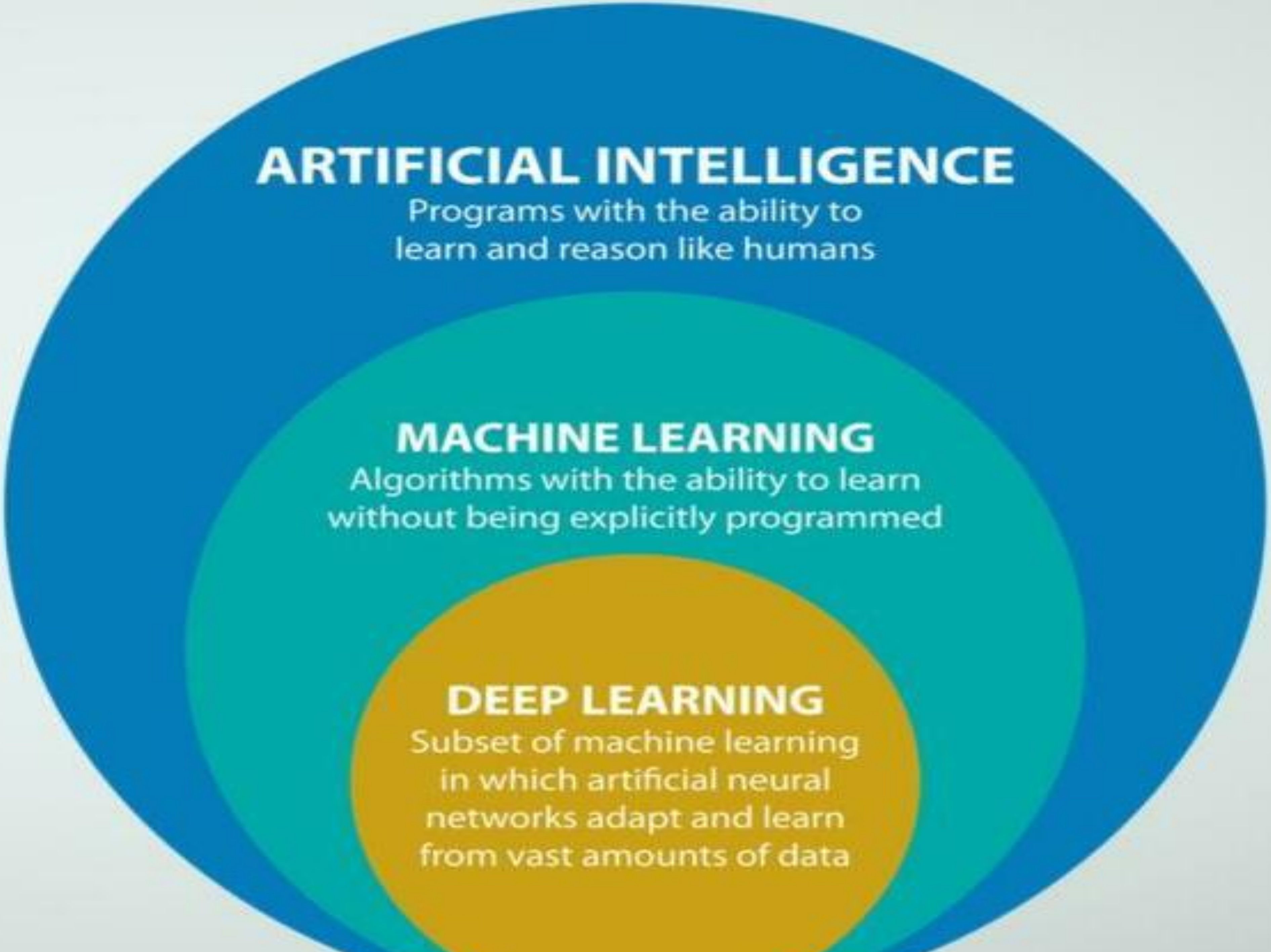
DEEP LEARNING

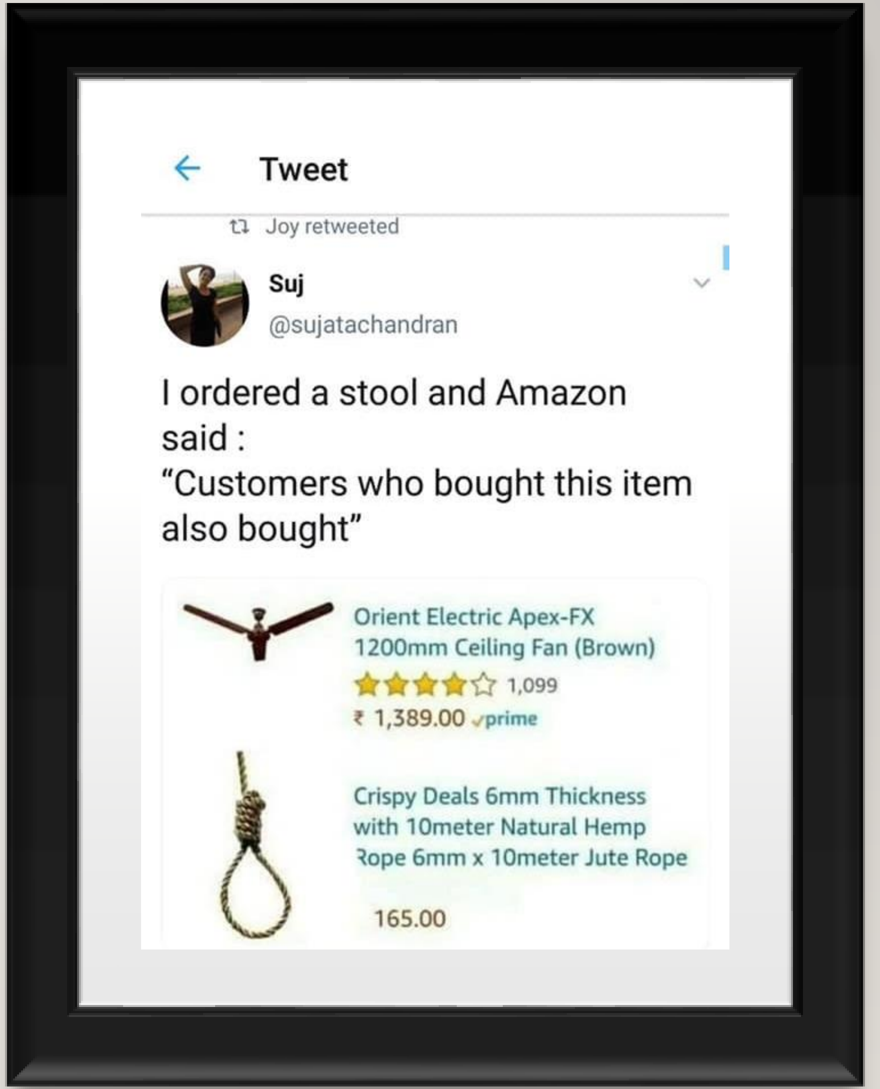
AGENDA

- INTRODUCTION TO DEEP LEARNING
- WHY DEEP LEARNING?
- WHY DEEP LEARNING NOW?
- INTRODUCTION TO CNN
- BASICS OF CNN
- DIGIT RECOGNITION USING LENET


ARTIFICIAL INTELLIGENCE
Programs with the ability to learn and reason like humans
MACHINE LEARNING
