

# CNN for CV

AI for CV Group  
2020



# Week 6. CNN I. Outline & Layers

# Contents:

## I. CNN Outlines

- A. What CNN does in CV
- B. CNN Outlines

## II. Fundamental Layers

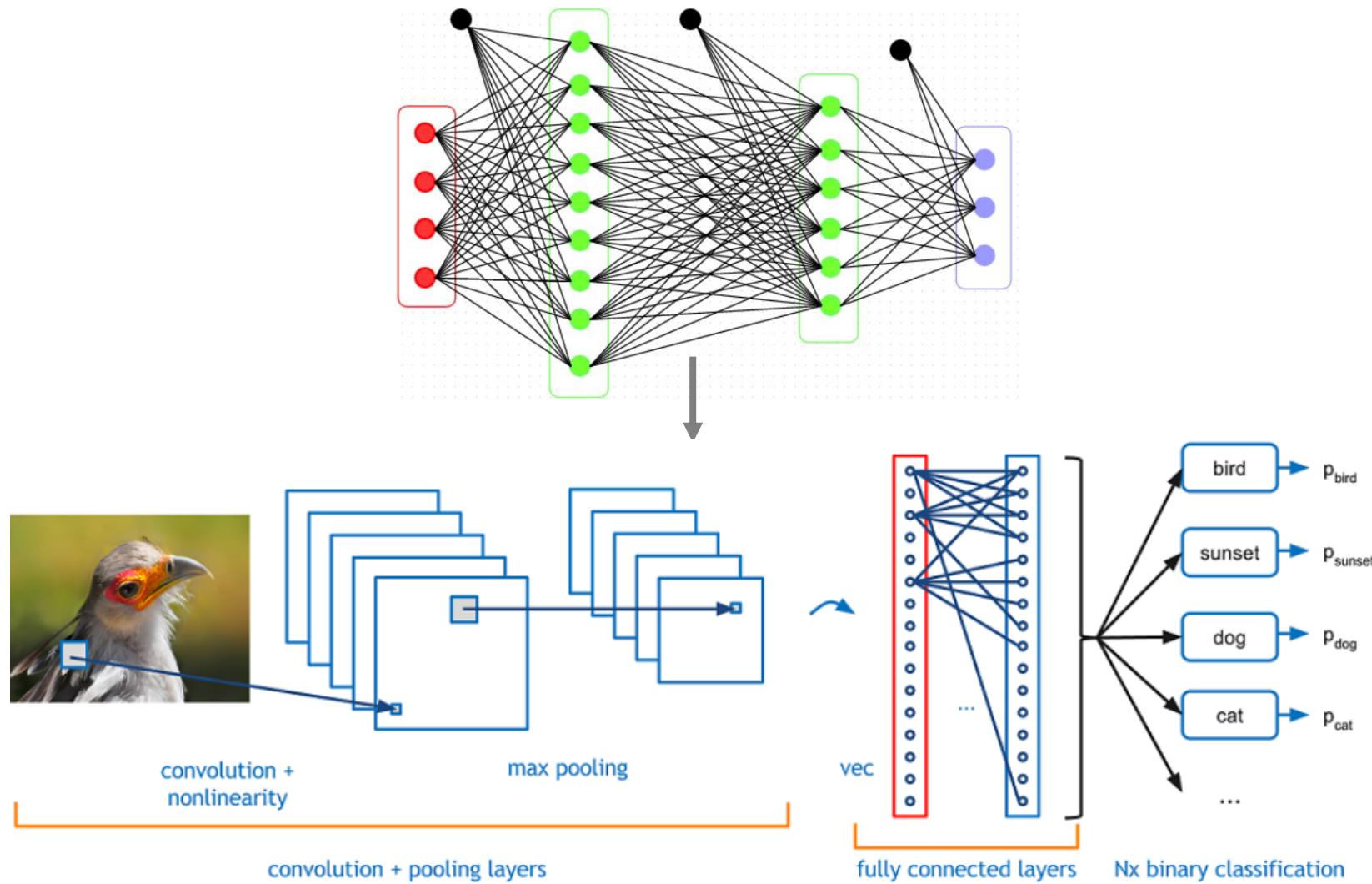
- C. Layers Concepts & Implementations

## III. Functional Layers

- D. Functional Layers Concepts & Implementations

# I. CNN Outlines

# I. CNN Outlines



# I. CNN Outlines

## A. What CNN does in CV:

# I. CNN Outlines

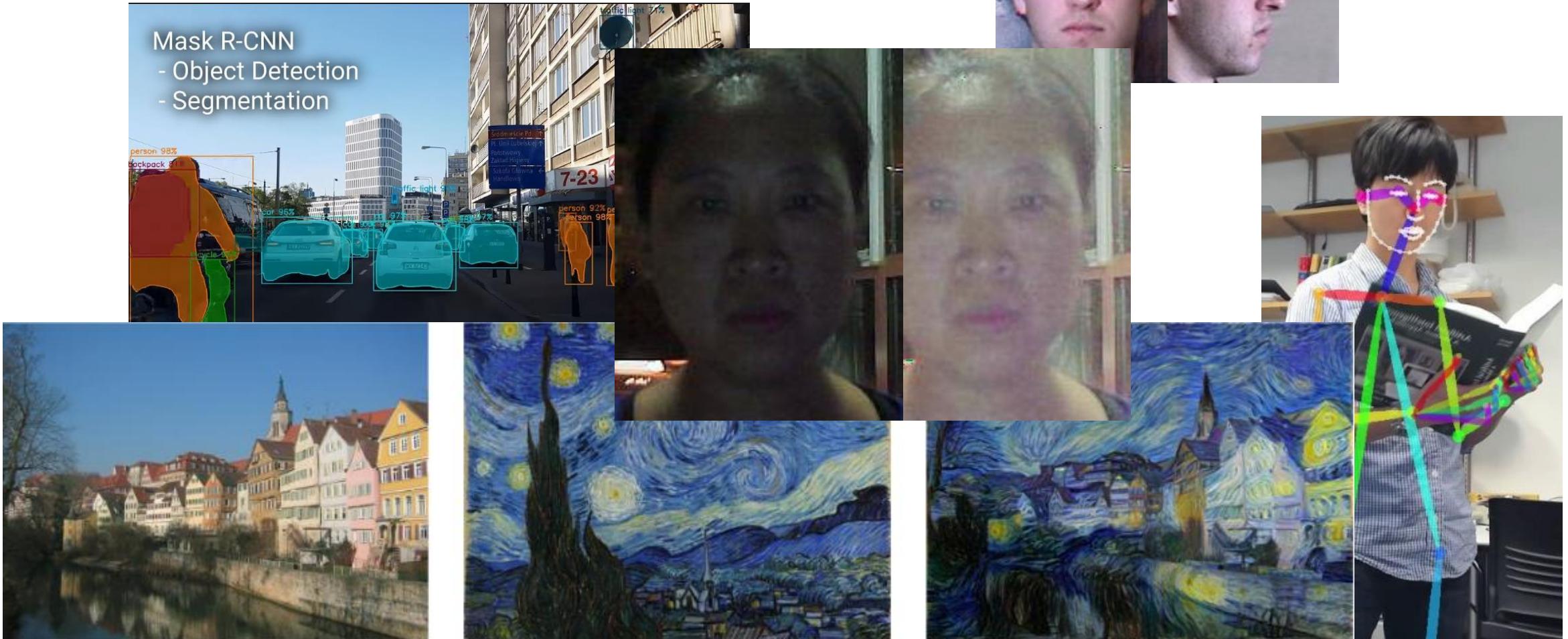
## A. What CNN does in CV:

**Everything**

(Almost)

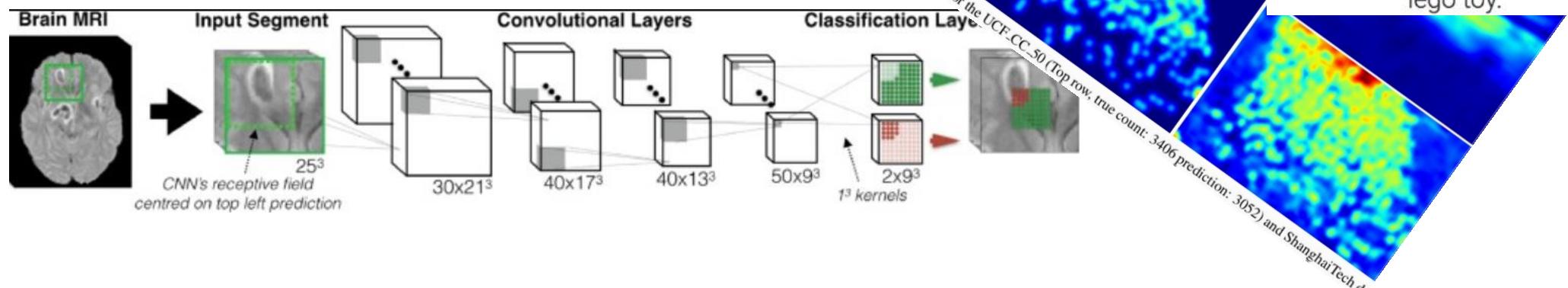
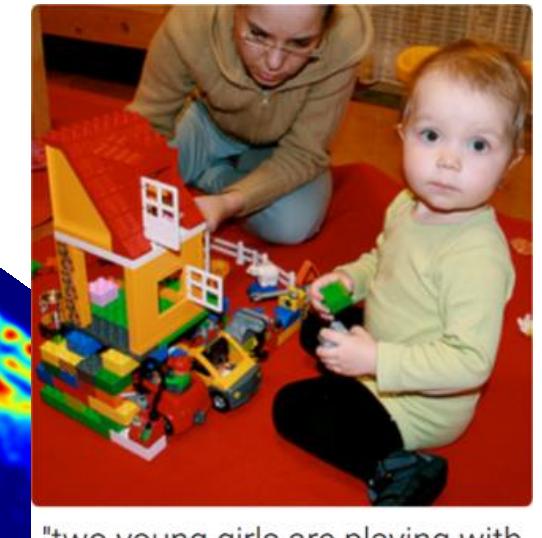
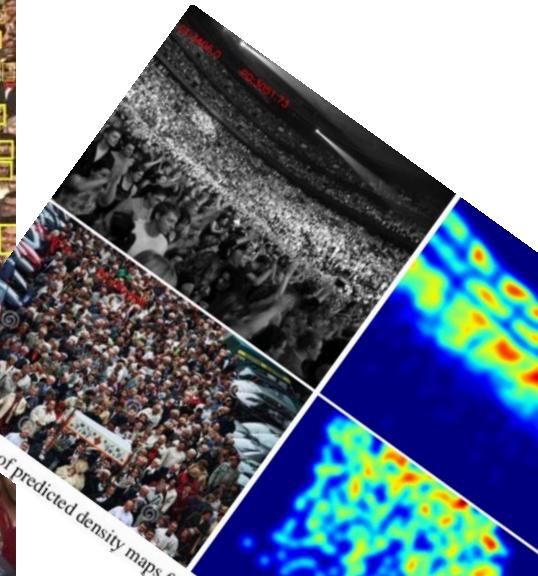
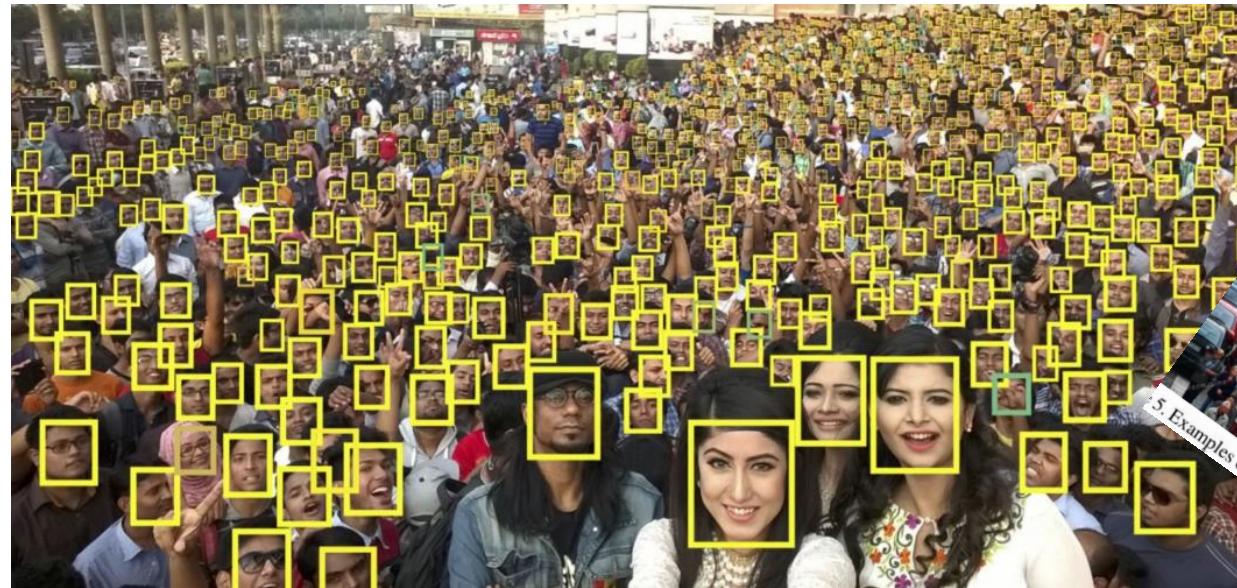
# I. CNN Outlines

## A. What CNN does in CV:



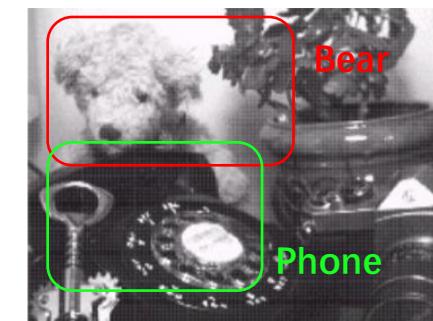
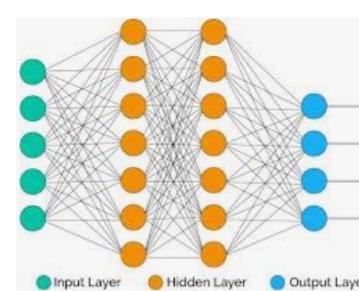
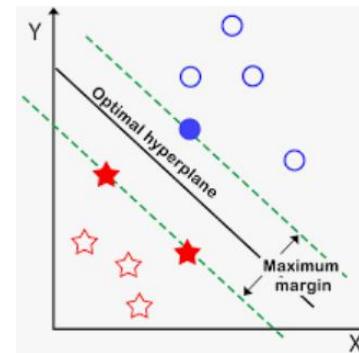
# I. CNN Outlines

## A. What CNN does in CV:



# I. CNN Outlines

## B. CNN Outlines: Old Fashion Way:



Input



Get Manually

Feature



Choose Carefully

Classic ML Tools



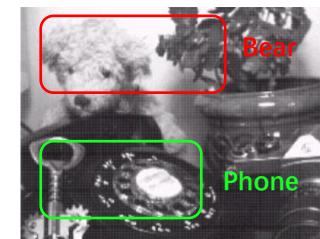
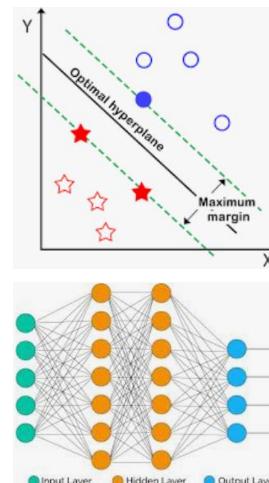
Use Hopelessly

Target

# I. CNN Outlines

## B. CNN Outlines:

**Old Fashion Way:**



Input

Feature

Classic ML  
Tools

Target

Get  
Manually

Choose  
Carefully

Use  
Hopelessly

## Pros & Cons

- Mathematical
- Explainable
- Difficult & Messy
- Separated Processes
- Poor results
- Hard to learn & implement

# I. CNN Outlines

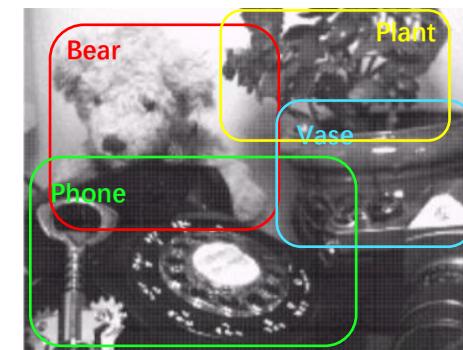
## B. CNN Outlines:

**CNN / Modern Way:**



Put In

Only Focus  
On  
This Part



Use It

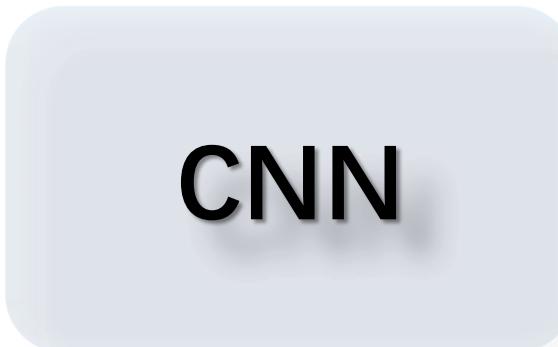
# I. CNN Outlines

## B. CNN Outlines:

**CNN / Modern Way:**

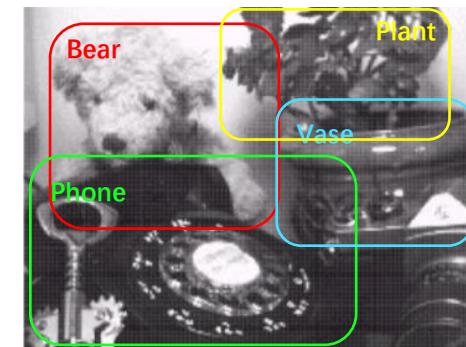
### Pros & Cons

- End-to-end
- Easy to implement
- No need to learn part by part
- Unexplainable



Put In

Only Focus  
On  
This Part

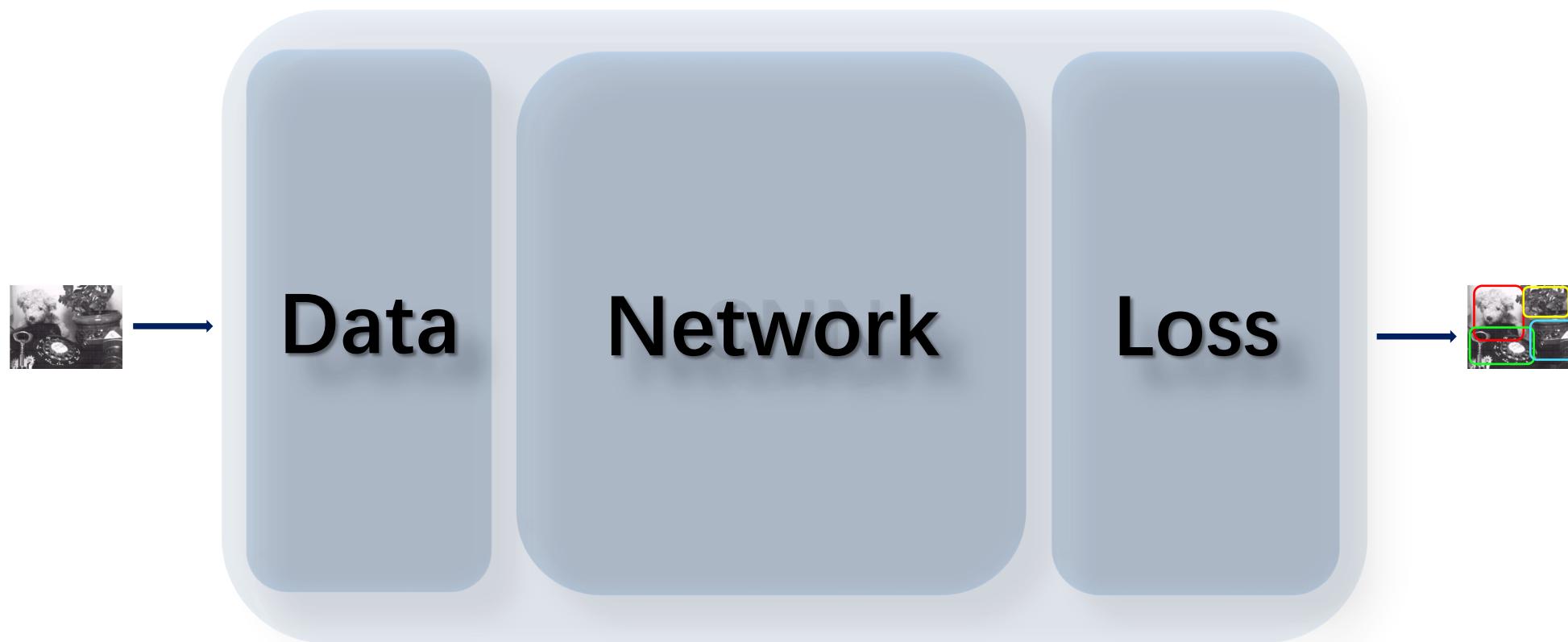


Use The Result

# I. CNN Outlines

## B. CNN Outlines:

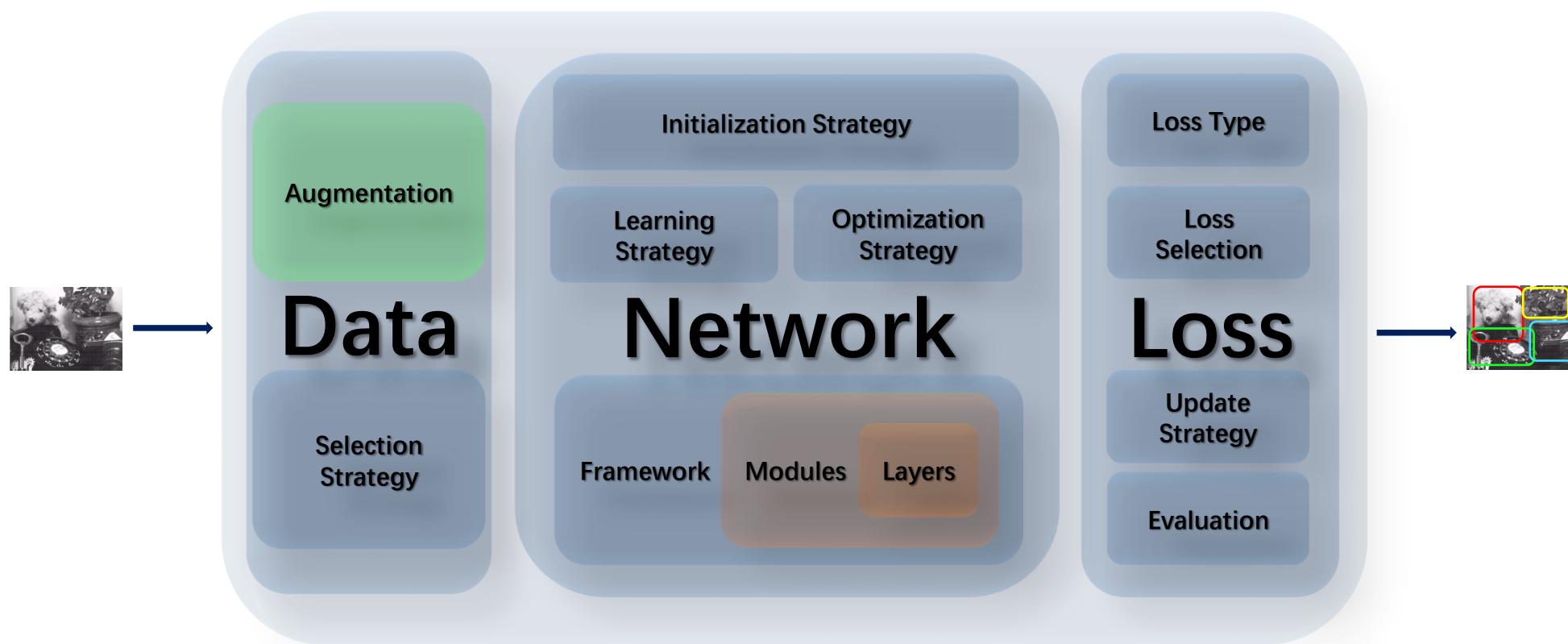
**CNN / Modern Way:**



# I. CNN Outlines

## B. CNN Outlines:

**CNN / Modern Way:**



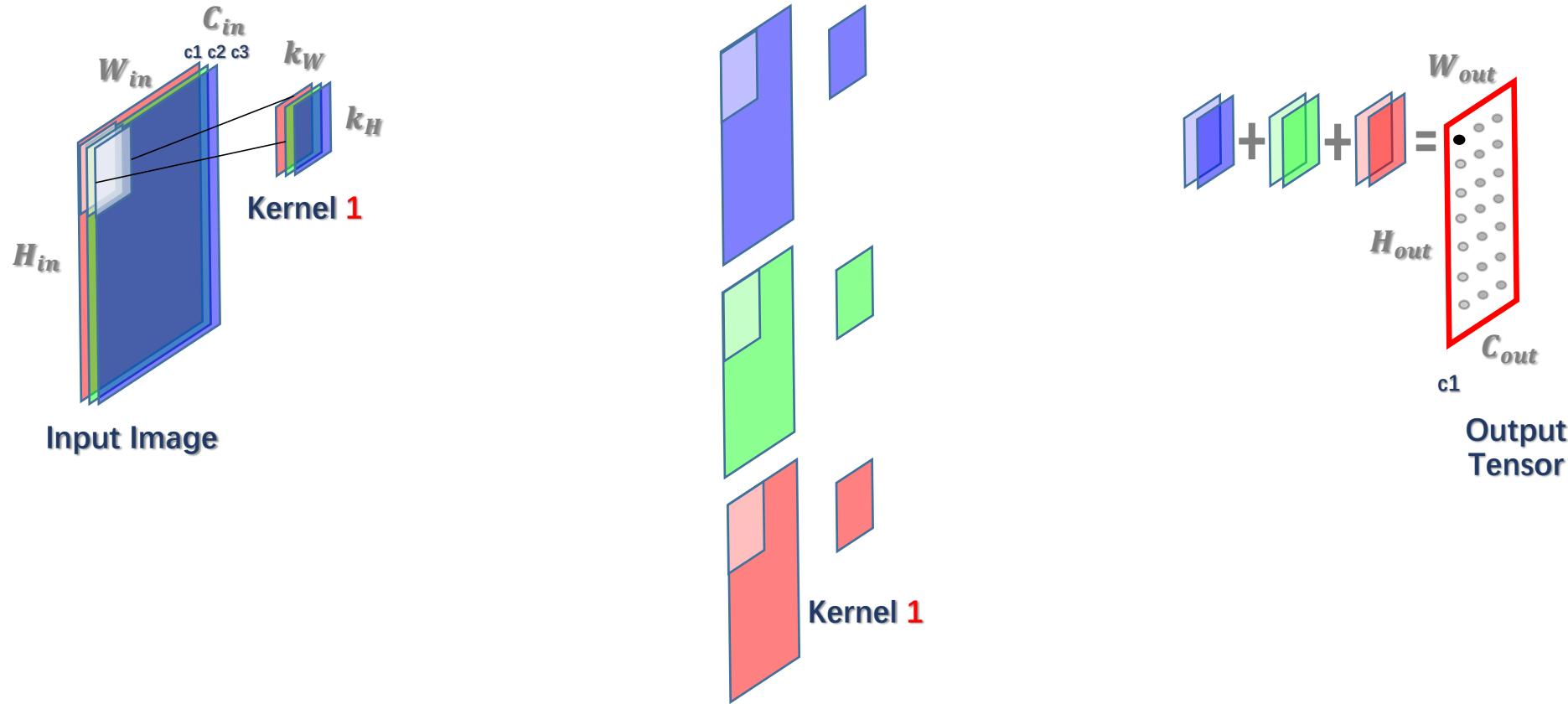
## II. Fundamental Layers



# II. Fundamental Layers

## C. Layers Concepts & Implements

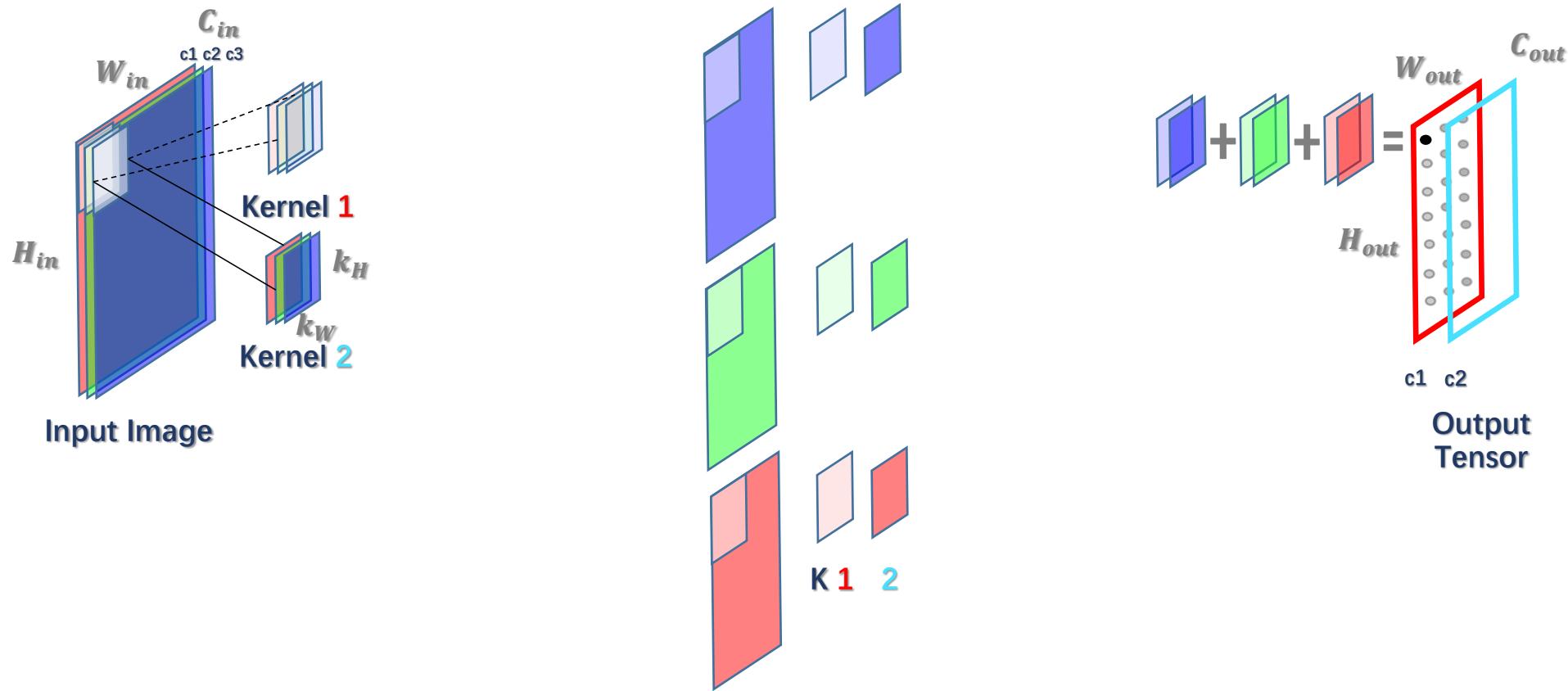
### C1. Convolution AGAIN!



