Sapere utile

IFOA
Istituto Formazione Operatori Aziendali

BIG DATA e Analisi dei Dati

Mauro Bellone,
Robotics and AI researcher

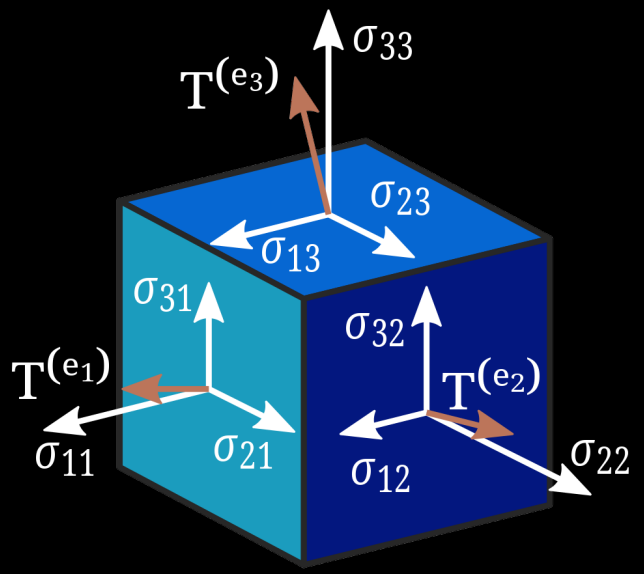
bellonemauro@gmail.com
www.maurobellone.com

Obiettivo

- ✓ Tensori
- ✓ Reti convoluzionali
- ✓ Tutorial su learning supervisionato con rete lineare e convoluzionale

Tensori

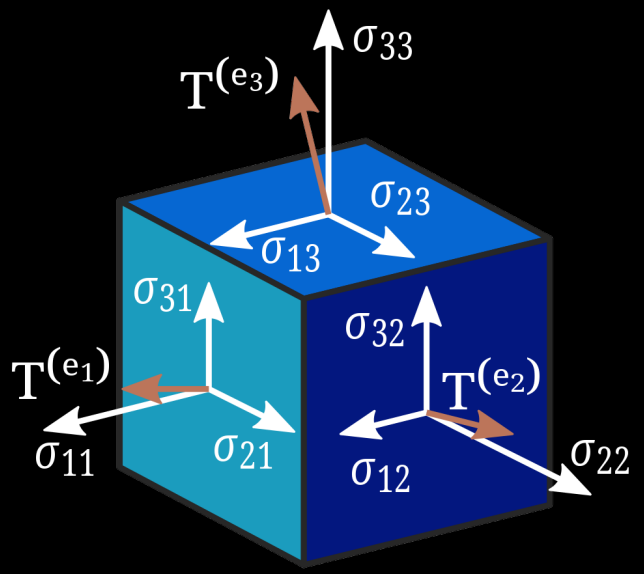
Un tensore è un oggetto algebrico che descrive una relazione multilineare tra insiemi di oggetti algebrici in uno spazio vettoriale



Tensori

Un tensore è un oggetto algebrico che descrive una relazione multilineare tra insiemi di oggetti algebrici in uno spazio vettoriale

Di fatto sono una generalizzazione di vettori e matrici rappresentati come array multidimensionali

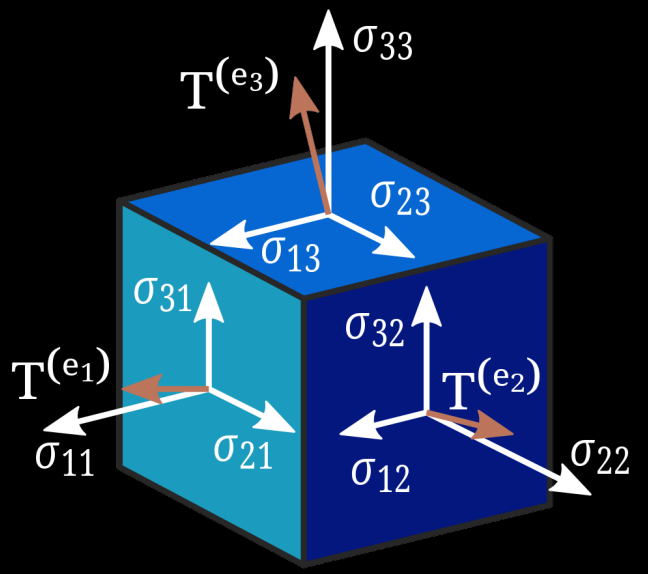


Tensori

Un tensore è un oggetto algebrico che descrive una relazione multilineare tra insiemi di oggetti algebrici in uno spazio vettoriale

Di fatto sono una generalizzazione di vettori e matrici rappresentati come array multidimensionali

Le dimensioni sono comunemente indicate come “cardinalità”



Tensori

$T[1] = t_1 \in \mathbb{R}$ ← scalare

Tensori

$$T[1] = t_1 \in \mathbb{R}$$

$$T[n] = [t_1 \cdots t_n] \in \mathbb{R}^n$$

vettore

Tensori

$$T[1] = t_1 \in \mathbb{R}$$

$$T[n] = [t_1 \cdots t_n] \in \mathbb{R}^n$$

$$T[n,n] = \begin{bmatrix} t_{1,1} & \cdots & t_{1,n} \\ \vdots & \ddots & \vdots \\ t_{n,1} & \cdots & t_{n,n} \end{bmatrix} \in \mathbb{R}^{n,n}$$

matrice

Tensori

$$T[1] = t_1 \in \mathbb{R}$$

$$T[n] = [t_1 \cdots t_n] \in \mathbb{R}^n$$

$$T[n,n] = \begin{bmatrix} t_{1,1} & \cdots & t_{1,n} \\ \vdots & \ddots & \vdots \\ t_{n,1} & \cdots & t_{n,n} \end{bmatrix} \in \mathbb{R}^{n,n}$$

$$T[n,n,n] = \left[\begin{bmatrix} t_{1,1,1} & \cdots & t_{1,n,1} \\ \vdots & \ddots & \vdots \\ t_{n,1,1} & \cdots & t_{n,n,1} \end{bmatrix}, \dots, \begin{bmatrix} t_{1,1,n} & \cdots & t_{1,n,n} \\ \vdots & \ddots & \vdots \\ t_{n,1,n} & \cdots & t_{n,n,n} \end{bmatrix} \right] \in \mathbb{R}^{n,n,n}$$

Tensore

Operazioni tra tensori

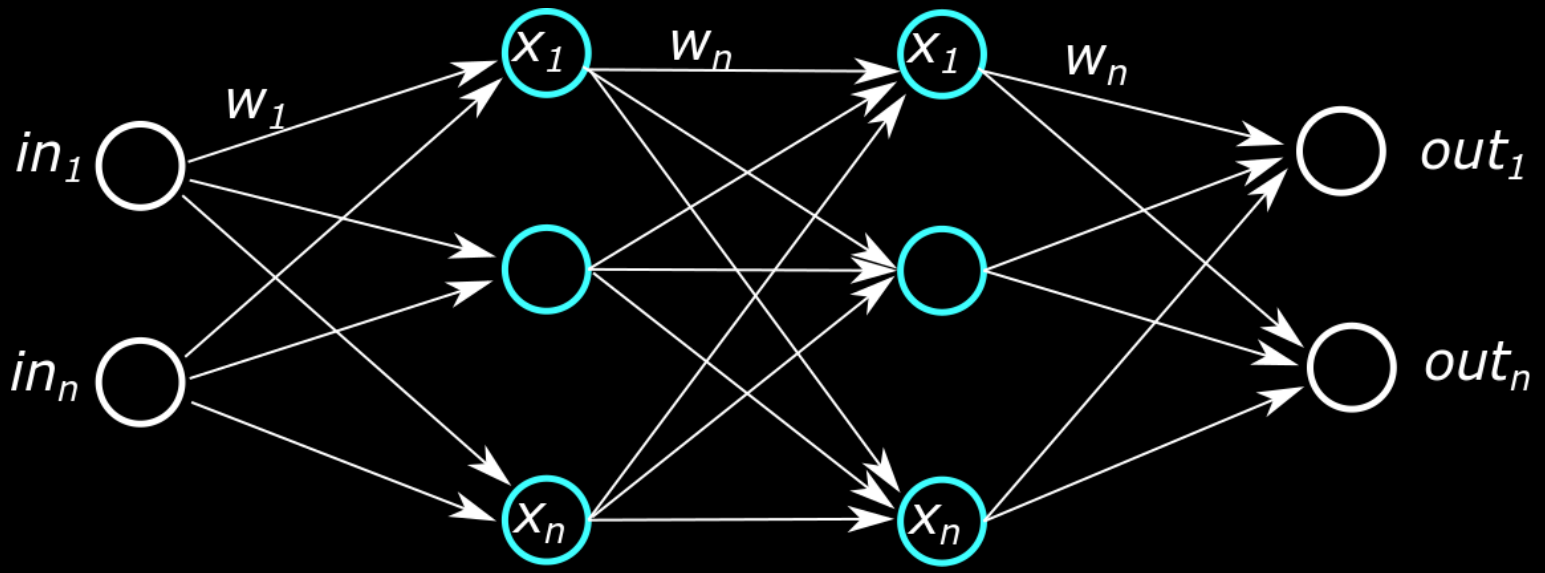
- ✓ Addizione, sottrazione (tensori della stessa dimensione)
- ✓ Moltiplicazione, divisione – elemento per elemento (Hadamard product)
- ✓ Prodotto tensoriale (come il prodotto vettoriale ma su tensori, il prodotto vettoriale è un caso particolare di prodotto tensoriale)
- ✓ Reshape

Rappresentazione del mondo reale in tensori

Il tensore è l'oggetto usato per rappresentare il mondo reale come una serie ordinata di informazioni in una rappresentazione normale e standardizzata

$T[in] = [in_1, in_2] \in \mathbb{R}^2$

$T[out] = [out_1, out_2] \in \mathbb{R}^2$



Tensori – Immagine BW

Una immagine nelle reti è tipicamente rappresentata come un tensore [h,w,BW] dove *h* è l'altezza dell'immagine (risoluzione in px), *w* è la larghezza (in px), e BW è la profondità del colore (in questo caso scala di grigi).

$$T[h, w, 1] = \left[\begin{bmatrix} t_{1,1,1} & \cdots & t_{1,w,1} \\ \vdots & \ddots & \vdots \\ t_{h,1,1} & \cdots & t_{h,w,1} \end{bmatrix}\right] \in \mathbb{R}^{h,w,1}$$

In bianco e nero potevamo rappresentare l'immagine come un tensore [h,w]

Tensori – Immagine RGB

Una immagine nelle reti è tipicamente rappresentata come un tensore [h,w,RGB] dove *h* è l'altezza dell'immagine (risoluzione in px), *w* è la larghezza (in px), e RGB è la profondità del colore 3 (rosso, verde, blu).

$$T[h,w,3] = \left[\begin{bmatrix} t_{1,1,1} & \cdots & t_{1,w,1} \\ \vdots & \ddots & \vdots \\ t_{h,1,1} & \cdots & t_{h,w,1} \end{bmatrix}, \begin{bmatrix} t_{1,1,2} & \cdots & t_{1,w,2} \\ \vdots & \ddots & \vdots \\ t_{h,1,2} & \cdots & t_{h,w,2} \end{bmatrix}, \begin{bmatrix} t_{1,1,3} & \cdots & t_{1,w,3} \\ \vdots & \ddots & \vdots \\ t_{h,1,3} & \cdots & t_{h,w,3} \end{bmatrix}\right] \in \mathbb{R}^{h,w,3}$$

Tensori – Immagine RGB

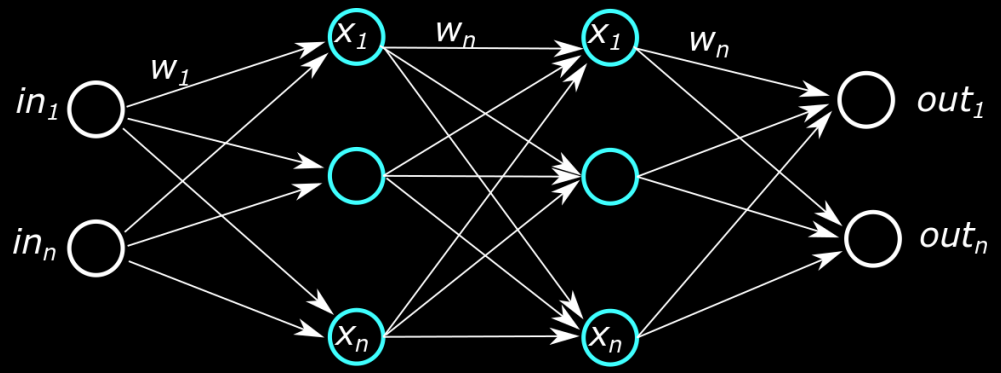
Una immagine nelle reti è tipicamente rappresentata come un tensore [h,w,RGB] dove *h* è l'altezza dell'immagine (risoluzione in px), *w* è la larghezza (in px), e RGB è la profondità del colore 3 (rosso, verde, blu).

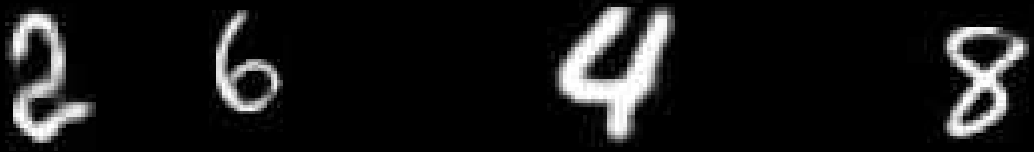
$$T[h,w,3] = \left[\begin{bmatrix} t_{1,1,1} & \cdots & t_{1,w,1} \\ \vdots & \ddots & \vdots \\ t_{h,1,1} & \cdots & t_{h,w,1} \end{bmatrix}, \begin{bmatrix} t_{1,1,2} & \cdots & t_{1,w,2} \\ \vdots & \ddots & \vdots \\ t_{h,1,2} & \cdots & t_{h,w,2} \end{bmatrix}, \begin{bmatrix} t_{1,1,3} & \cdots & t_{1,w,3} \\ \vdots & \ddots & \vdots \\ t_{h,1,3} & \cdots & t_{h,w,3} \end{bmatrix}\right] \in \mathbb{R}^{h,w,3}$$

In realtà nelle rappresentazioni computazionali è necessario indicare il numero di bit rappresentati (16bit, 32bit, etc.)

MNIST

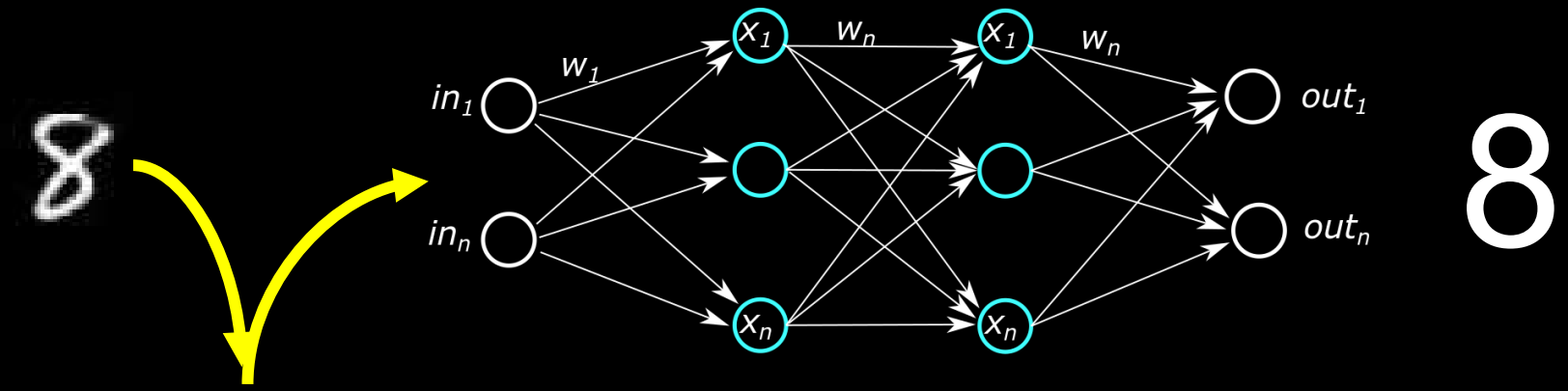
Creato nel 1998 per il riconoscimento di caratteri “hand-written” contiene 60.000 immagini di training e 10.000 immagini di test di risoluzione 28x28 px in bw.





MNIST

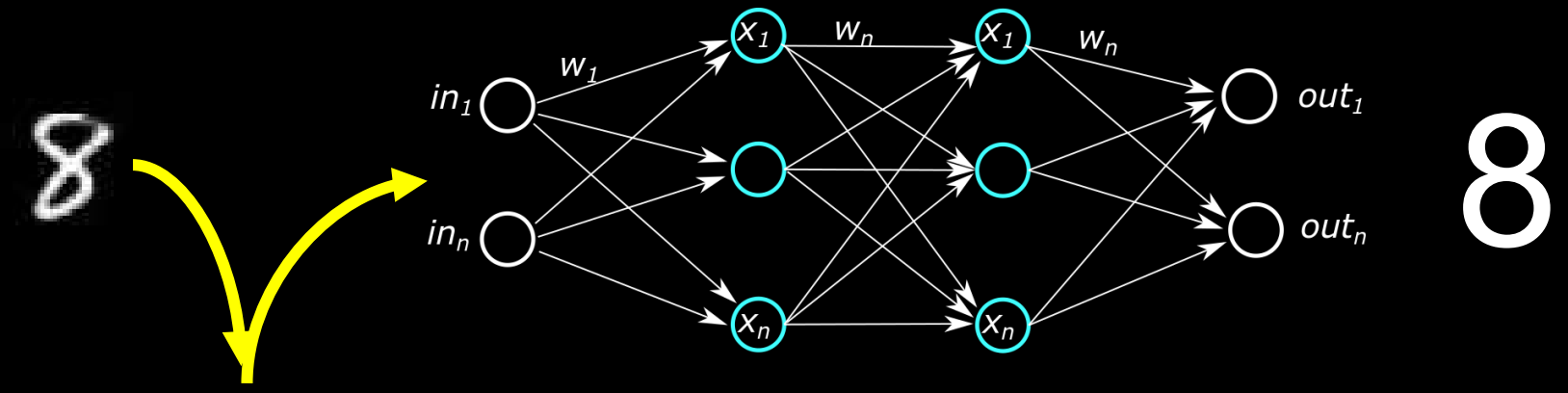
Creato nel 1998 per il riconoscimento di caratteri “hand-written” contiene 60.000 immagini di training e 10.000 immagini di test di risoluzione 28x28 px in bw.



$$8 = [1 \quad 0 \quad 0 \quad \cdots \quad 0 \quad 1 \quad 0]^T$$ operatore flatten o vettorializzazione

MNIST

Creato nel 1998 per il riconoscimento di caratteri “hand-written” contiene 60.000 immagini di training e 10.000 immagini di test di risoluzione 28x28 px in bw.



