
# Deep Learning

Con  il termine DNN (Deep Neural Network) si denotano reti « profonde » composte da molti livelli (almeno 2 hidden ) organizzati gerarchicamente

Le DNN oggi maggiormente utilizzate consistono di un numero di livelli compreso tra 7 e 50

Reti più profonde (100 livelli e oltre) hanno dimostrato di poter garantire prestazioni leggermente migliori, a discapito però dell'efficienza

# Principali tipologie di DNN

**Modelli feedforward « discriminativi » per la classificazione (o regressione) con training prevalentemente supervisionato**

- **FC-DNN: Fully Connected DNN** (MLP con almeno due livelli hidden)
- **CNN:** Convolutional Neural Network (o ConvNet)

**Modelli ricorrenti con memoria e attenzione (utilizzati per sequenze, speech recognition natural language processing)**

- **RNN**
- **Recurrent Neural Network**
- **LSTM**
- **Long Short Term Memory**
- **Transformers**

**Addestramento non supervisionato: modelli addestrati a ricostruire l'input originale prendendo come input una versione a più bassa dimensionalità ( utilizzati per denoising, anomaly detection)**

- **Autoencoders**

**Modelli « generativi » per generare dataset sintetici (data augmentation) style transfer, art applications**

- GAN Generative Adversarial Networks
- VAE Variational Autoencoders

**Reinforcement learning (per apprendere comportamenti)**

- **Deep Q Learning**

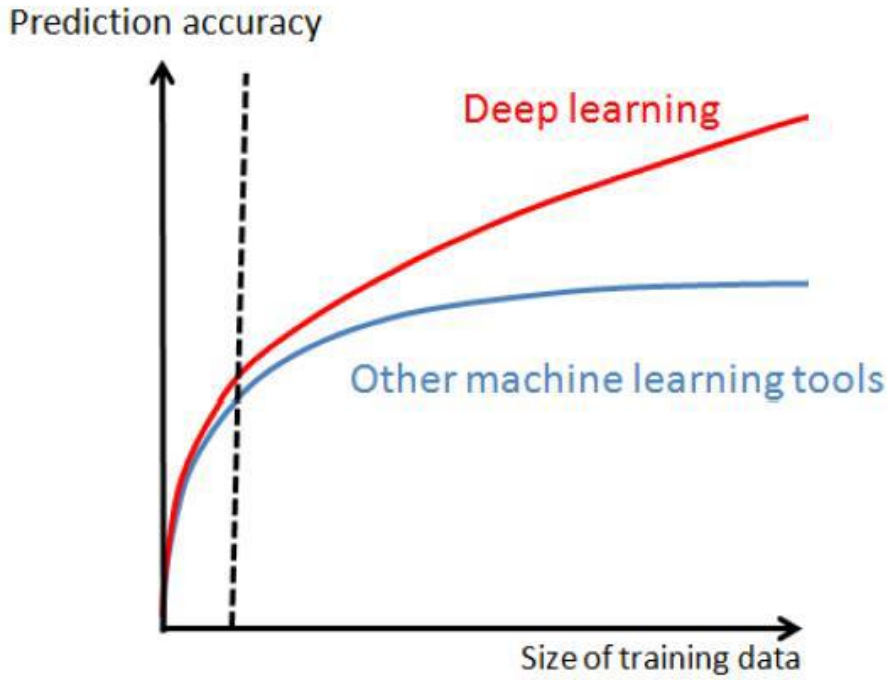
# Ingredienti necessari

**BigData**

disponibilità di dataset etichettati di grandi dimensioni (es ImageNet milioni di immagini, decine di migliaia di classi)

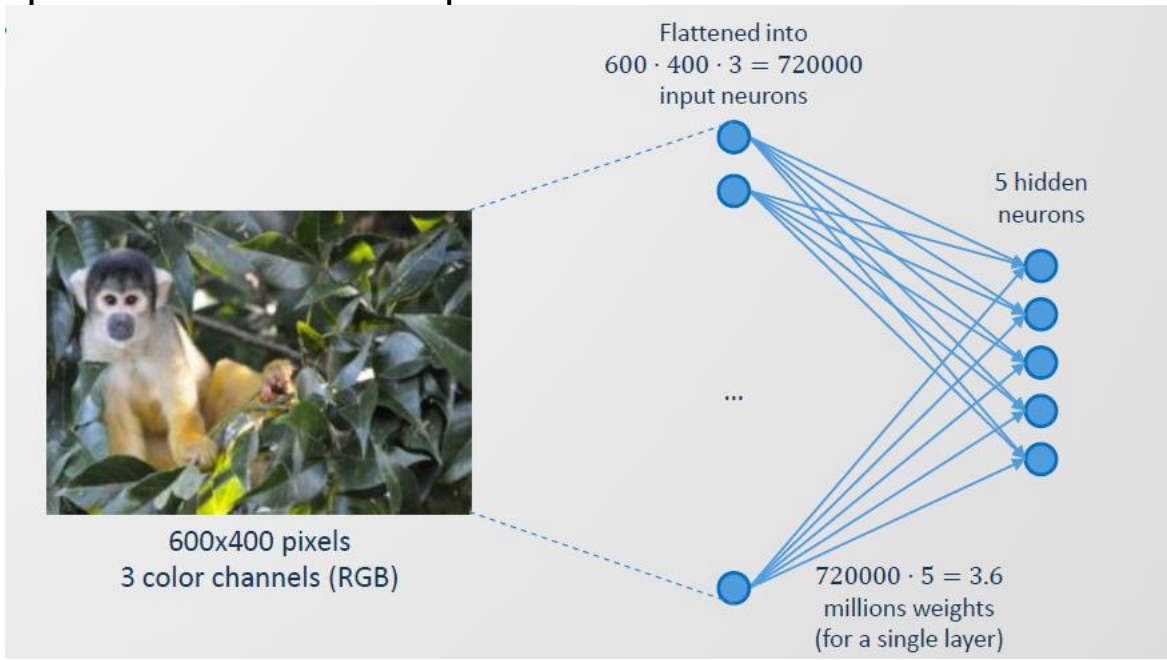
**GPU computing :**

il training di modelli complessi (profondi e con molti pesi e connessioni) richiede elevate potenze computazionali La disponibilità di GPU con migliaia di core e GB di memoria interna ha consentito di ridurre drasticamente i tempi di training da mesi a giorni.




La superiorità delle tecniche di deep learning rispetto ad altri approcci si manifesta quando sono disponibili grandi quantità di dati di training.

- Le reti di tipo MLP sono computazionalmente troppo pesanti per essere impiegate **nell'elaborazione di immagini**.
- L'immagine viene appiattita e non si tiene conto della sua struttura 2D, **si prevede un neurone per ogni pixel che è connesso con tutti i neuroni del layer successivo.**
- Poiché ciascun neurone viene connesso a tutti i neuroni del layer successivo, il numero di parametri da stimare sarebbe eccessivo per i casi di interesse pratico..



Le reti MLP **non hanno alcuna invarianza per traslazione**.

Una MLP reagirà in **modo diverso ad un'immagine ed ad una sua versione traslata**, anche se le immagini sono visivamente simili. Ciò è dovuto al fatto che la posizione dei pixel nell'immagine è un fattore importante per determinare l'output della rete.

Le informazioni spaziali più importanti vengono perse quando l'immagine viene appiattita in un MLP. Conoscere le relazioni di vicinanza tra i pixel è importante perché aiuta a definire le caratteristiche di un'immagine.

Le **Convolutional Neural Network (CNN)** sono progettate per essere invarianti alla traslazione e ciò le rende più efficaci nell'elaborazione delle immagini. Le CNN **utilizzano strati convoluzionali** che applicano filtri all'immagine, consentendo alla rete di identificare le caratteristiche indipendentemente dalla loro posizione nell'immagine. Inoltre, vengono utilizzati strati di pooling per ridurre la dimensione dell'output degli strati convoluzionali, aumentando ulteriormente l'invarianza alla traslazione della rete.

Le reti neurali convoluzionali (CNN) sono reti profonde (deep) ispirate alle ricerche biologiche di Hubel e Wiesel durante lo studio del cervello dei gatti.

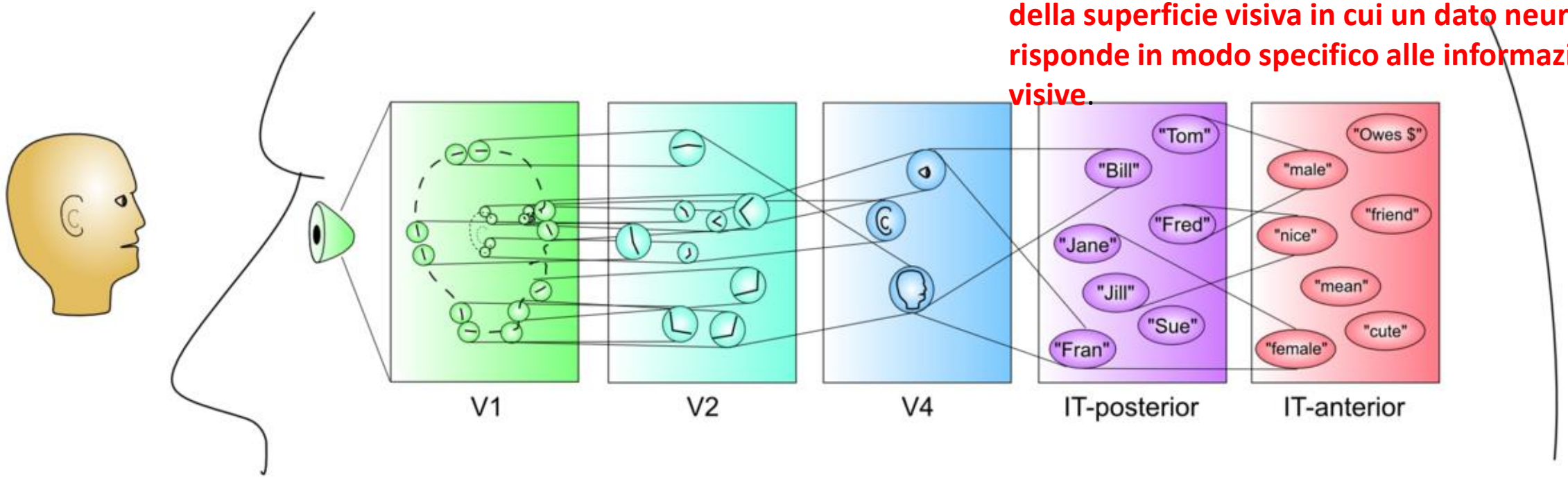
## Corteccia visiva

- Nel 1965 DH Hubel e TN Wiesel hanno dimostrato che i mammiferi percepiscono visivamente il mondo che li circonda utilizzando un'architettura a strati di neuroni nel cervello.

- La struttura della **corteccia visiva è a strati. Man mano che le informazioni passano dai nostri occhi al cervello, si formano rappresentazioni di ordine sempre più alto.**

# VISIONE UMANA

Il **campo recettivo** di un neurone visivo è **l'area della superficie visiva in cui un dato neurone risponde in modo specifico alle informazioni visive**.

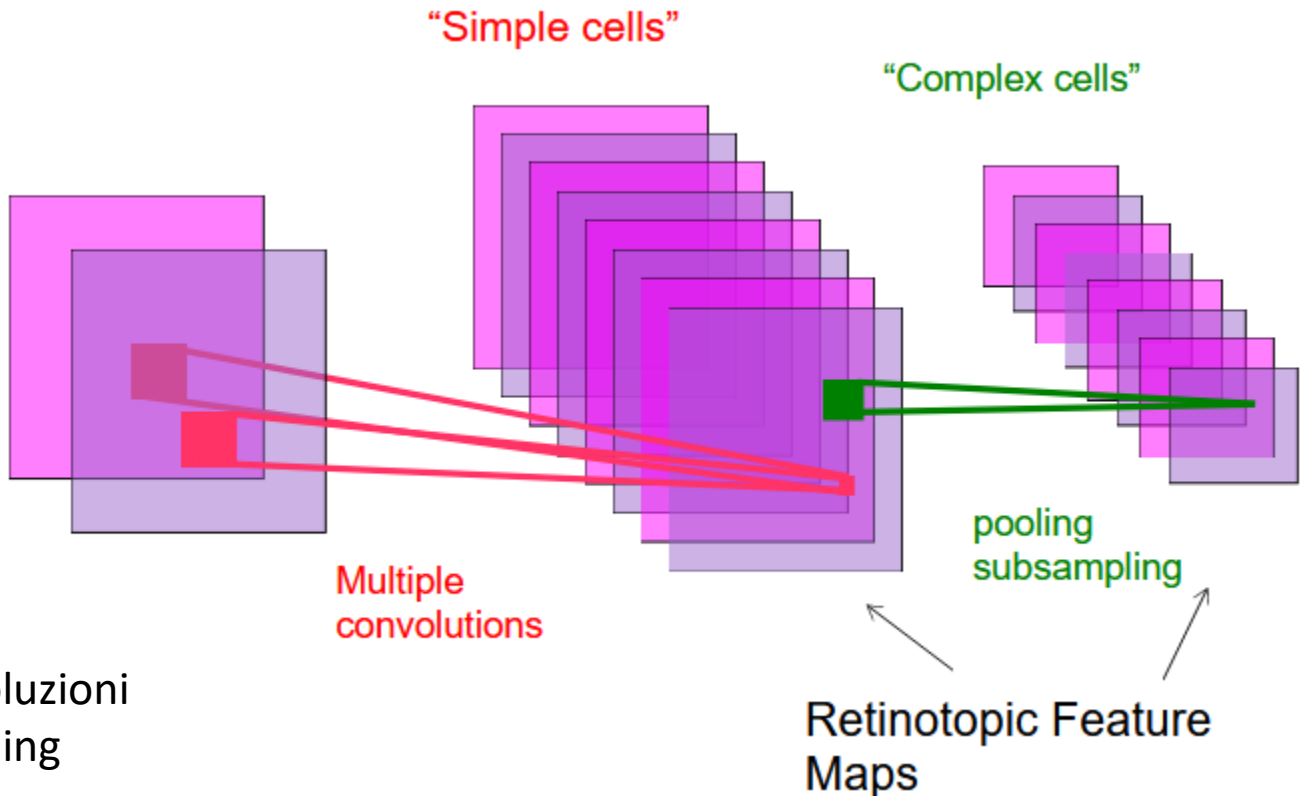



- Il sistema visivo umano elabora le caratteristiche dell'oggetto in un **approccio feed-forward e gerarchico.**
- **La gerarchia inizia dalla Corteccia Visiva Primaria (V1), caratterizzata da neuroni con campi recettivi piccoli e specializzati, elabora le informazioni visive di base, bordi e linee.**
- **Queste informazioni vengono poi elaborate dalle Aree visive (V2 e V4),** specializzate nella percezione di informazioni visive più complesse, come forme e oggetti (e sono caratterizzate da neuroni con campi recettivi più estesi).
- Infine, le informazioni visive vengono elaborate dalla **corteccia inferotemporale (IT),** che è specializzata nella percezione di oggetti complessi, come il riconoscimento facciale (campi recettivi più ampi e completi)