# II. Fundamental Layers

## C. Layers Concepts & Implements

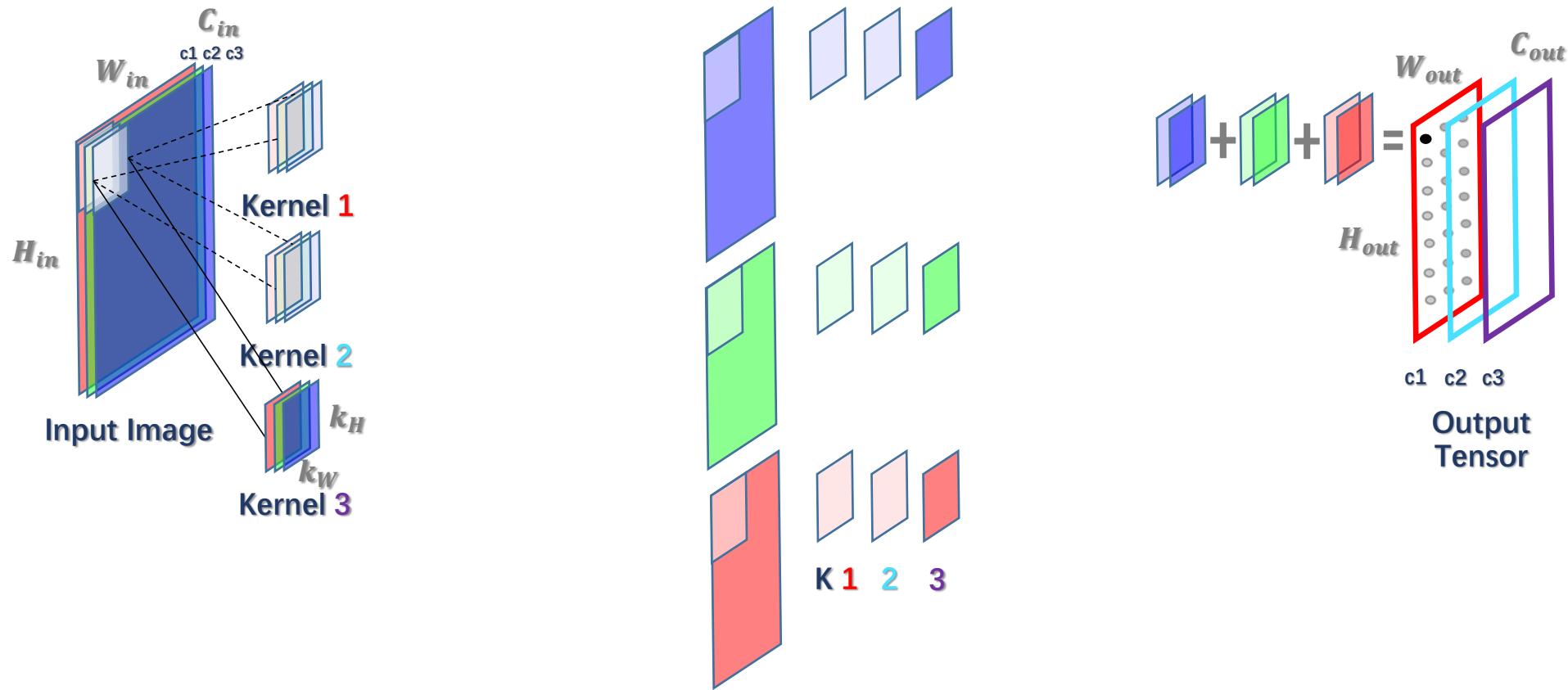
### C1. Convolution - General



# II. Fundamental Layers

## C. Layers Concepts & Implements

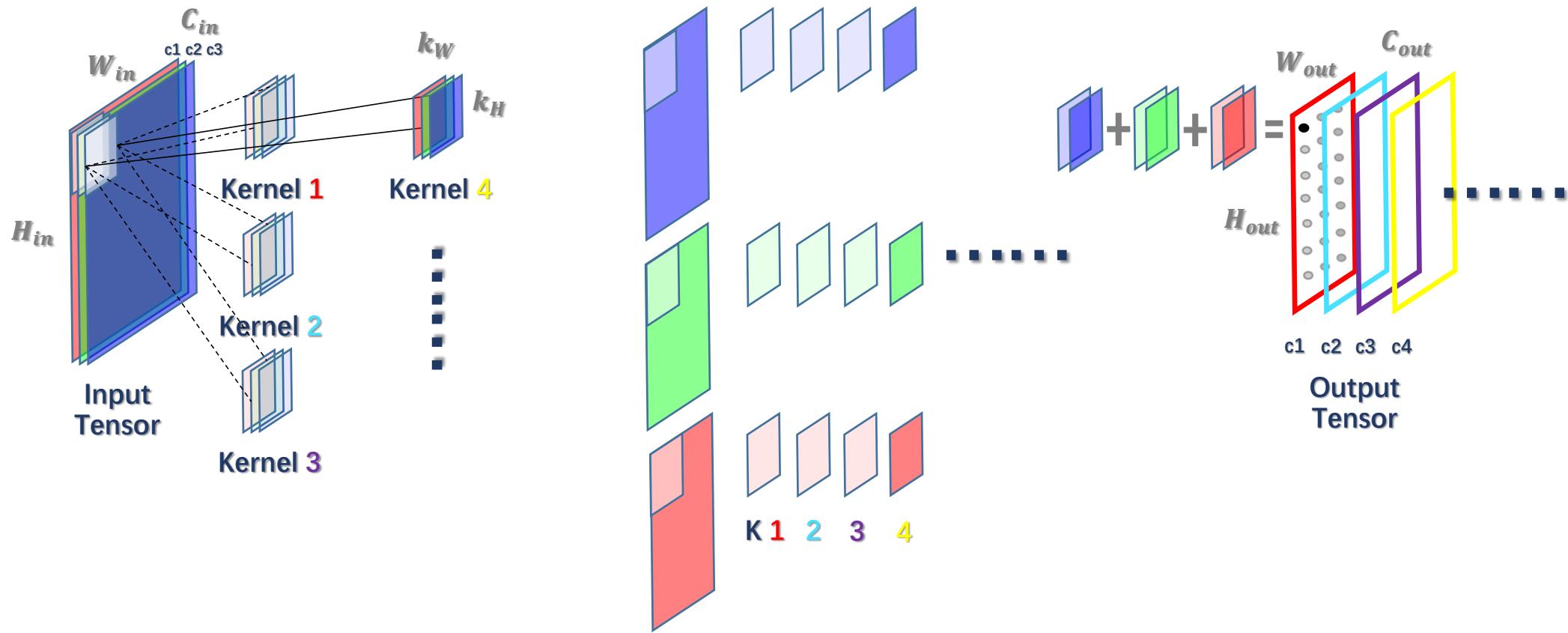
### C1. Convolution - General



# II. Fundamental Layers

## C. Layers Concepts & Implements

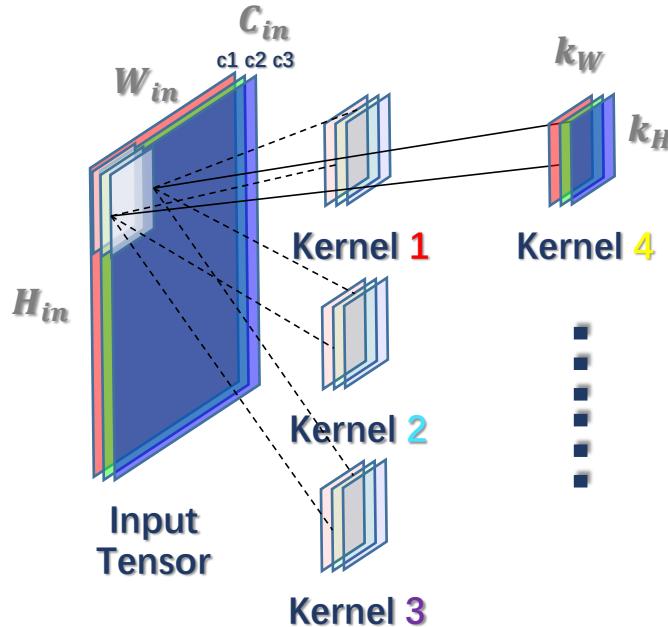
### C1. Convolution - General



# II. Fundamental Layers

## C. Layers Concepts & Implements

### C1. Convolution - General

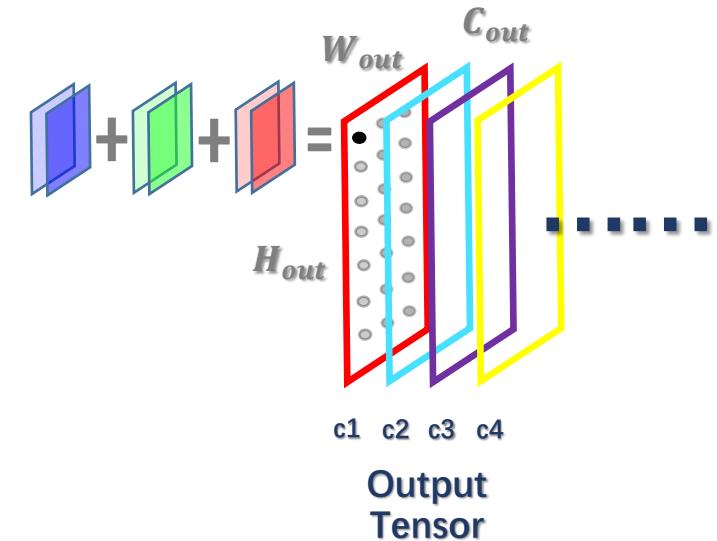


*Output Tensor Shape:*

$$C_{out} = \# \text{ of Kernel}$$

$$W_{out} = ?$$

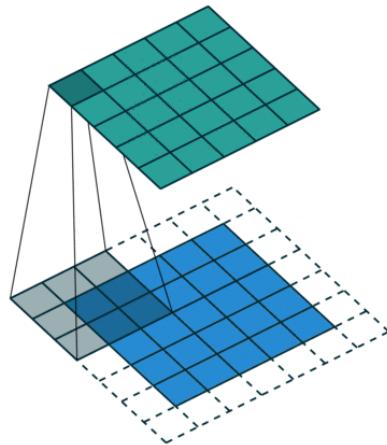
$$H_{out} = ?$$



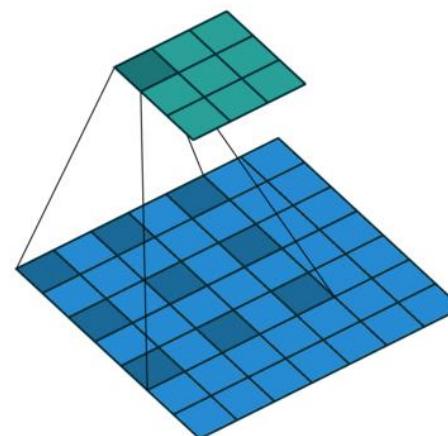
# II. Fundamental Layers

## C. Layers Concepts & Implements

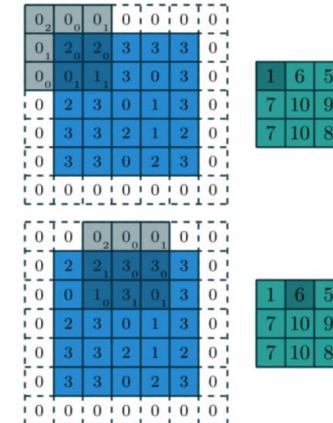
### C1. Convolution - General



**Padding:**  
Get same input size



**Dilation:**  
Bigger receptive field

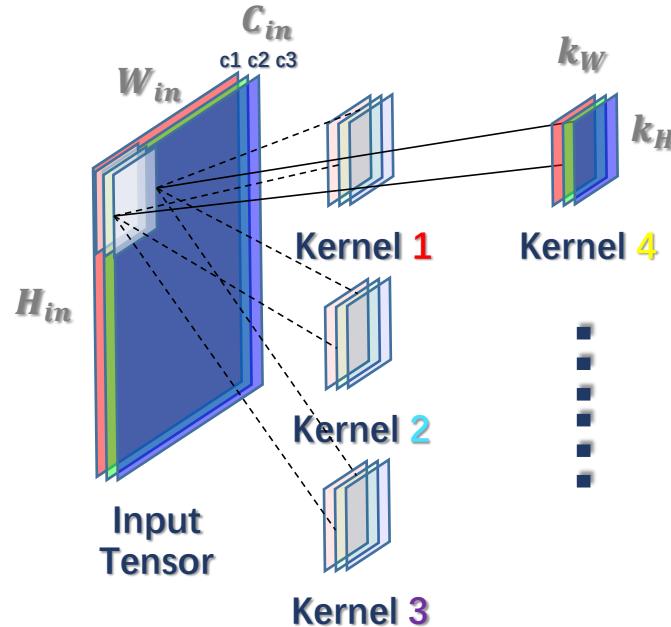


**Stride:**  
Downsample

# II. Fundamental Layers

## C. Layers Concepts & Implements

### C1. Convolution - General



*Output Tensor Shape:*

$C_{out}$  = # of Kernel

$$H_{out} = \left\lfloor \frac{H_{in} + 2 \times pad_H - dilation_H \times (k_H - 1) - 1}{stride_H} + 1 \right\rfloor$$

$$W_{out} = \left\lfloor \frac{W_{in} + 2 \times pad_W - dilation_W \times (k_W - 1) - 1}{stride_W} + 1 \right\rfloor$$

# II. Fundamental Layers

## C. Layers Concepts & Implements

### C1. Convolution - Calculation

#### Forward

$I_0$	$I_1$	$I_2$
$I_3$	$I_4$	$I_5$
$I_6$	$I_7$	$I_8$

$$\otimes = \begin{bmatrix} h_{00} & h_{01} \\ h_{10} & h_{11} \end{bmatrix}$$

$w_{00}$	$w_{01}$
$w_{10}$	$w_{11}$

$$\begin{cases} h_{00} = w_{00}I_0 + w_{01}I_1 + w_{10}I_3 + w_{11}I_4 \\ h_{01} = w_{00}I_1 + w_{01}I_2 + w_{10}I_4 + w_{11}I_5 \\ h_{10} = w_{00}I_3 + w_{01}I_4 + w_{10}I_6 + w_{11}I_7 \\ h_{11} = w_{00}I_4 + w_{01}I_5 + w_{10}I_7 + w_{11}I_8 \end{cases}$$

$$h = CI$$

# II. Fundamental Layers

## C. Layers Concepts & Implements

### C1. Convolution - Calculation

Forward

$$\mathbf{h} = \mathbf{C}\mathbf{I}$$

$$\begin{cases} h_{00} = w_{00}I_0 + w_{01}I_1 + w_{10}I_3 + w_{11}I_4 \\ h_{01} = w_{00}I_1 + w_{01}I_2 + w_{10}I_4 + w_{11}I_5 \\ h_{10} = w_{00}I_3 + w_{01}I_4 + w_{10}I_6 + w_{11}I_7 \\ h_{11} = w_{00}I_4 + w_{01}I_5 + w_{10}I_7 + w_{11}I_8 \end{cases}$$

$$\begin{array}{|c|c|c|} \hline I_0 & I_1 & I_2 \\ \hline I_3 & I_4 & I_5 \\ \hline I_6 & I_7 & I_8 \\ \hline \end{array} \quad \otimes \quad = \quad \begin{array}{|c|c|} \hline h_{00} & h_{01} \\ \hline h_{10} & h_{11} \\ \hline \end{array}$$
  
$$\begin{array}{|c|c|} \hline w_{00} & w_{01} \\ \hline w_{10} & w_{11} \\ \hline \end{array}$$

# II. Fundamental Layers

## C. Layers Concepts & Implements

### C1. Convolution - Calculation

#### Transposed Weight Matrix C

$$\begin{array}{|c|c|c|} \hline I_0 & I_1 & I_2 \\ \hline I_3 & I_4 & I_5 \\ \hline I_6 & I_7 & I_8 \\ \hline \end{array} \quad \otimes \quad = \quad \begin{array}{|c|c|} \hline h_{00} & h_{01} \\ \hline h_{10} & h_{11} \\ \hline \end{array}$$
  
$$\begin{array}{|c|c|} \hline w_{00} & w_{01} \\ \hline w_{10} & w_{11} \\ \hline \end{array}$$

# II. Fundamental Layers

## C. Layers Concepts & Implements

### C1. Convolution - Calculation

#### Backward / BP

$$Loss = \partial h_{00} + \partial h_{01} + \partial h_{10} + \partial h_{11}$$

$$\begin{array}{|c|c|c|}\hline I_0 & I_1 & I_2 \\ \hline I_3 & I_4 & I_5 \\ \hline I_6 & I_7 & I_8 \\ \hline\end{array} \quad \otimes \quad = \quad \begin{array}{|c|c|}\hline h_{00} & h_{01} \\ \hline h_{10} & h_{11} \\ \hline\end{array}$$
  
$$\begin{array}{|c|c|}\hline w_{00} & w_{01} \\ \hline w_{10} & w_{11} \\ \hline\end{array}$$

$$\begin{cases} h_{00} = w_{00}I_0 + w_{01}I_1 + w_{10}I_3 + w_{11}I_4 \\ h_{01} = w_{00}I_1 + w_{01}I_2 + w_{10}I_4 + w_{11}I_5 \\ h_{10} = w_{00}I_3 + w_{01}I_4 + w_{10}I_6 + w_{11}I_7 \\ h_{11} = w_{00}I_4 + w_{01}I_5 + w_{10}I_7 + w_{11}I_8 \end{cases}$$

$$\partial w = ?$$

$$\partial I = ?$$

# II. Fundamental Layers

## C. Layers Concepts & Implements

### C1. Convolution - Calculation

#### Backward / BP

$$Loss = \partial h_{00} + \partial h_{01} + \partial h_{10} + \partial h_{11}$$

$$\begin{array}{|c|c|c|}\hline I_0 & I_1 & I_2 \\ \hline I_3 & I_4 & I_5 \\ \hline I_6 & I_7 & I_8 \\ \hline\end{array} \quad \otimes \quad = \quad \begin{array}{|c|c|}\hline h_{00} & h_{01} \\ \hline h_{10} & h_{11} \\ \hline\end{array}$$

$$\begin{cases} h_{00} = w_{00}I_0 + w_{01}I_1 + w_{10}I_3 + w_{11}I_4 \\ h_{01} = w_{00}I_1 + w_{01}I_2 + w_{10}I_4 + w_{11}I_5 \\ h_{10} = w_{00}I_3 + w_{01}I_4 + w_{10}I_6 + w_{11}I_7 \\ h_{11} = w_{00}I_4 + w_{01}I_5 + w_{10}I_7 + w_{11}I_8 \end{cases}$$

$$\begin{array}{|c|c|}\hline w_{00} & w_{01} \\ \hline w_{10} & w_{11} \\ \hline\end{array}$$

$$\frac{\partial L}{\partial w} = \frac{\partial L}{\partial h} \frac{\partial h}{\partial w}$$

$$\frac{\partial L}{\partial I} = \frac{\partial L}{\partial h} \frac{\partial h}{\partial I}$$

$$\begin{cases} \partial w_{00} = \partial h_{00}I_0 + \partial h_{01}I_1 + \partial h_{10}I_3 + \partial h_{11}I_4 \\ \partial w_{01} = \partial h_{00}I_1 + \partial h_{01}I_2 + \partial h_{10}I_4 + \partial h_{11}I_5 \\ \partial w_{10} = \partial h_{00}I_3 + \partial h_{01}I_4 + \partial h_{10}I_6 + \partial h_{11}I_7 \\ \partial w_{11} = \partial h_{00}I_4 + \partial h_{01}I_5 + \partial h_{10}I_7 + \partial h_{11}I_8 \end{cases}$$

# II. Fundamental Layers

## C. Layers Concepts & Implements

### C1. Convolution - Calculation

Backward / BP

$$\begin{array}{|c|c|c|} \hline I_0 & I_1 & I_2 \\ \hline I_3 & I_4 & I_5 \\ \hline I_6 & I_7 & I_8 \\ \hline \end{array} \quad \otimes \quad = \quad \begin{array}{|c|c|} \hline h_{00} & h_{01} \\ \hline h_{10} & h_{11} \\ \hline \end{array}$$
  
$$\begin{array}{|c|c|} \hline w_{00} & w_{01} \\ \hline w_{10} & w_{11} \\ \hline \end{array}$$

$$\frac{\partial L}{\partial w} = \frac{\partial L}{\partial h} \frac{\partial h}{\partial w}$$
$$\frac{\partial L}{\partial I} = \frac{\partial L}{\partial h} \frac{\partial h}{\partial I}$$
$$\begin{aligned}\partial w_{00} &= \partial h_{00} \\ \partial w_{01} &= I_M \cong C \frac{\partial h_{01}}{\partial h_{10}} \\ \partial w_{10} &= \partial h_{10} \\ \partial w_{11} &= \partial h_{11}\end{aligned}$$

# II. Fundamental Layers

## C. Layers Concepts & Implements

### C1. Convolution - Calculation

$$\begin{array}{|c|c|c|} \hline I_0 & I_1 & I_2 \\ \hline I_3 & I_4 & I_5 \\ \hline I_6 & I_7 & I_8 \\ \hline \end{array} \quad \otimes \quad = \quad \begin{array}{|c|c|} \hline h_{00} & h_{01} \\ \hline h_{10} & h_{11} \\ \hline \end{array}$$

$$\partial L / \partial w = \frac{\partial L}{\partial h} \frac{\partial h}{\partial w}$$

$$\partial L / \partial I = \frac{\partial L}{\partial h} \frac{\partial h}{\partial I}$$

### Backward / BP

$$\begin{aligned} \partial w_{00} &= & \partial h_{00} \\ \partial w_{01} &= & \partial h_{01} \\ \partial w_{10} &= & I_c \cong C \quad \partial h_{10} \\ \partial w_{11} &= & \partial h_{11} \end{aligned}$$

# II. Fundamental Layers

## C. Layers Concepts & Implements

### C1. Convolution - Calculation

#### Backward / BP

$$Loss = \partial h_{00} + \partial h_{01} + \partial h_{10} + \partial h_{11}$$

$$\begin{array}{|c|c|c|}\hline I_0 & I_1 & I_2 \\ \hline I_3 & I_4 & I_5 \\ \hline I_6 & I_7 & I_8 \\ \hline\end{array} \quad \otimes \quad = \quad \begin{array}{|c|c|}\hline h_{00} & h_{01} \\ \hline h_{10} & h_{11} \\ \hline\end{array}$$

$$\begin{cases} h_{00} = w_{00}I_0 + w_{01}I_1 + w_{10}I_3 + w_{11}I_4 \\ h_{01} = w_{00}I_1 + w_{01}I_2 + w_{10}I_4 + w_{11}I_5 \\ h_{10} = w_{00}I_3 + w_{01}I_4 + w_{10}I_6 + w_{11}I_7 \\ h_{11} = w_{00}I_4 + w_{01}I_5 + w_{10}I_7 + w_{11}I_8 \end{cases}$$

$$\partial w = \frac{\partial L}{\partial h} \frac{\partial h}{\partial w}$$

$$\partial I = \frac{\partial L}{\partial h} \frac{\partial h}{\partial I}$$

$$\begin{cases} \partial I_0 = \partial h_{00}w_{00} \\ \partial I_1 = \partial h_{00}w_{01} + \partial h_{01}w_{00} \\ \partial I_2 = \partial h_{01}w_{01} \\ \partial I_3 = \partial h_{00}w_{10} + \partial h_{10}w_{00} \\ \vdots \end{cases}$$

# II. Fundamental Layers

## C. Layers Concepts & Implements

### C1. Convolution - Calculation

#### Backward / BP

$$\begin{array}{|c|c|c|} \hline I_0 & I_1 & I_2 \\ \hline I_3 & I_4 & I_5 \\ \hline I_6 & I_7 & I_8 \\ \hline \end{array} \quad \otimes \quad = \quad \begin{array}{|c|c|} \hline h_{00} & h_{01} \\ \hline h_{10} & h_{11} \\ \hline \end{array}$$
  
$$\begin{array}{|c|c|} \hline w_{00} & w_{01} \\ \hline w_{10} & w_{11} \\ \hline \end{array}$$

$$\partial_w = \frac{\partial L}{\partial h} \frac{\partial h}{\partial w}$$
  
$$\partial_I = \frac{\partial L}{\partial h} \frac{\partial h}{\partial I}$$

$$\begin{aligned}\partial I_0 &= \frac{\partial h_{00}}{\partial h} \\ \partial I_1 &= C^T \frac{\partial h_{01}}{\partial h} \\ \partial I_2 &= \frac{\partial h_{10}}{\partial h} \\ \partial I_3 &= \frac{\partial h_{11}}{\partial h} \\ &\vdots\end{aligned}$$

# II. Fundamental Layers

## C. Layers Concepts & Implements

### C1. Convolution - Calculation

$I_0$	$I_1$	$I_2$
$I_3$	$I_4$	$I_5$
$I_6$	$I_7$	$I_8$

$$\otimes = \begin{bmatrix} h_{00} & h_{01} \\ h_{10} & h_{11} \end{bmatrix}$$

$w_{00}$	$w_{01}$
$w_{10}$	$w_{11}$

$$\partial w = \frac{\partial L}{\partial h} \frac{\partial h}{\partial w}$$

$$\partial I = \frac{\partial L}{\partial h} \frac{\partial h}{\partial I}$$

### Backward / BP

$$\begin{aligned}\partial I_0 &= \frac{\partial h_{00}}{\partial h_{00}} \\ \partial I_1 &= \frac{\partial h_{01}}{\partial h_{01}} \\ \partial I_2 &= C^T \frac{\partial h_{10}}{\partial h_{10}} \\ \partial I_3 &= \frac{\partial h_{11}}{\partial h_{11}} \\ &\vdots\end{aligned}$$

# II. Fundamental Layers

## C. Layers Concepts & Implements

### C1. Convolution - Calculation

$$\partial I = \frac{\partial L}{\partial h} \frac{\partial h}{\partial I}$$

$I_0$	$I_1$	$I_2$
$I_3$	$I_4$	$I_5$
$I_6$	$I_7$	$I_8$

$$\otimes = \begin{bmatrix} h_{00} & h_{01} \\ h_{10} & h_{11} \end{bmatrix}$$

$w_{00}$	$w_{01}$
$w_{10}$	$w_{11}$

Thinking

*Conv Forward*

$I \rightarrow h$

$3x3 \rightarrow 2x2$

*Downsample*

$$\begin{aligned}\partial I_0 &= \frac{\partial h_{00}}{\partial I} \\ \partial I_1 &= \frac{\partial h_{01}}{\partial I} \\ \partial I_2 &= \mathbf{C}^T \frac{\partial h_{10}}{\partial I} \\ \partial I_3 &= \frac{\partial h_{11}}{\partial I} \\ &\vdots\end{aligned}$$

# II. Fundamental Layers

## C. Layers Concepts & Implements

### C1. Convolution - Calculation

$$\partial I = \frac{\partial L}{\partial h} \frac{\partial h}{\partial I}$$

$I_0$	$I_1$	$I_2$
$I_3$	$I_4$	$I_5$
$I_6$	$I_7$	$I_8$

$$\otimes \quad = \quad \begin{matrix} h_{00} & h_{01} \\ h_{10} & h_{11} \end{matrix}$$

$w_{00}$	$w_{01}$
$w_{10}$	$w_{11}$

Thinking

*Conv Forward*

$I \rightarrow h$

$3x3 \rightarrow 2x2$

*Downsample*

$$\begin{aligned}\partial I_0 &= \frac{\partial h_{00}}{\partial I} \\ \partial I_1 &= \frac{\partial h_{01}}{\partial I} \\ \partial I_2 &= C^T \frac{\partial h_{10}}{\partial I} \\ \partial I_3 &= \frac{\partial h_{11}}{\partial I} \\ &\vdots\end{aligned}$$

