Introduction to Computer Vision

Computer vision is a field of artificial intelligence that enables computers and systems to derive meaningful information from **digital images**, **videos**, and **other visual inputs**, and to act or make recommendations based on that information.

The key aspects of computer vision, summarized as main points:

- **Image Recognition:** Identifying objects, people, and other elements within images.
- **Object Detection:** Recognizing and locating objects within an image using bounding boxes or other markers.
- **Image Segmentation:** Dividing an image into parts to simplify the analysis, often used in applications like medical imaging.
- **Pattern Recognition:** Recognizing patterns in visual data, such as shapes or movements.
- **Scene Reconstruction:** Reconstructing a 3D scene from images, used in augmented reality and robotics.
- **Video Tracking:** Tracking objects or individuals across a video sequence.
- **Image Restoration:** Restoring or enhancing the quality of degraded images.

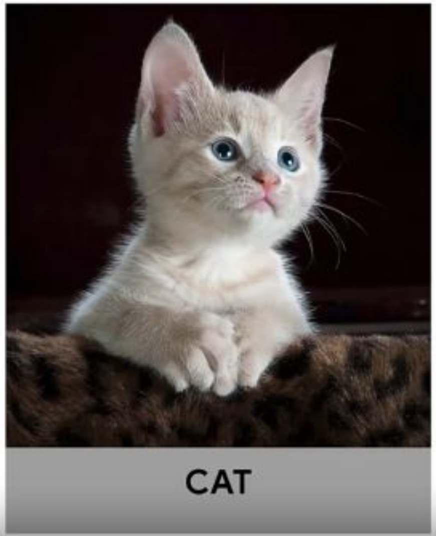
Image Classification

- Multi-class Classification
  - Binary Classification (Subset of the problem)
- Multi-label Classification

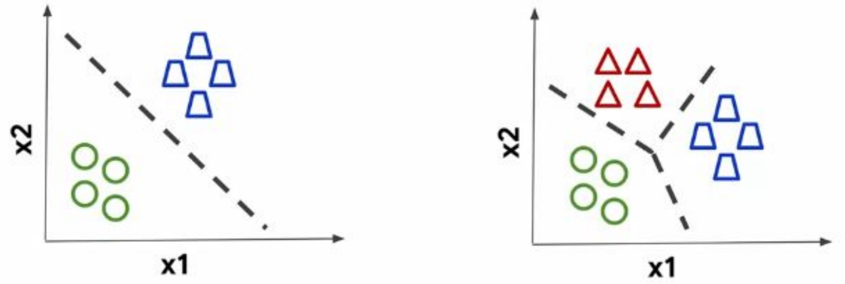
Image Classification

- Multi-class Classification
  - Binary Classification (Subset of the problem)
- Multi-label Classification

Image Classification



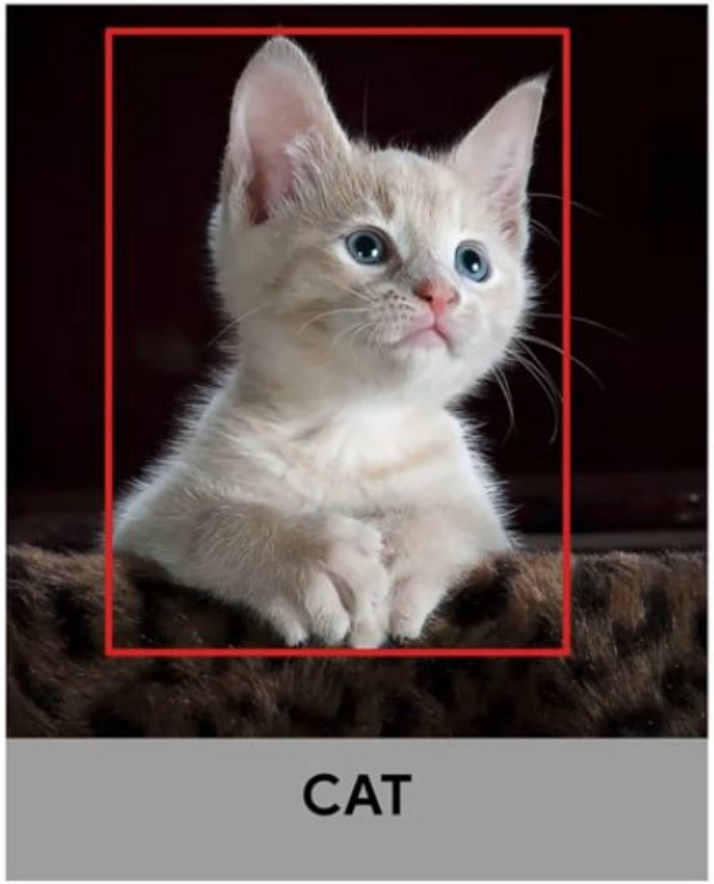
Binary vs. Multi Class Classification



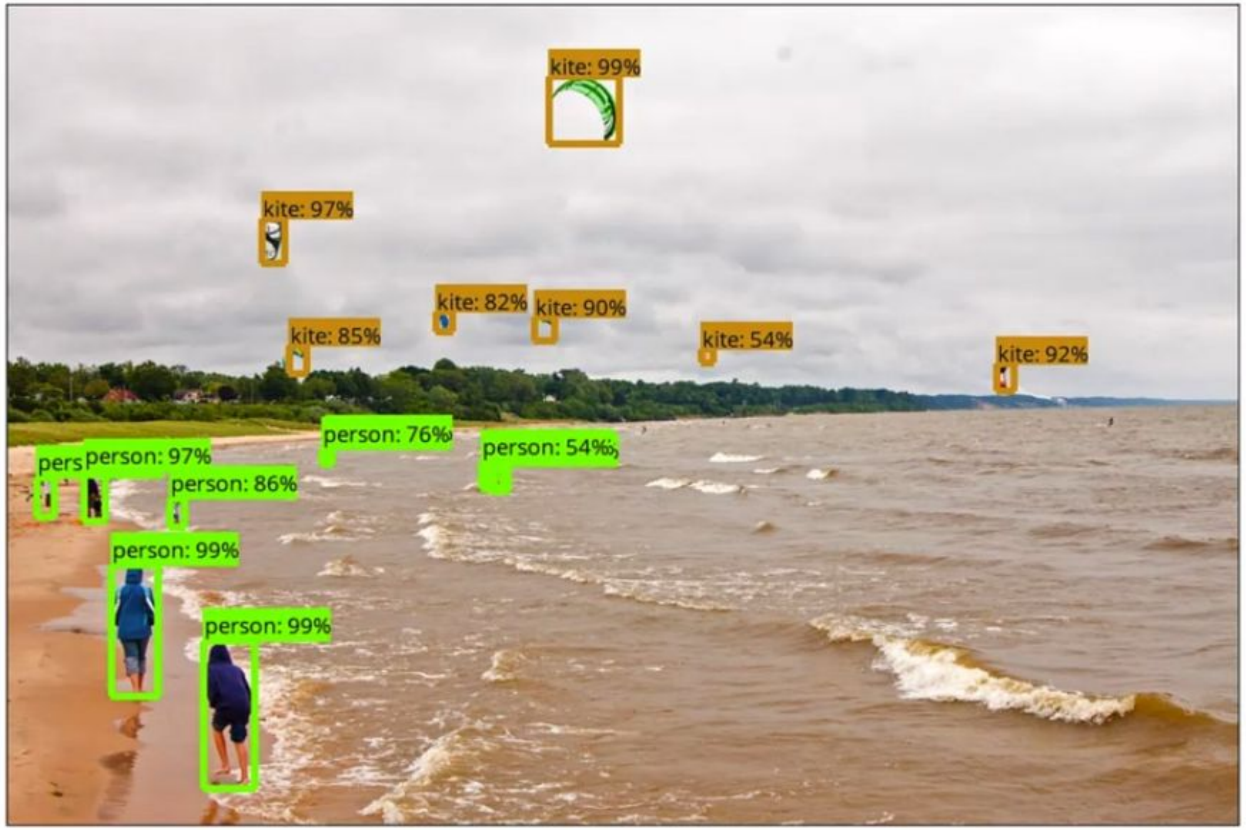
Binary Classification   Multi-class Classification

Multi-label Classification

Object Localization



Object Detection



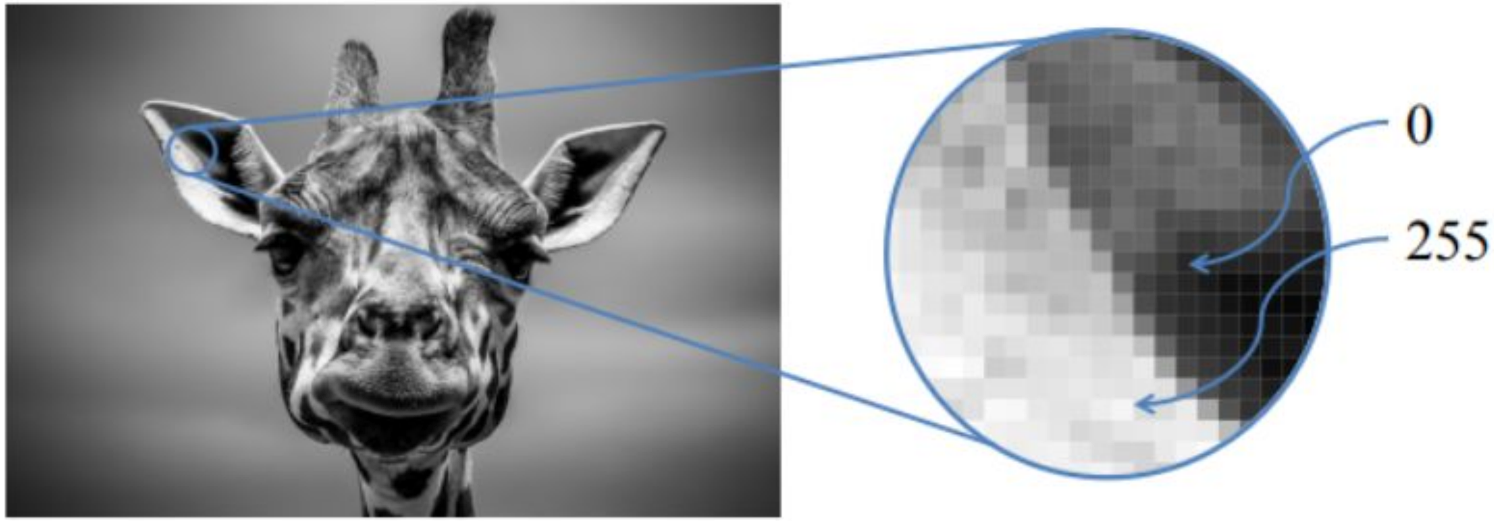
Semantic vs. Instance Segmentation

Semantic Segmentation

Instance Segmentation

Digital representation of an image

- Grayscale image is a matrix of pixels (**picture elements**)
- Dimensions of this matrix are called image resolution (e.g. 300 x 300)
- Each pixel stores its brightness (or **intensity**) ranging from 0 to 255, 0 intensity corresponds to black color:



- Color images store pixel intensities for 3 channels: red, green and blue

Image as a neural network input

- Normalize input pixels: $x_{norm} = \frac{x}{255} - 0.5$
- Maybe MLP will work?

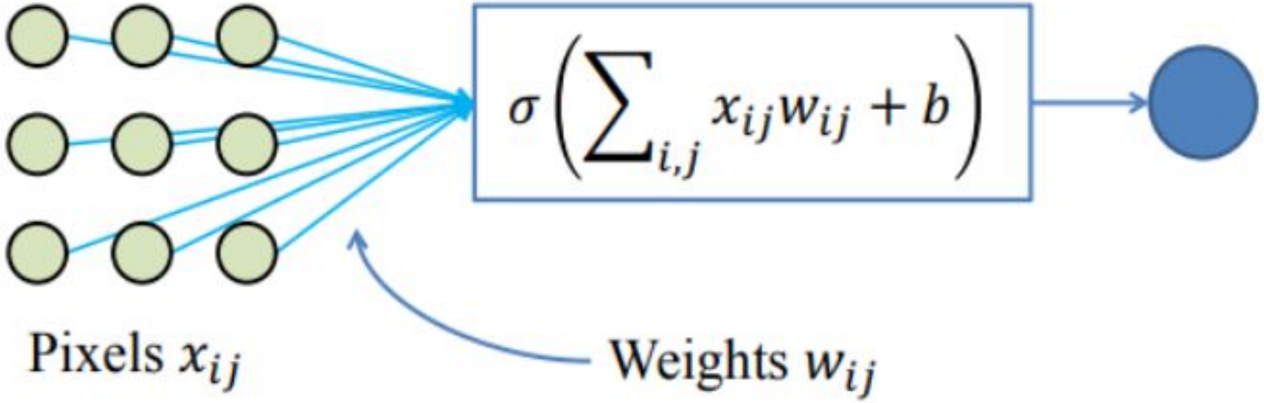
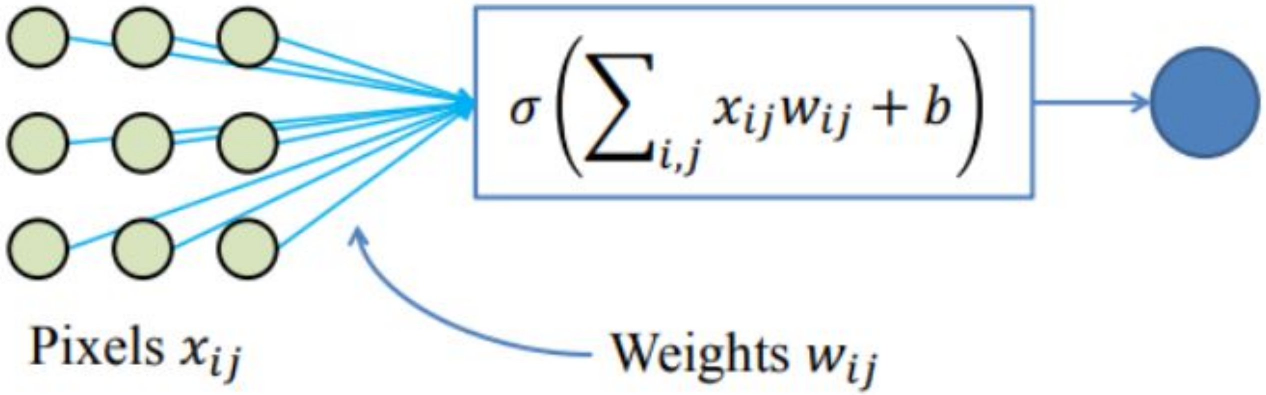


Image as a neural network input

- Normalize input pixels: $x_{norm} = \frac{x}{255} - 0.5$

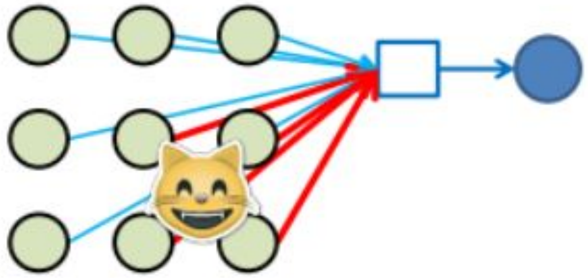
- Maybe MLP will work?



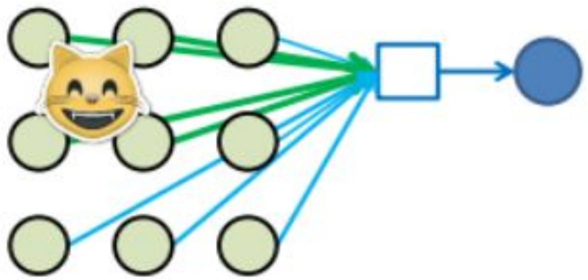
- Actually, no!

Why not MLP?

- Let's say we want to train a "cat detector"



On this training image red weights w_{ij} will change a little bit to better detect a cat



On this training image green weights w_{ij} will change…

- We learn the same "cat features" in different areas and don't fully utilize the training set!
- What if cats in the test set appear in different places?

- **Overfitting** due too many parameters(~millions), while working with medium-large sized images!
- Fail to handle variance in images - translation, rotation, illumination, size etc!

