

# Deep Learning

## Chapter 3 Convolutional Neural Network

Dr. Van-Toi NGUYEN

*EEE, Phenikaa University*

# Chapter 3: Convolutional Neural Network

1. Convolutional operator
2. History of CNN
3. Deep Convolutional Models
4. Layers in CNN
5. Applications of CNN
6. Practice



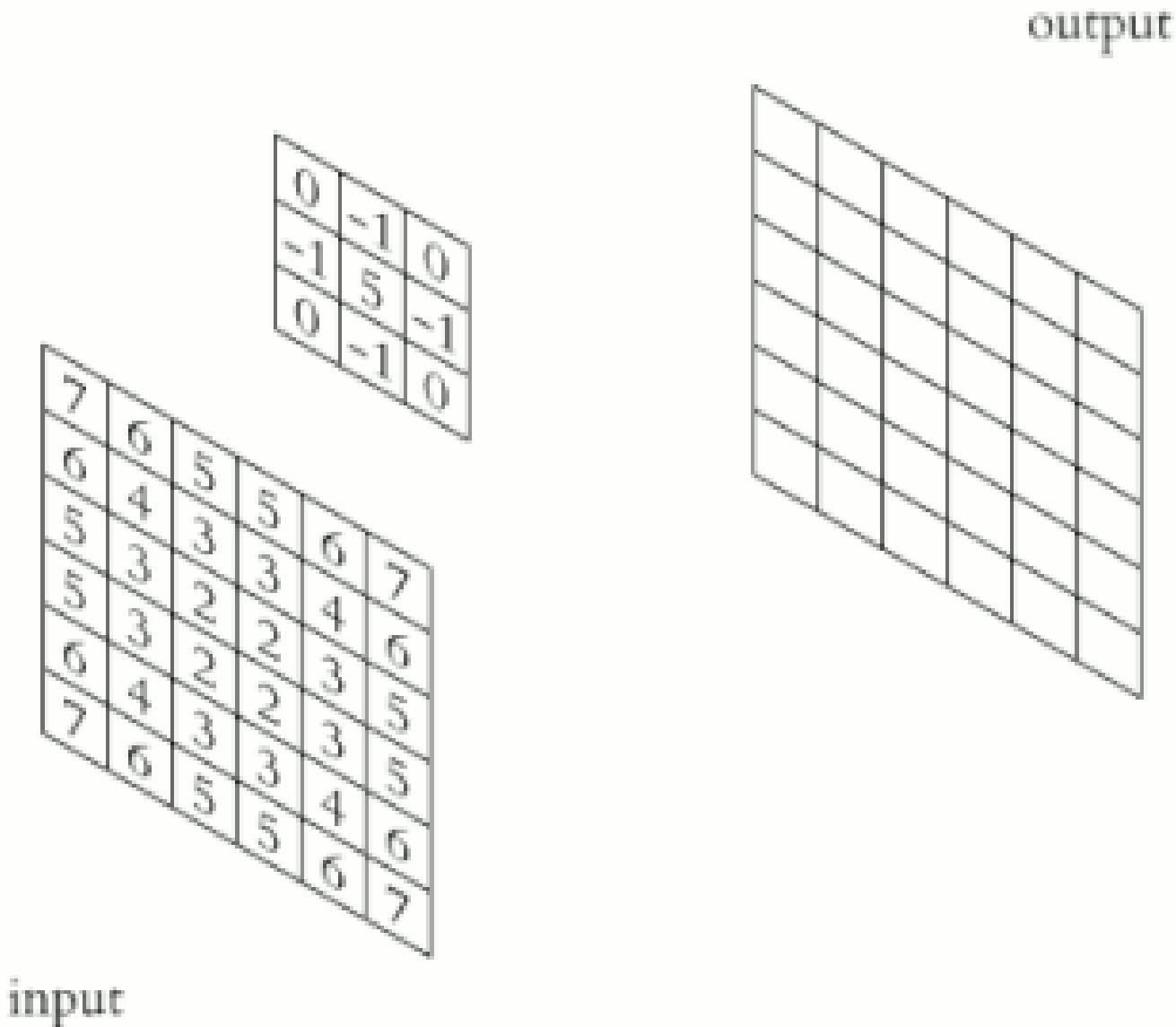
### 3.1 Convolutional Operator

IA  
Y

The general form for matrix convolution is

$$\begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ x_{m1} & x_{m2} & \cdots & x_{mn} \end{bmatrix} * \begin{bmatrix} y_{11} & y_{12} & \cdots & y_{1n} \\ y_{21} & y_{22} & \cdots & y_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ y_{m1} & y_{m2} & \cdots & y_{mn} \end{bmatrix} = \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} x_{(m-i)(n-j)} y_{(1+i)(1+j)}$$

# 3.1 Convolutional Operator



# 3.1 Convolutional Operator

3 <sup>1</sup>	0 <sup>0</sup>	1 <sup>-0</sup>	2 <sup>-10</sup>	7 <sup>-0</sup>	4 <sup>-1</sup>
1 <sup>1</sup>	5 <sup>0</sup>	8 <sup>-0</sup>	9 <sup>-0</sup>	3 <sup>-0</sup>	1 <sup>-1</sup>