*Conv Backward*

$\partial h \rightarrow \partial I$

$2x2 \rightarrow 3x3$

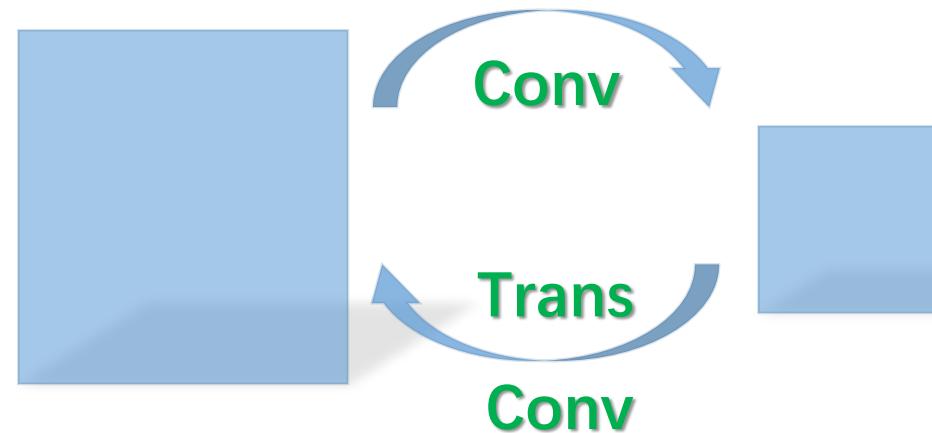
*UPSAMPLE*

# II. Fundamental Layers

## C. Layers Concepts & Implements

### C1. Convolution

-Transposed Conv / Deconv / Fractionally Strided Conv

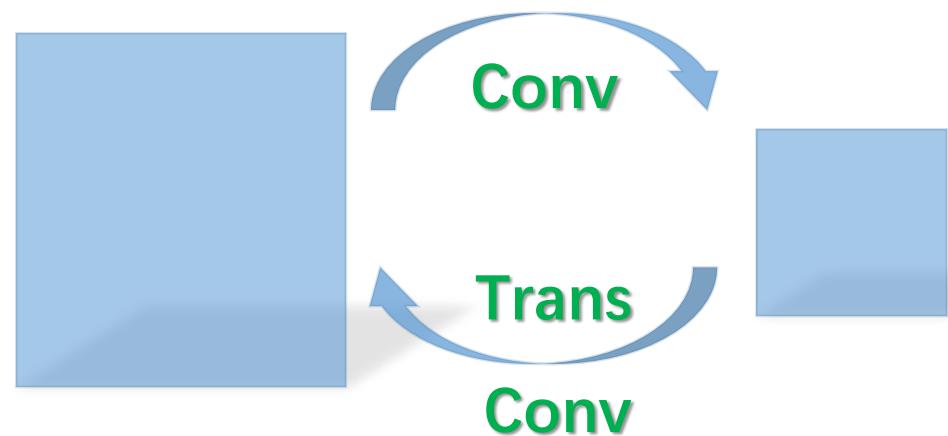


# II. Fundamental Layers

## C. Layers Concepts & Implements

### C1. Convolution

-Transposed Conv / Deconv / Fractionally Strided Conv

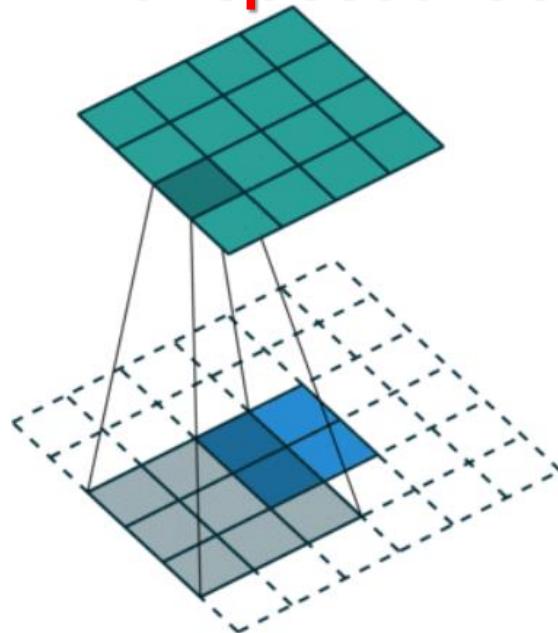


# II. Fundamental Layers

## C. Layers Concepts & Implements

### C1. Convolution

-Transposed Conv / Deconv / Fractionally Strided Conv



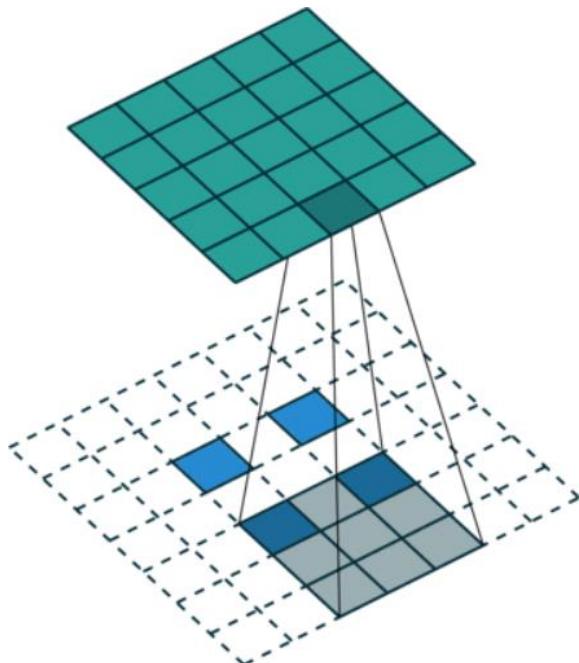
Transposed Conv + Padding  
↔  
Conv

# II. Fundamental Layers

## C. Layers Concepts & Implements

### C1. Convolution

-Transposed Conv / Deconv / Fractionally Strided Conv



Conv with stride > 0

Add 0 among elements  
when transposed conv

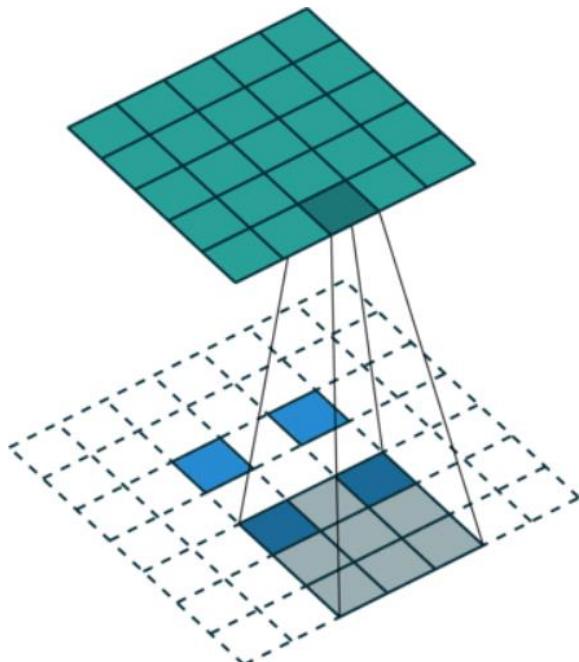
$$\text{Stride}_{\text{trans\_conv}} = \frac{1}{\text{Stride}_{\text{conv}}}$$

# II. Fundamental Layers

## C. Layers Concepts & Implements

### C1. Convolution

-Transposed Conv / Deconv / Fractionally Strided Conv



*Output Tensor Shape:*

$$C_{out} = \# \text{ of Kernel}$$

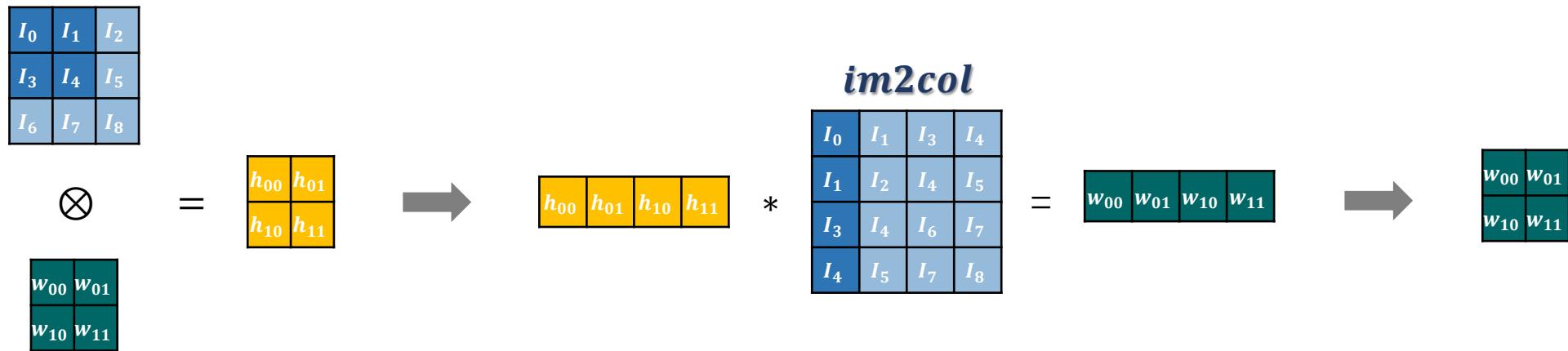
$$H_{out} = (H_{in}-1) \times stride_H - 2 \times pad_H + k_H + out\_pad_H$$

$$W_{out} = (W_{in}-1) \times stride_W - 2 \times pad_W + k_W + out\_pad_W$$

# II. Fundamental Layers

## C. Layers Concepts & Implements

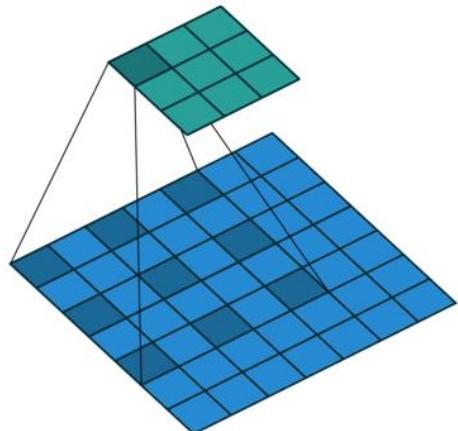
### C1. Convolution-In Reality



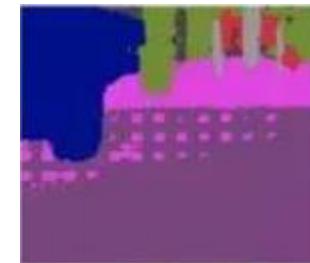
# II. Fundamental Layers

## C. Layers Concepts & Implements

### C1. Convolution - Dilated Conv



*Get better view;*



Dilated Conv

*Mostly used in segmentation*

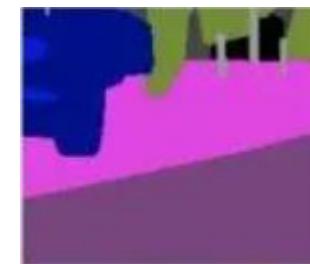


Hybrid  
Dilated Conv

*Some drawbacks;*

*Some improvements*

*Will revisit in Segmentation part*



Ground Truth

# II. Fundamental Layers

## C. Layers Concepts & Implements

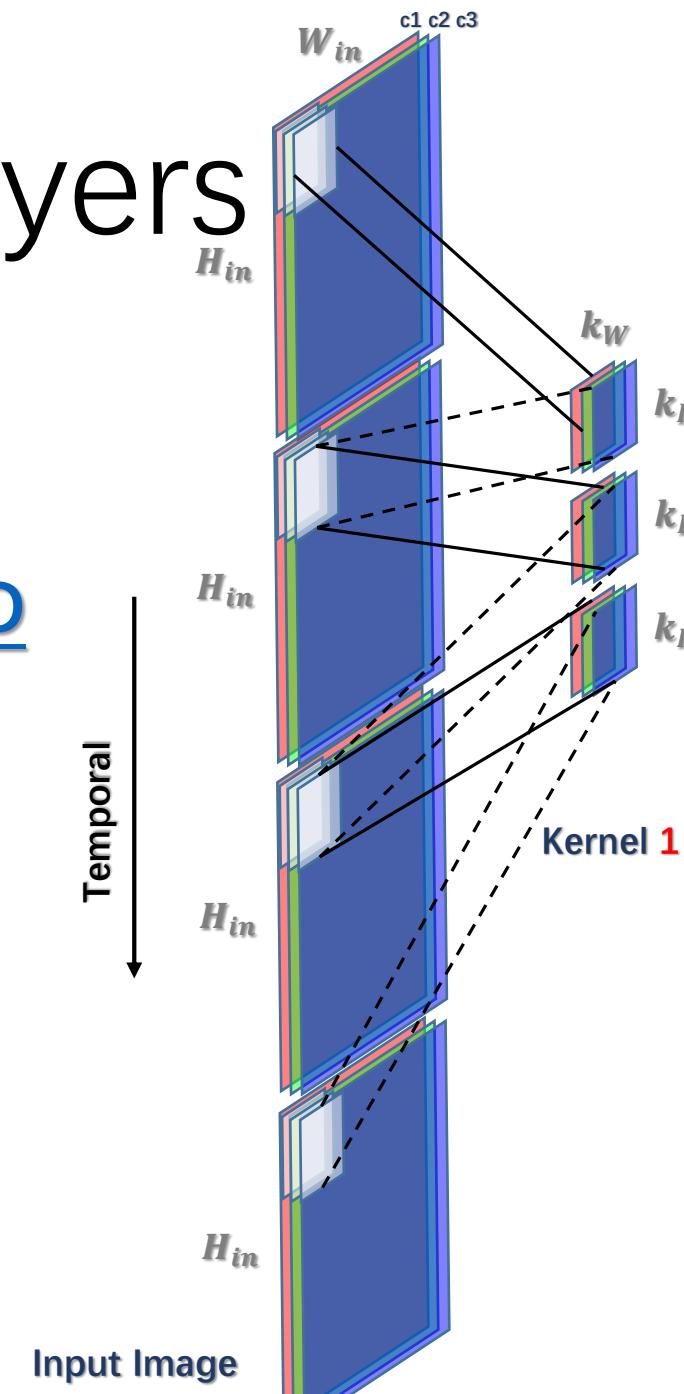
### C1. Convolution-3D Conv / C3D

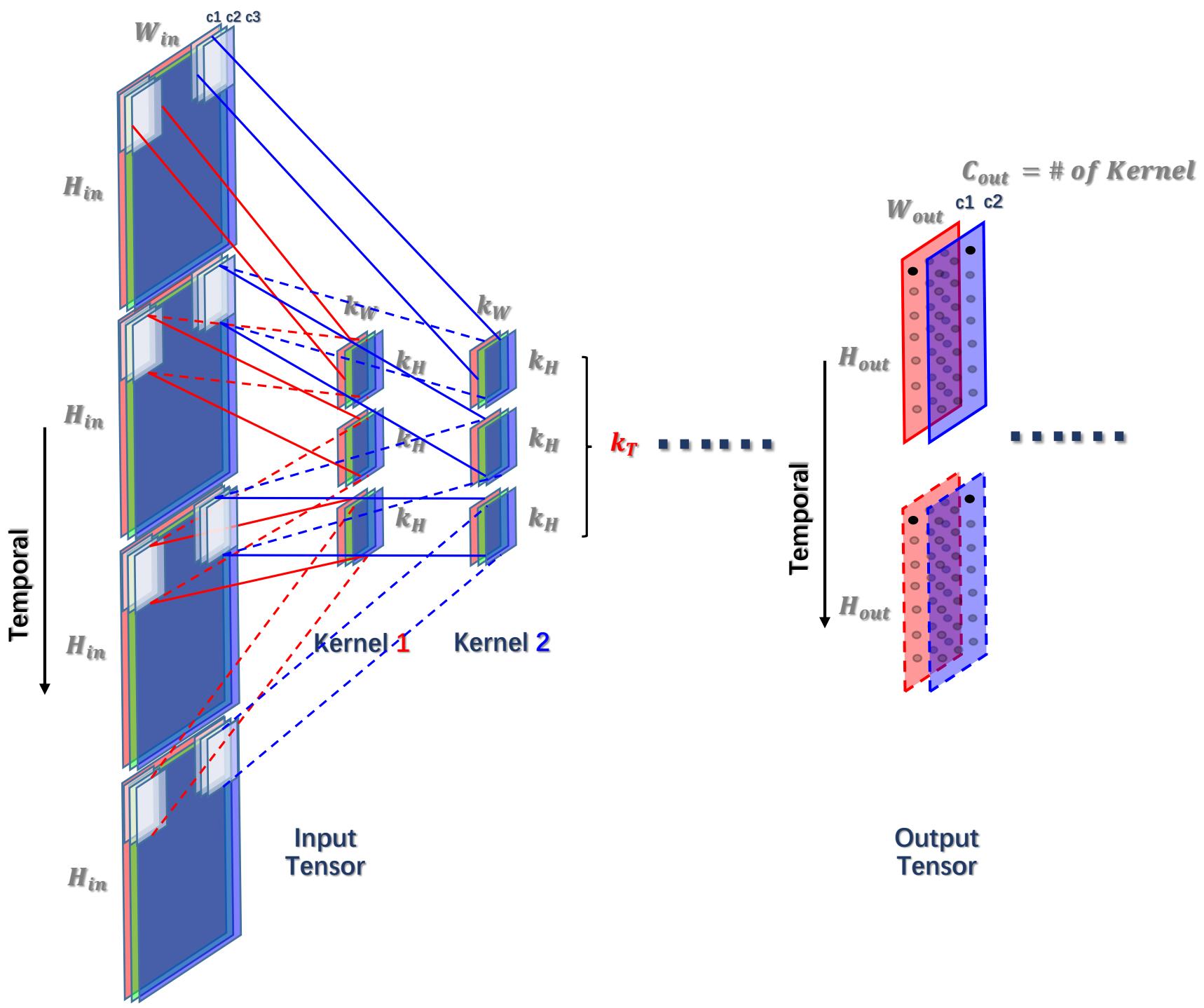
*Trend:*

*CNN can do everything.*

*Necessity:*

*Time info is critical*





## *Question:*

*Write code of 2D image median filtering*

## *Attentions:*

1. *Padding*
2. *Stride*
3. *Kernel Size*
4. *How to get the median number*

# II. Fundamental Layers

## C. Layers Concepts & Implements

### C1. Convolution – Other Convs

*Depthwise Convolution*

*Group Convolution*

Will Revisit Later

## II. Fundamental Layers

### C. Layers Concepts & Implements

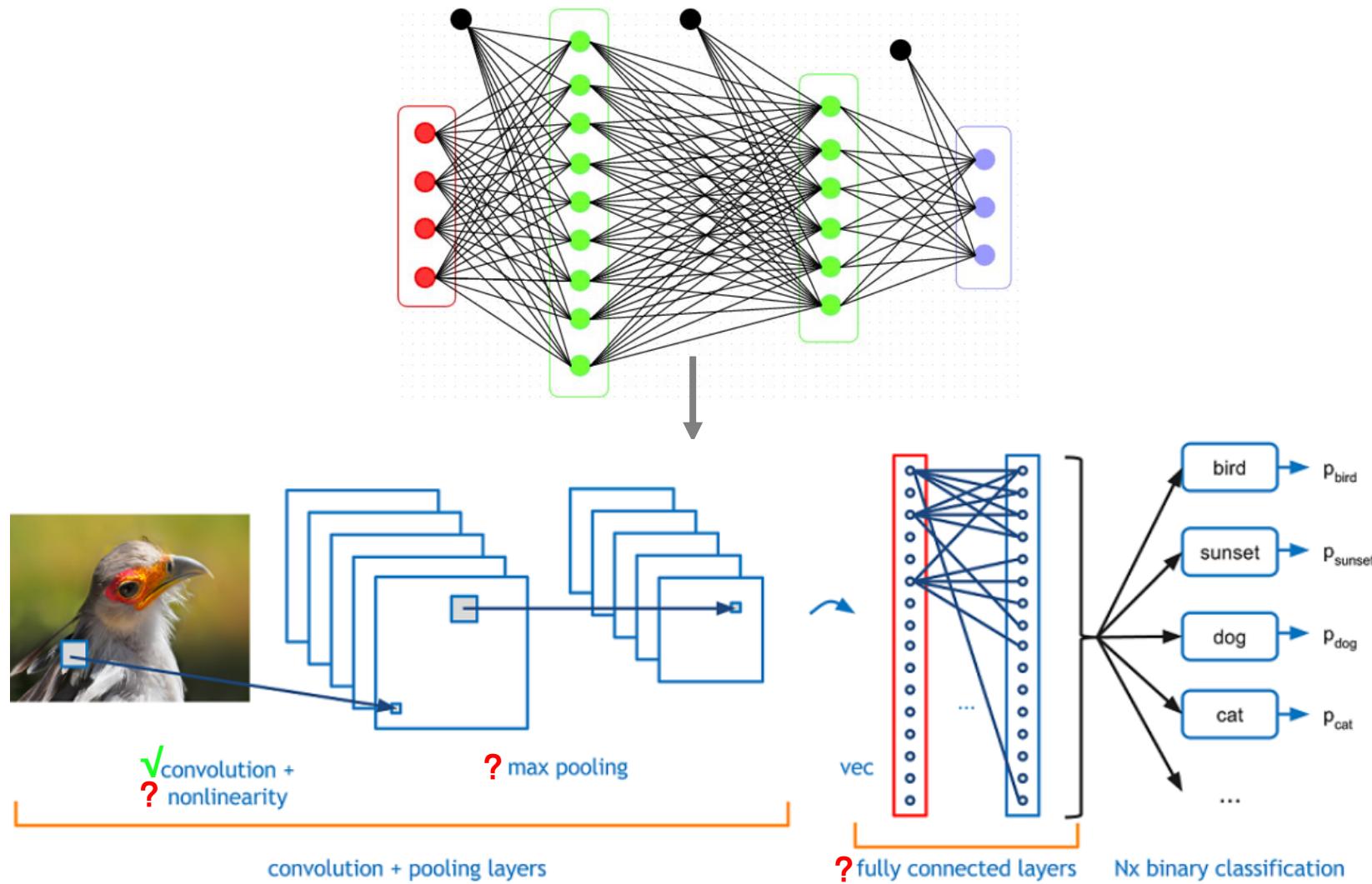
#### C1. Convolution – Advanced Convs

##### ***Deformable Convolution***

(2017, MSRA)

Read If U Have Interests

# I. CNN Outlines - Revisit



# II. Fundamental Layers

## C. Layers Concepts & Implements

### C2. ReLU - Nonlinearity

#### *Questions:*

1. *What is nonlinearity*
2. *What is nonlinearity for*
3. *What is ReLU*
4. *Why ReLU*

#### *Answers:*

- 1.
- 2.
- 3.
- 4.

