FPT Education

FPT SCHOOL OF BUSINESS & TECHNOLOGY

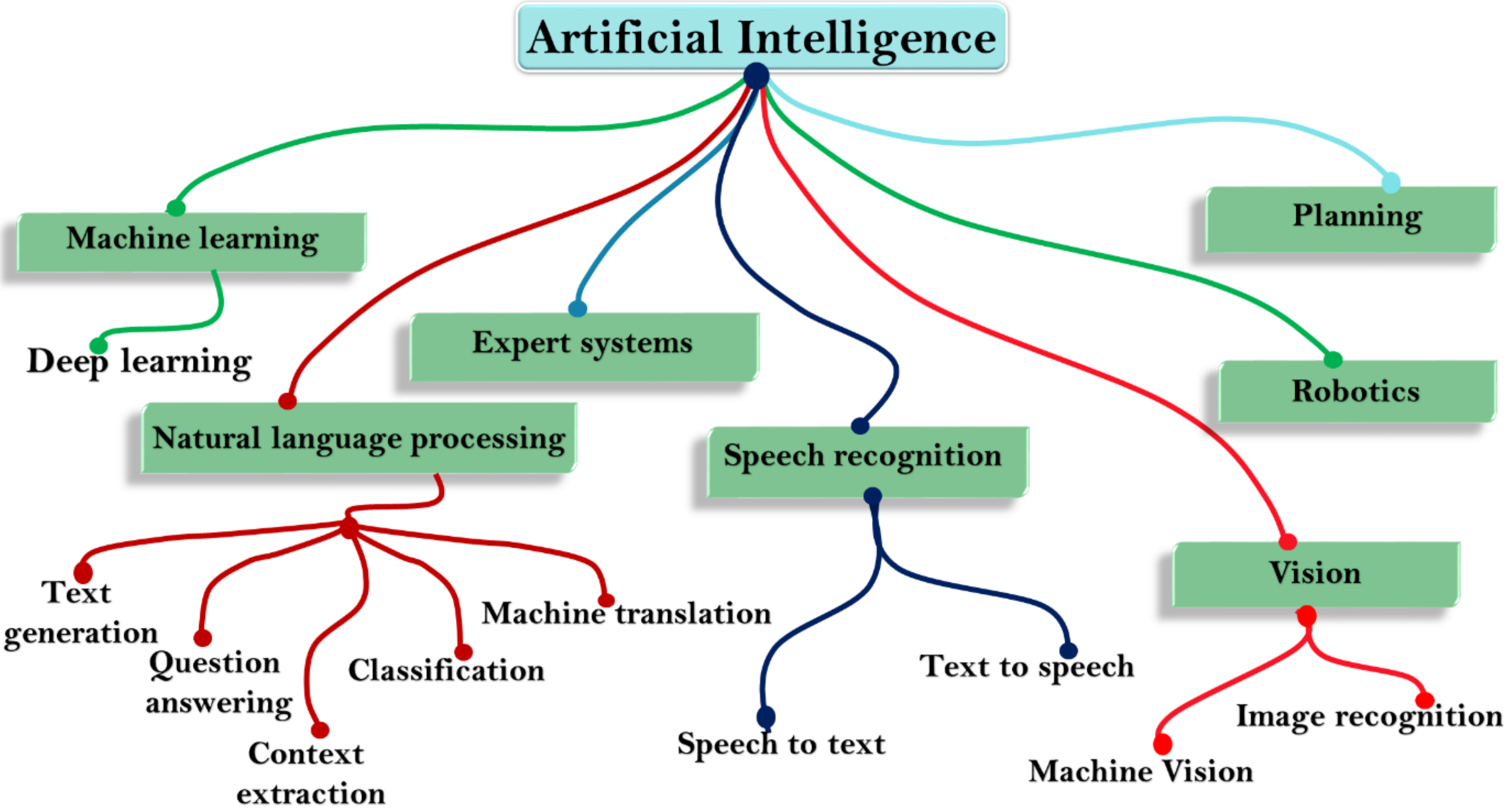
Digital Signal Processing

Giới thiệu về mạng CNN

Phd. Trần Thanh Trúc

RECAP

RECAP - AI



Artificial Intelligence
Machine learning
Deep learning
Expert systems
Natural language processing
Speech recognition
Planning
Robotics
Vision
Text generation
Question answering
Context extraction
Classification
Machine translation
Speech to text
Text to speech
Image recognition
Machine Vision

RECAP: AI

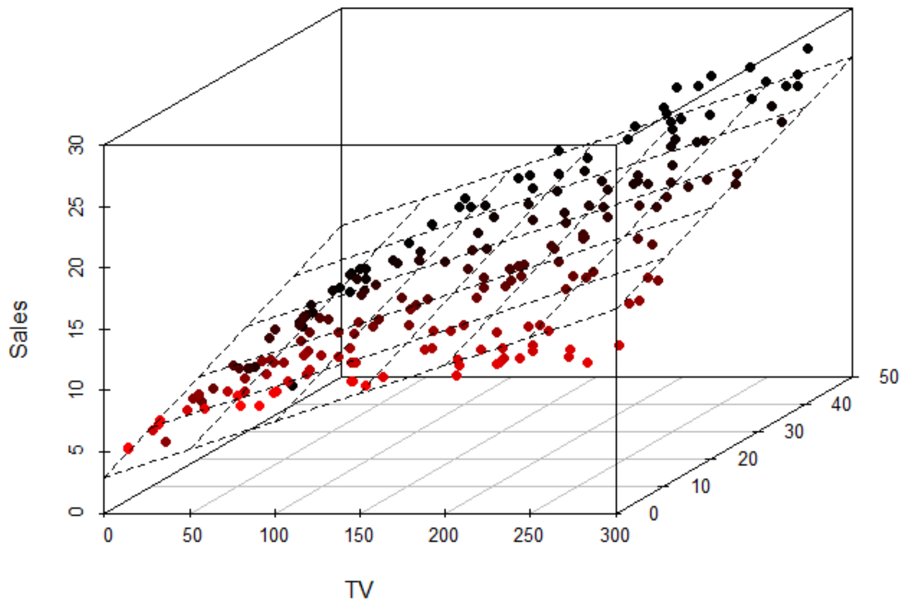
AI approaching includes 3 parts:

1. Machine Learning
2. Expert System
3. Evolution Algorithm

RECAP: LR

$$\mathbf{y} = X\boldsymbol{\beta} + \boldsymbol{\varepsilon}$$

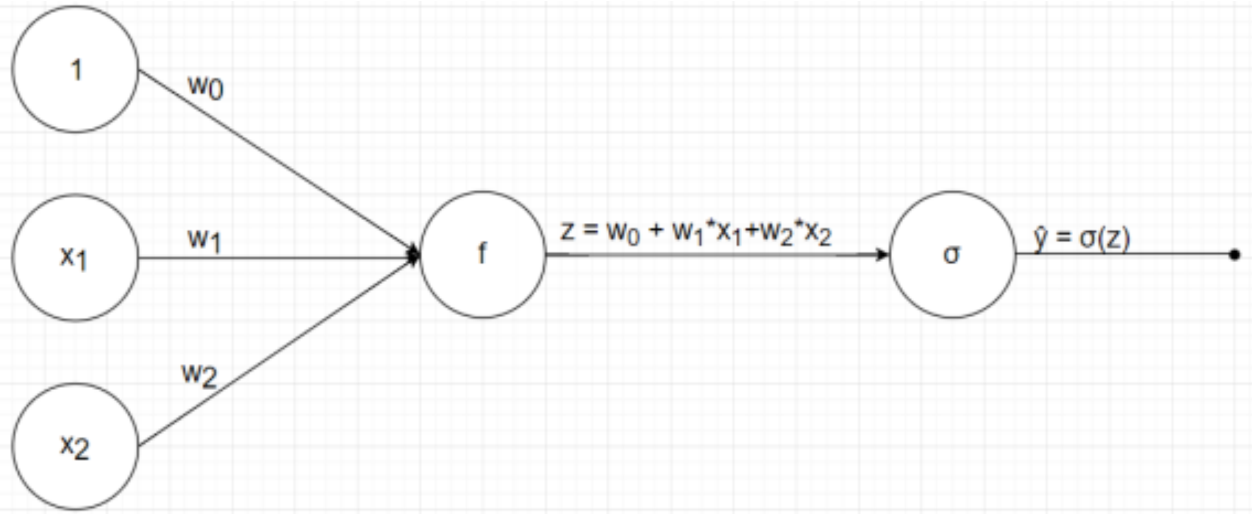
$$\mathbf{y} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}, \quad X = \begin{pmatrix} \mathbf{x}_1^\top \\ \mathbf{x}_2^\top \\ \vdots \\ \mathbf{x}_n^\top \end{pmatrix} = \begin{pmatrix} 1 & x_{11} & \cdots & x_{1p} \\ 1 & x_{21} & \cdots & x_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n1} & \cdots & x_{np} \end{pmatrix}$$



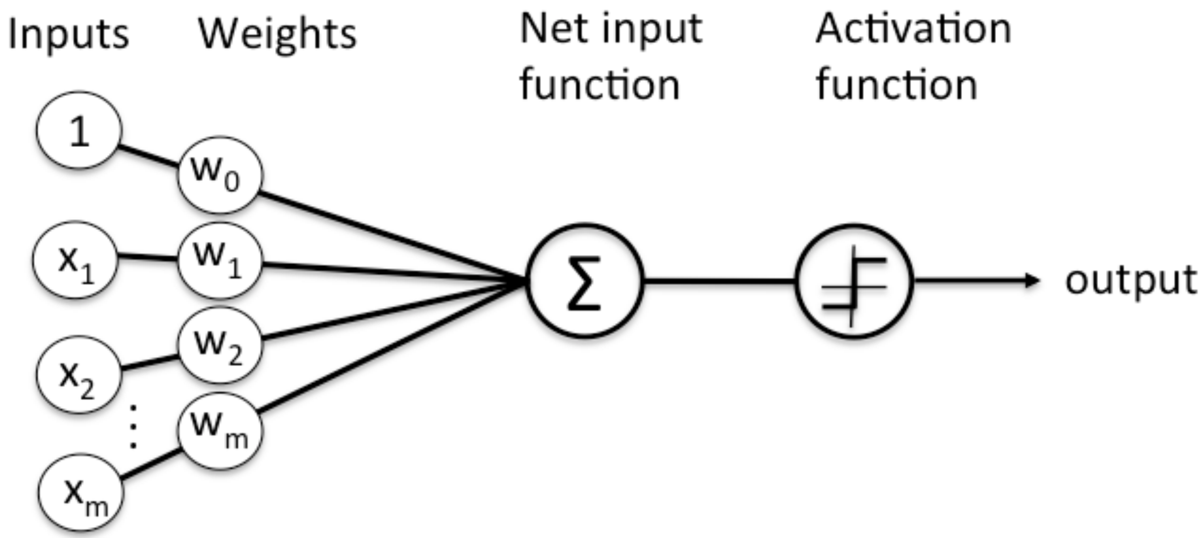
$$\boldsymbol{\beta} = \begin{pmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \vdots \\ \beta_p \end{pmatrix}, \quad \boldsymbol{\varepsilon} = \begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{pmatrix}$$

$$y_i = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip} + \varepsilon_i = \mathbf{x}_i^\top \boldsymbol{\beta} + \varepsilon_i, \qquad i = 1, \ldots, n,$$

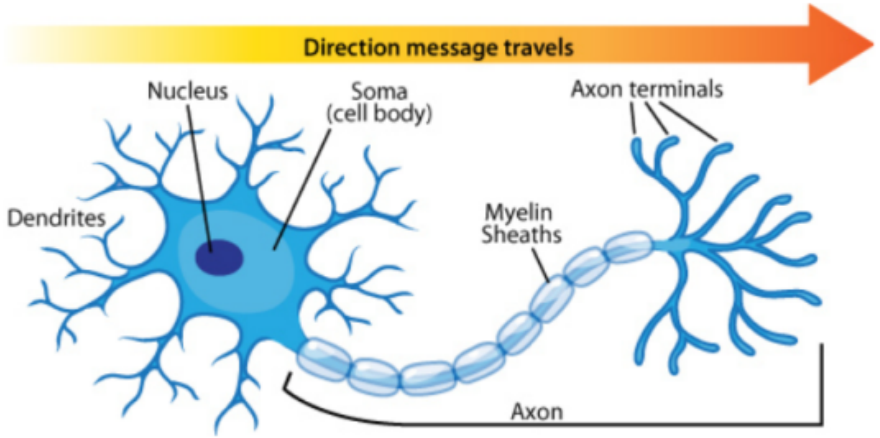
RECAP : ANN



1
w0
x1
w1
x2
w2
f
z = w0 + w1*x1+w2*x2
σ
ŷ = σ(z)

