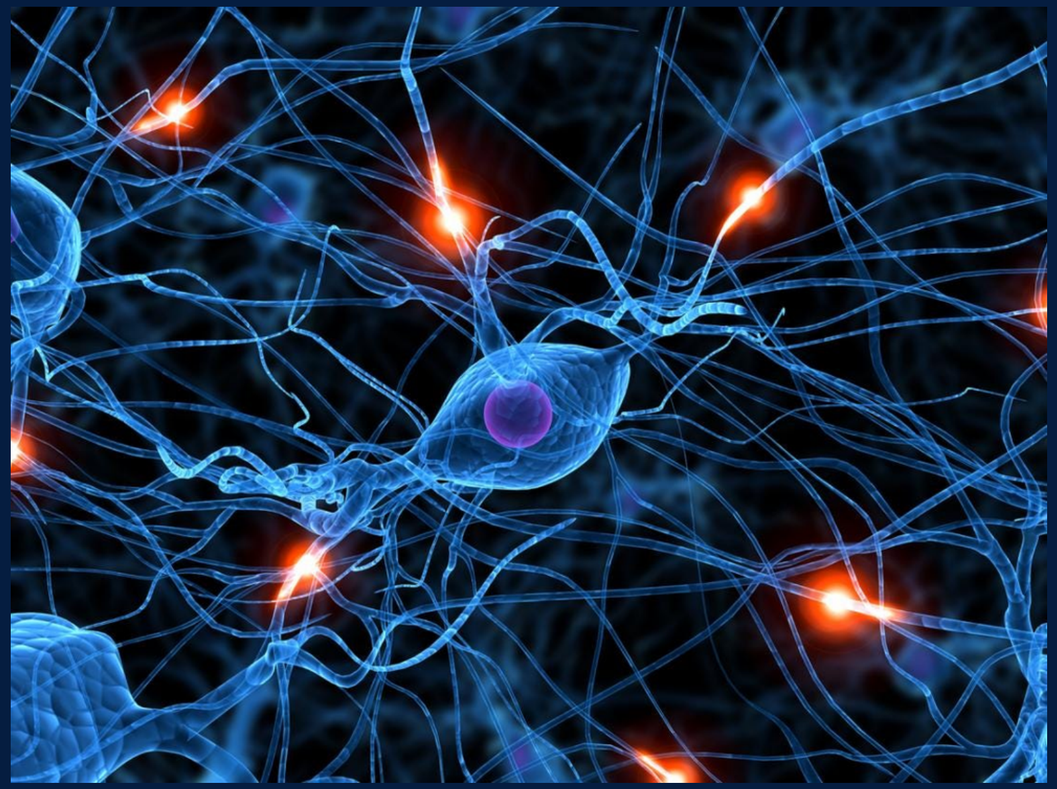
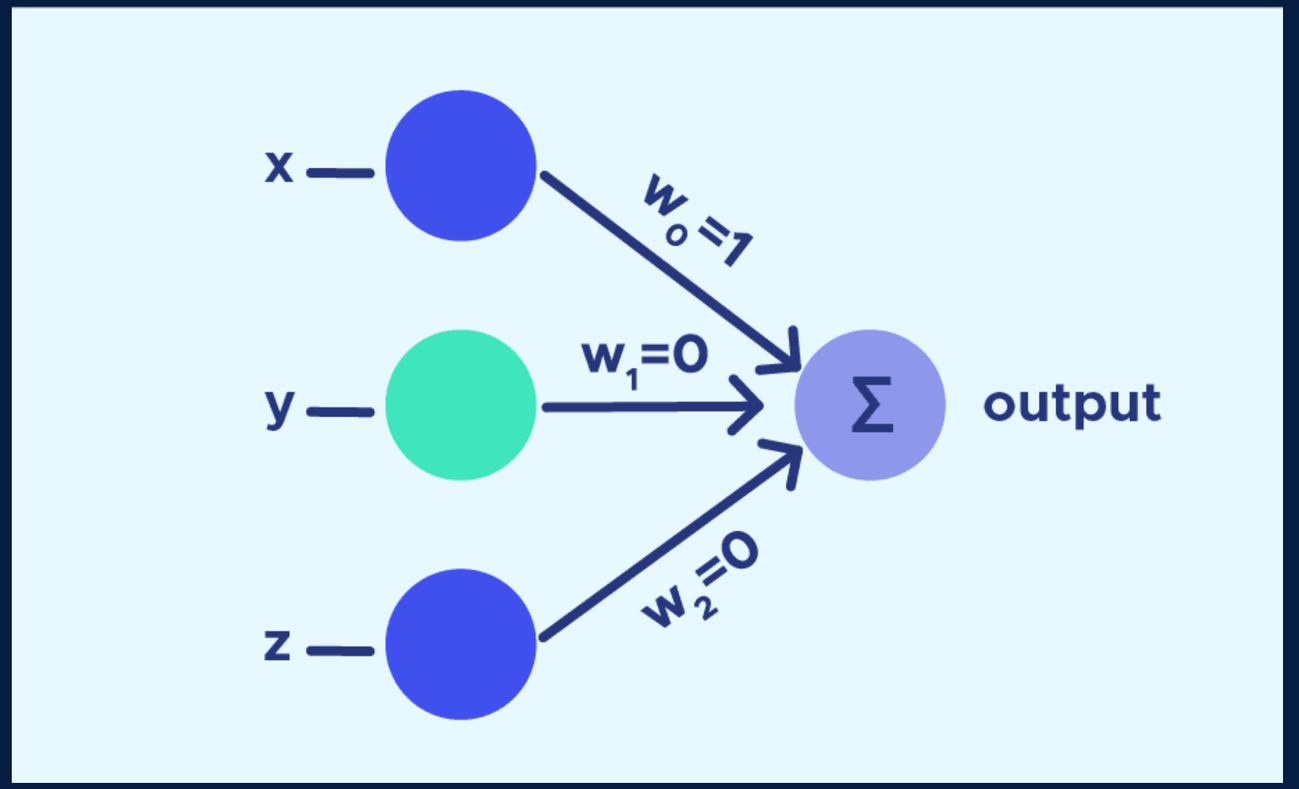
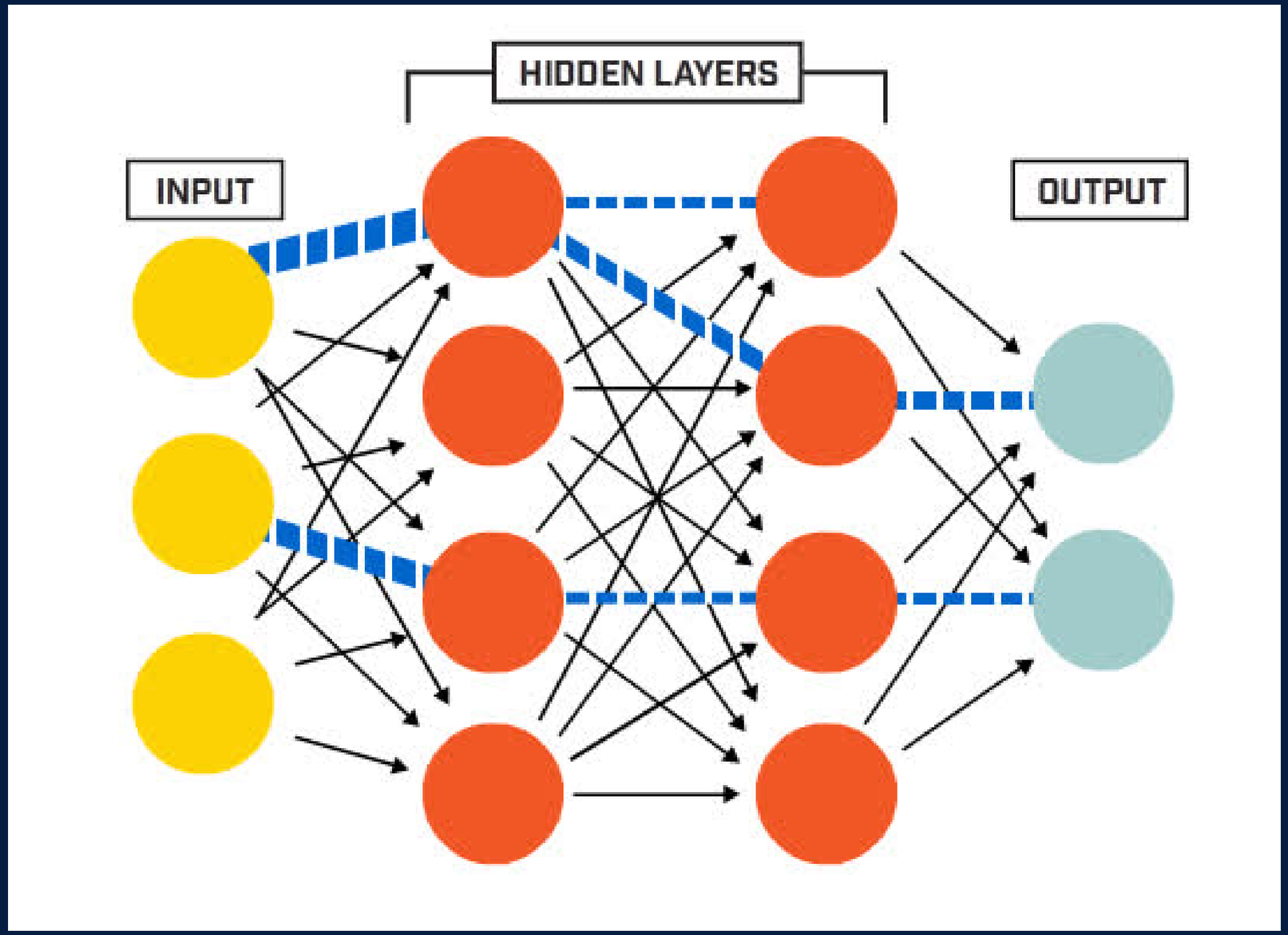
INTRODUCTION TO CNN