# II. Fundamental Layers

## C. Layers Concepts & Implements

### C2. ReLU - Nonlinearity

#### *Questions:*

1. *What is nonlinearity*
2. *What is nonlinearity for*
3. *What is ReLU*
4. *Why ReLU*

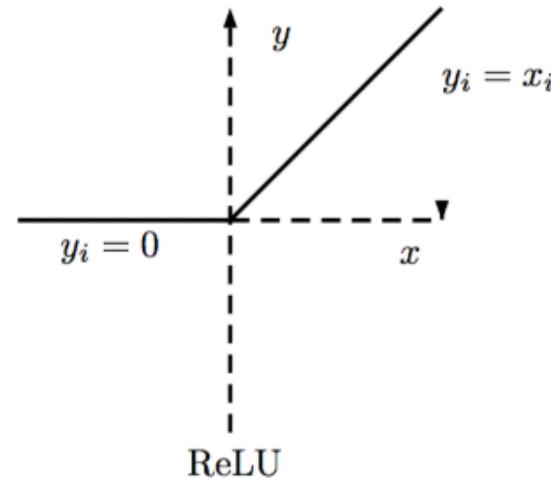
#### *Answers:*

1.  $d_{\text{out}} \neq k d_{\text{in}}$
2. *to fit the complex system*
- 3.
- 4.

# II. Fundamental Layers

## C. Layers Concepts & Implements

### C2. ReLU - Nonlinearity

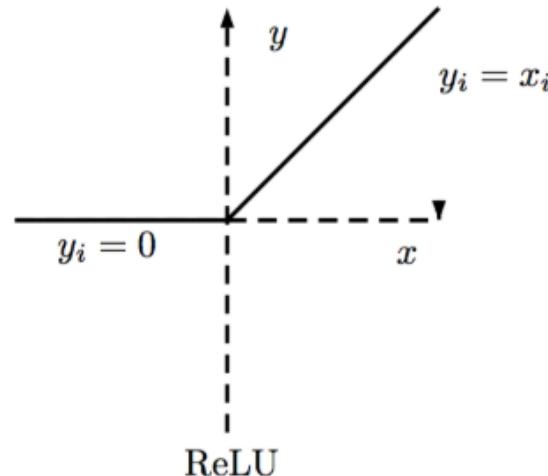


$$ReLU(x) = \max(0, x)$$

# II. Fundamental Layers

## C. Layers Concepts & Implements

### C2. ReLU - Nonlinearity



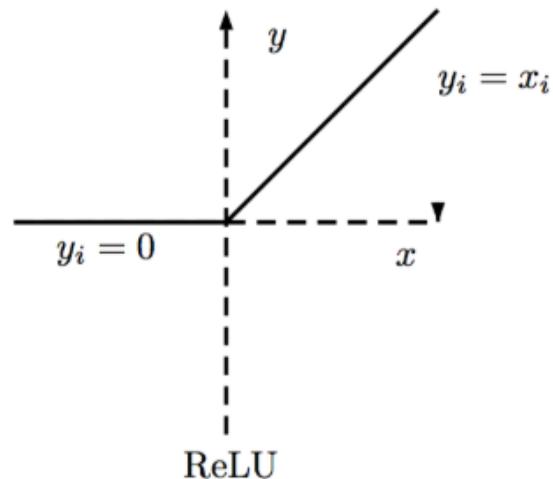
$$ReLU(x) = \max(0, x)$$

$$\frac{\partial L}{\partial x} = \begin{cases} 0, & x \leq 0 \\ 1, & x > 0 \end{cases}$$

# II. Main Layers

## C. Layers Concepts & Implements

### C2. ReLU - Nonlinearity



$$ReLU(x) = \max(0, x)$$

$$\frac{\partial L}{\partial x} = \begin{cases} 0, & x \leq 0 \\ 1, & x > 0 \end{cases}$$

*Good Enough?*

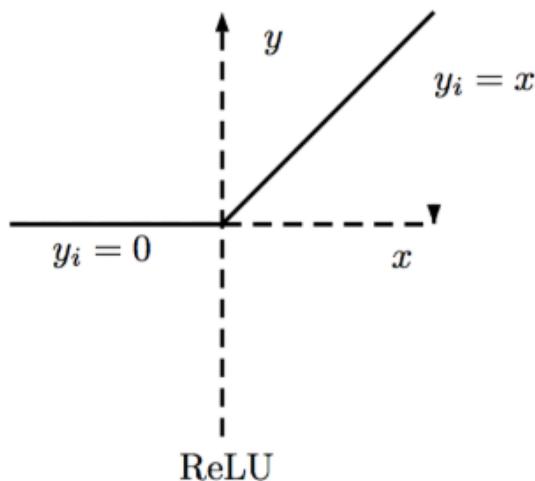
*Partial elements die. Loss info*

*What Else Could Do?*

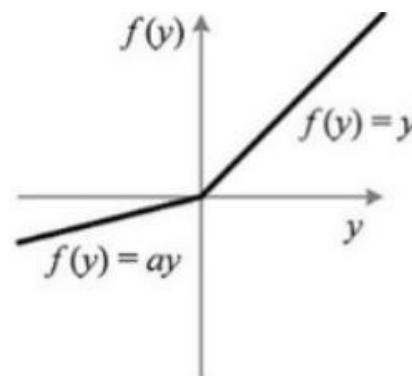
# II. Fundamental Layers

## C. Layers Concepts & Implements

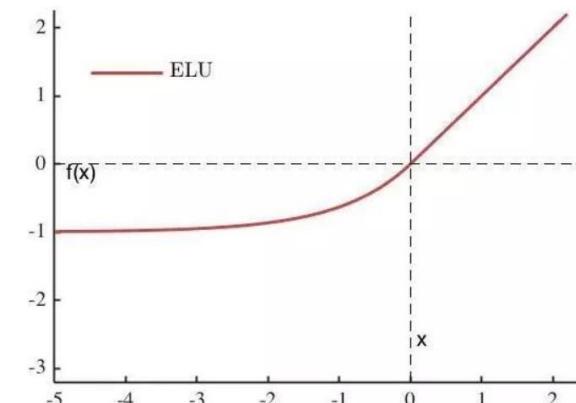
### C2. ReLU - Nonlinearity



$ReLU(x) = \max(0, x)$   
(rectified linear unit)



$Leaky\_ReLU(x) = \max(0.01x, x)$   
 $PReLU(x) = \max(\alpha x, x)$   
(parametric relu)



$$(exponential \text{ relu}) \quad ELU(x) = \begin{cases} \alpha(\exp(x) - 1), & x \leq 0 \\ x, & x > 0 \end{cases}$$

$$\frac{\partial L}{\partial x} = \begin{cases} ELU(x) + \alpha, & x \leq 0 \\ 1, & x > 0 \end{cases}$$

# II. Fundamental Layers

## C. Layers Concepts & Implements

### C2. ReLU - Nonlinearity

#### *Questions:*

1. *What is nonlinearity*
2. *What is nonlinearity for*
3. *What is ReLU*
4. *Why ReLU*

#### *Answers:*

1.  $d_{\text{out}} \neq k d_{\text{in}}$
2. *to fit the complex system*
3. *activation func like Sigmoid, Tanh ...*
4. *easy to implement, calculate, loss-friendly*

# II. Fundamental Layers

## C. Layers Concepts & Implements

### C2. ReLU - Nonlinearity

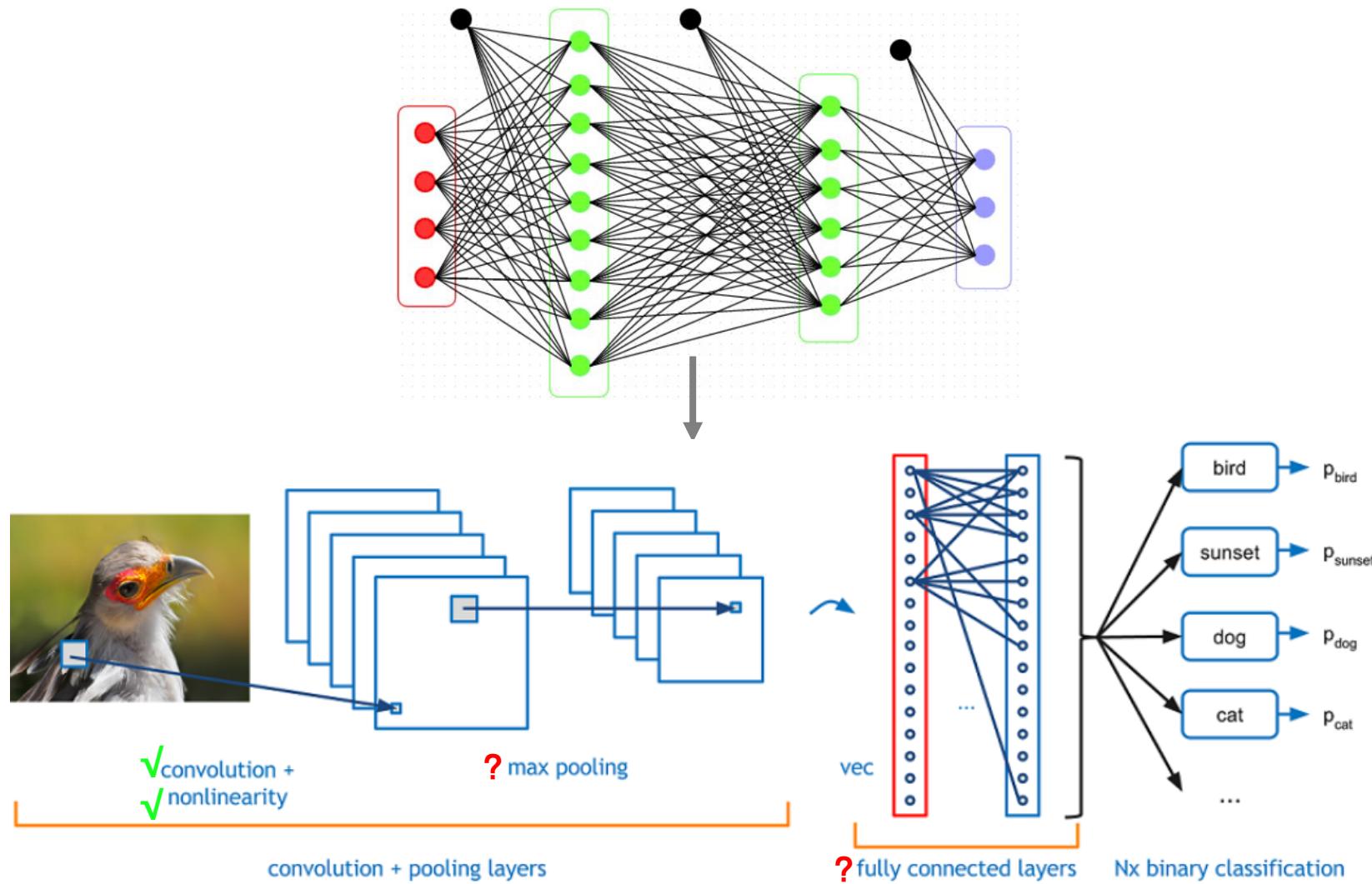
#### *Questions:*

5. *Which one is better*
6. *Any other types*

#### *Answers*

5. *ReLU could be 1<sup>st</sup> choice  
PReLU looks better  
Never use sigmoid in middle layers*
6. *Maxout, SeLU. (If U have interests)*

# I. CNN Outlines - Revisit



# II. Fundamental Layers

## C. Layers Concepts & Implements

### C3. Pooling

#### *Questions:*

1. *What is pooling*
2. *What is pooling for*
3. *What kind of pooling*

#### *Answers:*

- 1.
- 2.
- 3.

# II. Fundamental Layers

## C. Layers Concepts & Implements

### C3. Pooling (池化)

2	3	3	4
6	7	8	9
4	3	2	0
2	3	4	5

Pooling: Reduce tensor's size



7	9
4	5

# II. Fundamental Layers

## C. Layers Concepts & Implements

### C3. Pooling (池化)

2	3	3	4
6	7	8	9
4	3	2	0
2	3	4	5

**2x2, /2 Max Pooling:**

Reduce tensor's size

---

**Aim:**

Get the most robust responded features of exact spatial location



7	9
4	5

# II. Fundamental Layers

## C. Layers Concepts & Implements

### C3. Pooling (池化)

2	3	3	4
6	7	8	9
4	3	2	0
2	3	4	5

**2x2, /2 Ave Pooling:**

Reduce tensor's size

---

4.5	6
3	2.75

**Aim:**

Get the averaged features of exact spatial location;

Usually used to change tensor's shape at the end of the network

# II. Fundamental Layers

## C. Layers Concepts & Implements

### C3. Pooling - BP

2	3	3	4
6	7	8	9
4	3	2	0
2	3	4	5

2x2, /2 Max Pooling:  
Reduce tensor's size

---

BP:

$$Loss_i * \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

7	9
4	5

# II. Fundamental Layers

## C. Layers Concepts & Implements

### C3. Pooling - BP

2	3	3	4
6	7	8	9
4	3	2	0
2	3	4	5

2x2, /2 Ave Pooling:

Reduce tensor's size

BP:

$$Loss_i * \begin{bmatrix} \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} \\ \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} \\ \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} \\ \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} \end{bmatrix}$$

7	9
4	5

# II. Fundamental Layers

## C. Layers Concepts & Implements

### C3. Pooling

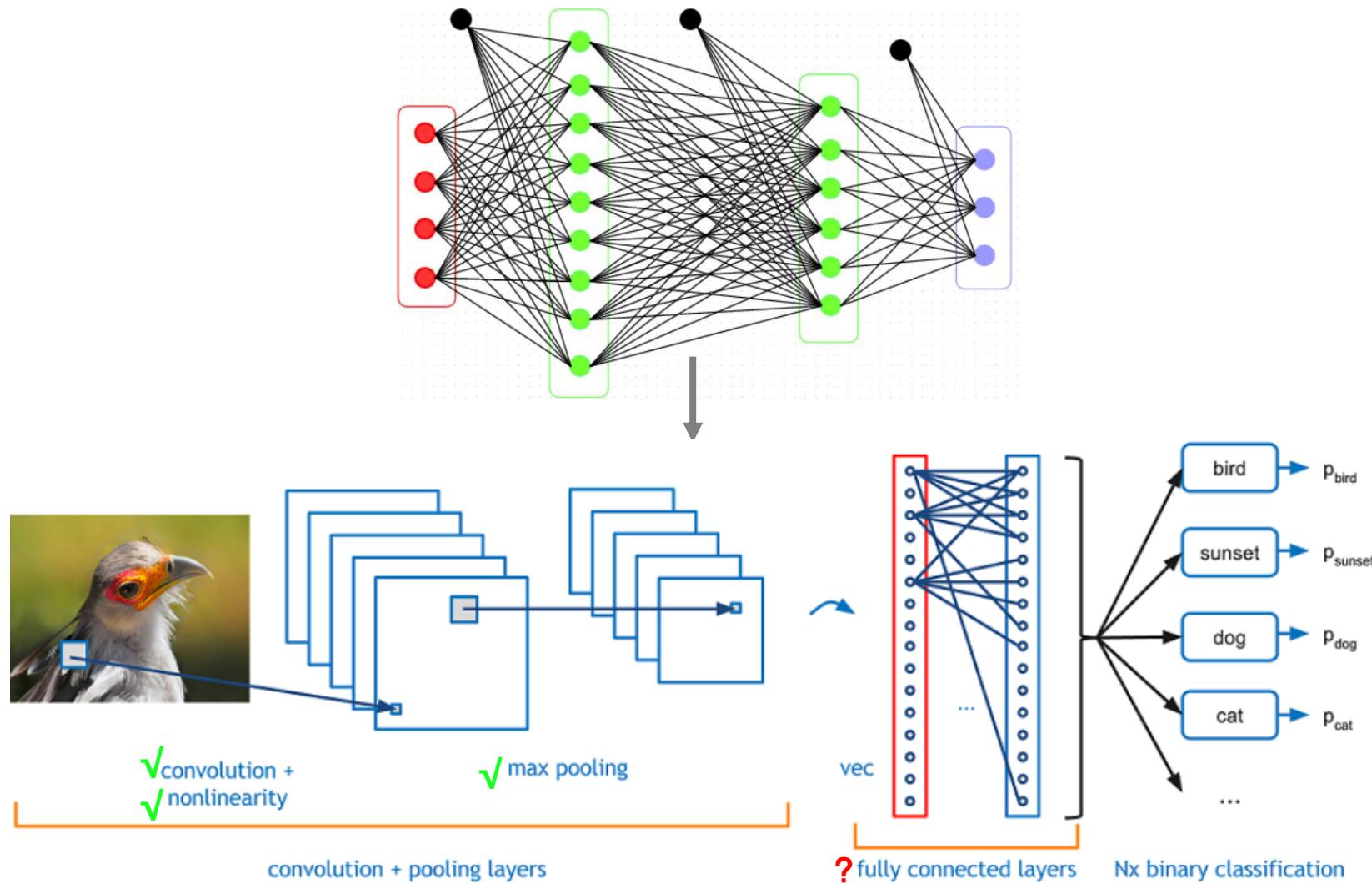
#### *Questions:*

1. *What is pooling*
2. *What is pooling for*
3. *What kind of pooling*
4. *How to do BP*

#### *Answers:*

- 1.
- 2.
- 3.
- 4.