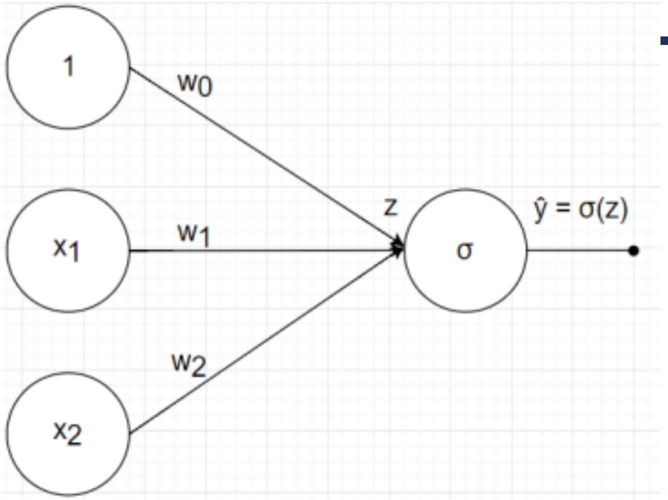


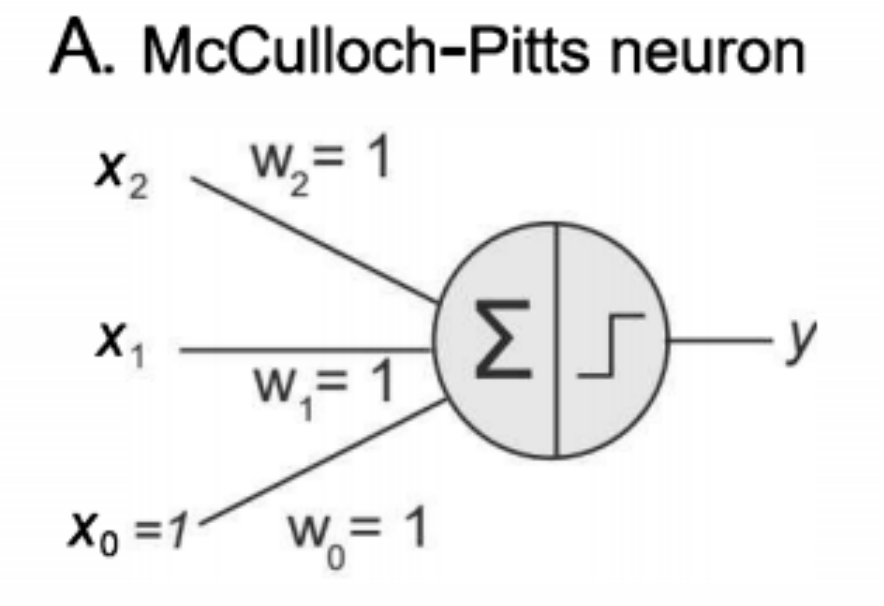
Inputs
Weights
Net input function
Activation function
1
x1
x2
xm
w0
w1
w2
wm
Σ
output

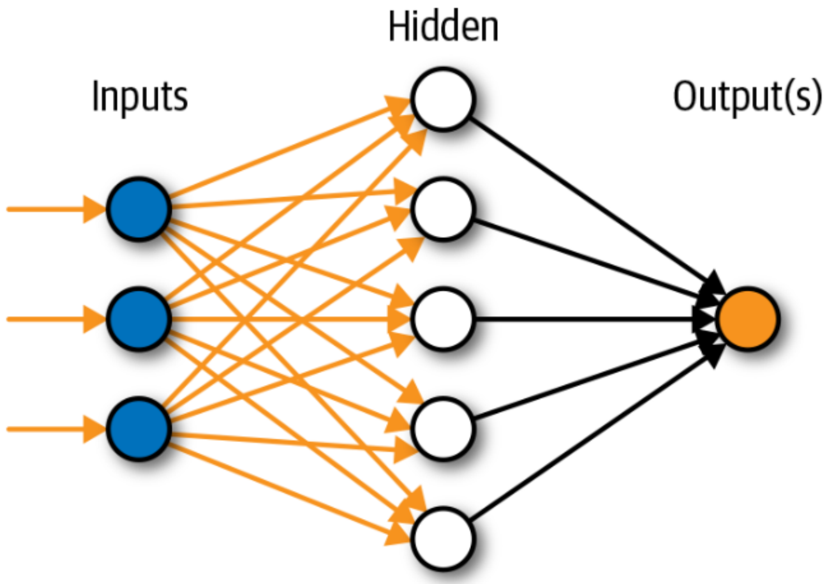


Direction message travels
Nucleus
Soma (cell body)
Axon terminals
Dendrites
Myelin Sheaths
Axon

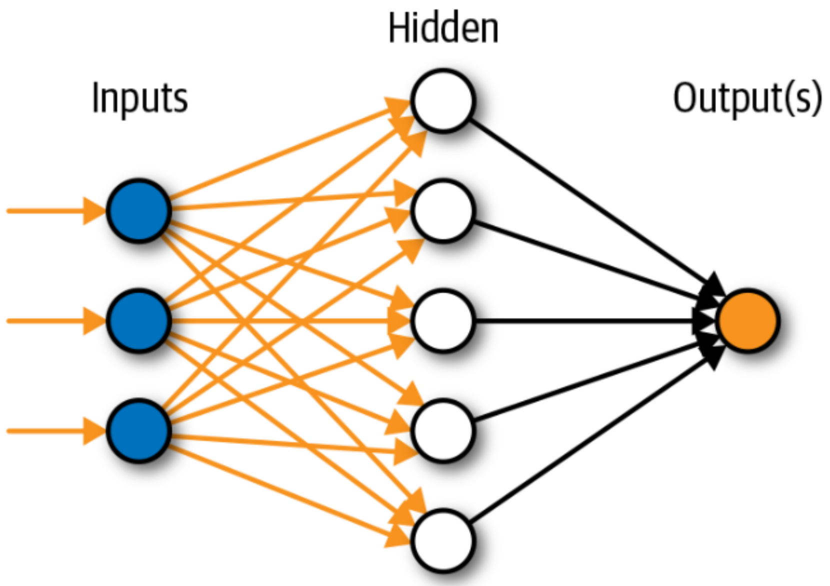
RECAP : ANN


1
w0
x1
w1
x2
w2
z
σ
ŷ = σ(z)

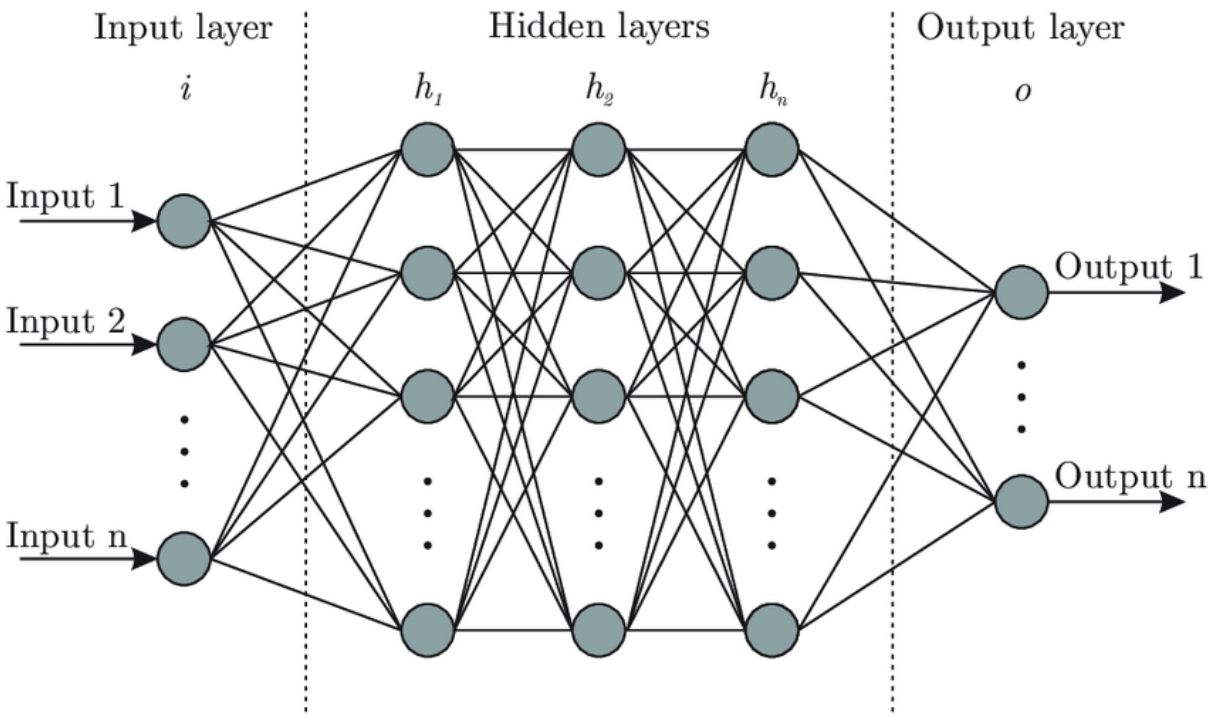

A. McCulloch-Pitts neuron
x2
w2= 1
x1
w1= 1
x0 =1
w0= 1
Σ
y


Inputs
Hidden
Output(s)