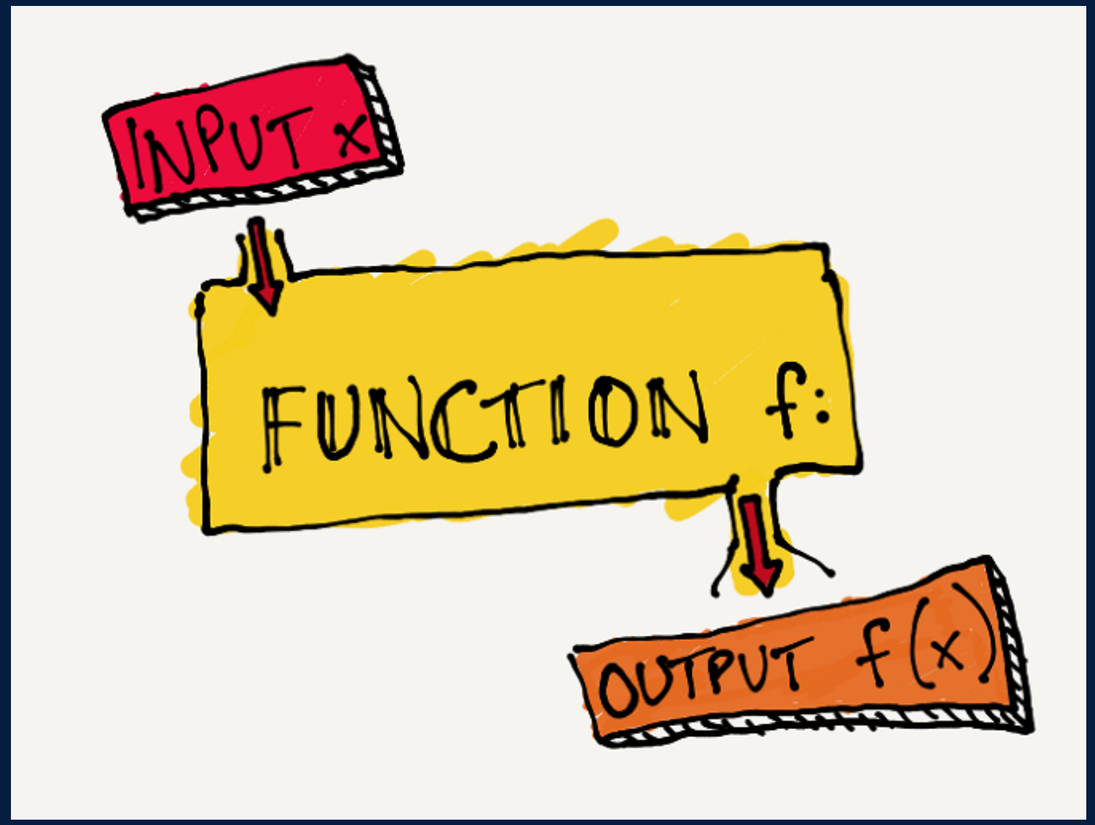
CONVOLUTION NEURAL NETWORK

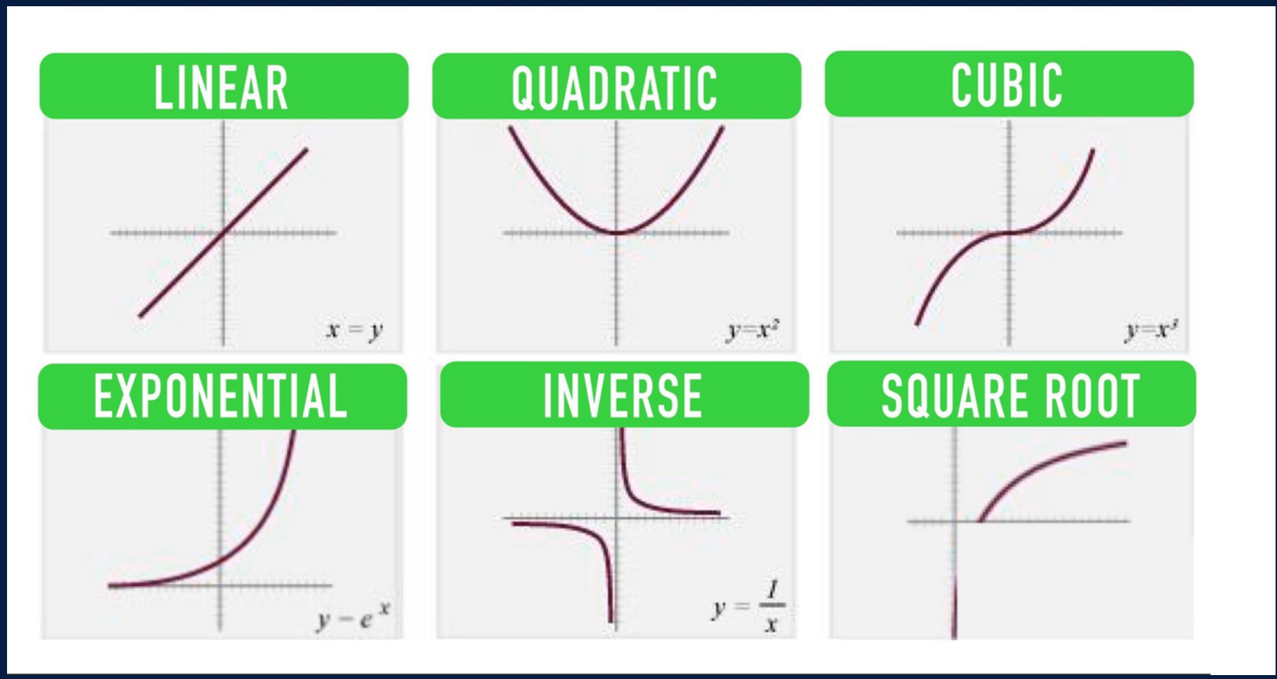
WHAT IS A NEURAL NETWORK?