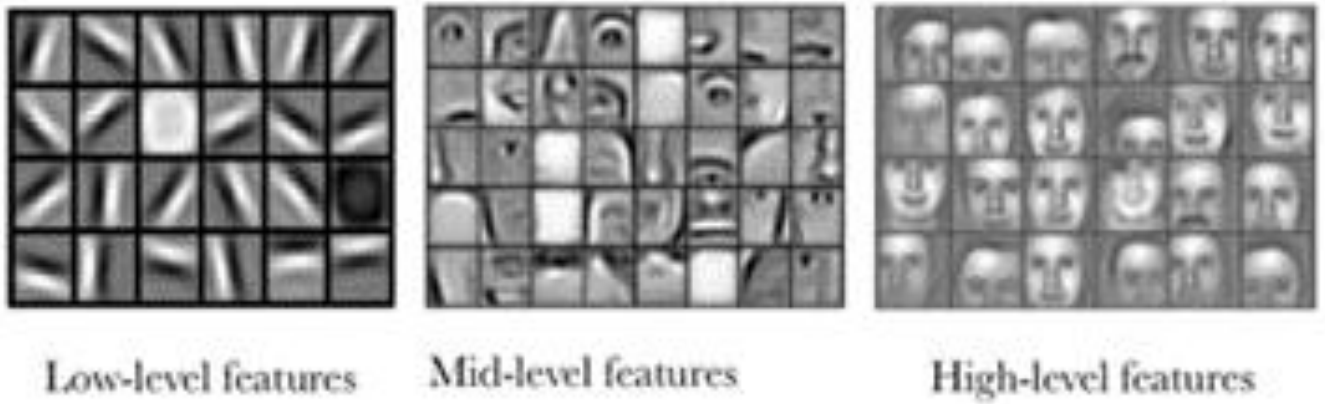
# [Hubel & Wiesel 1962]:

**Due tipologie di neuroni nella corteccia visiva del gatto**

- **celle semplici** : specializzate nella rilevazione di caratteristiche locali dell'input visivo (feature extractor),
- **celle complesse :** specializzate nell'integrazione (pooling) delle informazioni provenienti da diverse posizioni retinotopiche per formare una rappresentazione globale dell'input visivo, preservando le caratteristiche invarianti per posizione.




Celle semplici → Convoluzioni
Celle complesse -> Pooling




- Come il sistema visivo umano, le CNN sono composte da una serie di strati o livelli di elaborazione, ognuno dei quali svolge una particolare funzione nell'elaborazione dell'immagine di input.
- I primi strati di una CNN sono specializzati nella rilevazione di bordi e linee, proprio come i neuroni visivi nella corteccia visiva primaria (V1).
- Strati successivi delle CNN sono invece specializzati nella percezione di caratteristiche visive più complesse, come forme e oggetti, in modo simile alle aree visive V2 e V4.
- Infine, gli strati più profondi delle CNN possono essere considerati analoghi alla corteccia inferotemporale (IT) nella gerarchia visiva, poiché sono in grado di elaborare informazioni visive complesse e riconoscere oggetti e classi di oggetti.

# Architettura di una CNN

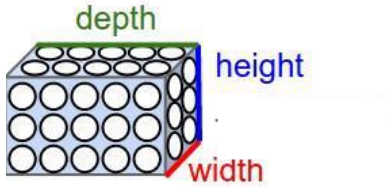
**Convolutional Neural Networks CNN introdotte da LeCun et al a partire dal 1998**

Le principali differenze rispetto a MLP

usano gli operatori convoluzionali per estrarre le features

- **Processing locale**: i neuroni sono connessi solo localmente ai neuroni del livello precedente. Ogni neurone esegue quindi un'elaborazione locale. Forte riduzione numero di connessioni.
- **Pesi condivisi**: i pesi sono condivisi a gruppi. Neuroni diversi dello stesso livello eseguono lo stesso tipo di elaborazione su porzioni diverse dell'input. Forte riduzione numero di pesi
- **Alternanza livelli di feature extraction e pooling**

I neuroni in ogni livello sono organizzati secondo una piccola griglia 3d,
Width
Height
depth

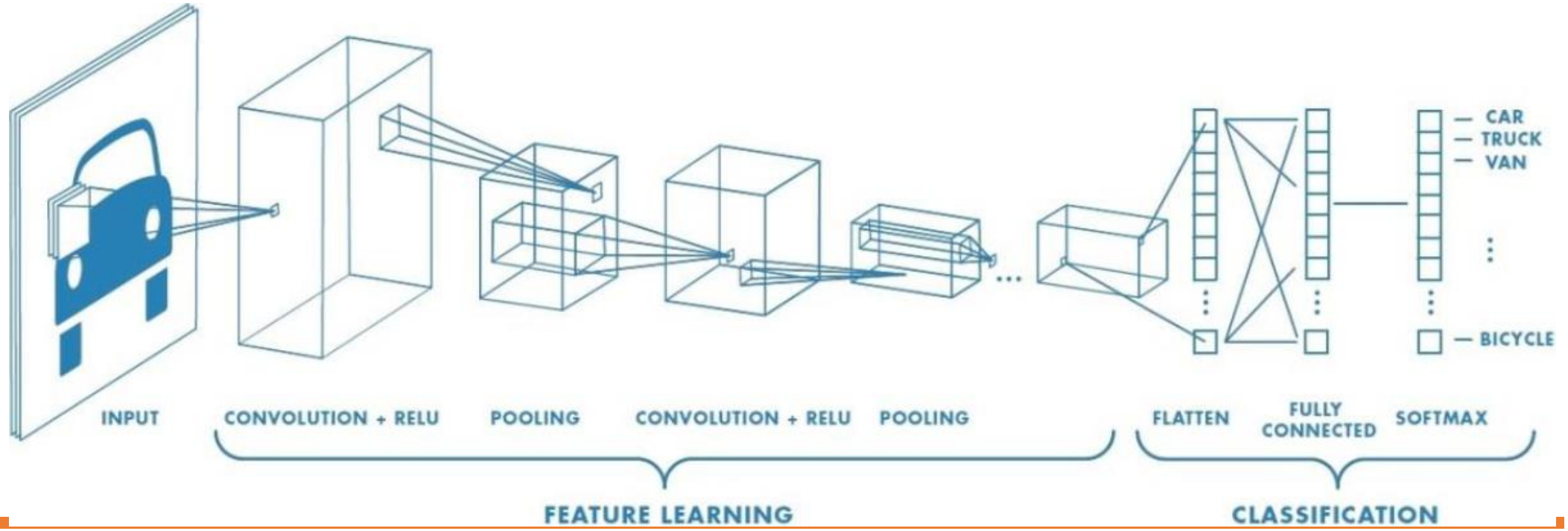



che ha dimensioni piccole se confrontate con l'immagine, in genere Larghezza ed altezza variano tra 3,5,7,11

# Architettura di una CNN

Una CNN è una combinazione di due parti fondamentali

• La parte convoluzionale consiste di strati convoluzionali seguite da funzioni di attivazione non lineare tipo(RELU)  e di pooling. Questa parte costituisce il componente essenziale dell'estrazione di feature
• La parte fully-connected  consiste in un'architettura di rete neurale completamente connessa.  Questa parte esegue il compito di classificazione in base all'input dalla parte convoluzionale

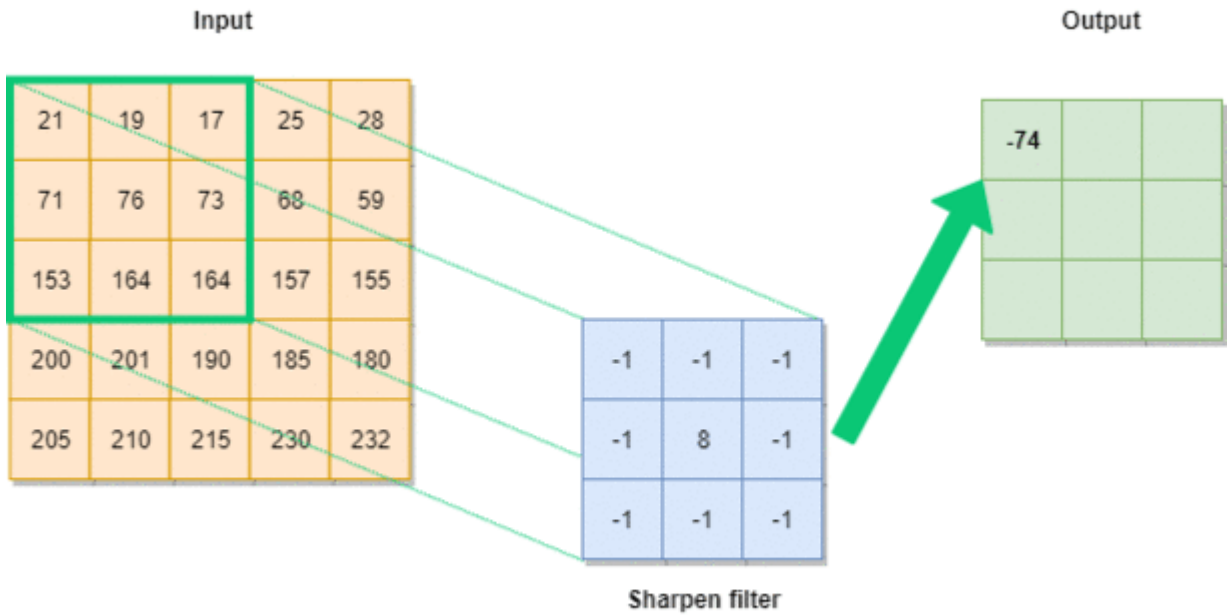


- Tre tipi principali di livelli vengono utilizzati per costruire architetture CNN
- **Strato convoluzionale:** contiene una serie di filtri apprendibili. La larghezza e l'altezza dei filtri sono più piccoli di quelli del volume di ingresso. Il filtro scorre attraverso l'ingresso e il prodotto scalare tra l'input e il filtro vengono calcolati in ogni posizione spaziale. E' seguito da una funzione di attivazione non lineare (tipo RELU)
- **Strato di pooling** riduce il numero di parametri e preserva l'invarianza per traslazione
- I neuroni dello **strato fully connected** in uno strato completamente connesso hanno connessioni complete con tutte le attivazioni nel livello precedente, come nelle ANNs tradizionali

# Convoluzione

- La convoluzione è una delle più importanti operazioni di image processing attraverso la quale si applicano filtri digitali, per estrarre feature dalle immagini.
- Un filtro digitale $h$ (una piccola maschera 2D di pesi, di dimensione $F \times F$ ) viene fatto scorrere su ogni pixel $(x, y)$ di un' immagine di input, $f$, *di dimensione* $m \times n$) per ogni posizione viene generato un valore di output $g(x, y)$, eseguendo il prodotto scalare tra la maschera e la porzione dell'input coperta (entrambi trattati come vettori). L'output $g(x, y)$ prende il nome di *features map.*

$$g(x,y) = (f * h)(x,y) = \sum_{i=0}^{F-1} \sum_{j=0}^{F-1} f\left(x - \left\lceil \frac{F}{2} \right\rceil + i, y - \left\lceil \frac{F}{2} \right\rceil + j\right) h(i,j)$$


Input
Output
21 19 17 25 28
71 76 73 68 59
153 164 164 157 155
200 201 190 185 180
205 210 215 230 232
-1 -1 -1
-1 8 -1
-1 -1 -1
-74
Sharpen filter
AIGeekProgrammer.com © 2019

# Gestione dei bordi dell'immagine durante la convoluzione.

Quando si applica la convoluzione su un'immagine, ci si trova spesso di fronte al problema dei bordi dell'immagine. Infatti, **se si applica la convoluzione direttamente sull'immagine, i pixel ai bordi dell'immagine vengono trattati in modo diverso rispetto ai pixel al centro dell'immagine, poiché il filtro non può essere centrato su di essi.**

Ci sono diverse strategie per **gestire i bordi dell'immagine durante la convoluzione.**
Una soluzione comune è quella di aggiungere dei bordi di padding all'immagine prima di applicare la convoluzione.
Il **padding** consiste **nell'aggiungere dei pixel intorno ai bordi dell'immagine, in modo da creare una cornice di pixel che permette di applicare la convoluzione anche sui bordi dell'immagine**.