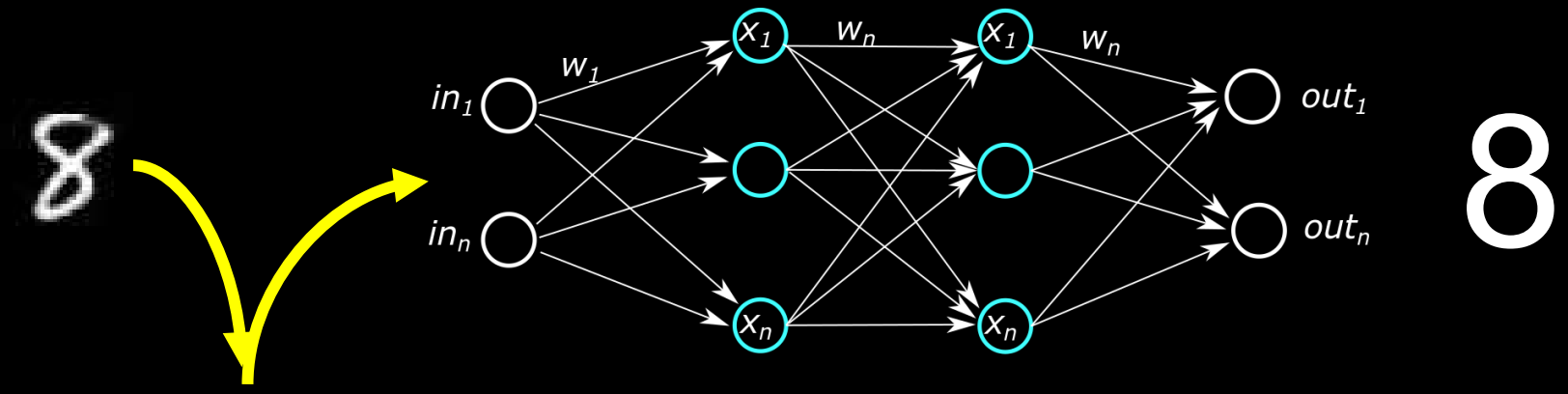
$8 = [1 \quad 0 \quad 0 \quad \cdots \quad 0 \quad 1 \quad 0]^T$ operatore flatten o vettorializzazione

Il database è disponibile al seguente link: http://yann.lecun.com/exdb/mnist/

dove è possibile anche trovare una lista di reti testate su questo dataset e le relative prestazioni

MNIST

Creato nel 1998 per il riconoscimento di caratteri “hand-written” contiene 60.000 immagini di training e 10.000 immagini di test di risoluzione 28x28 px in bw.

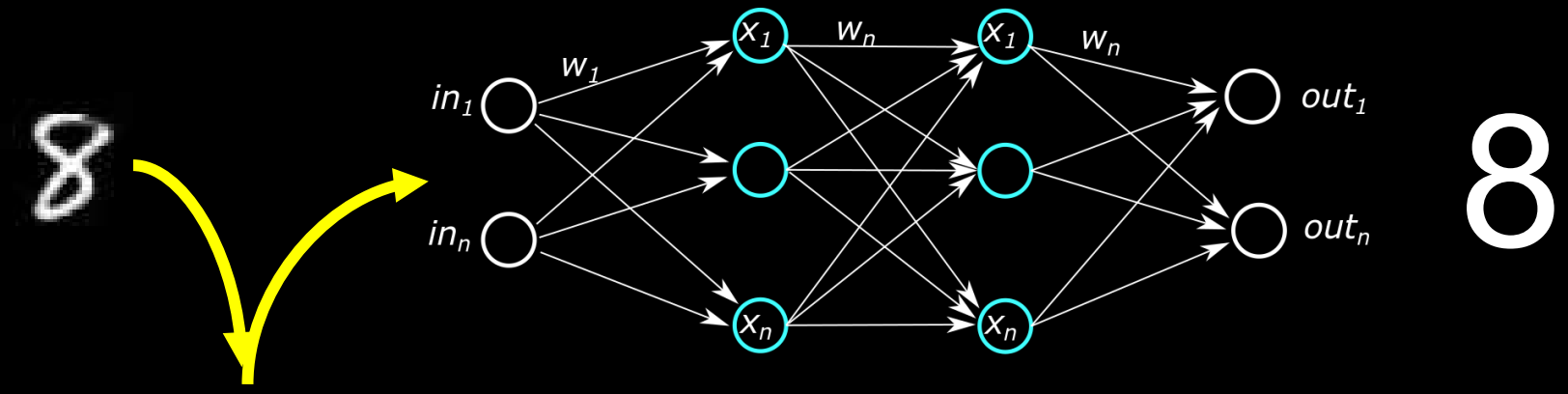


$8 = [1 \quad 0 \quad 0 \quad \cdots \quad 0 \quad 1 \quad 0]^T$ operatore flatten o vettorializzazione

DOMANDA: quanti parametri dobbiamo avere per allenare una rete singolo livello?

MNIST

Creato nel 1998 per il riconoscimento di caratteri “hand-written” contiene 60.000 immagini di training e 10.000 immagini di test di risoluzione 28x28 px in bw.



$8 = [1 \quad 0 \quad 0 \quad \cdots \quad 0 \quad 1 \quad 0]^T$ operatore flatten o vettorializzazione

DOMANDA: quanti parametri dobbiamo avere per allenare una rete singolo livello?

n_neuroni_in x n_neuroni_out

MNIST - Tutorial

Il database è disponibile al seguente link: http://yann.lecun.com/exdb/mnist/

dove è possibile anche trovare una lista di reti testate su questo dataset e le relative prestazioni

CIFAR10

Il datast CIFAR-10 (Canadian Institute For Advanced Research) è un insieme di immagini che sono comunemente usate per il training delle reti neurali e in generale testare algoritmi di machine learning.

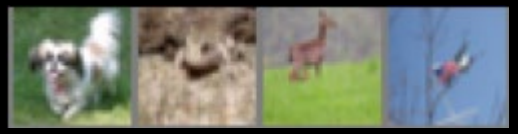
Il dataset CIFAR-10 contiene 60.000 immagini a colori della dipmensione pari a 32x32 divisi in 10 classi:

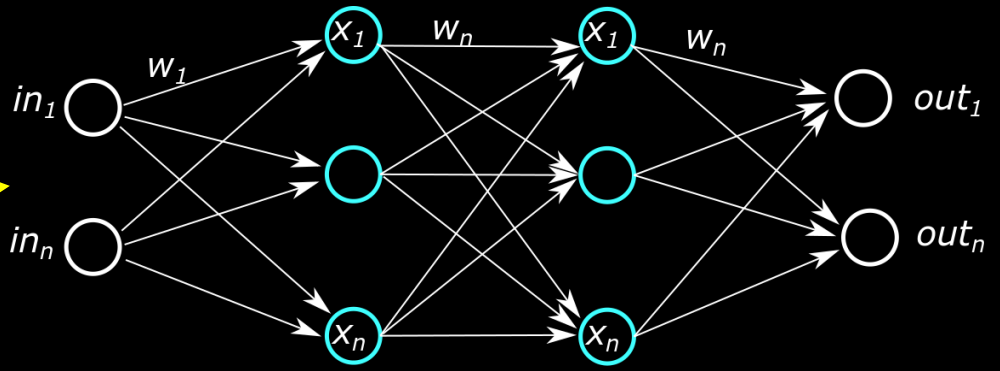
- ✓ aerei
- ✓ automobili
- ✓ uccelli
- ✓ gatti
- ✓ cervi
- ✓ cani
- ✓ rane
- ✓ cavalli
- ✓ navi
- ✓ camion

CIFAR10

Il datast CIFAR-10 (Canadian Institute For Advanced Research) è un insieme di immagini che sono comunemente usate per il training delle reti neurali e in generale testare algoritmi di machine learning.

Il dataset CIFAR-10 contiene 60.000 immagini a colori della dimensione pari a 32x32 divisi in 10 classi:





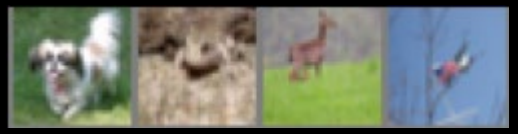
cane
rana
cervo
aereo

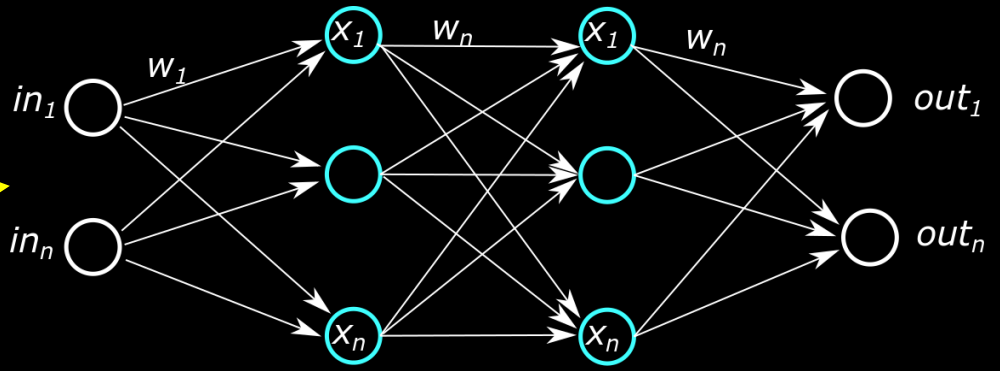
$$rana = [1 \quad 0 \quad 0 \quad \cdots \quad 0 \quad 1 \quad 0]^T$$ operatore flatten

Tutorial – CIFAR10

Il datast CIFAR-10 (Canadian Institute For Advanced Research) è un insieme di immagini che sono comunemente usate per il training delle reti neurali e in generale testare algoritmi di machine learning.

Il dataset CIFAR-10 contiene 60.000 immagini a colori della dimensione pari a 32x32 divisi in 10 classi:





cane
rana
cervo
aereo

$$rana = [1 \quad 0 \quad 0 \quad \cdots \quad 0 \quad 1 \quad 0]^T$$ operatore flatten

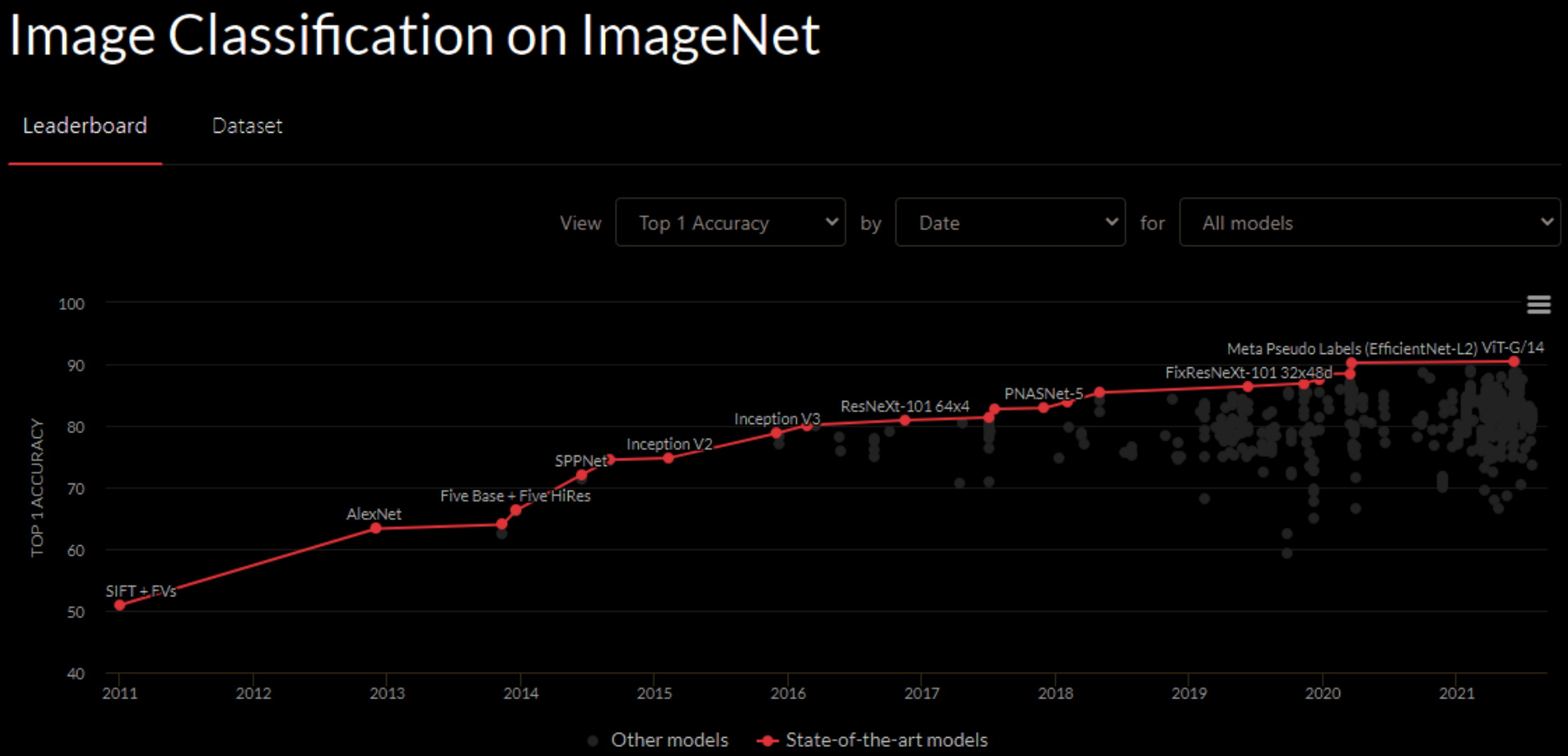
Benchmark disponibile al seguente link: https://benchmarks.ai/cifar-10

ImageNET

Iniziato nel 2006 ha l'obiettivo di creare un dataset per un task di classificazione di immagini (attualmente oltre 20.000 categorie in 14 milioni di immagini) .

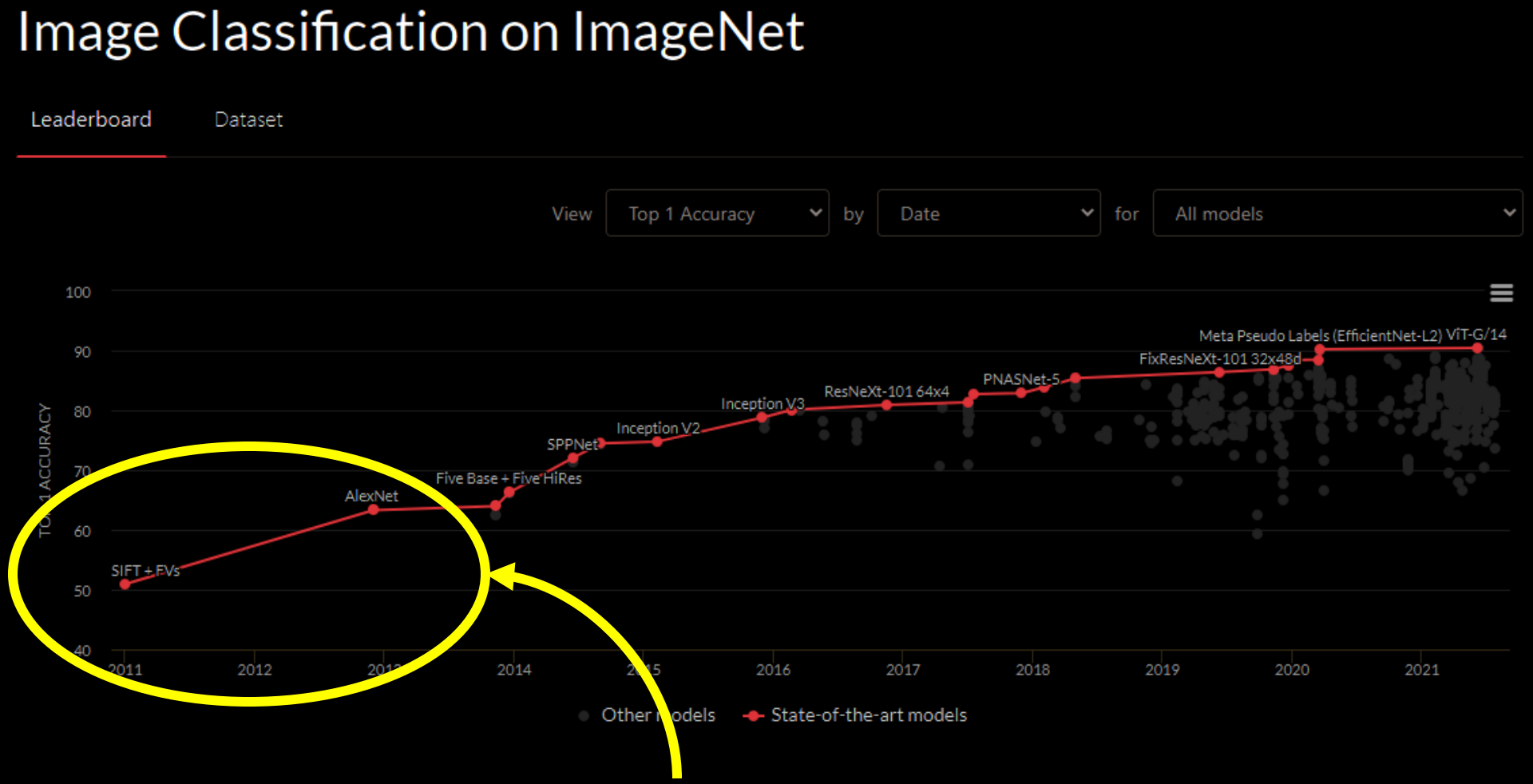
source: Deng, J., Dong, W., Socher, R., Li, L.J., Li, K. and Fei-Fei, L., 2009, June. Imagenet: A large-scale hierarchical image database. In 2009 IEEE conference on computer vision and pattern recognition (pp. 248-255). Ieee.