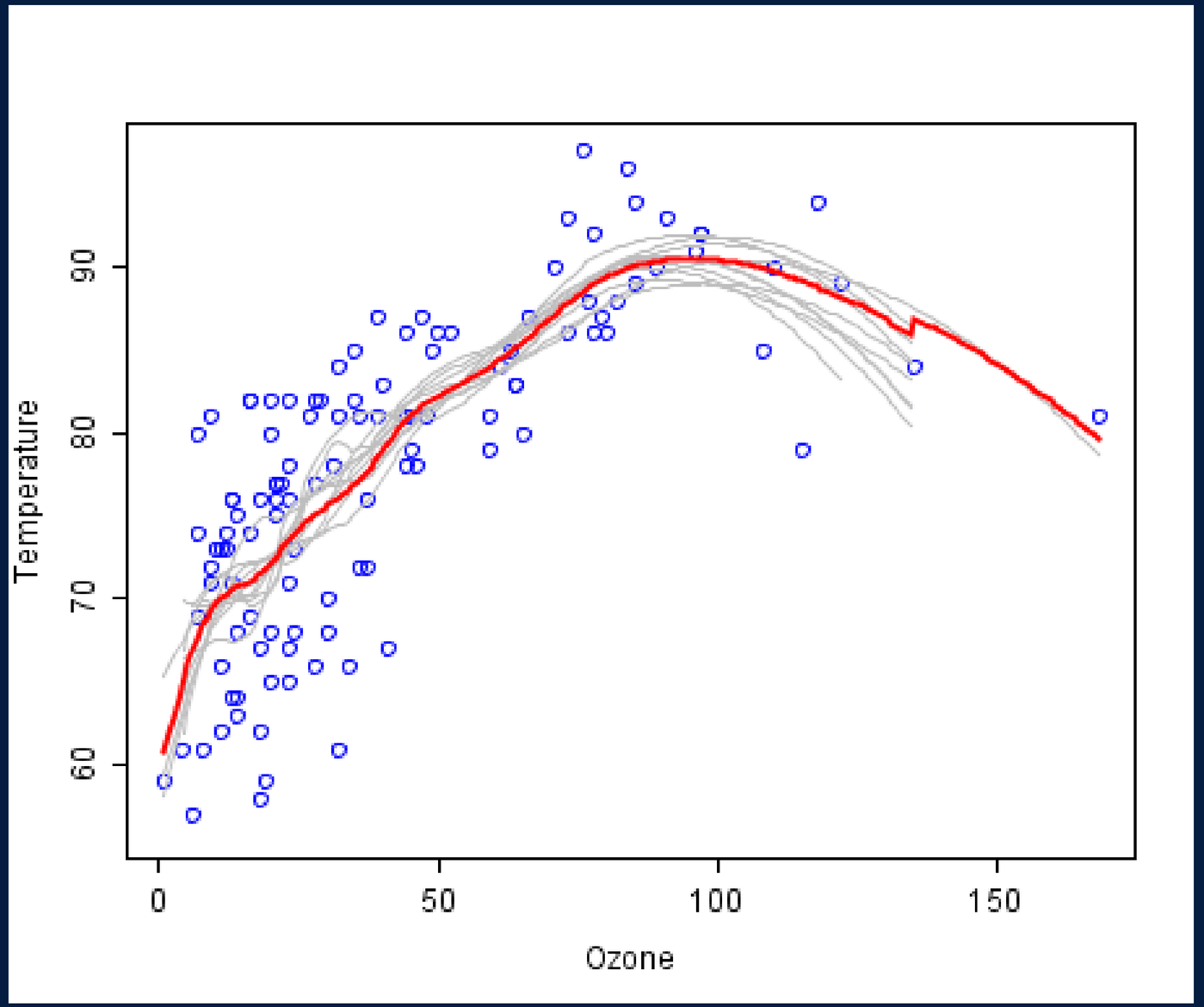





HIDDEN LAYERS
INPUT
OUTPUT


INPUT x
FUNCTION f:
OUTPUT f(x)