RECAP : ANN

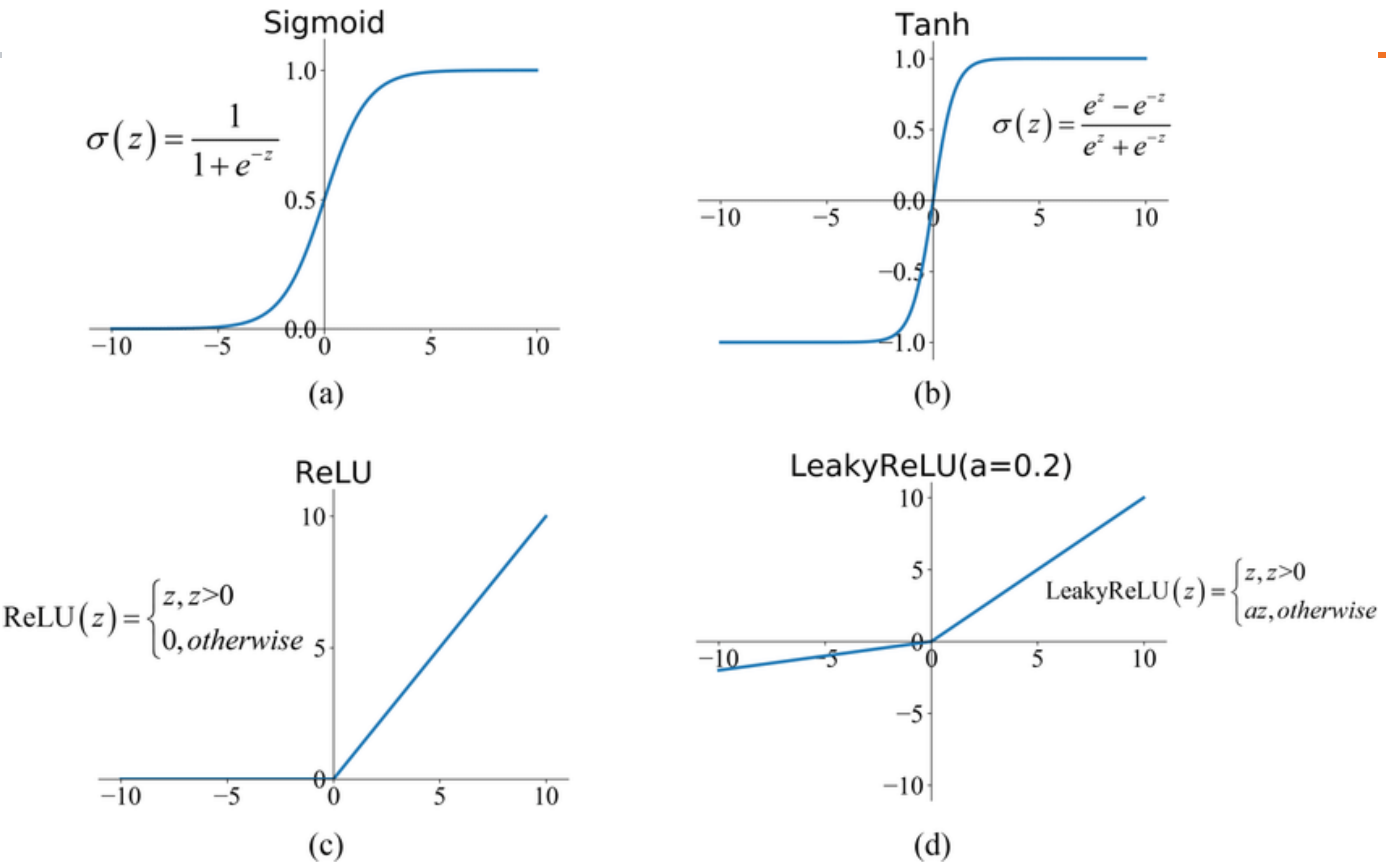


Single hidden layer ANN



Multiple hidden layers ANN

RECAP: ACTIVATION FUNCTION



RECAP: BACKPROPAGATION

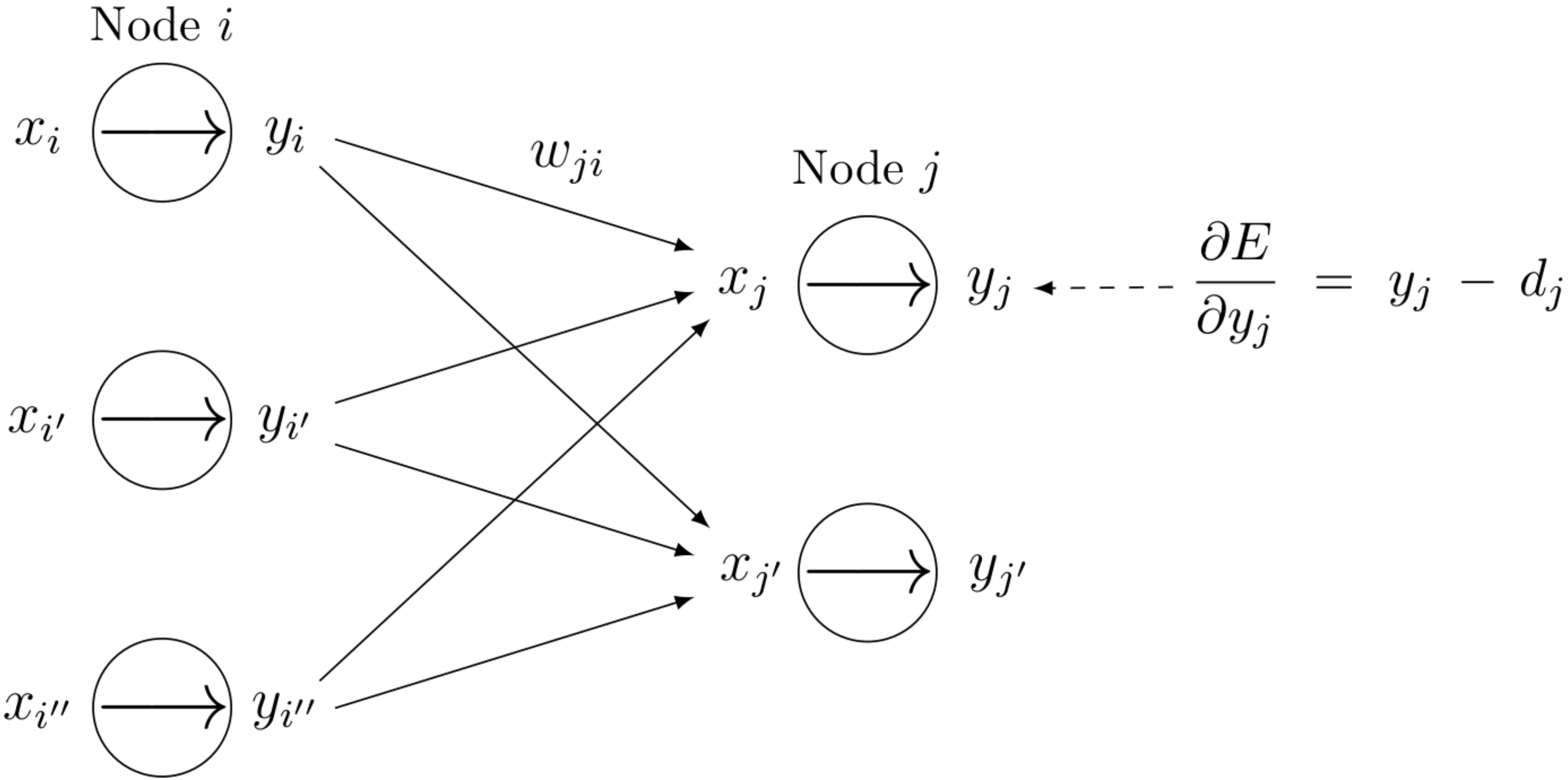
Forward pass: Compute outputs

$$a^{(0)} \Rightarrow z^{(1)} \Rightarrow a^{(1)} \Rightarrow z^{(2)} \Rightarrow a^{(2)} \Rightarrow z^{(3)} \Rightarrow a^{(3)} = \hat{y}$$

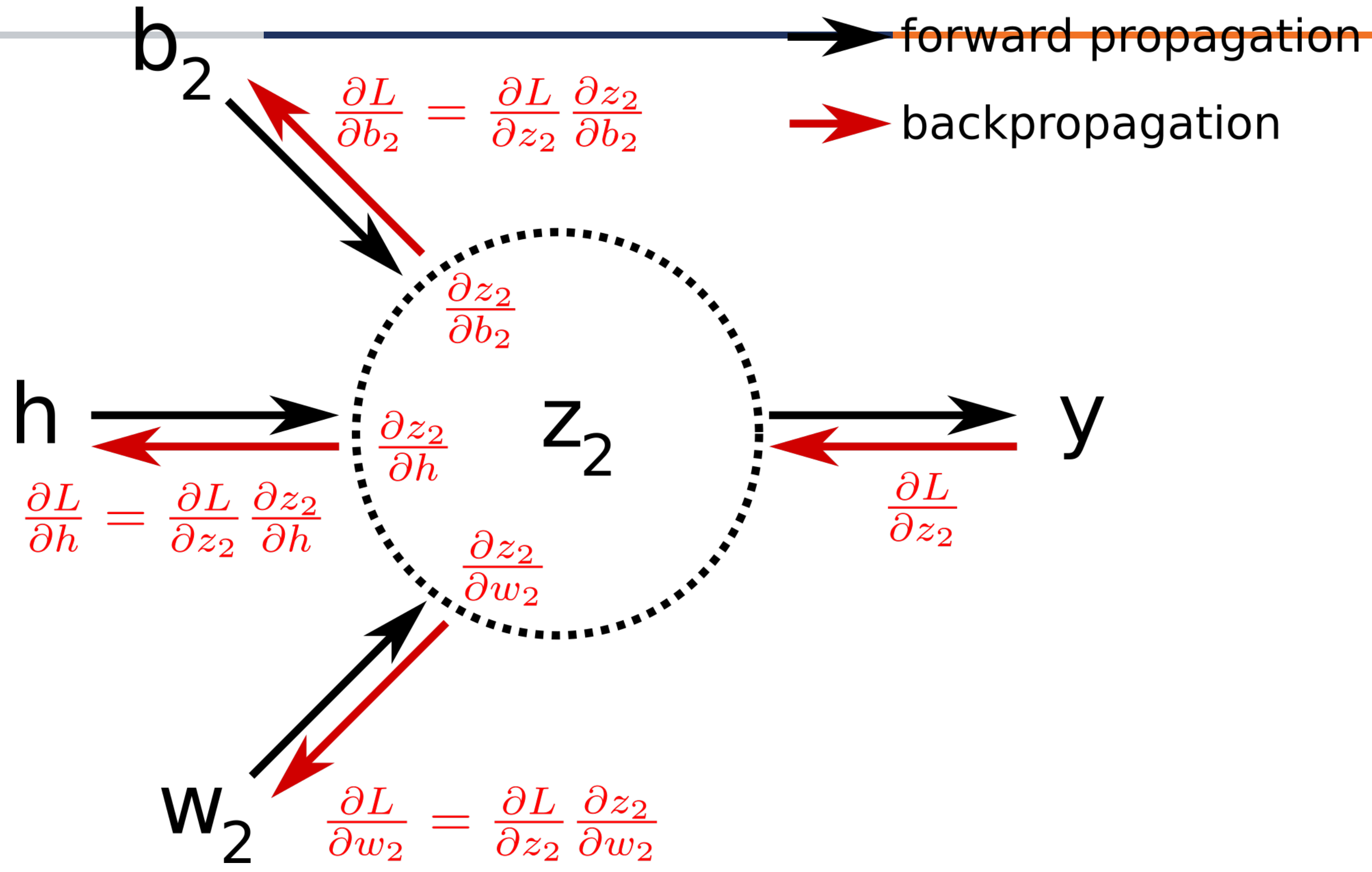
Backward pass: Compute gradients

$$\overset{\dfrac{\partial J}{\partial w^{(1)}} \quad \dfrac{\partial J}{\partial b^{(1)}}}{\Longleftarrow} \frac{\partial J}{\partial a^{(1)}} \overset{\dfrac{\partial J}{\partial w^{(2)}} \quad \dfrac{\partial J}{\partial b^{(2)}}}{\Longleftarrow} \frac{\partial J}{\partial a^{(2)}} \overset{\dfrac{\partial J}{\partial w^{(3)}} \quad \dfrac{\partial J}{\partial b^{(3)}}}{\Longleftarrow} \frac{\partial J}{\partial \hat{y}}$$

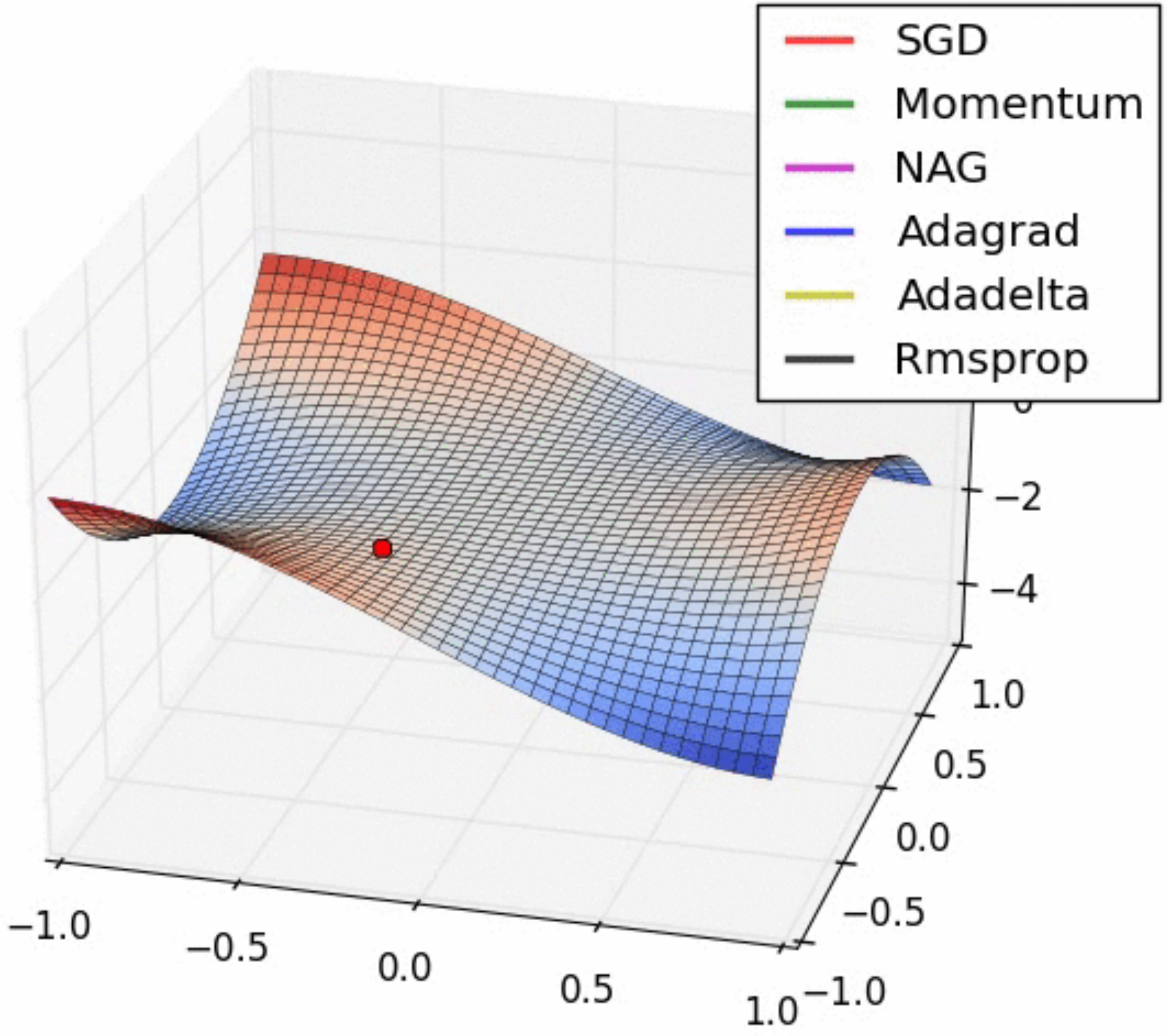
RECAP: BACKPROPAGATION



RECAP: BACKPROPAGATION



RECAP: OPTIMIZER

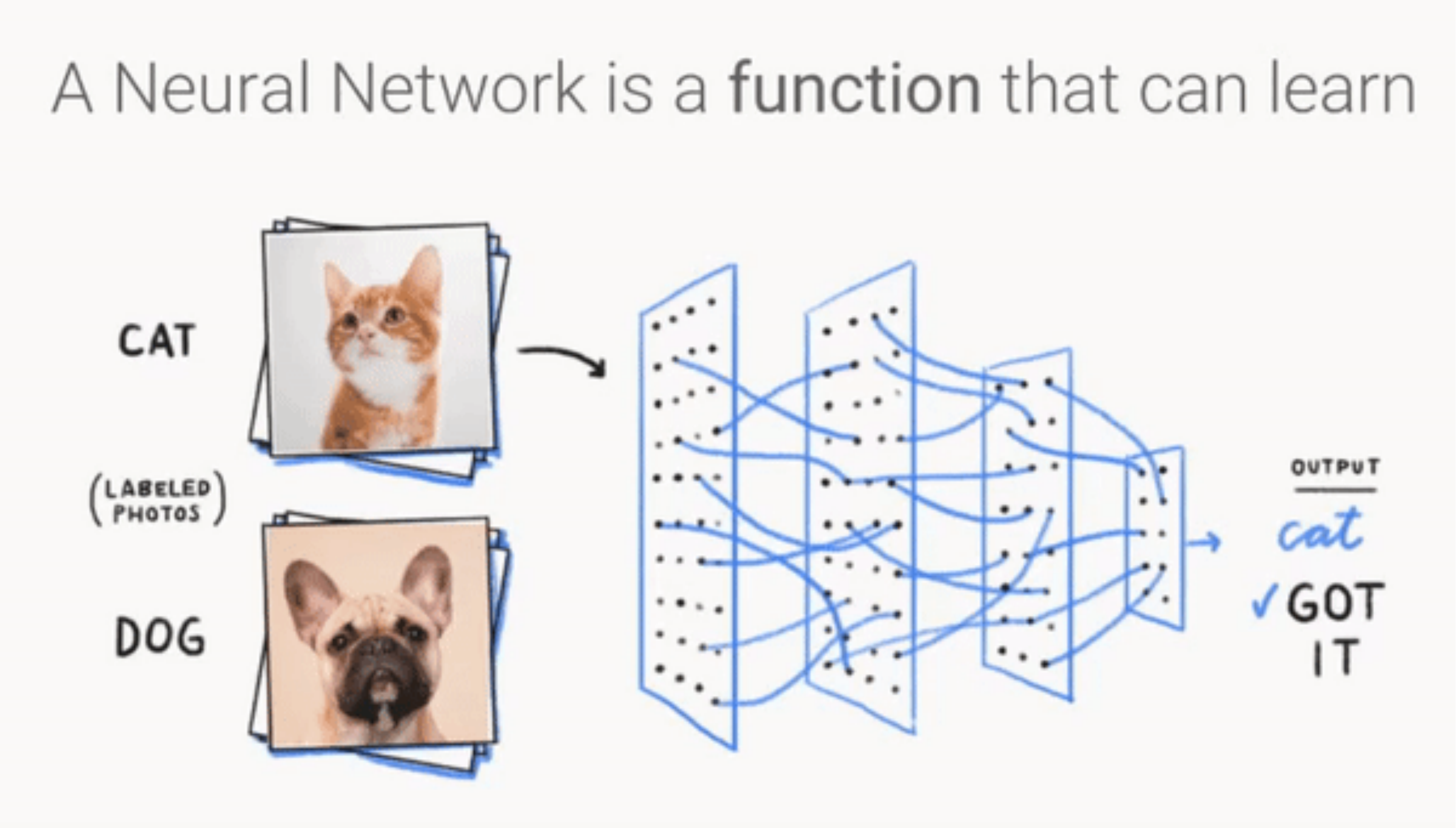


CNN – Convolution Neural Network

Content

- Understand about CNN and its component
- Understand CNN architecture and how it works
- Build a simple CNN model
- Practice based on your project dataset

What is CNN?