AI-challenges



source: https://paperswithcode.com/sota/image-classification-on-imagenet

AI-challenges



Discutiamo i primi due approcci di questa lista, SIFT e AlexNet che hanno tra loro una differenza del 15% in termini di accuratezza

Feature engineering

Feature engineering è il processo di rappresentazione di una immagine attraverso delle sue caratteristiche salienti chiamate «features»

E’ necessario, all’interno di una immagine, trovare delle features che siano indipendenti rispetto alle trasformazioni di base come cambiamenti di illuminazione, trasformazione di scala, rotazione, cambio del punto di osservazione

SIFT: Scale Invariant Feature Transform

SIFT è un approccio per rilevare e descrivere delle regioni di interesse in una immagine

SIFT: Scale Invariant Feature Transform

SIFT è un approccio per rilevare e descrivere delle regioni di interesse in una immagine

E’ invariante (accettabilmente) rispetto a: rotazione, scala, illuminazione e viewpoint

SIFT: Scale Invariant Feature Transform

SIFT è un approccio per rilevare e descrivere delle regioni di interesse in una immagine

E’ invariante (accettabilmente) rispetto a: rotazione, scala, illuminazione e viewpoint

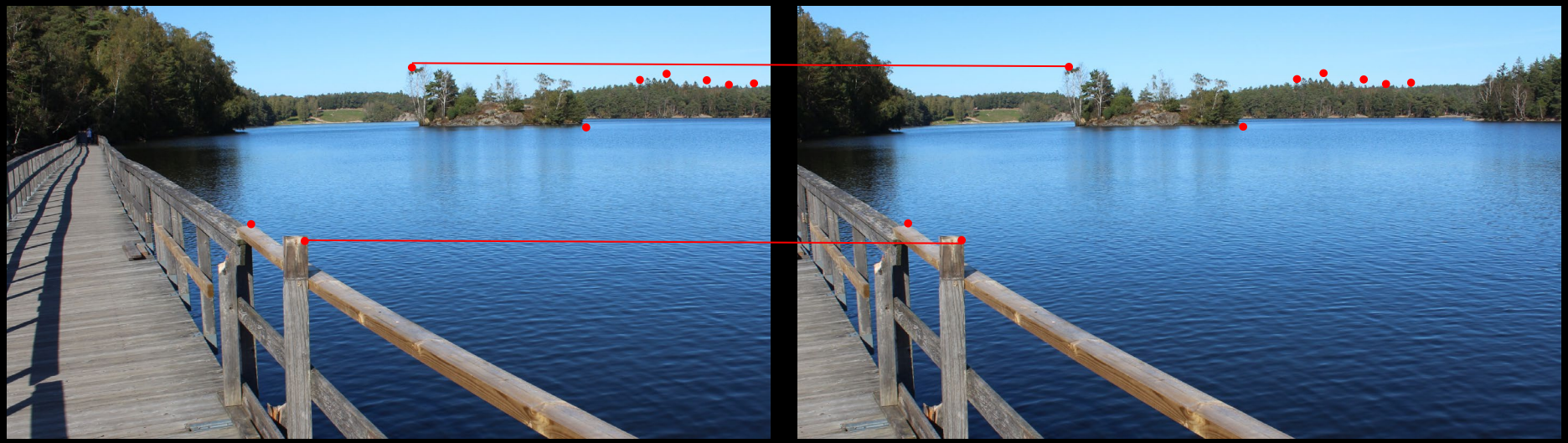
E’ potente nel rilevare e descrivere delle caratteristiche distintive ma anche molto onerosa dal punto di vista computazionale

SIFT: Scale Invariant Feature Transform

SIFT è un approccio per rilevare e descrivere delle regioni di interesse in una immagine

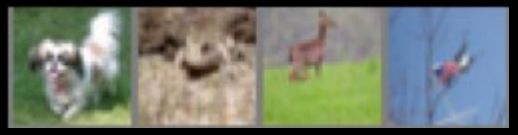
E’ invariante (accettabilmente) rispetto a: rotazione, scala, illuminazione e viewpoint

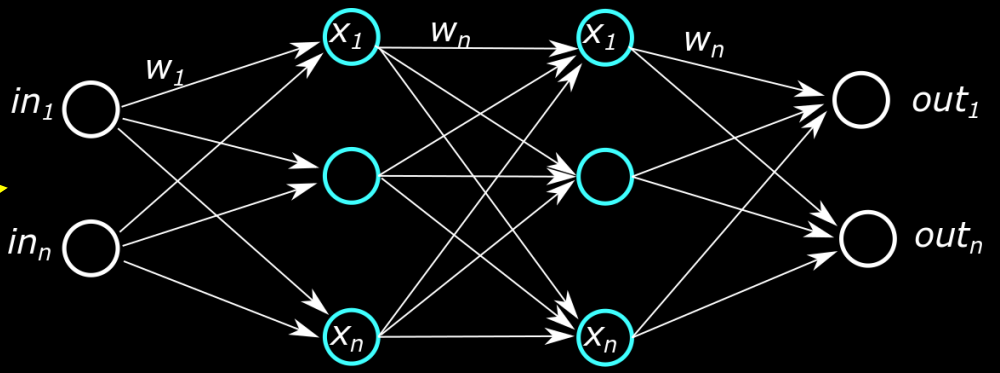
E’ potente nel rilevare e descrivere delle caratteristiche distintive ma anche molto onerosa dal punto di vista computazionale



Addestramento NN sulle features

L'idea, per evitare di assegnare alla rete neurale l'immagine raw, è quella di assegnare al vettore in input una serie di features che siano maggiormente discriminatve





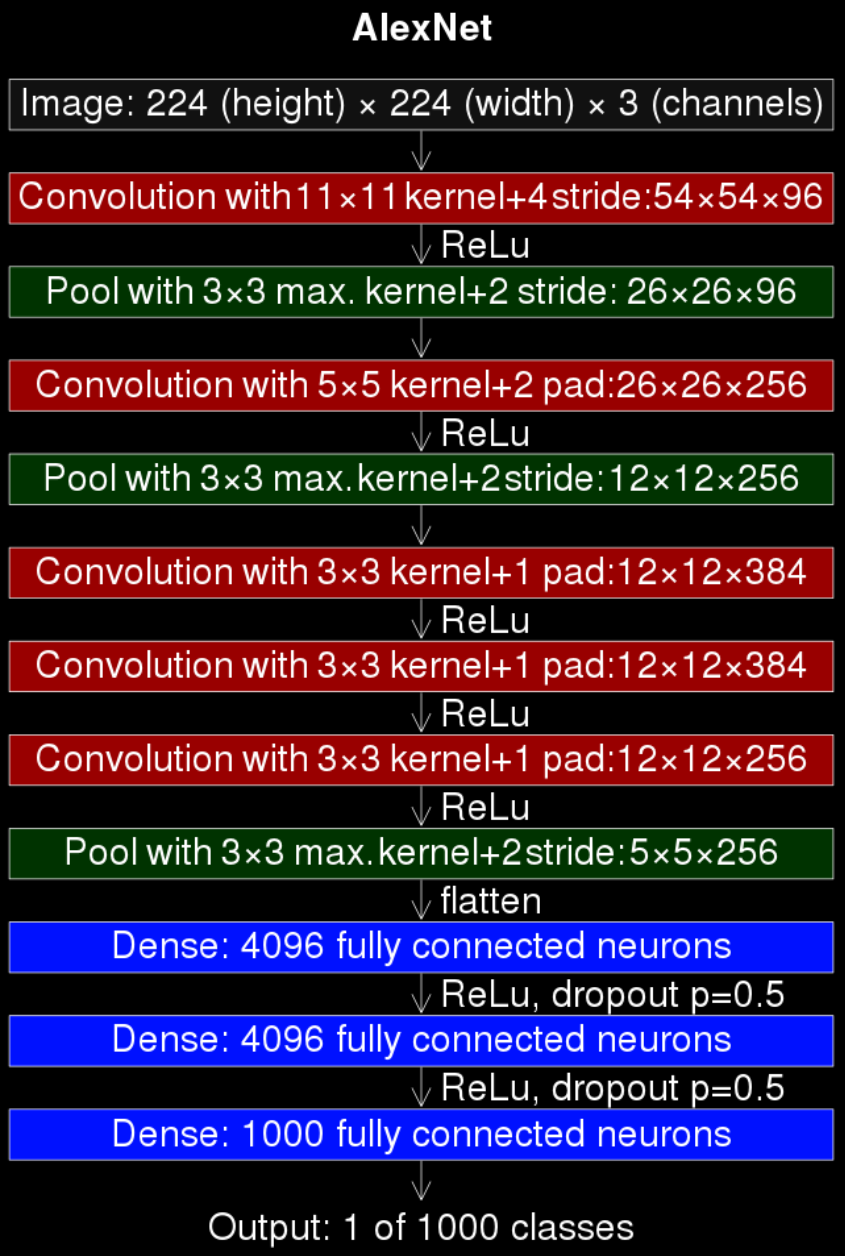
dog
frog
deer
plane

$$[SIFT \quad \cdots altra\ feature \ \ldots \quad Feature\ n]^T$$

AlexNet

Approccio con feature extraction

- ✓ Immagini RGB su 3 canali
- ✓ Kernel di convoluzione
- ✓ linearizzazione finale
- ✓ NN lineare



source: https://en.wikipedia.org/wiki/AlexNet

Feature engineering Vs feature identification

L’idea di base era quella di lasciare che la rete identificasse da se le features rilevanti per la classificazione, facendo inferenza statistica sui «kernel» convoluzionali

feature map identificate in alexNet paper

source: Krizhevsky, A., Sutskever, I. and Hinton, G.E., 2012. Imagenet classification with deep convolutional neural networks. Advances in neural information processing systems, 25, pp.1097-1105.

Feature engineering Vs feature identification

L'idea di base era quella di lasciare che la rete identificasse da se le features rilevanti per la classificazione, facendo inferenza statistica sui «kernel» convoluzionali

Concause:

- ✓ Sviluppo di kernel convoluzionali (neocognitron, 1979, sviluppato per riconoscere caratteri giapponesi)
- ✓ Creazione di un big-dataset ImageNet
- ✓ Disponibilità di GPU nvidia

feature map identificate in alexNet paper

source: Krizhevsky, A., Sutskever, I. and Hinton, G.E., 2012. Imagenet classification with deep convolutional neural networks. Advances in neural information processing systems, 25, pp.1097-1105.

Kernel convoluzionale

La convoluzione nelle reti neurali convoluzionali è essenzialmente una moltiplicazione matriciale con una finestra mobile

Kernel convoluzionale

La convoluzione nelle reti neurali convoluzionali è essenzialmente una moltiplicazione matriciale con una finestra mobile

Per semplicità indichiamo $K[n, m]$ un tensore bi-dimensionale con *n* righe ed *m* colonne:

$$K[n,m] = \begin{bmatrix} k_{1,1} & \cdots & k_{1,m} \\ \vdots & \ddots & \vdots \\ k_{n,1} & \cdots & k_{n,m} \end{bmatrix} \in \mathbb{R}^{n,m}$$

Kernel convoluzionale

La convoluzione nelle reti neurali convoluzionali è essenzialmente una moltiplicazione matriciale con una finestra mobile

Per semplicità indichiamo $K[n, m]$ un tensore bi-dimensionale con *n* righe ed *m* colonne:

$$K[n,m] = \begin{bmatrix} k_{1,1} & \cdots & k_{1,m} \\ \vdots & \ddots & \vdots \\ k_{n,1} & \cdots & k_{n,m} \end{bmatrix} \in \mathbb{R}^{n,m}$$

e con $I[h, w, 1]$ una immagine bidimensionale (scala di grigi):

$$I[h,w,1] = \begin{bmatrix} \begin{bmatrix} i_{1,1,1} & \cdots & i_{1,w,1} \\ \vdots & \ddots & \vdots \\ i_{h,1,1} & \cdots & i_{h,w,1} \end{bmatrix} \end{bmatrix} \in \mathbb{R}^{h,w,1}$$

Kernel convoluzionale

La convoluzione nelle reti neurali convoluzionali è essenzialmente una moltiplicazione matriciale con una finestra mobile

La convoluzione tra l'immagine è il kernel fornirà in uscita una nuova immagine, filtrata attraverso il kernel scelto.

$$I[h, w, 1] * K[n, m] = I_l \left[h - \frac{n}{2}, w - \frac{m}{2}, 1 \right]$$

Kernel convoluzionale

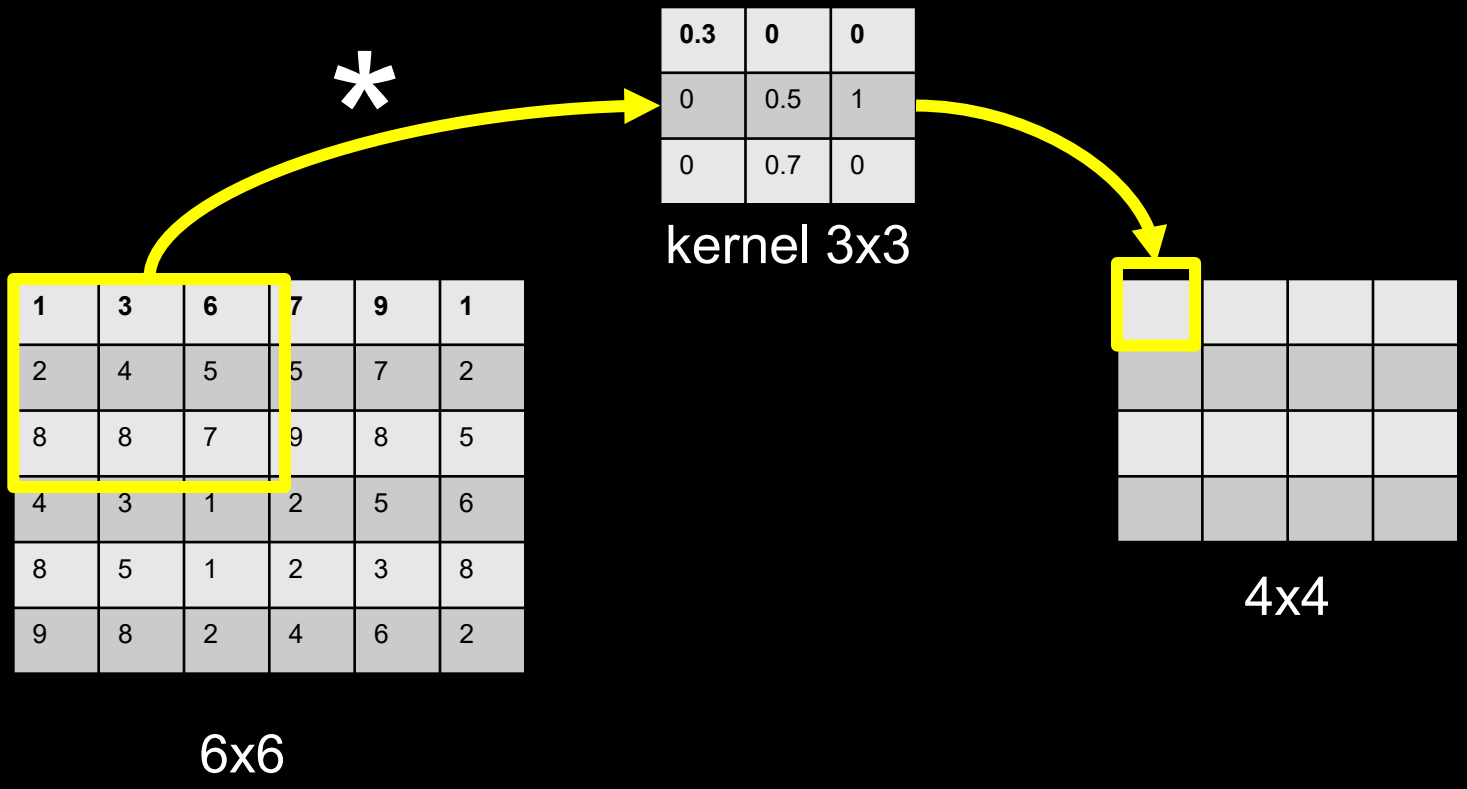
La convoluzione nelle reti neurali convoluzionali è essenzialmente una moltiplicazione matriciale con una finestra mobile

La convoluzione tra l'immagine è il kernel fornirà in uscita una nuova immagine, filtrata attraverso il kernel scelto.