Esistono diversi tipi di padding, ad esempio:

**Padding con zeri (zero-padding):** si aggiungono dei pixel con valore zero intorno ai bordi dell'immagine.
**Padding a specchio (mirror-padding):** si copiano i pixel lungo i bordi dell'immagine, in modo da creare una specie di riflesso rispetto ai bordi.

# Zero padding

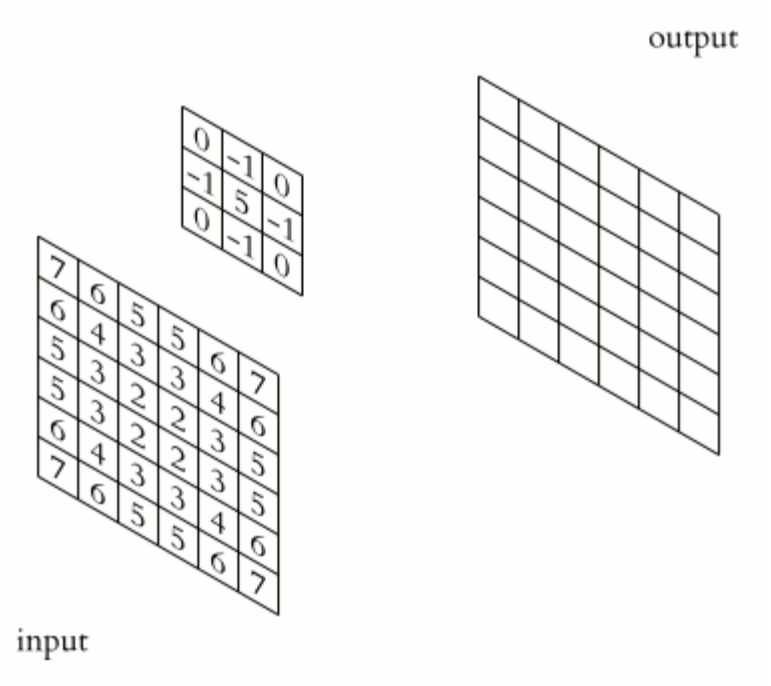
| | | | | | | |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 60 | 113 | 56 | 139 | 85 | 0 |
| 0 | 73 | 121 | 54 | 84 | 128 | 0 |
| 0 | 131 | 99 | 70 | 129 | 127 | 0 |
| 0 | 80 | 57 | 115 | 69 | 134 | 0 |
| 0 | 104 | 126 | 123 | 95 | 130 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Kernel

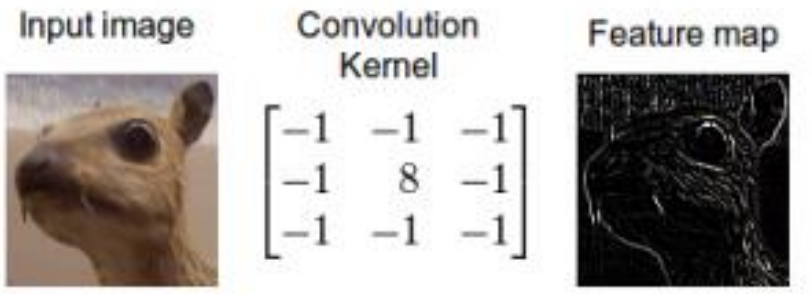
| | | |
|---|---|---|
| 0 | -1 | 0 |
| -1 | 5 | -1 |
| 0 | -1 | 0 |

| | | | | |
|---|---|---|---|---|
| 114 | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |

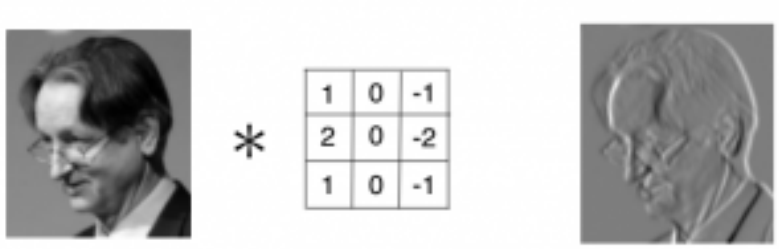
# mirror-padding



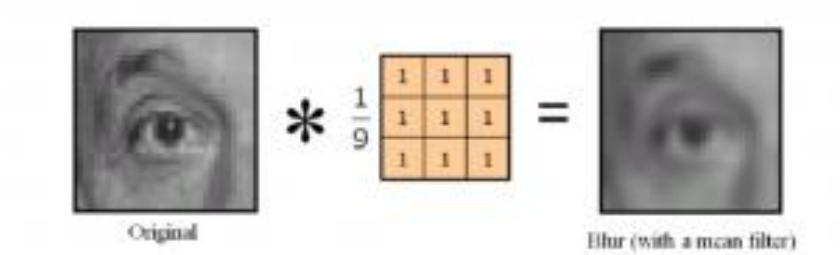
# Kernel diversi individuano features differenti



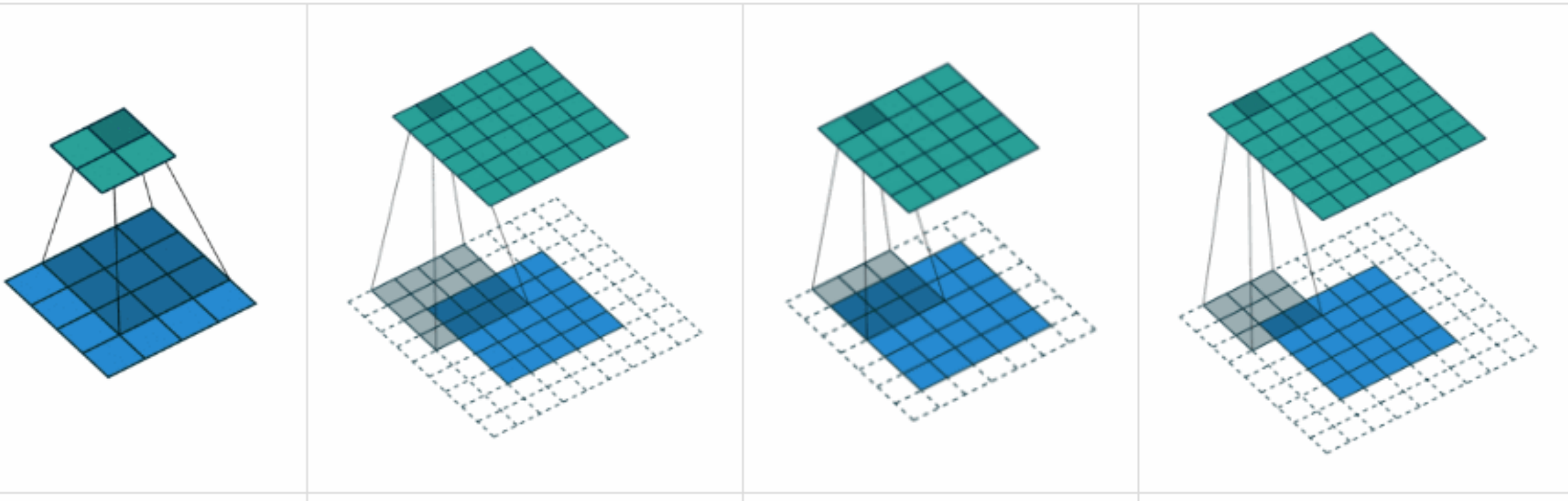

Esempio di kernel che evidenzia i contorni di una immagine.



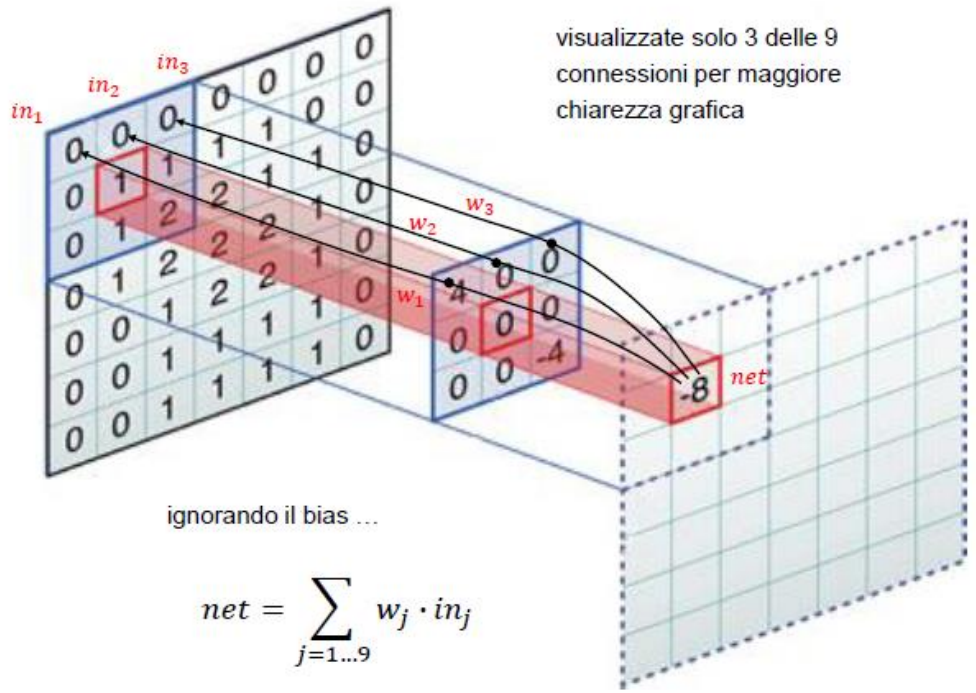

Esempio di kernel che produce un effetto embossed.




Esempi di kernel che produce un effetto blur

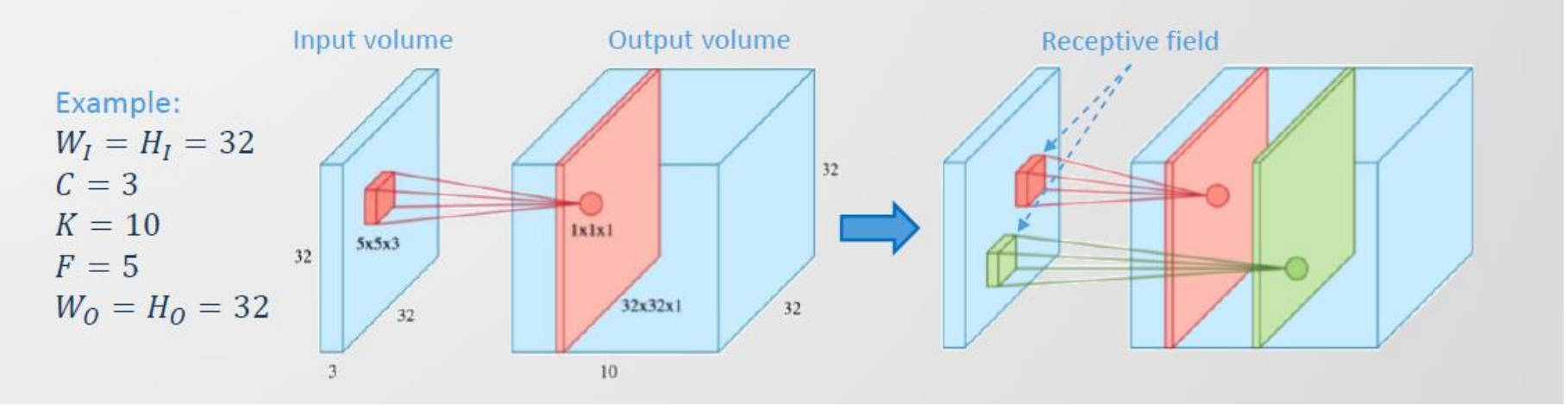


Le reti neurali convoluzionali  hanno i layer e i neuroni ad essi appartenenti, organizzati secondo larghezza, altezza e profondità (volume 3D) e non completamente connessi e con valori sinaptici condivisi, così da permettere di ridurre i parametri (pesi e bias) in gioco.



In una CNN, l'obiettivo dell'operazione di convoluzione è quello di estrarre caratteristiche dall'immagine di input mantenendo la relazione spaziale tra i pixel.

