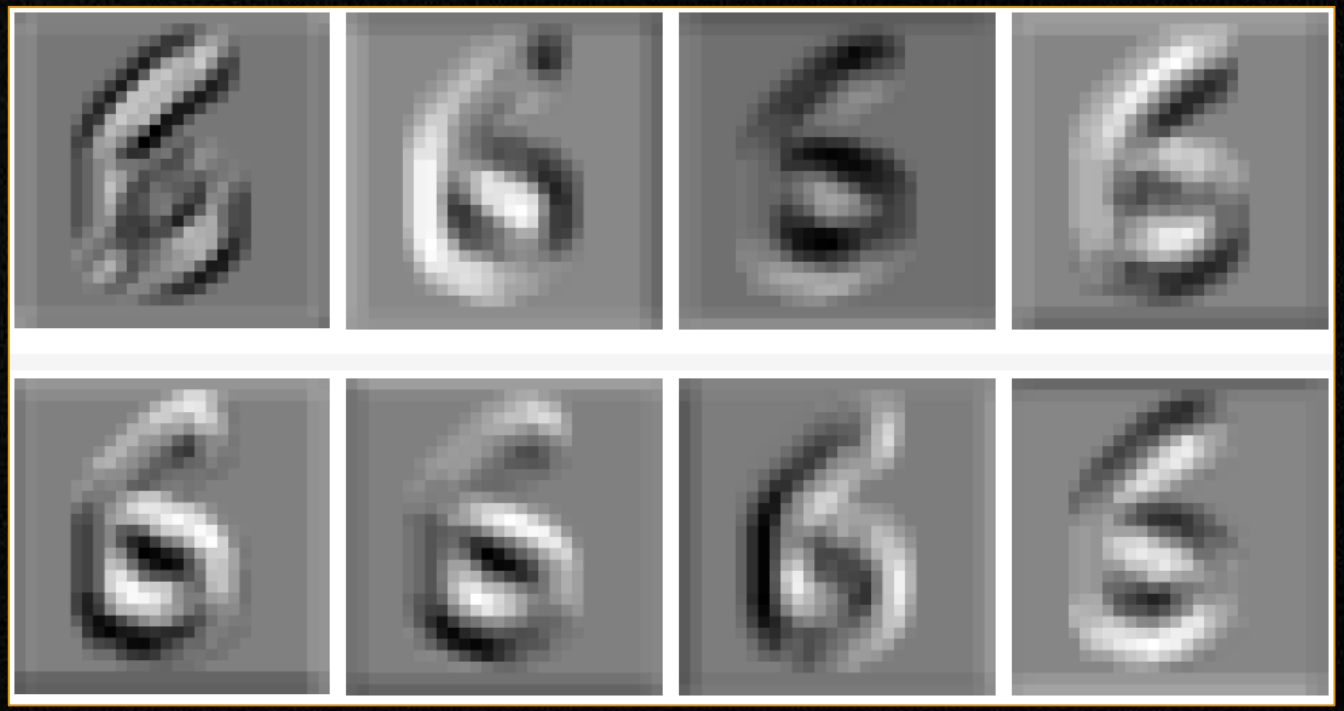

```
# Carica la rete da file
net = cv.dnn.readNet('minst/model.onnx')

# Carica un'immagine di test
img = cv.imread('minst/test_6.png', cv.IMREAD_GRAYSCALE)

# Fornisce l'immagine di test in ingresso alla rete
net.setInput(cv.dnn.blobFromImage(img))

# Esegue la rete per generare l'ouput (inferenza)
out = net.forward()
classe = np.argmax(out) # Risultato della classificazione (0..9)
```

- Esempio di utilizzo di una rete (già addestrata) per classificare immagini di cifre numeriche scritte a mano
- La rete riceve in ingresso un'immagine grayscale di 28x28 pixel e restituisce un vettore di 10 elementi (corrispondenti alle cifre [0,9]): la classe più probabile secondo la rete è quella corrispondente al valore più alto nel vettore.

