Introduction to Neural Nets

By Sana Iqbal

Data X

What is a Neural Network?

Like other machine learning methods that we saw earlier in the class , it is a technique to:

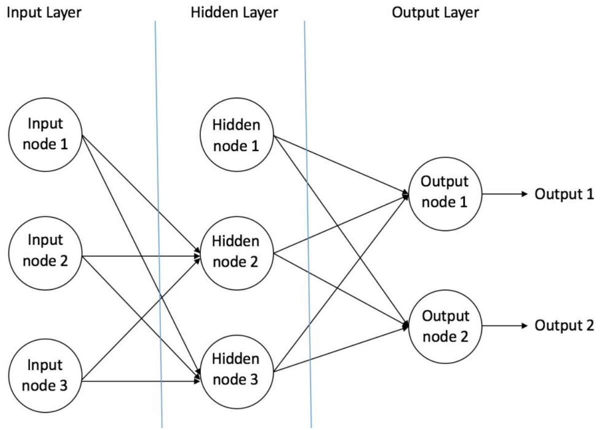
- ***map features*** *to labels or some dependant continuous value.*

or

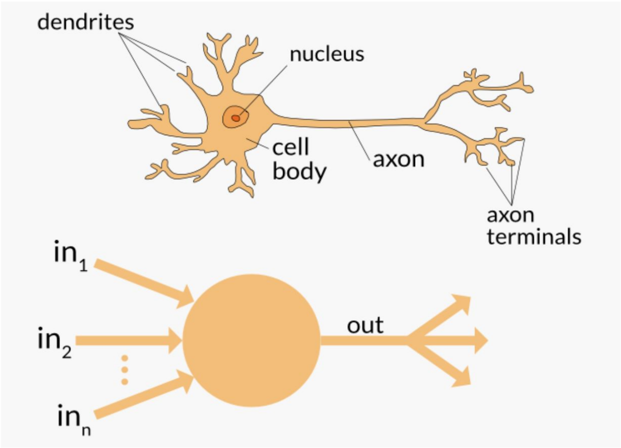
- ***compute the function*** *that relates features to labels or some dependant continuous value.*

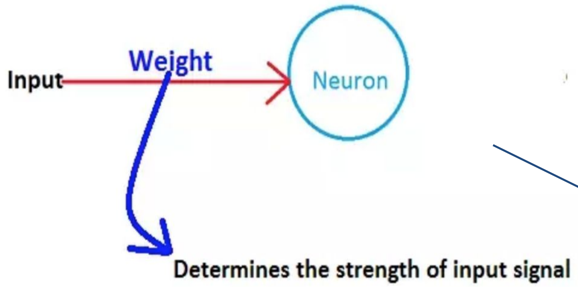
Neural Network

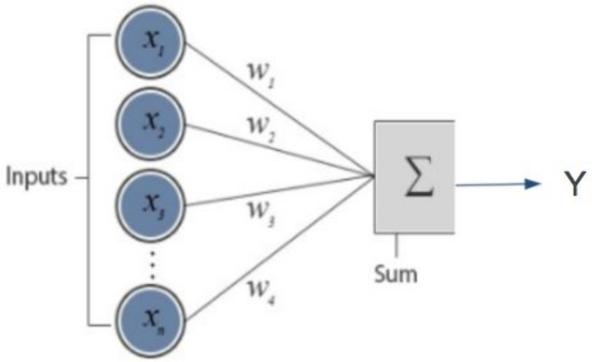
Network:
A network of neurons/nodes connected by a set of weights .



