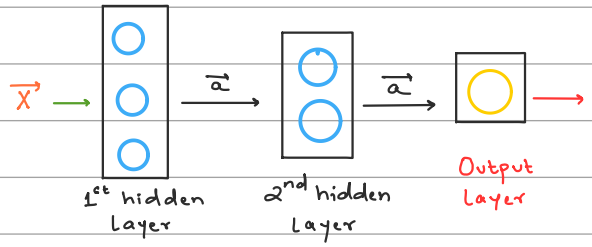
Week 1

○ How does a neural network work ?

- It works on intuition of how biological neurons work in our brain.
- There's a neuron which, on the basis of input, predicts if there is a possibility of a certain output.
- A neural network is build on top of multiple layers which gets an input $\vec{x}$ and gives out an output $\vec{a}$ to another layer.



○ Neural Network Architecture

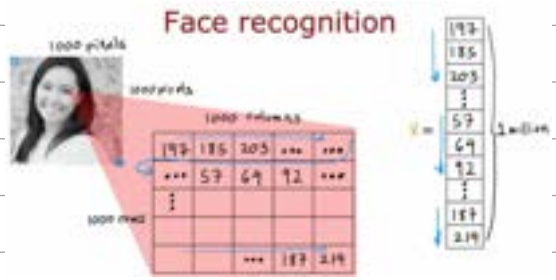
- It's the process of selecting the amount of hidden layers and the number of features in each of the layer.
- It has a huge impact in the efficiency & performance of the Neural Network.

○ Application of a Neural Network

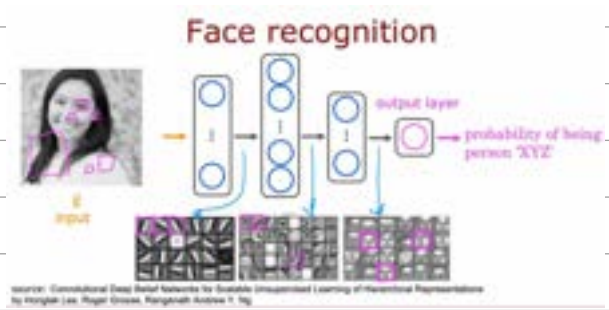
- Neural networks are used in speech recognition, face detection etc.

- let's understan how does face recognition works:-

1. The image is converted into a matrix of values of brightness in different pixels.
2. Then the matrix is converted into a vector and given as an input to the input layer of the Neural Network



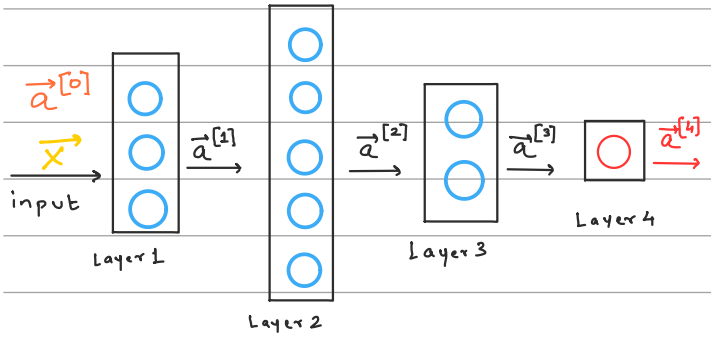
3. Then the neural network starts solving a puzzle by matching edges and forming the sub section of the image
4. As the size of the subsection increases, we recognize the image.



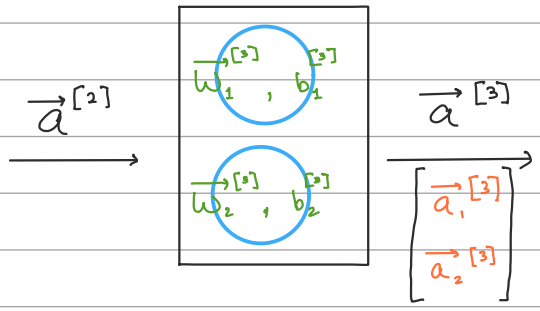
5. Then the output layer predicts if it is a person or not.

★ Google uses this to sort out your friends faces in Google Photos.

○ Neural Network structure.



- Let's understand what's happening in Layer 3



$$\vec{a}_1^{[3]} = g\left(\vec{w}_1^{[3]} \cdot \vec{a}^{[2]} + b_1^{[3]}\right)$$

$$\vec{a}_2^{[3]} = g\left(\vec{w}_2^{[3]} \cdot \vec{a}^{[2]} + b_2^{[3]}\right)$$

- Here we use the sigmoid function to calculate 0 or 1 in every computation

$$g(z) = \frac{1}{1 + e^{-z}}$$

- Hence the $\vec{a}^{[3]}$ is calculated and given as an input to the next layer ie Layer 4

○ Notation for Activation

$$a_j^{[L]} = g\left(\vec{w}_j^{[L]} \cdot \vec{a}^{[L-1]} + b_j^{[L]}\right)$$

$[L]$ = Layer of neuron

J = position of neuron in current layer

g = Sigmoid or Activation Function

$\vec{w}, b$ = Parameters of the neuron

a = Activation of neuron.

○ Handling arrays in Tensorflow

- Unlike 1-D vectors in regression & classification, Tensorflow requires a Matrix because is build to use huge data sets.

○ Building a neural network Architecture

★ Dense → used to create a layer

★ Sequential → Used to create a model by giving layers as an input

★ Compile → Function that runs the the previously build model

★ fit → Function that takes training data x & y as input and puts it in the Neural Network

★ Predict → Finally predicts the activation

☆ Is there a path to AGI ?

- We have made significant progress in the field of AI, but that doesn't mean that we're on our way to AGI

- AI can be divided in two section

AI

ANI — Artificial Narrow Intelligence

- Smart speaker
- Self-driving car
- Web search
- AI in factories

AGI — Artificial General Intelligence

☆ Can do anything that a human brain can do.

☆ Mapping in our brain very uncertain hence people suspect that one part of brain is capable of doing multiple tasks

☆ The One Learning Algorithm hypothesis State that one part of brain, if feeded with a different type of data, can be used to produce the same output.

- Seeing with your tongue
- Human echolocation (sonar)
- Haptic belt : Direction sense
- Implantin a 3rd eye.

○ Understanding Vectorization

- Dot products:

vector - vector multiplication

$$z = \vec{a} \cdot \vec{w} \qquad z = \vec{a}^T \vec{w}$$

is same as

where $\vec{a}^T$ is the transpose of $\vec{a}$.

☆ Whenever you see a Transpose, think of a Row

☆ Whenever you see a Matrix, think of a Column

☆ Rule for Matrix Multiplication

$$\begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \quad \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

Columns = Rows

○ Numpy function for Matrix Multiplication.

```
Z = np.matmul(AT, W)
```

or

```
Z = AT @ W
```