2 <sup>-1</sup>	7 <sup>0</sup>	2 <sup>-0</sup>	5 <sup>-0</sup>	1 <sup>-0</sup>	3 <sup>-1</sup>
0 <sup>1</sup>	1 <sup>0</sup>	3 <sup>-0</sup>	1 <sup>-0</sup>	7 <sup>-0</sup>	8 <sup>-1</sup>
4	2	1	6	2	8
2	4	5	2	3	9

convolution



1	0	-1
1	0	-1
1	0	-1

3 x 3 filter (kernel)

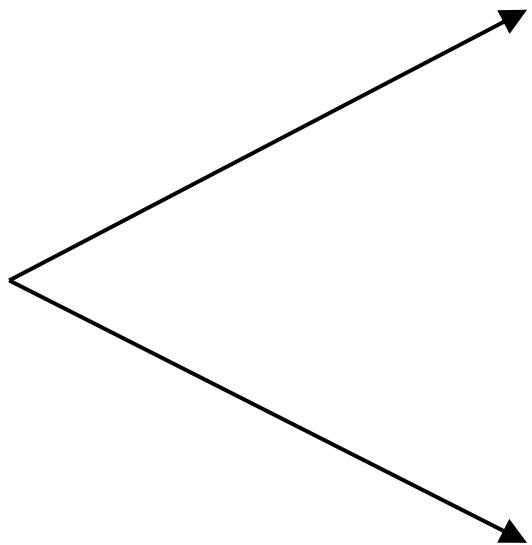
=

-5	-4	0	9
-10	-2	2	3
0	-2	-4	-7
-3	-2	-3	-16

# 3.1 Convolutional Operator



## To Extract Edges in an Image



vertical edges



horizontal edges

# 3.1 Convolutional Operator

To Extract Edges in an Image

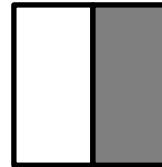
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0

\*

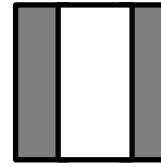
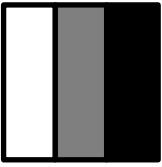
1	0	-1
1	0	-1
1	0	-1

=

0	30	30	0
0	30	30	0
0	30	30	0
0	30	30	0



\*



# 3.1 Convolutional Operator

To Extract Edges in an Image

1	0	-1
1	0	-1
1	0	-1

Vertical

10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
0	0	0	10	10	10
0	0	0	10	10	10
0	0	0	10	10	10