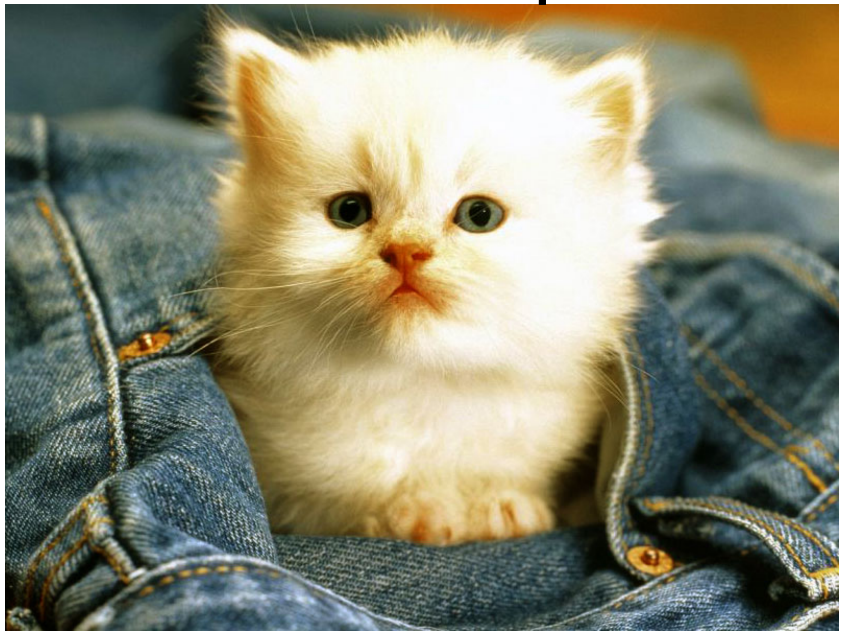
A Neural Network is a function that can learn
CAT
(LABELED PHOTOS)
DOG
OUTPUT
cat
✓GOT IT

COMPUTER VISION - COLORS

- RGB: is primary colors of light including Red, Green, and Blue
- RGB are combined in various ways to create a broad spectrum of colors
- Number of colors?

COMPUTER VISION - PIXELS

800x600 pixels



800x600 matrix

$$\begin{bmatrix} w_{1,1} & w_{1,2} & \cdots & w_{1,800} \\ w_{2,1} & w_{2,2} & \cdots & w_{2,800} \\ \cdots & \cdots & \cdots & \cdots \\ w_{600,1} & w_{600,2} & \cdots & w_{600,800} \end{bmatrix}$$

Each w_{ij} = (r, g, b) is a pixel in computer vision

COMPUTER VISION - PIXELS

Pixels data storing

$$\begin{bmatrix} 100 & 101 & 131 \\ 150 & 10 & 111 \\ 10 & 200 & 100 \end{bmatrix} \begin{bmatrix} 100 & 112 & 20 \\ 210 & 120 & 120 \\ 260 & 20 & 20 \end{bmatrix} \begin{bmatrix} 50 & 3 & 80 \\ 130 & 130 & 130 \\ 30 & 30 & 3 \end{bmatrix}$$

R G B

COMPUTER VISION - PIXELS

Image in computer vision is a 3D Tensor

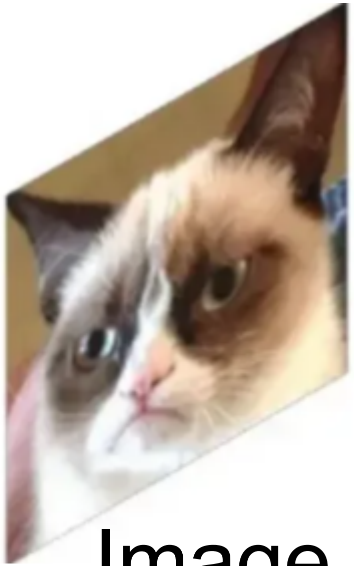
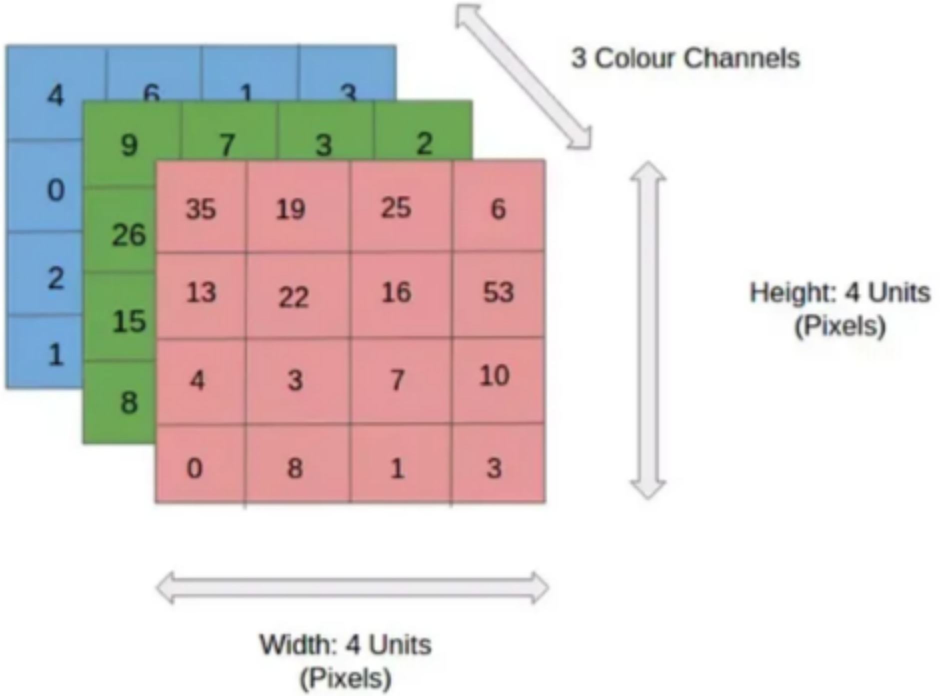


Image
(RGB)



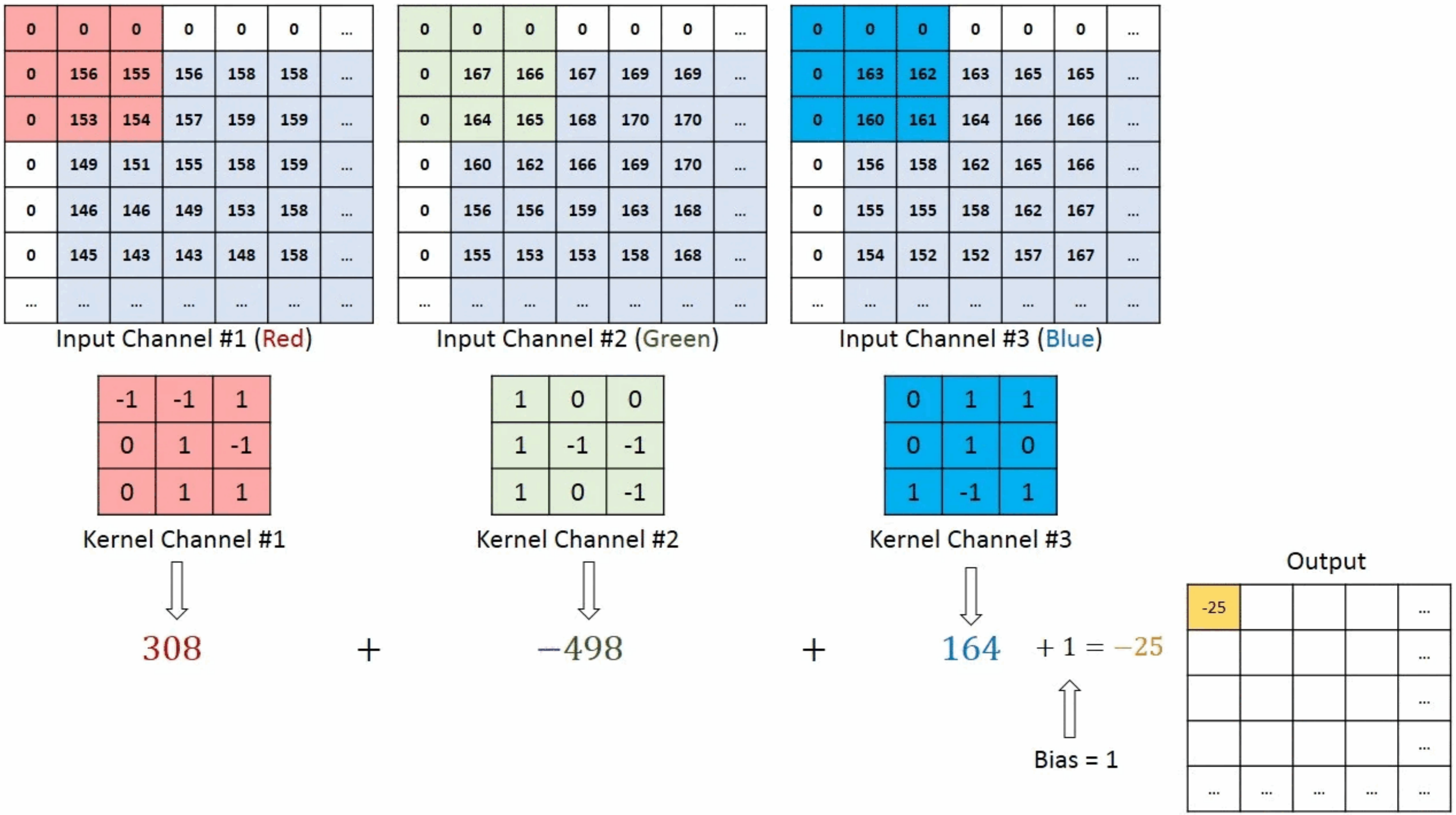
COMPUTER VISION – Grey Scale

Convert formula: w = r*0.299 + g*0.587 + b*0.114

Image
(RGB)

Image
(Grey scale)

What is Convolution?



What is Convolution?

Input Image

0	0	0	0	0	0	0
0	1	1	1	1	1	0
0	1	0	0	1	0	0
0	1	0	1	0	0	0
0	1	1	0	0	0	0
0	1	0	0	0	0	0
0	0	0	0	0	0	0

