Convolution and Cross-correlation

Dr. Thanh-Sach LE
LTSACH@hcmut.edu.vn

GVLab:
Graphics and Vision Laboratory

Faculty of Computer Science and Engineering,
HCMUT

Contents

Convolution and Cross-correlation

What & Why?

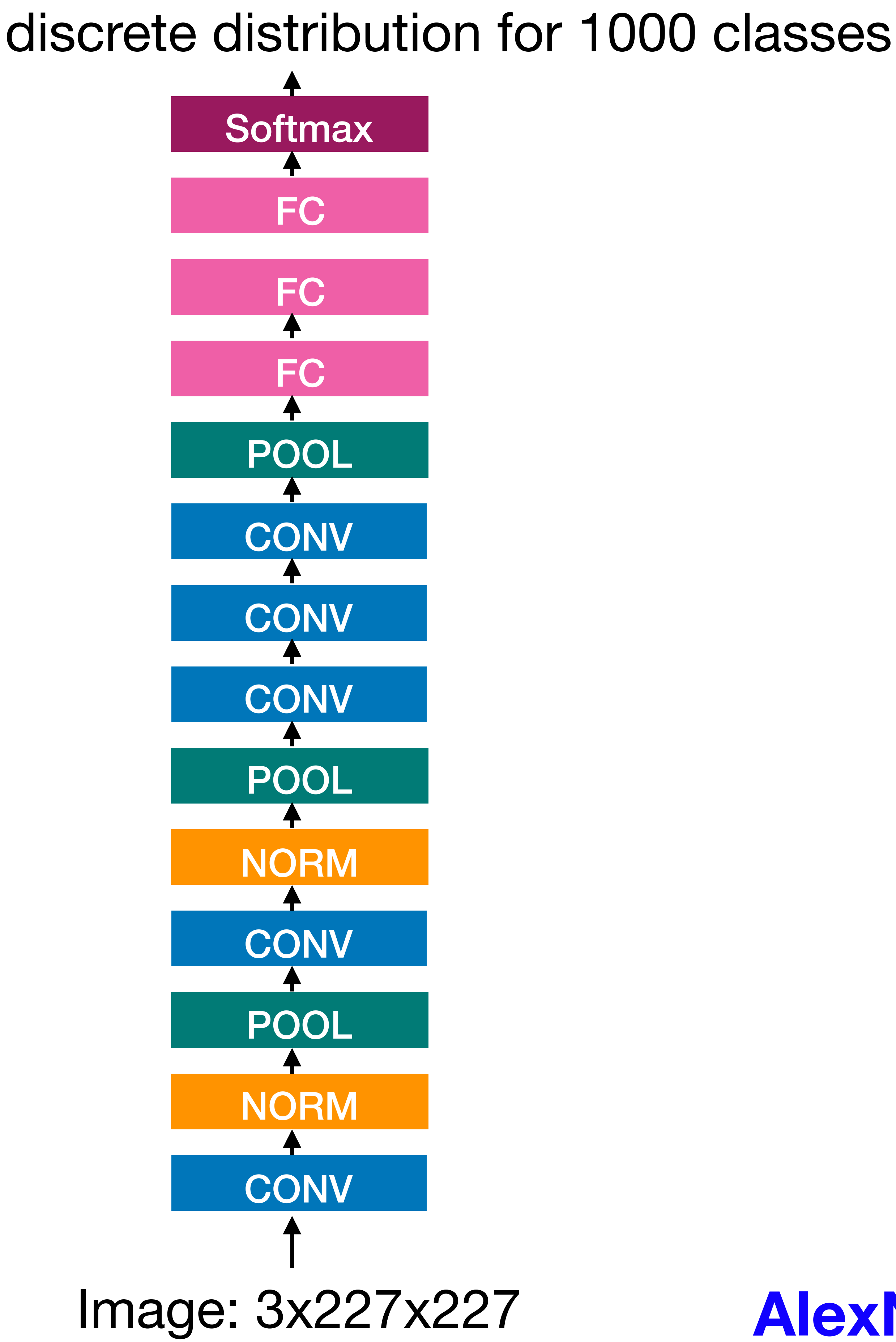
Dr. Thanh-Sach LE
LTSACH@hcmut.edu.vn

GVLab:
Graphics and Vision Laboratory

Faculty of Computer Science and Engineering,
HCMUT

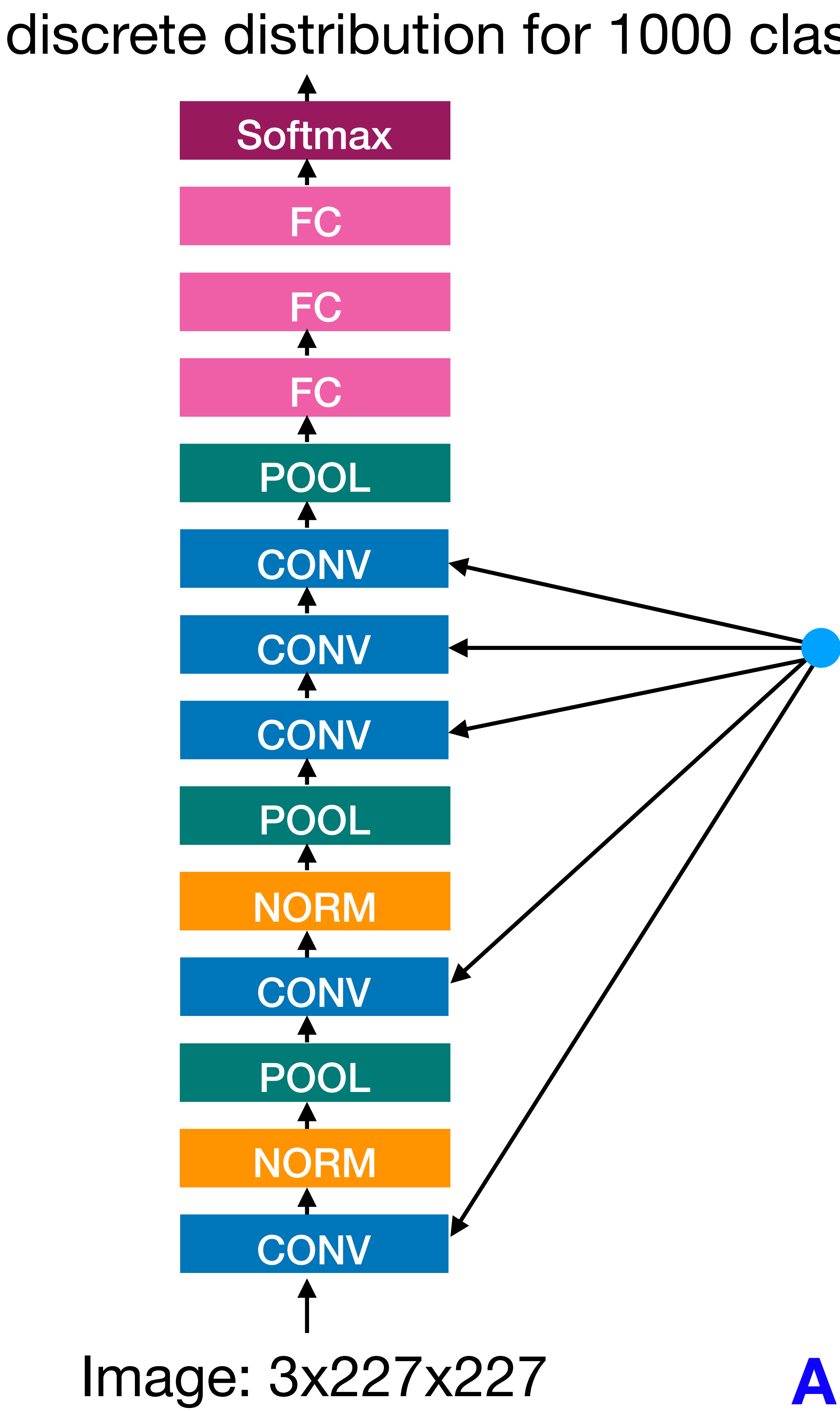
What & Why?