\*

1	1	1
0	0	0
-1	-1	-1

1	1	1
0	0	0
-1	-1	-1

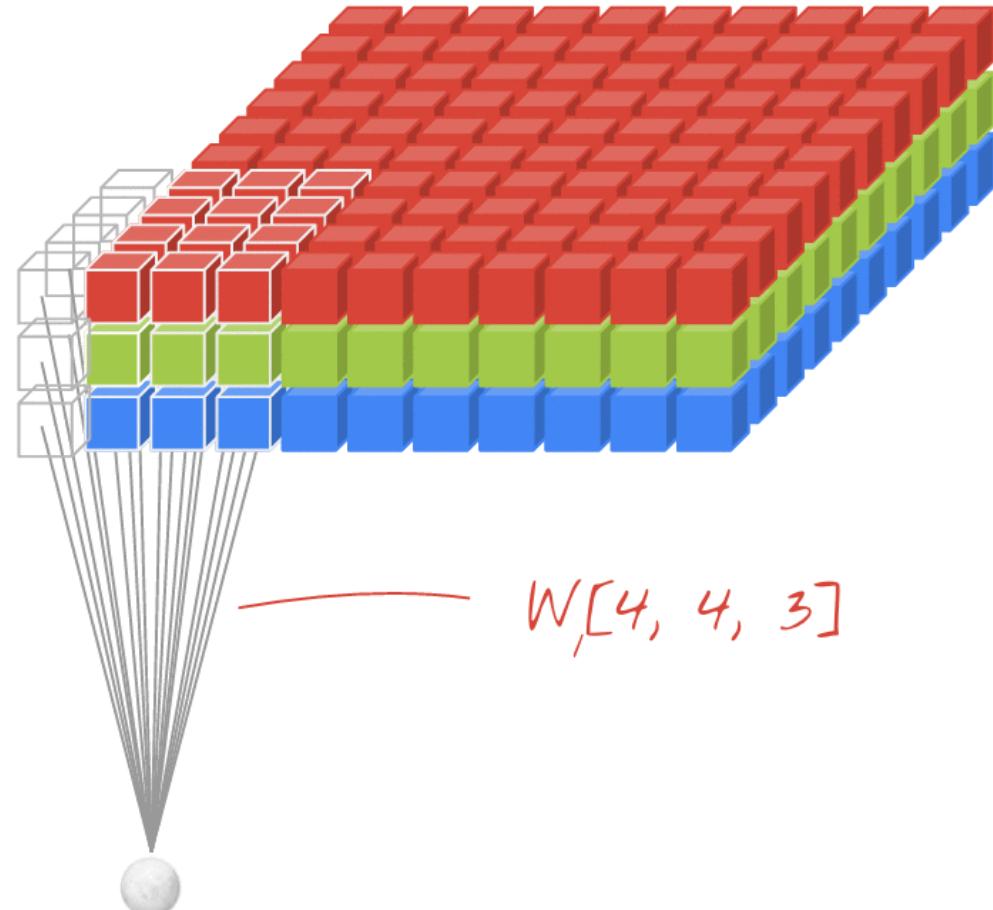
Horizontal

=

0	0	0	0
30	10	-10	-30
30	10	-10	-30
0	0	0	0

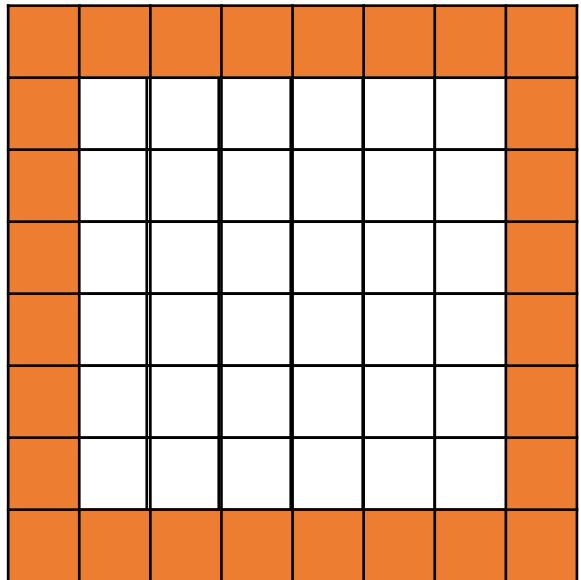
## 3.4 Convolutional layer

### Convolutional layer (Conv)

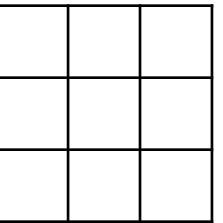


# 3.4 Convolutional layer

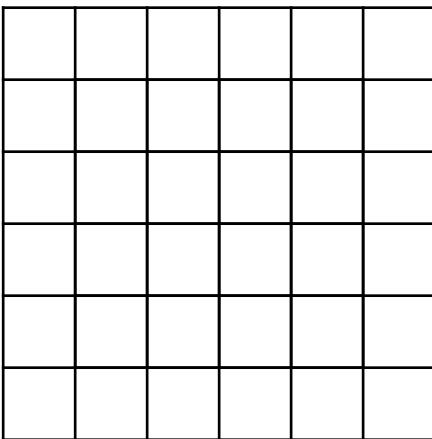
## Padding



\*



=



**Size of input data:**  $n \times n$ ,

**Size of filter:**  $f \times f$

**Size of output:**  $(n - f + 1) \times (n - f + 1)$

**Example:**  $6 - 3 + 1 = 4$ , hence  $4 \times 4$ , size is reduced!

**Use padding, extra border of 1 all around, ( $p = 1$ ) gives output of**  
 $(n + 2p - f + 1) \times (n + 2p - f + 1) = 6 \times 6$  (same as original data size)

## 3.4 Convolutional layer

### Padding

- “**Valid**” (no padding):  $n \times n * f \times f \rightarrow (n - f + 1) \times (n - f + 1)$ 
  - $6 \times 6 * 3 \times 3 \rightarrow 4 \times 4$
- “**Same**”: Pad so that output size is the same as the input size
  - $n + 2p - f + 1 = n \Rightarrow p = \frac{f-1}{2}$
  - $f = 3, p = \frac{3-1}{2} = 1, \text{ or } f = 5, p = 2$

# 3.4 Convolutional layer



## Stride

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

$$\begin{array}{c} \text{Input: } 7 \times 7 \\ \text{Kernel: } 3 \times 3 \\ \text{Stride: } 2 \\ \text{Output: } 3 \times 3 \end{array}$$