Algorithms with the ability to learn without being explicitly programmed
DEEP LEARNING
Subset of machine learning in which artificial neural networks adapt and learn from vast amounts of data

MACHINE LEARNING USE CASE

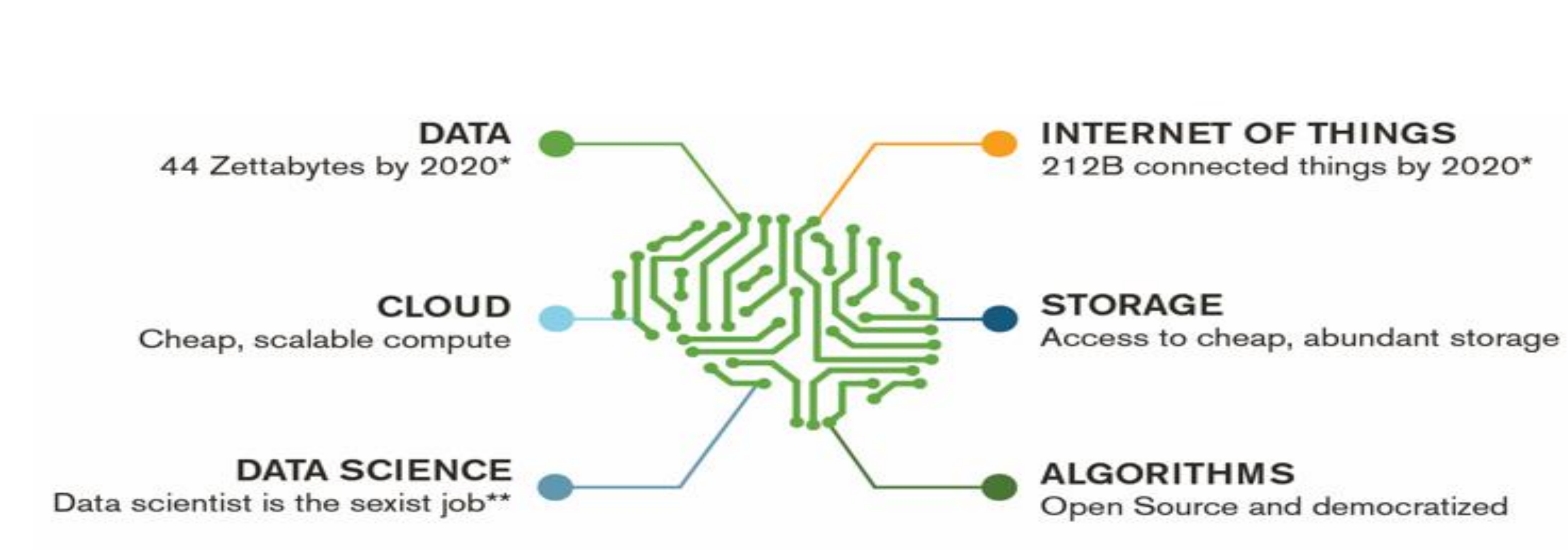
- Amazon uses machine learning to improve user experiences and recommendation based on their usage
- Netflix uses machine learning to suggest content to users