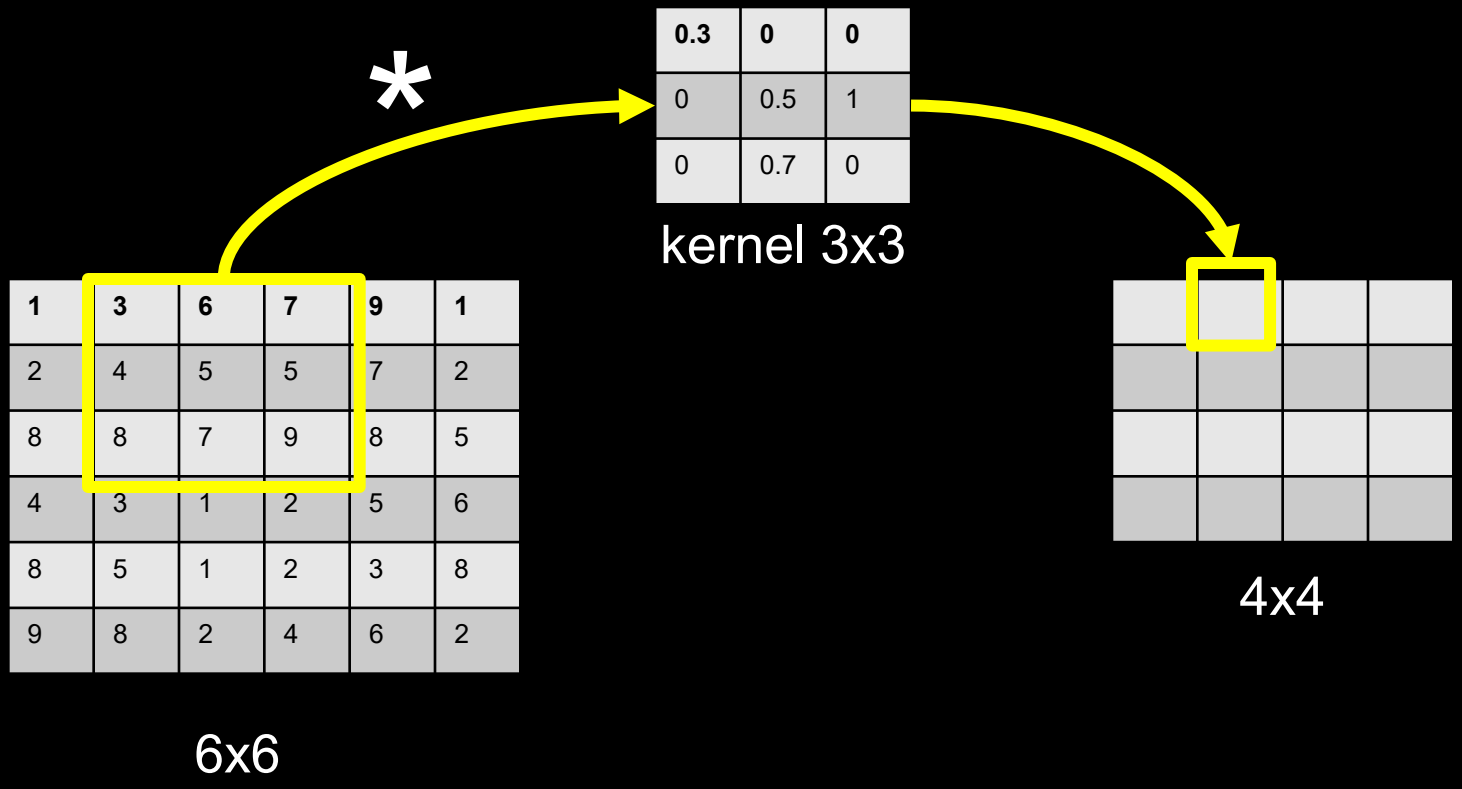
$$I[h, w, 1] * K[n, m] = I_l \left[h - \frac{n}{2}, w - \frac{m}{2}, 1 \right]$$

Notare che le dimensioni dell'immagine cambiano in base alla dimensione del kernel a causa dello scorrimento (il valore n/2 è appossimato all'intero se dispari)