Introduction to Computer Vision

Translation Invariance

Rotation/Viewpoint Invariance

Size Invariance

Illumination Invariance

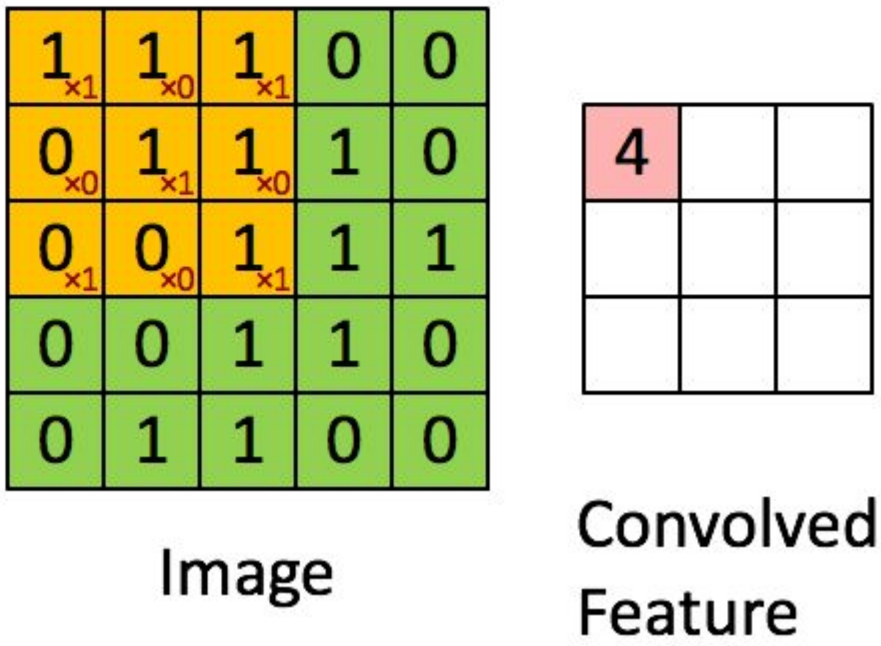
Convolutional Neural Networks

CNN can understand different position/size of the features

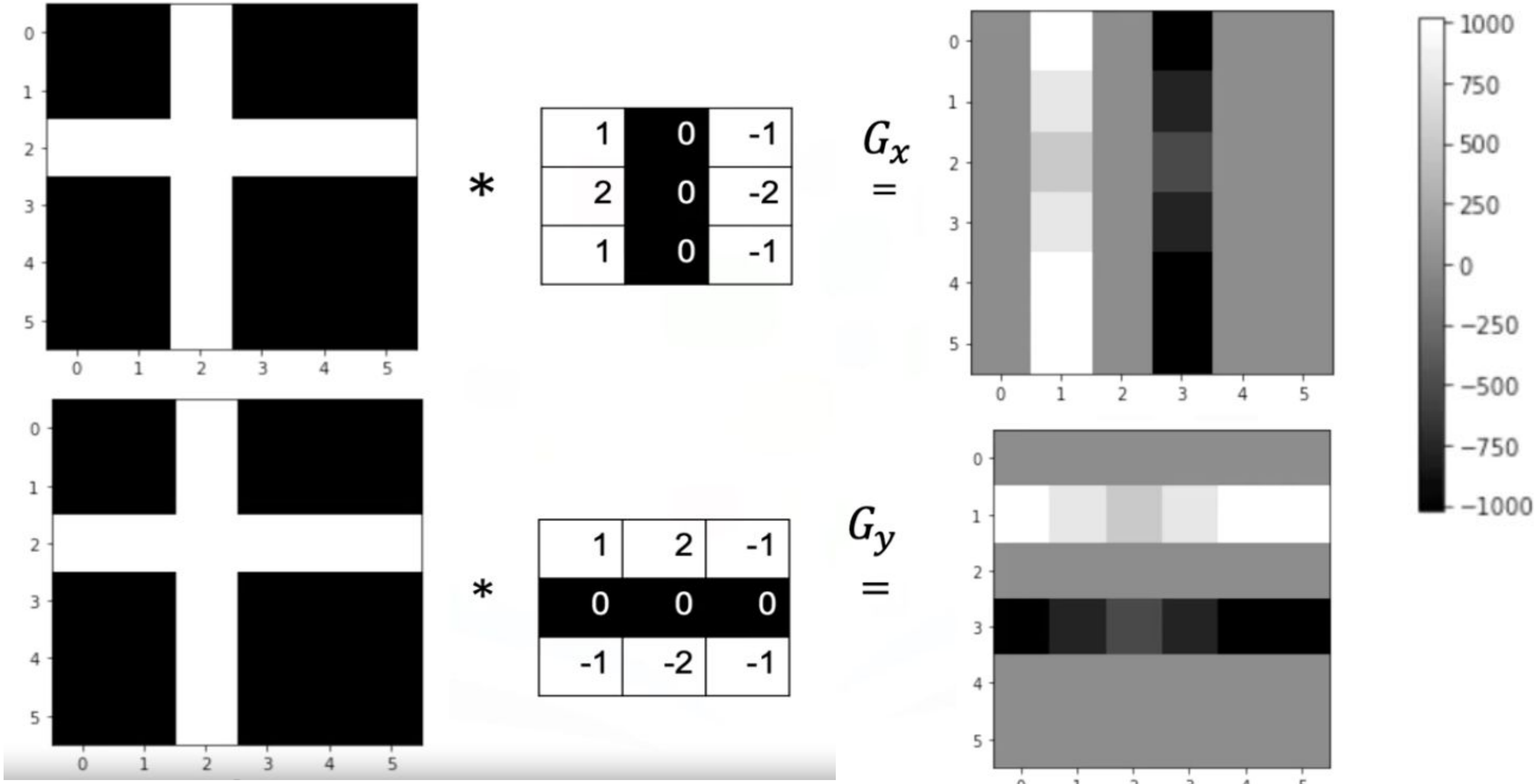




Convolutional Neural Networks



Convolutional Neural Networks



Convolutional Neural Networks

Convolutions have been used for a while

Original image

Kernel

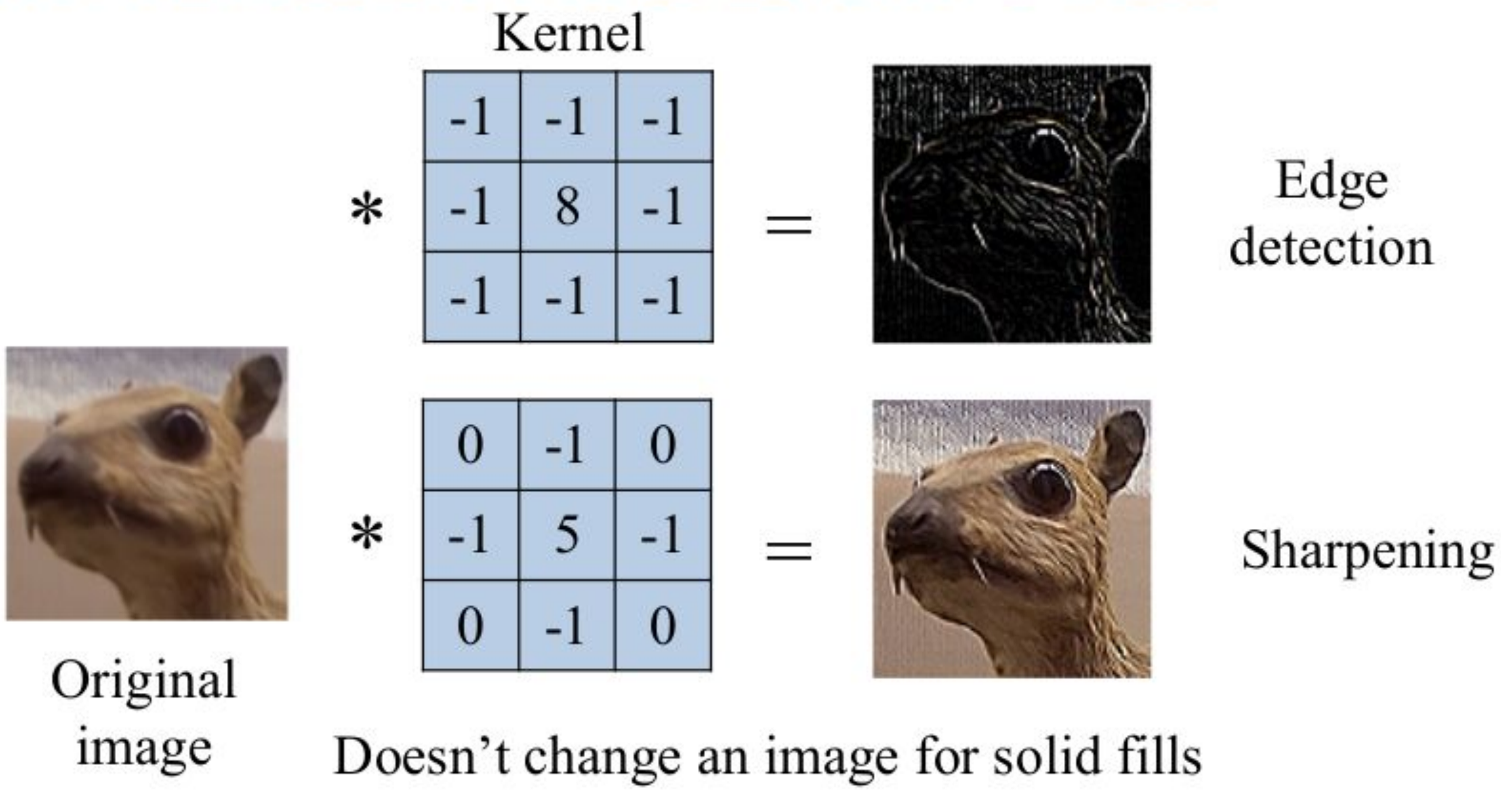
$$* \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix} =$$

Edge detection

Sums up to 0 (black color) when the patch is a solid fill

Convolutional Neural Networks

Convolutions have been used for a while



Doesn’t change an image for solid fills

Adds a little intensity on the edges

Convolutions have been used for a while

Original image

Kernel

$$* \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix} =$$

Edge detection

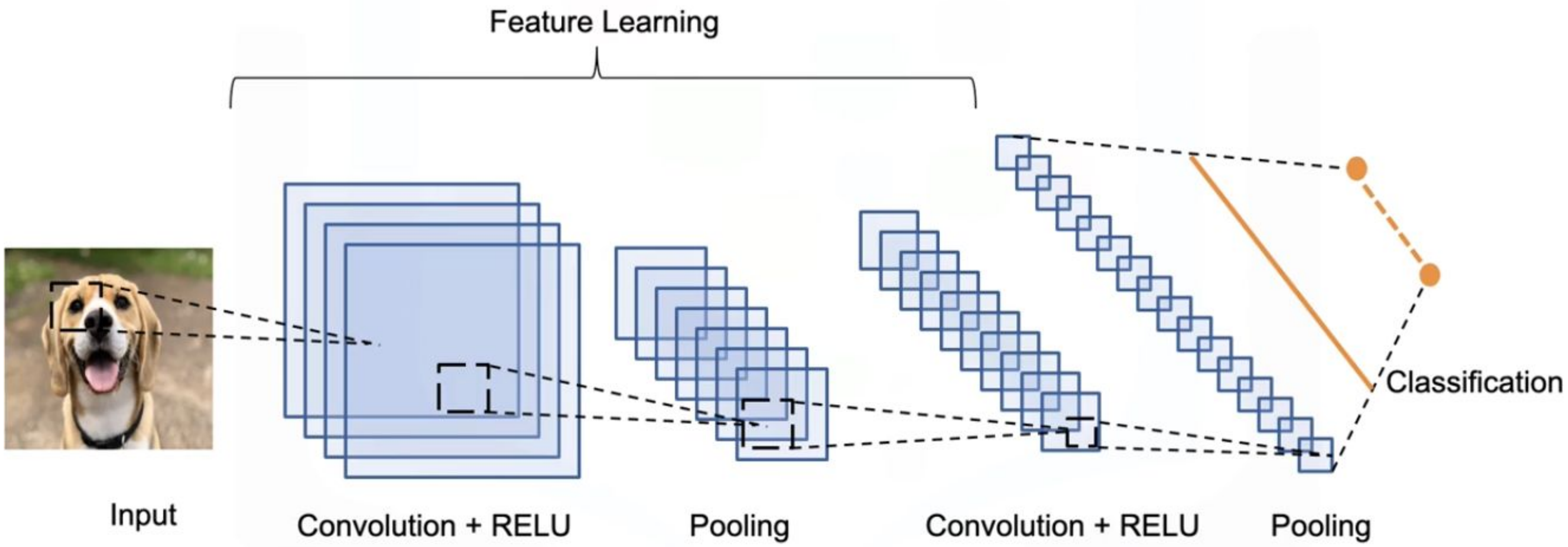
$$* \begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix} =$$