- Convolution and cross-correlation
  - important operations in signal processing
  - used to transform input signal (e.g. image) to output feature map and from input feature map to output feature map
  - hence, in deep neural network they are used to learn (to extract) features from input signals
- Convolution in Deep Learning
  - Convolution is widely used to design deep neural networks (see following slides)



AlexNet (2012)

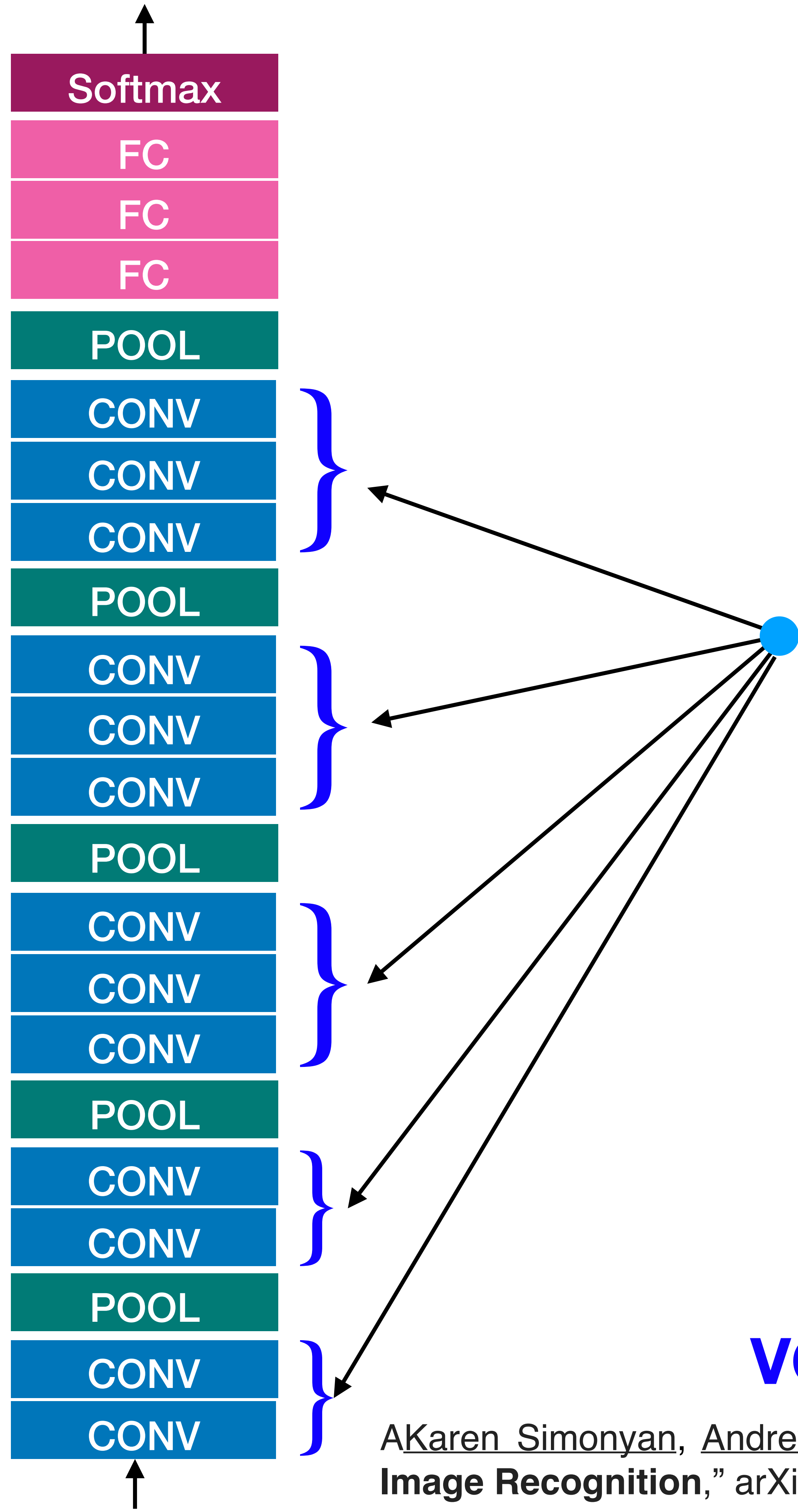
Alex Krizhevsky and Sutskever, Ilya and Hinton, Geoffrey E, "**ImageNet Classification with Deep Convolutional Neural Networks**," in Advances in Neural Information Processing Systems, pp.1097-1105, 2012



They used convolution to extract features by transforming from raw pixels to meaningful representation

AlexNet (2012)

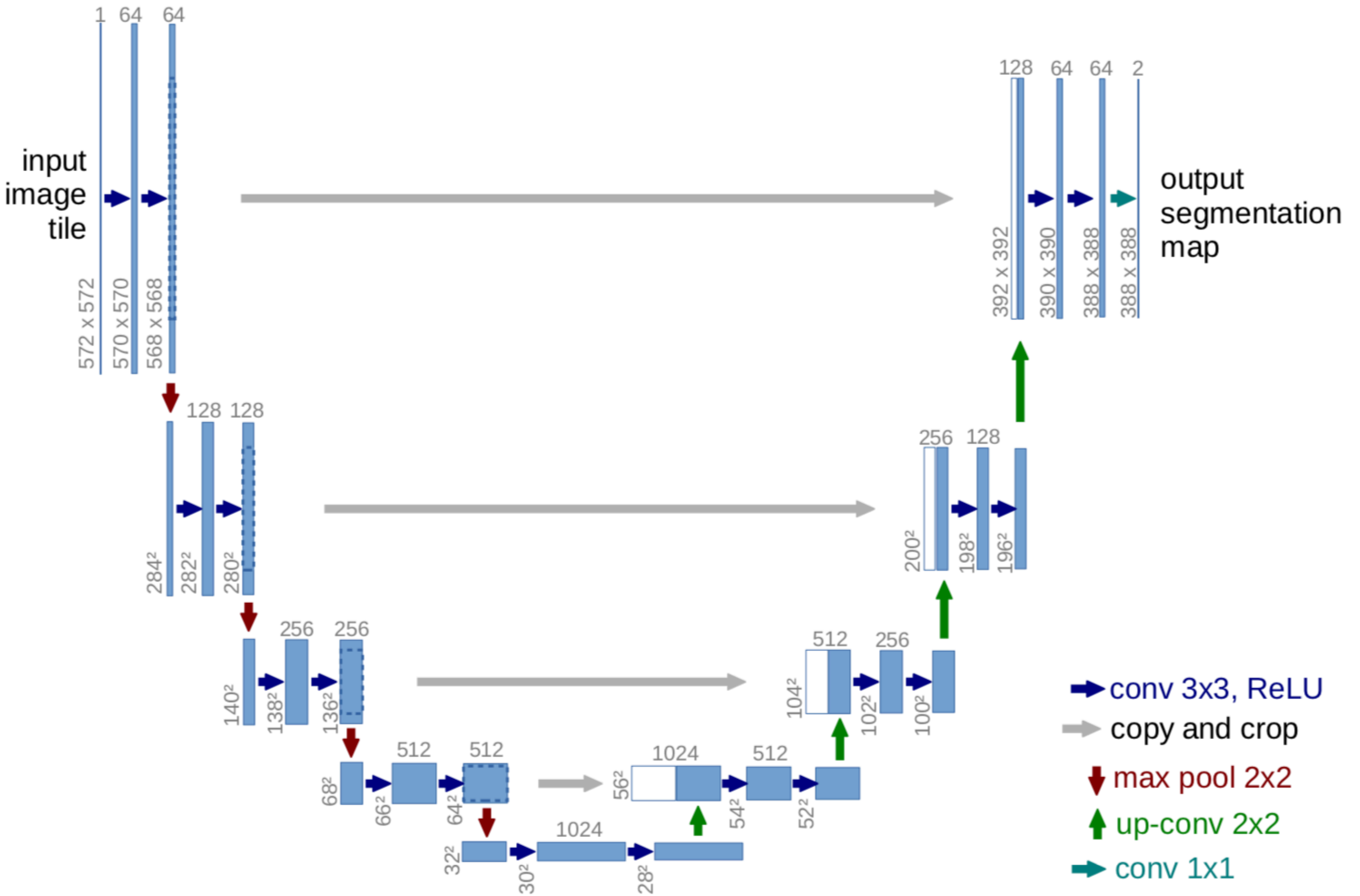
Alex Krizhevsky and Sutskever, Ilya and Hinton, Geoffrey E, "**ImageNet Classification with Deep Convolutional Neural Networks**," in Advances in Neural Information Processing Systems, pp.1097-1105, 2012