Neural:
Inspired by the way biological neural networks in the human brain process

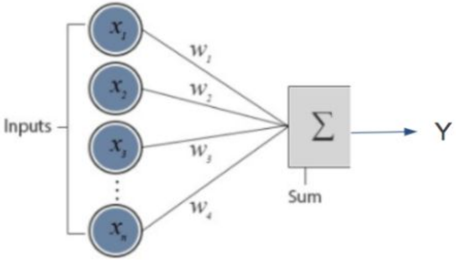






$Y = x1*w1 + x2*w2 + x3*w3 + \cdots + xn*wn$ **--linear regression**

Example:

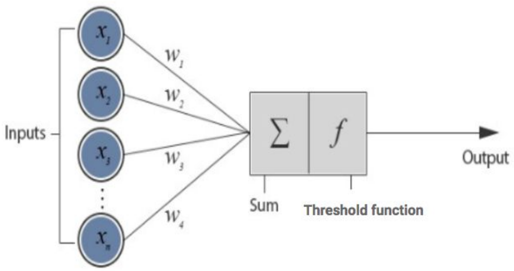


Y = x1∗w1 + x2∗w2 + x3∗w3 +· · · · ·+ xn∗wn **--linear regression**

For sample 1:				
x	6	5	3	1
w	**0.3**	**0.2**	**-0.5**	**0**
Y = sum(x * w) =**1.3**				

For sample 2:				
x	20	5	3	1
w	**0.3**	**0.2**	**-0.5**	**0**
Y = sum(x * w) = 5.5				

Lets us apply a **threshold function** on the output:

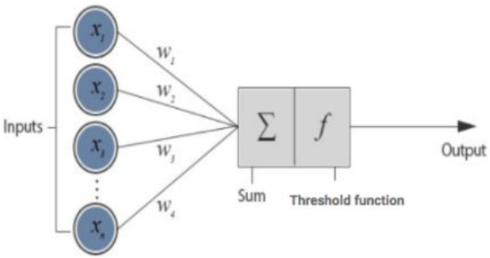


$$f(t) = \begin{cases} t & \text{if } t < 3 \\ 0 & \text{otherwise} \end{cases}$$

For sample 1:				
x	6	5	3	1
w	**0.3**	**0.2**	**-0.5**	**0**
Y = f(sum(x * w)) = f(1.3)= 1.3				

For sample 2:				
x	20	5	3	1
w	**0.3**	**0.2**	**-0.5**	**0**
Y = f(sum(x * w))= f(5.5)=0				

*Now, if we apply a **logistic/sigmoid function** on the output it will squeeze all the output between 0 and 1 :*



Y = Sigmoid(x1∗w1 + x2∗w2 + .. + xn∗wn) --**logistic regression**

Logistic/sigmoid function

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

For sample 1:

x	6	5	3	1
w	**0.3**	**0.2**	**-0.5**	**0**

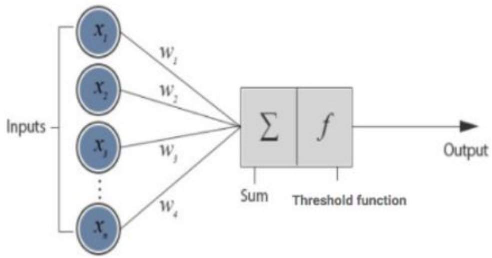
Y = σ(sum(x * w)) = **σ(1.3) = 0.78**

For sample 2:

x	20	5	3	1
w	**0.3**	**0.2**	**-0.5**	**0**

Y = σ(sum(x * w)) = **σ(5.5) = 0.99**

*Now, if we apply a **logistic/sigmoid function** on the output it will squeeze all the output between 0 and 1 :*



Y = Sigmoid(x1∗w1 + x2∗w2 + .. + xn∗wn) --**logistic regression**

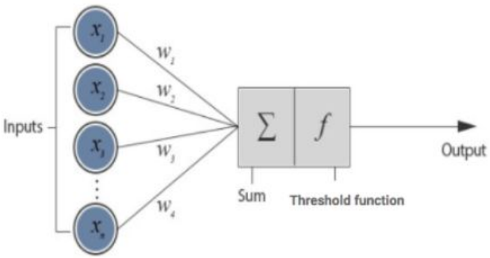
Logistic/sigmoid function

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

For sample 1:				
x	6	5	3	1
w	**0.3**	**0.2**	**-0.5**	**0**
Y = σ(sum(x * w)) = **σ(1.3) =** **0.78**				