LINEAR
x = y
QUADRATIC
y=x²
CUBIC
y=x³
EXPONENTIAL
y = e^x
INVERSE
y = 1/x
SQUARE ROOT


Temperature
60
70
80
90
0
50
100
150
Ozone

CONVUOLUTION NEURAL NETWORKS

IMAGE CLASSIFICATION

INPUT

OUTPUT

Cat 0.9

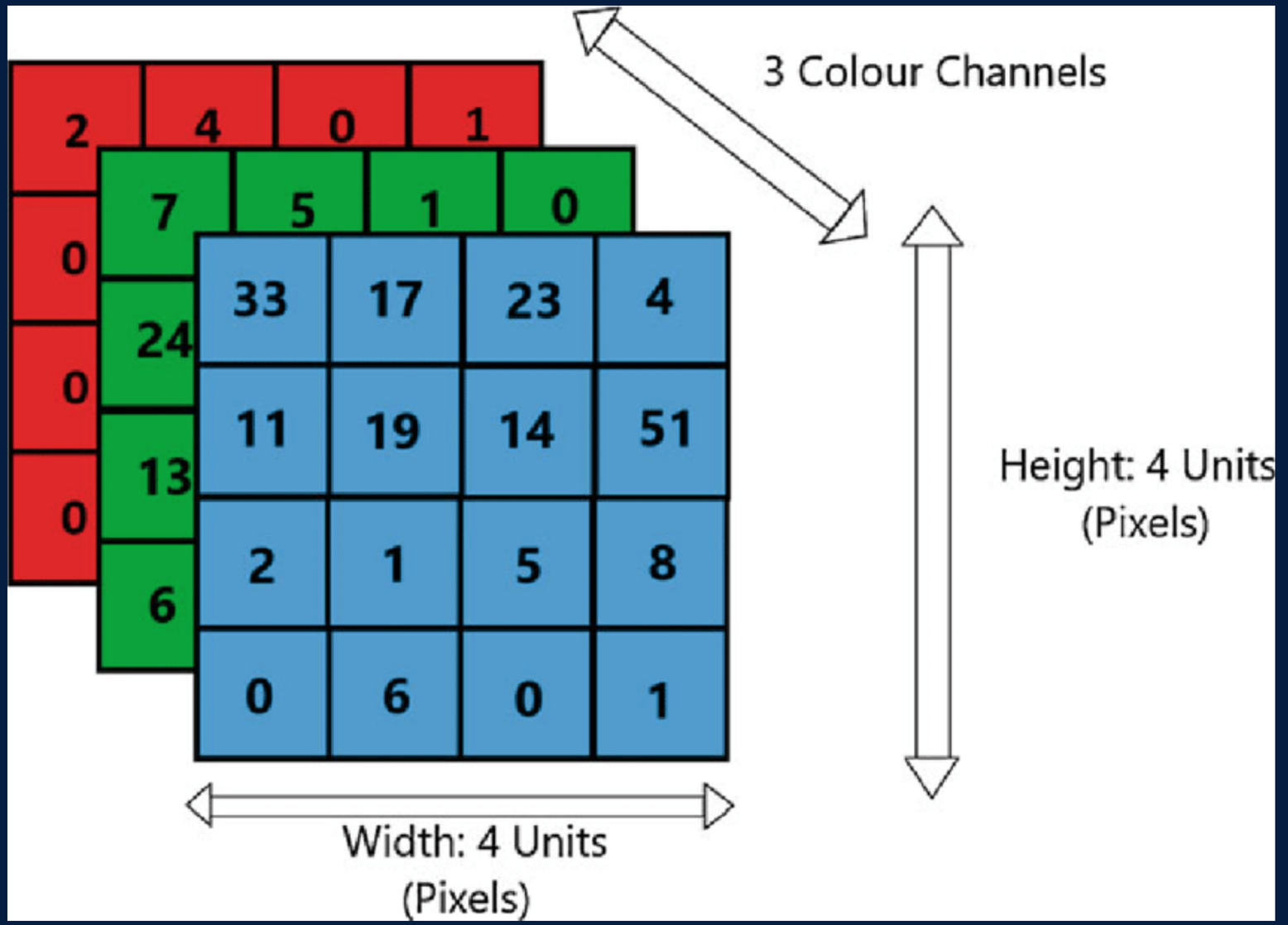
Dog 0.6

Bird 0.1

OBJECT DETECTION

INPUT

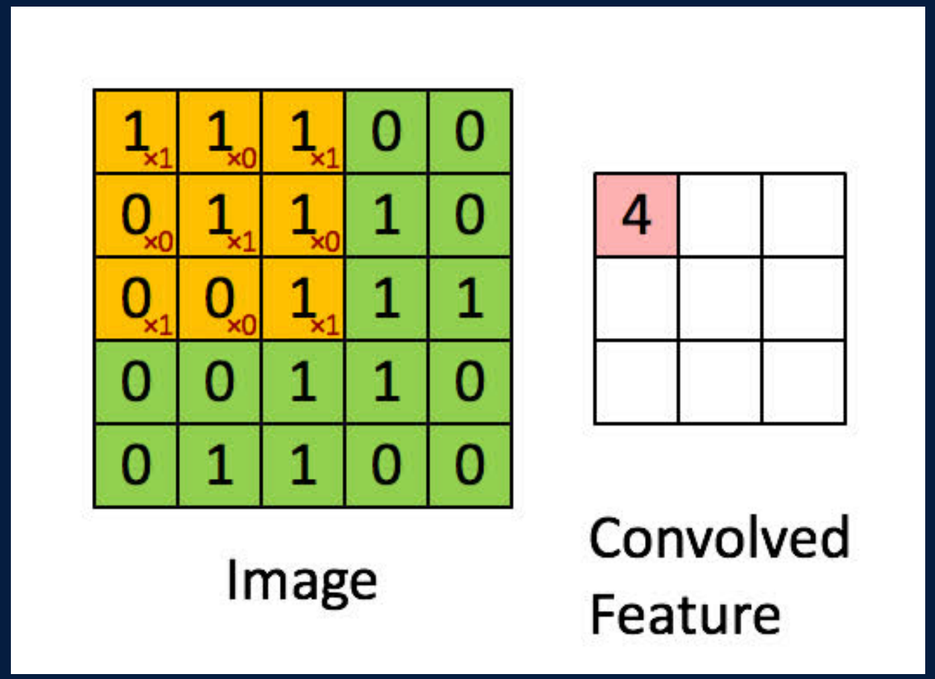
OUTPUT


3 Colour Channels
2
4
0
1
7
5
1
0
0
24
0
13
0
6
33
17
23
4
11
19
14
51
2
1
5
8
0
6
0
1
Height: 4 Units
(Pixels)
Width: 4 Units
(Pixels)

BASIC CONVOLUTION OPERATION

- It’s like a mathematical operation that blends two functions to produce a third.

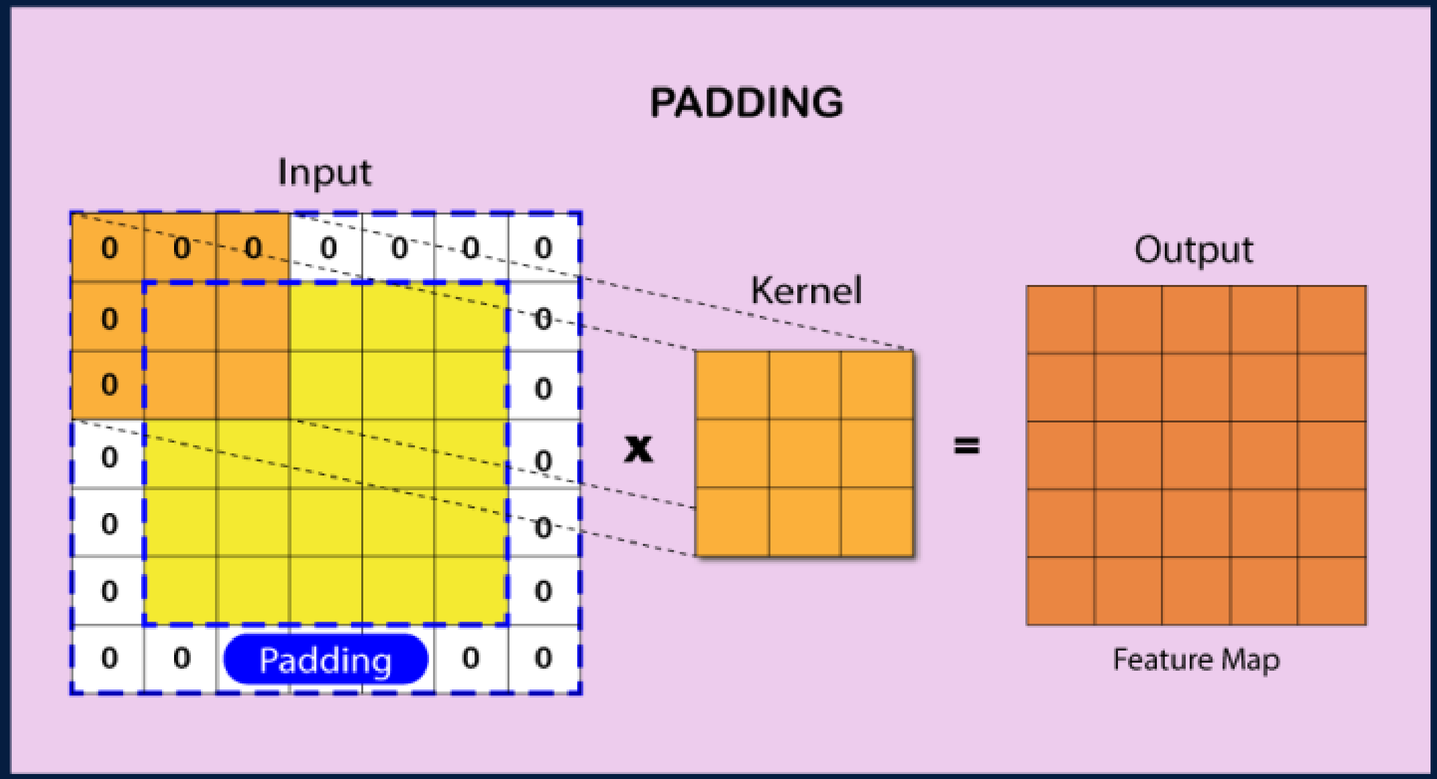
- How convolution works in CNNs: a small filter (also called kernel) slides over the input image, performing element-wise multiplication and summing the results to create a feature map.
- Stride: how much the filter moves in each step.



PADDING & STRIDE

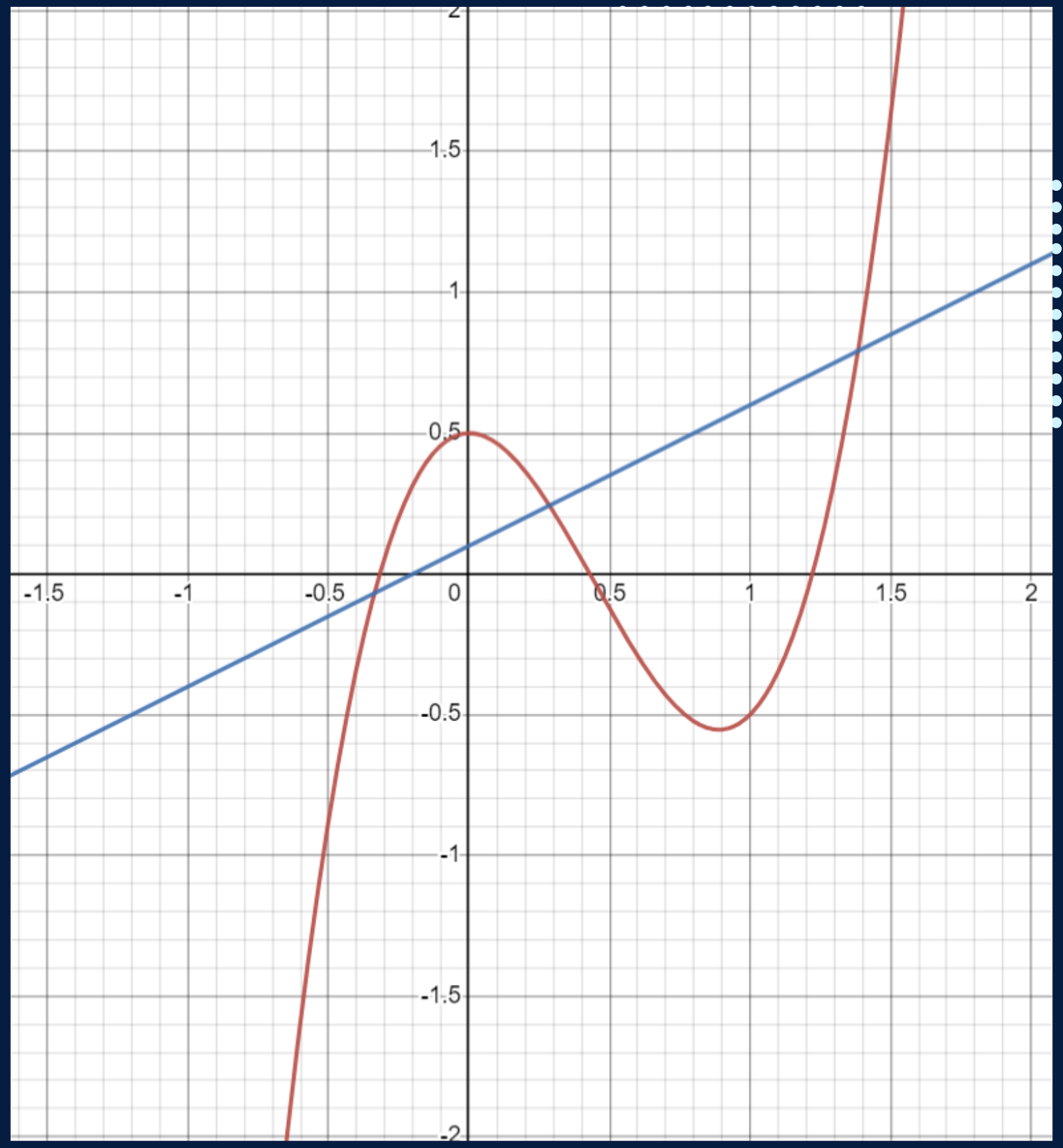
- Padding: adding extra pixels around the input image, which helps in preserving spatial dimensions after convolution.