They used convolution to extract features by transforming from raw pixels to meaningful representation

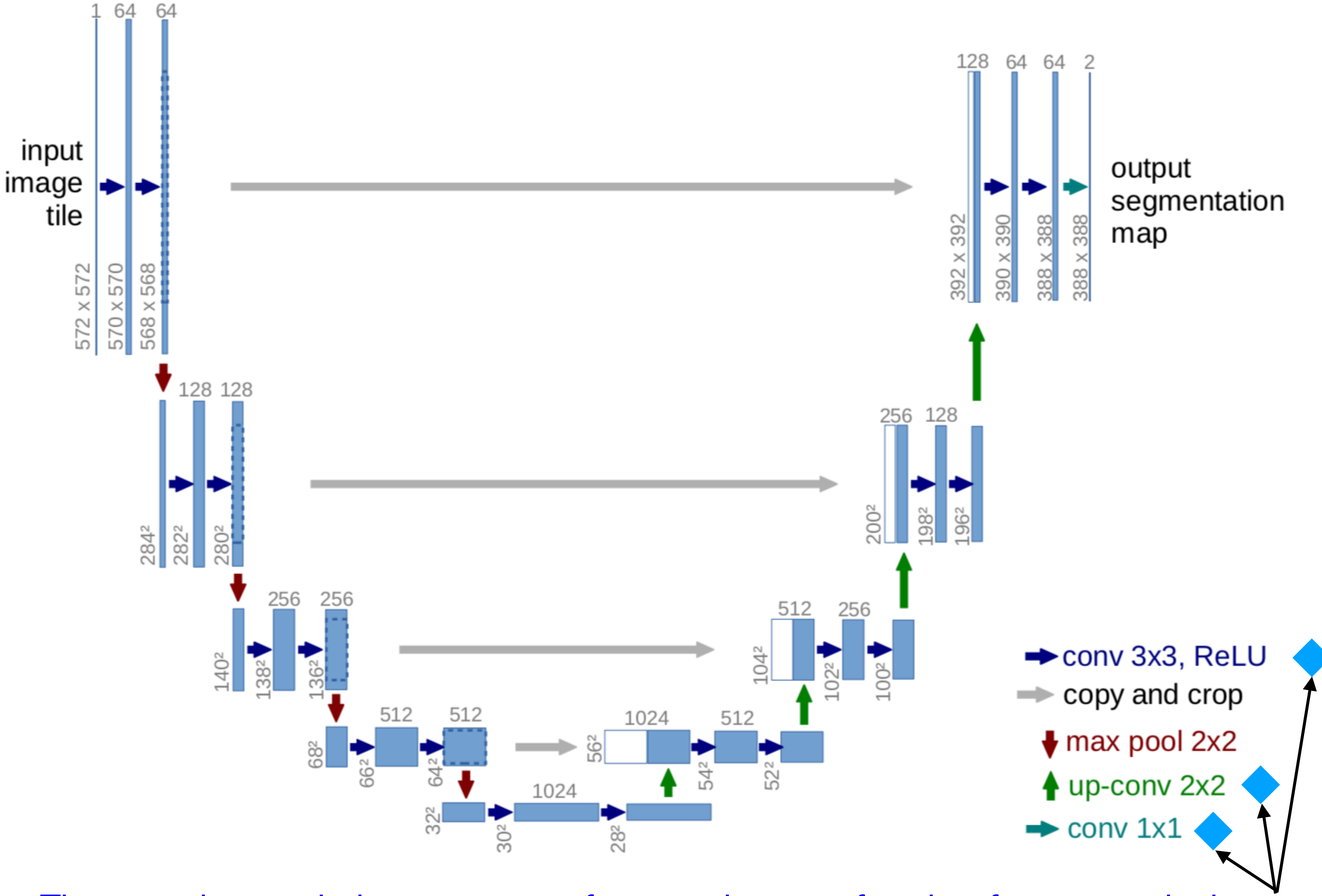
VGG-16 (2014)

AKaren Simonyan, Andrew Zisserman, “**Very Deep Convolutional Networks for Large-Scale Image Recognition**,” arXiv:1409.1556v6



UNet (2015)

Olaf Ronneberger, Philipp Fischer, Thomas Brox, "**U-Net: Convolutional Networks for Biomedical Image Segmentation**," arXiv:1505.04597v1 [cs.CV]



They used convolution to extract features by transforming from raw pixels to meaningful representation

UNet (2015)

Olaf Ronneberger, Philipp Fischer, Thomas Brox, "**U-Net: Convolutional Networks for Biomedical Image Segmentation**," arXiv:1505.04597v1 [cs.CV]

Convolution and Cross-correlation

Mathematical Definition

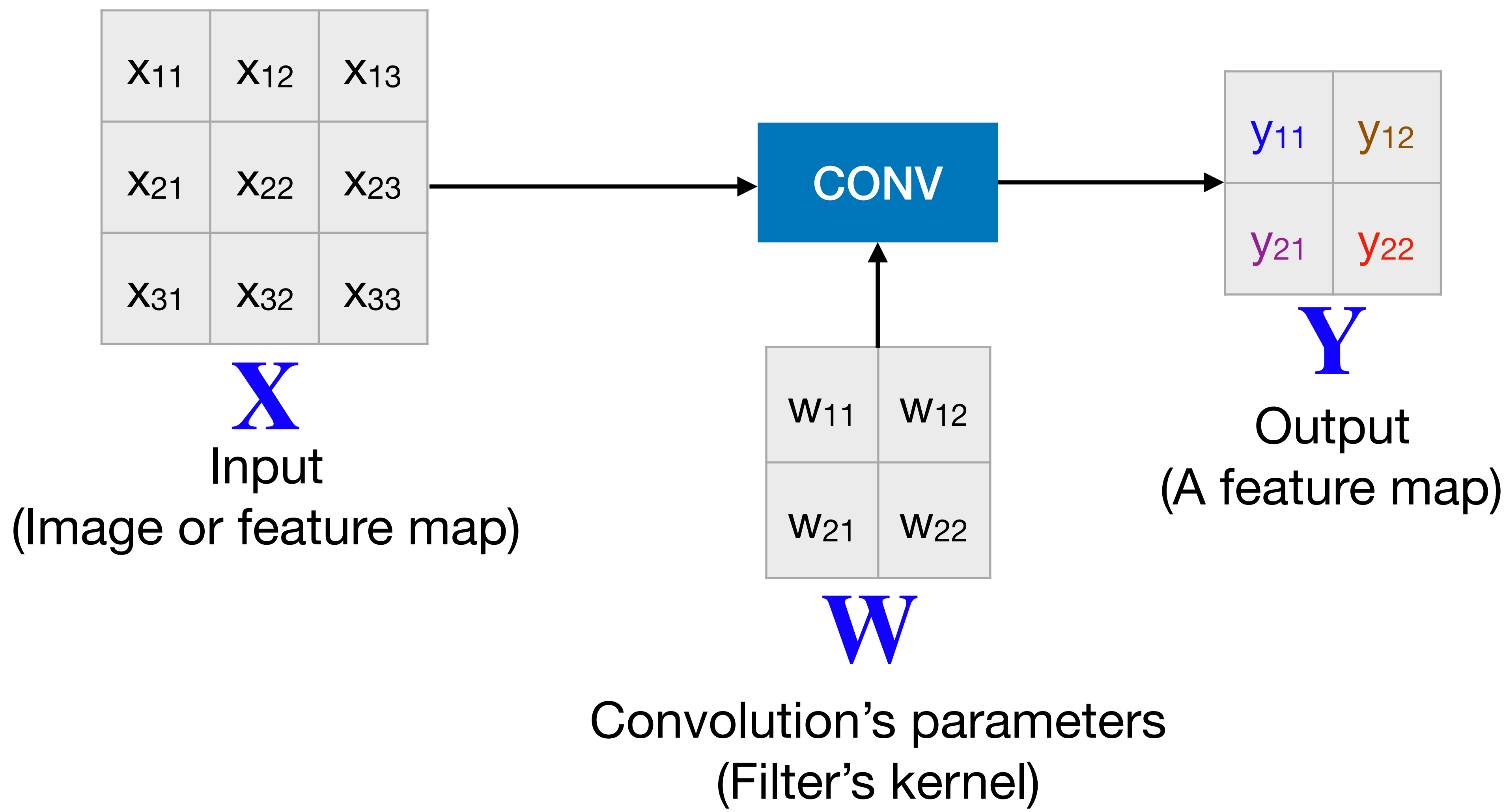
Dr. Thanh-Sach LE
LTSACH@hcmut.edu.vn

GVLab:
Graphics and Vision Laboratory

Faculty of Computer Science and Engineering,
HCMUT

Mathematical Definition

❖ Computation node:

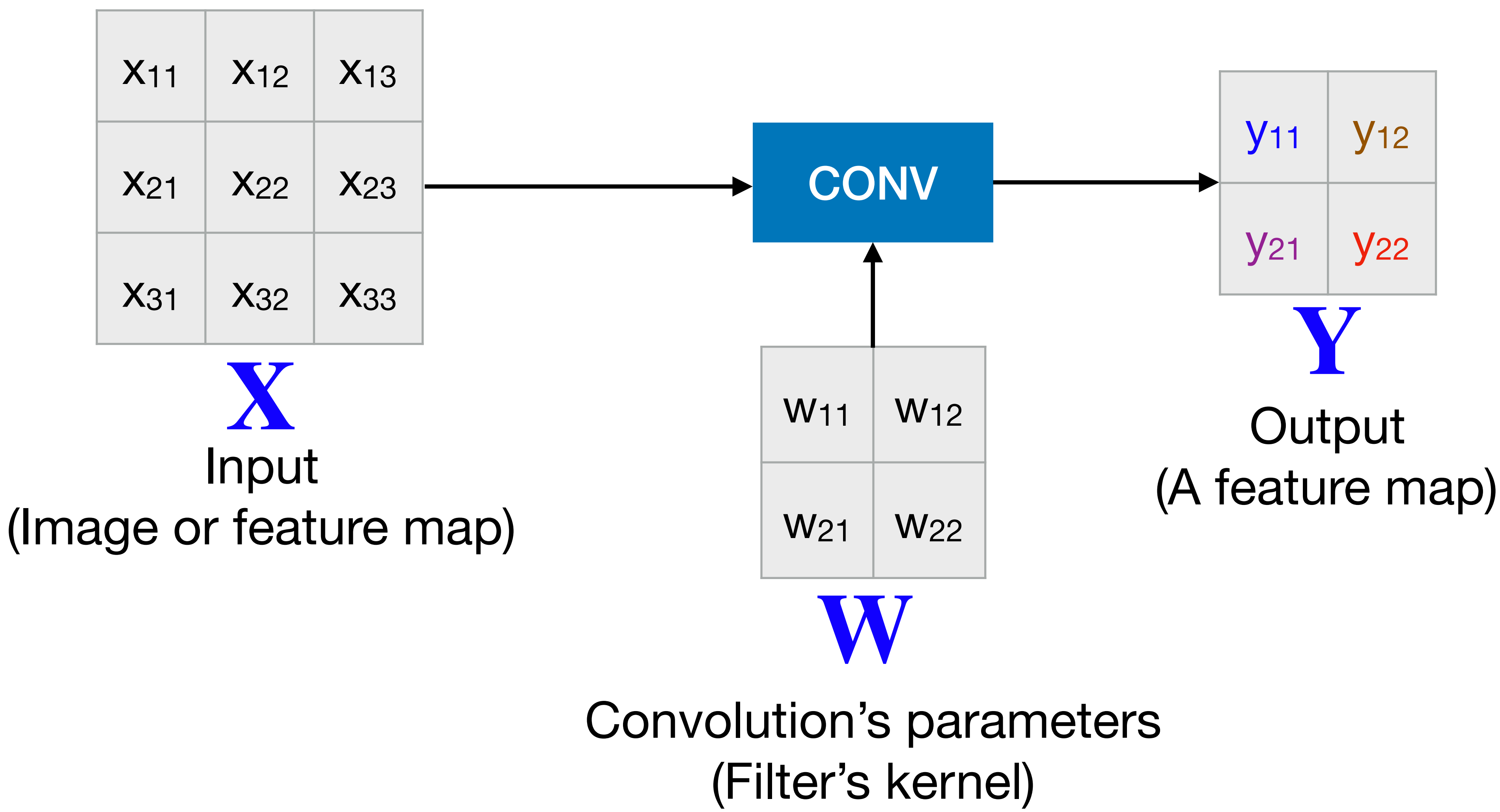


❖ Notation:

$$\mathbf{Y} = \mathbf{X} * \mathbf{W}$$

Mathematical Definition

❖ Computation node:



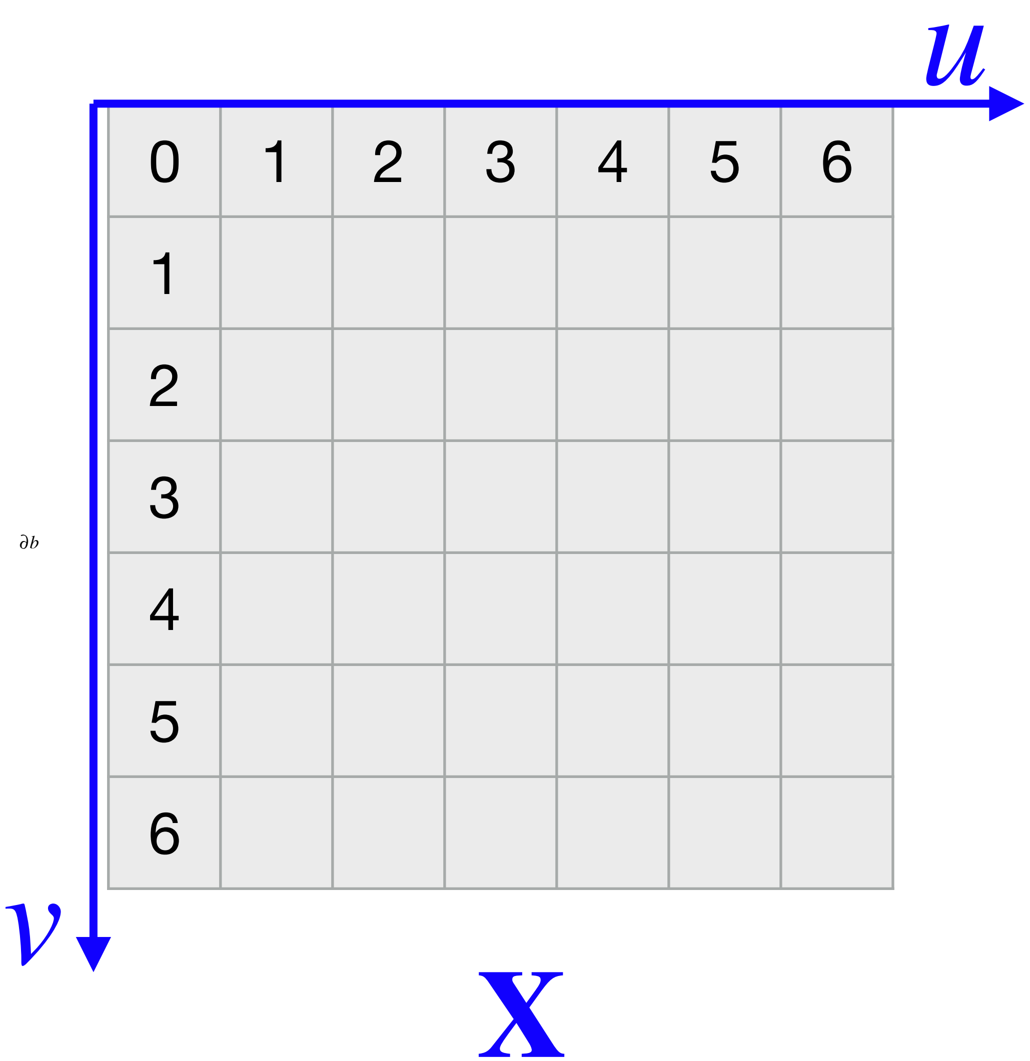
❖ Notation:

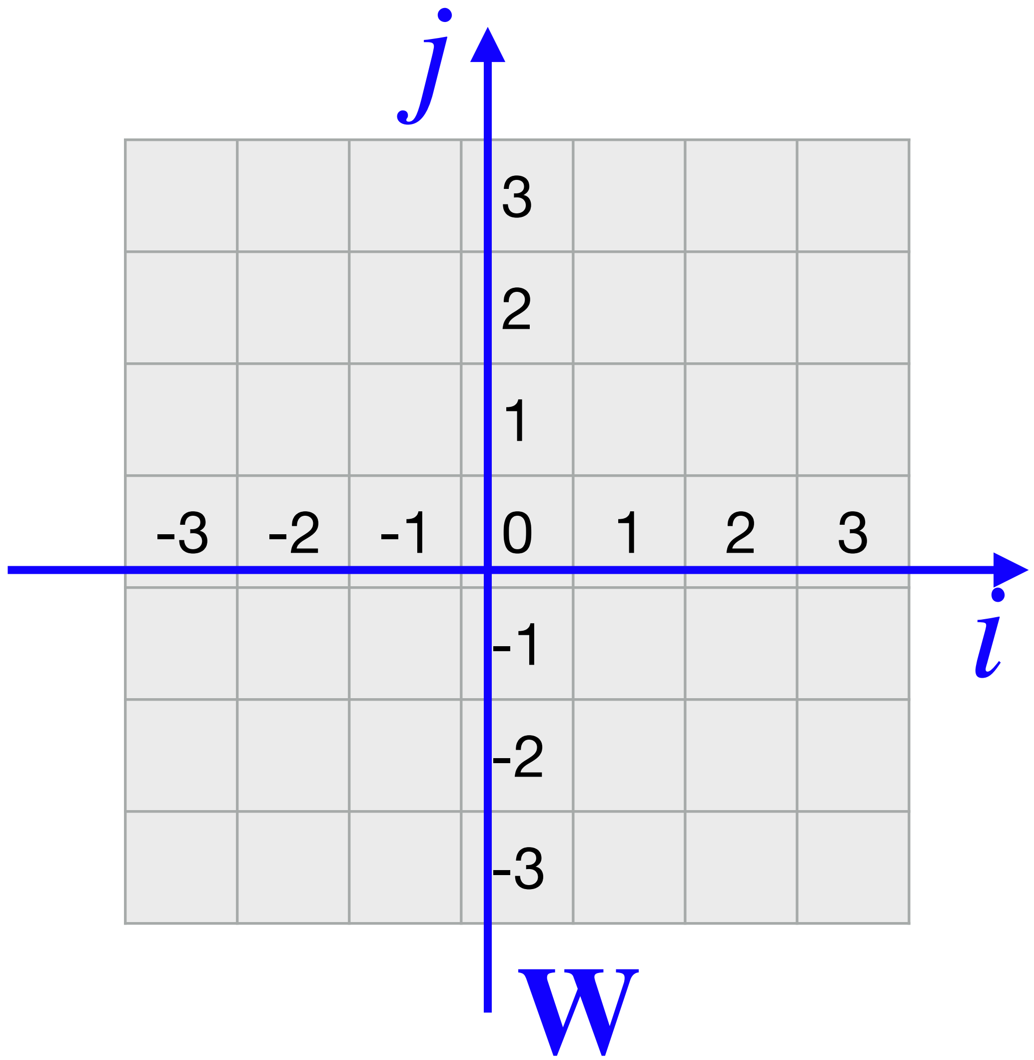
$$\mathbf{Y} = \mathbf{X} * \mathbf{W}$$

* : **NOT a matrix multiplication**

Mathematical Definition

❖ Definition:





Radius of kernel:

$$\mathbf{W} : 7x7 \Rightarrow r = \left\lfloor \frac{7}{2} \right\rfloor = 3$$

Mathematical Definition

❖ Definition:

∂b

Convolution

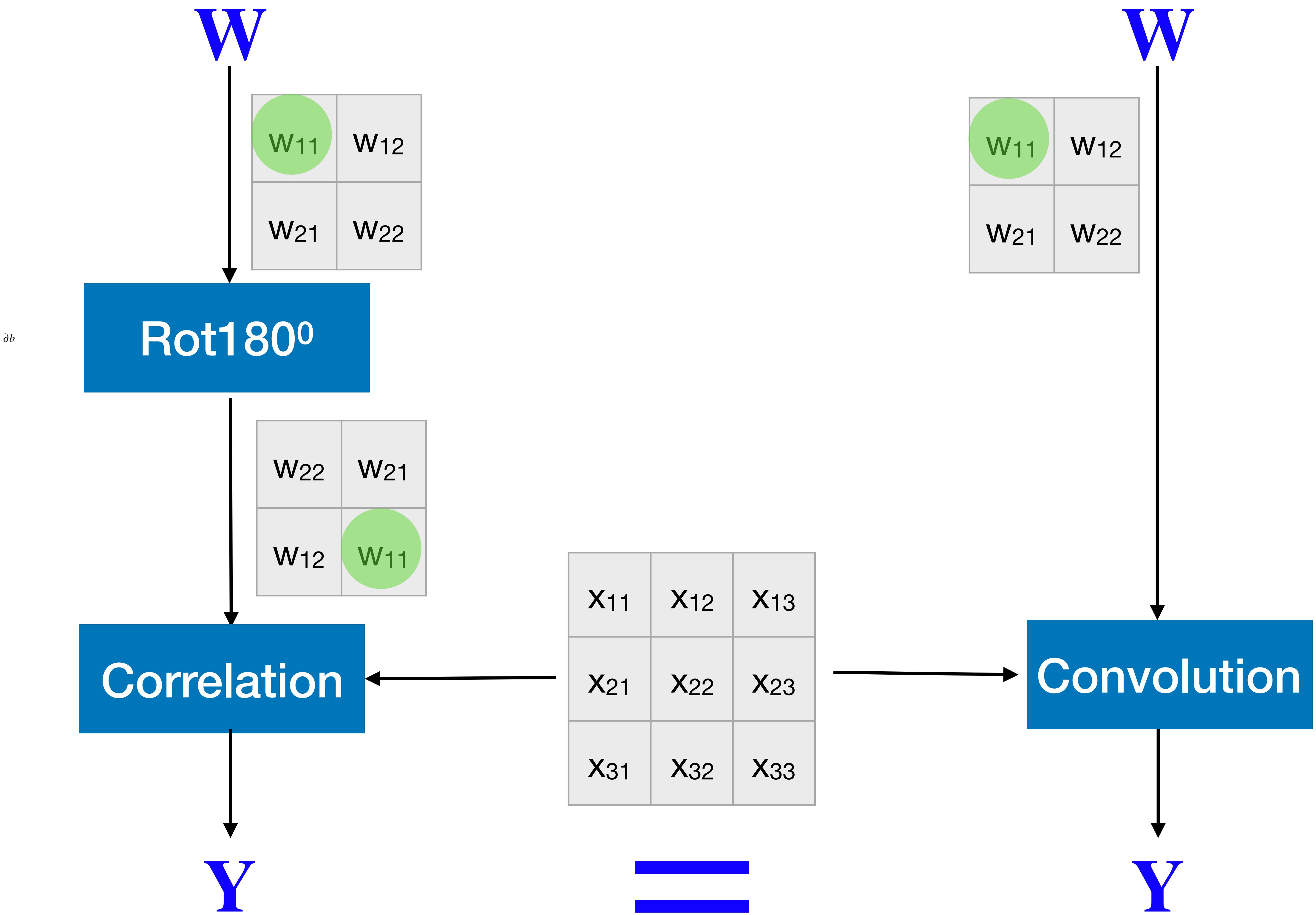
$$\begin{cases} \mathbf{Y}(u,v) = \mathbf{X} * \mathbf{W} \\ \qquad\quad\;\; = \displaystyle\sum_{i=-r}^{r} \sum_{j=-r}^{r} \mathbf{X}(u-i, v-j)\mathbf{W}(i,j) \end{cases}$$

Cross-Correlation

$$\begin{cases} \mathbf{Y}(u,v) = \mathbf{X} \star \mathbf{W} \\ \qquad\quad\;\; = \displaystyle\sum_{i=-r}^{r} \sum_{j=-r}^{r} \mathbf{X}(u+i, v+j)\mathbf{W}(i,j) \end{cases}$$