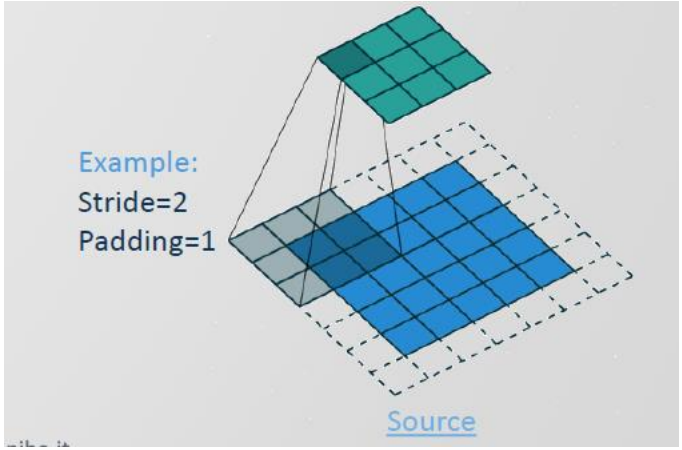
Ogni layer convoluzionale contiene un insieme di filtri (o kernel convoluzionali).
Data una volume di input tridimensionale (ad esempio l'immagine di input) di dimensioni $W_I \; x \; H_I \; x \; C$ e un insieme di K kernel di dimensioni $F \; x \; F \; x \; C$, l'output di un layer convoluzionale è un volume tridimensionale di dimensioni $W_O \; \times H_O \times \; K$ composto da K mappe di features  di dimensioni $W_O \; \times H_O$

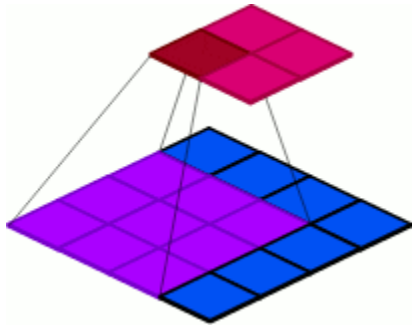


**La dimensione del volume di output di una convoluzione dipende da:**

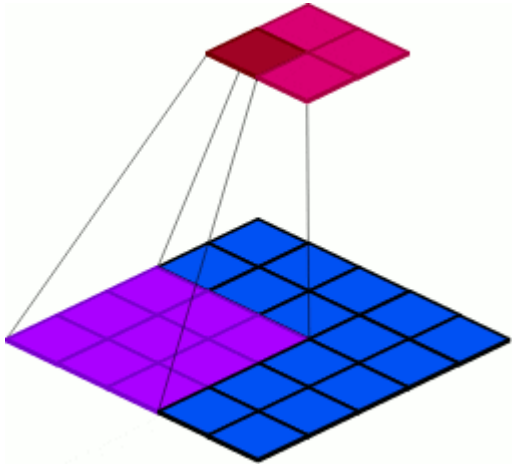
- **Profondità:** corrisponde al numero di filtri K che desideriamo utilizzare, ognuno dei quali impara a cercare qualcosa di diverso (ad esempio bordi orientati o blocchi di colore) nell'input.
- **Stride**: è la quantità di movimento tra l'applicazione del filtro al volume di input. Riduce la dimensione spaziale del volume di output e di conseguenza il numero di connessioni.
- **Padding:** aggiunge un bordo al volume di input per ottenere una dimensione spaziale specifica. Ci consente di controllare la dimensione spaziale del volume di output.

$$W_o = \frac{(W_{in} - F + 2 \cdot Padding)}{Stride} + 1$$

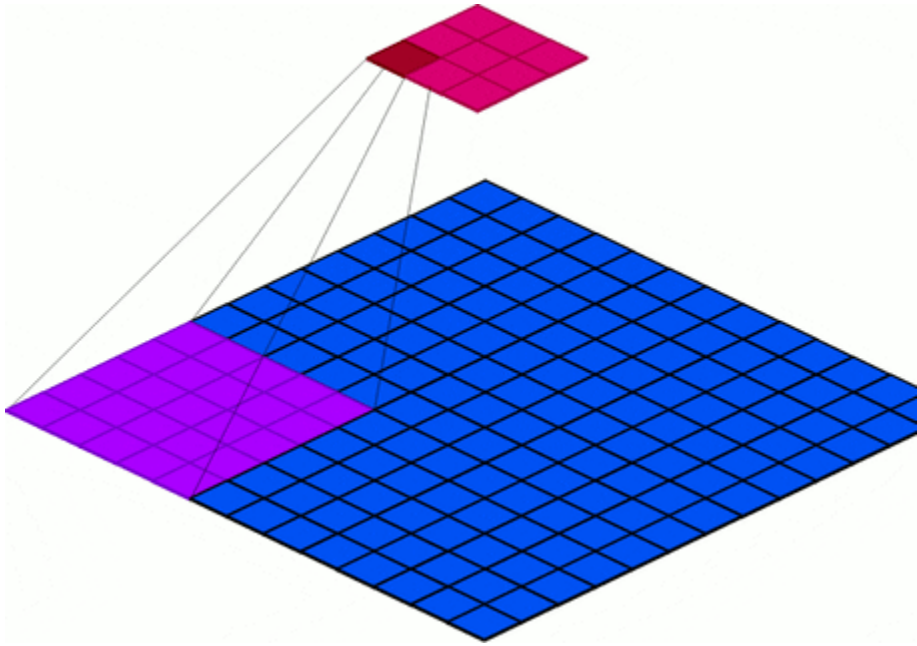




Dimensione del filtro (F) 3×3
**Stride (S): 1**
Padding: 0
Dimensione input: ( Wi × Hi ): 4×4
Dimensione output ( W0 × H0 ): 2×2



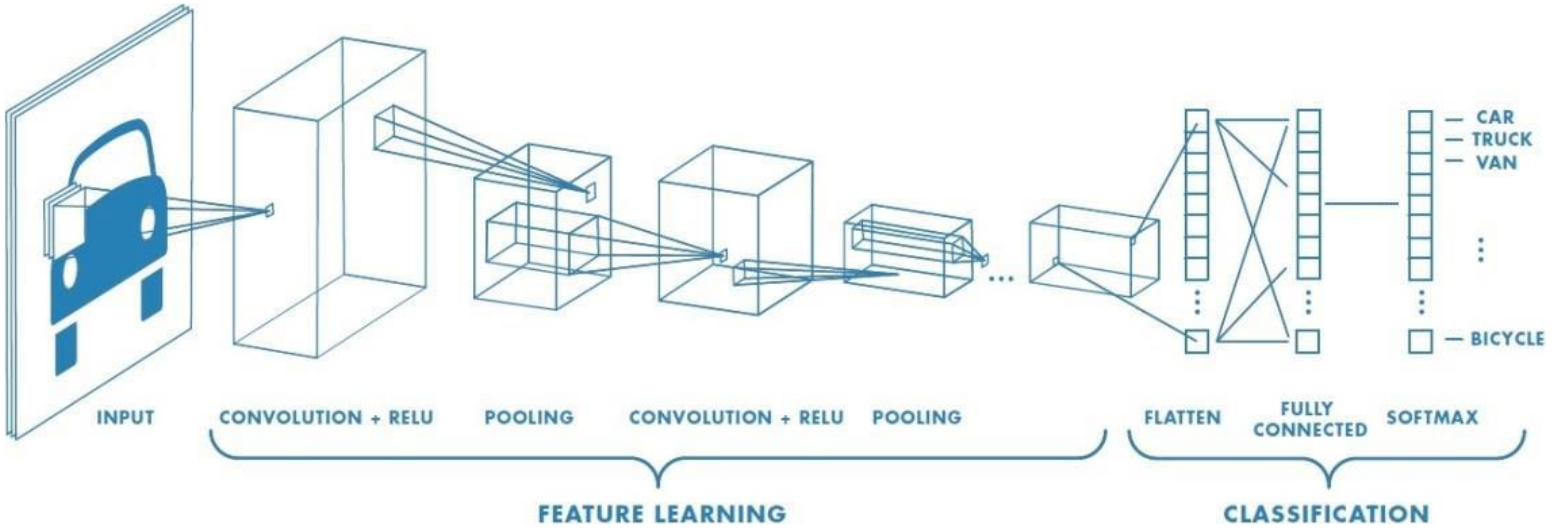
Dimensione del filtro (F) 3×3
**Stride (S): 2**
Padding (P): 0
Dimensione input ( Wi × Hi ): 5 × 5
Dimensione output( W0 × H0 ): 2 × 2

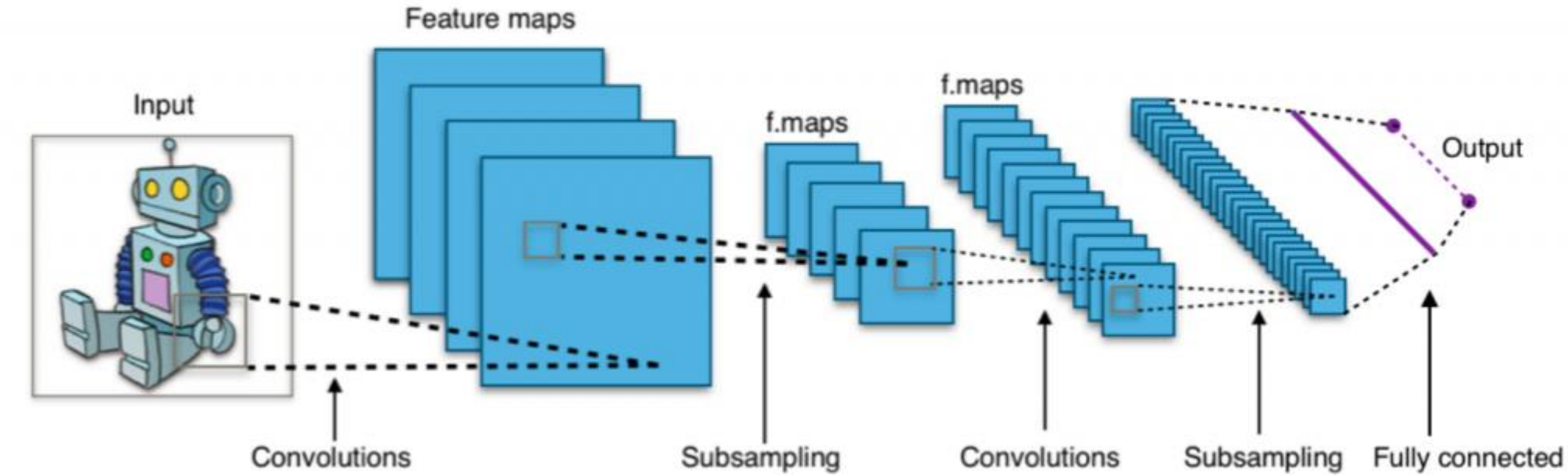


Dimensione del filtro (F) 5×5
**Stride: 4,**
padding (P): 0,
Dimensione input( Wi × Hi ): 13×13,
Dimensione output( W0 × H0 ): 3×3

# Architettura di una CNN

- L'architettura di una rete neurale convoluzionale differisce da una comune rete neurale per il suo strato nascosto che non è composto da soli layer fully connected (neuroni interconnessi ad ogni altro neurone dei livelli precedenti e successivi) ma anche da **livelli parzialmente connessi**.
- La struttura tipica di una rete ConvNet è quella che osserviamo nella figura




Feature maps
Input
f.maps
f.maps
Output
Convolutions
Subsampling
Convolutions
Subsampling
Fully connected

- **L'input layer** è lo strato in cui i dati vengono forniti alla rete. Questo livello è strutturato ad hoc sul dato in ingresso secondo le sue features specifiche, ad esempio in caso di un'immagine il livello di input è rappresentato come un insieme di pixel (e.g. h 224× w224× ch3).

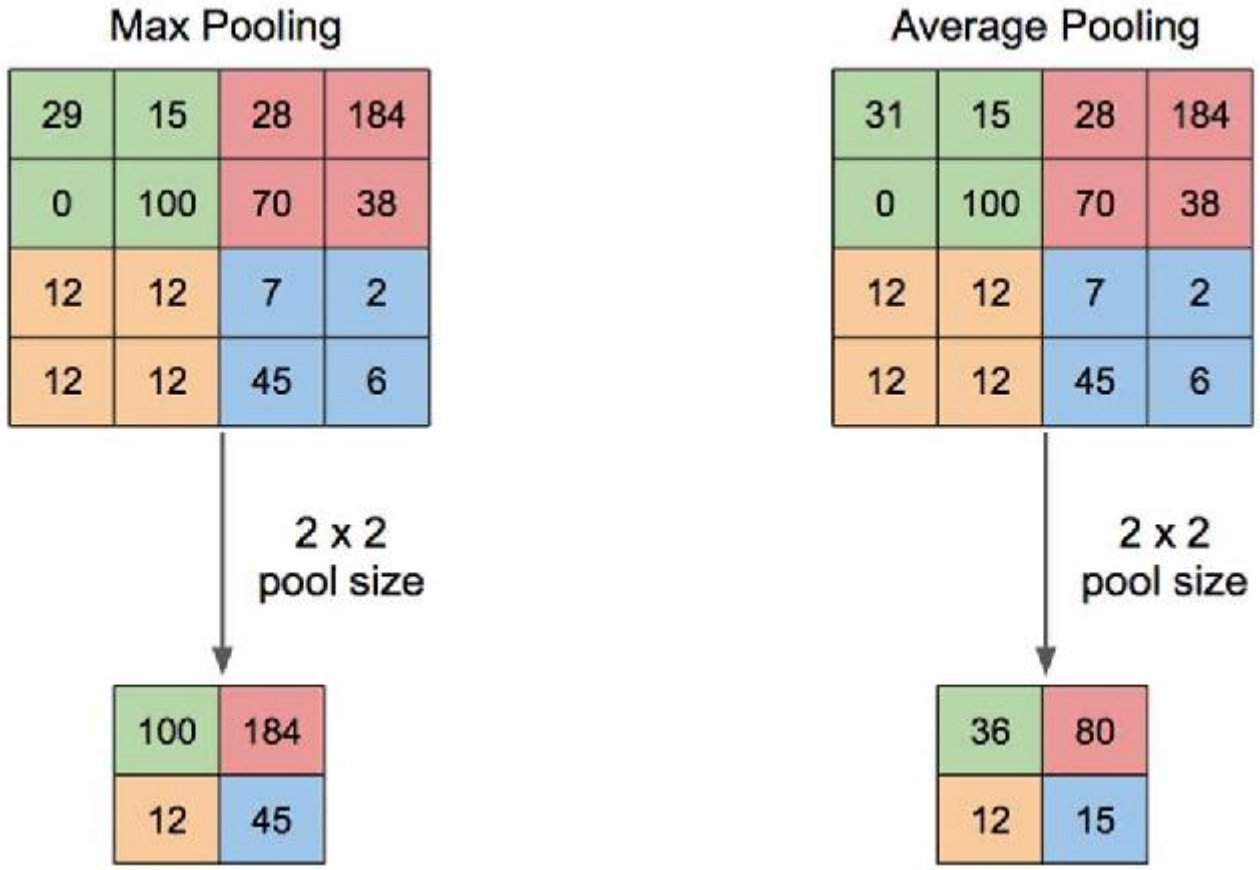
- Il ***Conv Layer*** ha lo scopo di riconoscere pattern specifici (quadrati, angoli, texture, curvature) in modo efficiente ed accurato. Una CNN può essere formata da più livelli di questo genere e ognuno di questi è in grado di riconoscere un pattern specifico differente.
- Aumentando il numero di *Conv Layer* è possibile raggiungere il riconoscimento di caratteristiche sempre più complesse.
- Lo strato successivo ad ogni convolutional layer è **il *Rectified Linear Units layer*** (abbreviato ReLu) che, come la funzione di attivazione lineare da cui prende il nome, ha l'obiettivo di rendere trascurabili i valori negativi derivanti dai layer precedenti.