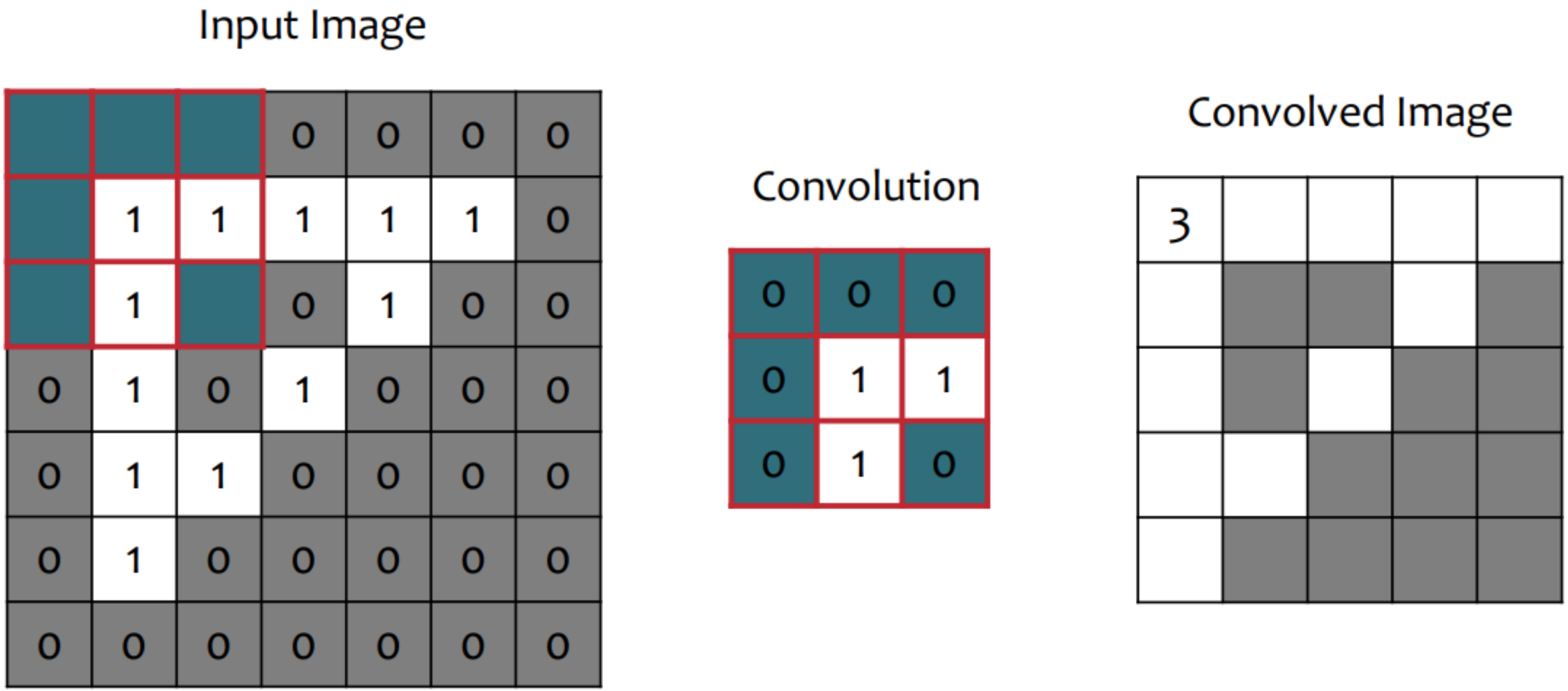
Convolution

0	0	0
0	1	1
0	1	0

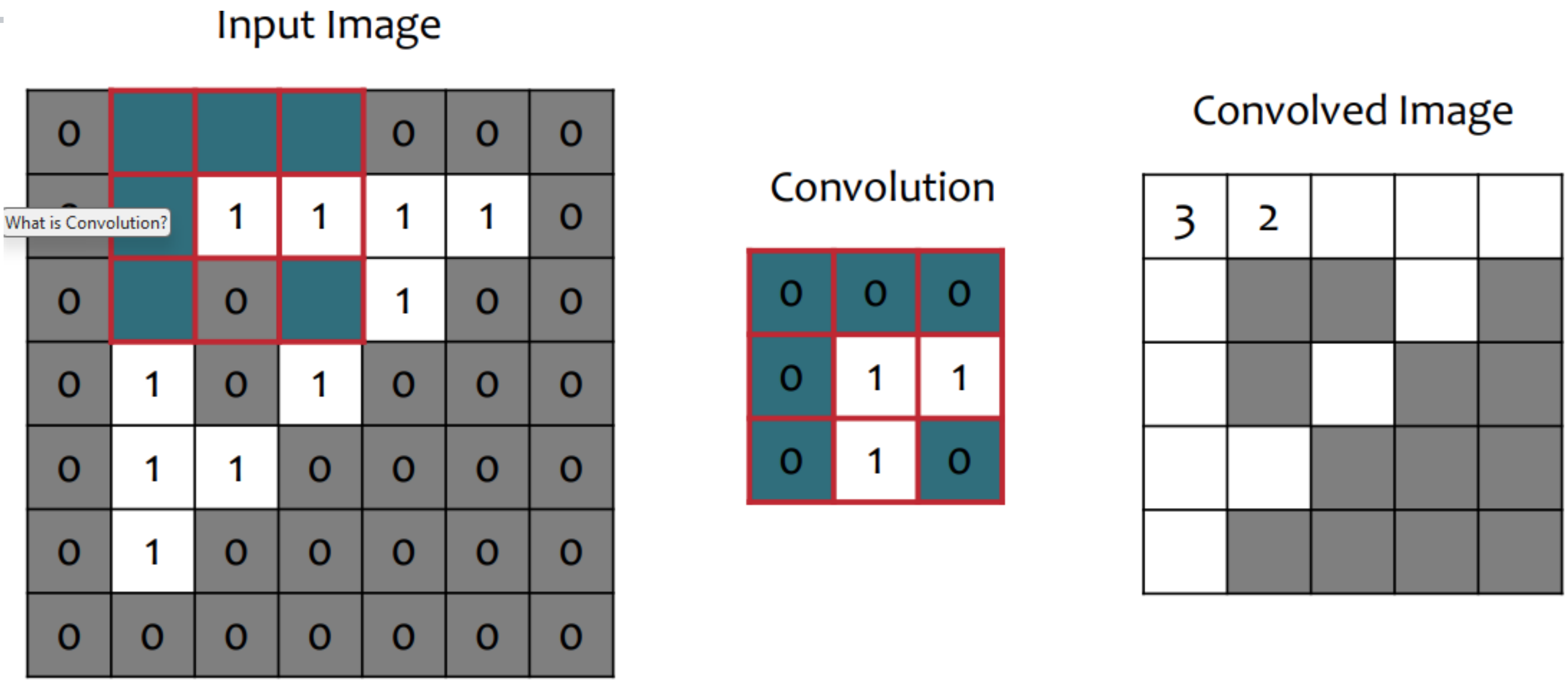
Convolved Image

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

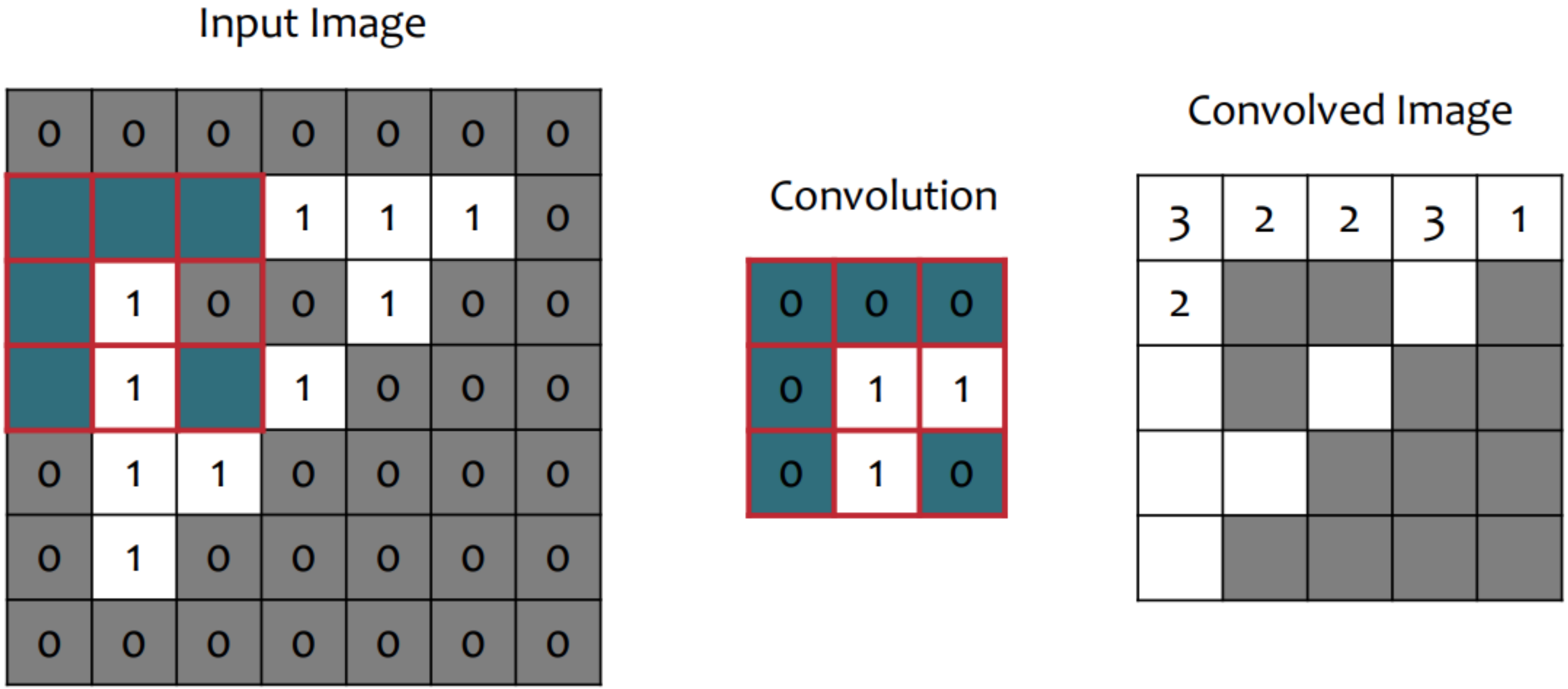
What is Convolution?



What is Convolution?



What is Convolution?



Convolution and Image processing

- A convolution matrix is used in image processing for tasks such as edge detection, bluring, sharpening, …

Convolution and Image processing - Identity

FPT Education SCHOOL OF BUSINESS & TECHNOLOGY

Input Image

0	0	0	0	0	0	0
0	1	1	1	1	1	0
0	1	0	0	1	0	0
0	1	0	1	0	0	0
0	1	1	0	0	0	0
0	1	0	0	0	0	0
0	0	0	0	0	0	0

Identity Convolution

0	0	0
0	1	0
0	0	0

Convolved Image

1	1	1	1	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	0	0	0	0

Convolution and Image processing - Blurring

Input Image

0	0	0	0	0	0	0
0	1	1	1	1	1	0
0	1	0	0	1	0	0
0	1	0	1	0	0	0
0	1	1	0	0	0	0
0	1	0	0	0	0	0
0	0	0	0	0	0	0