- Stride: the step size with which the filter moves while performing convolution. Higher stride reduces the output dimensions.



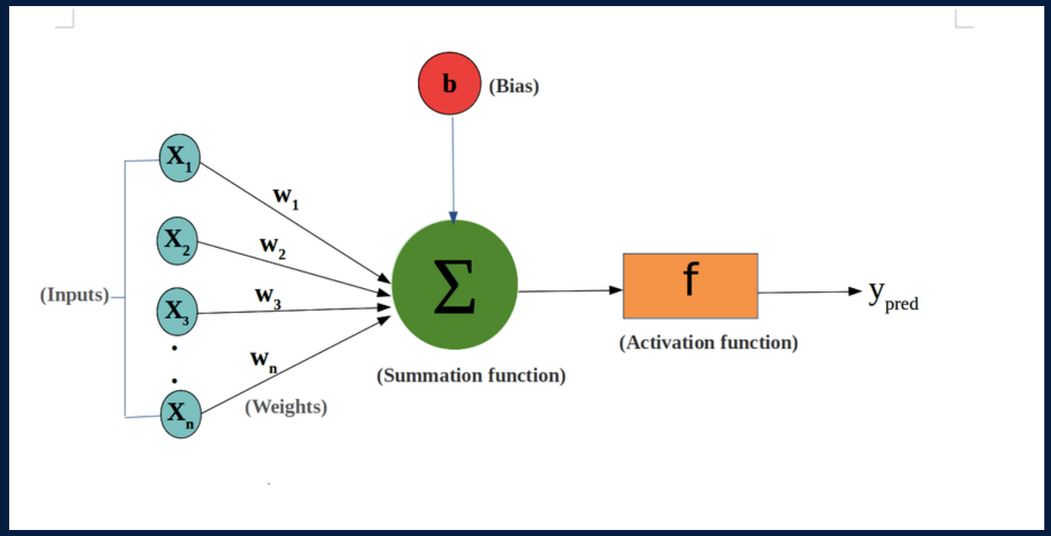
$$N_0(x) = -5x + -7.7$$
$$N_1(x) = -1.2x + -1.3$$
$$N_2(x) = 1.2x + 1$$
$$N_3(x) = 1.2x + -0.2$$
$$N_4(x) = 2x + -1.1$$
$$N_5(x) = 5x + -5$$

$$-1(-5x + -7.7)+$$
$$-1(-1.2x + -1.3)+$$
$$-1(1.2x + 1)+$$
$$1(1.2x + -0.2)+$$
$$1(2x + -1.1)+$$
$$1(5x + -5)$$



ACTIVATION FUNCTIONS

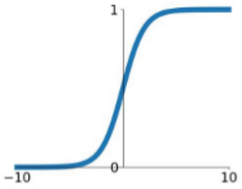
- These are non-linear functions applied element-wise after the convolution operation in each neuron.
- To introduce non-linearity, enabling the network to learn and represent complex relationships in the data.