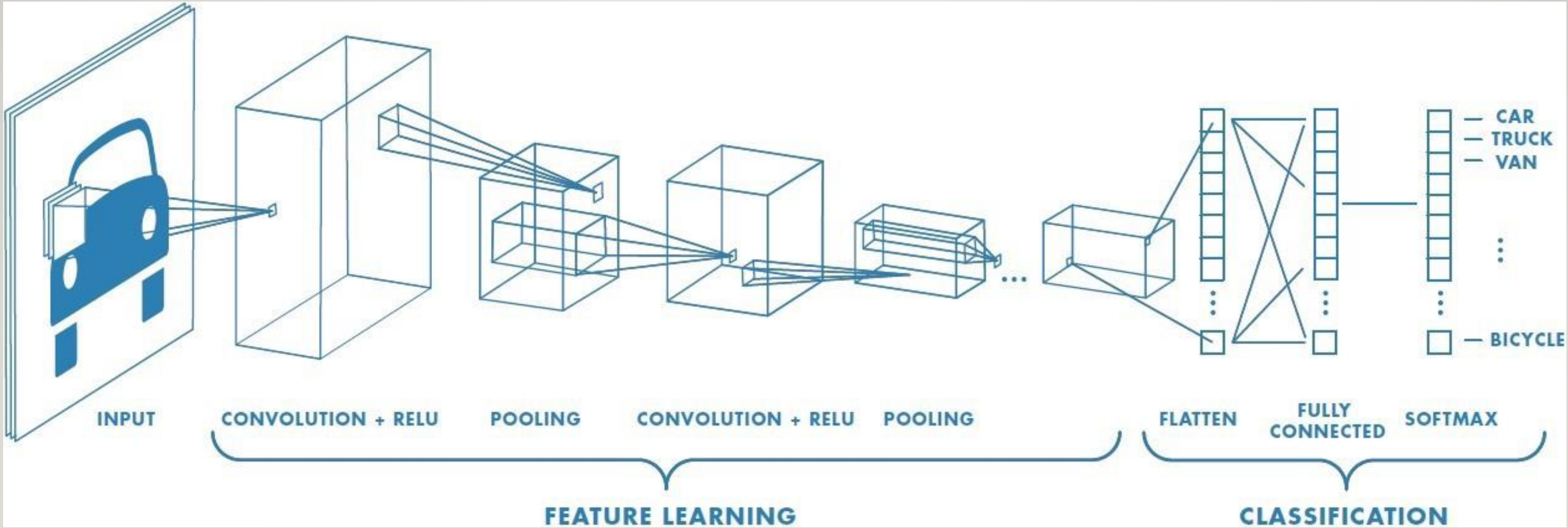
WHY DEEP LEARNING?

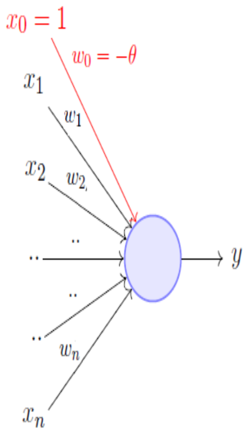
- Speech Recognition
- Computer Vision
- Natural Language Processing

WHY DEEP LEARNING NOW?



TYPICAL CONVOLUTIONAL NEURAL NETWORK





A more accepted convention,

$$y = 1 \quad if \sum_{i=0}^{n} w_i * x_i \geq 0$$

$$= 0 \quad if \sum_{i=0}^{n} w_i * x_i < 0$$

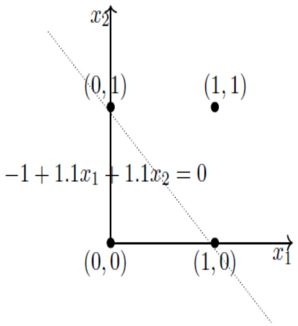
$where, \quad x_0 = 1 \quad and \quad w_0 = -\theta$

x_1	x_2	OR	
0	0	0	$w_0 + \sum_{i=1}^{2} w_i x_i < 0$
1	0	1	$w_0 + \sum_{i=1}^{2} w_i x_i \geq 0$
0	1	1	$w_0 + \sum_{i=1}^{2} w_i x_i \geq 0$
1	1	1	$w_0 + \sum_{i=1}^{2} w_i x_i \geq 0$

$$w_0 + w_1 \cdot 0 + w_2 \cdot 0 < 0 \implies w_0 < 0$$
$$w_0 + w_1 \cdot 0 + w_2 \cdot 1 \geq 0 \implies w_2 > -w_0$$
$$w_0 + w_1 \cdot 1 + w_2 \cdot 0 \geq 0 \implies w_1 > -w_0$$
$$w_0 + w_1 \cdot 1 + w_2 \cdot 1 \geq 0 \implies w_1 + w_2 > -w_0$$