Sharpening

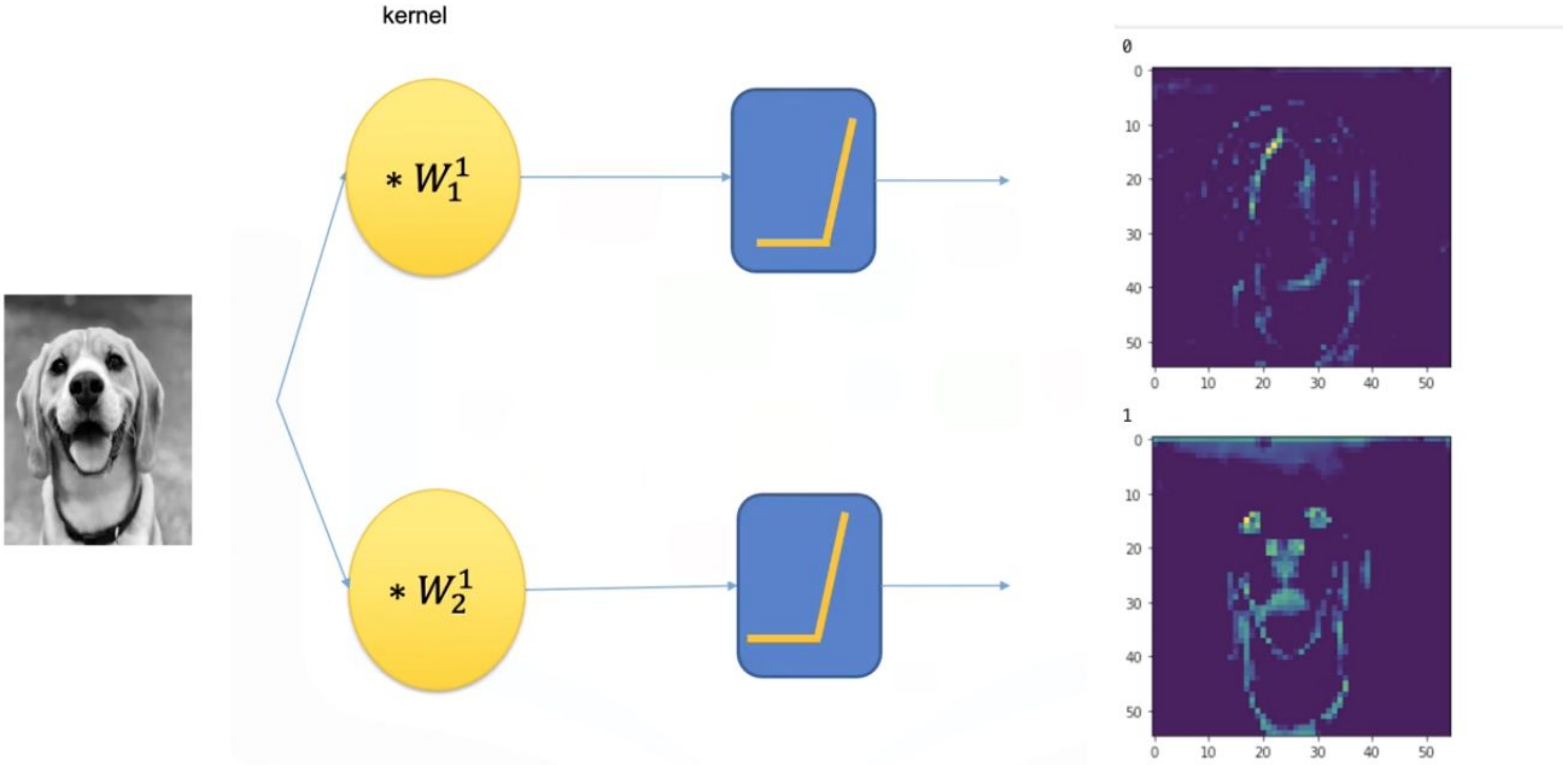
$$* \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} =$$