# I. CNN Outlines - Revisit



## II. Fundamental Layers

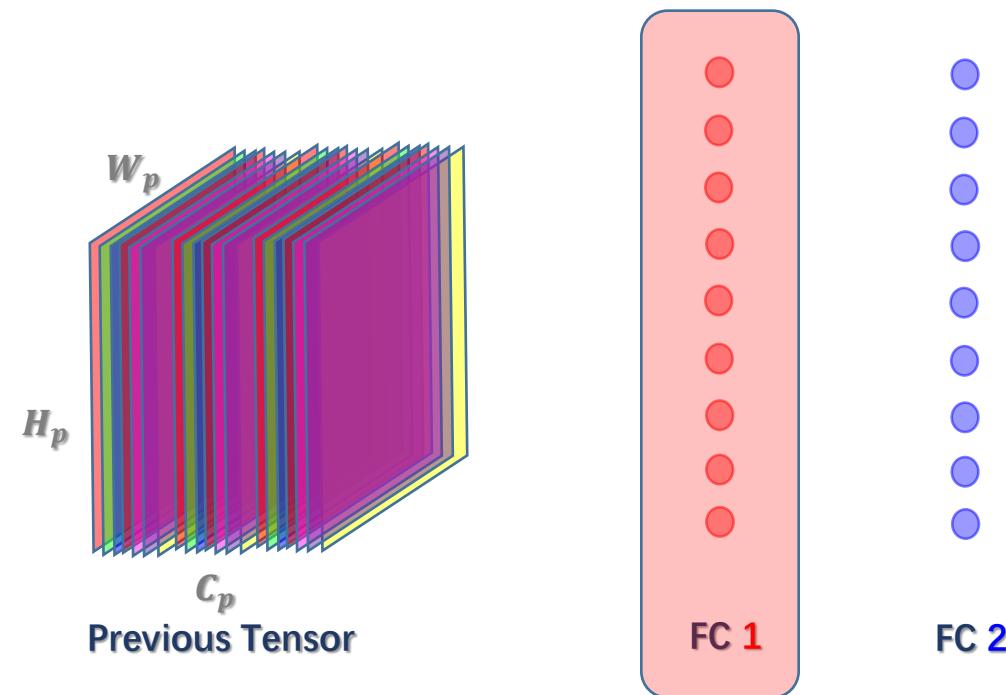
### C. Layers Concepts & Implements

#### C4. Fully Connected / Inner Product

# II. Fundamental Layers

## C. Layers Concepts & Implements

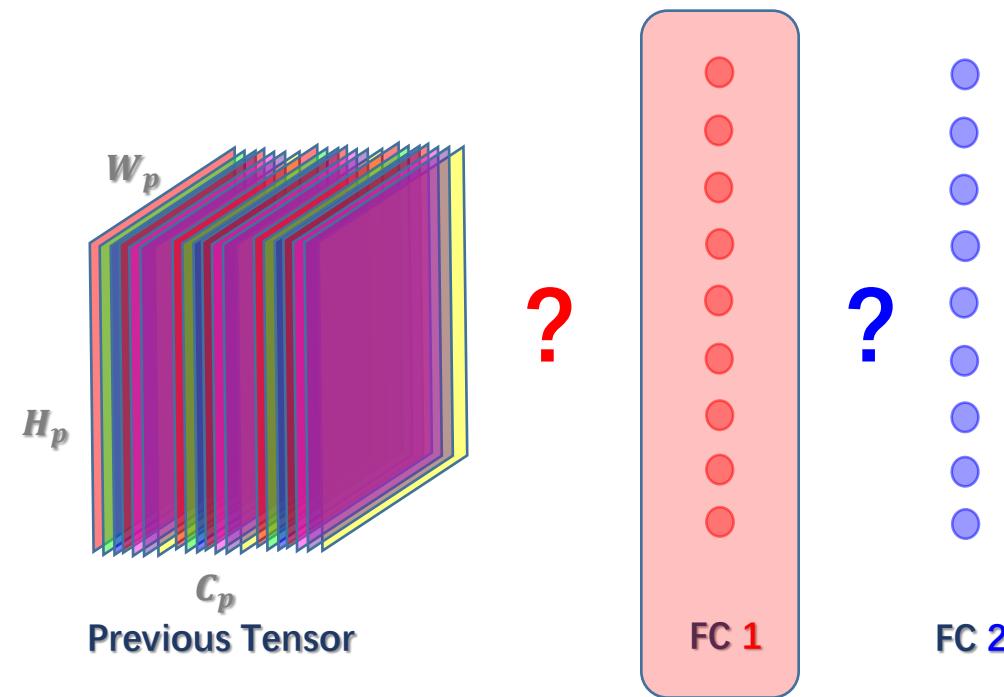
### C4. Fully Connected / Inner Product



# II. Fundamental Layers

## C. Layers Concepts & Implements

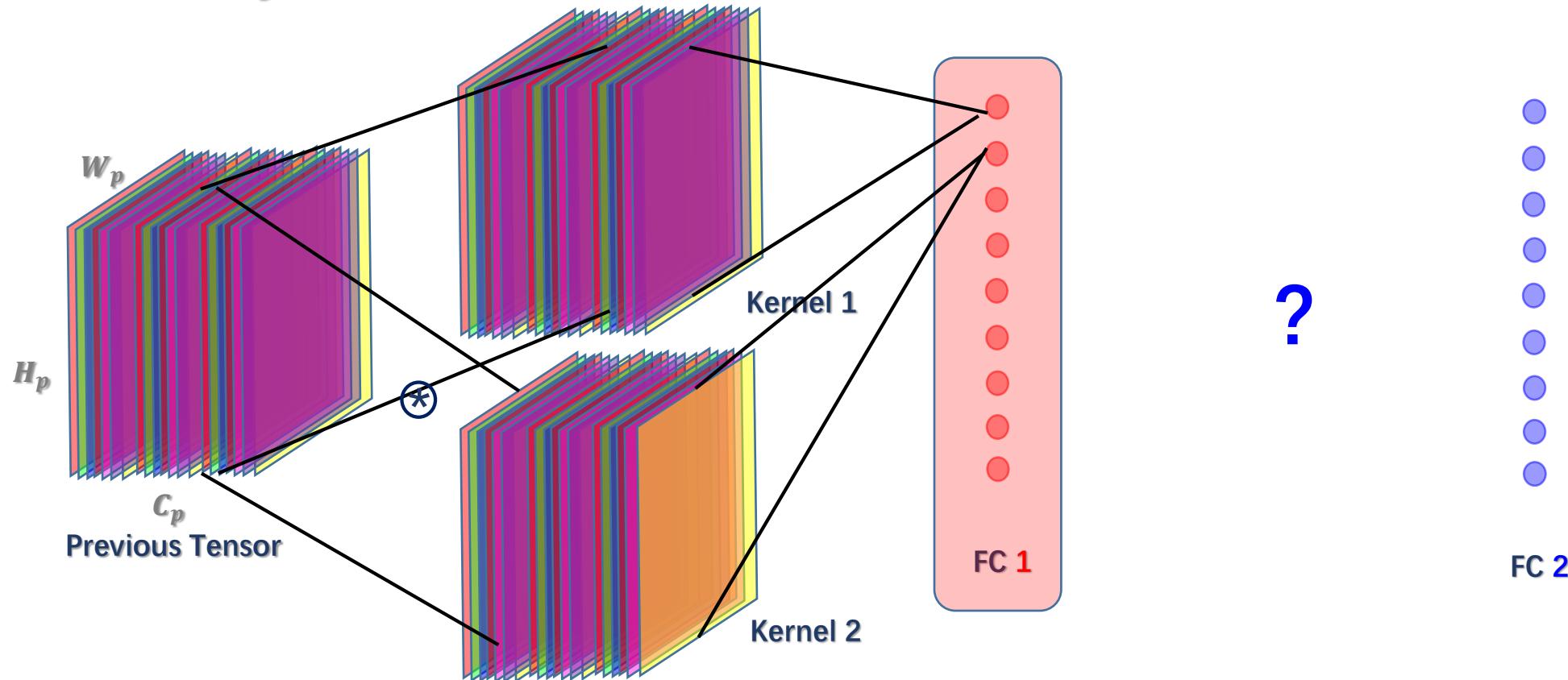
### C4. Fully Connected / Inner Product



# II. Fundamental Layers

## C. Layers Concepts & Implements

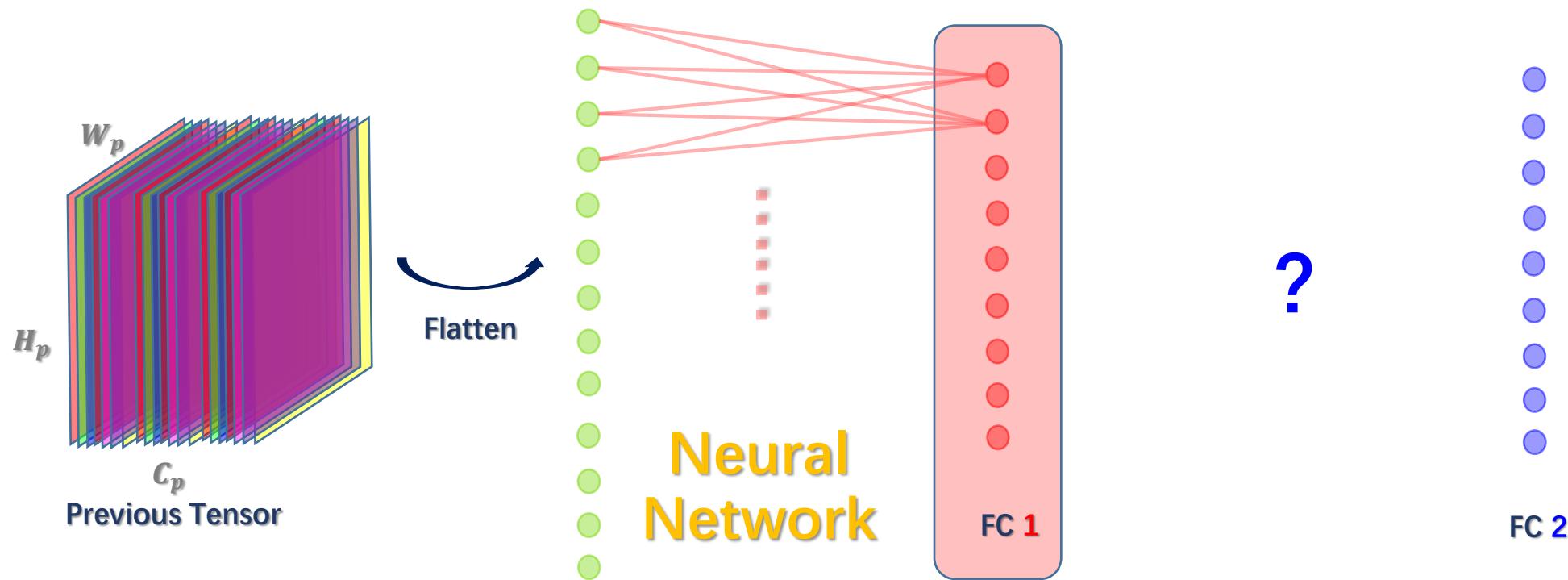
### C4. Fully Connected / Inner Product



# II. Fundamental Layers

## C. Layers Concepts & Implements

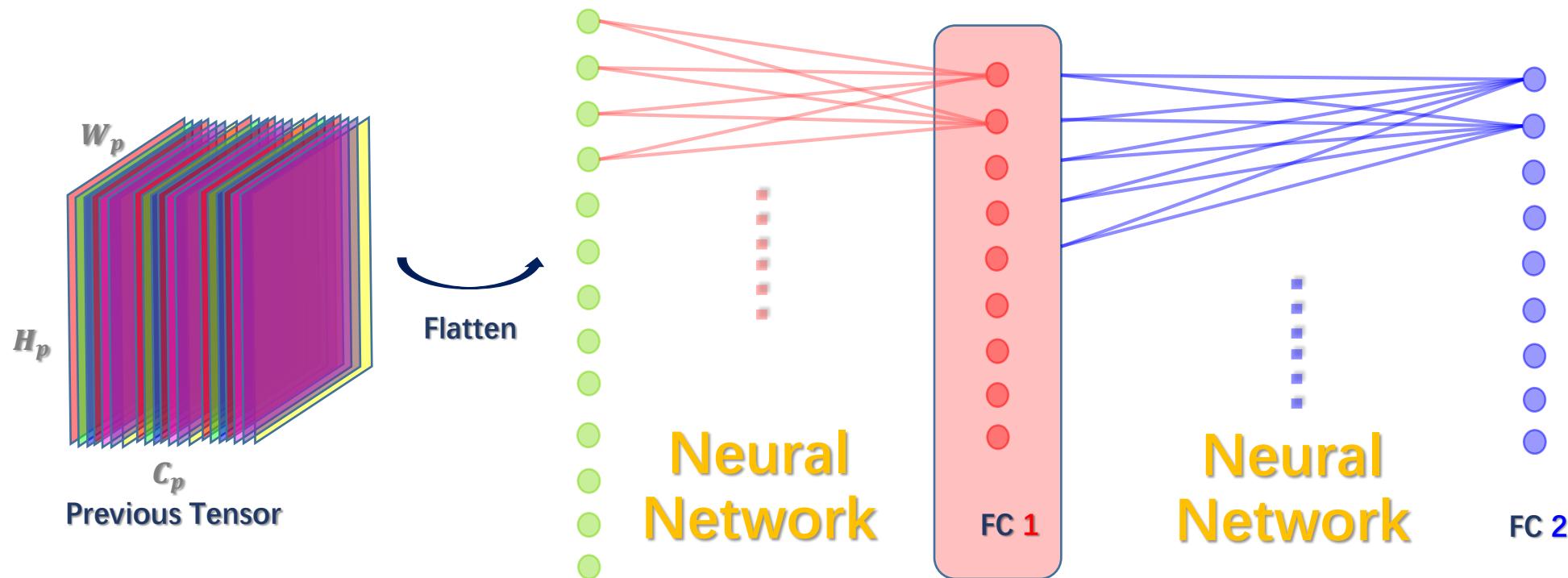
### C4. Fully Connected / Inner Product



# II. Fundamental Layers

## C. Layers Concepts & Implements

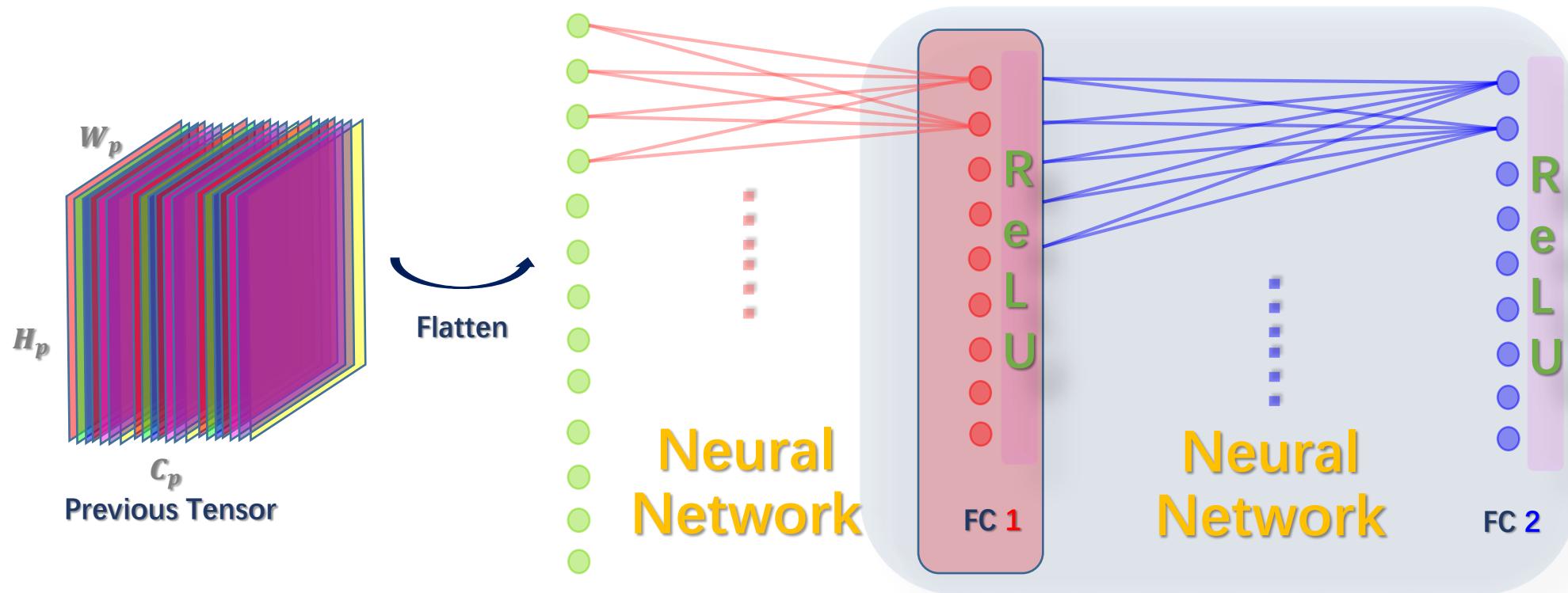
### C4. Fully Connected / Inner Product



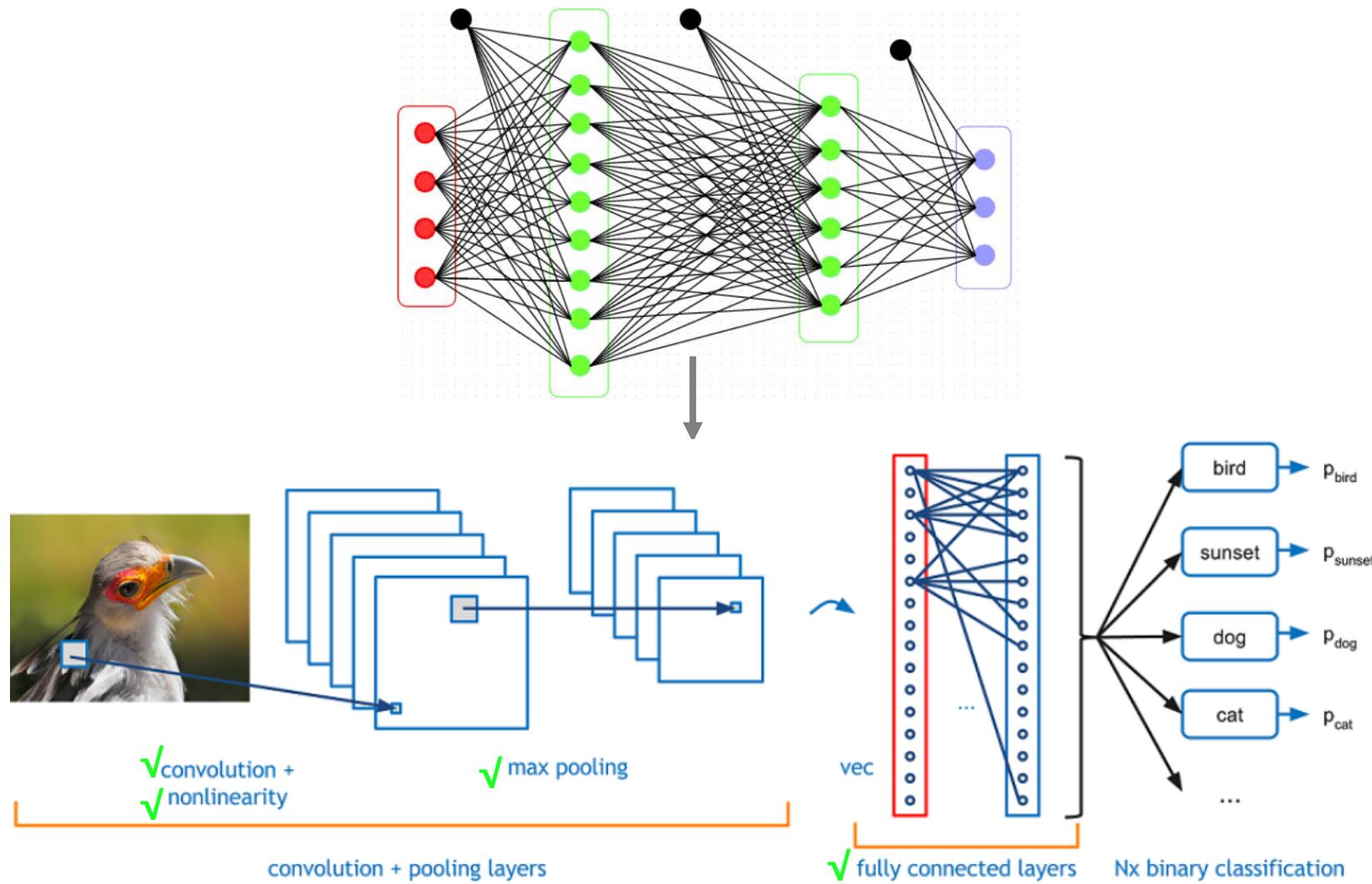
# II. Fundamental Layers

## C. Layers Concepts & Implements

### C4. Fully Connected / Inner Product



# I. CNN Outlines - Revisit



## II. Fundamental Layers

### C. Layers Concepts & Implements

#### C4. Fully Connected / Inner Product

**Really?**

## II. Fundamental Layers

### C. Layers Concepts & Implements

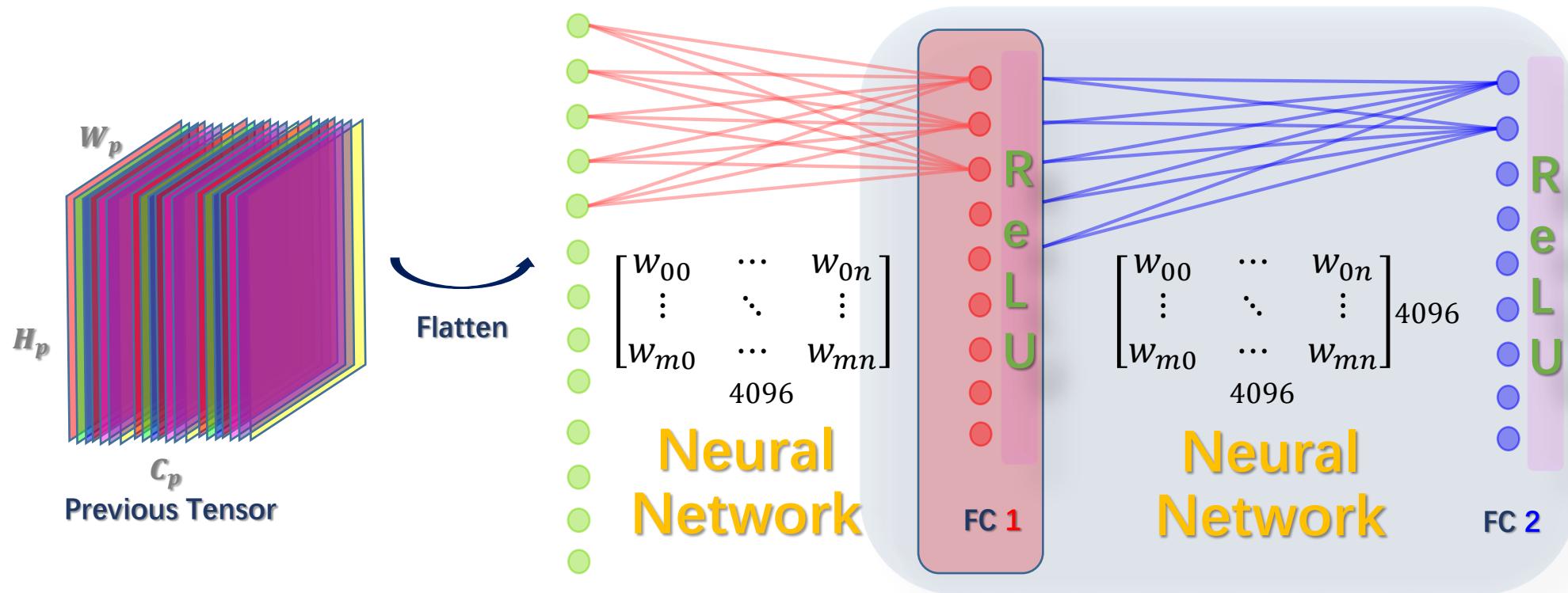
#### C4. Fully Connected / Inner Product

Any  
Thoughts?

# II. Fundamental Layers

## C. Layers Concepts & Implements

### C4. Fully Connected / Inner Product



## II. Fundamental Layers

### C. Layers Concepts & Implements

#### C4. Fully Connected / Inner Product

1. Too many parameters, sometimes
2. Too many calculations, sometimes
3. Too much resources needed, sometimes
4. Prone to over fit

## II. Fundamental Layers

### C. Layers Concepts & Implements

#### C4. Fully Connected / Inner Product

1. Too many parameters, sometimes
2. Too many calculations, sometimes
3. Too much resources needed, sometimes
4. Prone to over fit

Any Features?