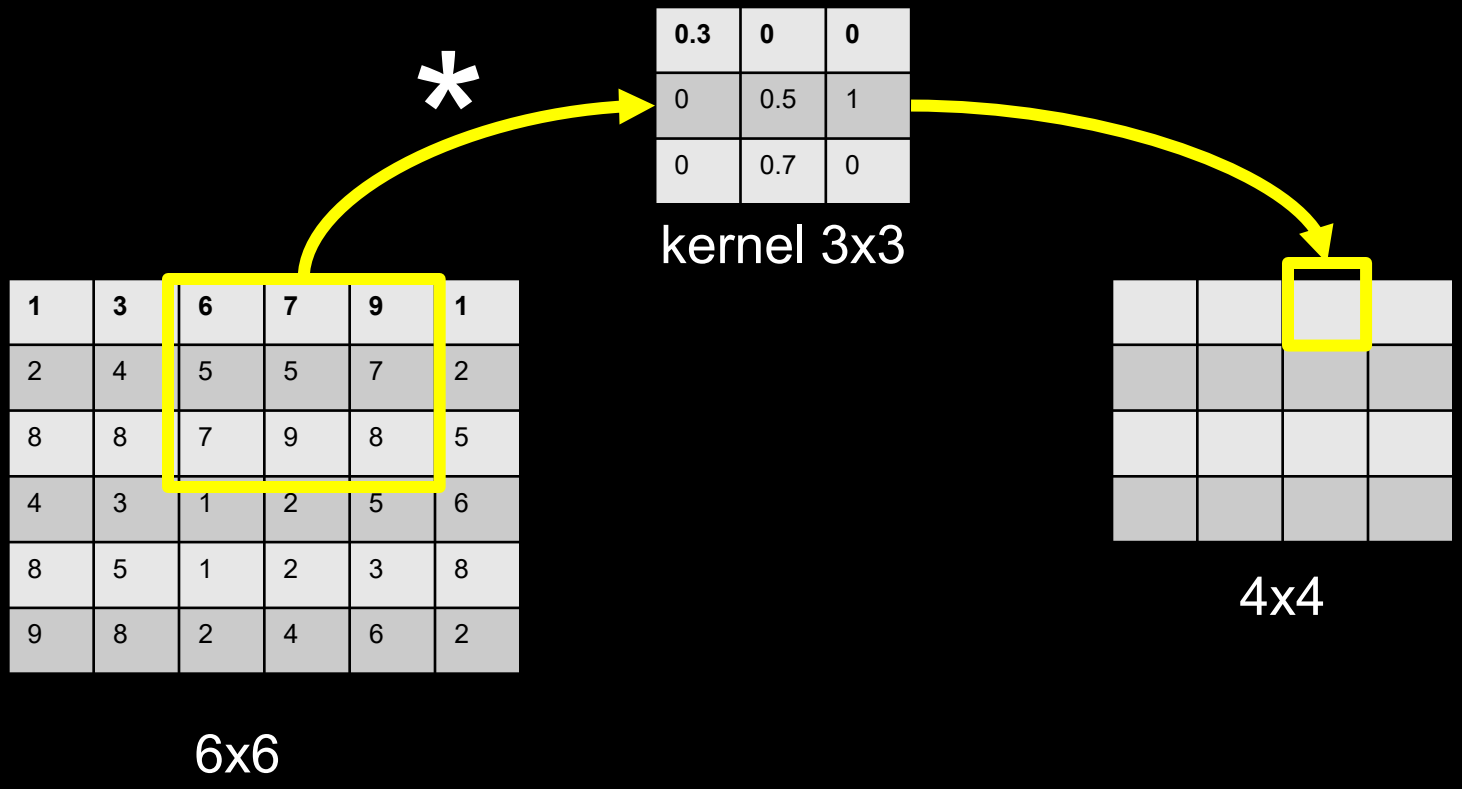
Kernel convoluzionale



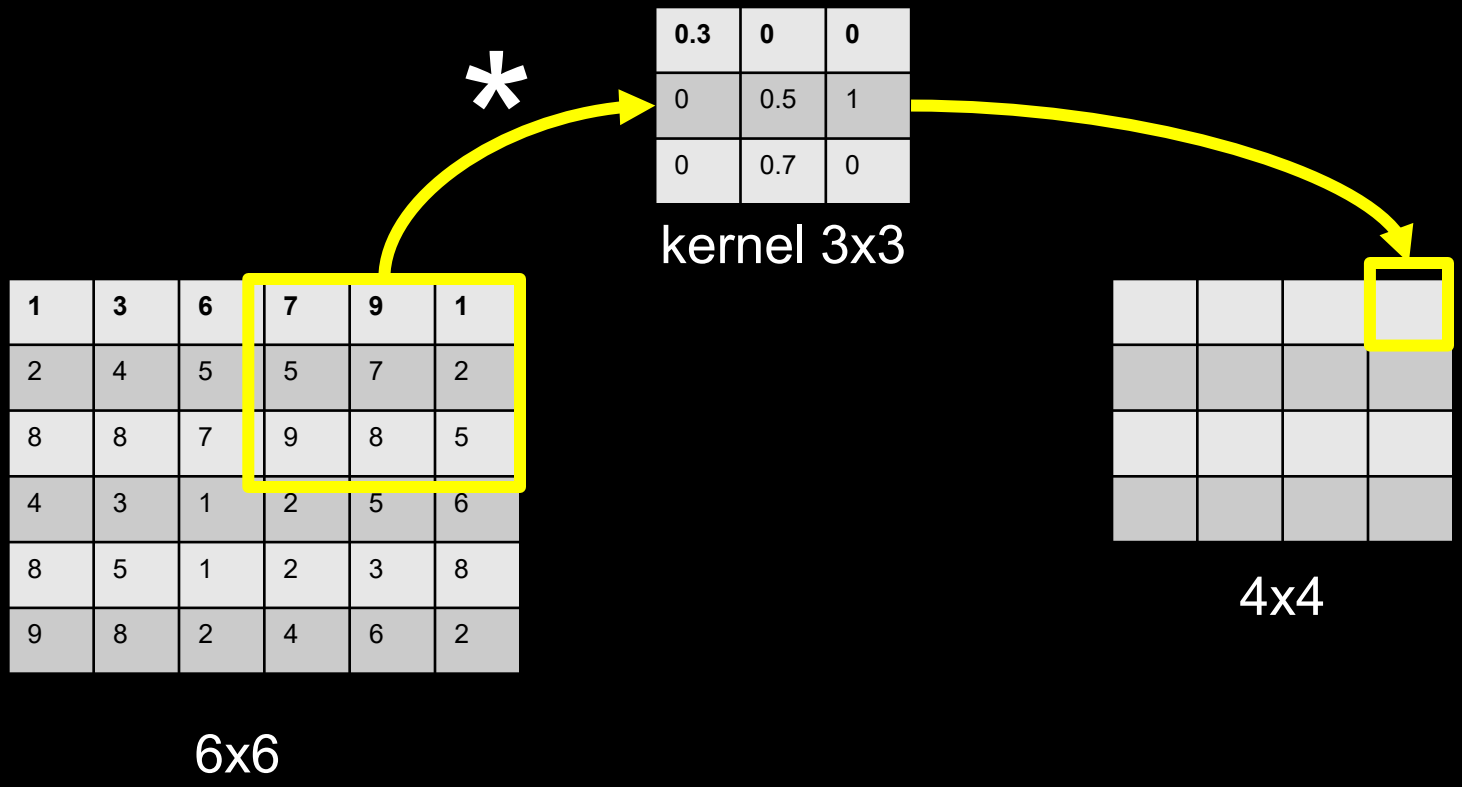
Kernel convoluzionale



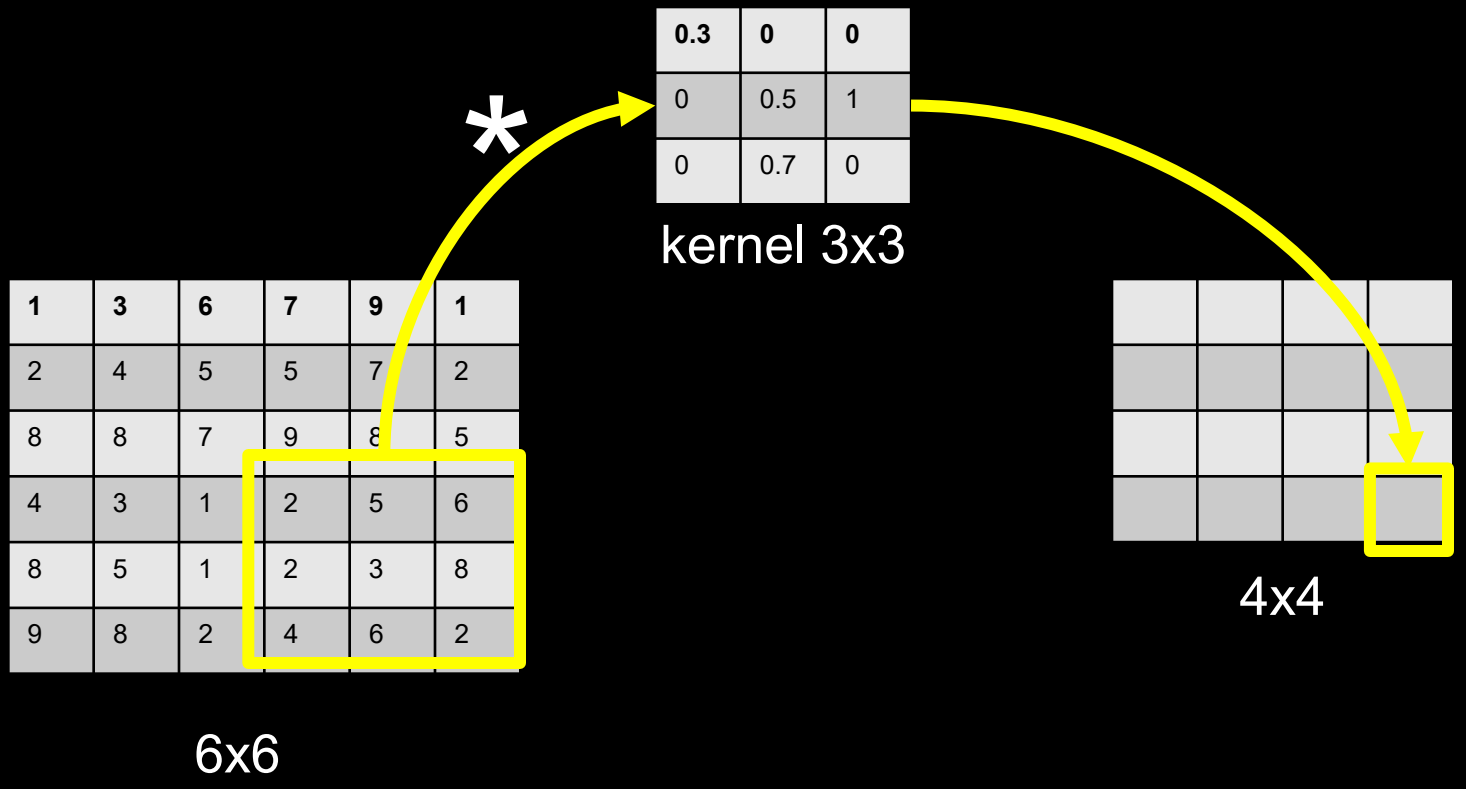
Kernel convoluzionale



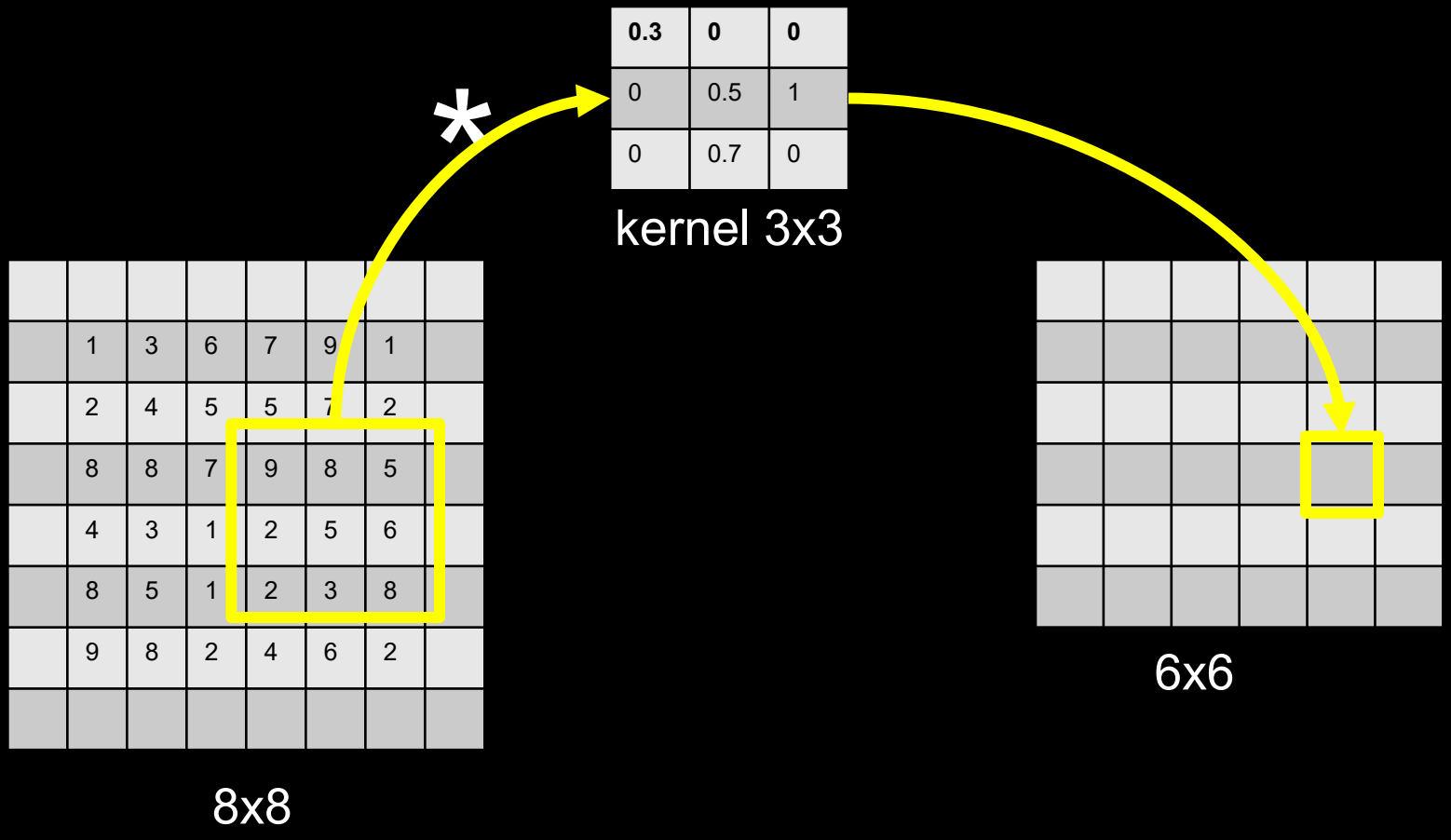
Kernel convoluzionale



Kernel convoluzionale

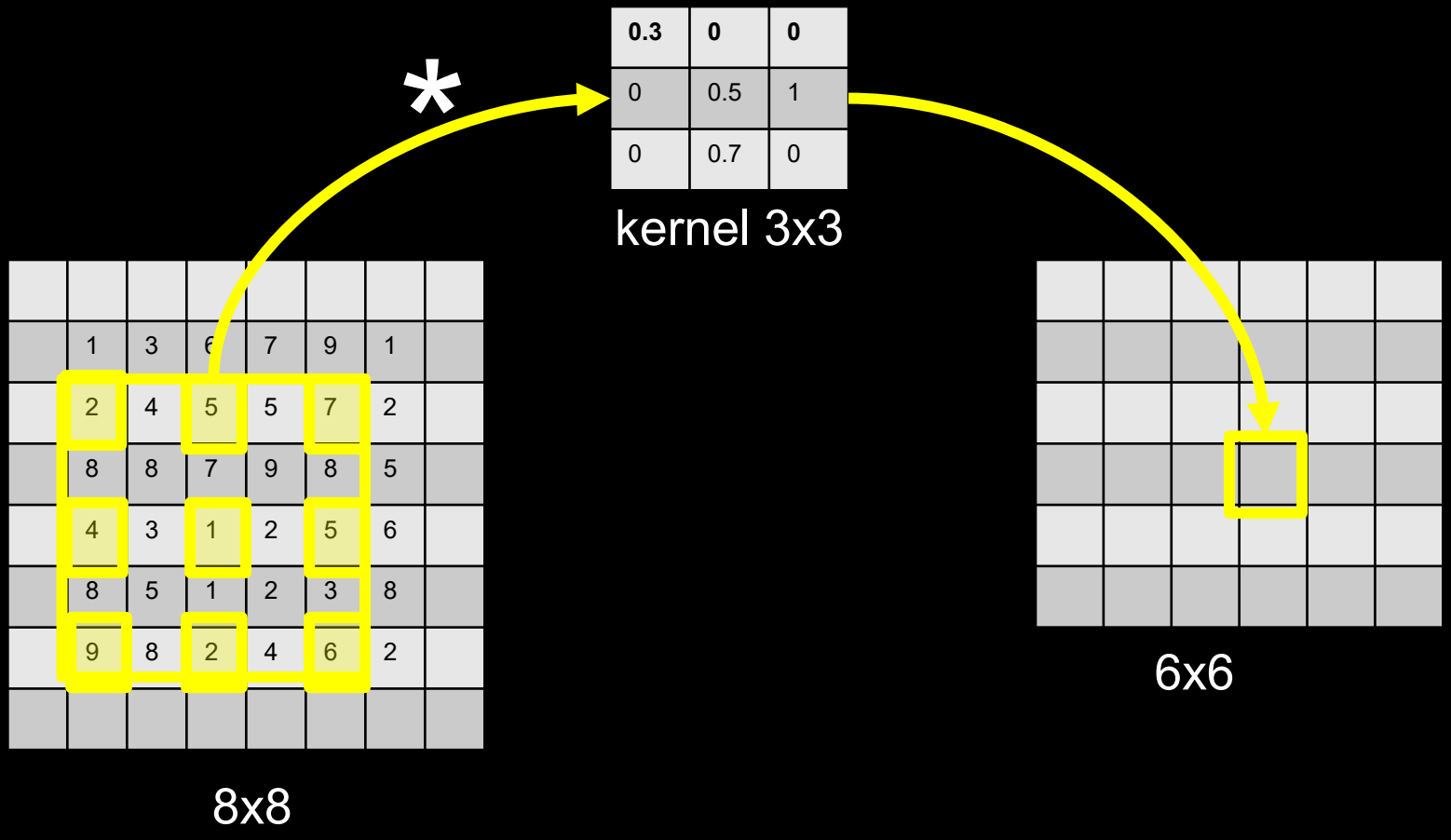


Kernel convoluzionale - padding



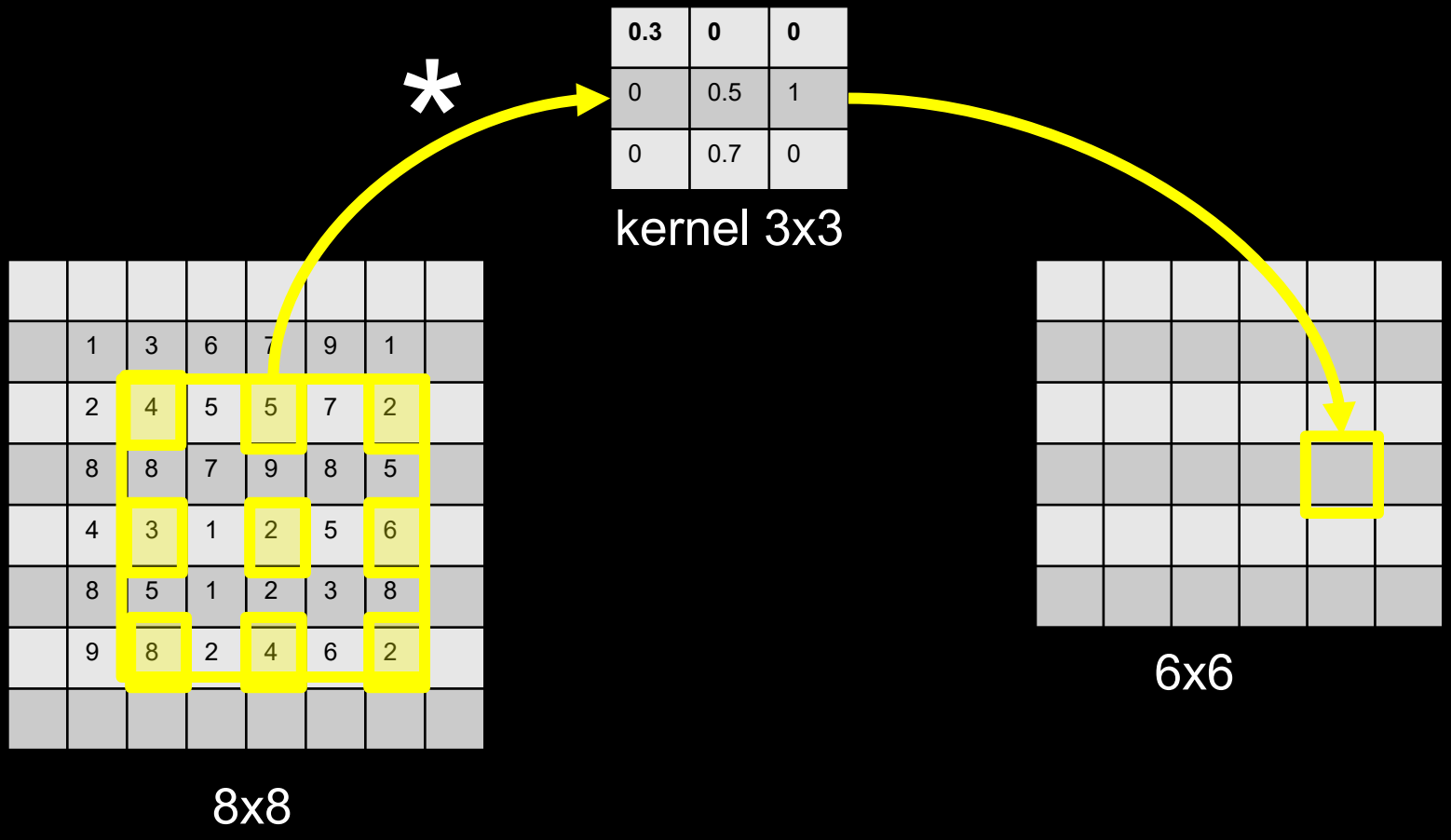
per evitare gli effetti di bordo si possono usare dei padding contenenti 0, il valore del padding e la sua ampiezza rientra negli hyperparameter da impostare

Kernel convoluzionale - dilatation



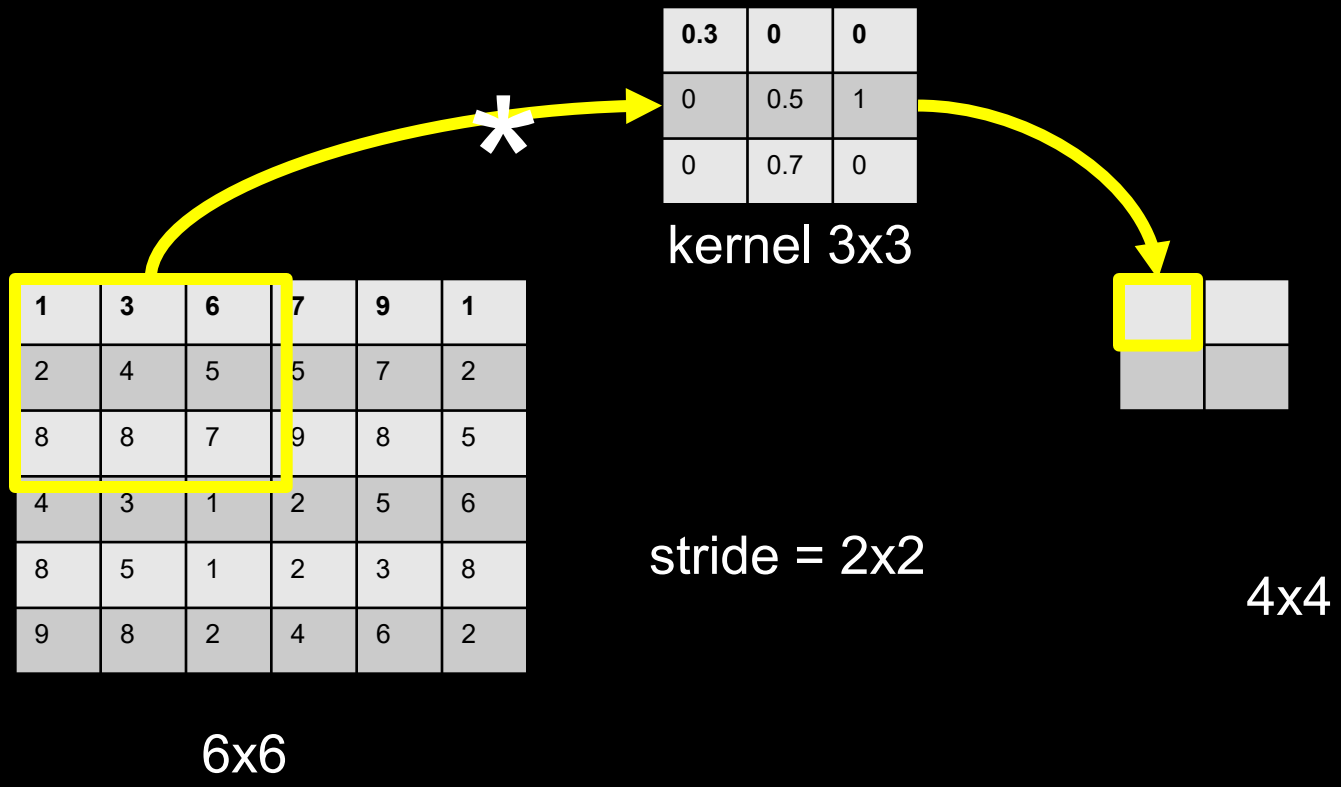
la dilatazione può essere usata in convoluzione per mixare informazioni tra pixel non attigui

Kernel convoluzionale - dilatation



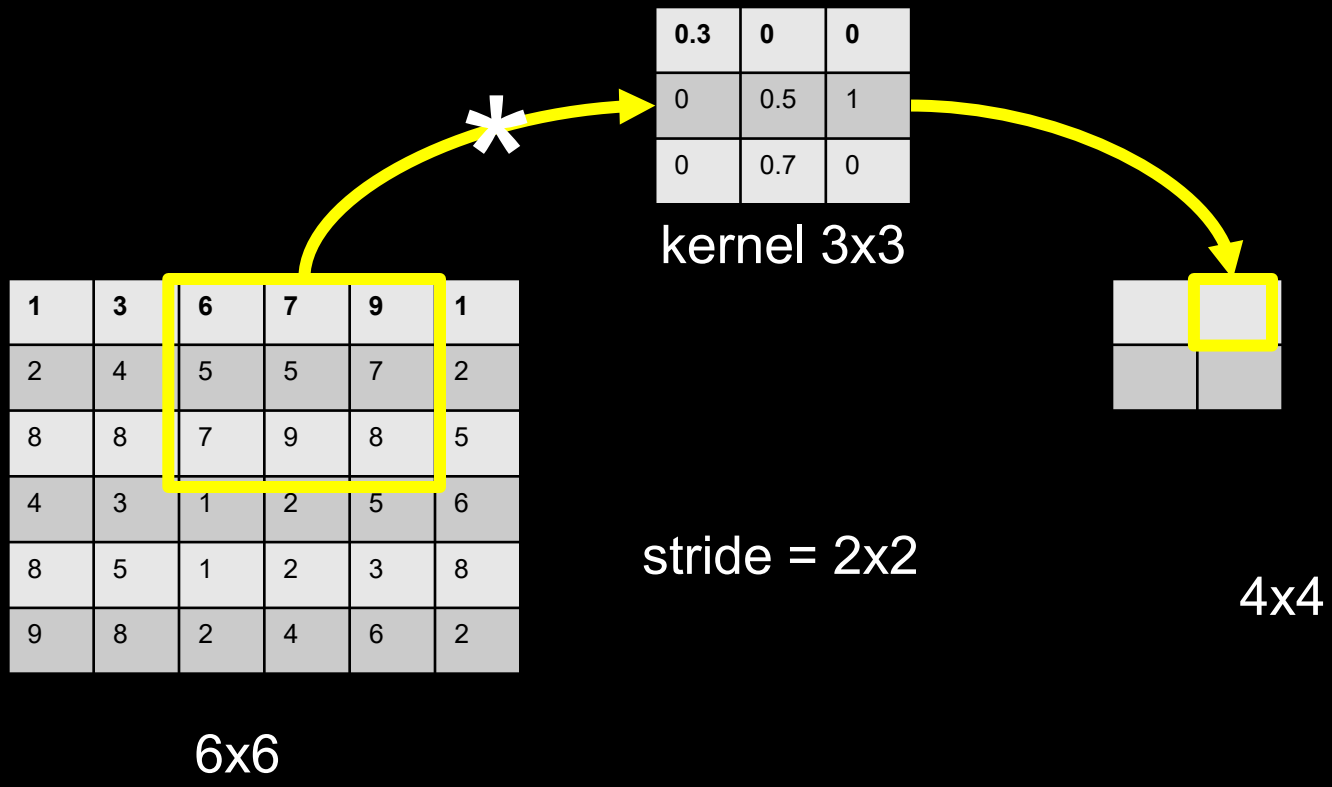
la dilatazione può essere usata in convoluzione per mixare informazioni tra pixel non attigui

Kernel convoluzionale - striding



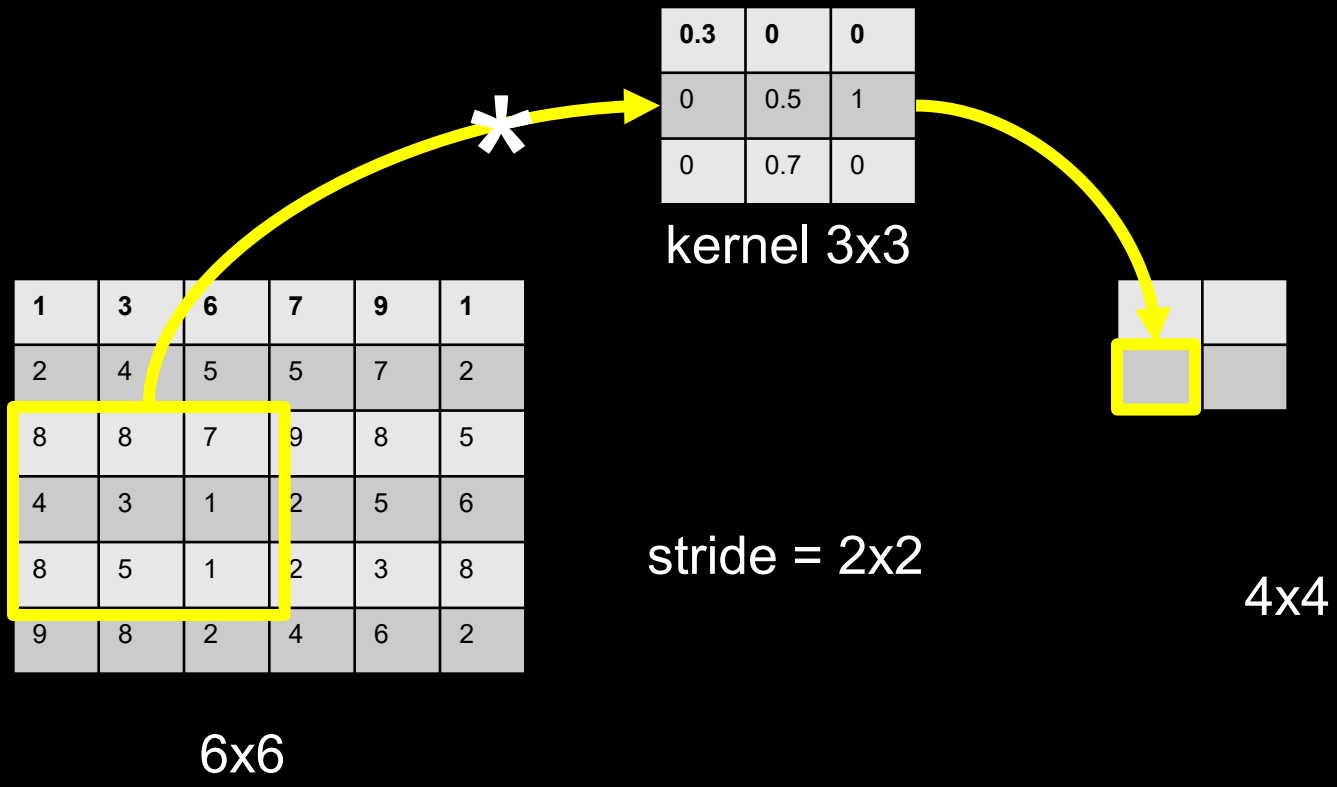
Al fine di comprimere l'informazione possiamo effettuare operazioni di striding, quindi scorrere saltando un certo numero di pixels, l'immagine si comprime

Kernel convoluzionale - striding



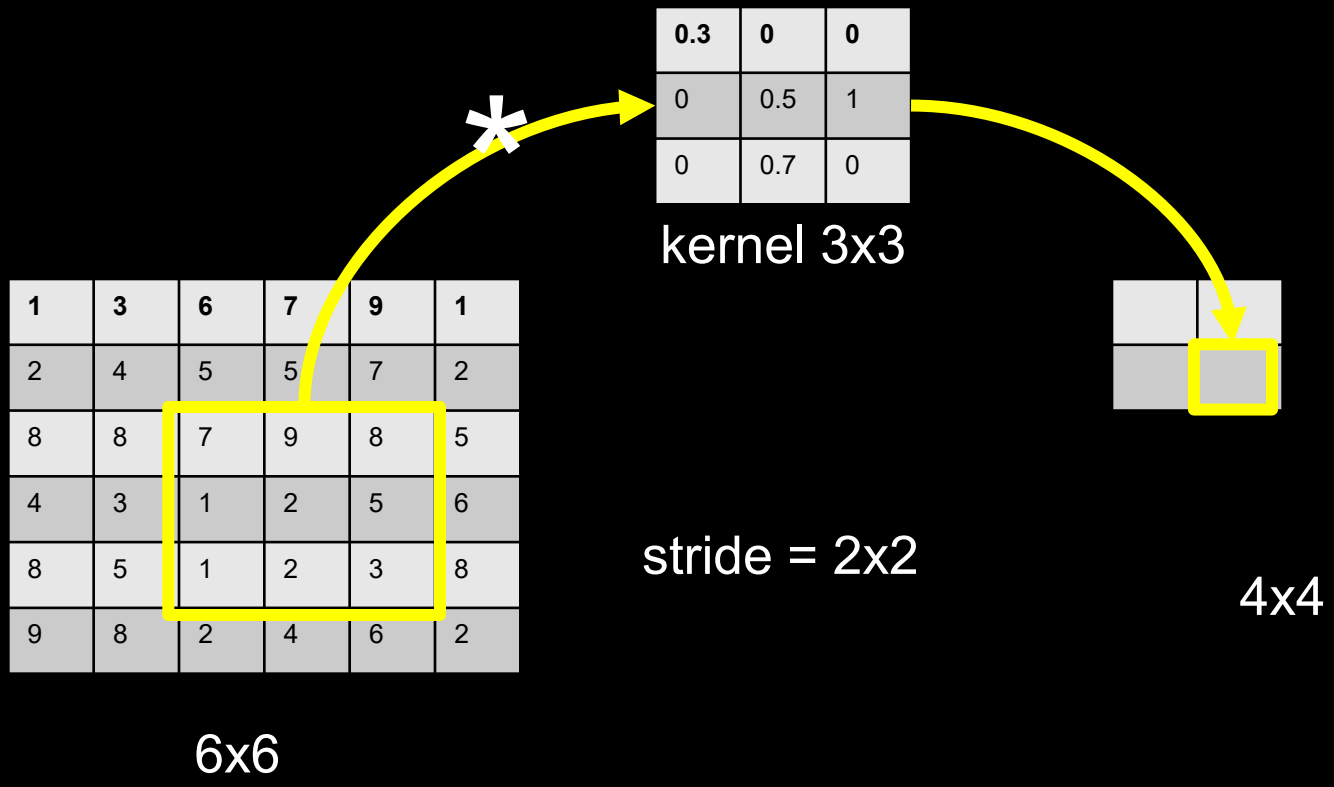
Al fine di comprimere l'informazione possiamo effettuare operazioni di striding, quindi scorrere saltando un certo numero di pixels, l'immagine si comprime