- Altro livello estremamente importante posto sempre dopo il conv layer e il ReLu layer è lo strato di **Pool**. Questo livello rende in grado di **individuare gli elementi principali e più caratteristici di un'immagine e di ridurne la dimensionalità eliminando ciò che è superfluo per la rete per svolgere una corretta elaborazione e classificazione (successivamente verranno approfondite le modalità con cui questi livelli operano). L'immagine in output da un livello di pool sarà notevolmente semplificata e più grezza rispetto all'originale.**

Posta alla fine della rete neurale convoluzionale, c'è una rete fully connected utilizzata per la classificazione delle immagini, quindi della generazione dell'output di una CNN.
Questo strato di livelli completamente connessi riceve in input l'immagine manipolata dai precedenti layer e produce un vettore di N dimensione dove N corrisponde al numero di classi in cui si vuole classificare il volume in input.

# Pooling

- L'operatore di  Pooling è costituito da una finestra di forma fissa che scorre su tutte le regioni nell'input e calcola un singolo output per ogni posizione
- A differenza dei livelli convoluzionali, il layer di pooling non contiene parametri addestrabili
- Non è un livello addestrabile ma deterministico che in genere calcola il valore massimo o medio degli elementi nella finestra di pooling



# Convoluzione (Padding 0)

| | | |
|---|---|---|
| 1 | -1 | -1 |
| -1 | 1 | -1 |
| -1 | -1 | 1 |

Filtro 1

stride=1

| | | | | | |
|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 0 | 1 |
| 0 | 1 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 1 | 0 | 0 |
| 1 | 0 | 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 0 | 1 | 0 |

Prodotto scalare


Immagine 6 x 6

# Convoluzione

| | | |
|---|---|---|
| 1 | -1 | -1 |
| -1 | 1 | -1 |
| -1 | -1 | 1 |

Filtro 1

Se stride=2

| | | | | | |
|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 0 | 1 |
| 0 | 1 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 1 | 0 | 0 |
| 1 | 0 | 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 0 | 1 | 0 |

Immagine 6 x 6

3 -3

# Convoluzione

| | | |
|---|---|---|
| 1 | -1 | -1 |
| -1 | 1 | -1 |
| -1 | -1 | 1 |

Filtro 1

stride=1

| | | | | | |
|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 0 | 1 |
| 0 | 1 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 1 | 0 | 0 |
| 1 | 0 | 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 0 | 1 | 0 |

Immagine 6 x 6

| | | | |
|---|---|---|---|
| 3 | -1 | -3 | -1 |
| -3 | 1 | 0 | -3 |
| -3 | -3 | 0 | 1 |
| 3 | -2 | -2 | -1 |

# Convoluzione

| | | |
|---|---|---|
| -1 | 1 | -1 |
| -1 | 1 | -1 |
| -1 | 1 | -1 |

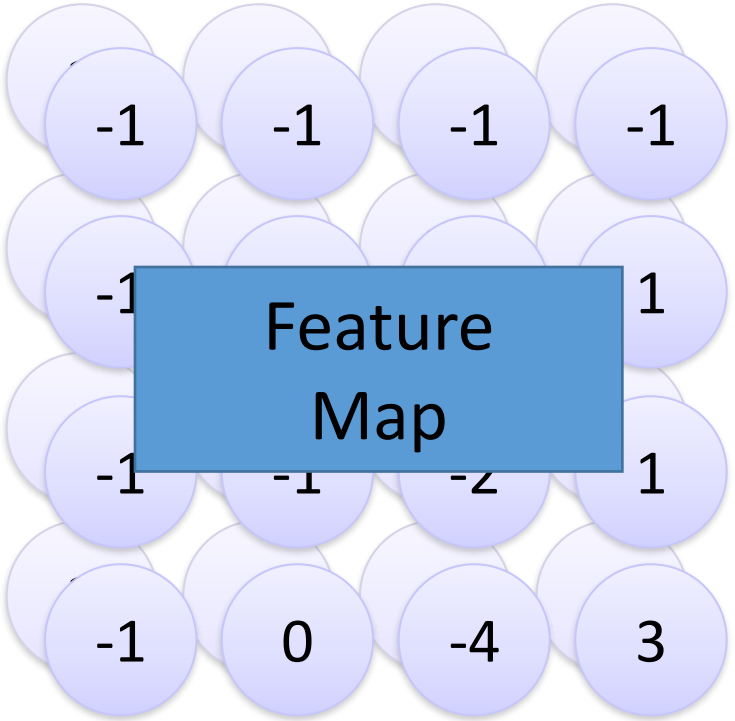
Filtro 2

stride=1

| | | | | | |
|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 0 | 1 |
| 0 | 1 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 1 | 0 | 0 |
| 1 | 0 | 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 0 | 1 | 0 |

6 x 6 image

Ripetere per ogni filtro



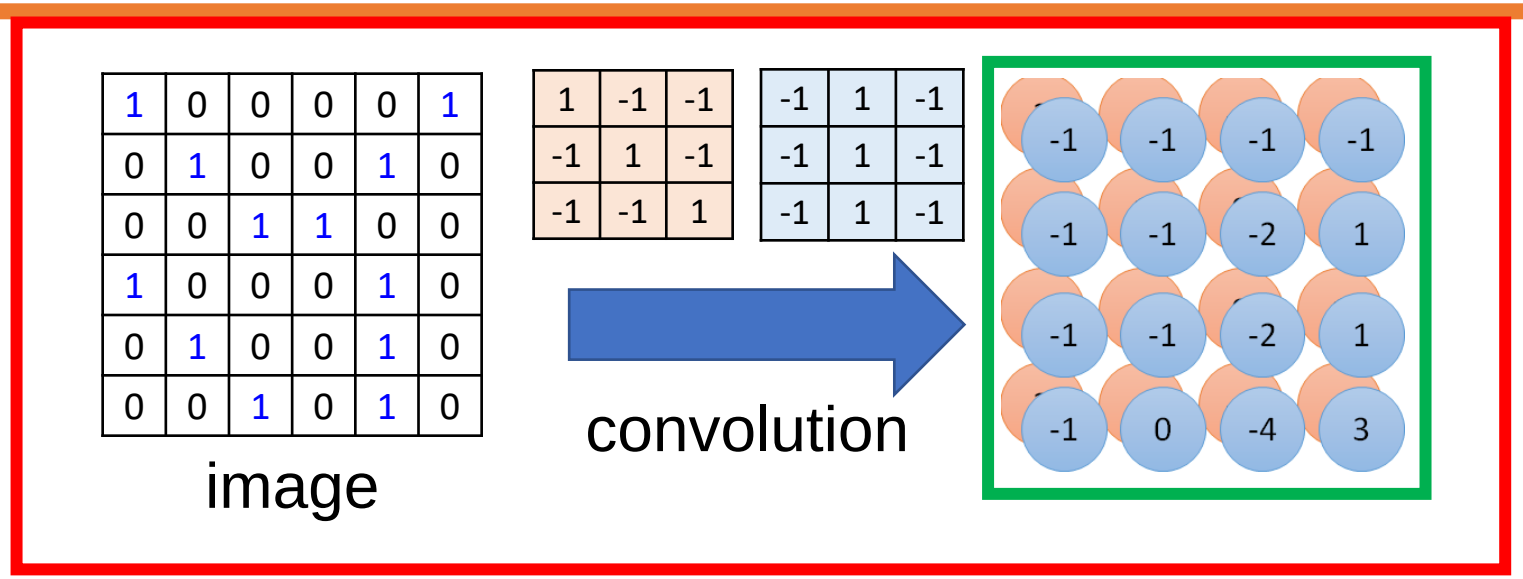

Due immagini 4 x 4

Formano una matrice 4 x4 x2

Nelle CNN i neuroni  in uno strato nascosto  sono collegati  solo a una piccola regione dello strato precedente (chiamato campo recettivo locale)

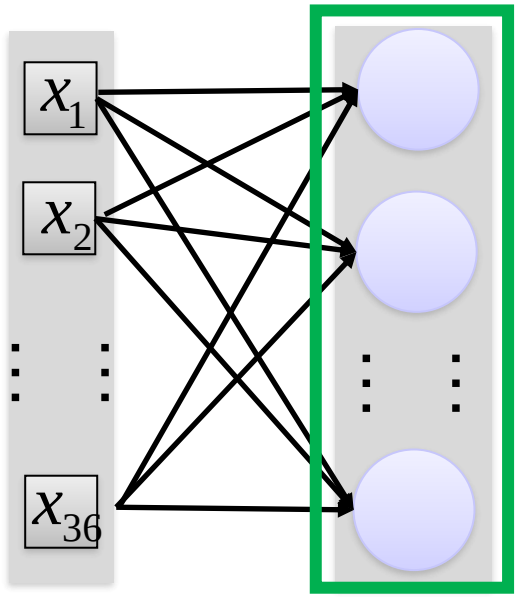
La profondità di ogni feature map corrisponde al numero di filtri convoluzionali utilizzati in ogni layer