One possible solution is

$w_0 = -1, \; w_1 = 1.1, \; w_2 = 1.1$



INTRODUCTION TO CNN: PERCEPTRON

PERCEPTRON LEARNING ALGORITHM

$P \leftarrow inputs \quad with \quad label \quad 1;$

$N \leftarrow inputs \quad with \quad label \quad 0;$

Initialize $\mathbf{w} = [w_1, w_2, \ldots, w_N]$ randomly;

while $!convergence$ **do**

  Pick random $\mathbf{x} \in P \cup N$;

  if $\mathbf{x} \in P \quad and \quad \sum_{i=0}^{n} w_i * x_i < 0$ **then**

    $\mathbf{w} = \mathbf{w} + \mathbf{x}$;

  end

  if $\mathbf{x} \in N \quad and \quad \sum_{i=0}^{n} w_i * x_i \geq 0$ **then**

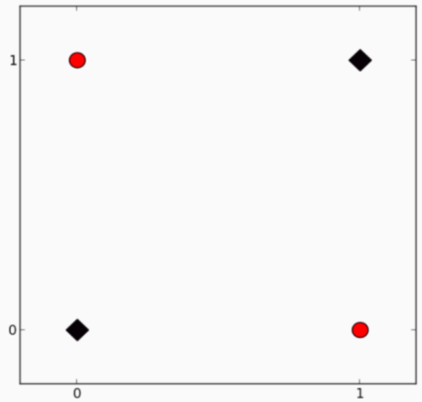
    $\mathbf{w} = \mathbf{w} - \mathbf{x}$;