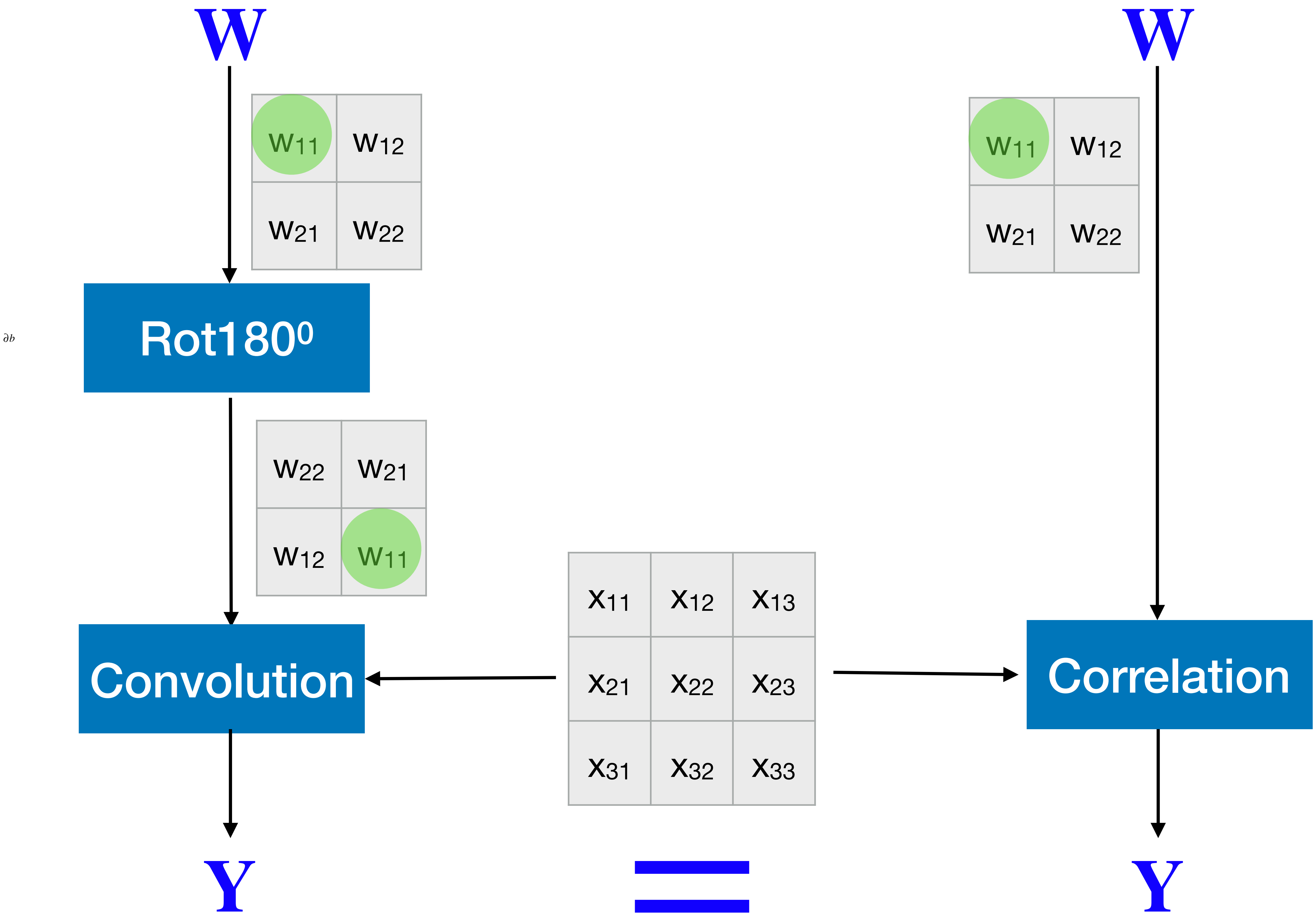
Mathematical Definition

❖ Convolution vs cross-correlation:



Mathematical Definition

❖ Convolution vs cross-correlation:



Convolution and Cross-correlation

Computation of convolution

Dr. Thanh-Sach LE
LTSACH@hcmut.edu.vn

GVLab:
Graphics and Vision Laboratory

Faculty of Computer Science and Engineering,
HCMUT

Computation of convolution

- **How to perform the computation (Logical view):**

  Step 1:

  - Flip the kernel around the x-axis and then around the y-axis
  - Or, rotate the kernel 180^o around its center

∂b

$$\begin{bmatrix} W_{11} & W_{12} \\ W_{21} & W_{22} \end{bmatrix} \rightarrow \begin{bmatrix} W_{22} & W_{21} \\ W_{12} & W_{11} \end{bmatrix}$$

$$\begin{bmatrix} W_{11} & W_{12} & W_{13} \\ W_{21} & W_{22} & W_{23} \\ W_{31} & W_{32} & W_{33} \end{bmatrix} \rightarrow \begin{bmatrix} W_{33} & W_{32} & W_{31} \\ W_{23} & W_{22} & W_{21} \\ W_{13} & W_{12} & W_{11} \end{bmatrix}$$

Computation of convolution

- **How to perform the computation (Logical view):**

  Step 2:

  - Compute the cross-correlation between the kernel (obtained from Step 1) with the input image, by:

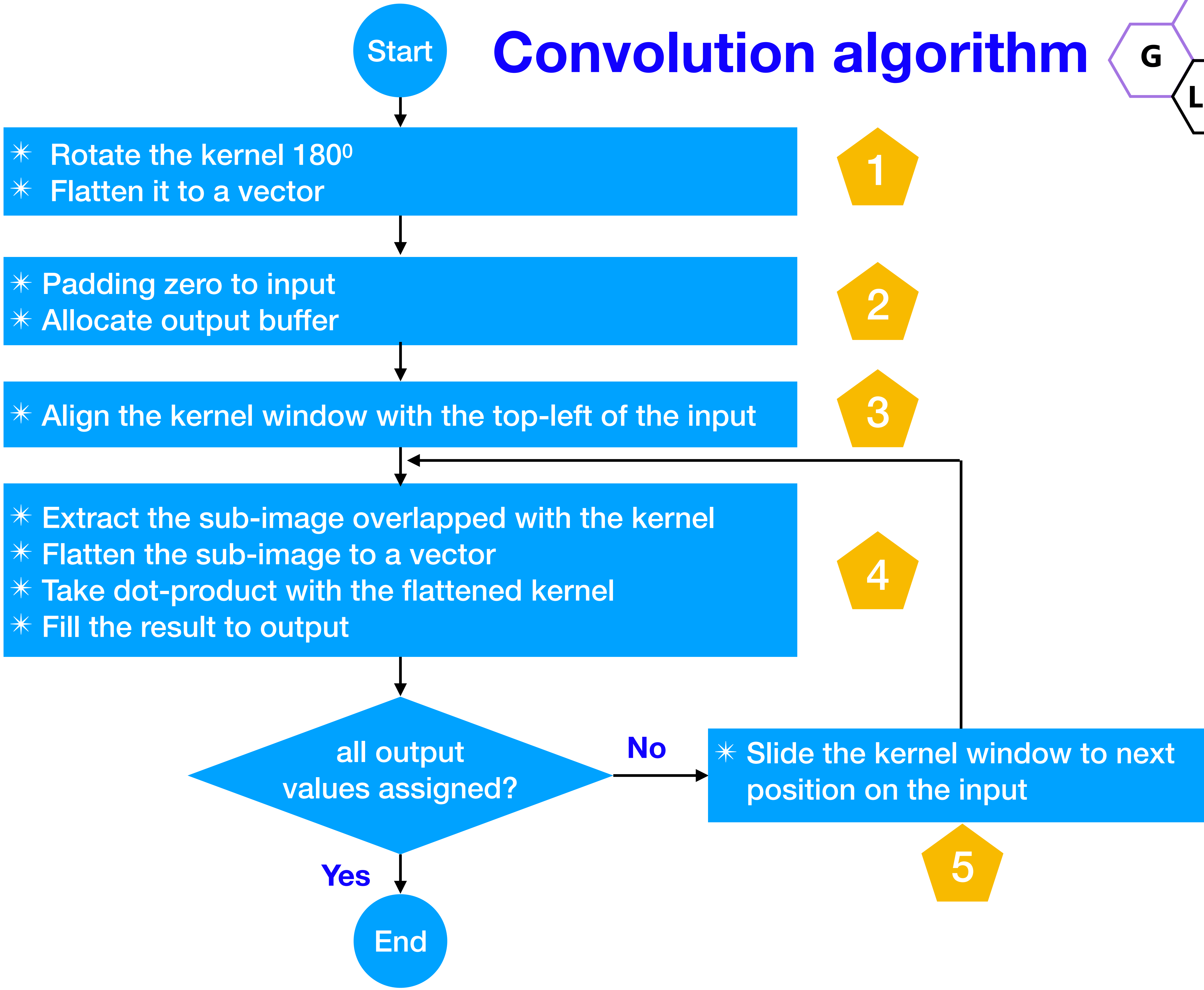
    (b) Place the kernel aligned with the left-top of the image