Kernel convoluzionale - striding



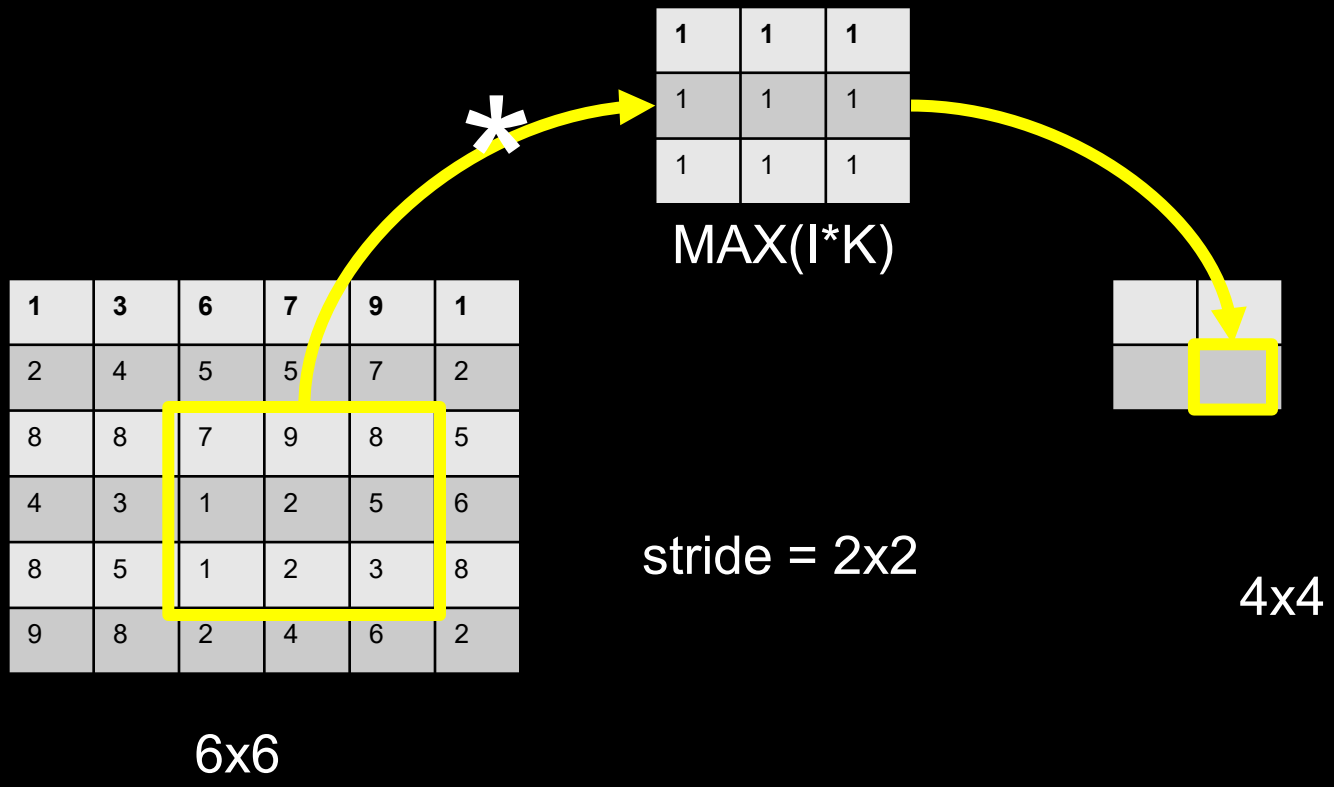
Al fine di comprimere l'informazione possiamo effettuare operazioni di striding, quindi scorrere saltando un certo numero di pixels, l'immagine si comprime

Kernel convoluzionale - striding



Al fine di comprimere l'informazione possiamo effettuare operazioni di striding, quindi scorrere saltando un certo numero di pixels, l'immagine si comprime

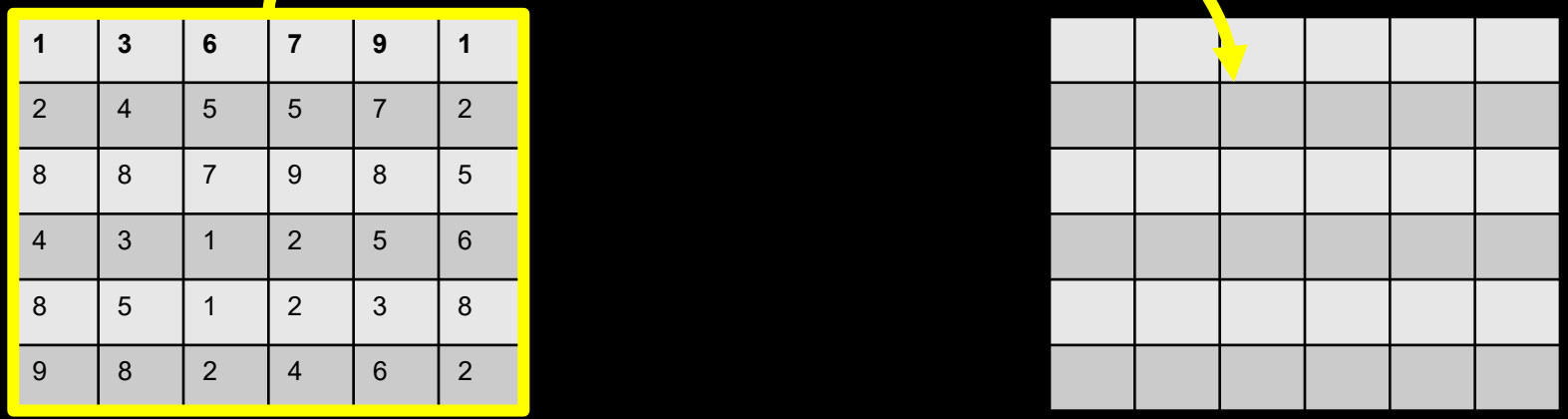
Pooling layers



i pooling usano una convoluzione con dei kernel unitari e ritornano solo una specifica funzione della finestra es. massimo, minimo, media etc.

Batch normalization

$$I_{stand} = \frac{I - E[I]}{\sigma[I]}$$



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

i livelli di normalizzazione del batch accelerano la convergenza delle reti attraverso operazioni di scalatura, normalizzazione e ricentramento dei valori su ogni livello

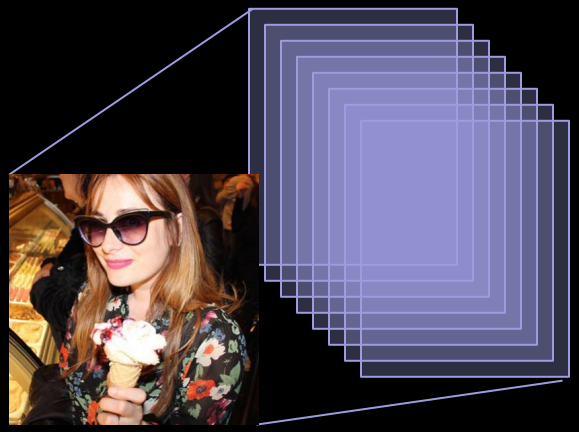
Reti convoluzionali

Esempio da:

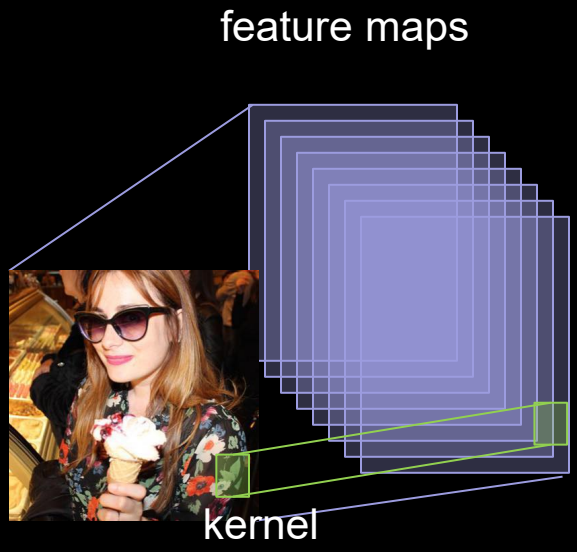
- https://cs231n.github.io/convolutional-networks/