  end

end

//the algorithm converges when all the inputs are classified correctly

Input		Output
x	y	x⊕y
0	0	0
0	1	1
1	0	1
1	1	0

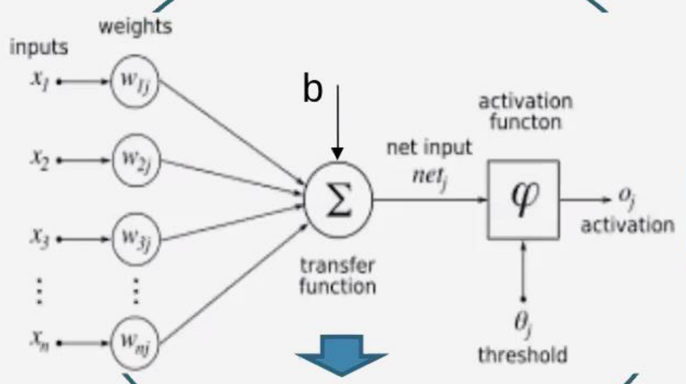


LIMITATIONS OF PERCEPTRON


WE NEED TO GO
DEEPER

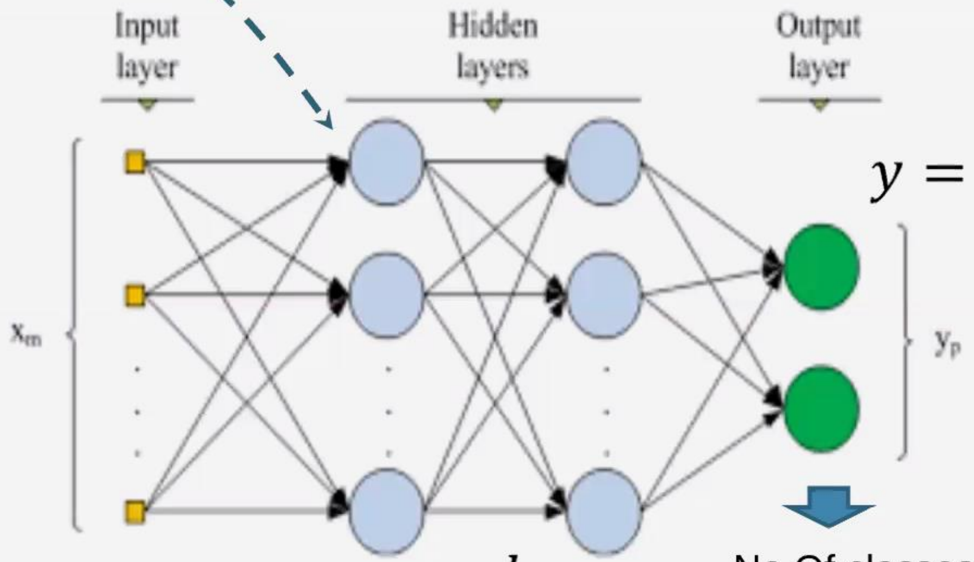
CNN – Neural Net classifier

Single Neuron architecture



Y += Wi * Xi + b :MAC operation

MLP: Fully Connected Neural Net



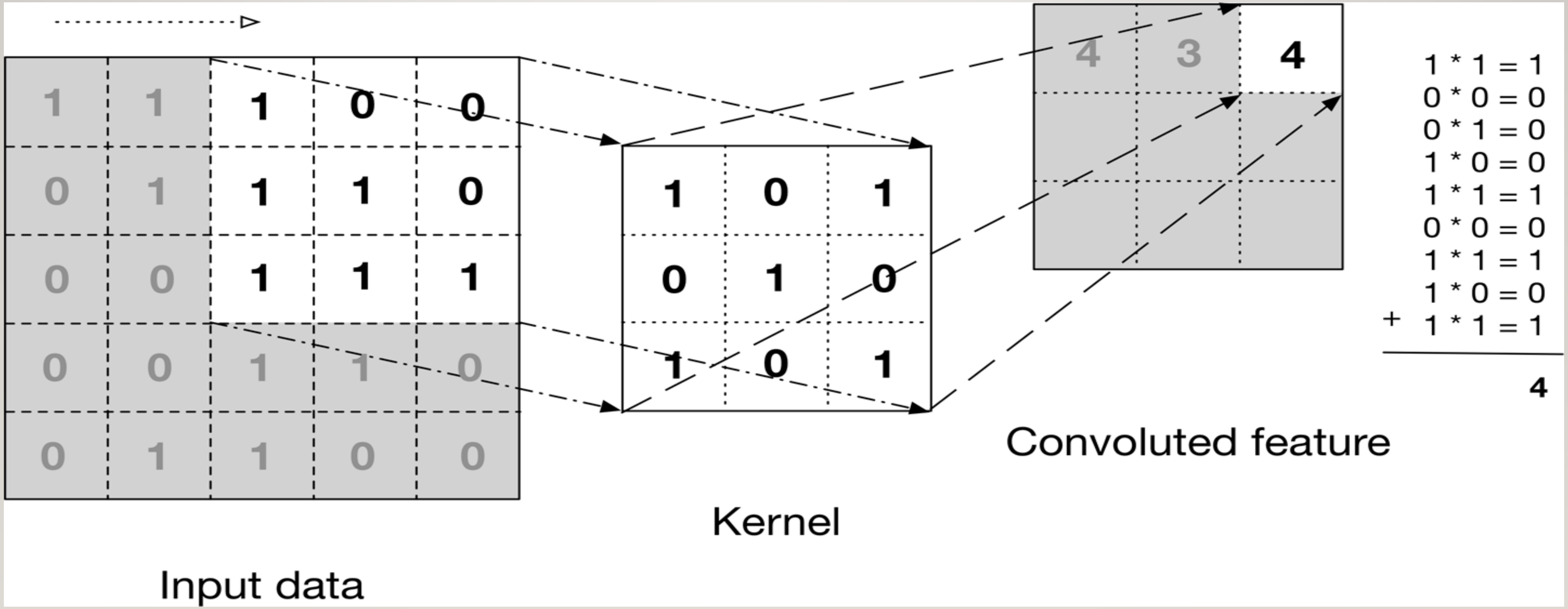
$$y = f(x)$$

No Of classes

l = 1 W_{ij}^{l} l = 2

Activation function = Nonlinear function => exp, tanh etc

CONVOLUTION

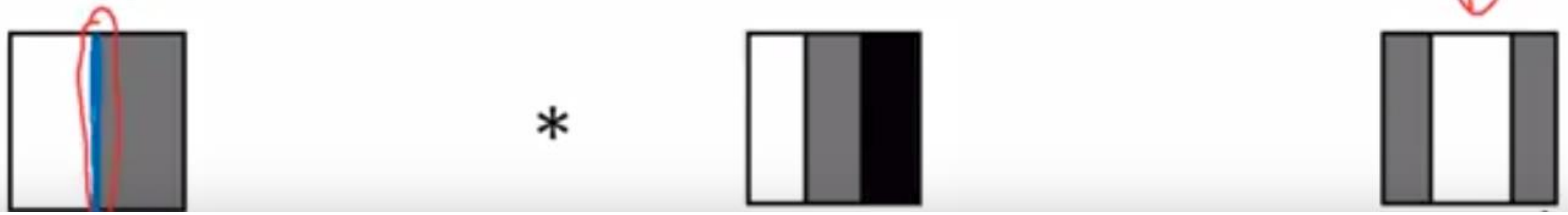


TERMINOLOGY IN CNN

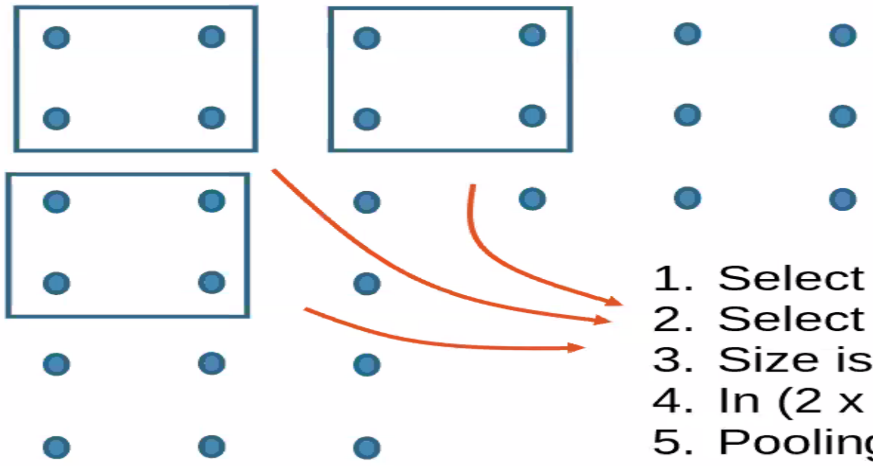
1. Kernel/filter/Weights: Learnable parameters of our model
2. Stride: how many cells to move
3. Bias: how far off our predictions are from real values
4. Feature Maps: output of Intermediator layers

Vertical edge detection

$$\underbrace{\begin{bmatrix} 10 & 10 & 10 & 0 & 0 & 0 \\ 10 & 10 & 10 & 0 & 0 & 0 \\ 10 & 10 & 10 & 0 & 0 & 0 \\ 10 & 10 & 10 & 0 & 0 & 0 \\ 10 & 10 & 10 & 0 & 0 & 0 \\ 10 & 10 & 10 & 0 & 0 & 0 \end{bmatrix}}_{6\times 6} * \underbrace{\begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix}}_{3\times 3} = \underbrace{\begin{bmatrix} 0 & 30 & 30 & 0 \\ 0 & 30 & 30 & 0 \\ 0 & 30 & 30 & 0 \\ 0 & 30 & 30 & 0 \end{bmatrix}}_{4\times 4}$$