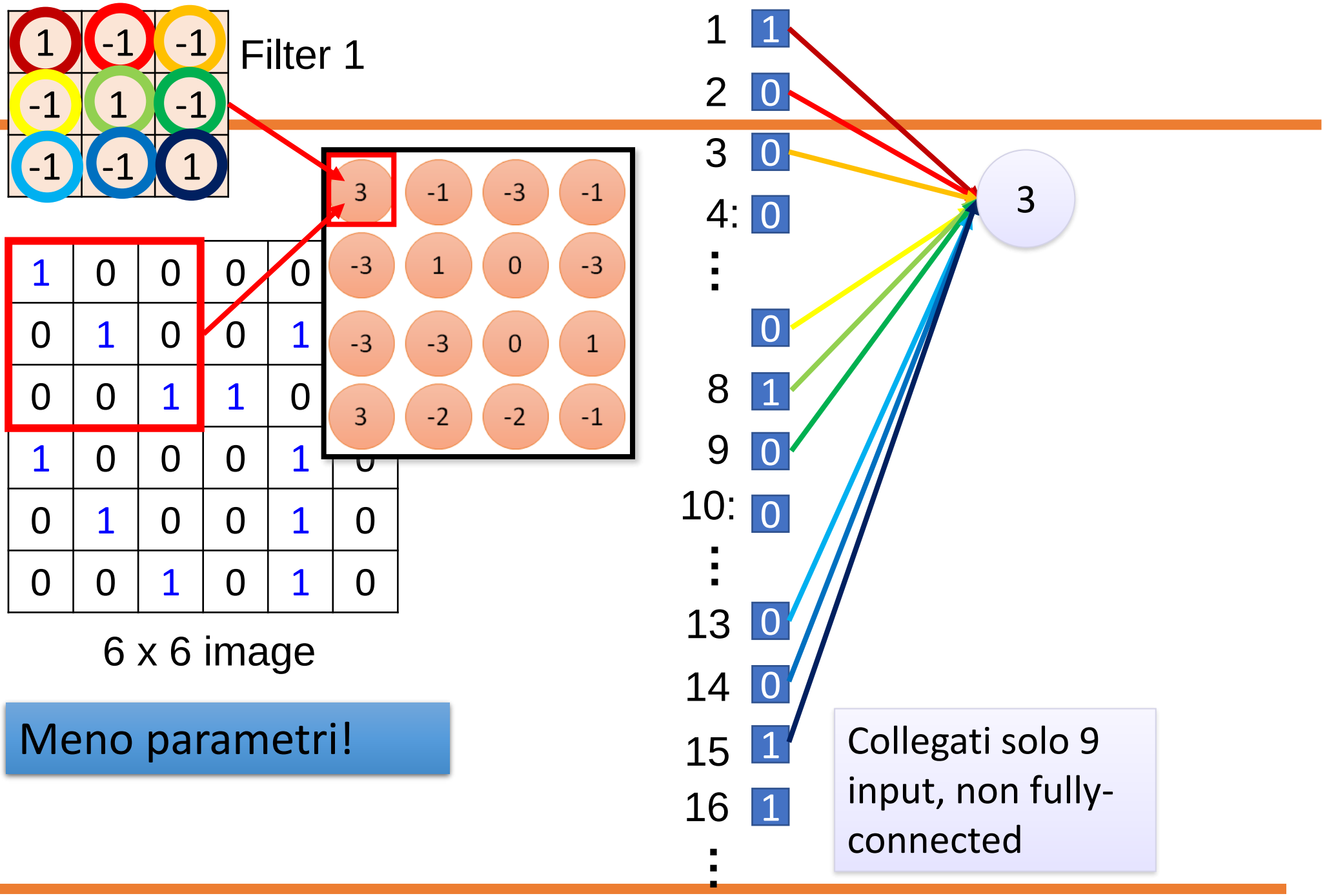
# Convolution v.s. Fully Connected

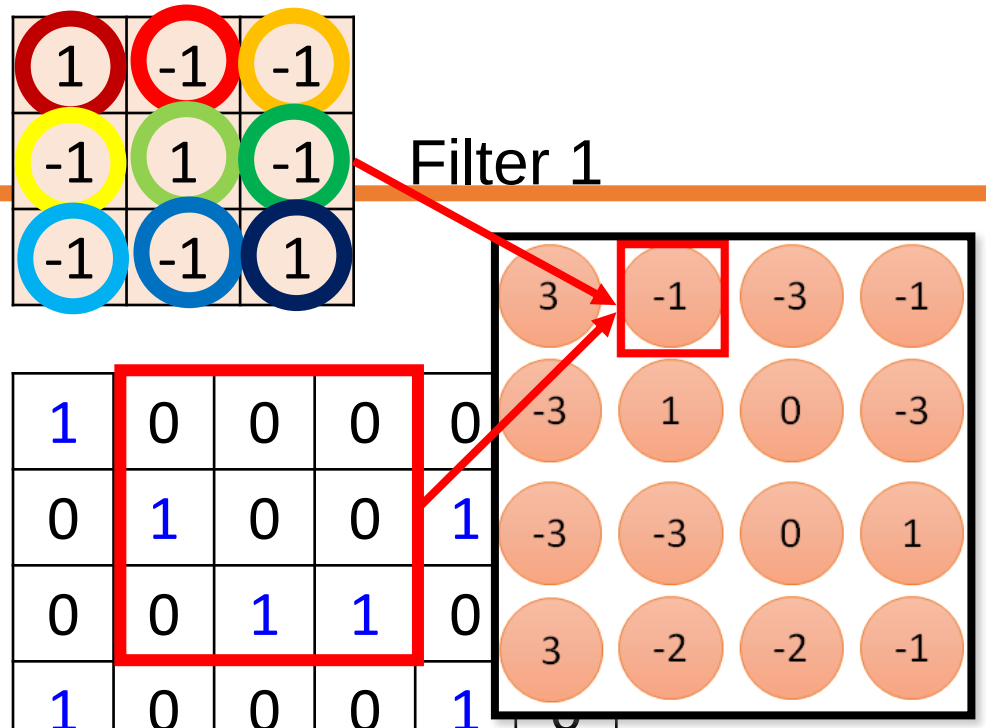


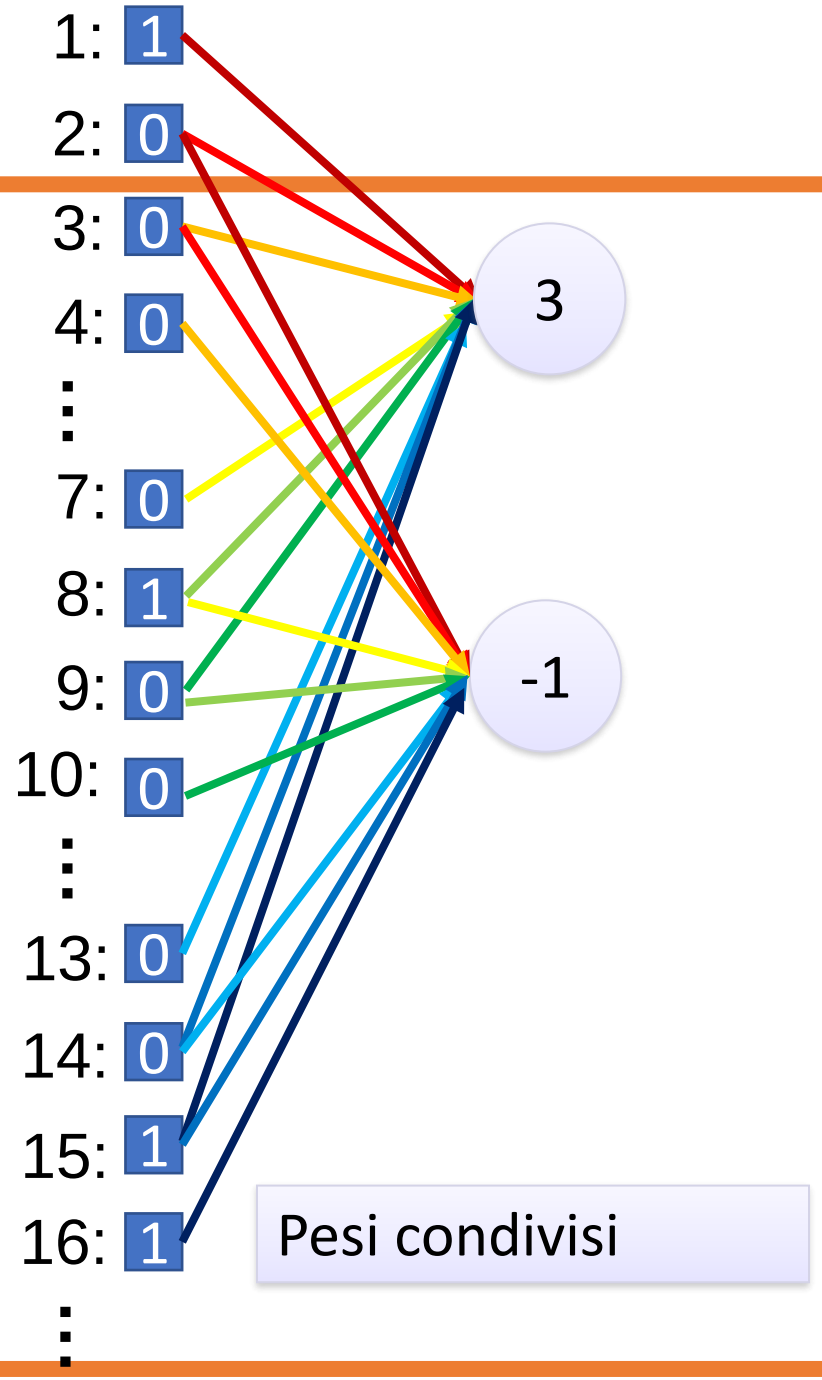

Fully-connected

| | | | | | |
|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 0 | 1 |
| 0 | 1 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 1 | 0 | 0 |
| 1 | 0 | 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 0 | 1 | 0 |




1 -1 -1
-1 1 -1
-1 -1 1
Filter 1
3 -1 -3 -1
-3 1 0 -3
-3 -3 0 1
3 -2 -2 -1
1 0 0 0 0
0 1 0 0 1
0 0 1 1 0
1 0 0 0 1
0 1 0 0 1 0
0 0 1 0 1 0
6 x 6 image
1 1
2 0
3 0
4: 0
0
8 1
9 0
10: 0
13 0
14 0
15 1
16 1
3
Meno parametri!
Collegati solo 9 input, non fully-connected



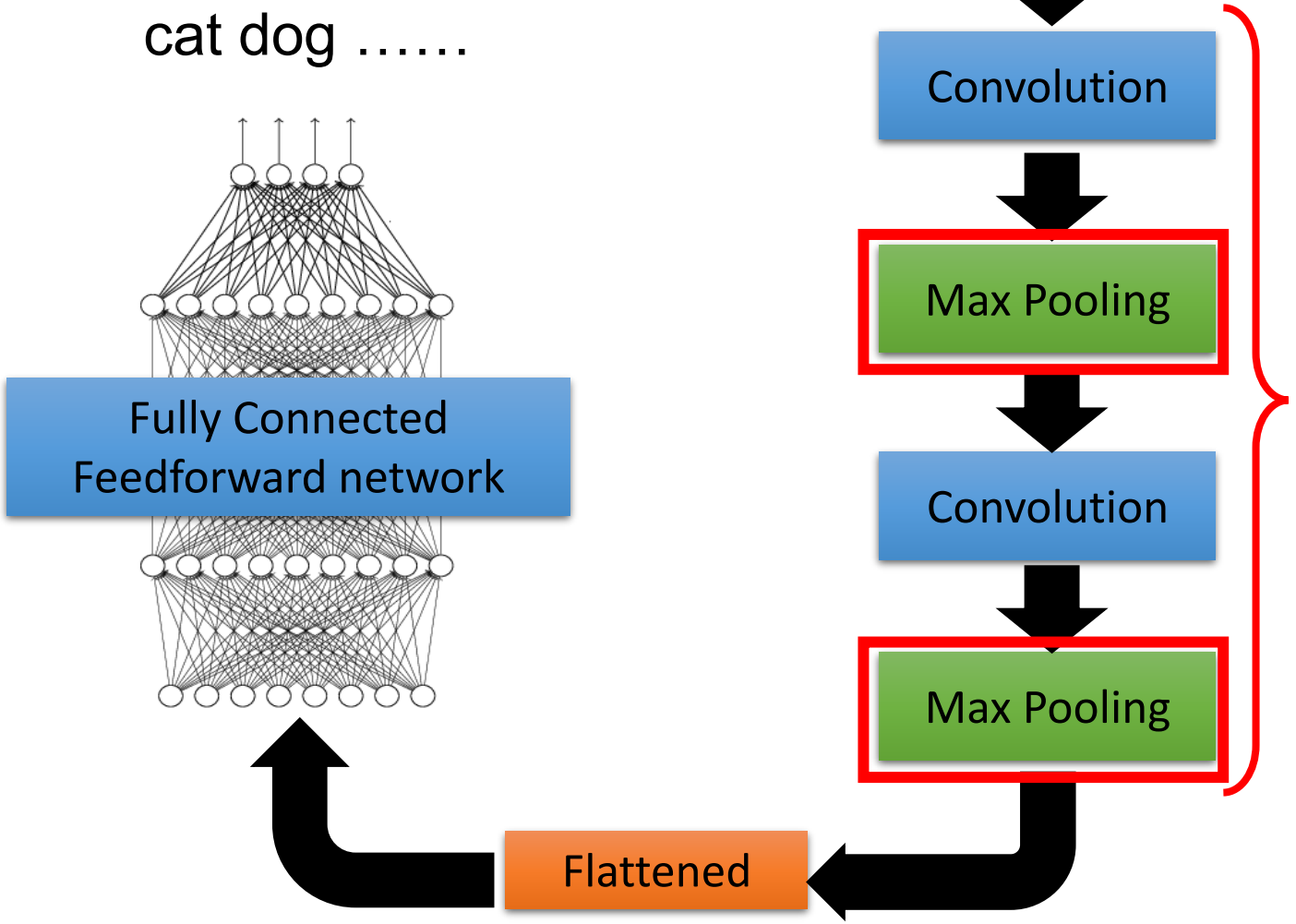




Numero di parametri inferiore

Ancora meno parametri

Pesi condivisi

# La CNN al completo



# Max Pooling

| | | |
|---|---|---|
| 1 | -1 | -1 |
| -1 | 1 | -1 |
| -1 | -1 | 1 |

Filter 1

| | | |
|---|---|---|
| -1 | 1 | -1 |
| -1 | 1 | -1 |
| -1 | 1 | -1 |

Filter 2

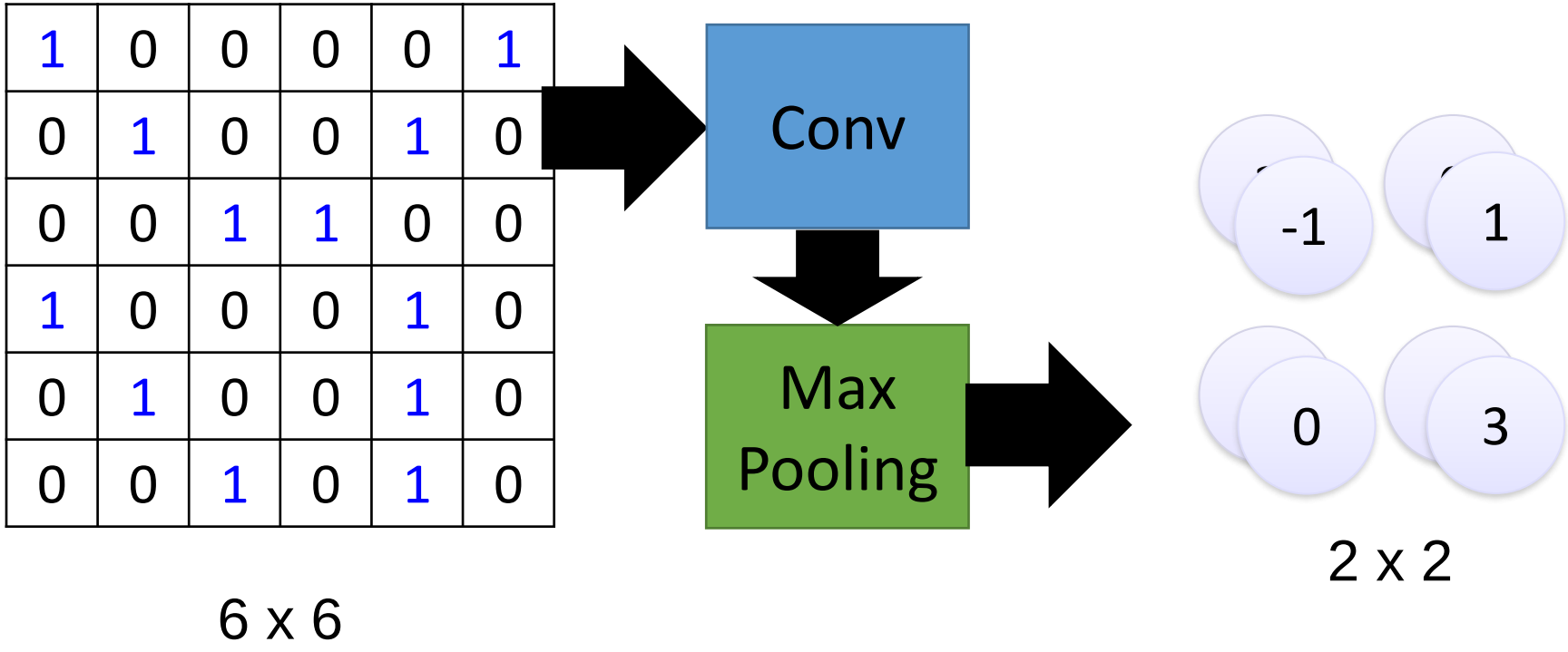
| | | | |
|---|---|---|---|
| 3 | -1 | -3 | -1 |
| -3 | 1 | 0 | -3 |
| -3 | -3 | 0 | 1 |
| 3 | -2 | -2 | -1 |

| | | | |
|---|---|---|---|
| -1 | -1 | -1 | -1 |
| -1 | -1 | -2 | 1 |
| -1 | -1 | -2 | 1 |
| -1 | 0 | -4 | 3 |