Approccio convoluzionale alexNet

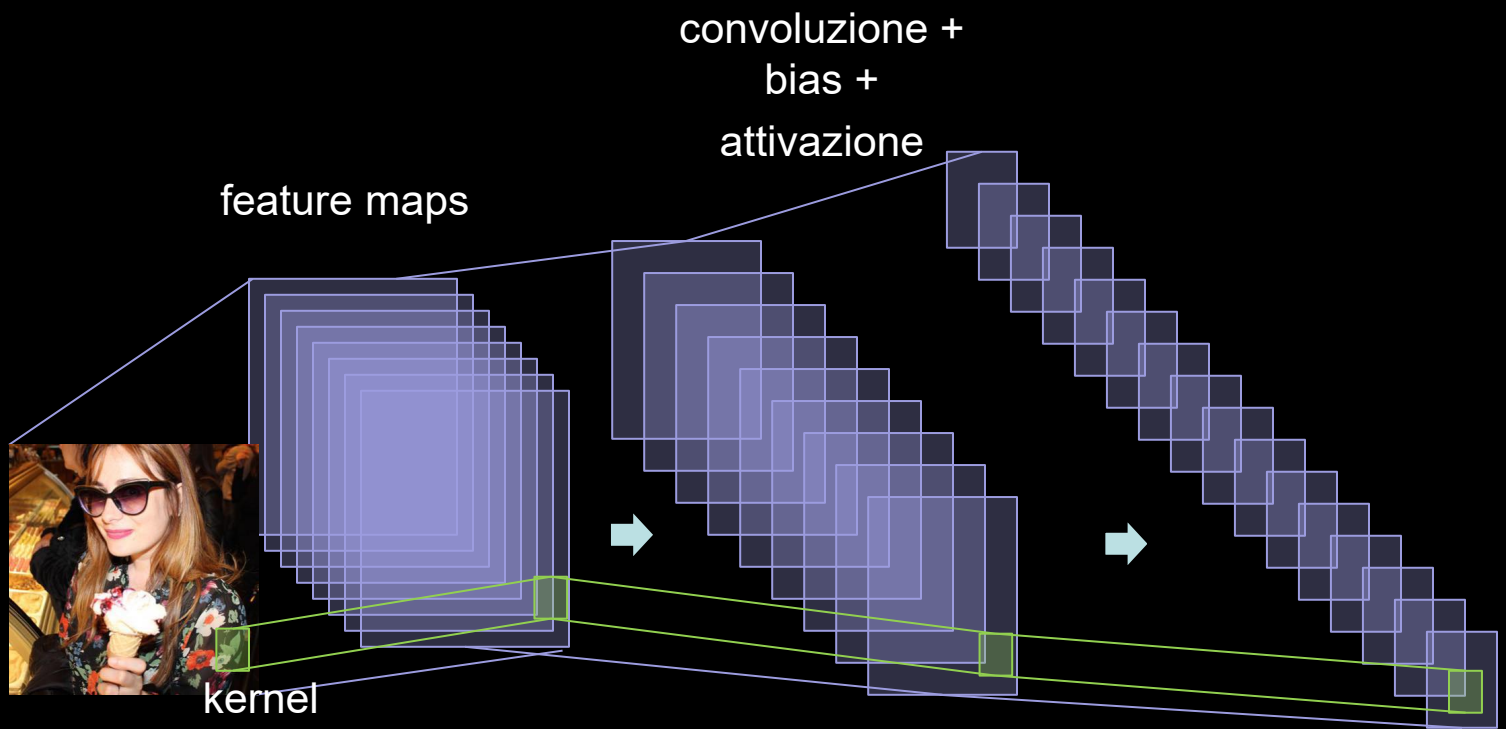
feature maps



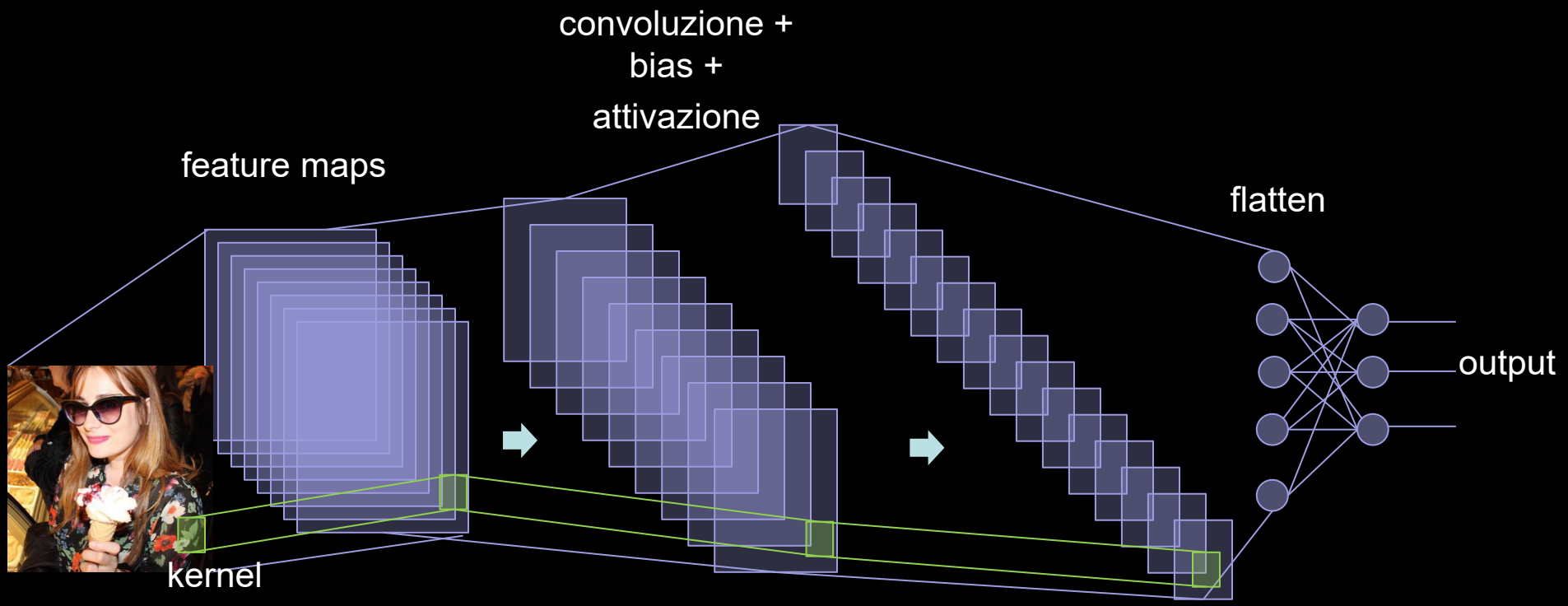
Approccio convoluzionale alexNet



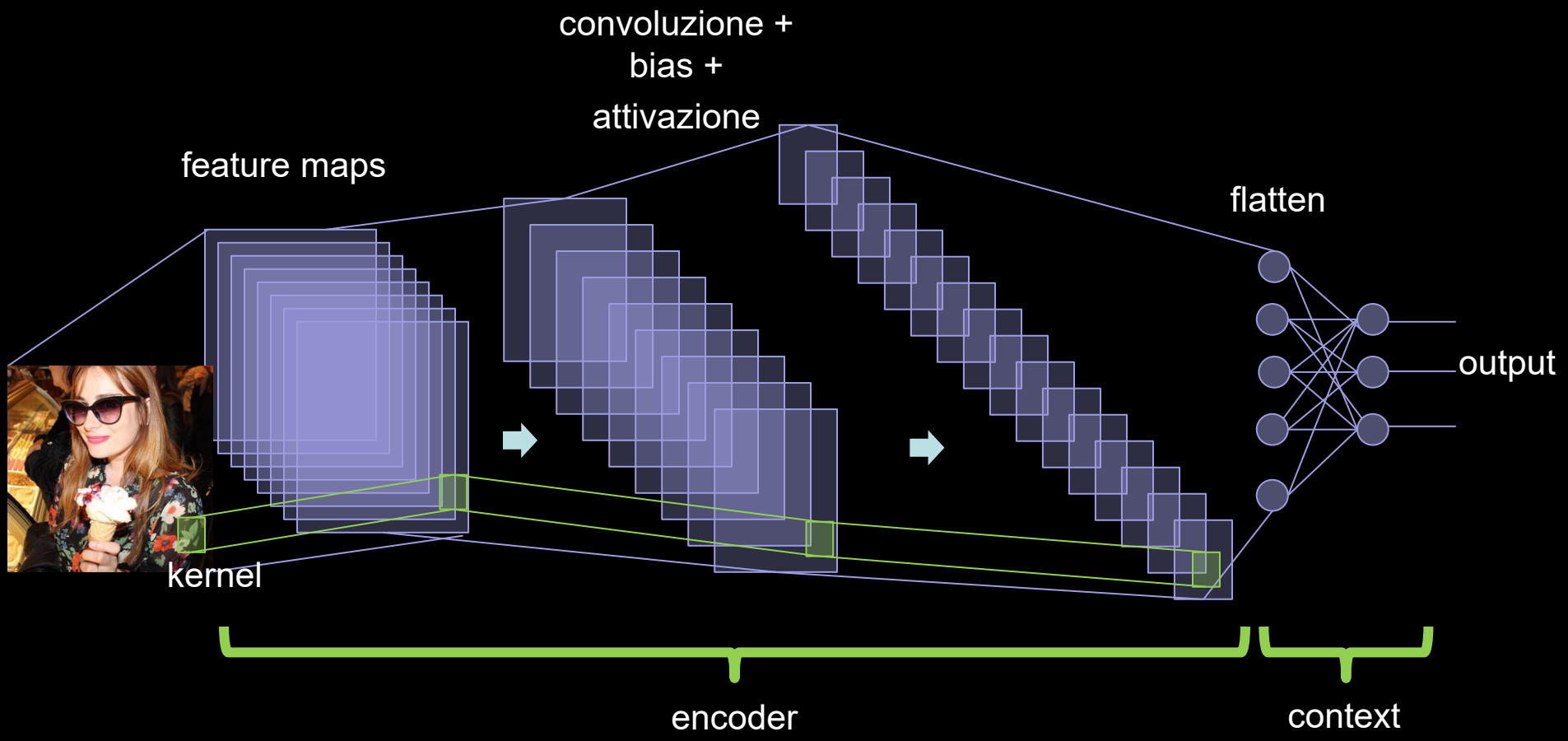
Approccio convoluzionale alexNet



Approccio convoluzionale alexNet



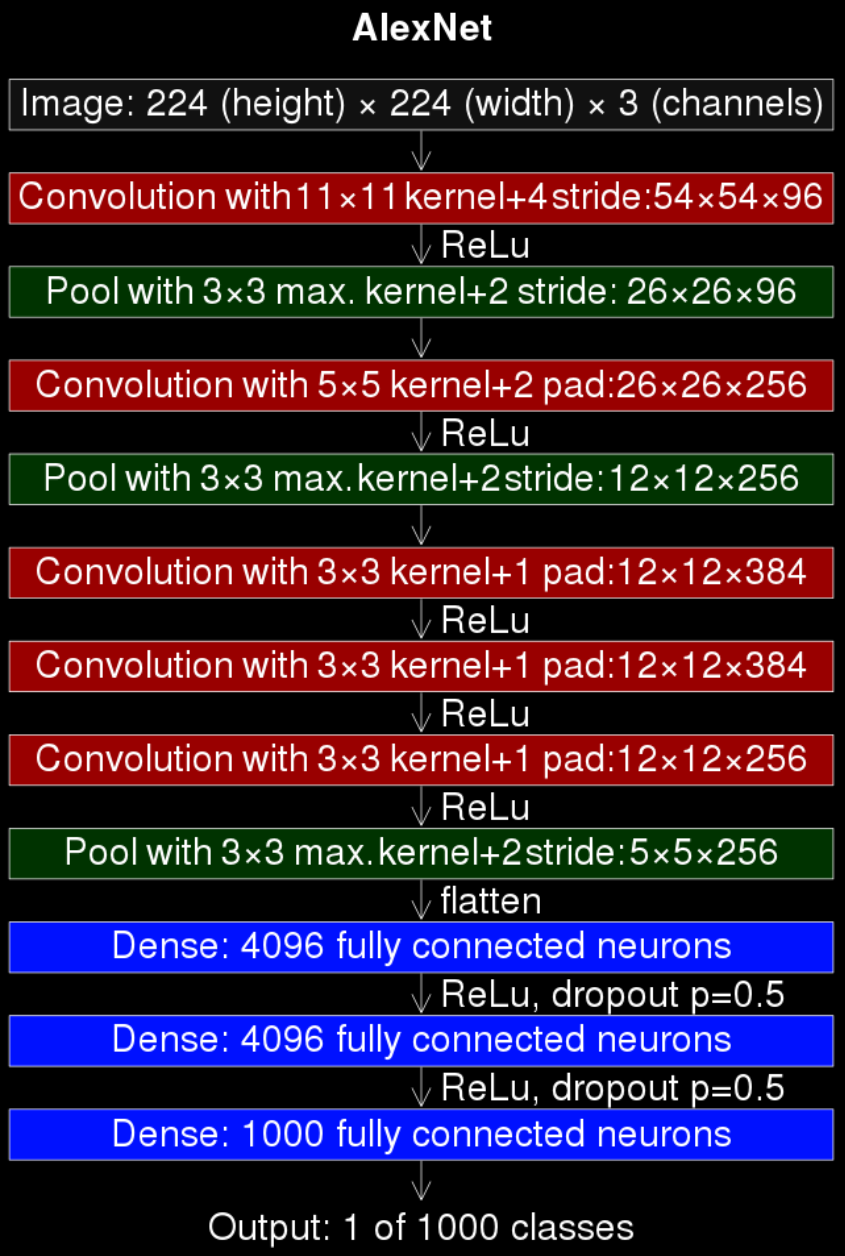
Approccio convoluzionale alexNet



AlexNet

Approccio con feature extraction

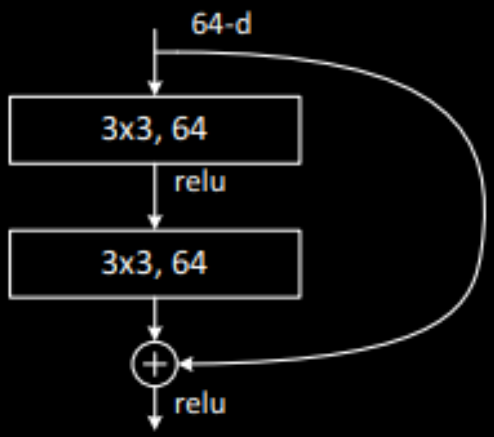
- ✓ kernel decrescenti – da 11x11 a 3x3
- ✓ polling
- ✓ 2x stride

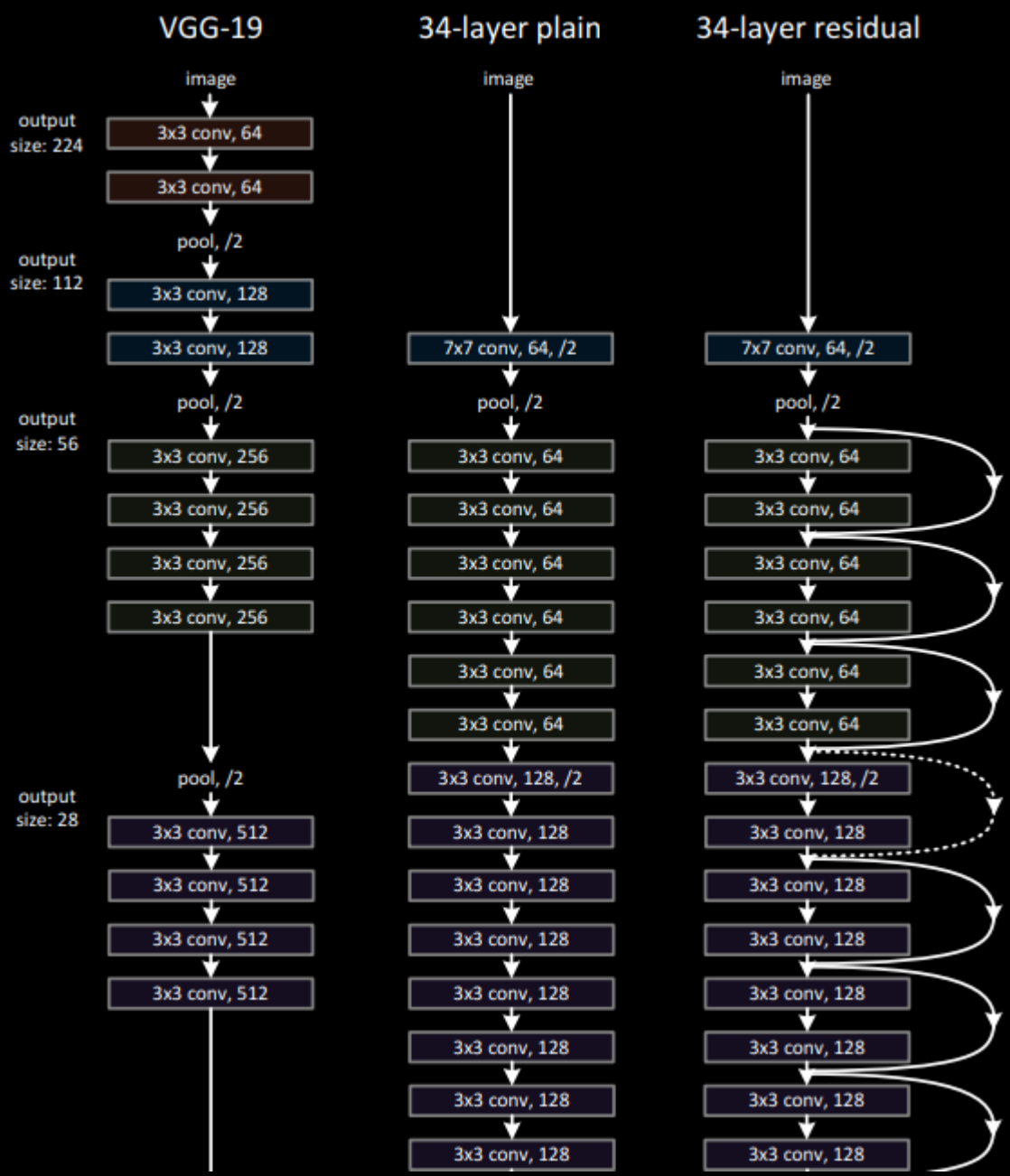


source: https://en.wikipedia.org/wiki/AlexNet

Strutture reti

- ✓ ResNet
- ✓ VGG
- ✓ U-Net

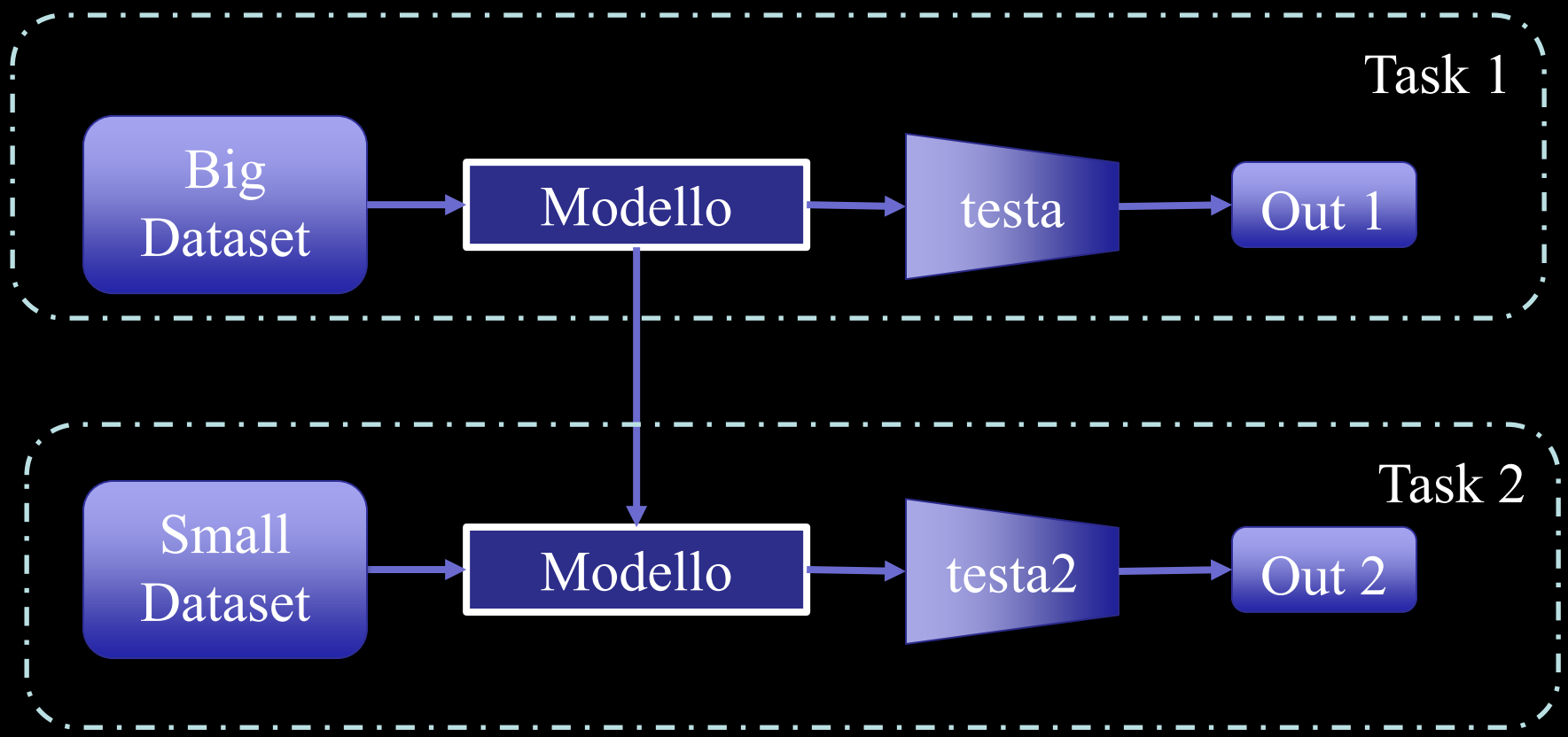




source: He, K., Zhang, X., Ren, S. and Sun, J., 2016. Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778).

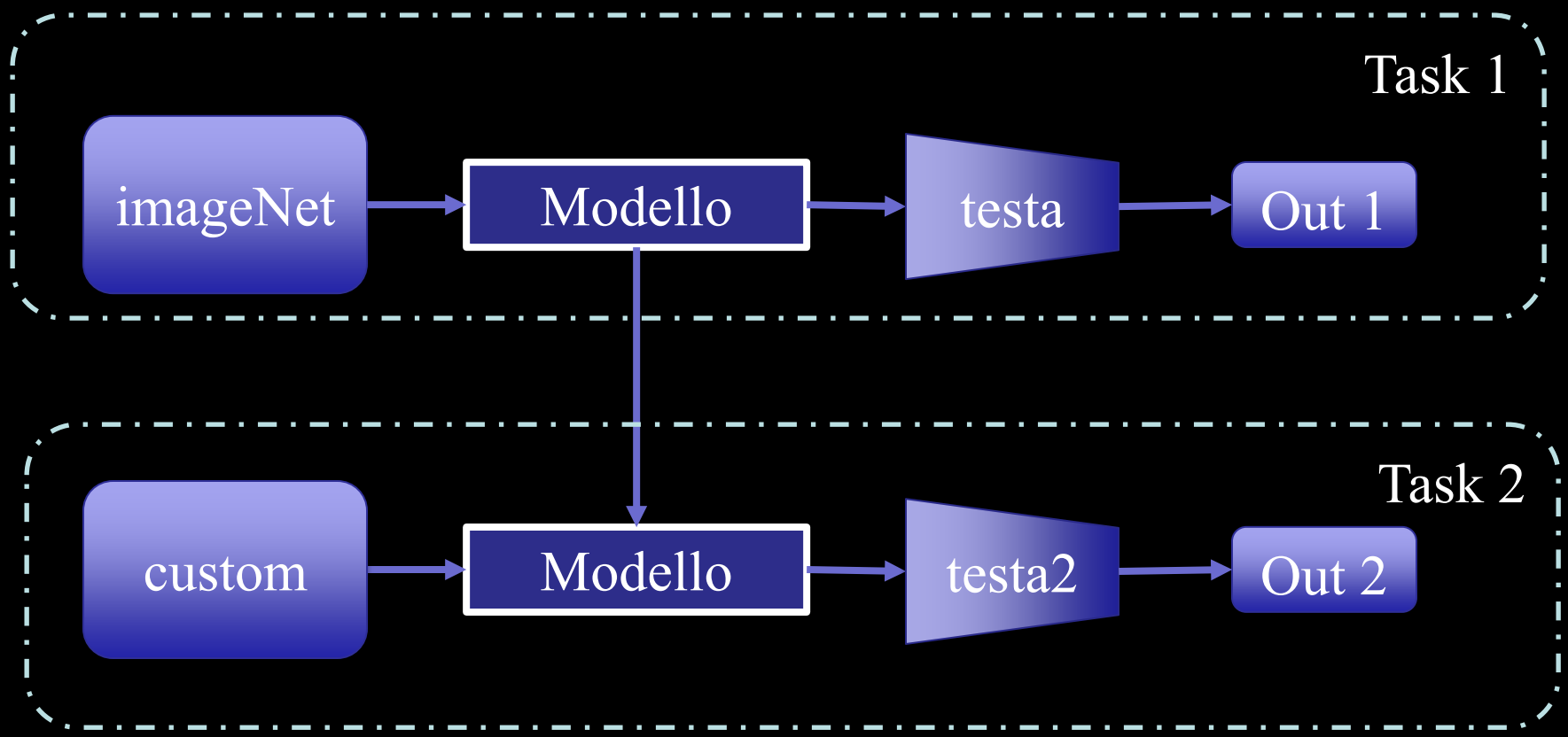
Transfer learning

Le inizializzazioni dei pesi possono essere fatte in maniera random oppure usando un altro modello pre-addestrato su dati diversi