POOLING: DOWN SAMPLING



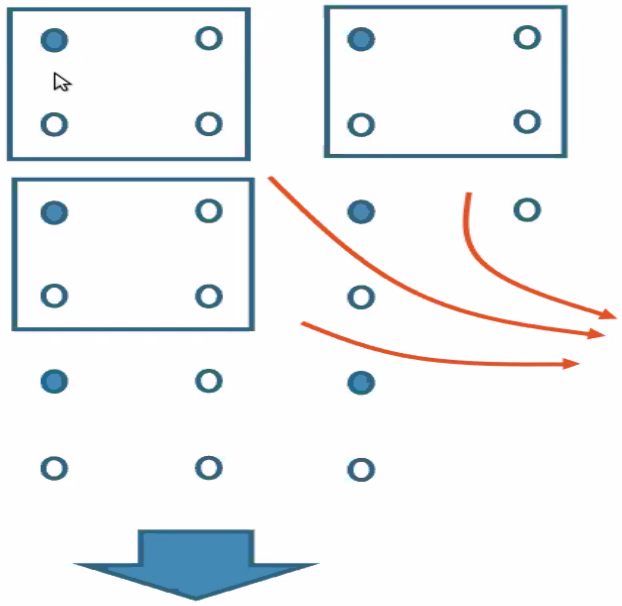
Scan Right by H-stride

1. Select max out of 4 pixel in Max Pooling
2. Select average of all 4 pixel in Average Pooling
3. Size is typically (2 x 2) but it can be any integer
4. In (2 x 2) stride is 2
5. Pooling reduces computations in CNN model and handles dimensionality problem