Blurring Convolution

.1	.1	.1
.1	.2	.1
.1	.1	.1

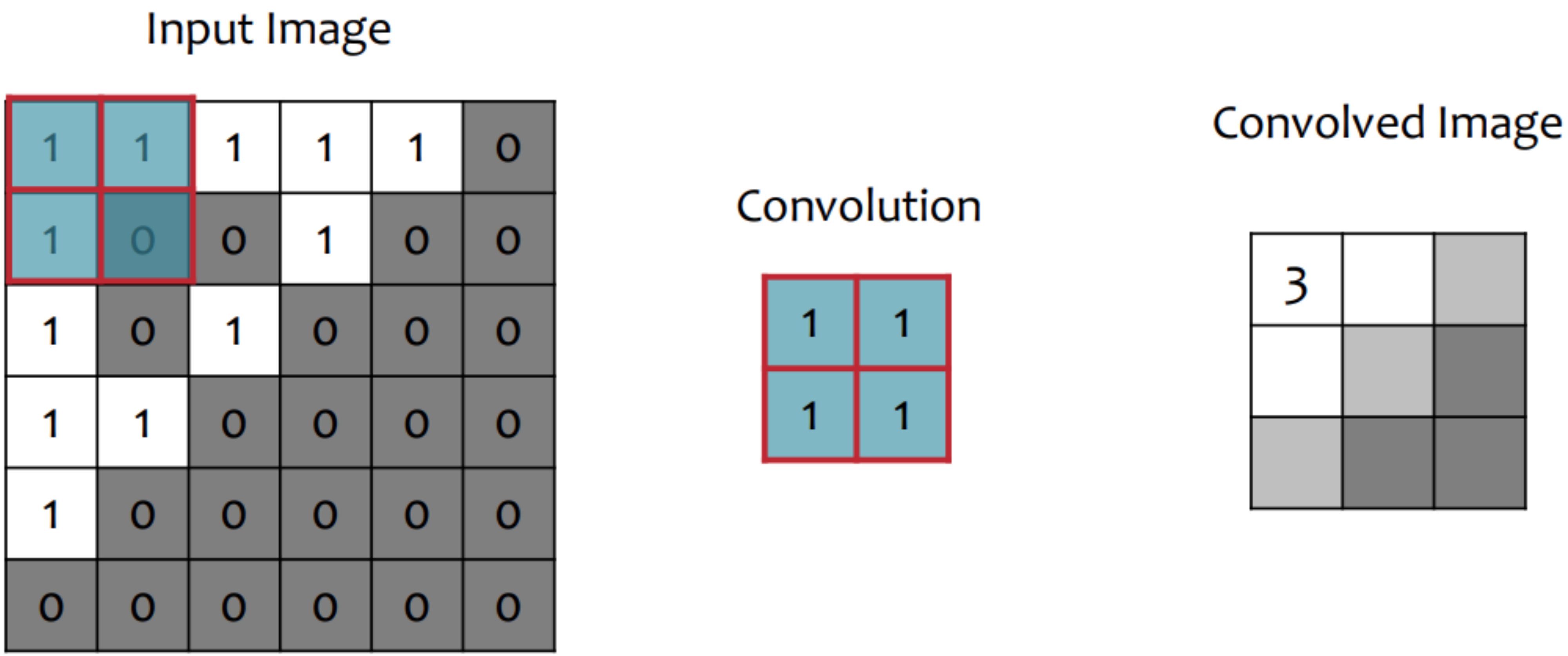
Convolved Image

.4	.5	.5	.5	.4
.4	.2	.3	.6	.3
.5	.4	.4	.2	.1
.5	.6	.2	.1	0
.4	.3	.1	0	0

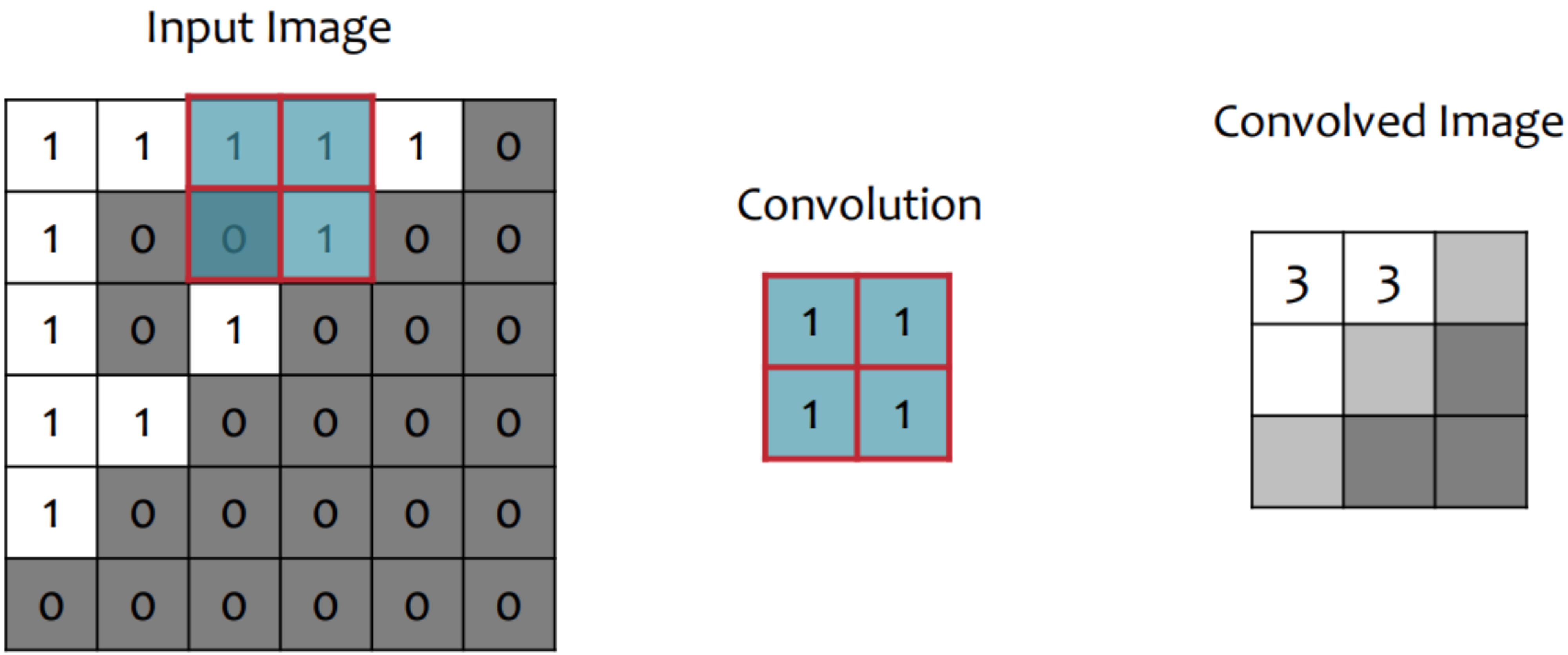
Convolution – Downsampling (Stride)

- **Stride** refers to the number of pixels by which the filter (or kernel) moves across the input image during the convolution operation.
- Output_size = (input_size – kernel_size)/stride + 1

Convolution – Downsampling (Stride)



Convolution – Downsampling (Stride)



Convolution – Downsampling (Stride)

Input Image

1	1	1	1	1	0
1	0	0	1	0	0
1	0	1	0	0	0
1	1	0	0	0	0
1	0	0	0	0	0
0	0	0	0	0	0

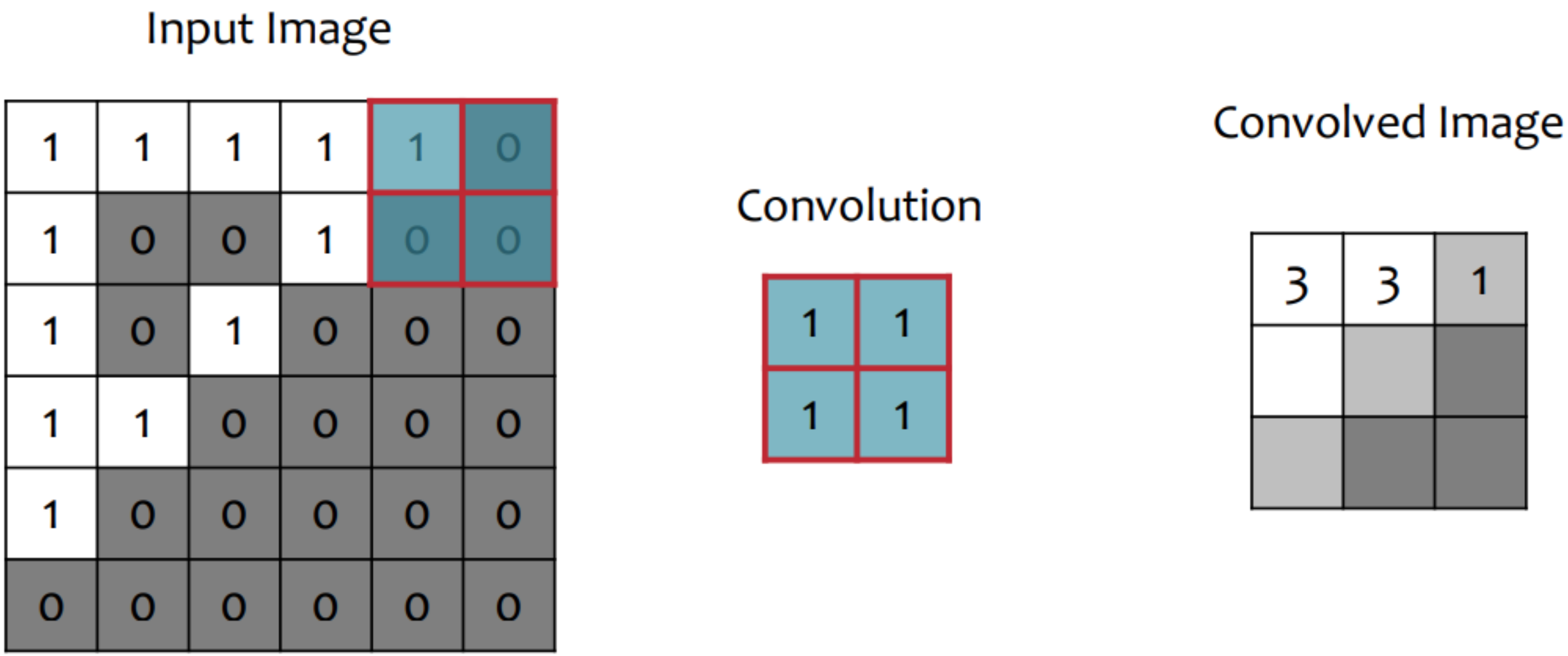
Convolution

1	1
1	1

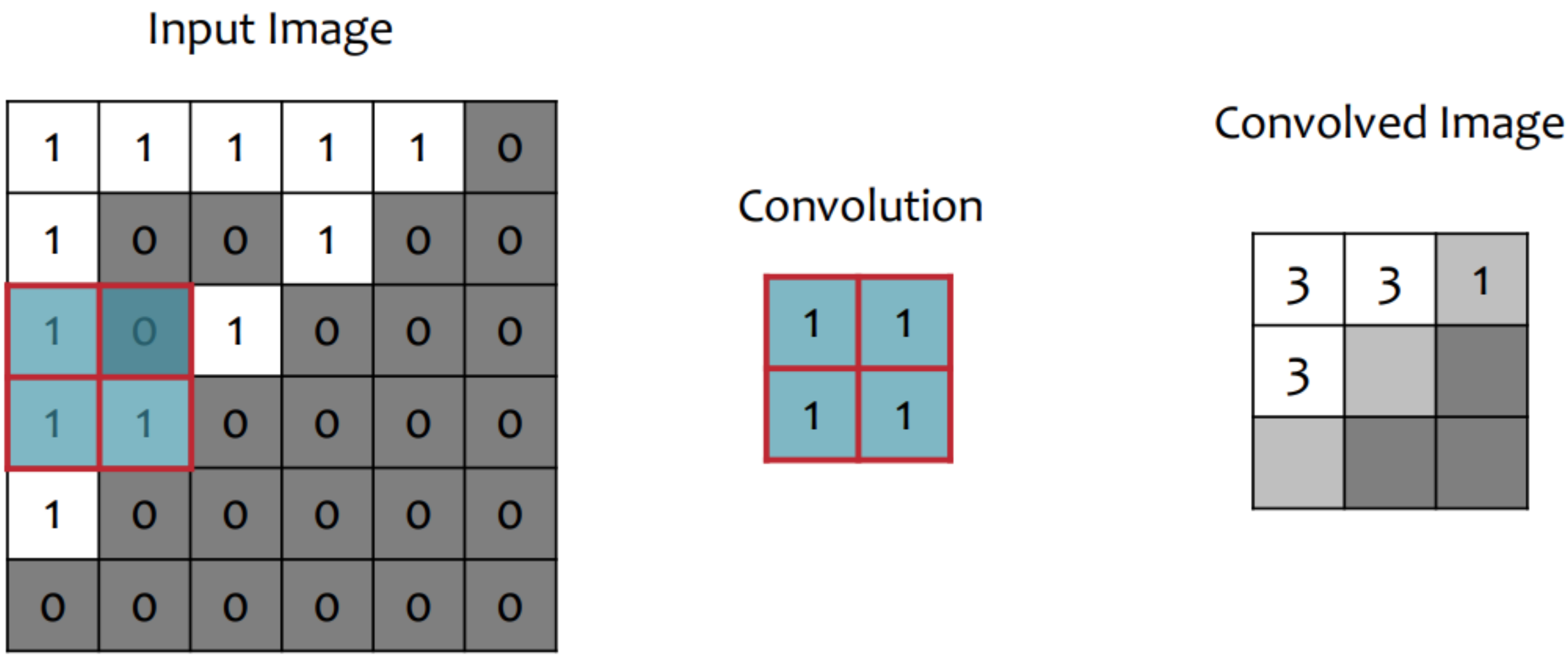
Convolved Image

3	3	1

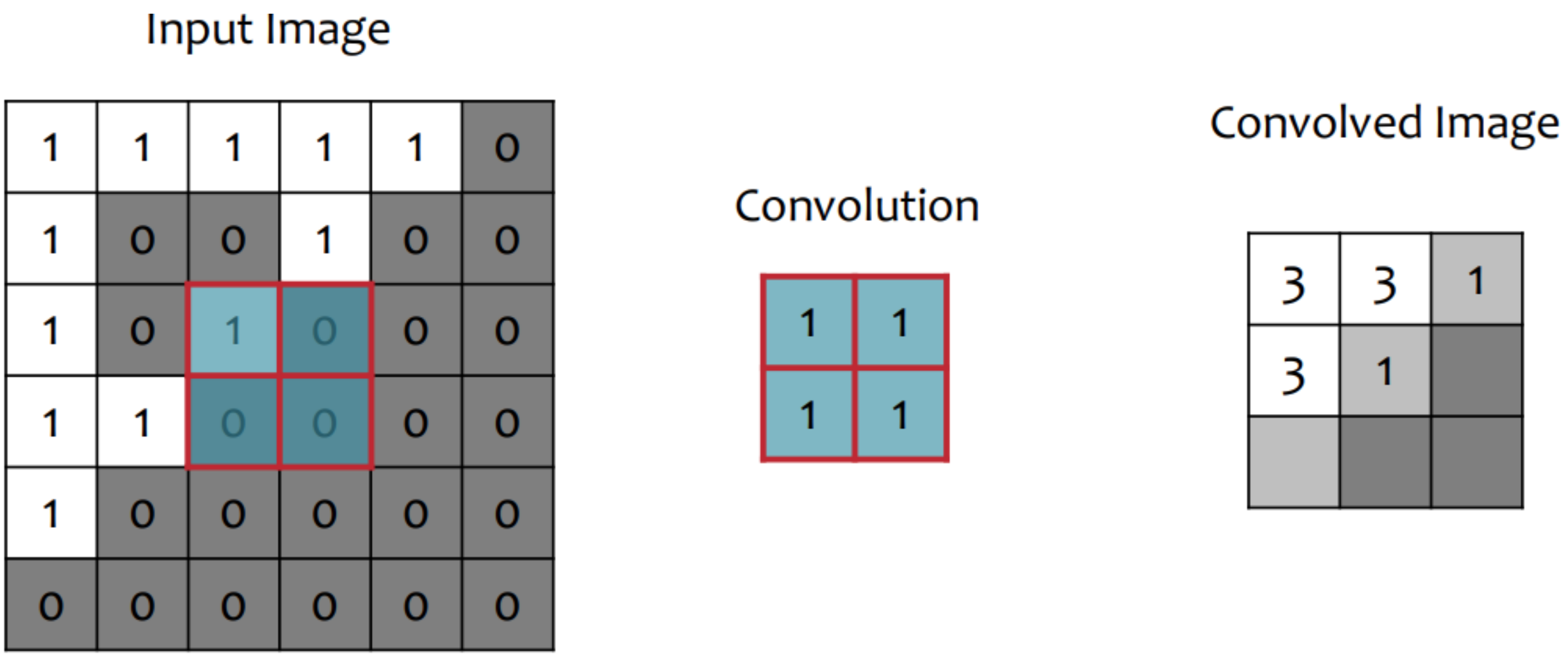
Convolution – Downsampling (Stride)