Activation Functions

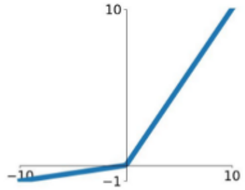
Sigmoid

$\sigma(x) = \frac{1}{1+e^{-x}}$



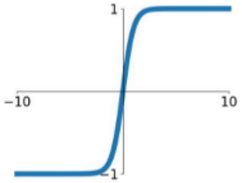
Leaky ReLU

$\max(0.1x, x)$



tanh

$\tanh(x)$

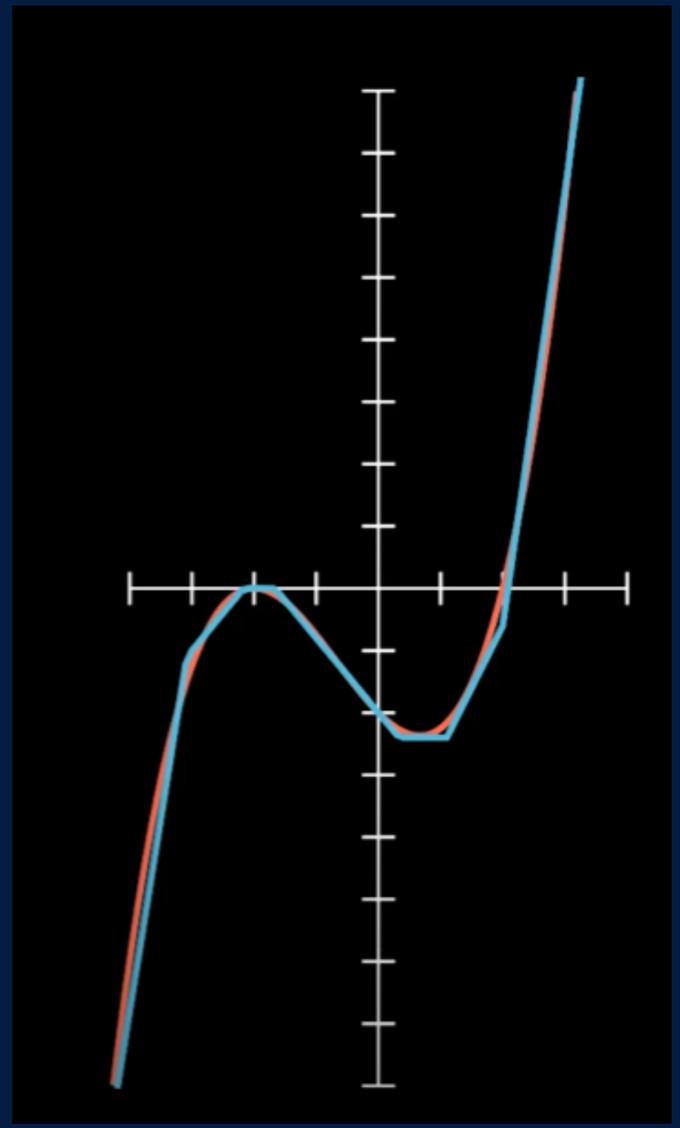


Maxout

$\max(w_1^T x + b_1, w_2^T x + b_2)$

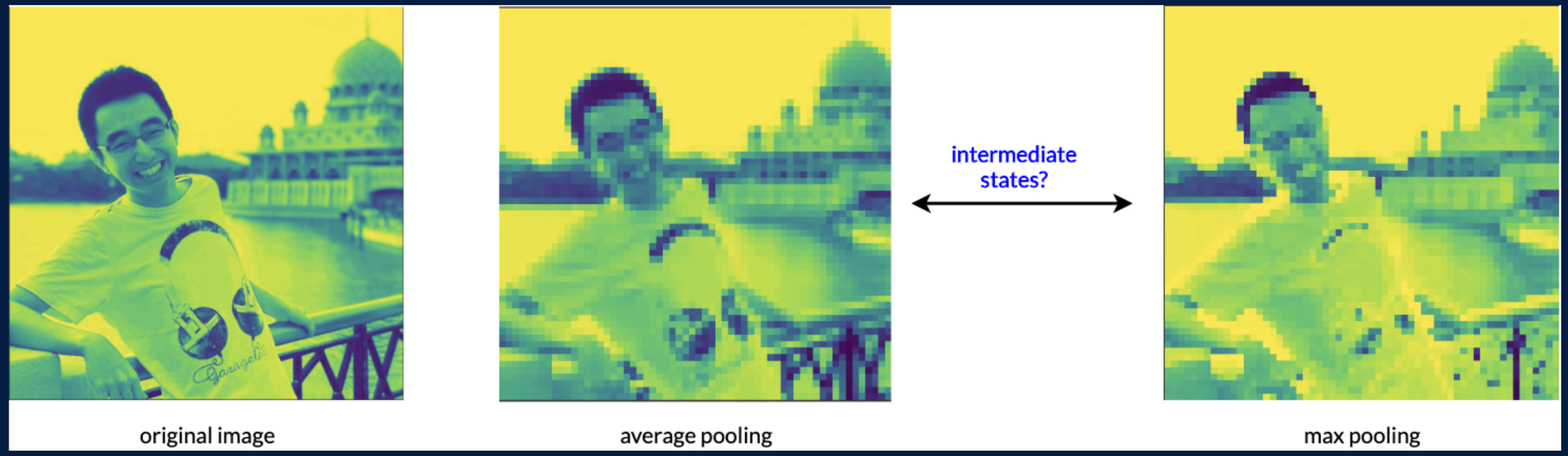
$$ReLU(x) = MAX(x, 0)$$

$$\begin{aligned}
&-max(-5x + -7.7, 0)+ \\
&-max(-1.2x + -1.3, 0)+ \\
&-max(1.2x + 1, 0)+ \\
&max(1.2x + -0.2, 0)+ \\
&max(2x + -1.1, 0)+ \\
&max(5x + -5, 0)
\end{aligned}$$



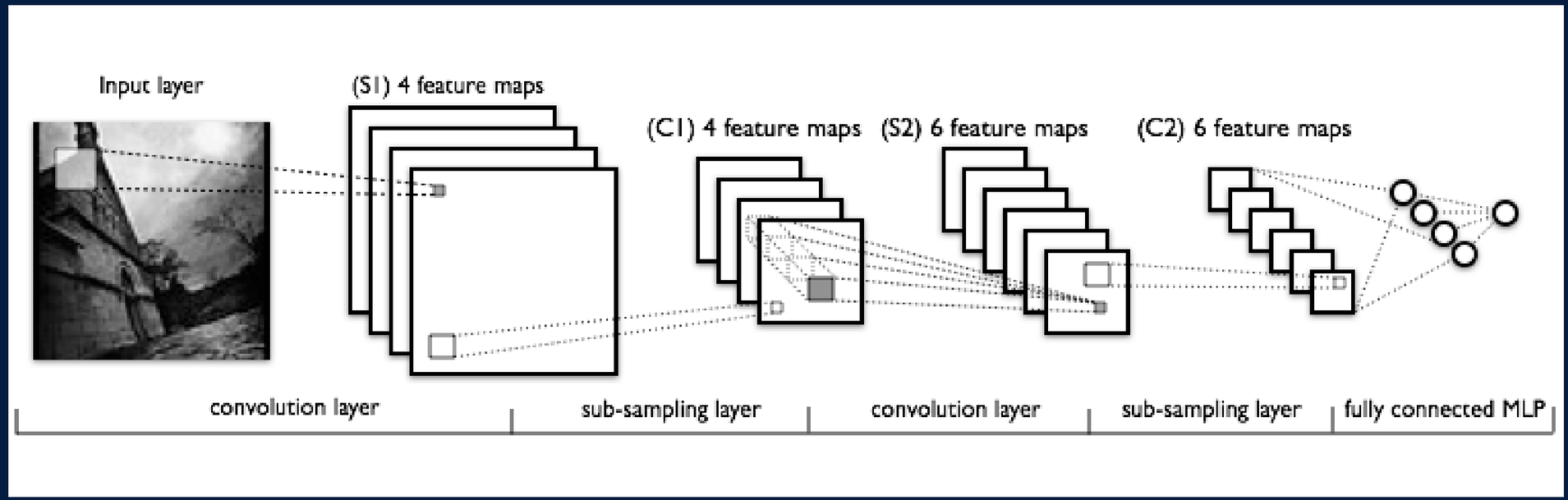
POOLING LAYERS
(MAX POOLING AND AVERAGE POOLING)

- These layers are used to reduce spatial dimensions and control overfitting.
- Max pooling: the maximum value within a small window (e.g., 2x2) is retained and the rest are discarded. It helps in preserving the most important features.
- Average pooling: the average value within a small window is calculated and used to downsample the feature map. It provides a smoother reduction in spatial dimensions.
- Pooling layers help in reducing computational complexity and making the network more robust to variations in input data.



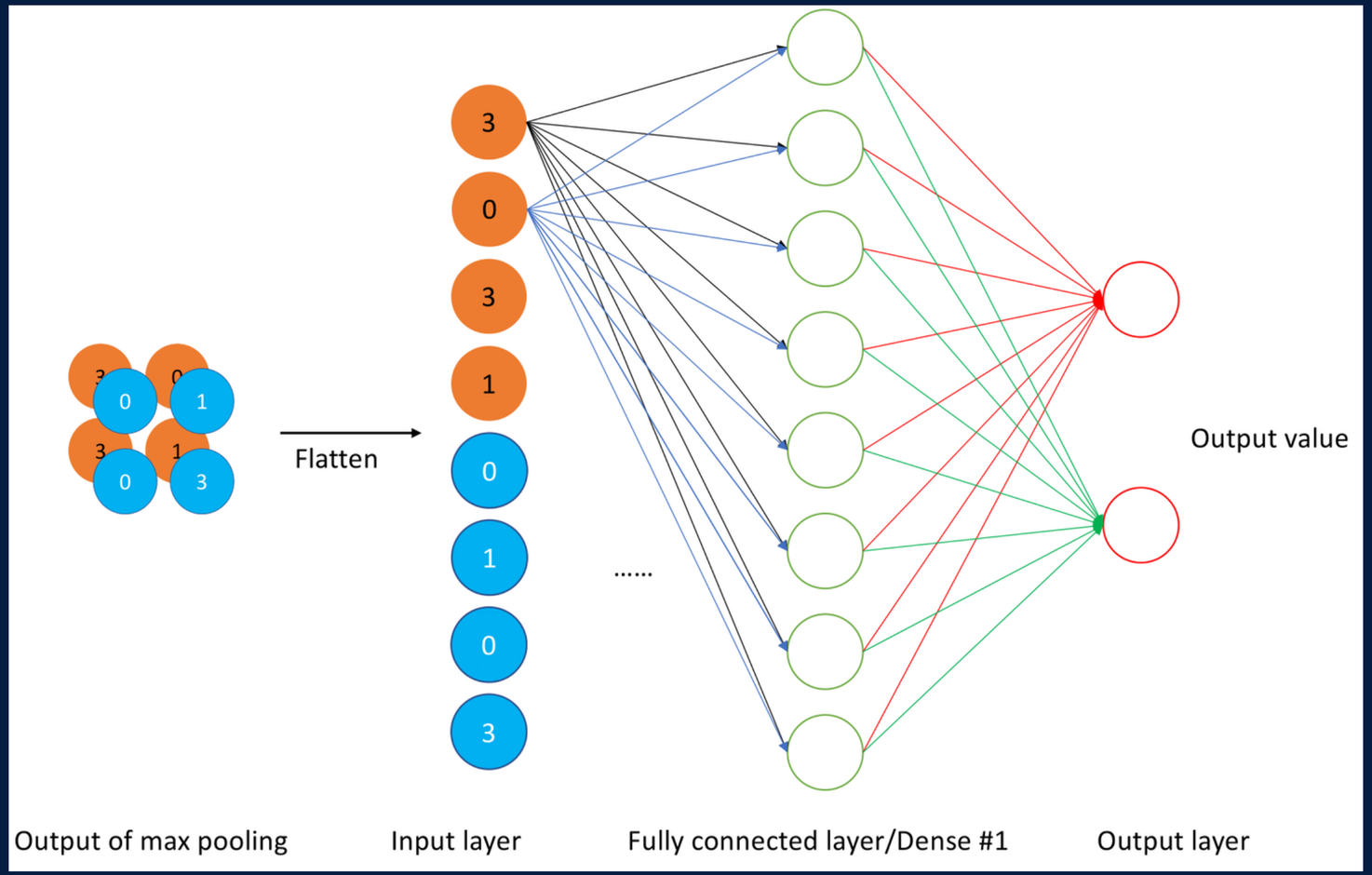
CONVOLUTIONAL LAYERS

- CNNs consist of multiple layers, with convolutional layers being the key building blocks.
- Each layer learns different features by using various filters. Early layers capture simple features like edges, while deeper layers capture complex features like textures and shapes.
- Layers closer to the output represent higher-level features relevant to the task.