Scan Down by V-stride

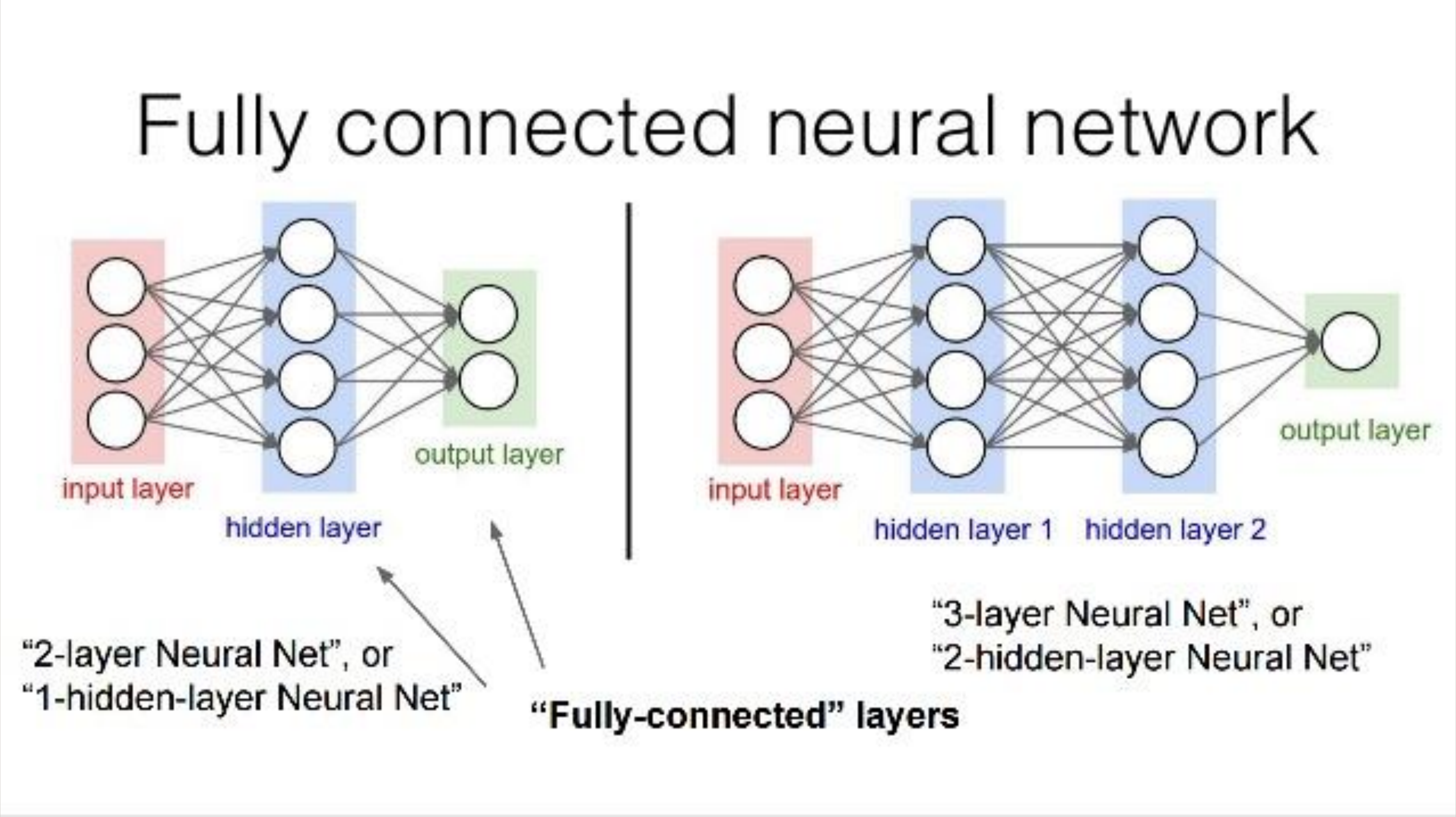
NEAREST UP SAMPLING POOLING

2x2 Nearest Up-sampling



1. Replicate 3 zero pixel by original pixel
2. Size is typically (2 x 2) but it can be any integer
3. Up-sampling:
   - Improves resolution for better features
   - Required in adder and concatenate blocks to add from previous outputs of higher resolutions
4. Bilinear up-sampling can be used for better quality at the cost of increase computations

Scan Down by V-stride


Fully connected neural network
input layer
hidden layer
output layer
input layer
hidden layer 1
hidden layer 2
output layer
“2-layer Neural Net”, or
“1-hidden-layer Neural Net”
“Fully-connected” layers
“3-layer Neural Net”, or
“2-hidden-layer Neural Net”