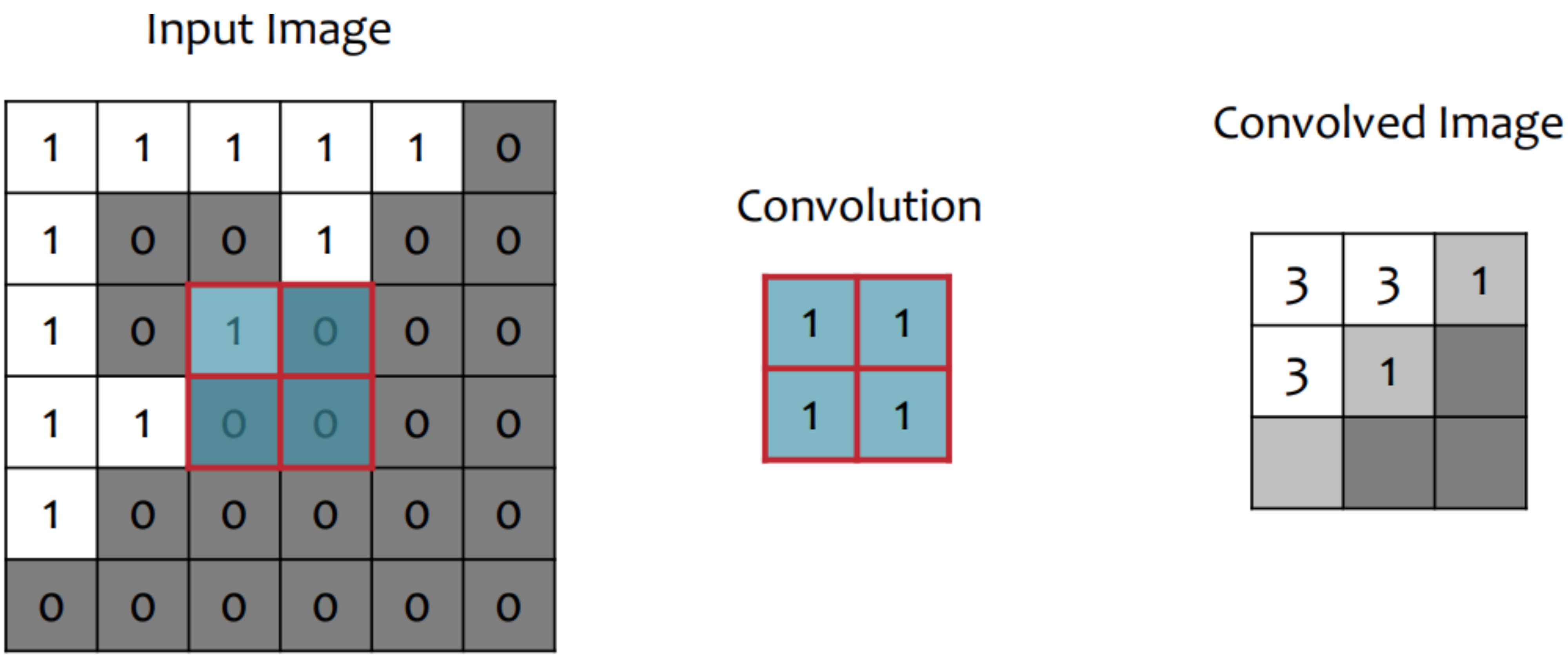
Convolution – Downsampling (Stride)



Convolution – Downsampling (Stride)



Convolution – Downsampling (Stride)



Convolution – Downsampling (Stride)

Input Image

1	1	1	1	1	0
1	0	0	1	0	0
1	0	1	0	0	0
1	1	0	0	0	0
1	0	0	0	0	0
0	0	0	0	0	0

Convolution

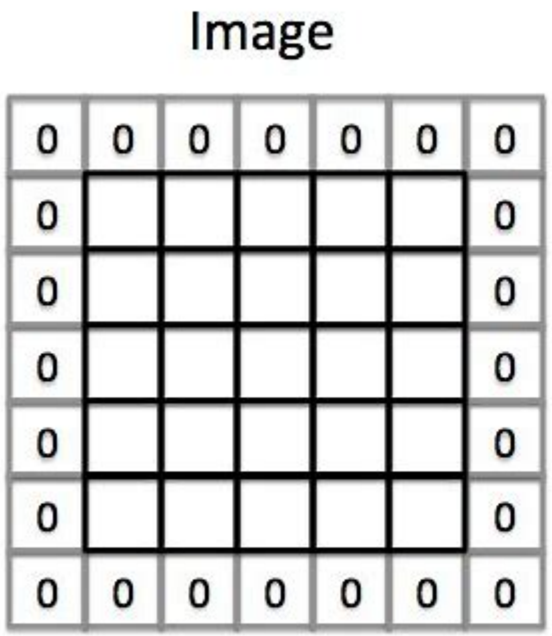
1	1
1	1

Convolved Image

3	3	1
3	1	0
1	0	0

Convolution – Padding

- **Padding** is the process of adding extra pixels around the border of an input image before applying a convolutional filter
- Example of zero padding:



Convolution – Padding

- Preserves Edge Information: Without padding, the pixels at the edges and corners of the image are used less frequently, leading to loss of information.
- Controls Output Size: Padding allows control over the output size for designing deep networks.
- Prevents Shrinking: Successive convolutions without padding would continuously reduce the size of the feature maps, eventually making them too small

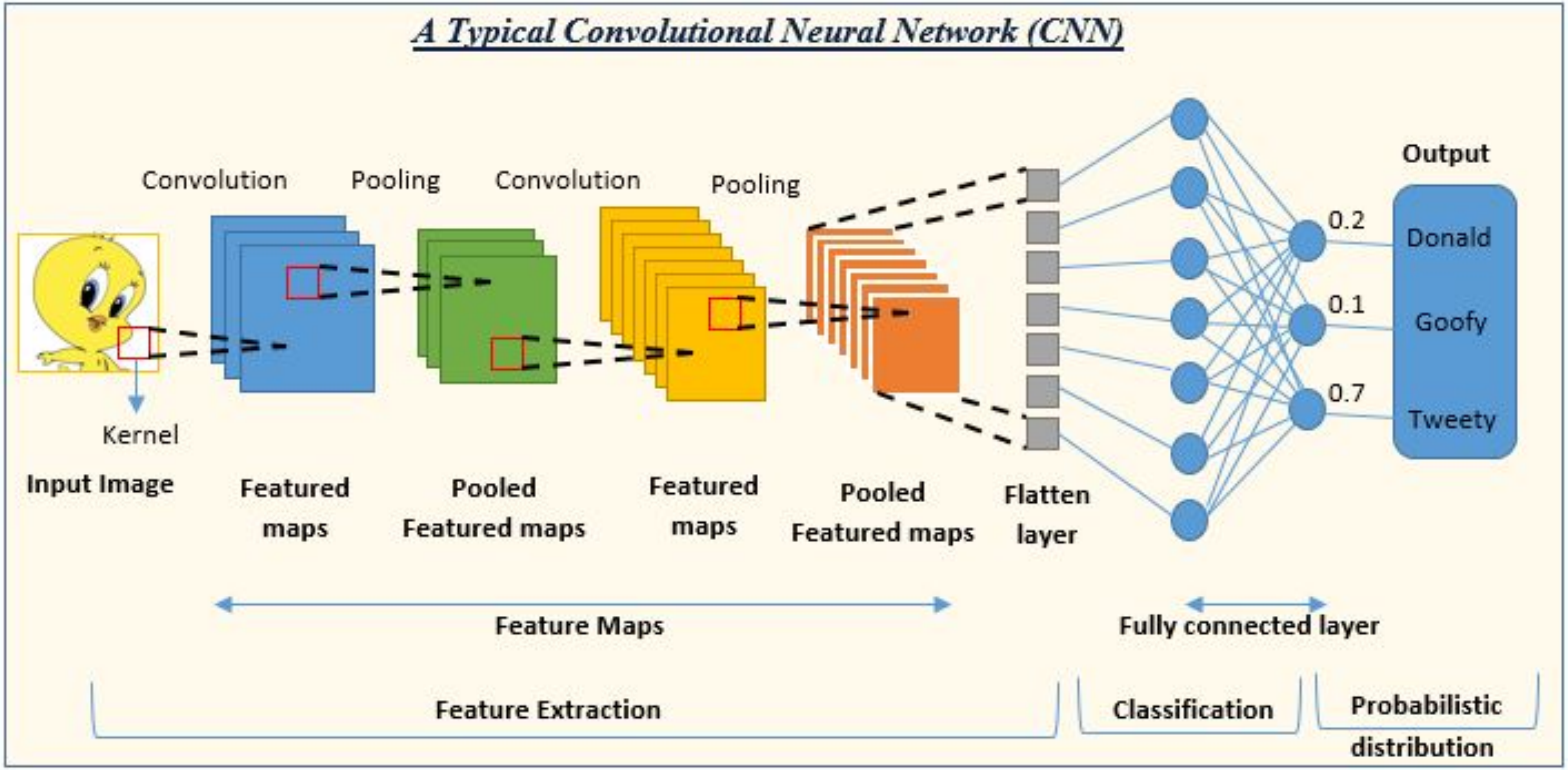
Convolution – Padding

- $outputsize = \frac{inputsize - kernel + 2 * padding}{stride} + 1$

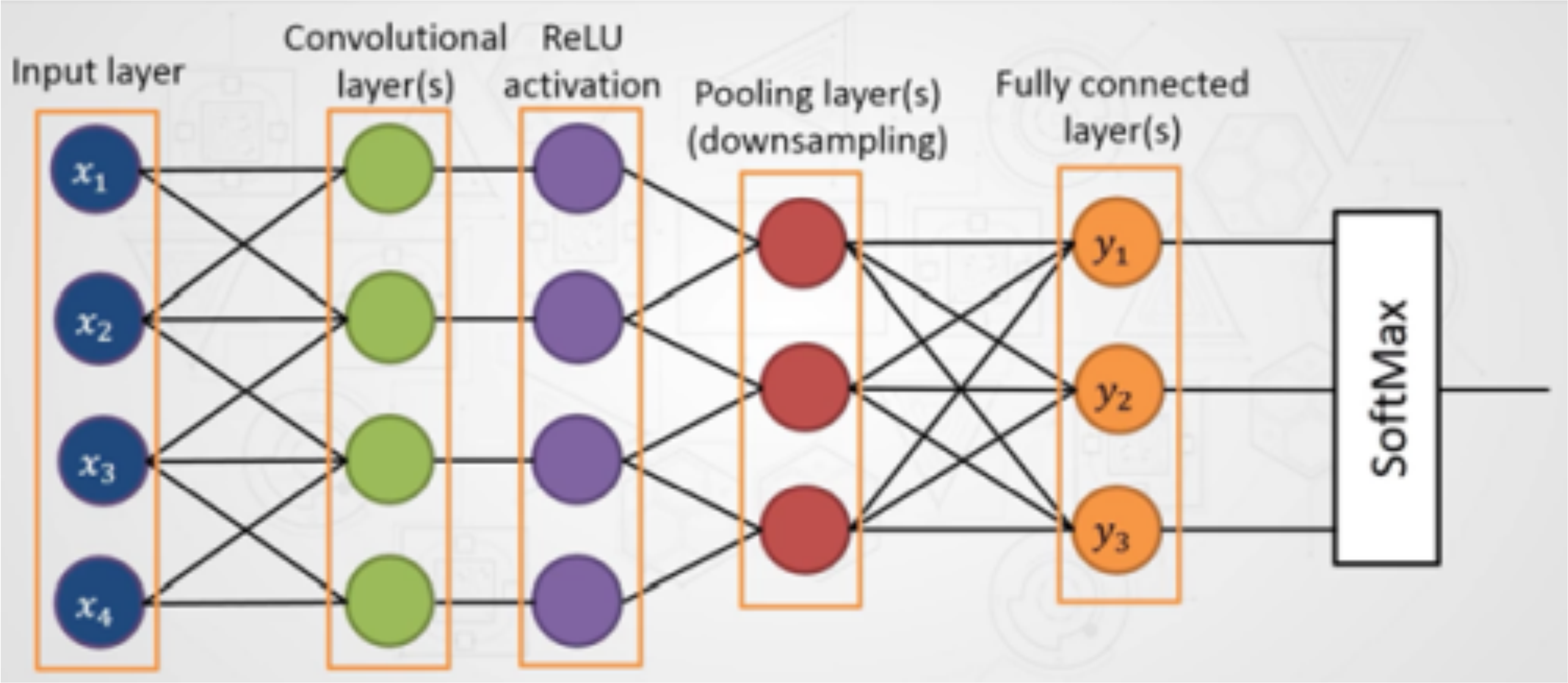
Convolution Neural Network - CNN

- A Convolutional Neural Network (CNN) is a specialized type of artificial neural network (ANN).
- CNN was designed to process and analyze visual data with some convolution layers.
- CNNs are widely used in computer vision applications such as image classification, object detection, and image segmentation.