\*

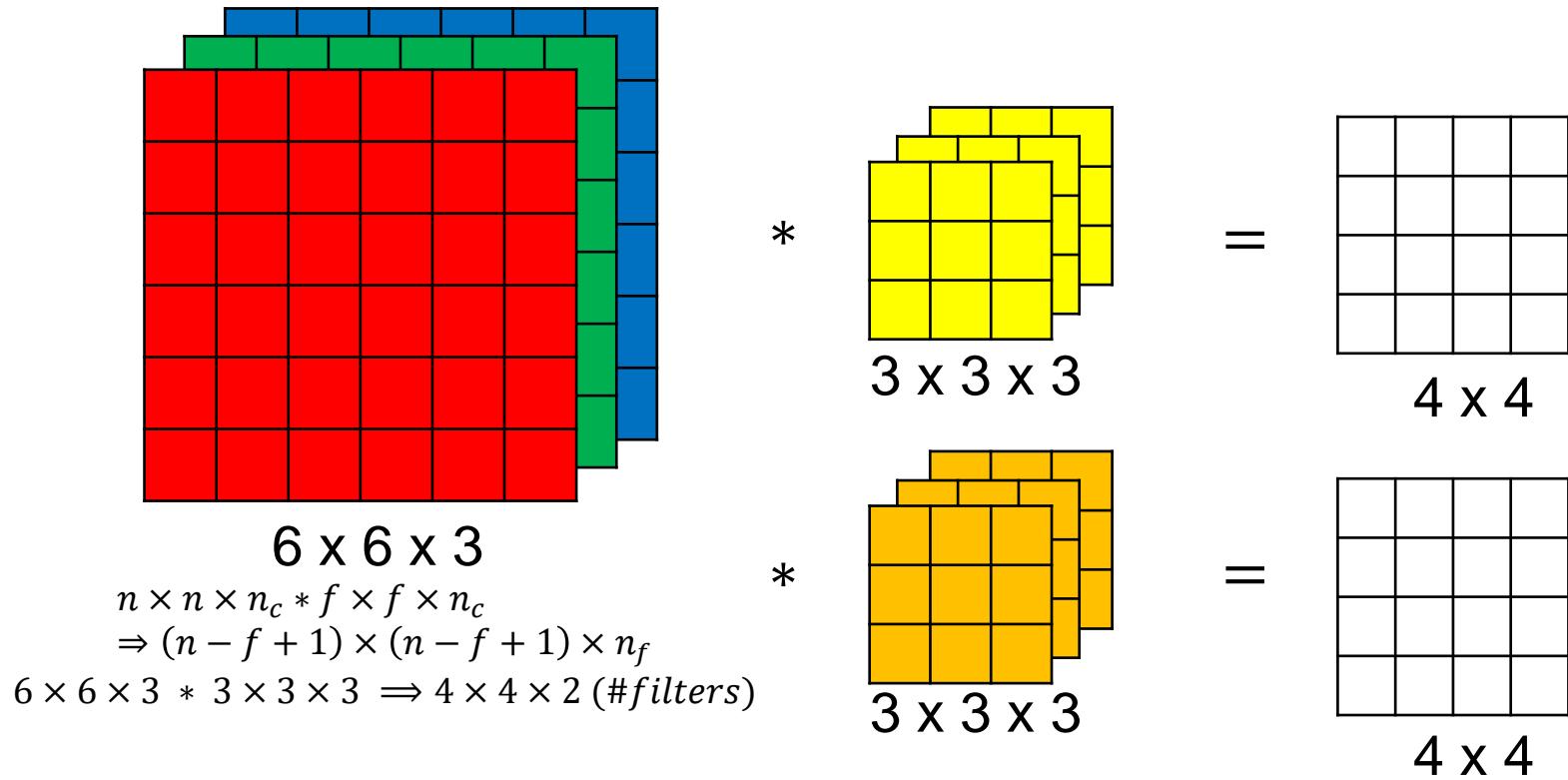
$$\begin{bmatrix} 3 & 4 & 4 \\ 1 & 0 & 2 \\ -1 & 0 & 3 \end{bmatrix} = \begin{bmatrix} 91 & 100 & 83 \\ 69 & 91 & 127 \\ 44 & 71 & 74 \end{bmatrix}$$

$$n \times n * f \times f \text{ (padding } p, \text{ stride } s) \Rightarrow \left\lfloor \frac{n + 2p - f}{s} + 1 \right\rfloor \times \left\lfloor \frac{n + 2p - f}{s} + 1 \right\rfloor$$
$$(7 + 0 - 3)/2 + 1 = 4/2 + 1 = 3$$