POPULAR FRAMEWORKS IN PYTHON FOR DEEP LEARNING/MACHINE LEARNING

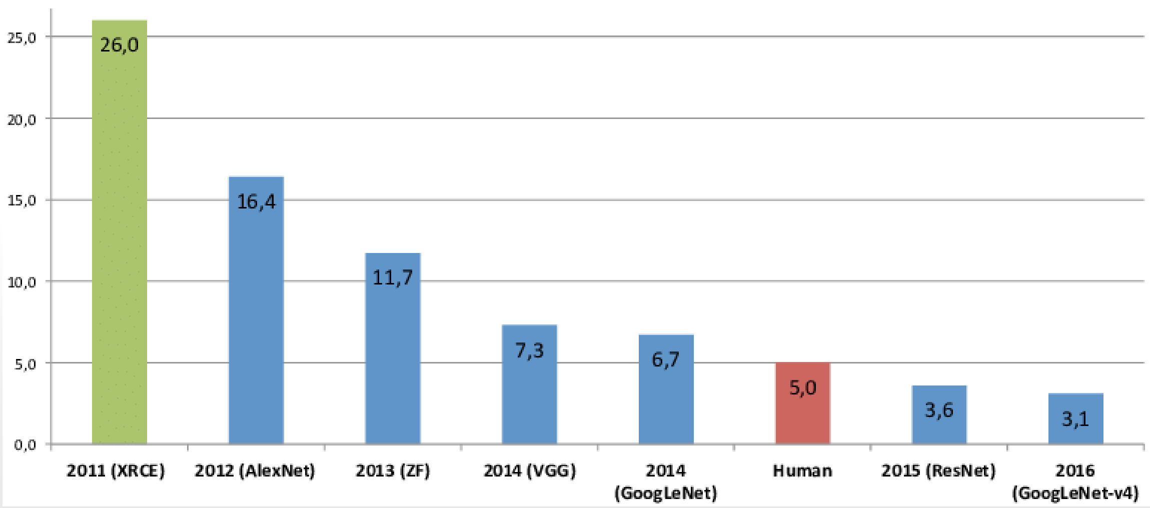
Caffe2

PYTORCH

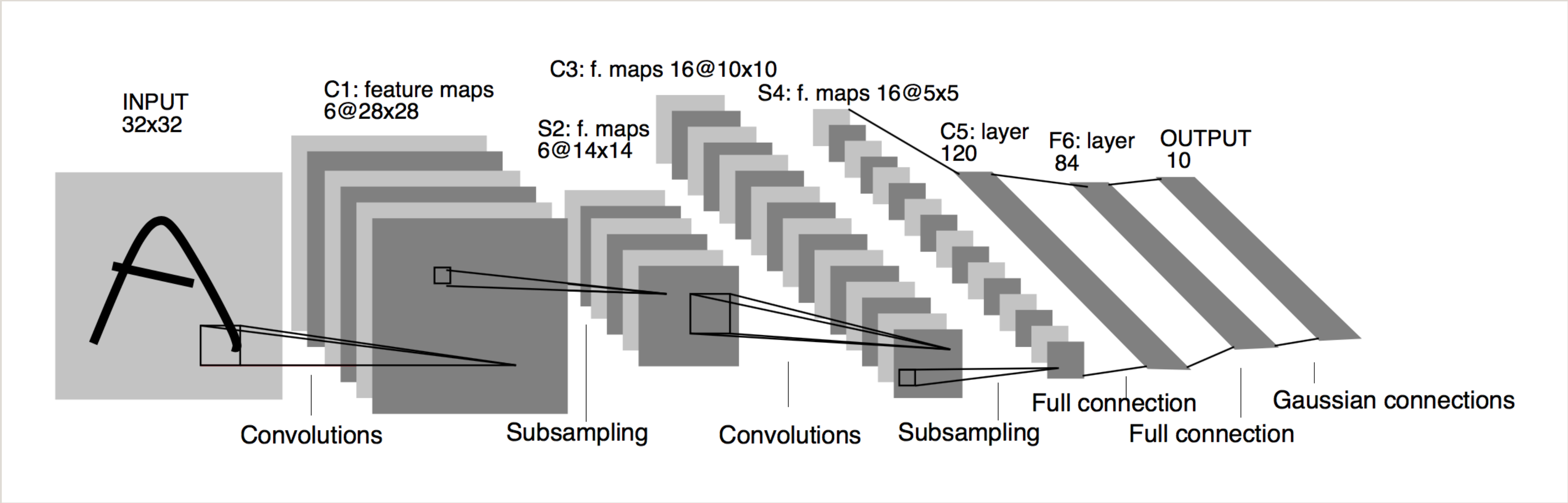
Caffe

TensorFlow

Keras



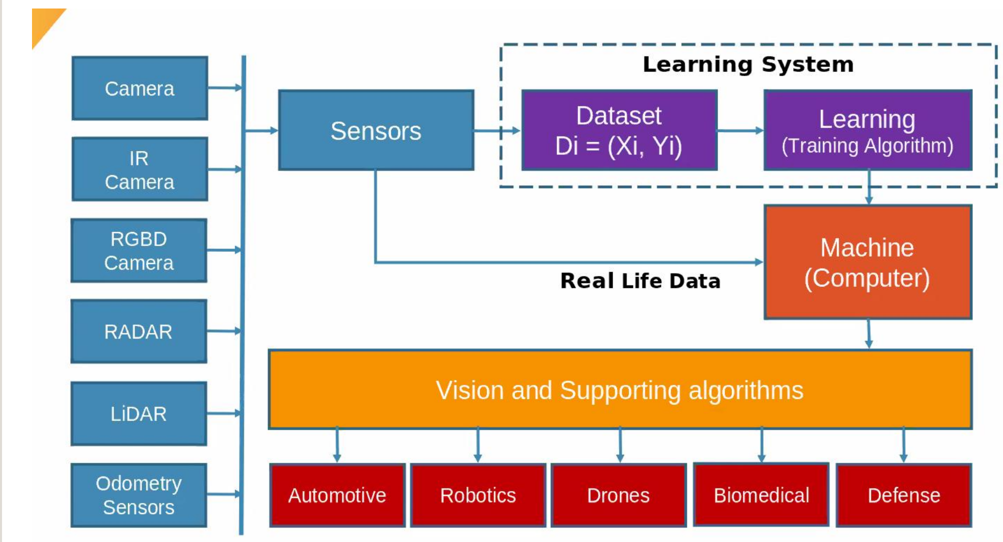
IMAGENET COMPITITION



LENET ARCHITECTURE FOR DIGIT RECOGNITION

HOW TO APPROACH A IMAGE CLASSIFICATION PROBLEM?

- Train Dataset
- Training algorithm
- Loss function
- Optimizer
- Vision algorithm
- Test Dataset



POPULAR DATASETS FOR COMPUTER VISION

- MNIST
- IMAGENET
- COCO
- CIFAR-10
- VISUALQA

PREPARING DATASET USING KERAS

- We choose MNIST dataset for digit recognition.
- We divide the training dataset into two sets
  - Train Dataset
  - Test Dataset

```
def load_from_mnist(self):
    mnist=keras.datasets.mnist
    (x_train, x_train_label),(x_test, x_test_label) = mnist.load_data()
    x_train = x_train.reshape(x_train.shape[0], 28, 28, 1)
    x_test = x_test.reshape(x_test.shape[0], 28, 28, 1)
    return x_test,x_test_label,x_train,x_train_label
```

WRITING A SEQUENTIAL MODEL

- We create a sequential model and add layer by layer.

```
def gen_model(self):
    self.model.add(keras.layers.Conv2D(filters=6, kernel_size=5, strides=(1, 1), padding="same", input_shape=(28, 28, 1),
                                  data_format="channels_last", activation="relu"))
    self.model.add(keras.layers.MaxPooling2D(pool_size=(2, 2), strides=(2, 2)))
    self.model.add(keras.layers.Conv2D(filters=16, kernel_size=5, strides=(1, 1), padding="valid", activation="relu"))
    self.model.add(keras.layers.MaxPooling2D(pool_size=(2, 2), strides=(2, 2)))
    self.model.add(keras.layers.Flatten())
    self.model.add(keras.layers.Dense(120, input_shape=(400,), activation='relu'))
    self.model.add(keras.layers.Dense(84, input_shape=(120,), activation='relu'))
    self.model.add(keras.layers.Dense(units=10, activation='softmax'))
```

TRAIN MODEL

- We need to select the optimizer and loss function based on our requirements.

```
def train_model(self):
    self.model.compile(loss='categorical_crossentropy',optimizer='adam',metrics=['accuracy'])
#     self.model.compile(loss='mean_squared_error', optimizer='sgd', metrics=['accuracy'])load_from_mnist
    self.model.fit(self.x_train, self.x_train_label, epochs=self.no_of_epochs, batch_size=self.batch_size)
```

```
def predict_output(self):
    # loss_and_metrics = model.evaluate(x_test, x_test_label,batch_size=32)
    classes = self.model.predict(self.x_test, batch_size=self.batch_size)
    classes = classes.argmax(axis=1)
    for row in range(0, 3):
        plt.title("label=%s" % classes[row])
        plt.imshow(np.reshape(self.x_test[row], (28, 28)), cmap='gray')
        plt.show()
```

EVALUATE/TEST MODEL

ANY QUESTIONS?

Contact:

G Venkata Durga Rao

9000486749

venkatadj@gmail.com