Blurring

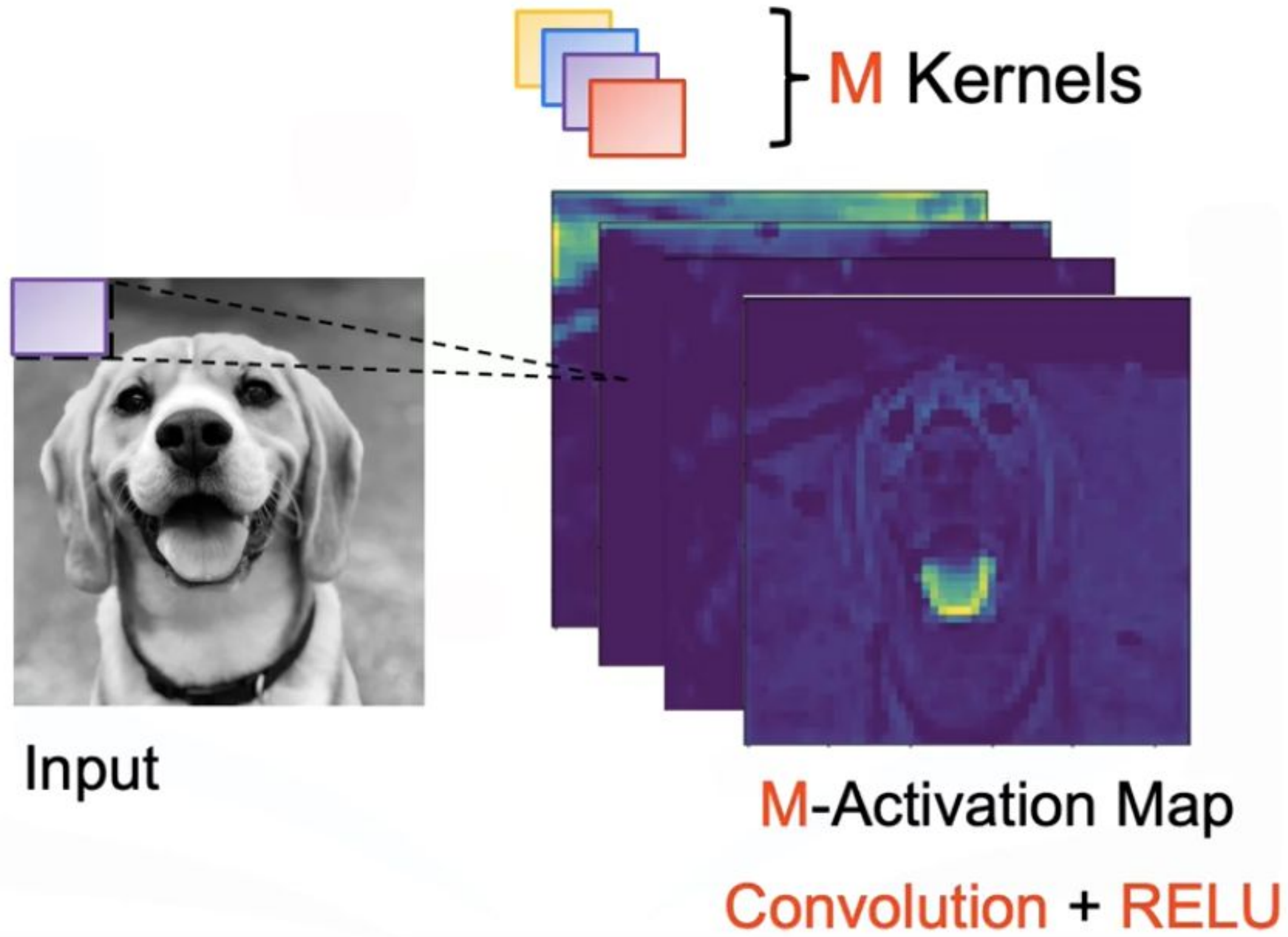
CNN for Image Classification



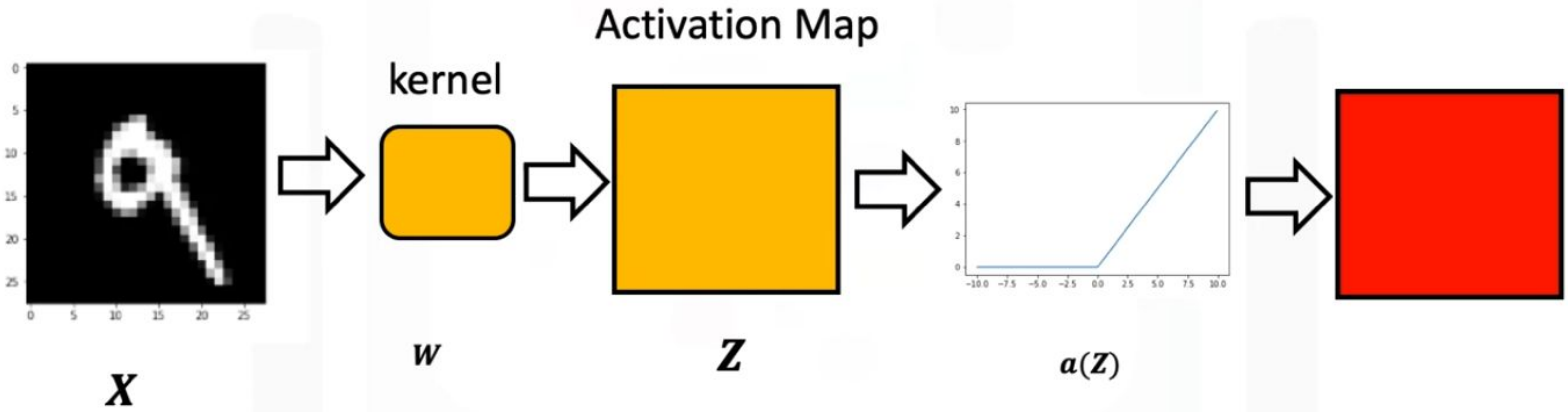
Convolutional Neural Networks



Convolutional Neural Networks



Convolutional Neural Networks

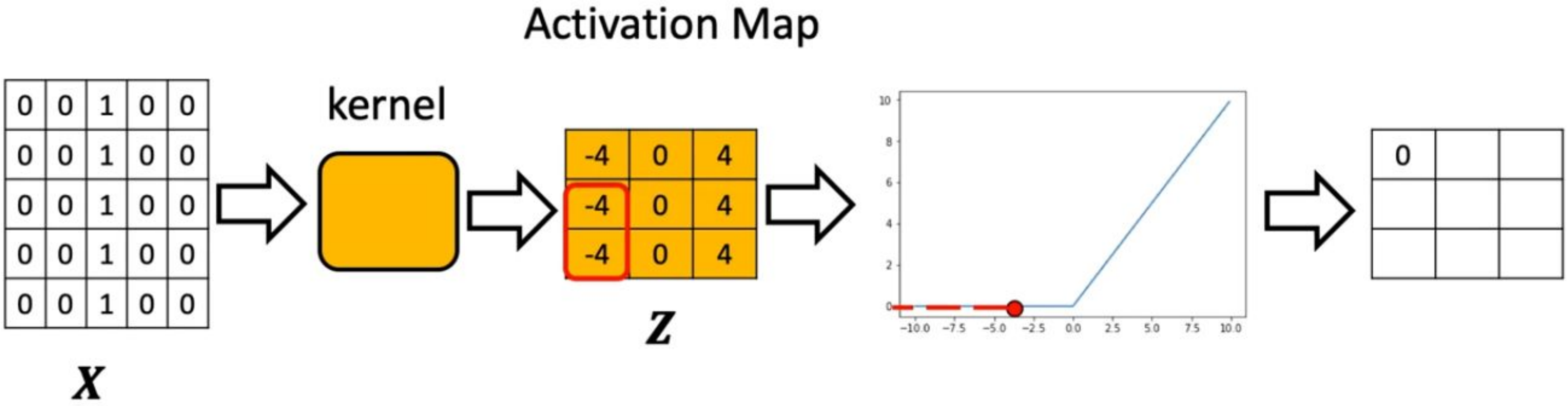


Activation Function

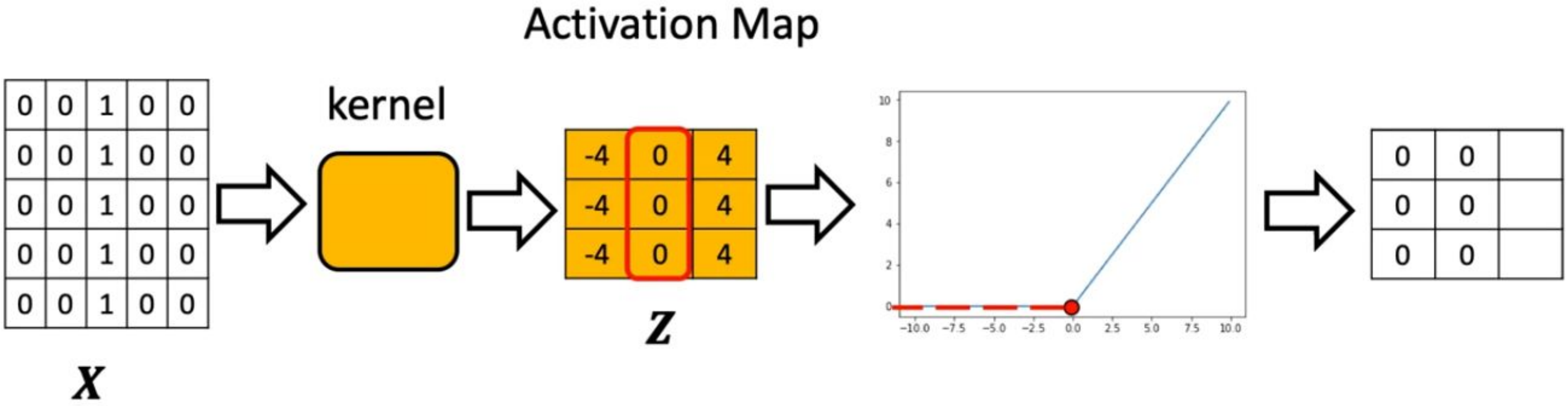
$$\boldsymbol{Z} = \boldsymbol{W} * \mathbf{X} + \mathbf{b}$$