A typical CNN



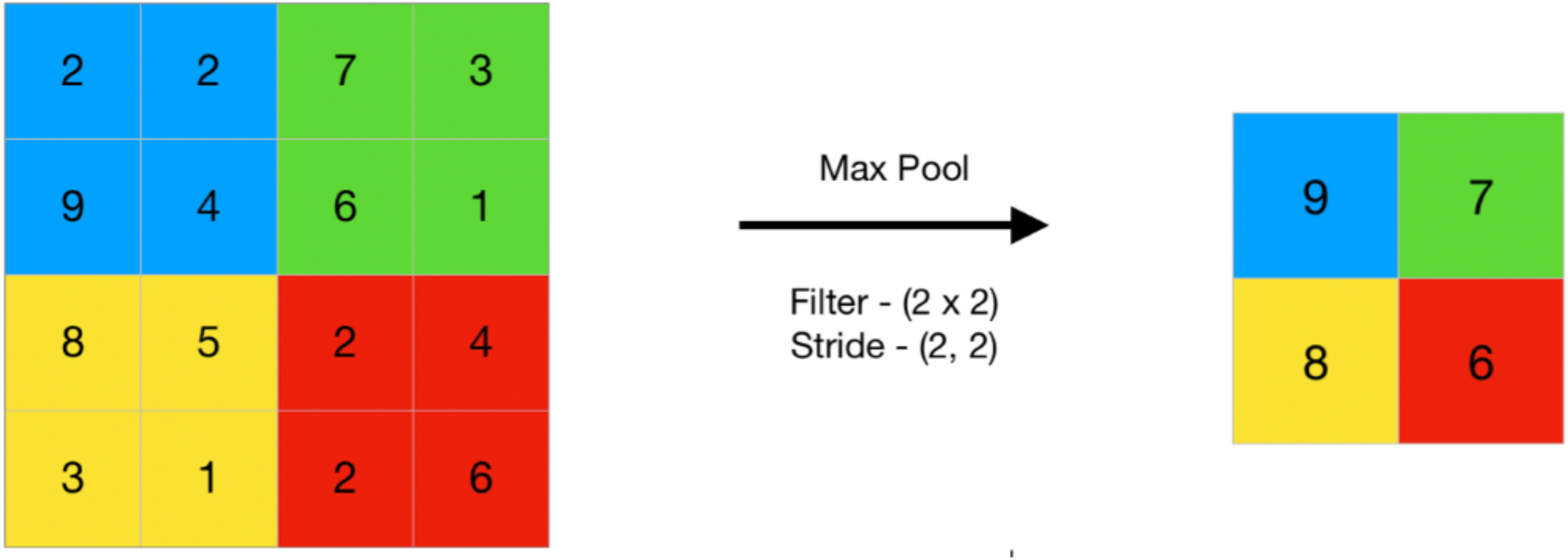
A typical CNN



CNN - Key Components

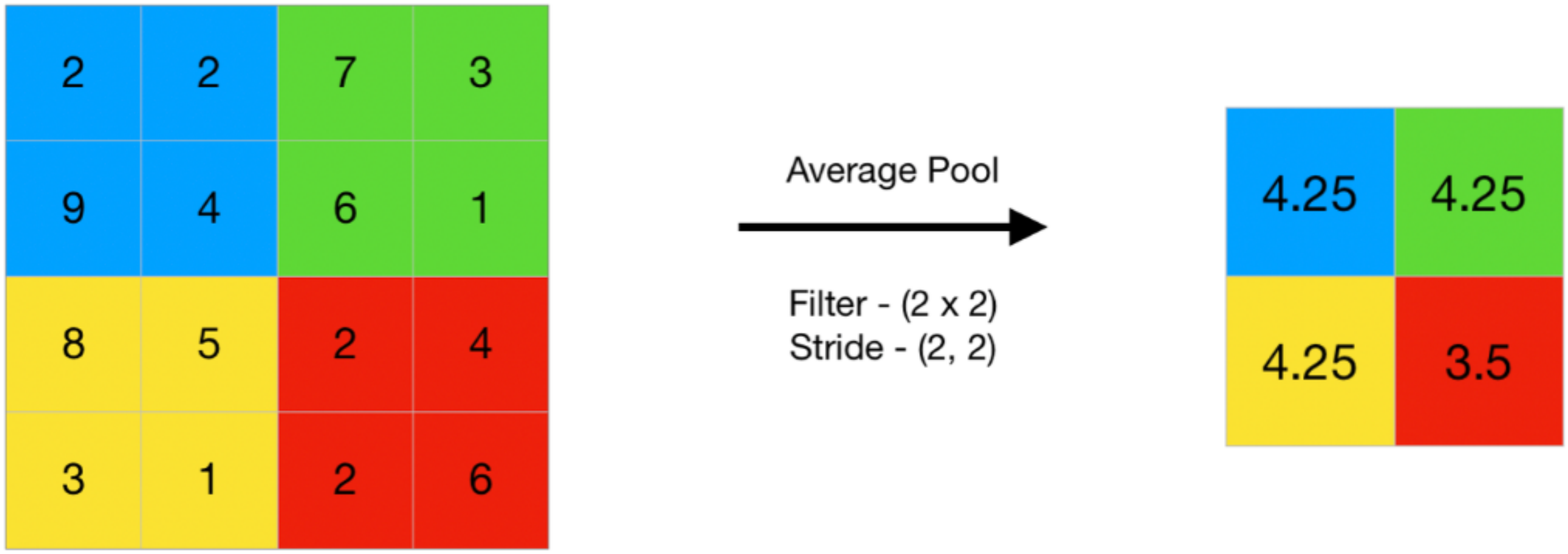
- **Convolutional Layers**: extract features such as edges, textures, and patterns.
- **Pooling Layers:** reduce the spatial dimensions, decrease the computational load and control overfitting. Common types include max pooling and average pooling.
- **Fully Connected Layers:** similar to traditional neural networks and are used to make the final classification or prediction based on the extracted features.
- **Activation Functions:** Functions like ReLU, Softmax.

CNN – Pooling layer



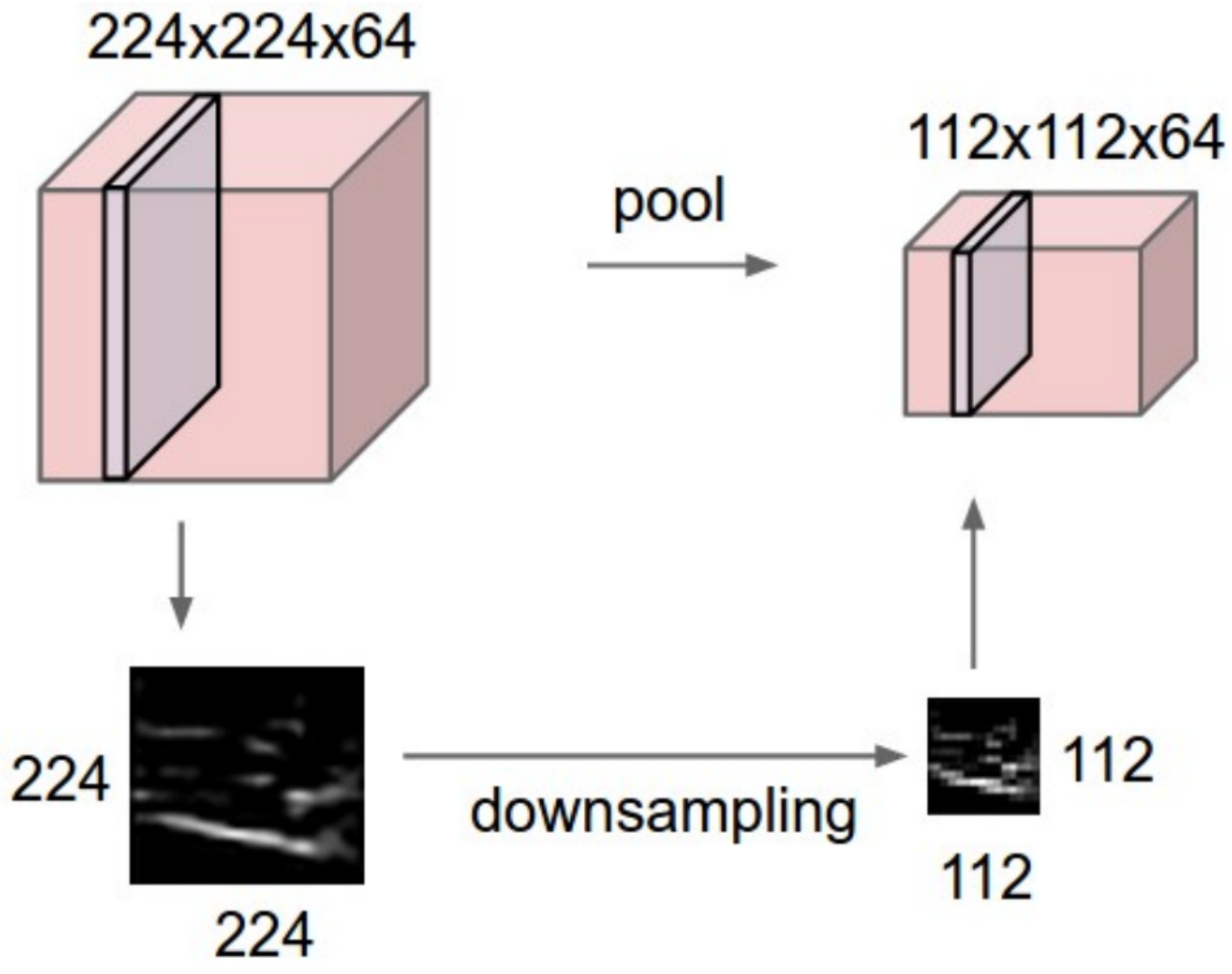
Max pooling with 2x2 matrix

CNN – Pooling layer



Average pooling with 2x2 matrix

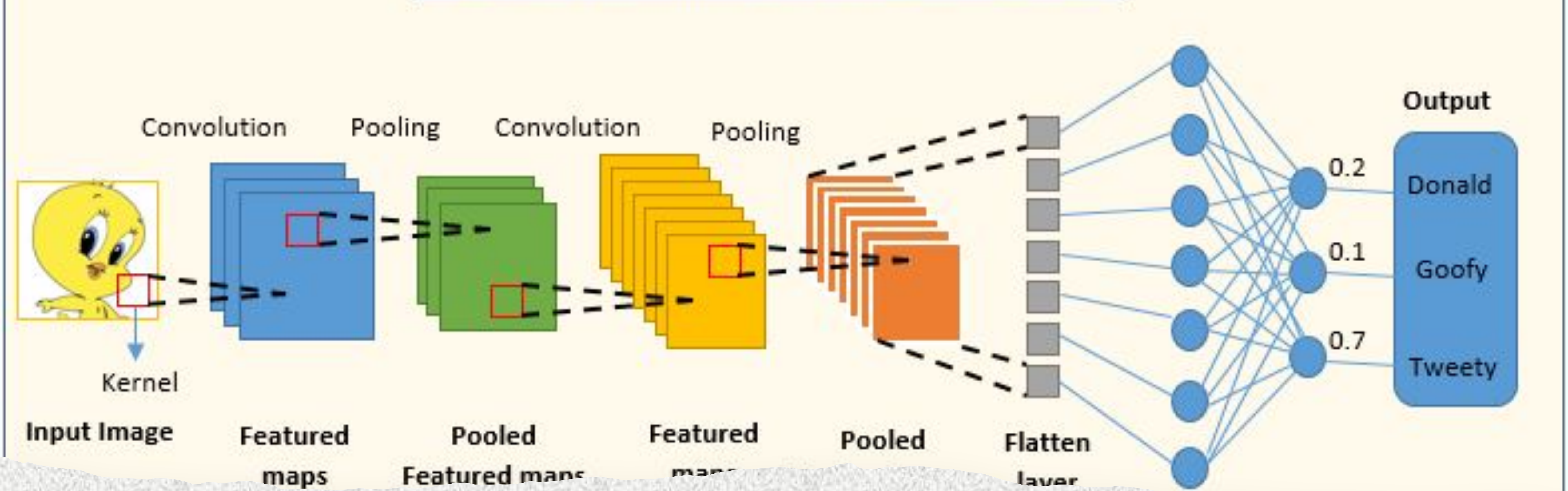
CNN – Pooling layer

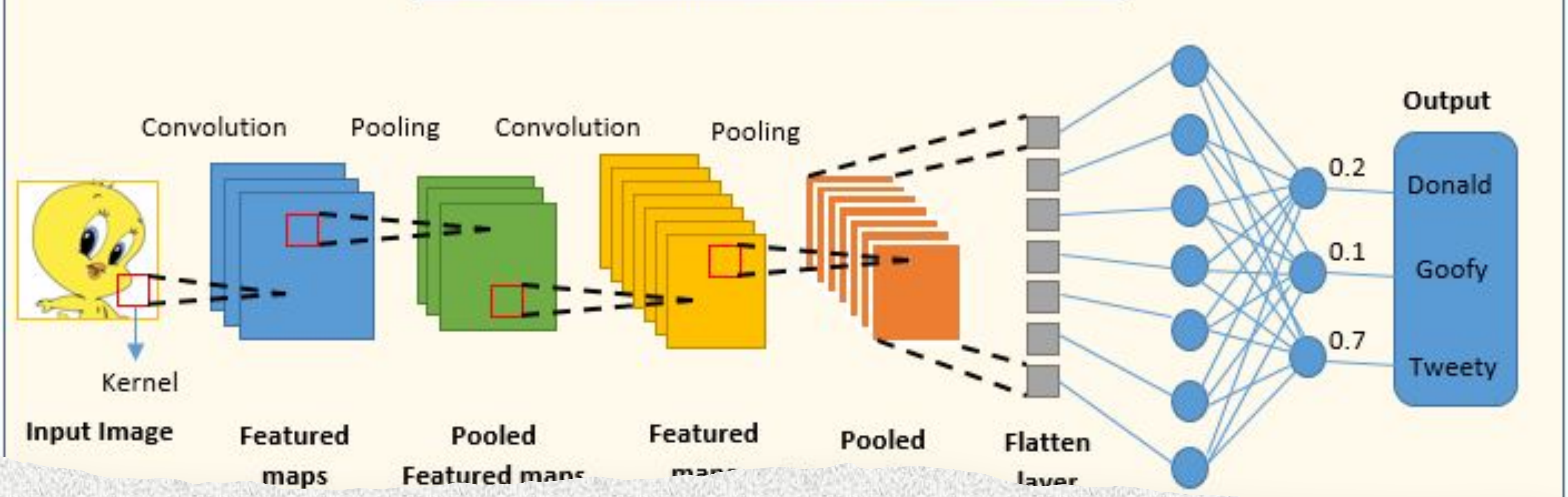


Pooling layer with 2x2 matrix

CNN – Pooling layer

- A pooling layer is used to:
  - Reduce the spatial dimensions (width and height) while preserving the depth (number of channels) and important features.
  - Down sample the feature maps, making the network more computationally efficient.


Convolution
Pooling
Convolution
Pooling
Output
Kernel
0.2
0.1
0.7
Donald
Goofy
Tweety
Input Image
Featured maps
Pooled Featured maps
Featured
Pooled
Flatten


Convolution
Pooling
Convolution
Pooling
Output
0.2
Donald
0.1
Goofy
0.7
Tweety
Kernel
Input Image
Featured
maps
Pooled
Featured maps
Featured
Pooled
Flatten