# 3.4 Convolutional layer

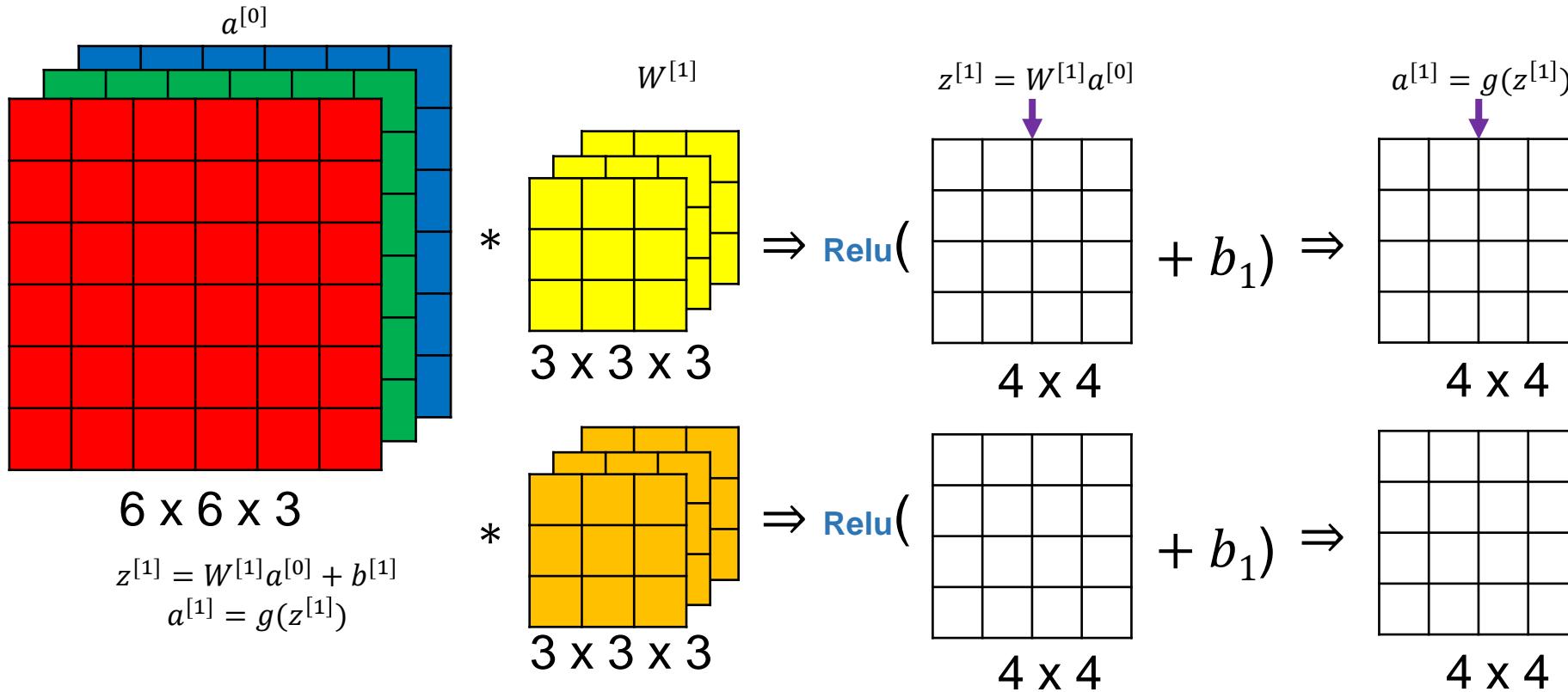


## Multiple Filter (Convs)



# 3.4 Convolutional layer

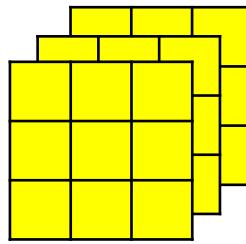
## Multiple Filter (Convs)



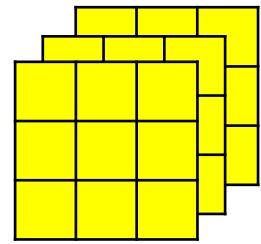
# 3.4 Convolutional layer

## No. Parameters

- If you have 10 filters that are  $3 \times 3 \times 3$  in one layer of a neural network, how many parameters does that layer have?

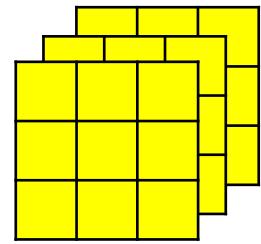


$3 \times 3 \times 3$



$3 \times 3 \times 3$

...



$3 \times 3 \times 3$

- $(3 \times 3 \times 3 + 1) \times 10 = 280$  parameters
- No matter how large the input image is, the number of parameters is fixed to 280 for 10 filters of size  $3 \times 3 \times 3$ .