$$\boldsymbol{A} = \boldsymbol{a}(\boldsymbol{Z})$$

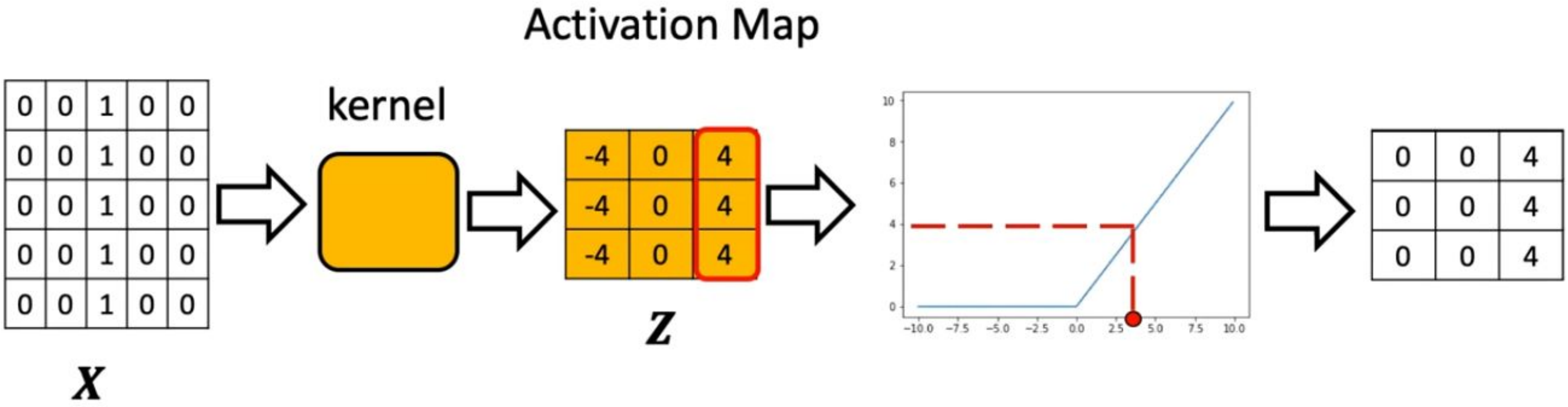
Convolutional Neural Networks



Convolutional Neural Networks



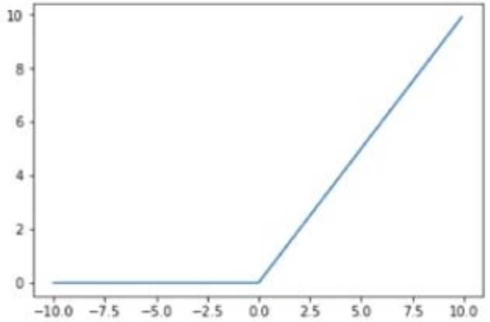
Convolutional Neural Networks



Convolutional Neural Networks

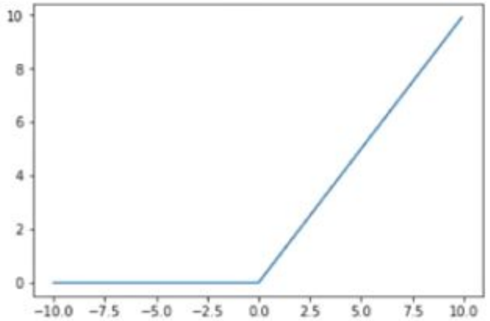
Activation Map 3 Channels

-1	-2	-1
0	0	0
1	2	1



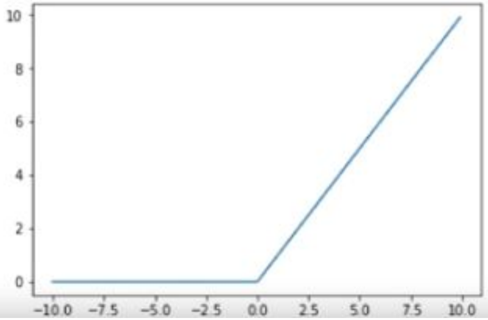
0	0	0
0	0	0
1	2	1

1	1	1
1	0	1
1	1	1



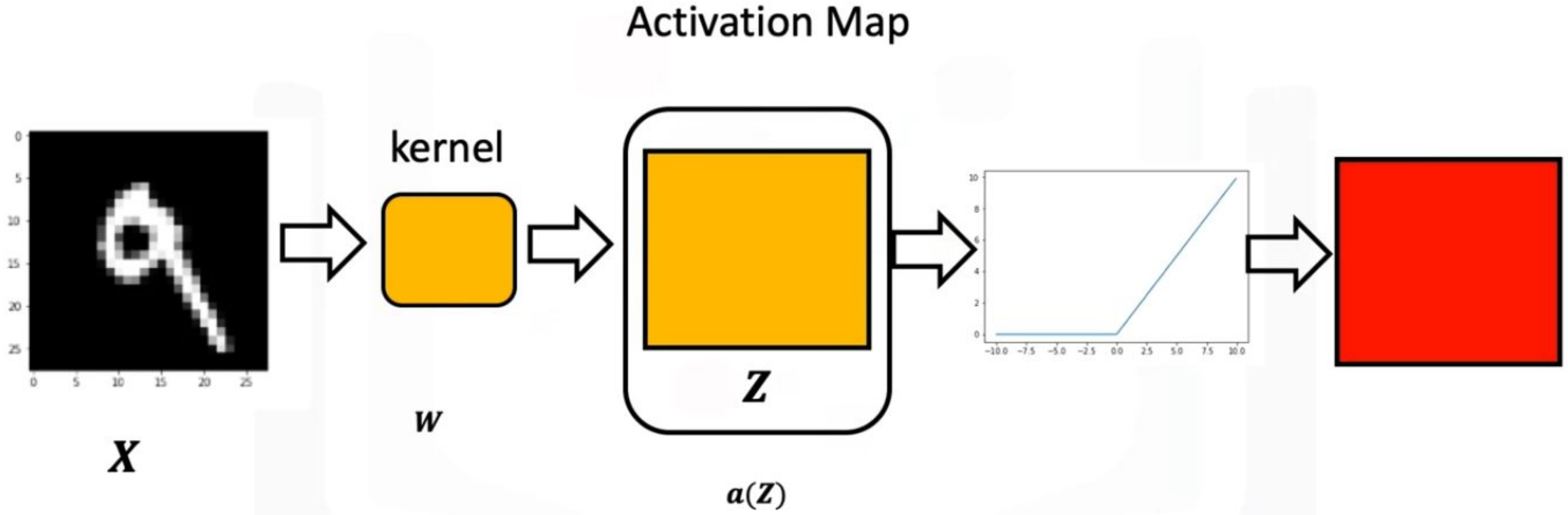
1	1	1
1	0	1
1	1	1

0.5	0.5	0.5
0.5	1	0.5
0.5	0.5	0.5



0.5	0.5	0.5
0.5	1	0.5
0.5	0.5	0.5

Convolutional Neural Networks

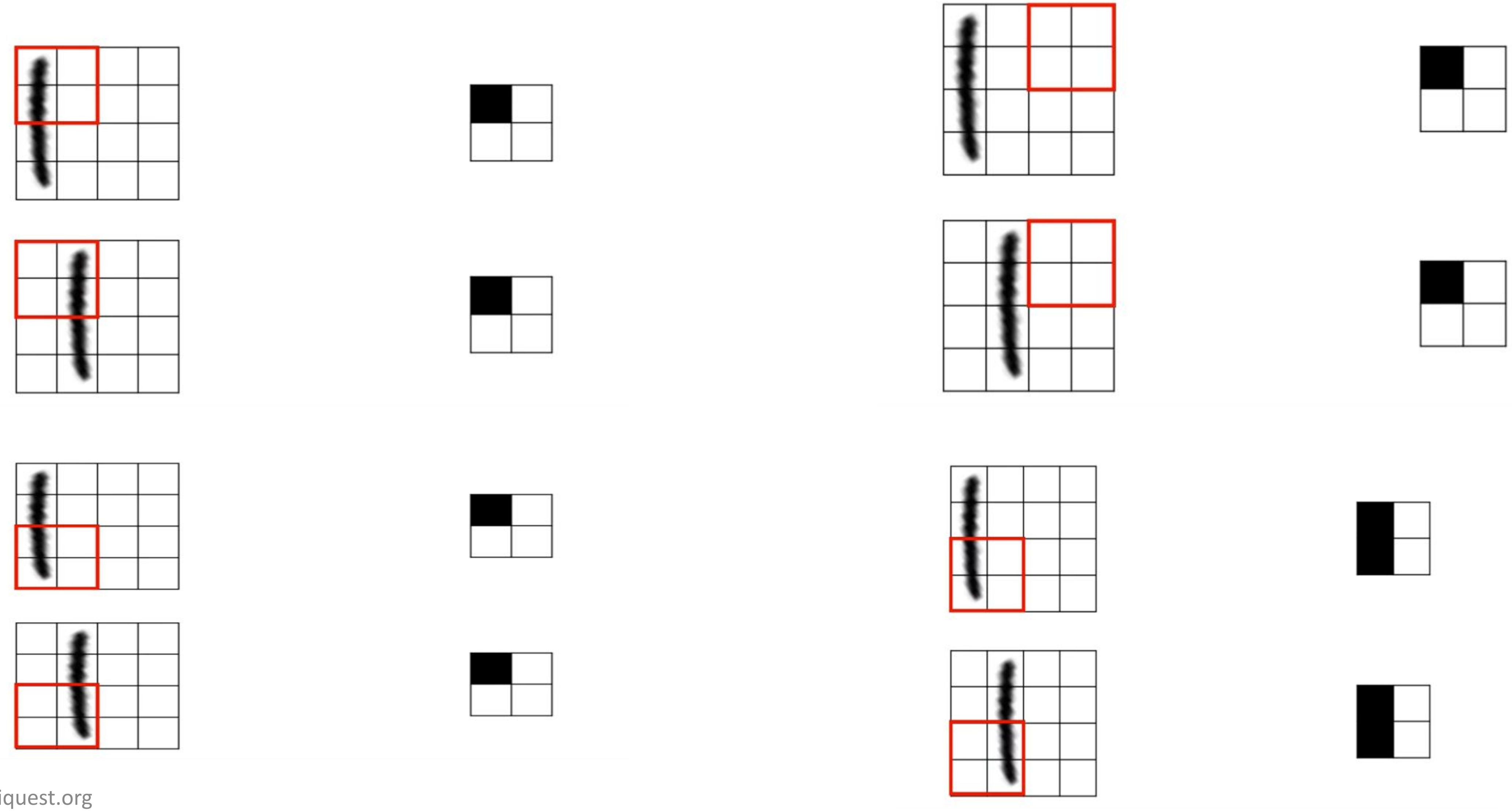