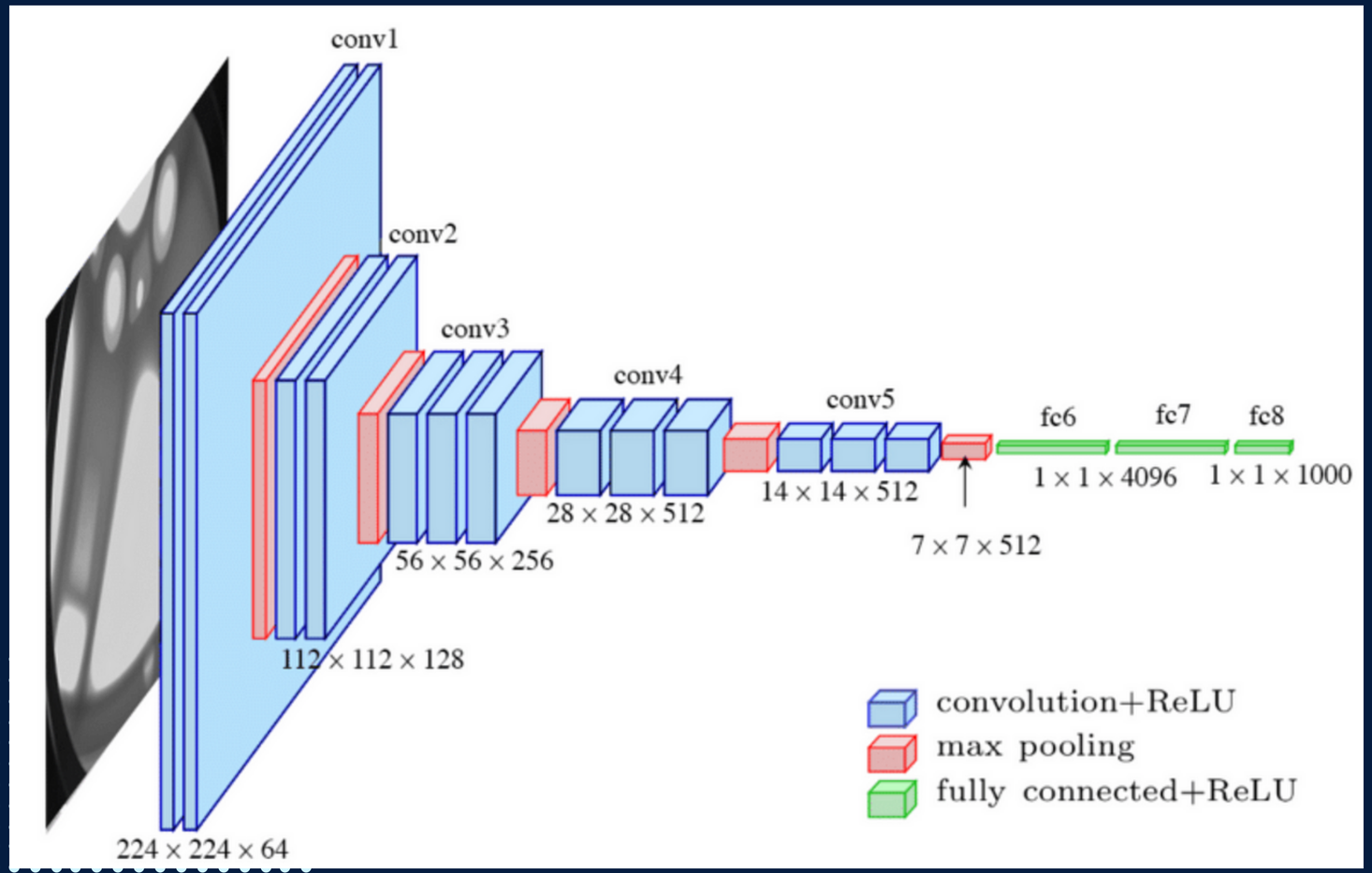


FULLY CONNECTED LAYERS

- These are densely connected layers where each neuron is connected to every neuron in the previous layer.
- They take the high-level features learned from earlier layers and combine them to make final predictions or classifications.
- Flattening step: converting the 2D or 3D output of convolutional and pooling layers into a 1D vector for input to the fully connected layers.



VGG 16(VISUAL GEOMETRY GROUP)

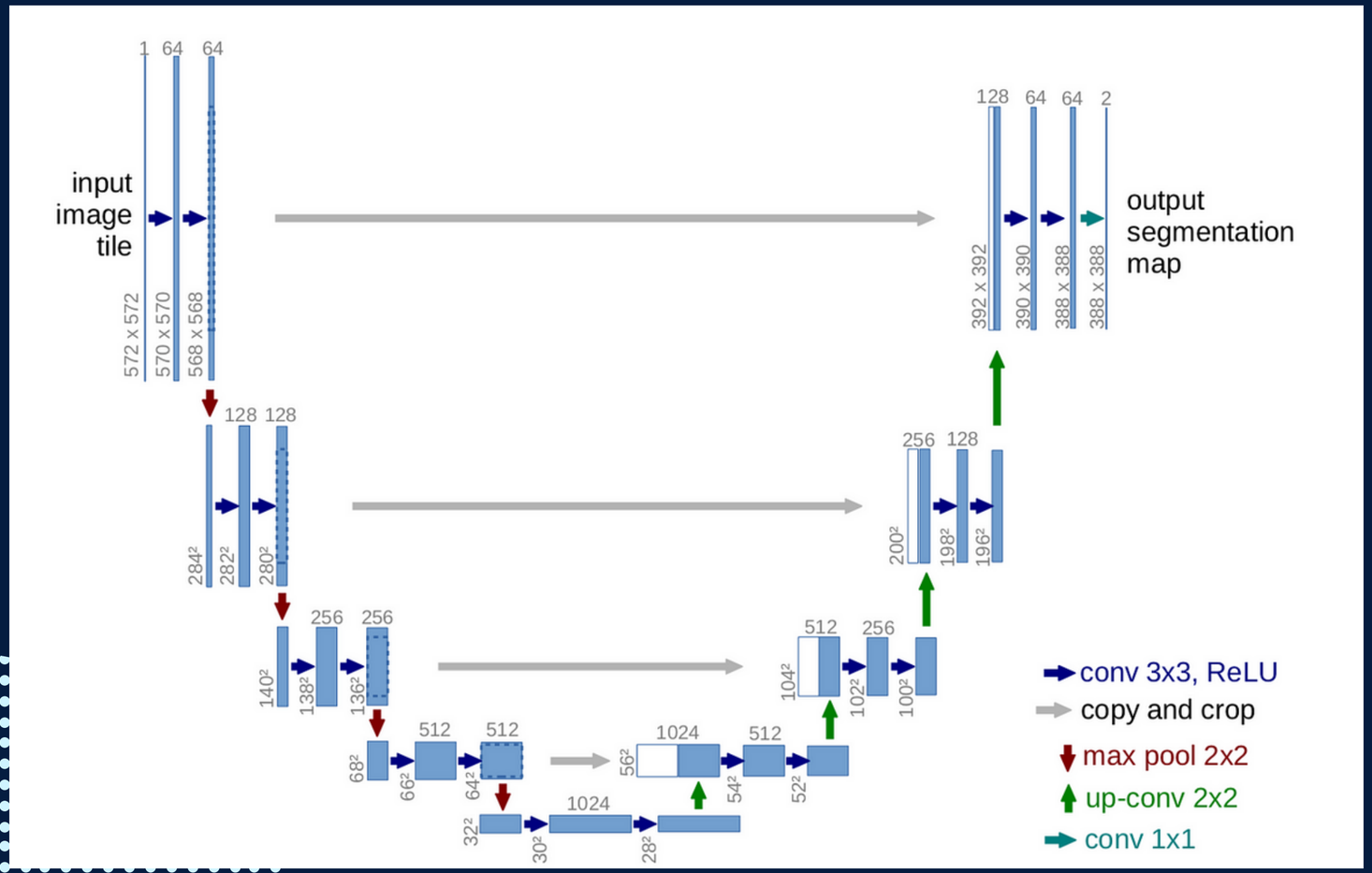


- This architecture is used in task like image classification like imageNet

- 16 in VGG16 symbolizes it contains 16 layers that is used for feature extracion