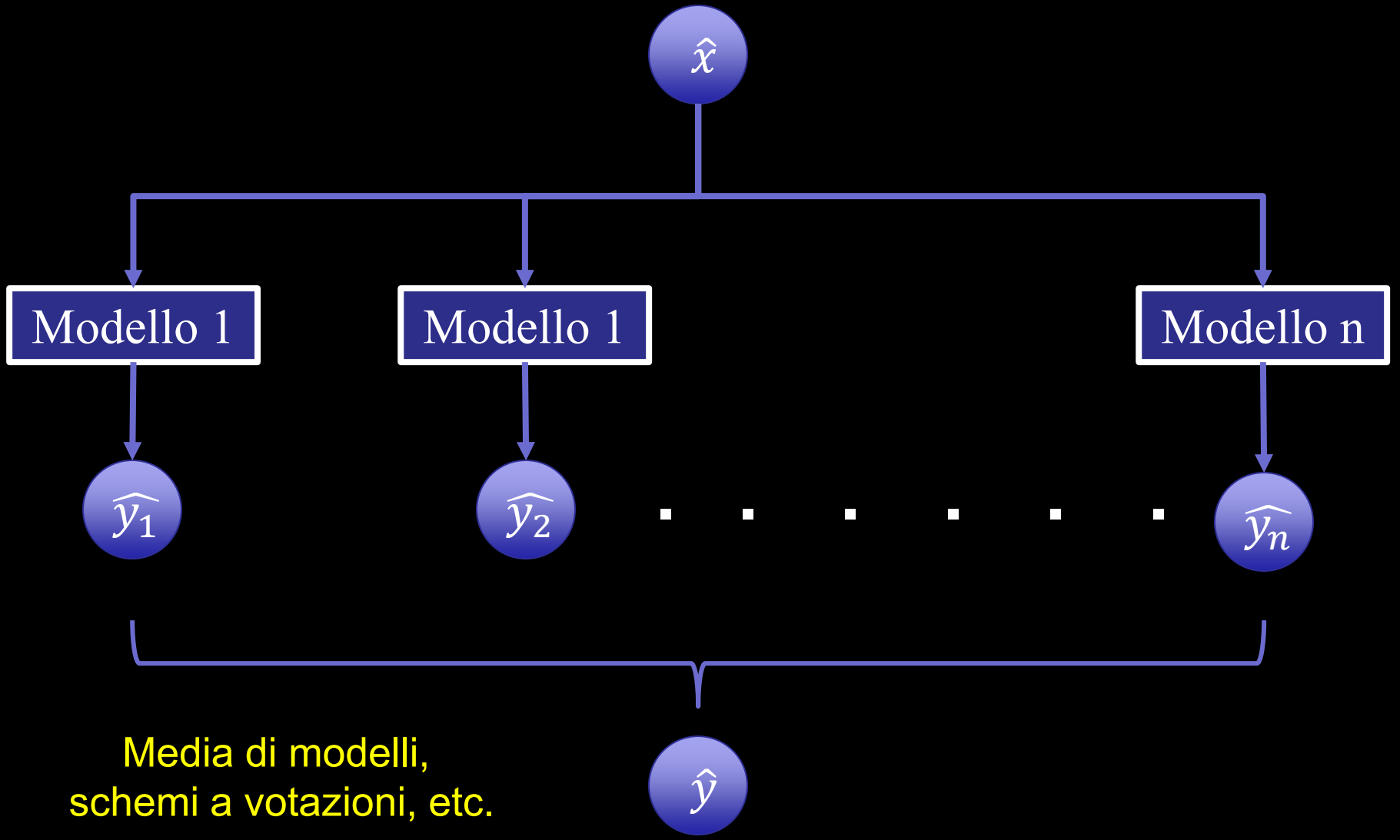


Transfer learning

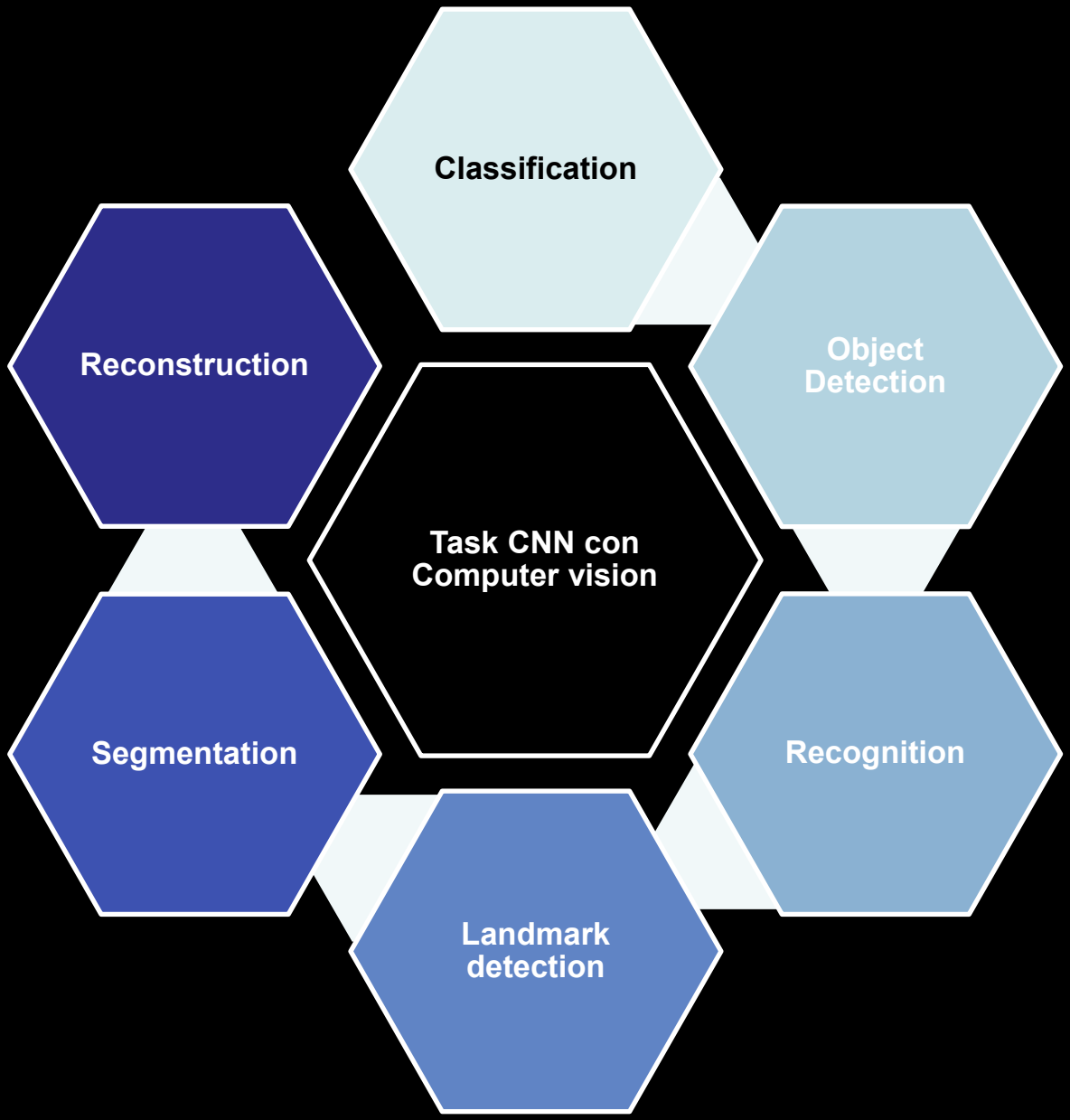
Le inizializzazioni dei pesi possono essere fatte in maniera random oppure usando un altro modello pre-addestrato su dati diversi



Ensamble learning



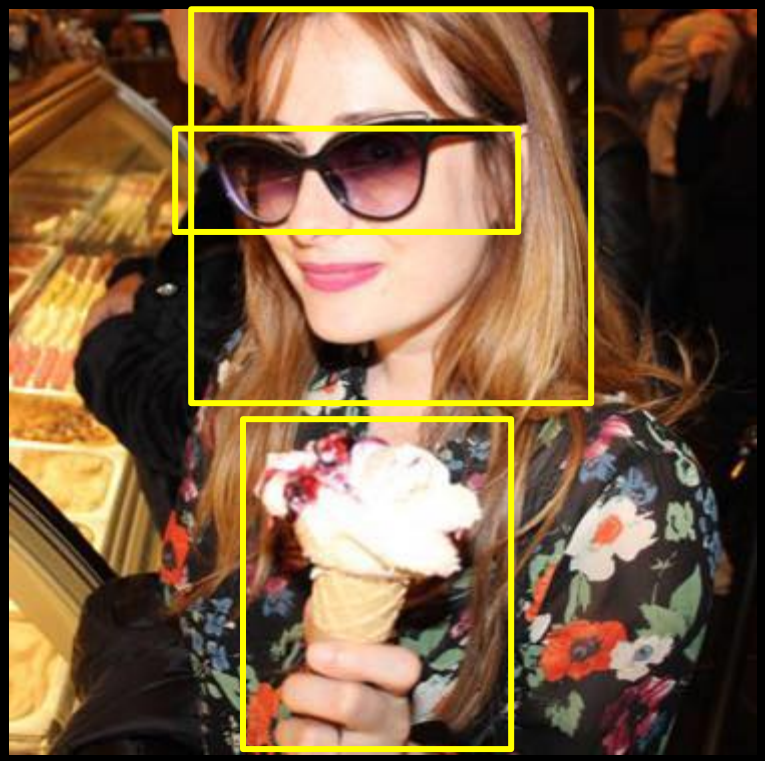
Media di modelli,
schemi a votazioni, etc.


Classification
Reconstruction
Object Detection
Task CNN con Computer vision
Segmentation
Recognition
Landmark detection

Classificazione

Gelato

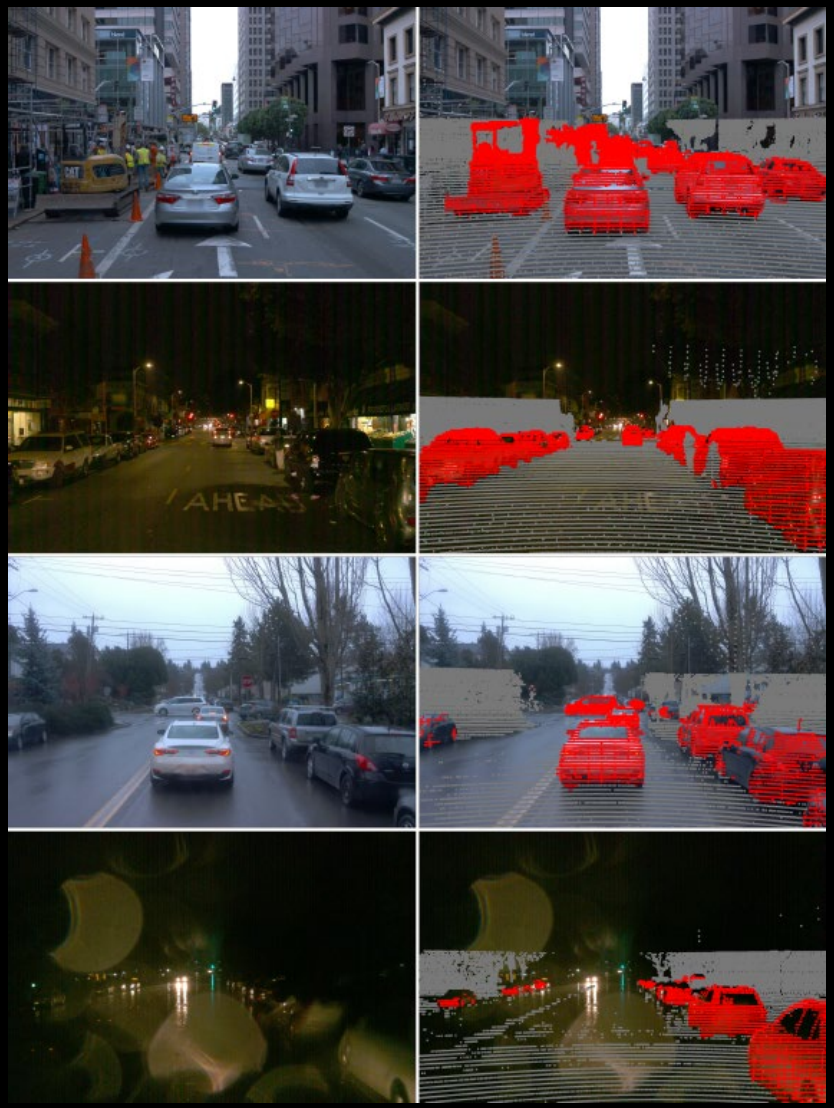
Detection



Recognition

Landmark recognition

Segmentazione



source: L. Caltagirone, M. Bellone, L. Svensson, M. Wahde, and R. Sell - "Lidar–Camera Semi-Supervised Learning for Semantic Segmentation" Sensors, 2021 Vol. 21, issue no. 14:4813 https://doi.org/10.3390/s21144813

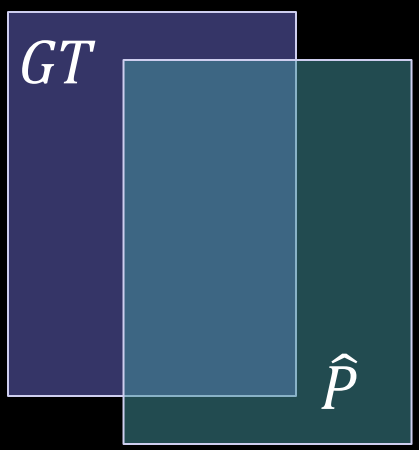
Intersection over union

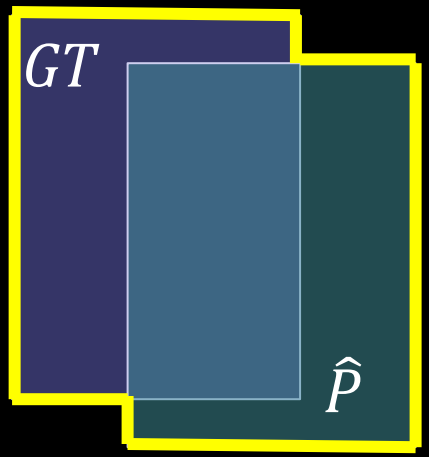
Metrica usata tipicamente nei task di segmentazione o detect definita come il rapporto tra l'area dell'intersezione e dell'unione tra la ground truth e la stima

GT = ground truth

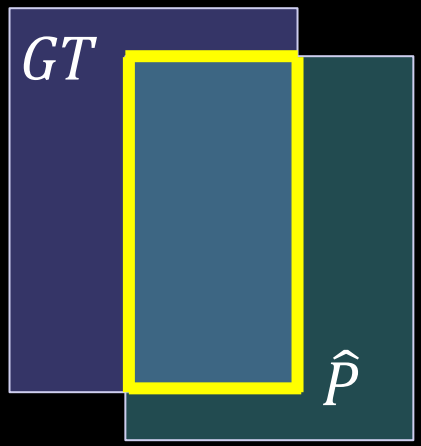
$\hat{P}$ = area stimata

$$IoU = \frac{A_{GT \cap \hat{P}}}{A_{GT \cup \hat{P}}}$$



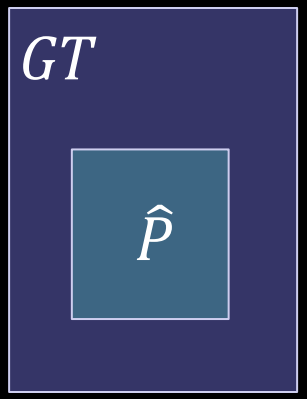


unione



intersezione

Intersection over union

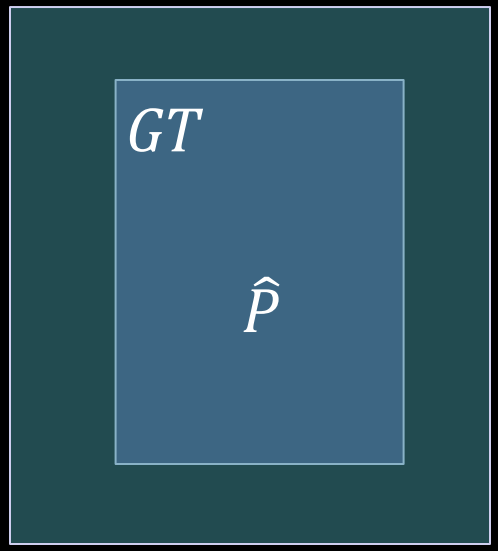


$IoU \ll 1$

$IoU = 0$

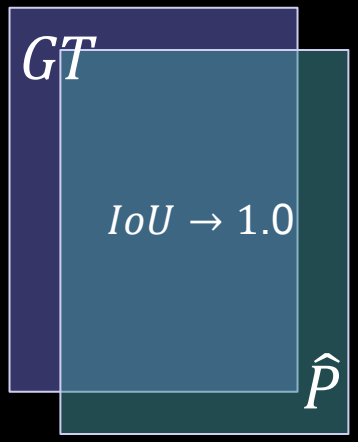
Nessuna sovrapposizione

Area unione > Area intersezione

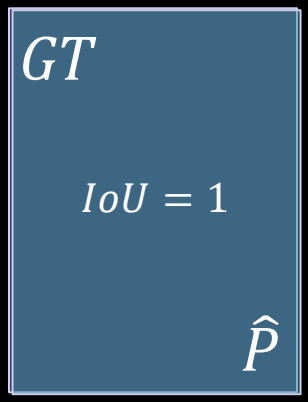


$IoU \ll 1$

Intersection over union

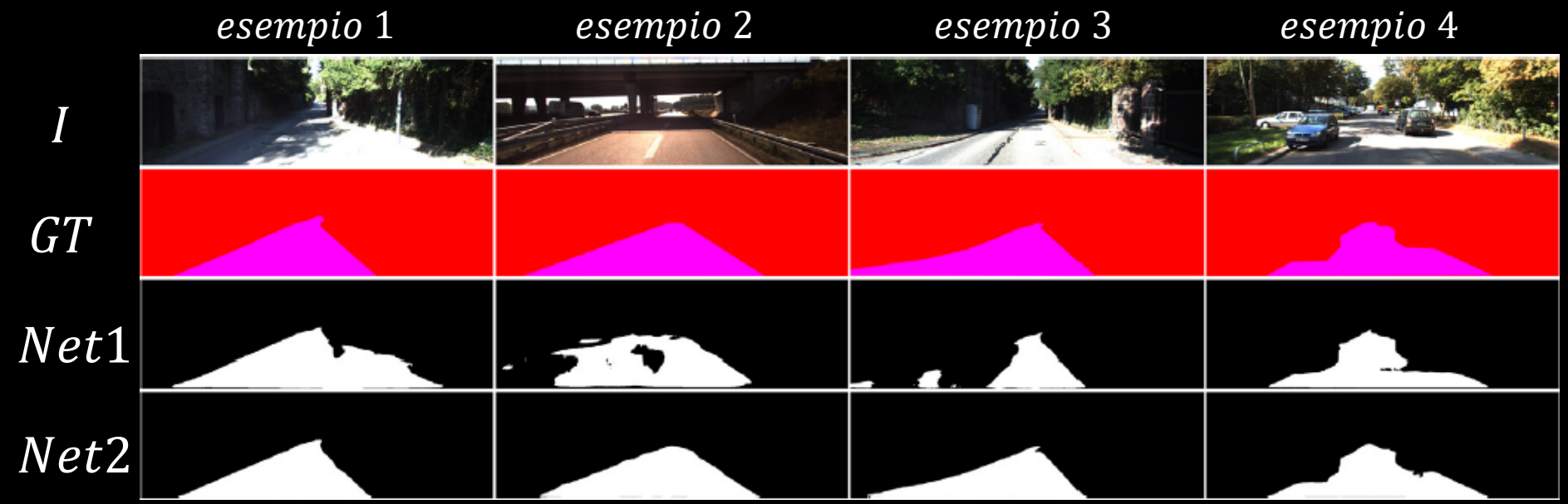


Discreta sovrapposizione



Sovrapposizione perfetta

Intersection over union - esempio



source: Caltagirone, L., Bellone, M., Svensson, L. and Wahde, M., 2019. LIDAR–camera fusion for road detection using fully convolutional neural networks. Robotics and Autonomous Systems, 111, pp.125-131.