    (c) Take the **dot product** between the kernel and the sub-image occupied by the kernel and assign the result to the output at the corresponding location

    (d) Slide the kernel to left and down; do task (b) after each sliding

See the following illustration for detail

Convolution algorithm



Computation of convolution

3	1	0	1
1	1	2	0
1	2	2	1
0	1	0	2

Input image

$\longrightarrow$ CONV $\longrightarrow$ **Output ?**

1	0	2
1	2	0
0	1	1

Filter's kernel

Computation of convolution

1. **Rotate the kernel**

$$W = \begin{bmatrix} 1 & 0 & 2 \\ 1 & 2 & 0 \\ 0 & 1 & 1 \end{bmatrix} \xrightarrow{\text{Rotation } 180^o} Rot180^0(W) = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 2 & 1 \\ 2 & 0 & 1 \end{bmatrix}$$

$Rot180^0(W) =$ **Flip on horizontal direction** $+$ **Flip on vertical direction**

23 Computation of convolution

1 Flatten the rotated kernel

$$W = \begin{bmatrix} 1 & 0 & 2 \\ 1 & 2 & 0 \\ 0 & 1 & 1 \end{bmatrix} \xrightarrow{\text{Rotation } 180^{o}} \mathrm{Rot180}^{0}(W) = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 2 & 1 \\ 2 & 0 & 1 \end{bmatrix}$$

Flattening

1	1	0	0	2	1	2	0	1

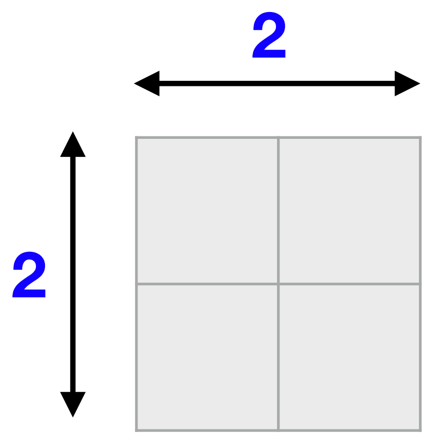
Computation of convolution

2 Padding the input

3	1	0	1
1	1	2	0
1	2	2	1
0	1	0	2

X

This section illustrates convolution with no padding, stride = 1



Y
Output

Output size: 2x2 will be clear shortly

Computation of convolution

3 Align the rotated kernel with the input image

3	1	0	1
1	1	2	0
1	2	2	1
0	1	0	2

Input image

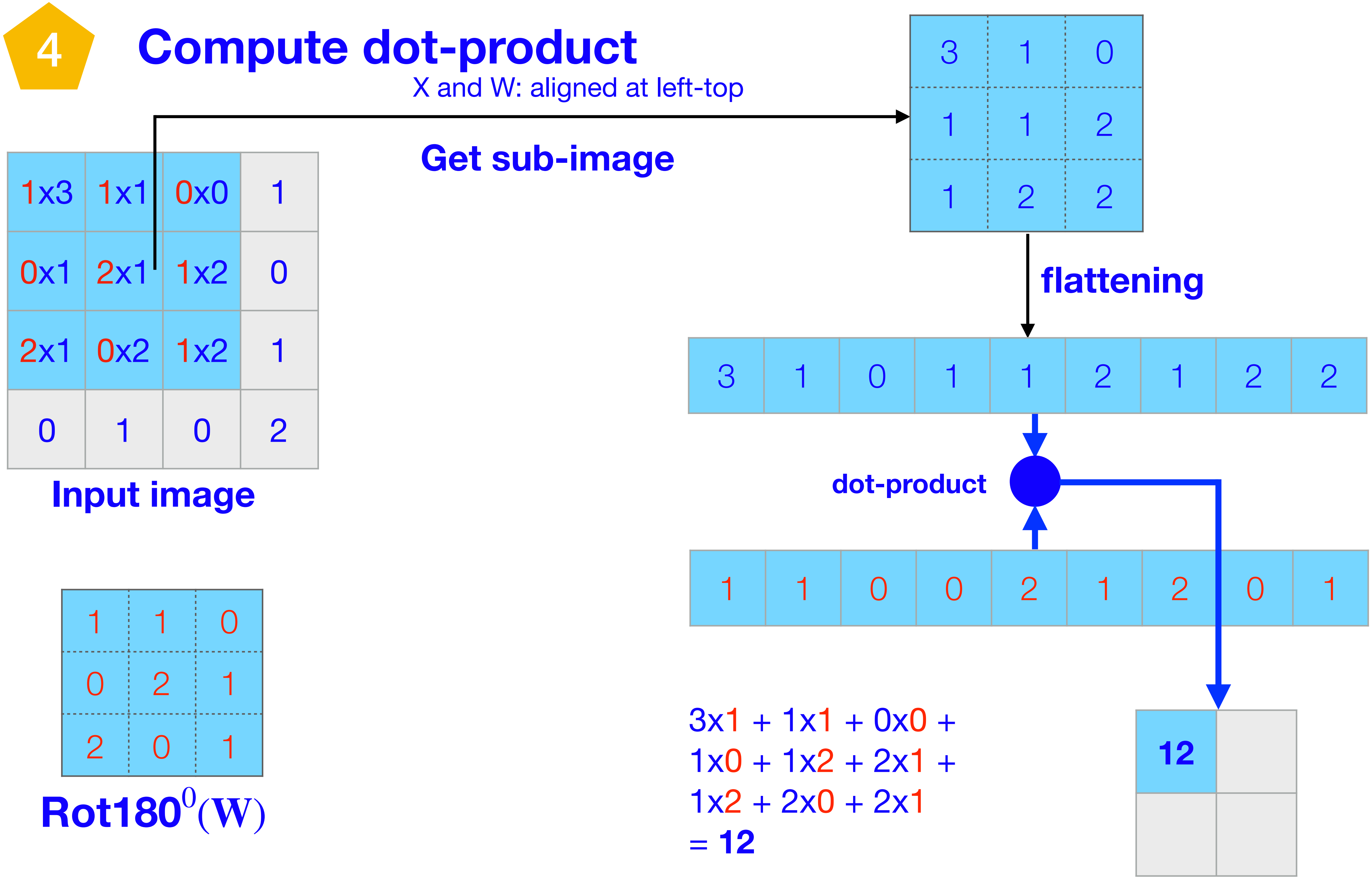
1	1	0
0	2	1
2	0	1

Rot180^0(W)