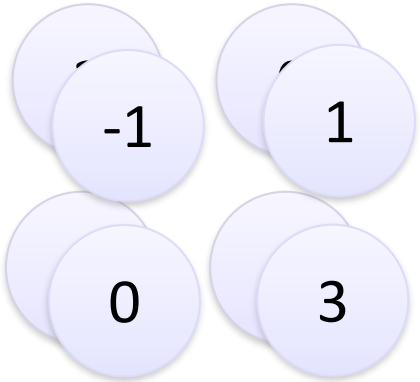
# Una CNN comprime una rete completamente connessa in due modi:

- Riducendo il numero di connessioni
- Pesi condivisi sugli edge
- Max pooling reduce ulteriormente la complessità.

# Max Pooling


1 0 0 0 0 1
0 1 0 0 1 0
0 0 1 1 0 0
1 0 0 0 1 0
0 1 0 0 1 0
0 0 1 0 1 0
6 x 6
Conv
Max Pooling
-1
1
0
3
2 x 2

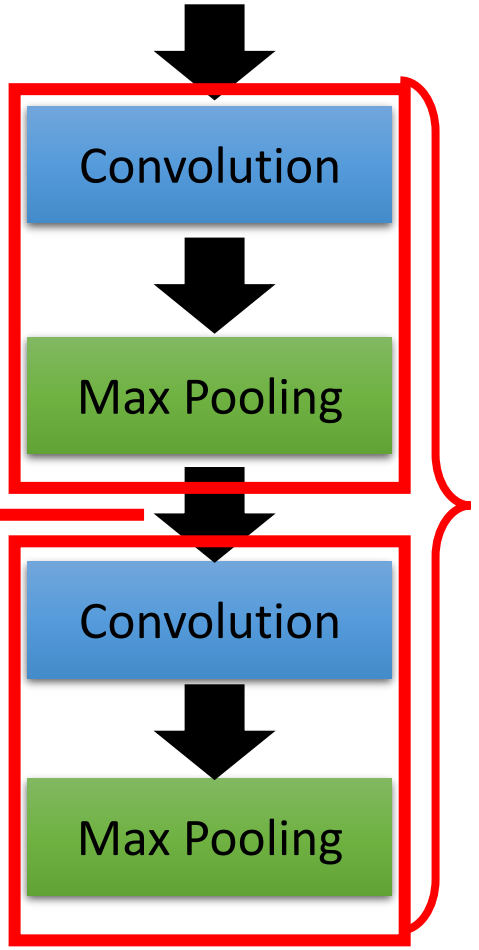
# La CNN al completo




Una nuova immagine

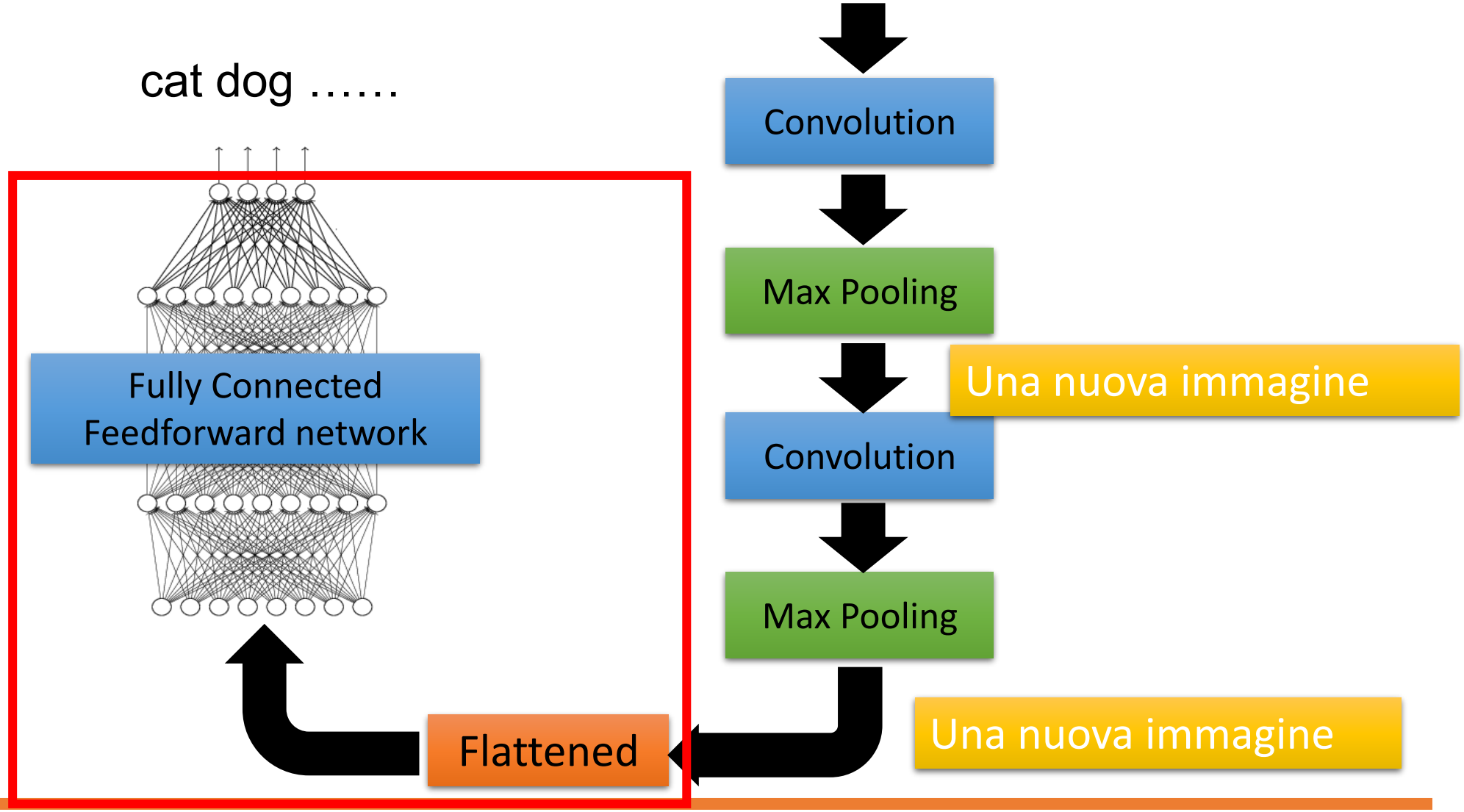
Più piccola dell'imagine originale

Il numero di canali è uguale al numero di filtri

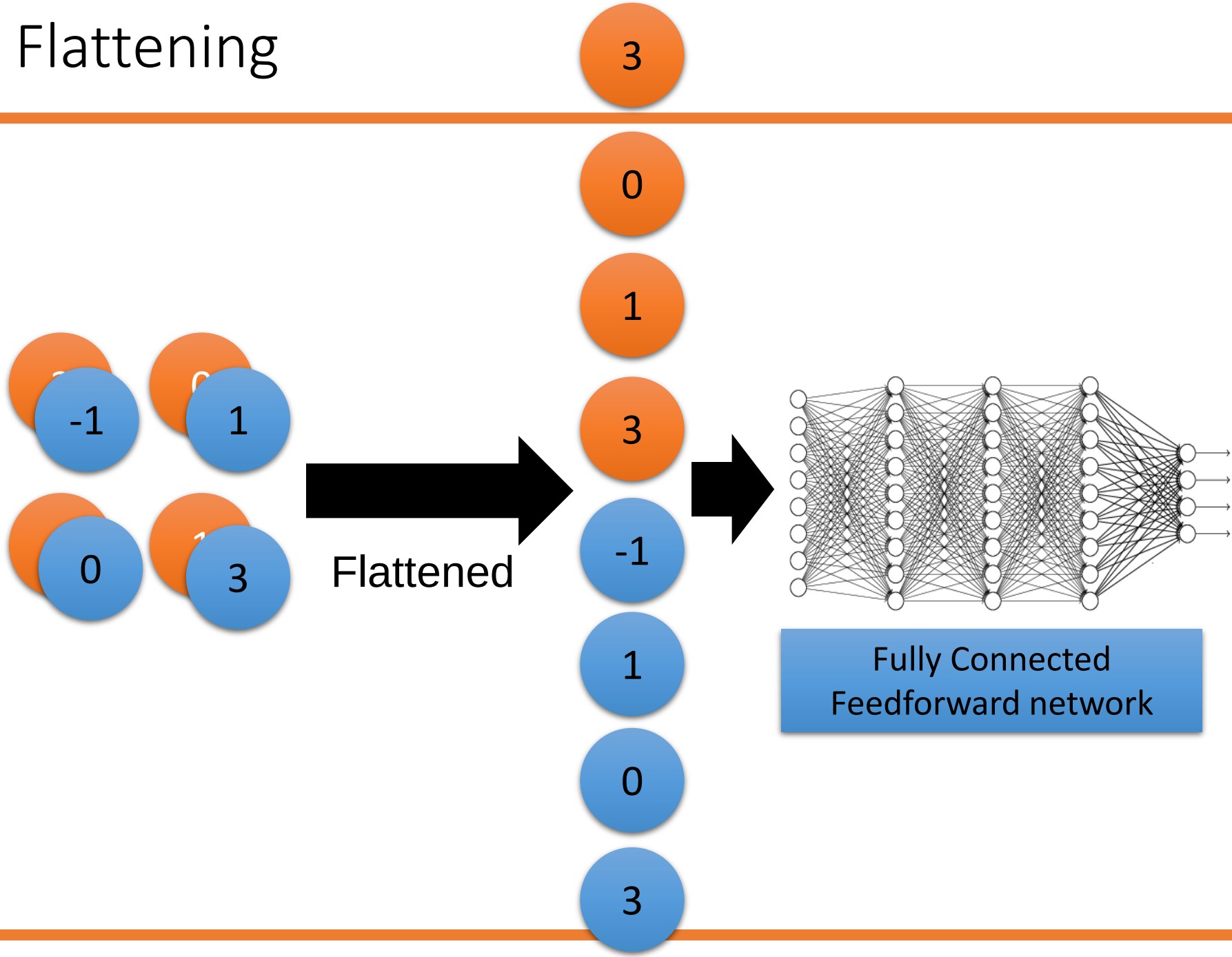



Si può ripetere più volte

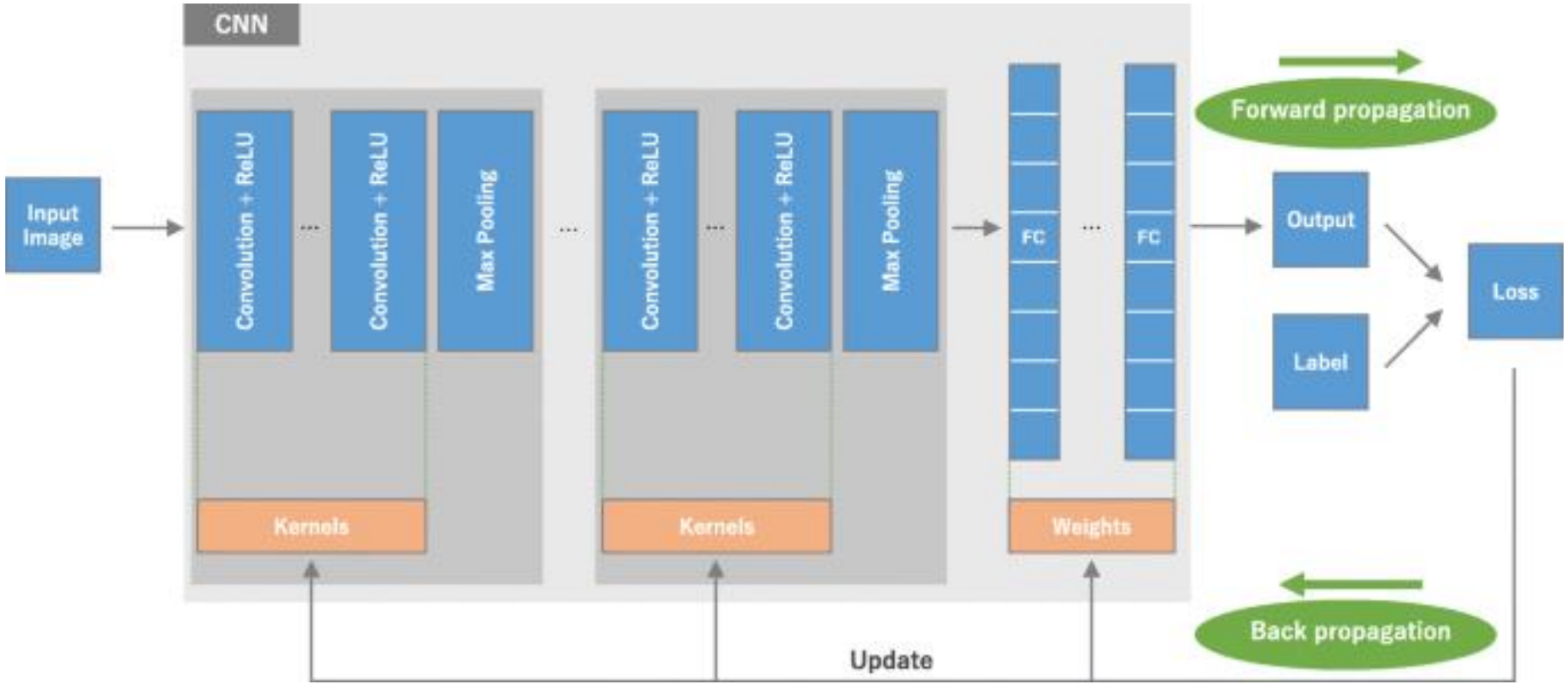
# La CNN al completo



# Flattening

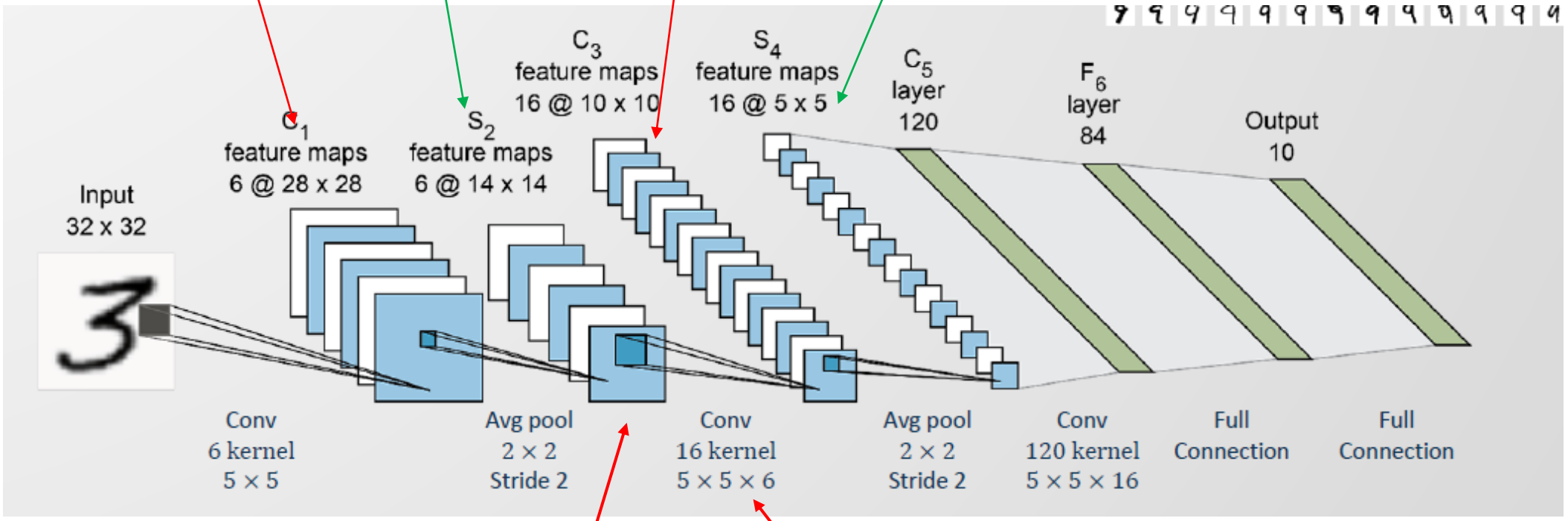

-1
1
0
3
Flattened
3
0
1
3
-1
1
0
3
Fully Connected
Feedforward network

# Training di una Convolutional Neural Network



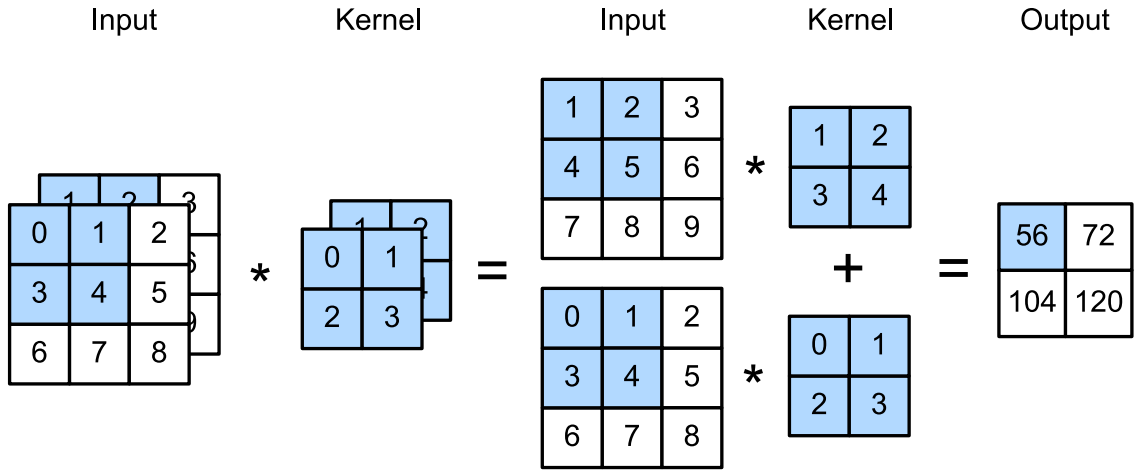
Simple cells Complex cells Simple cells Complex cells

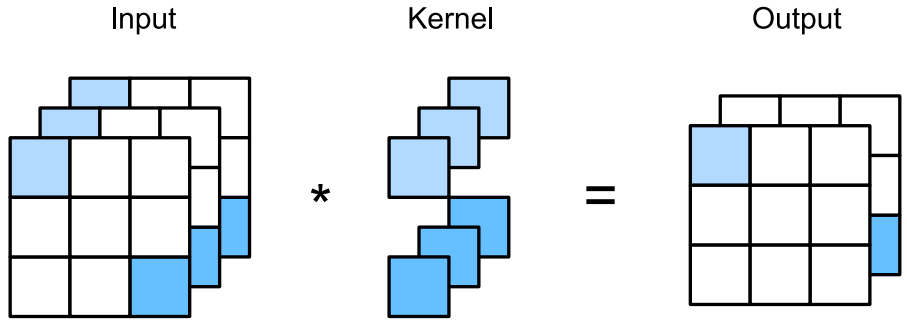
