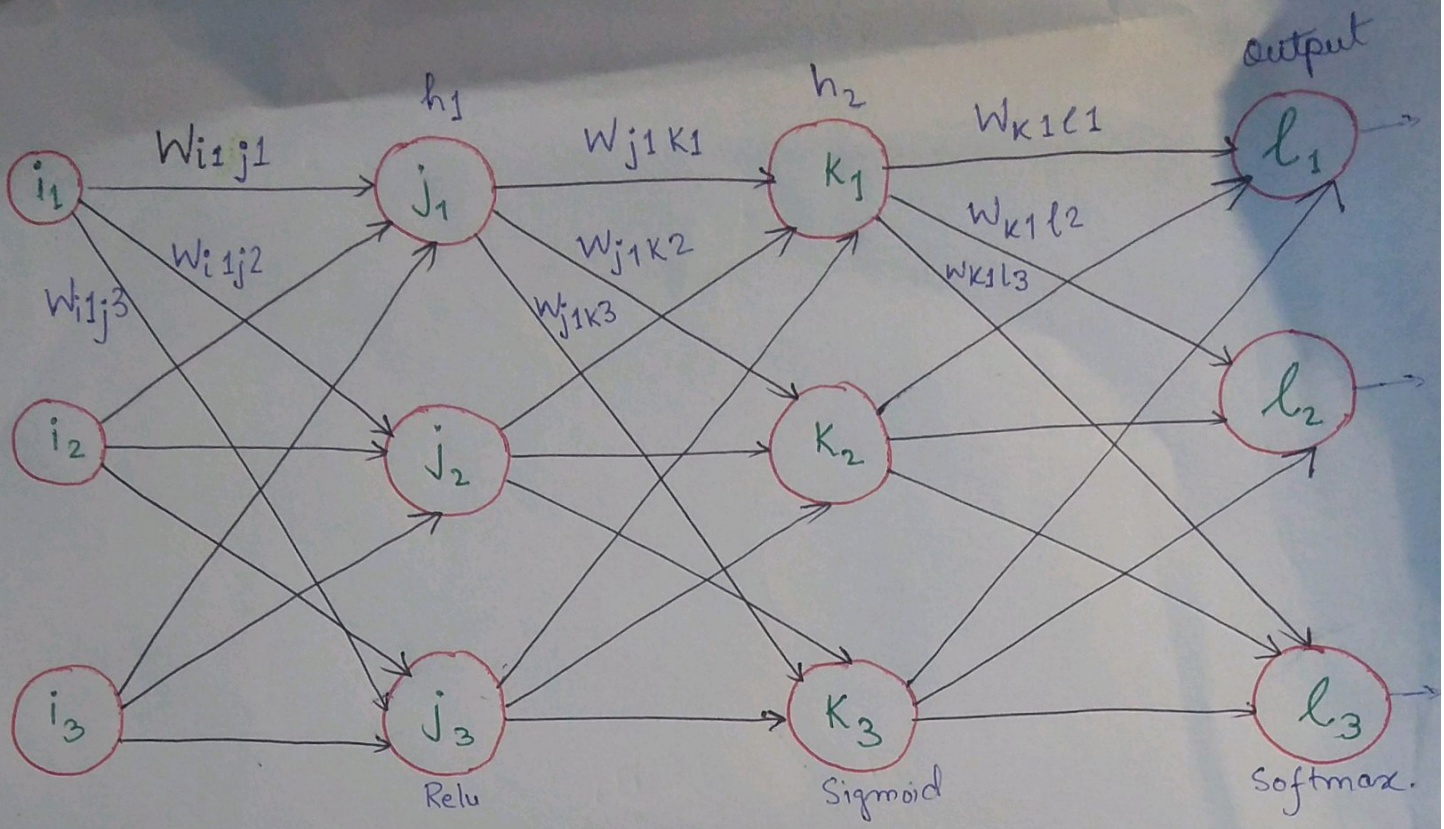


Feed Forward Neural Network.

$$\text{Input} = [0.1 \quad 0.2 \quad 0.7]$$

$$W_{ij} = \begin{bmatrix} W_{i1j1} & W_{i1j2} & W_{i1j3} \\ W_{i2j1} & W_{i2j2} & W_{i2j3} \\ W_{i3j1} & W_{i3j2} & W_{i3j3} \end{bmatrix} = \begin{bmatrix} 0.1 & 0.2 & 0.3 \\ 0.3 & 0.2 & 0.7 \\ 0.4 & 0.3 & 0.9 \end{bmatrix}$$