## II. Fundamental Layers

### C. Layers Concepts & Implements

#### C4. Fully Connected / Inner Product

1. Too many parameters, sometimes
2. Too many calculations, sometimes
3. Too much resources needed, sometimes
4. Prone to over fit
  1. CNN-based feature vector
  2. Global ave pooling as a trend

## II. Fundamental Layers

### C. Layers Concepts & Implements

#### C4. Fully Connected / Inner Product

##### *Questions:*

1. *What if we need parameters to fit?*
2. *But we don't want overfitting*
3. *And time / resource consuming*

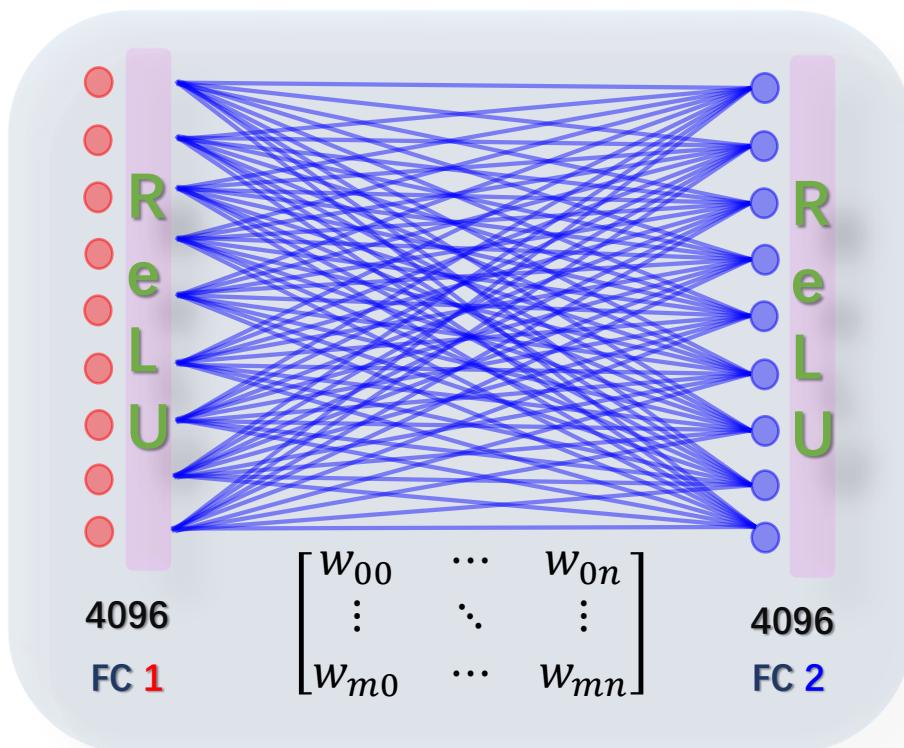
# III. Functional Layers



# II. Functional Layers

## D. Functional Layers Concepts & Implements

### D1. Dropout Layer

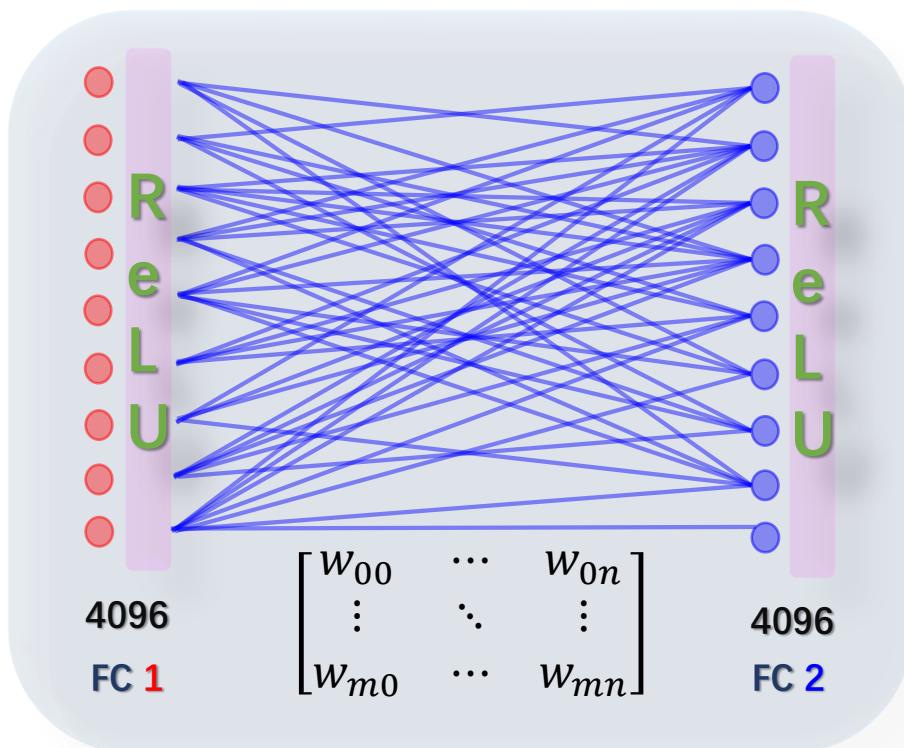


1. Too many parameters
2. Too many calculations
3. Too much resources needed
4. Prone to over fit

# II. Functional Layers

## D. Functional Layers Concepts & Implements

### D1. Dropout Layer

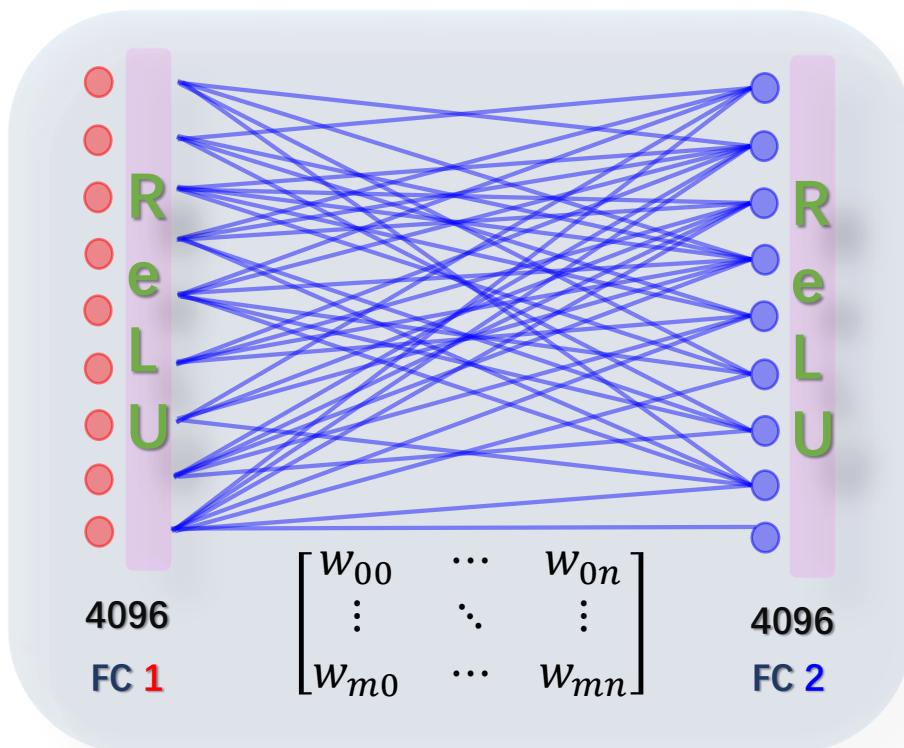


1. Too many parameters
2. Too many calculations
3. Too much resources needed
4. Prone to over fit

# II. Functional Layers

## D. Functional Layers Concepts & Implements

### D1. Dropout Layer

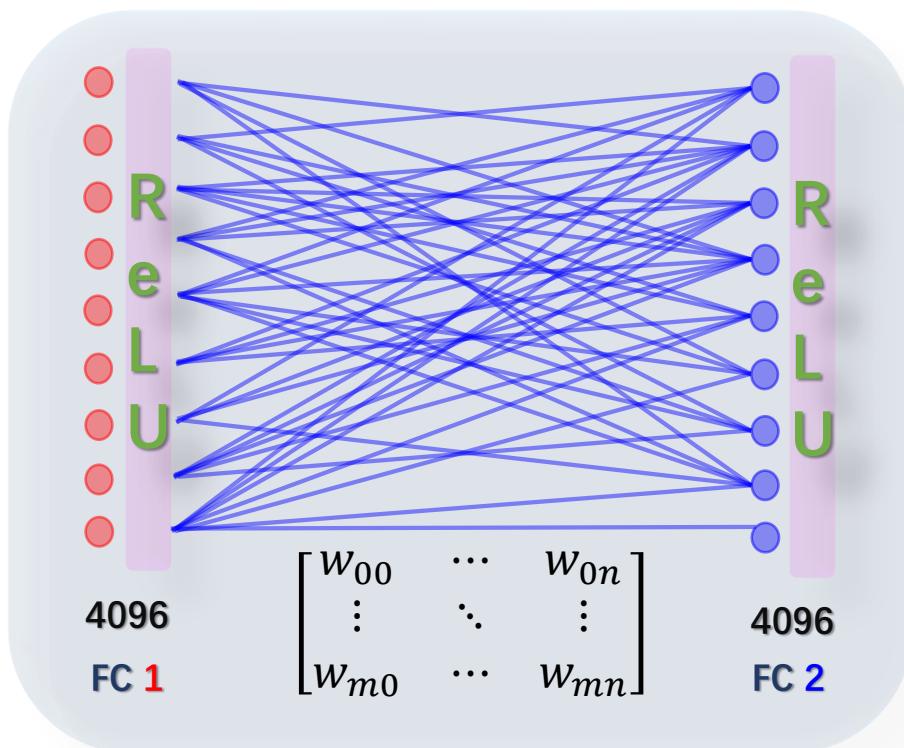


Randomly give up  
some connections  
under a particular  
ratio

# II. Functional Layers

## D. Functional Layers Concepts & Implements

### D1. Dropout Layer



Train:

```
def dropout_train_step(x):
    ReLU1 = np.maximum(0, FC1)
    mask1 = np.random.rand(*ReLU1.shape) < p
    Dropout1 = ReLU1 * mask1
```

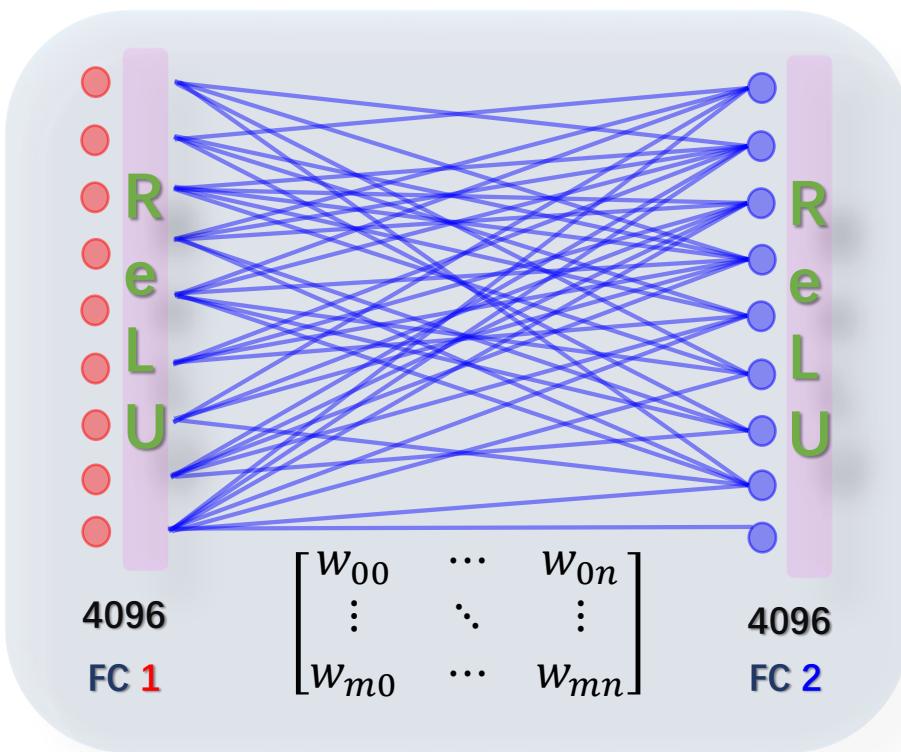
Test:

How ?

# II. Functional Layers

## D. Functional Layers Concepts & Implements

### D1. Dropout Layer



Train:

```
def dropout_train_step(x):
    ReLU1 = np.maximum(0, FC1)
    mask1 = np.random.rand(*ReLU1.shape) < p
    Dropout1 = ReLU1 * mask1
```

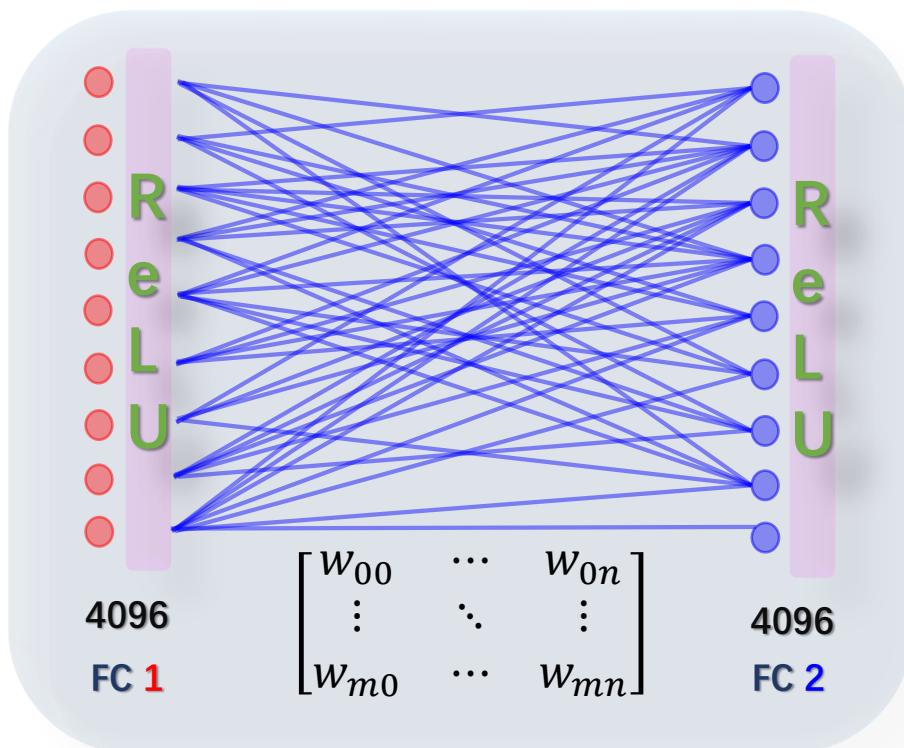
Test:

$$\begin{aligned} y &= w_0x_0 + w_1x_1 \\ &= w_0 * 1 + w_1 * 1 \\ &= [(w_0 * 0 + w_1 * 0) + (w_0 * 0 + w_1 * 1) + \\ &\quad (w_0 * 1 + w_1 * 0) + (w_1 * 1 + w_1 * 1)] * \frac{1}{4} \\ &= \frac{1}{4} * (2w_0x_0 + 2w_1x_1) \\ &= \frac{1}{2} * (w_0x_0 + w_1x_1) \\ &= E(y) \end{aligned}$$

# II. Functional Layers

## D. Functional Layers Concepts & Implements

### D1. Dropout Layer



Train:

```
def dropout_train_step(x):
    ReLU1 = np.maximum(0, FC1)
    mask1 = np.random.rand(*ReLU1.shape) < p
    Dropout1 = ReLU1 * mask1
```

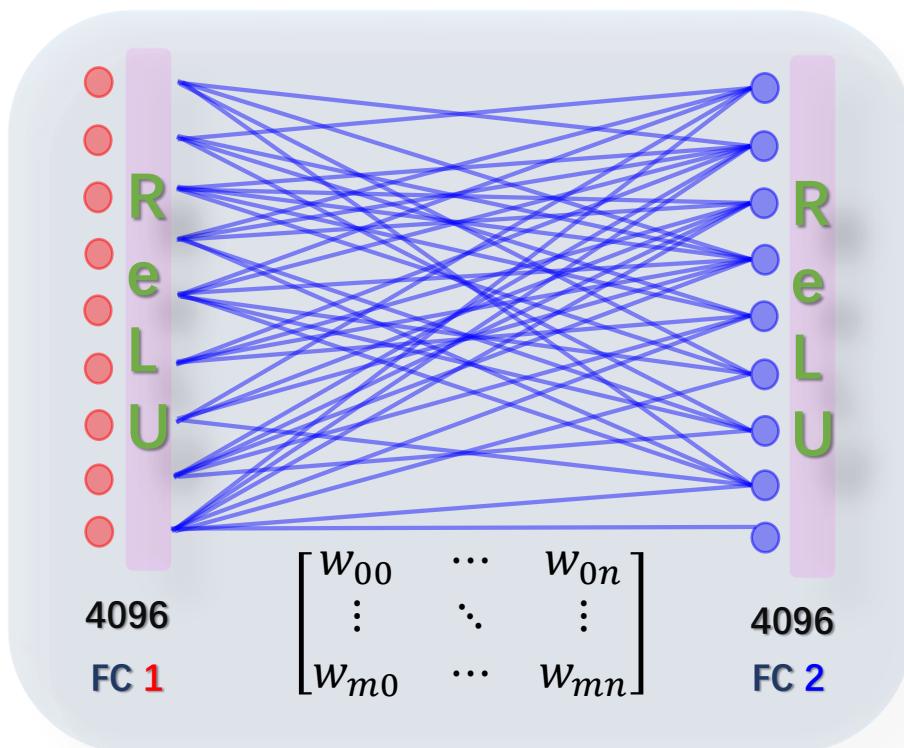
Test:

```
def dropout_test_step(x):
    ReLU1 = np.maximum(0, FC1) * p
```

# II. Functional Layers

## D. Functional Layers Concepts & Implements

### D1. Dropout Layer



Train:

```
def dropout_train_step(x):
    ReLU1 = np.maximum(0, FC1)
    mask1 = np.random.rand(*ReLU1.shape) < p
    Dropout1 = ReLU1 * mask1
```

Test:

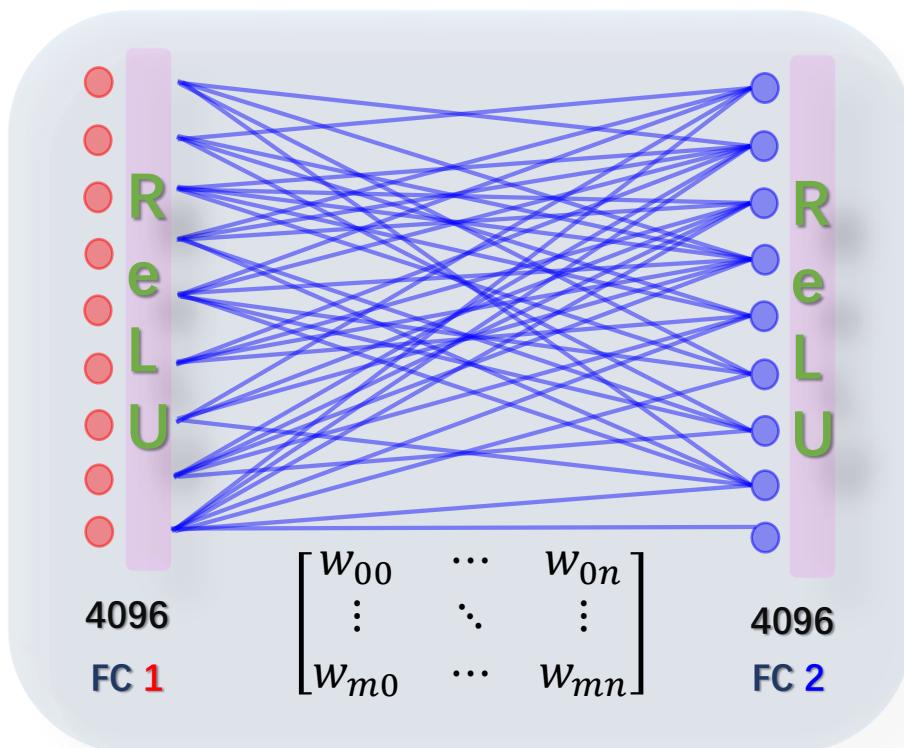
```
def dropout_test_step(x):
    ReLU1 = np.maximum(0, FC1) * p
```

We could do better

# II. Functional Layers

## D. Functional Layers Concepts & Implements

### D1. Dropout Layer



Train:

```
def dropout_train_step(x):
    ReLU1 = np.maximum(0, FC1)
    mask1 = (np.random.rand(*ReLU1.shape) < p) / p
    Dropout1 = ReLU1 * mask1
```

Test:

```
def dropout_test_step(x):
    ReLU1 = np.maximum(0, FC1)
```

# II. Functional Layers

## D. Functional Layers Concepts & Implements

### D2. Batch Normalization

# II. Functional Layers

## D. Functional Layers Concepts & Implements

### D2. Batch Normalization

#### *Questions:*

1. *What is BN*
2. *What is BN for*
3. *What kind N*
4. *Where to put BN*
5. *What does BN look like*
6. *How to compute*

#### *Answers:*

- 1.
- 2.
- 3.
- 4.
- 5.
- 6.