Max Pooling

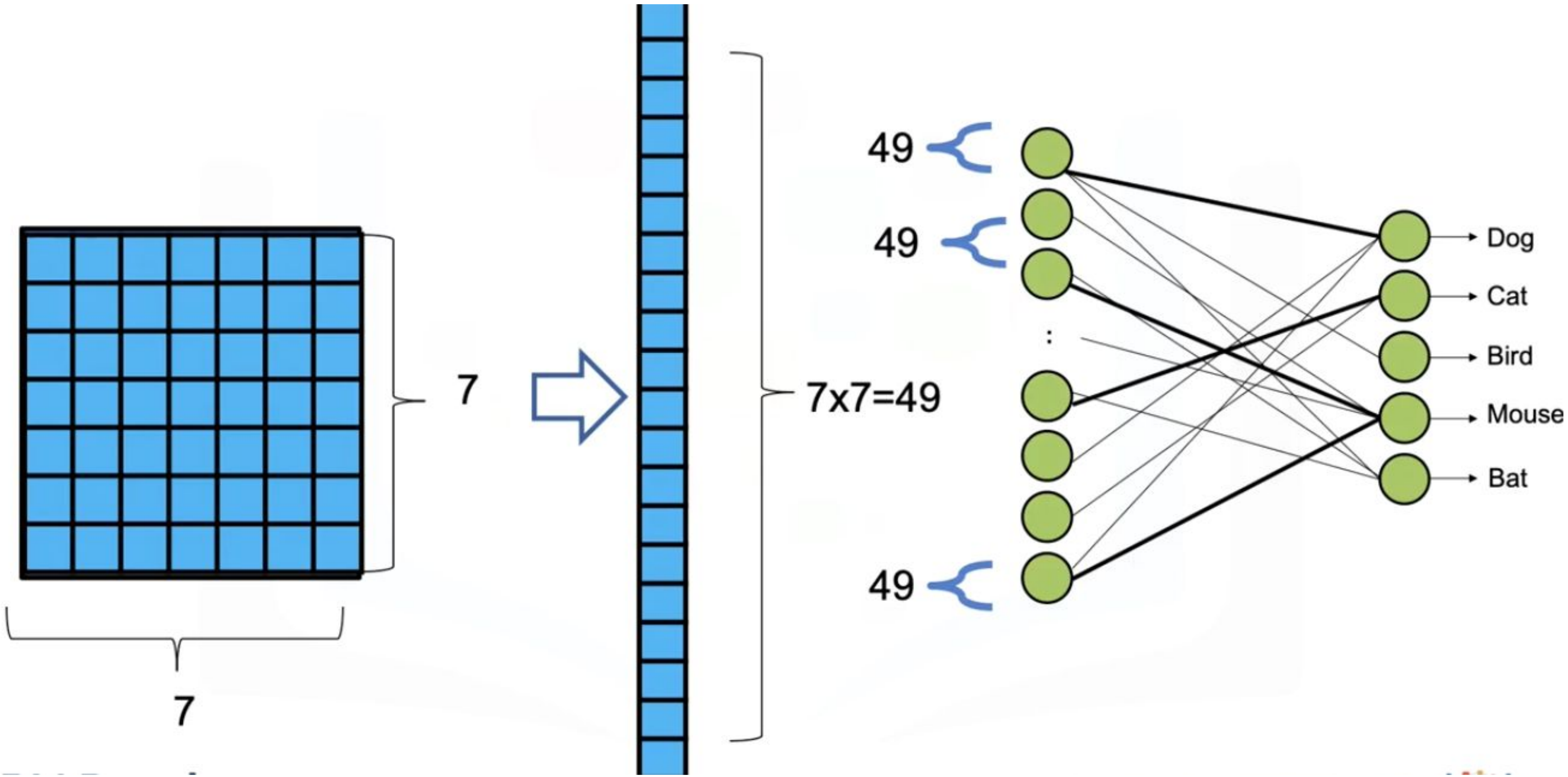
1	2	3	-4
0	2	-3	0
0	2	3	1
0	0	0	0

2	3	3
2	3	3
2	3	3

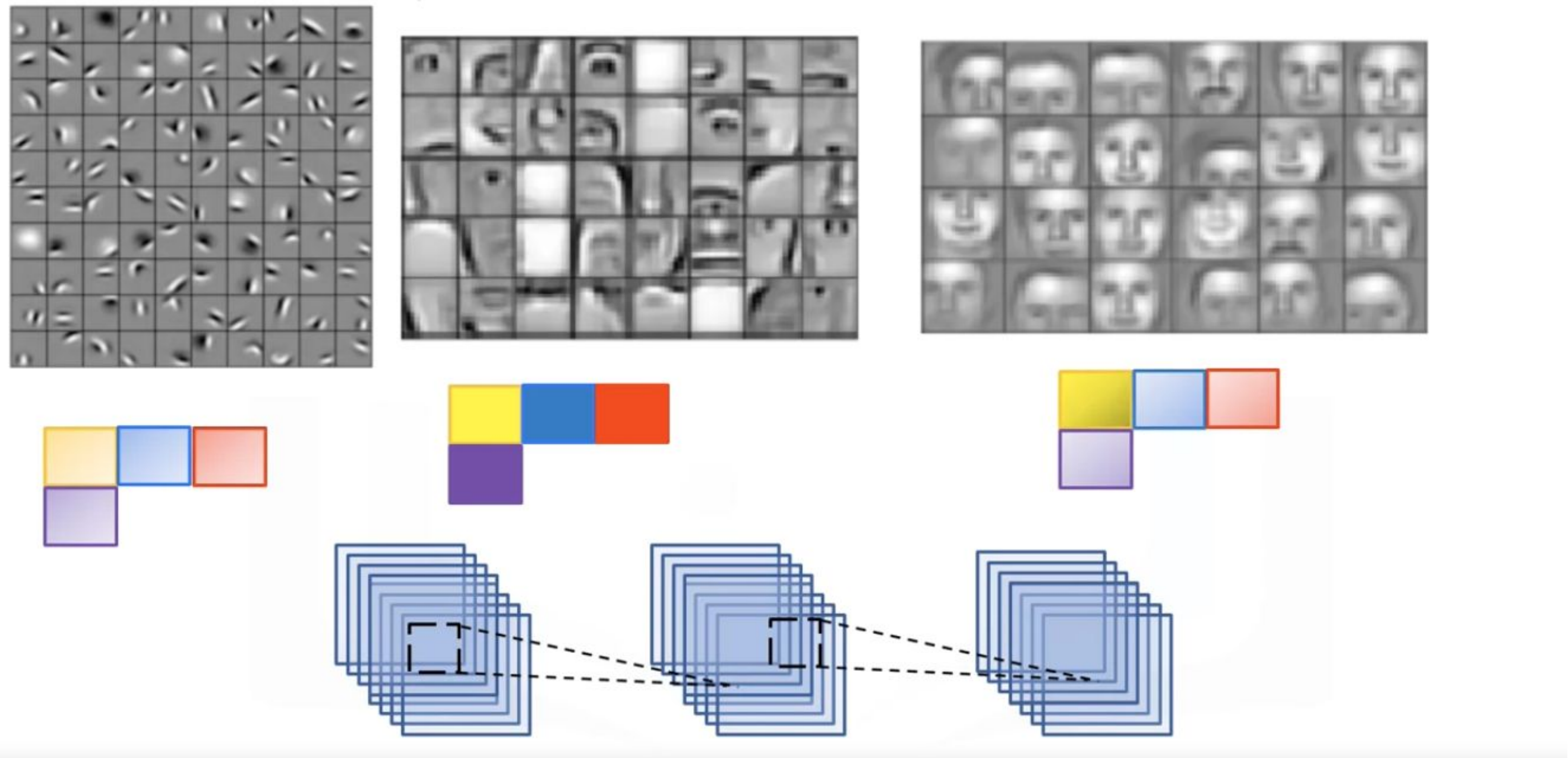
Max Pooling work as Feature Invariance



Convolutional Neural Networks

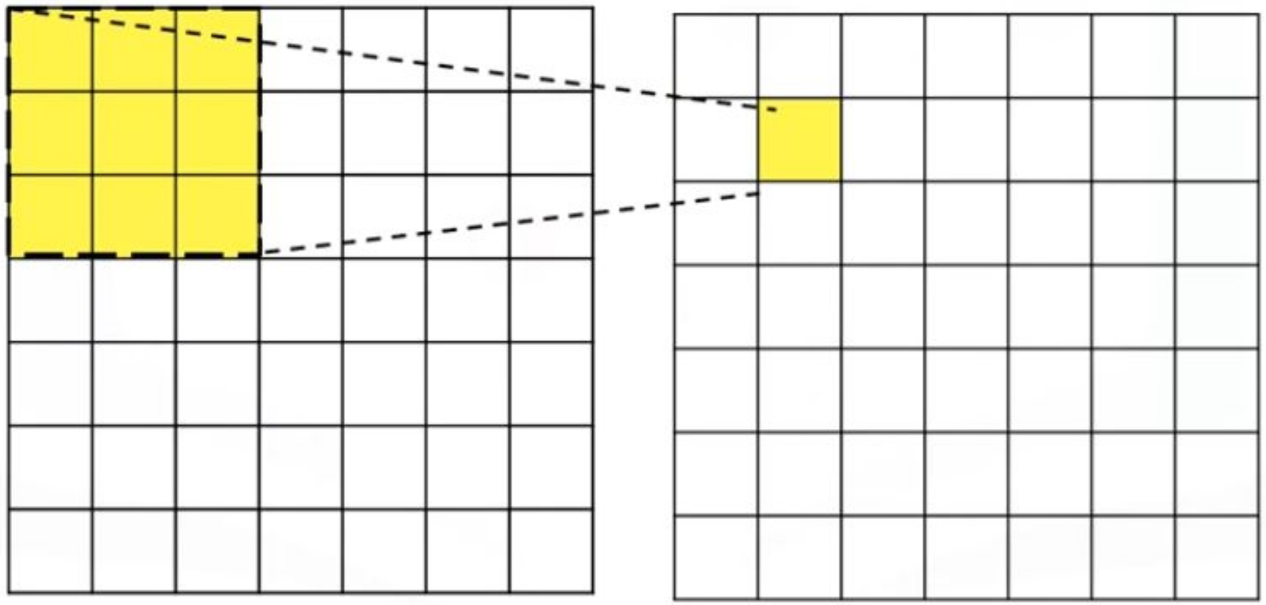


Convolutional Neural Networks

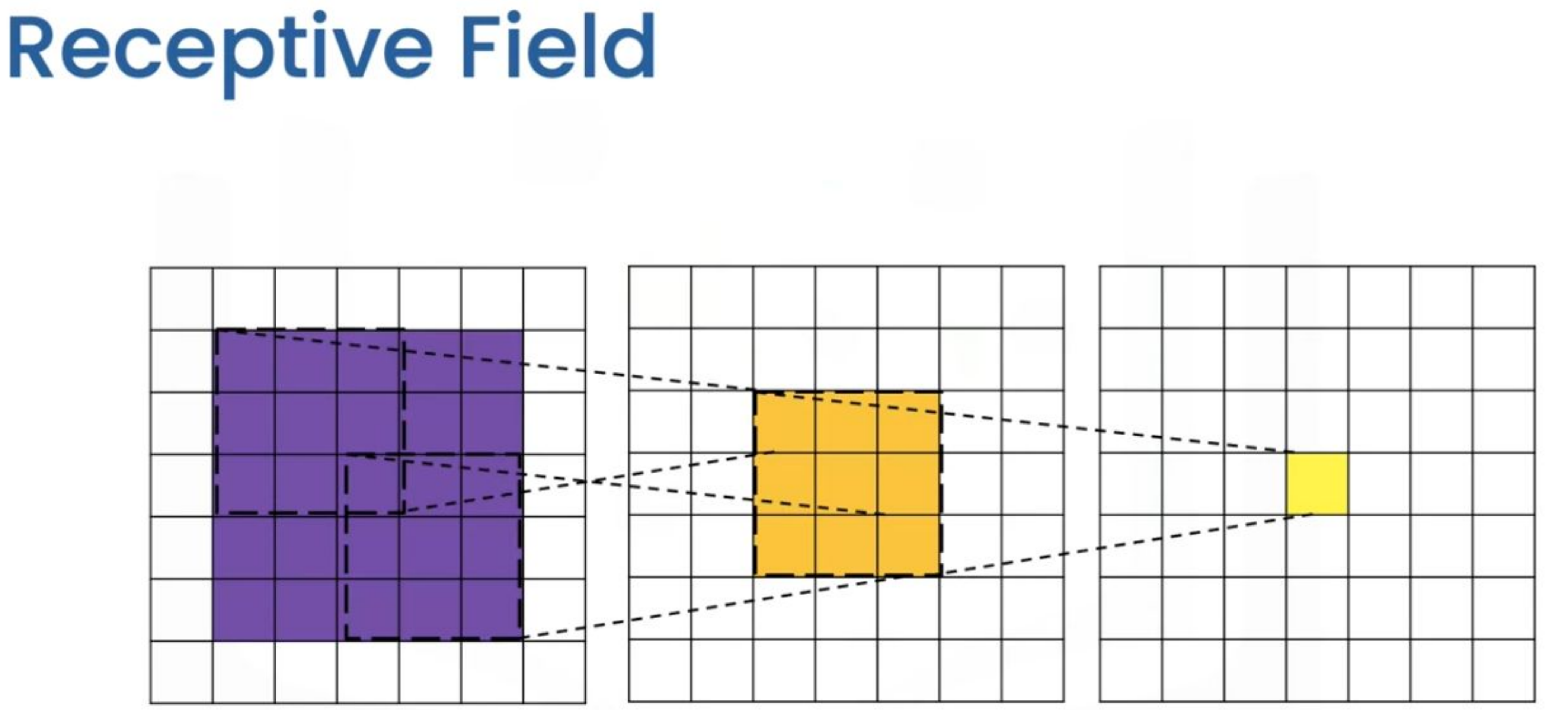


Receptive Field

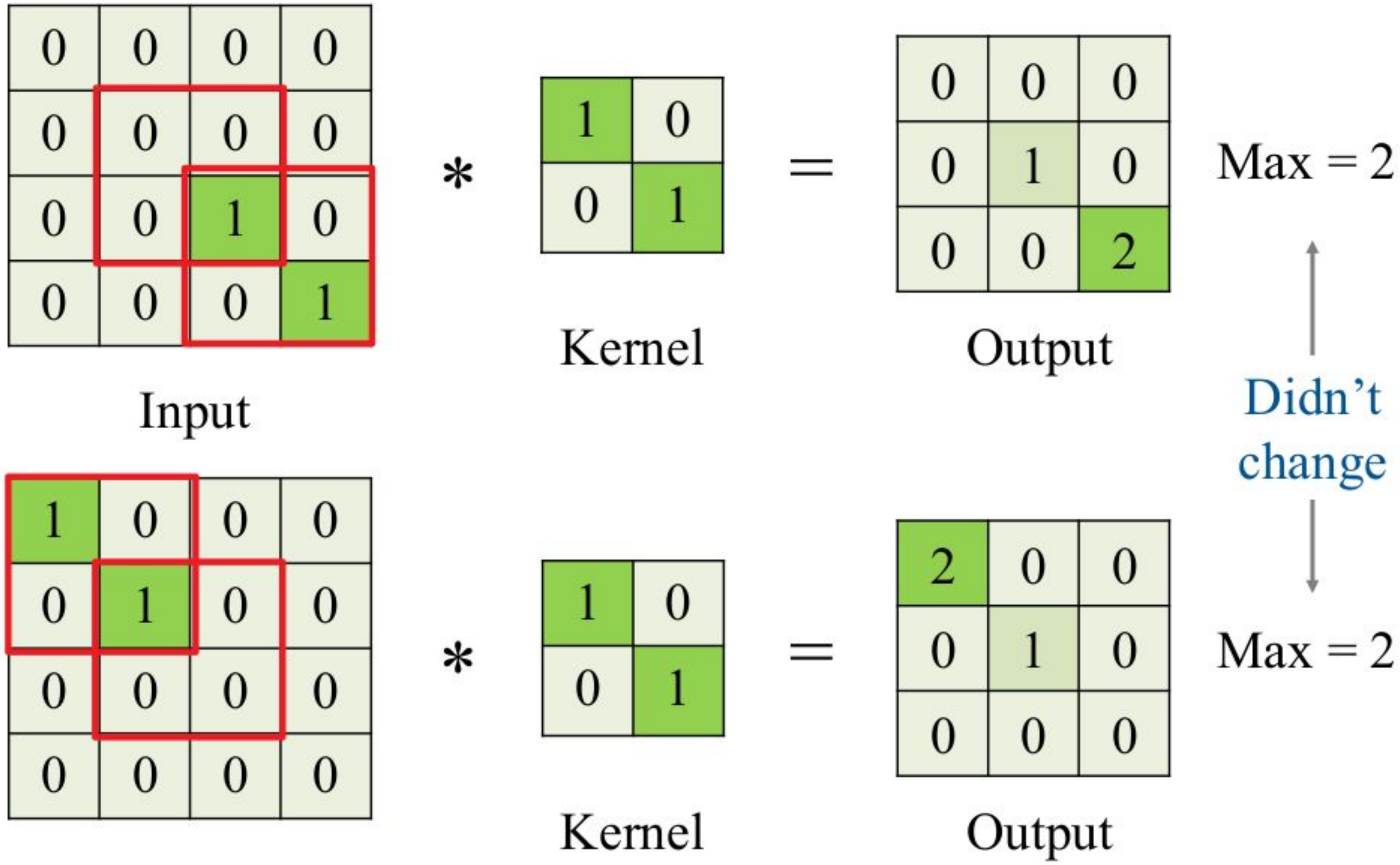
- Receptive Field is the size of the region in the input that produces a pixel value in the activation Map



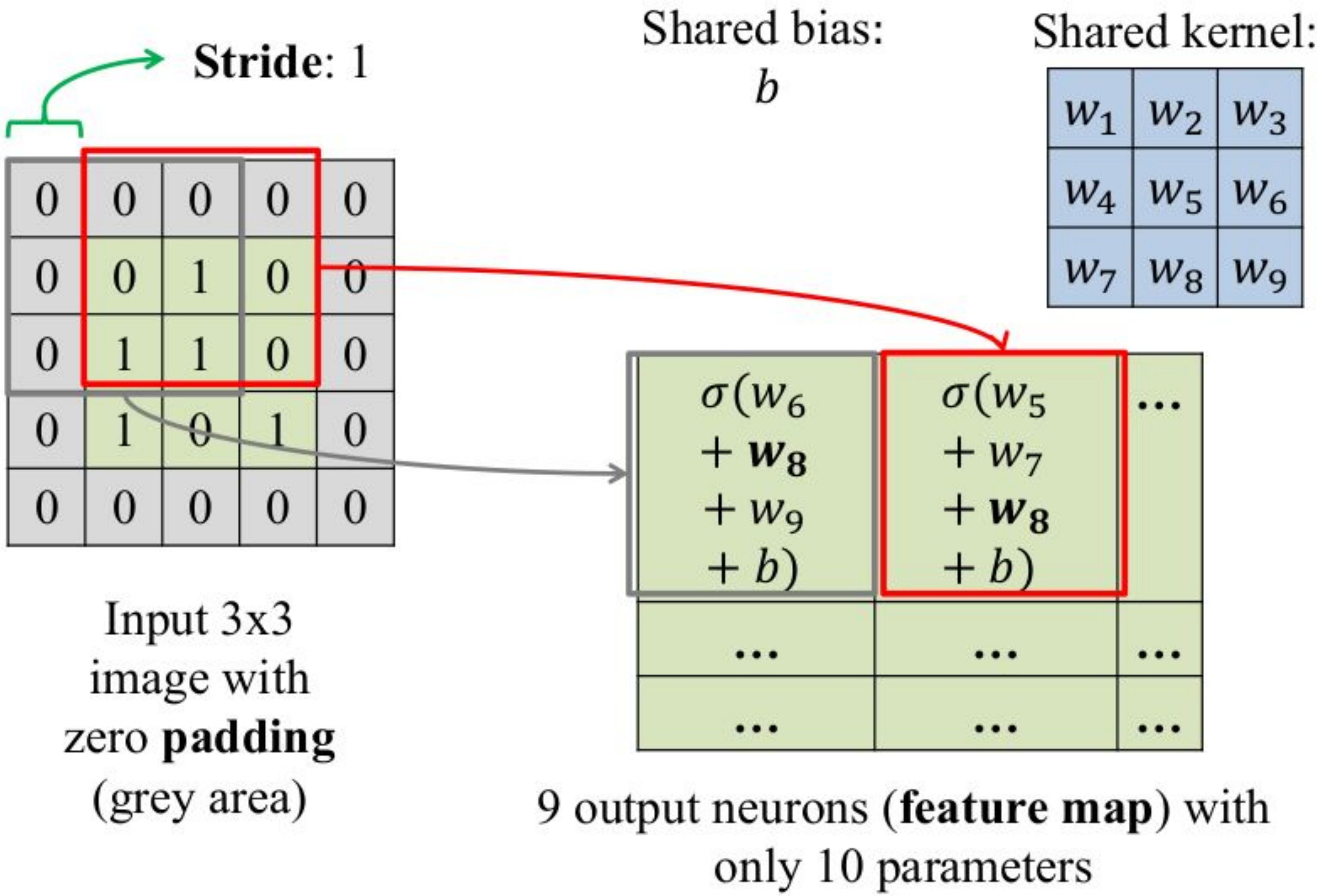
Increasing Layers increase the Receptive field