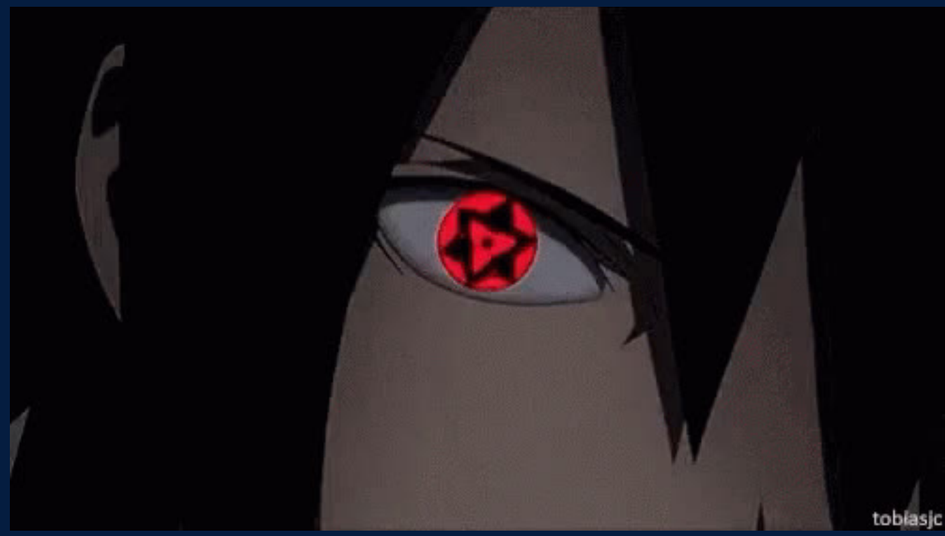
- The output layer has 1000 neurons, corresponding to the 1000 classes . It usually uses softmax activation for multi-class classification tasks.

U-Net



- 1.This architecture is used in task like image segmentation and deblurring task.
- 2. UNet comprises an encoder and a decoder.Encoder downsamples the image and decoder then upsamples these features to produce a segmentation mask.
- 3. Consists of skip connection to concatenates the features map from the encoder.

Let's look into some of the CODING stuff.



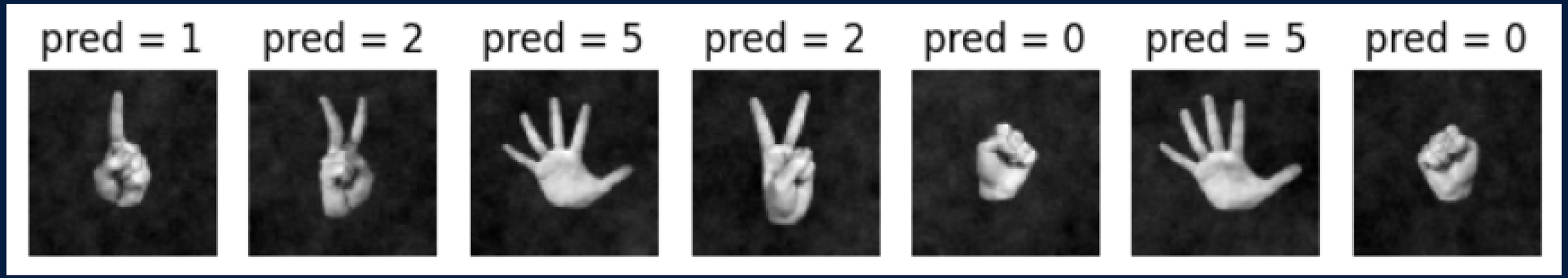
IMPORTANT LIBRARIES:

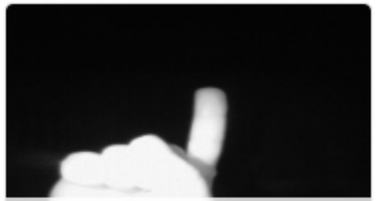
1. TensorFlow
2. Scikit
3. Numpy
4. CV2

THEORY
OF CNN.

HOW TO
IMPLEMENT
IT?

TASK
HAND GESTURE DETECTION



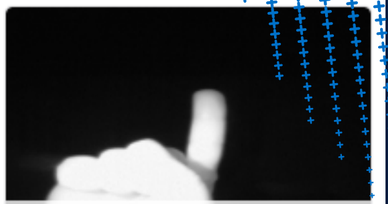


frame_05_06_0001.png
56.53 kB

frame_05_06_0002.png
57.53 kB

frame_05_06_0003.png
56.68 kB

frame_05_06_0004.png
57.72 kB



frame_05_06_0005.png
56.64 kB

frame_05_06_0006.png
57.56 kB

frame_05_06_0007.png
56.82 kB

frame_05_06_0008.png
57.78 kB

frame_05_06_0009.png
56.84 kB

frame_05_06_0010.png
57.8 kB

frame_05_06_0011.png
56.88 kB

frame_05_06_0012.png
57.75 kB

frame_05_06_0013.png
56.93 kB

frame_05_06_0014.png
58.03 kB

frame_05_06_0015.png
57.07 kB

EDA(Exploratory Data Analysis)

- EDA helps us analyse the entire dataset and summarise its main characteristics, like class distribution, size distribution, and so on. Visual methods are often used to display the results of this analysis.

EDA Techniques

- Visualisation
- Dealing with class imbalance
- Fill missing values(labels,features,etc)
- Normalization
- Pre-processing

Continued..

- Key points that we can consider from the dataset while making our model.
  - **Understand the problem and data**
  - **Research existing architectures**
  - **Develop a basline**
  - **Data augmentation**

ABOUT THE DATASET:

CONTENT:

1. The dataset contains 21600 grayscale images of left and right hand fingers.

```
print((x_train).shape)
```

```
(18000, 128, 128)
```

```
print((x_test).shape)
```

```
(3600, 128, 128)
```

2. Training set contains 18000 images and the rest is for testing.

LABELS:

00b11f87-98be-488b-a1de-2b10e6585936_5L.png

STEP 1: IMPORT DEPENDENCIES

```
import tensorflow as tf
from sklearn.model_selection import train_test_split
import numpy as np
import cv2
```

STEP 2: LOAD IMAGES AND LABELS

```
def load_data(directory):
    images = []
    labels = []
    for filename in os.listdir(directory):
        img_path = os.path.join(directory,filename)
        image = cv2.imread(img_path)
        images.append(image)
        label = int(filename[-6])
        labels.append(label)

    return images, labels
```

STEP 3: DIVISION INTO TESTING AND TRAINING

```
X,Y = load_data(...)
X = np.array(X)
Y = np.array(Y)
X_train,X_test,Y_train,Y_test = train_test_split(X,Y, test_size=0.3)
```

STEP 4: BUILD AND TRAIN THE MODEL

```
model models.Sequential()
# Adding a convolution layer
model.add(Conv2D(no_of_filters,kernel_size,padding,strides))
# Max pooling
model.add(MaxPooling2D((Pool_size,padding,strides)))
# Flattening layer
model.add(Flatten())
# Dense layer
model.add(Dense())
# Dense layer for classifaction
model.add(Dense(no_of_classes,activation='softmax'))
# Compilation
model.compile(optimizer,loss,metrics=['accuracy'])
history = model.fit(train_dataset, epochs, validation_data)
```

Leaders:

- Aniruddh Pramod
- Roy Shivam
- Priyanshu Tiwari
- Shlok Mishra

Thank you!

Secretaries:

- Vishal Kumar
- Debarbita Dash
- Aditya Prakash