# II. Functional Layers

## D. Functional Layers Concepts & Implements

### D2. Batch Normalization

*The deeper network, the harder to train*

*Vanishing gradient*

# II. Functional Layers

## D. Functional Layers Concepts & Implements

### D2. Batch Normalization

*The deeper network, the more difficult to train*

*Vanishing gradient*

Q. What kind of data is good to train

# II. Functional Layers

## D. Functional Layers Concepts & Implements

### D2. Batch Normalization

*The deeper network, the more difficult to train*

*Vanishing gradient*

Q. What kind of data is good to train (why sigmoid is much worse?)

# II. Functional Layers

## D. Functional Layers Concepts & Implements

### D2. Batch Normalization

*The deeper network, the more difficult to train*

*Vanishing gradient*

Q. What kind of data is good to train (why sigmoid is much worse?)

A:

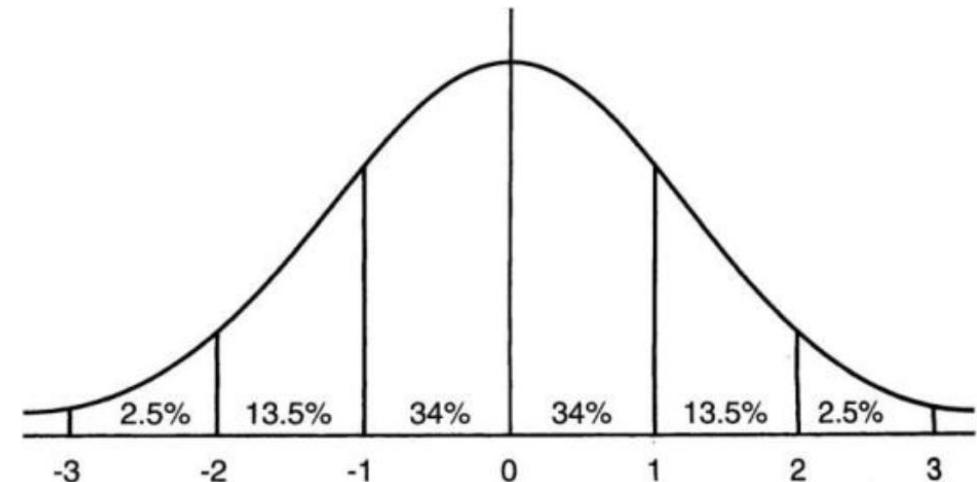
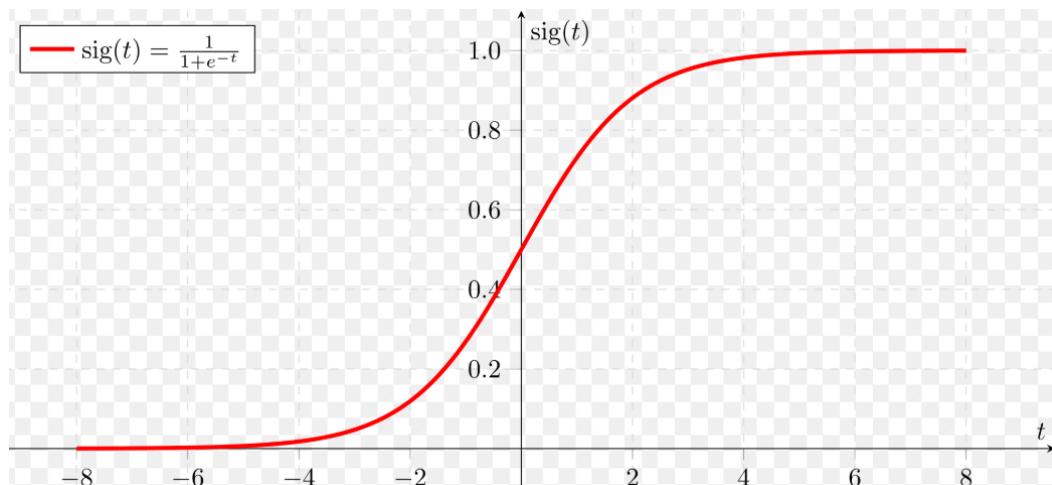
[Independent Identically Distribution & No Internal Covariate Shift]

The distribution is coincident to data and to the range of activation function where there's largest gradient.

# II. Functional Layers

## D. Functional Layers Concepts & Implements

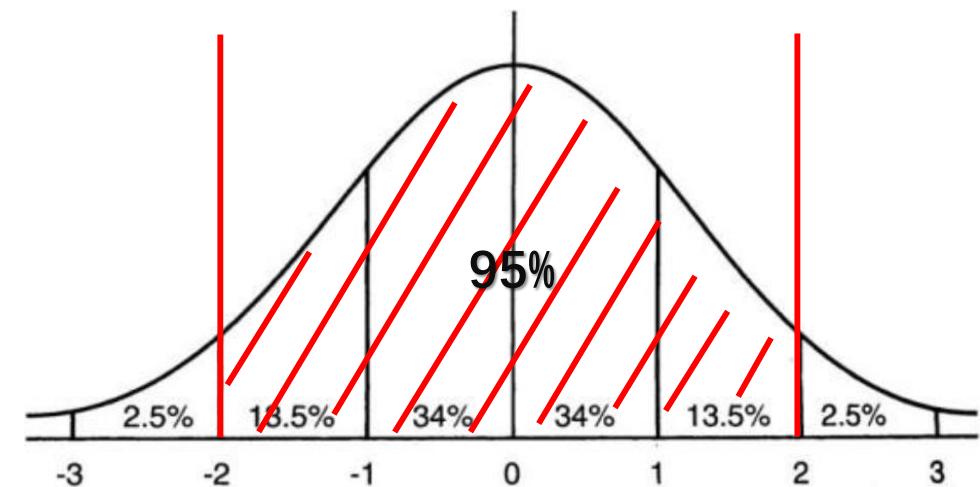
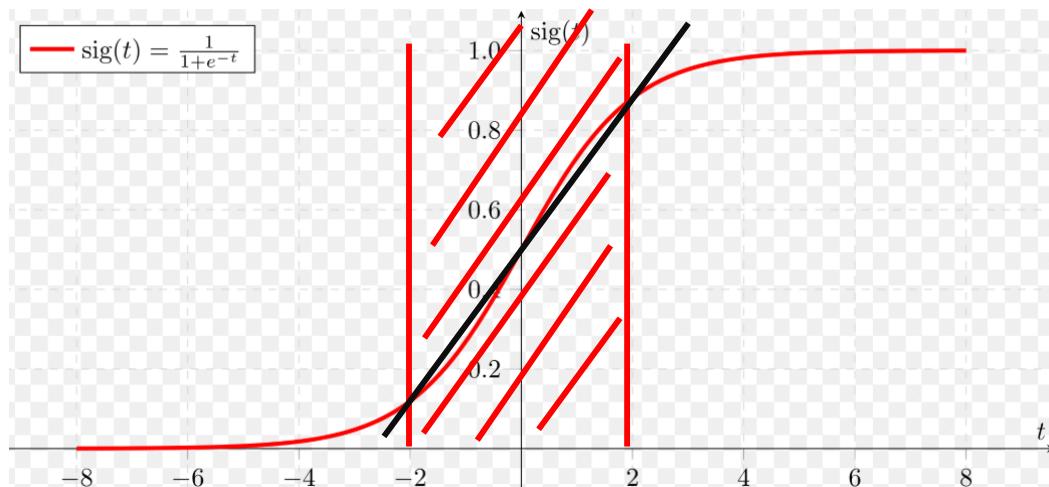
### D2. Batch Normalization-Theory



# II. Functional Layers

## D. Functional Layers Concepts & Implements

### D2. Batch Normalization-Theory



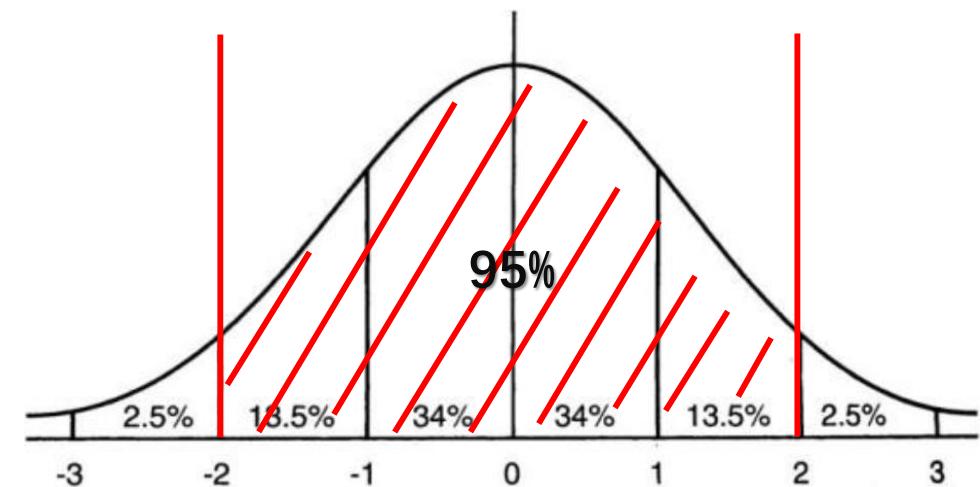
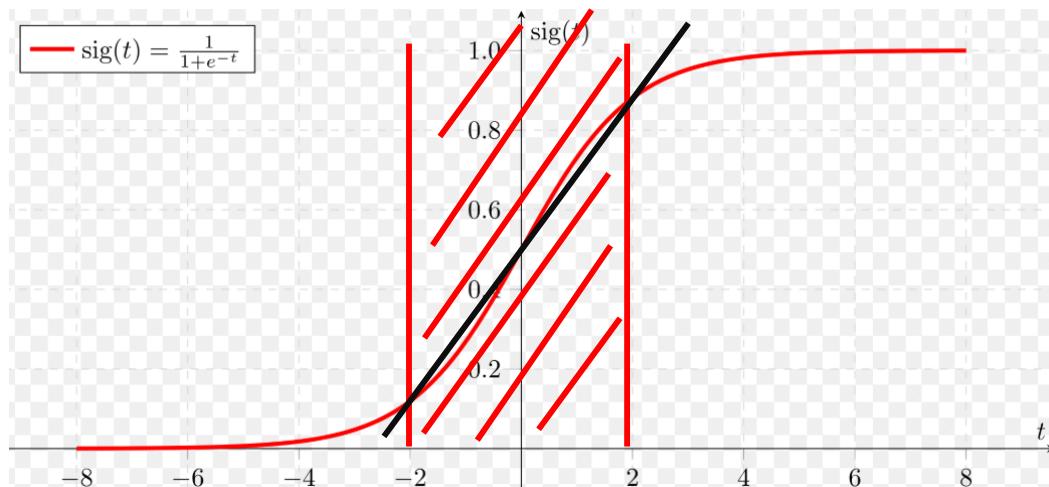
Layer-layer: Should have IID (Independent identical distribution)

Solve: Internal Covariate Shift

# II. Functional Layers

## D. Functional Layers Concepts & Implements

### D2. Batch Normalization-Theory



Gradient saturation: a. gradient losing; b. loss is reducing

BN: pull the data back to distribution having larger distribution.

Changing data distribution via changing data

# II. Functional Layers

## D. Functional Layers Concepts & Implements

### D2. Batch Normalization-Theory

**Input:** Values of  $x$  over a mini-batch:  $\mathcal{B} = \{x_{1\dots m}\}$ ;

Parameters to be learned:  $\gamma, \beta$

**Output:**  $\{y_i = \text{BN}_{\gamma, \beta}(x_i)\}$

$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m} \sum_{i=1}^m x_i \quad // \text{mini-batch mean}$$

$$\sigma_{\mathcal{B}}^2 \leftarrow \frac{1}{m} \sum_{i=1}^m (x_i - \mu_{\mathcal{B}})^2 \quad // \text{mini-batch variance}$$

$$\hat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}} \quad // \text{normalize}$$

$$y_i \leftarrow \gamma \hat{x}_i + \beta \equiv \text{BN}_{\gamma, \beta}(x_i) \quad // \text{scale and shift}$$

# II. Functional Layers

## D. Functional Layers Concepts & Implements

### D2. Batch Normalization-Theory

**Input:** Values of  $x$  over a mini-batch:  $\mathcal{B} = \{x_{1\dots m}\}$ ;

Parameters to be learned:  $\gamma, \beta$

**Output:**  $\{y_i = \text{BN}_{\gamma, \beta}(x_i)\}$

$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m} \sum_{i=1}^m x_i \quad // \text{mini-batch mean}$$

$$\sigma_{\mathcal{B}}^2 \leftarrow \frac{1}{m} \sum_{i=1}^m (x_i - \mu_{\mathcal{B}})^2 \quad // \text{mini-batch variance}$$

$$\hat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}} \quad // \text{normalize}$$

$$y_i \leftarrow \gamma \hat{x}_i + \beta \equiv \text{BN}_{\gamma, \beta}(x_i) \quad // \text{scale and shift}$$

# II. Functional Layers

## D. Functional Layers Concepts & Implements

### D2. Batch Normalization-Theory

**Input:** Values of  $x$  over a mini-batch:  $\mathcal{B} = \{x_{1\dots m}\}$ ;

Parameters to be learned:  $\gamma, \beta$

**Output:**  $\{y_i = \text{BN}_{\gamma, \beta}(x_i)\}$

$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m} \sum_{i=1}^m x_i \quad // \text{mini-batch mean}$$

$$\sigma_{\mathcal{B}}^2 \leftarrow \frac{1}{m} \sum_{i=1}^m (x_i - \mu_{\mathcal{B}})^2 \quad // \text{mini-batch variance}$$

$$\hat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}} \quad // \text{normalize}$$

$$y_i \leftarrow \gamma \hat{x}_i + \beta \equiv \text{BN}_{\gamma, \beta}(x_i) \quad // \text{scale and shift}$$

#### Phenomenon

1. Larger gradient -> more linear
2. More linear -> harder to fit complex system
3. Harder to fit -> bad result

#### Explanation

1. Finetune each input
2. Gradually map the normalized input to original one
3. Maintain the ability of fitting complex system

# II. Functional Layers

## D. Functional Layers Concepts & Implements

### D2. Batch Normalization-Theory

**Input:** Values of  $x$  over a mini-batch:  $\mathcal{B} = \{x_{1\dots m}\}$ ;

Parameters to be learned:  $\gamma, \beta$

**Output:**  $\{y_i = \text{BN}_{\gamma, \beta}(x_i)\}$

$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m} \sum_{i=1}^m x_i \quad // \text{mini-batch mean}$$

$$\sigma_{\mathcal{B}}^2 \leftarrow \frac{1}{m} \sum_{i=1}^m (x_i - \mu_{\mathcal{B}})^2 \quad // \text{mini-batch variance}$$

$$\hat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}} \quad // \text{normalize}$$

$$y_i \leftarrow \gamma \hat{x}_i + \beta \equiv \text{BN}_{\gamma, \beta}(x_i) \quad // \text{scale and shift}$$

**Benefits** (more exp. in course)

1. Increase learning rate
2. Speed up training
3. Less prone to overfit  
Remove or use less dropout  
Reduce L2 regularization weight
4. Less image augmentation

# II. Functional Layers

## D. Functional Layers Concepts & Implements

### D2. Batch Normalization-Practice

**Where To Put?**

**Usually:**

Conv -> Batch Norm -> ReLU->Pool

**Some argues:**

Conv -> ReLU -> Batch Norm ->Pool

# II. Functional Layers

## D. Functional Layers Concepts & Implements

### D2. Batch Normalization-Practice

#### Train & Test?

**Train:**

Get mean & std for each batch

**Test:**

Use recorded mean & std (usually last trained batch's)

# II. Functional Layers

## D. Functional Layers Concepts & Implements

### D2. Batch Normalization-Calculation

#### How To BP?

**Input:** Values of  $x$  over a mini-batch:  $\mathcal{B} = \{x_{1\dots m}\}$ ;

Parameters to be learned:  $\gamma, \beta$

**Output:**  $\{y_i = \text{BN}_{\gamma, \beta}(x_i)\}$

$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m} \sum_{i=1}^m x_i \quad // \text{mini-batch mean}$$

$$\sigma_{\mathcal{B}}^2 \leftarrow \frac{1}{m} \sum_{i=1}^m (x_i - \mu_{\mathcal{B}})^2 \quad // \text{mini-batch variance}$$

$$\hat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}} \quad // \text{normalize}$$

$$y_i \leftarrow \gamma \hat{x}_i + \beta \equiv \text{BN}_{\gamma, \beta}(x_i) \quad // \text{scale and shift}$$



# II. Functional Layers

## D. Functional Layers Concepts & Implements

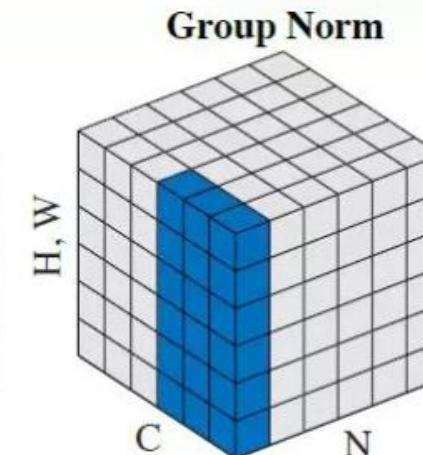
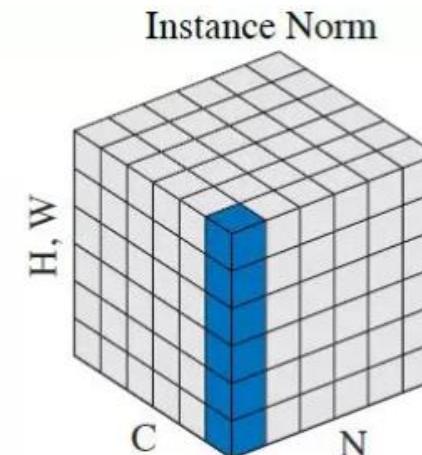
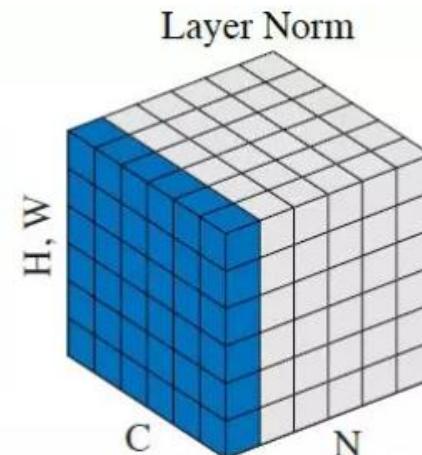
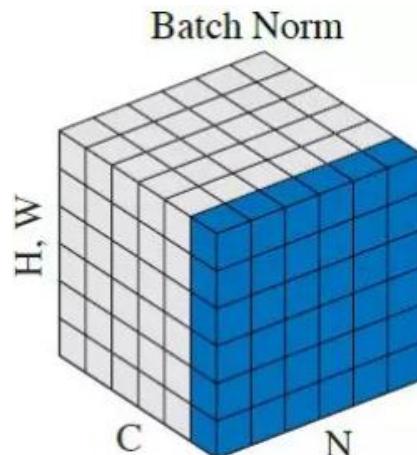
### D2. Batch Normalization-Others

Group Norm, Instance Norm, Layer Norm

*Reading [Optional]* 😊

**Answer me:**

1. Area to use.
2. When to use.
3. How to use.
4. Compare each.  
Pros & Cons



# Summary

## I. Fundamental Layers

**Conv: Conv, Dilated Conv, Transposed Conv, C3D (Deformable Conv)**

**ReLU: ReLU, Leaky-ReLU, PReLU, ELU (SeLU)**

**Pooling: Max Pooling, Ave Pooling**

**FC**

## II. Functional Layers

**Dropout**

**Batch Normalization (Group / Layer / Instance Norm)**