Convolution is translation equivariant

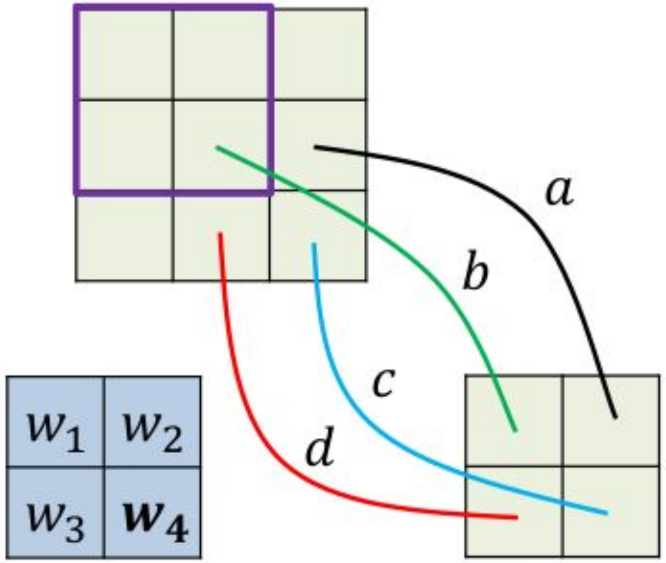


Convolutional layer in neural network



Backpropagation for CNN

Gradients are first calculated as if the kernel weights were not shared:



$$a = a - \gamma \frac{\partial L}{\partial a} \qquad b = b - \gamma \frac{\partial L}{\partial b}$$

$$c = c - \gamma \frac{\partial L}{\partial c} \qquad d = d - \gamma \frac{\partial L}{\partial d}$$

$$w_4 = w_4 - \gamma \left(\frac{\partial L}{\partial a} + \frac{\partial L}{\partial b} + \frac{\partial L}{\partial c} + \frac{\partial L}{\partial d} \right)$$

Gradients of the same shared weight are summed up!