## Architettura LENET-5- Rete CNN per riconoscere cifre scritte a mano



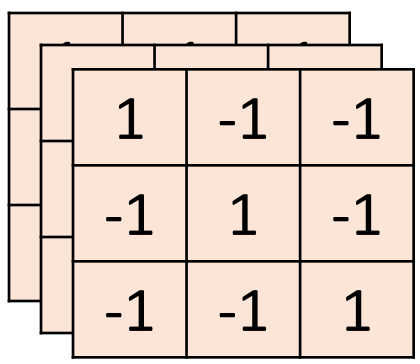

**Nota**: Quando il volume di input in un'operazione di convoluzione contiene più canali (cioè è un tensore tridimensionale), solitamente si **utilizzano kernel di convoluzione con lo stesso numero di canali del volume di input**. In questo modo, per ogni kernel si esegue un'operazione di convoluzione tra ogni canale di input e il corrispondente canale del kernel. Alla fine, i risultati delle convoluzioni su ciascun canale vengono sommati insieme per produrre la mappa delle caratteristiche bidimensionale finale.

# Caso in cui il volume di input in un'operazione di convoluzione contiene più canali

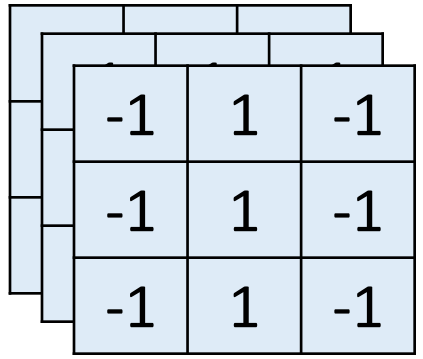


# Input immagine a colori: RGB 3 canali

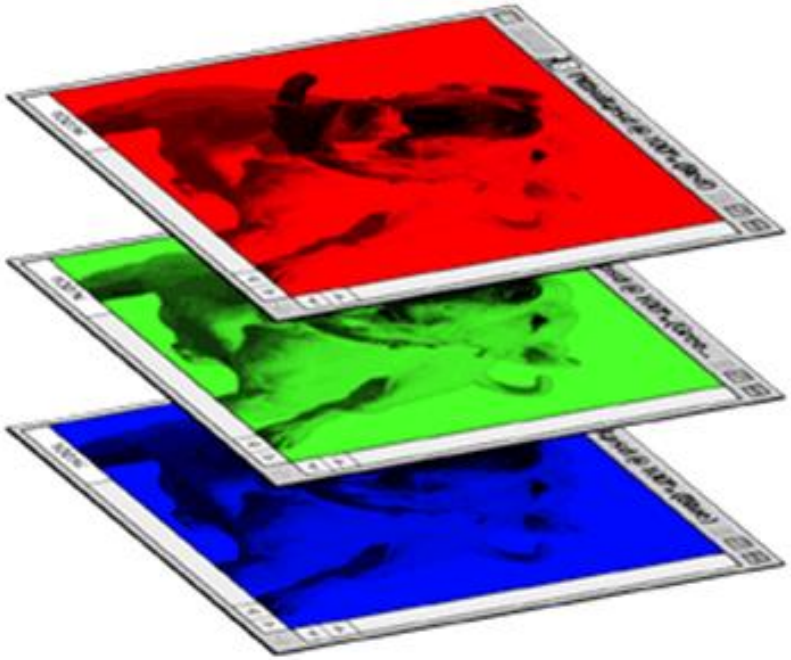


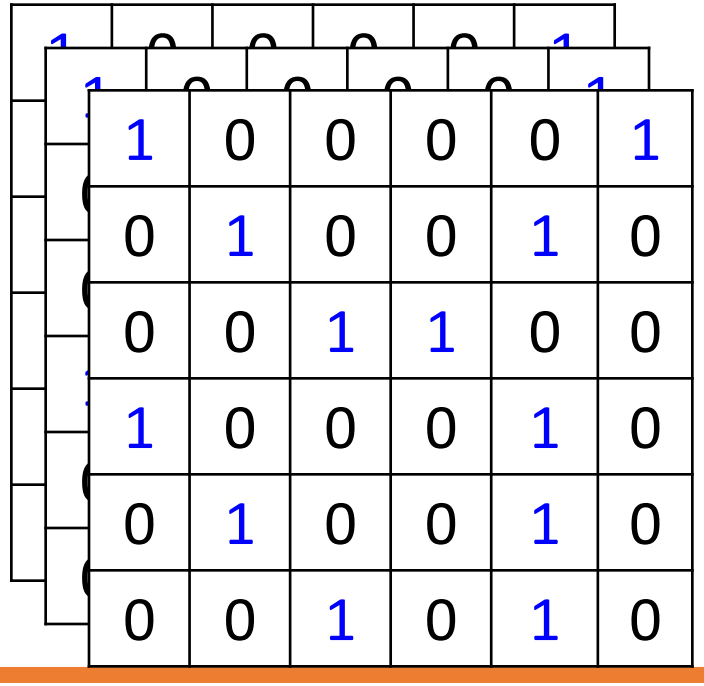

Filter 1

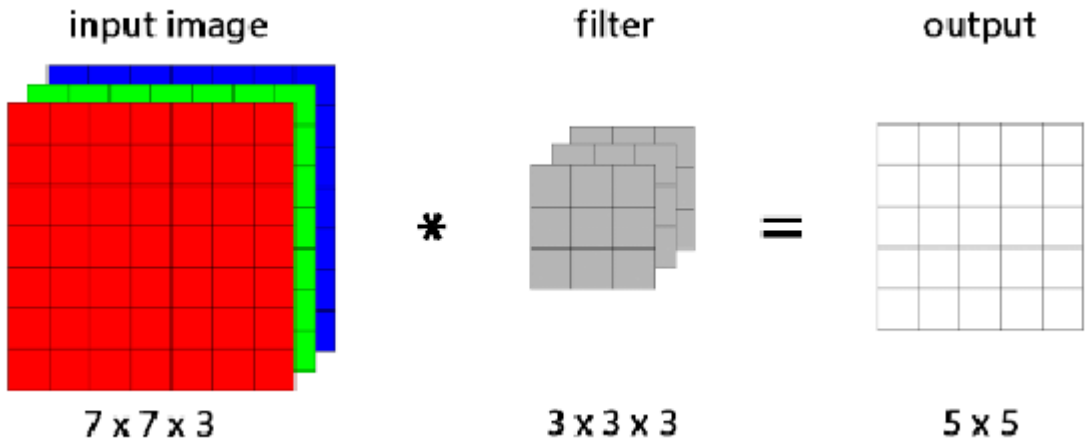



Filter 2

Immagine a colori






input image
filter
output
*
=
7 x 7 x 3
3 x 3 x 3
5 x 5