$$W_{jK} = \begin{bmatrix} W_{j1K1} & W_{jK2} & W_{j1K3} \\ W_{j2K1} & W_{j2K2} & W_{j2K3} \\ W_{j3K1} & W_{j3K2} & W_{j3K3} \end{bmatrix} = \begin{bmatrix} 0.2 & 0.3 & 0.5 \\ 0.3 & 0.5 & 0.7 \\ 0.6 & 0.4 & 0.8 \end{bmatrix}$$

$$W_{Kl} = \begin{bmatrix} W_{K1l1} & W_{K1l2} & W_{K1l3} \\ W_{K2l1} & W_{K2l2} & W_{K2l3} \\ W_{K3l1} & W_{K3l2} & W_{K3l3} \end{bmatrix} = \begin{bmatrix} 0.1 & 0.4 & 0.8 \\ 0.3 & 0.7 & 0.2 \\ 0.5 & 0.2 & 0.9 \end{bmatrix}$$

$$\text{Output} = [1.0 \quad 0.0 \quad 0.0]$$

Weights : used to connect the each neuron in one layer to the every neurons in the next layer.

It determine the strength of the connection of the neurons.

Weights near zero means changing input will not change the output. Many algorithms will automatically set it to zero to simplify the network.

Now, Activation function:

Artificial neuron, calculates a "weighted sum" of its input, add a bias and then decides whether it should be 'fired' or not.

Considering,

$$Y = \Sigma(\text{weight} * \text{input}) + \text{bias}$$

How to decide whether neuron should fire or not?

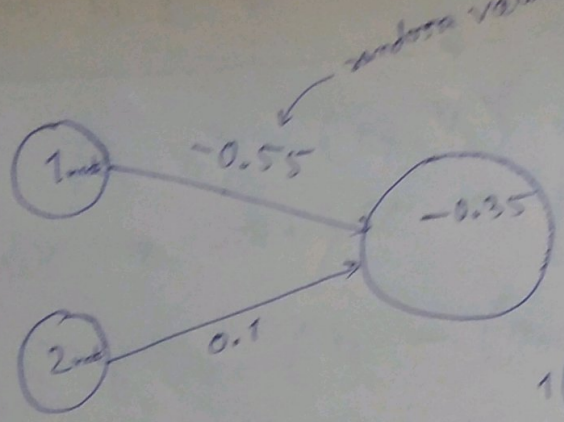
We added "activation functions"

Here,

Fired means activated.

- Bias determines if a neuron is activated.
- Bias increases the flexibility of the model.
- Each neuron has a bias

Relu
0 on input itself
≤0
>0



$1(-0.55) + 2(0.1) = -0.35$

relu $(-0.35) = 0$

fig: Neuron network.

Infact with Relu,
Any neuron whose weighted sum of input is less than or equal to '0' will not be firing, i.e no information from this activation fn will be passed forward to the rest of the network.

0 is the threshold.

Q. what if we want (−1) need to be the threshold?

This is where Bias comes into play.

* Bias gets activated ~~to the fun~~ added to the weighted sum before being passed to the activation fn.

The value assigned to bias is the opposite of so called threshold value.

Now,

Bias will be opposite of -1.

i.e 1.

Now, $1(-0.55) + 2(0.1) + (1) = 0.65$

(1 ↑ bias)

$$relu(0.65) = 0.65$$

Now, the model will fire.

Softmax : It is the output f^n of the last layer in neural networks (if the network has n layers ther n-th layer is the softmax function). This fact is important because the purpose of the last layer is to turn the score produced by the neural network into values that can be interpreted by humans.

Generally, this f^n ways will help in calculating the probabilities of each target class over all possible target classes.

Each layer of neural network filters and transforms the data before passing it to the next layer.

Building a neural network to process 224 x 224 color images including the 3 color (RGB), that means 224x224x3 = 150,528 input features.

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

Fig: A 3x3 filter

we can use an input image & a filter to produce an output image by convolving the filter with the input image.

Method

1. Overlaying the filter on top of the image at some location.
2. Performing element wise - multiplication between the values in the filter & their corresponding values in the image.
3. Summing up - all the element wise products. This sum is the output value for the destination pixel in the output image.
4. Repeating for all locations.

0	50	0	29
0	80	31	2
33	90	0	75
0	9	0	95

Fig: 4 × 4 image

{ 0 → is black
255 → is white }

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

Fig: filter (3 × 3).

To produce a 2 × 2 output we will convolve the input image & the filter.

?	?
?	?

Image value	Filter Value	Result
0	-1	0
50	0	0
0	1	0
0	-2	0
80	0	0
31	2	62
33	-1	-33
90	0	0
0	1	0

We sum up now the result

62 + (-33) = 29 → 1st pixel

29	?
?	?

→

29	-192
-35	-22

.