1x3	1x1	0x0	1
0x1	2x1	1x2	0
2x1	0x2	1x2	1
0	1	0	2

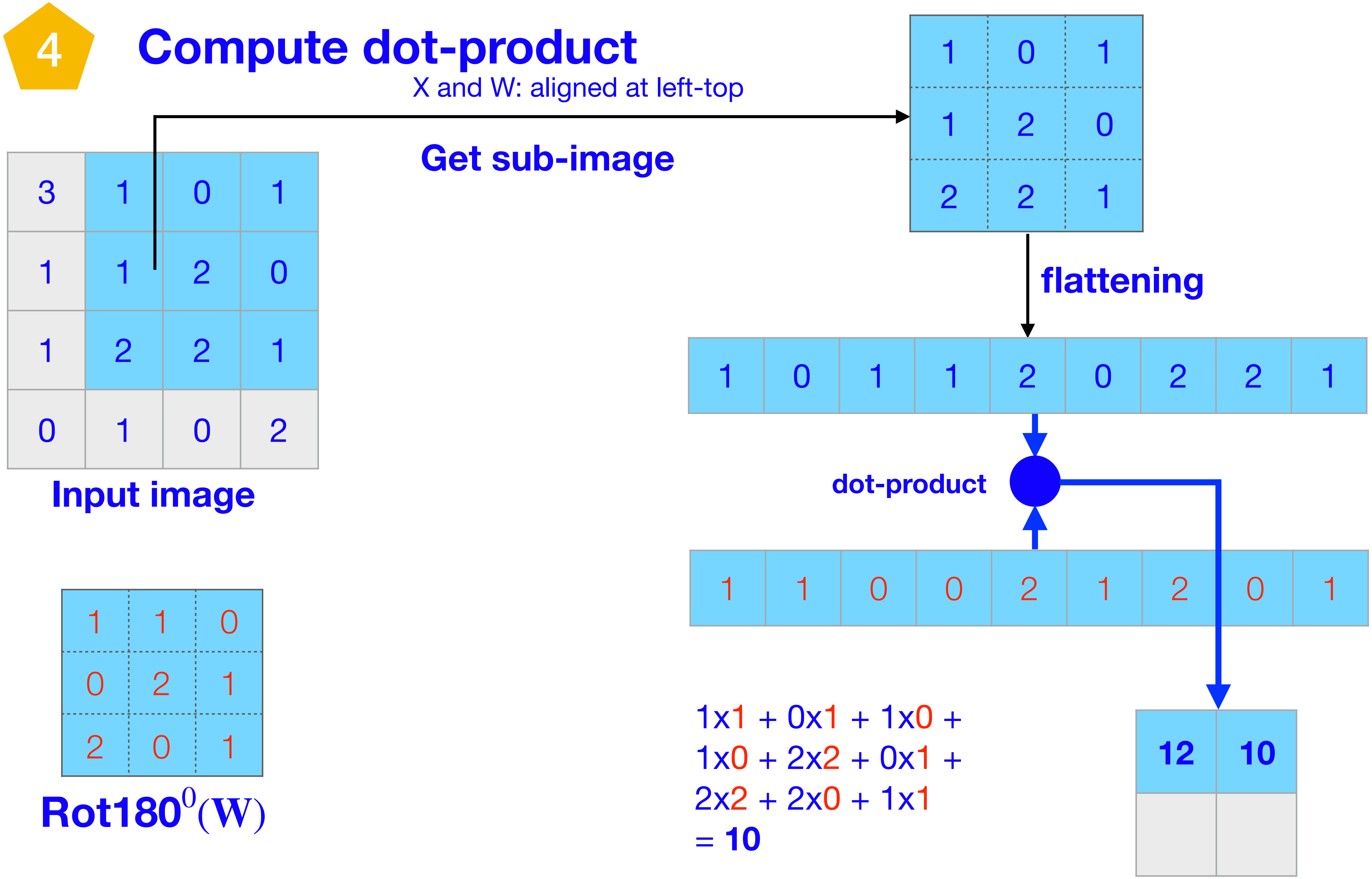
Computation of convolution

4 Compute dot-product



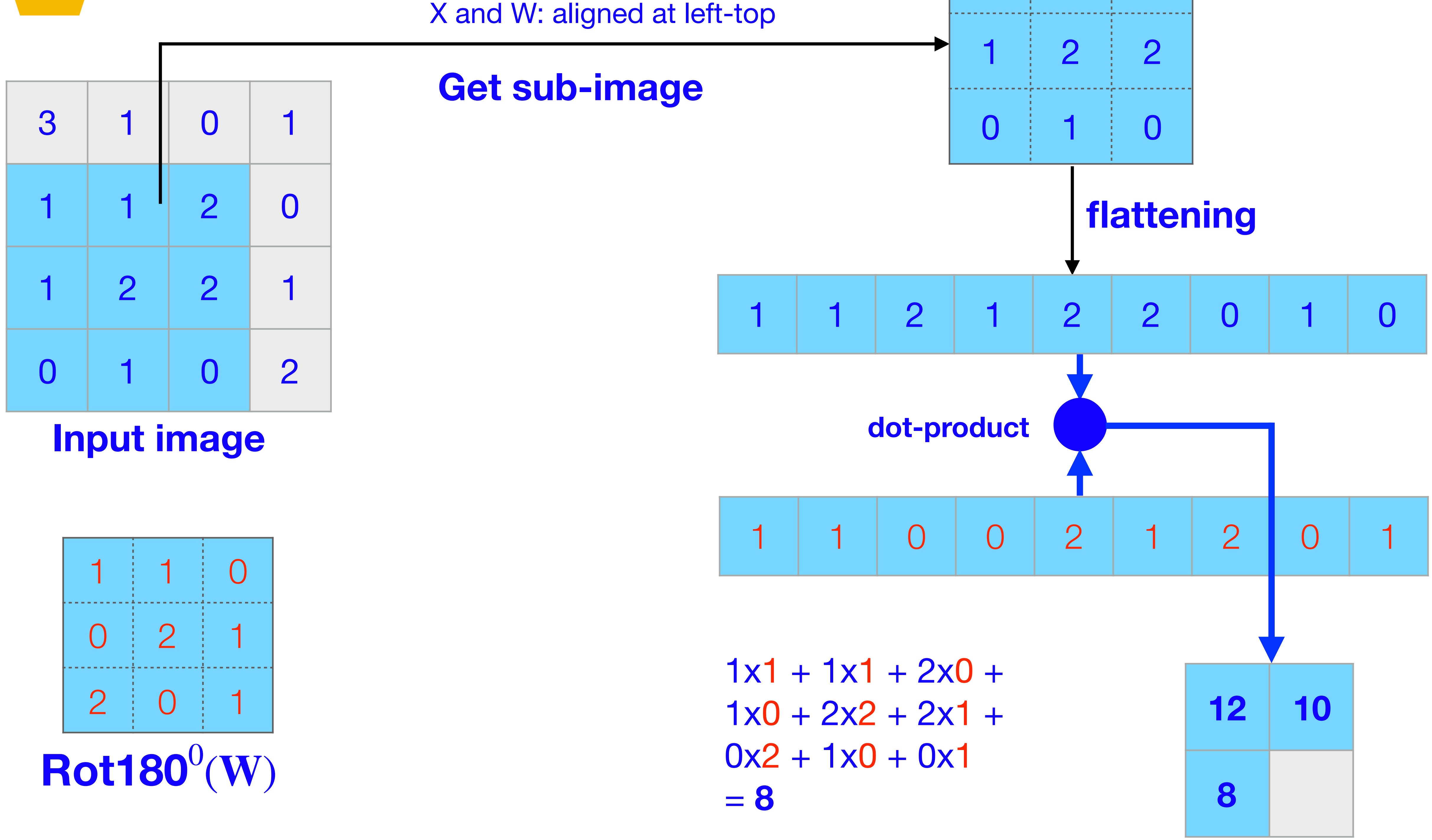
27 Computation of convolution

4 Compute dot-product



Computation of convolution

4 Compute dot-product



Computation of convolution

4 Compute dot-product

X and W: aligned at left-top

Get sub-image

3	1	0	1
1	1	2	0
1	2	2	1
0	1	0	2

Input image

1	2	0
2	2	1
1	0	2

flattening

1	2	0	2	2	1	1	0	2

dot-product

1	1	0	0	2	1	2	0	1

1	1	0
0	2	1
2	0	1

$\mathbf{Rot180^0(W)}$

1x1 + 2x1 + 0x0 +
2x0 + 2x2 + 1x1 +
1x2 + 0x0 + 2x1
= **12**

12	**10**
8	**12**

Computation of convolution

Final result

3	1	0	1
1	1	2	0
1	2	2	1
0	1	0	2

Input image

→ CONV →

12	10
8	12

Output

1	0	2
1	2	0
0	1	1

Filter's kernel

Computation of convolution

Final result

3	1	0	1
1	1	2	0
1	2	2	1
0	1	0	2

Input image

4×4

CONV

1	0	2
1	2	0
0	1	1

Filter's kernel

3×3

12	**10**
8	**12**

Output

$$(4-3+1) \times (4-3+1)$$

$$2 \times 2$$

Computation of convolution

Final result

3	1	0	1
1	1	2	0
1	2	2	1
0	1	0	2

Input image

4×4

CONV

1	0	2
1	2	0
0	1	1

Filter's kernel

3×3

12	**10**
8	**12**

Output

$(4-3+1) \times (4-3+1)$

2×2

$$i_1 \times i_2 \longrightarrow \underset{\substack{\uparrow \\ k_1 \times k_2}}{*} \longrightarrow (i_1 - k_1 + 1) \times (i_2 - k_2 + 1)$$