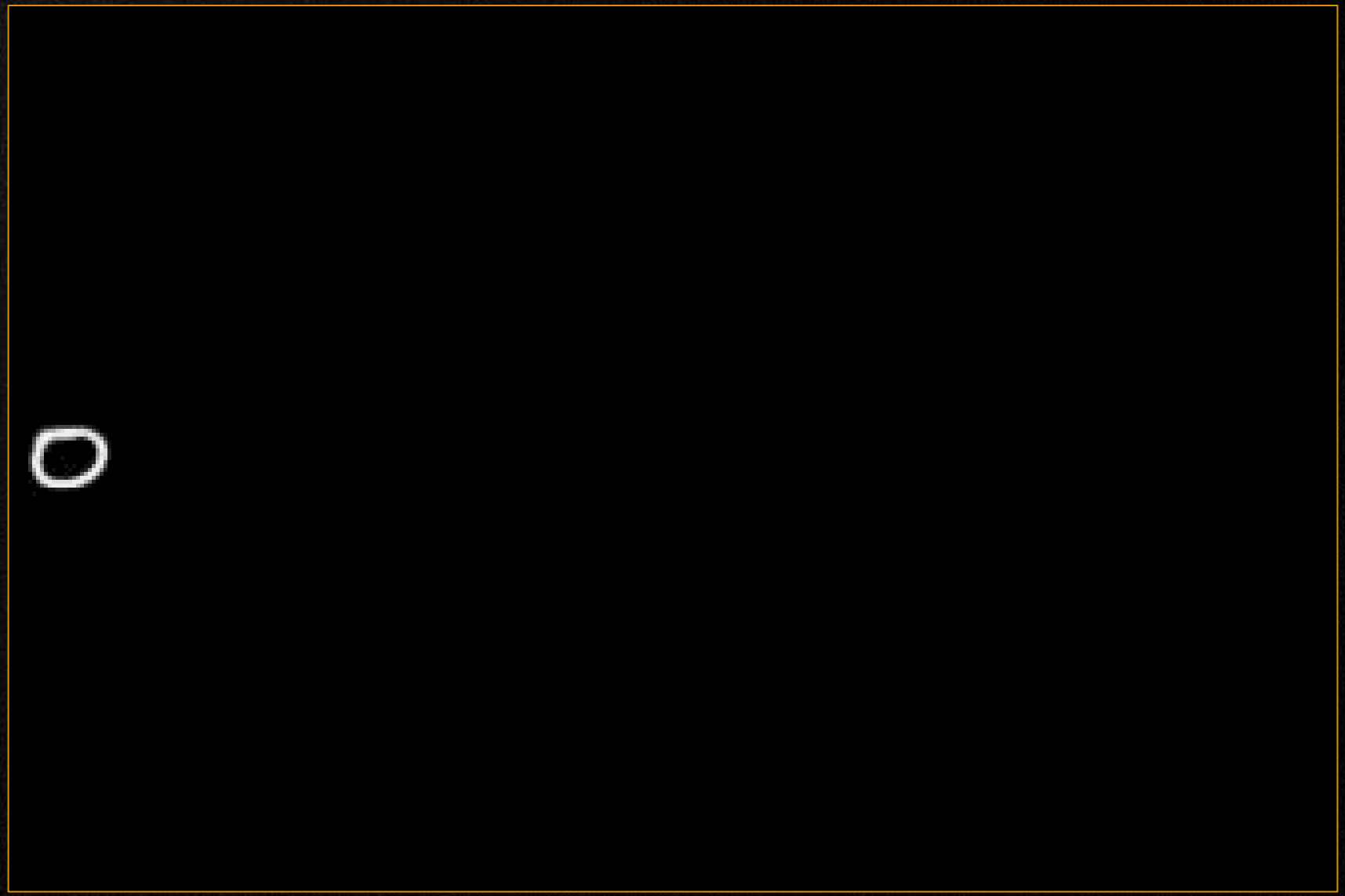
Ottenere informazioni sui livelli che compongono la rete


```
# Stampa tipo e forma dell'input e ouput di ogni livello
for i, l_in, l_out in zip(*net.getLayersShapes((1,1,28,28))):
    i = int(i[0]); l = net.getLayer(i)
    print(i, l.type, l_in[0].squeeze()[1:], l_out[0].squeeze()[1:])

# Esegue la rete e restituisce l'output di tutti i livelli
all_outs = net.forward(net.getLayerNames())
va.show(*all_outs[1][0]) # Visualizza l'output dei filtri del livello 1 come 8 immagini
```

Livello	Tipo	Input	Output
0		[1 28 28]	[1 28 28]
1	Convolution	[1 28 28]	[8 28 28]
2	Scale	[8 28 28]	[8 28 28]
3	Relu	[8 28 28]	[8 28 28]
4	Pooling	[8 28 28]	[8 14 14]
5	Convolution	[8 14 14]	[16 14 14]
6	Scale	[16 14 14]	[16 14 14]
7	Relu	[16 14 14]	[16 14 14]
8	Pooling	[16 14 14]	[16 4 4]
9	Reshape	[16 4 4]	[256]
10	InnerProduct	[256]	[10]
11	Scale	[10]	[10]



Visualizzazione ouput dei livelli intermedi e risultato finale



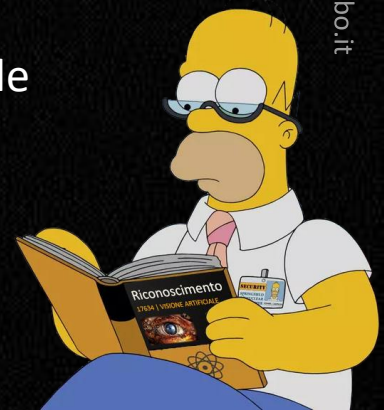
Un esempio di object detection: YOLO

- L'esempio seguente è stato ottenuto con OpenCV, fornendo in input l'immagine alla rete YOLO v3: si tratta di un sistema di object detection addestrato su 80 classi di oggetti, in grado di restituire, per ogni oggetto, il corrispondente bounding box.
  - Si tratta di una CNN con 106 livelli, per maggiori informazioni: https://pjreddie.com/darknet/yolo

Un esempio di object detection: YOLO

Riepilogo

- Riconoscimento di oggetti
  - Diverse possibili accezioni, con gradi diversi di difficoltà: classificazione, localizzazione, segmentazione delle istanze, etc.
  - Di solito il confronto di intere immagini pixel a pixel non può essere efficace
- Template matching
  - Funzionamento generale e misure di somiglianza fra template e immagine
  - Utilizzo del template matching in OpenCV
- Estrazione di caratteristiche (feature) da utilizzare per il riconoscimento
  - Approccio tradizionale: scelta manuale delle feature
  - Deep learning: anche le feature sono apprese a partire da esempi, addestrando reti neurali profonde
- Esempi di inferenza in OpenCV con reti neurali profonde (già addestrate)



Per approfondire

- Documentazione OpenCV del modulo «Image processing - Object detection»:
  - https://docs.opencv.org/master/df/dfb/group__imgproc__object.html
- Documentazione OpenCV del modulo «Deep Neural Network module»:
  - https://docs.opencv.org/master/d6/d0f/group__dnn.html
- Una trattazione approfondita di reti neurali e deep learning non è possibile in questo corso: ci siamo limitati a utilizzare alcune reti già addestrate; tuttavia chi volesse affrontare l'argomento, può partire da qui:
  - https://cs231n.github.io/convolutional-networks