# 3.4 Convolutional layer

## Summary

If layer 1 is a convolution layer:

**Activations:**  $a^{[l]} \Rightarrow n_H^{[l]} \times n_W^{[l]} \times n_C^{[l]}$

- $f^{[l]} = \text{filter size}$   
**Weights:**  $f^{[l]} \times f^{[l]} \times n_C^{[l-1]} \times n_C^{[l]}$
- $p^{[l]} = \text{padding}$   
**Bias:**  $n_C^{[l]}$
- $s^{[l]} = \text{stride}$
- $n_C^{[l]} = \text{number of filters}$
- Each filter is:  $f^{[l]} \times f^{[l]} \times n_C^{[l-1]}$
- Input Size:  $n_H^{[l-1]} \times n_W^{[l-1]} \times n_C^{[l-1]}$
- Output Size:  $n_H^{[l]} \times n_W^{[l]} \times n_C^{[l]}$
- $n_H^{[l]} = \left\lfloor \frac{n_H^{[l-1]} + 2p^{[l]} - f^{[l]}}{s^{[l]}} + 1 \right\rfloor$
- $n_W^{[l]} = \left\lfloor \frac{n_W^{[l-1]} + 2p^{[l]} - f^{[l]}}{s^{[l]}} + 1 \right\rfloor$

# 3.4 Convolutional layer

## Summary

If layer 1 is a convolution layer:

$$\text{Activations: } a^{[l]} \Rightarrow n_H^{[l]} \times n_W^{[l]} \times n_C^{[l]}$$

- $f^{[l]} = \text{filter size}$   
**Weights:**  $f^{[l]} \times f^{[l]} \times n_C^{[l-1]} \times n_C^{[l]}$
- $p^{[l]} = \text{padding}$   
**Bias:**  $n_C^{[l]}$
- $s^{[l]} = \text{stride}$
- $n_C^{[l]} = \text{number of filters}$
- Each filter is:  $f^{[l]} \times f^{[l]} \times n_C^{[l-1]}$
- Input Size:  $n_H^{[l-1]} \times n_W^{[l-1]} \times n_C^{[l-1]}$
- Output Size:  $n_H^{[l]} \times n_W^{[l]} \times n_C^{[l]}$
- $n_H^{[l]} = \left\lfloor \frac{n_H^{[l-1]} + 2p^{[l]} - f^{[l]}}{s^{[l]}} + 1 \right\rfloor$
- $n_W^{[l]} = \left\lfloor \frac{n_W^{[l-1]} + 2p^{[l]} - f^{[l]}}{s^{[l]}} + 1 \right\rfloor$

# 3.4 Max Pooling

1	3	2	1
2	9	1	1
1	3	2	3
5	6	1	2

Hyperparameters

$f = 2$

$s = 2$

Max Pooling

9	2
6	3

No parameters!

# 3.4 Max Pooling

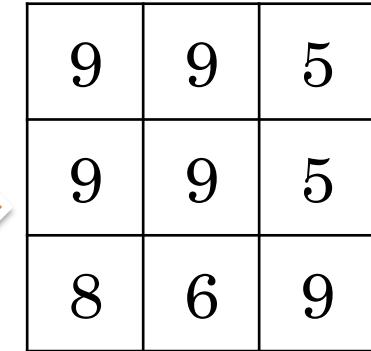


Hyperparameters

$$f = 3$$

$$s = 1$$

Max Pooling



The result of max pooling with a filter size of 3 and stride of 1 is a 3x3 matrix. The values are:

9	9	5
9	9	5
8	6	9

$3 \times 3$

Note: For multiple channels, the above max pooling is done for each channel



Max Pool

Filter - (2 x 2)  
Stride - (2, 2)



The result of max pooling with a filter size of 2 and stride of 2 is a 2x2 matrix. The values are:

9	7
8	6

### 3.4 Average Pooling



1	3	2	1
2	9	1	1
1	4	2	3
5	6	1	2

Hyperparameters

$$f = 2$$

$$s = 2$$



375	125
4	2

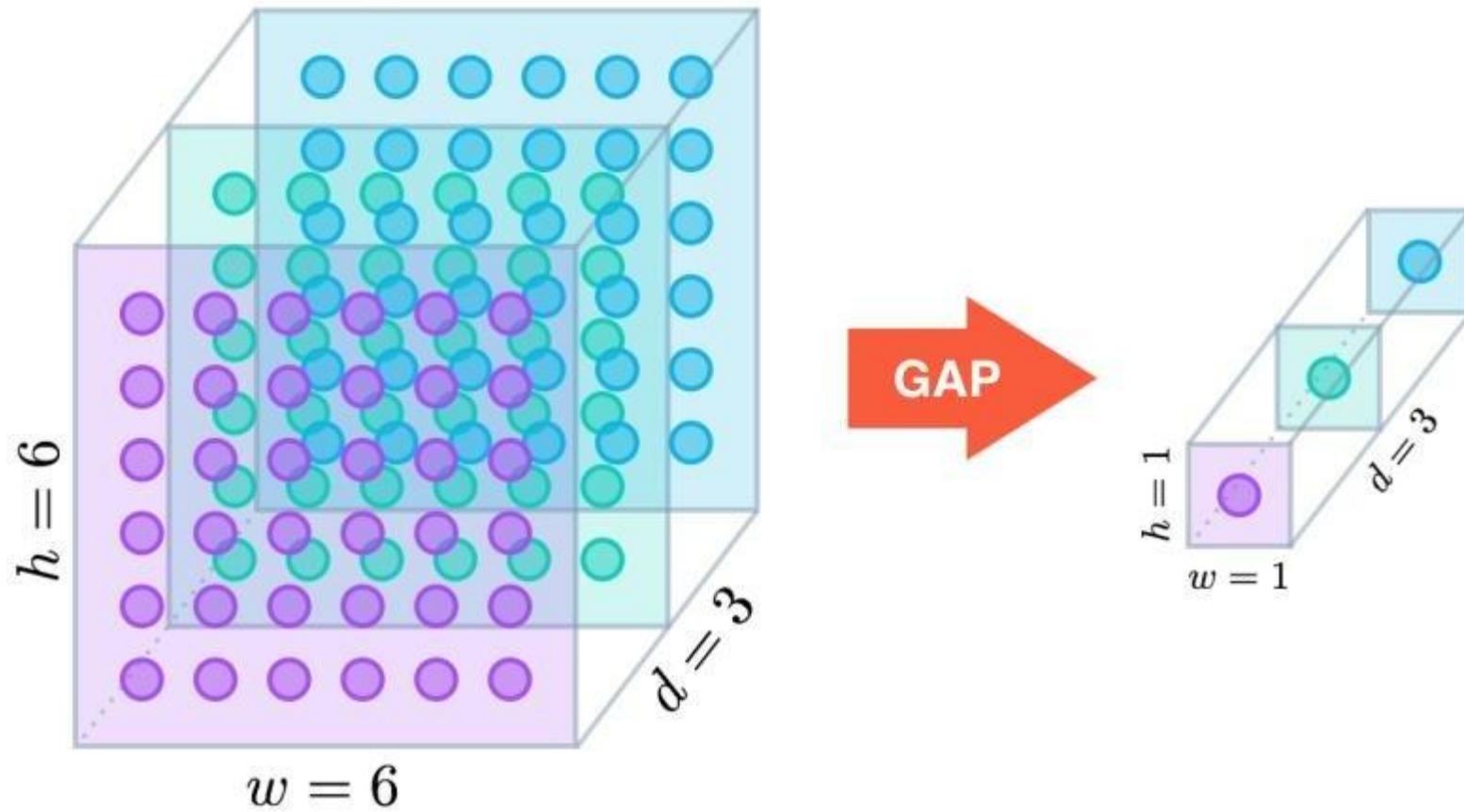
2	2	7	3
9	4	6	1
8	5	2	4
3	1	2	6

Average Pool

Filter -  $(2 \times 2)$   
Stride -  $(2, 2)$

4.25	4.25
4.25	3.5

### 3.4 Global Pooling



## Hyperparameters

- $f$ : filter size
- $s$ : stride
- Max or average pooling
- Usually,  $p = 0$ , no padding

$$n_H \times n_W \times n_C \rightarrow \left\lfloor \frac{n_H - f}{s} + 1 \right\rfloor \times \left\lfloor \frac{n_H - f}{s} + 1 \right\rfloor \times n_C$$