For sample 2:				
x	20	5	3	1
w	**0.3**	**0.2**	**-0.5**	**0**
Y = σ(sum(x * w)) = **σ(5.5) =** **0.99**				

Now, if we apply a threshold on the ***logistic/sigmoid*** *output it will set the final output as 0 or 1 :*



Logistic/sigmoid function

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

$$f(t) = \begin{cases} 1 & \text{if } t > 0.6 \\ 0 & \textit{otherwise} \end{cases}$$

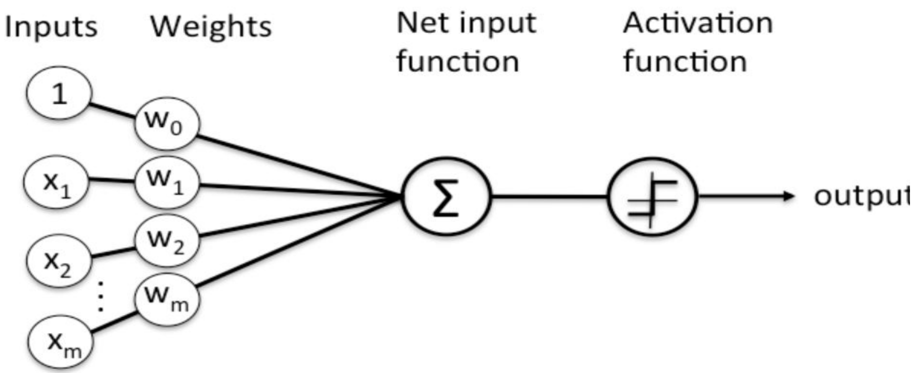
For sample 1:				
x	6	5	3	1
w	**0.3**	**0.2**	**-0.5**	**0**
Y = **f** (**σ**(sum(x * w)))= f(**σ**(1.3)) = f (0.78) =1				

For sample 2:				
x	20	5	3	1
w	**0.3**	**0.2**	**-0.5**	**0**
Y = **f**(**σ**(sum(x * w))) = f (**σ**(5.5)) = f (0.99)=1				

Review

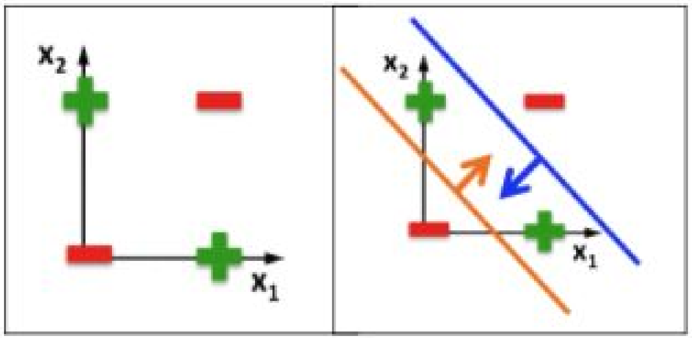
1. Neural nets want to find the function that maps features to outputs.
2. Neuron takes in weighted input(s)
3. Functions are used for transforming neuron output.

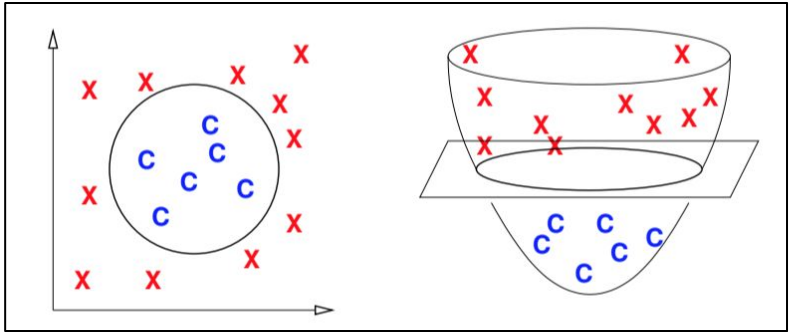
Basic Perceptron:



Schematic of Rosenblatt's perceptron.

All mapping functions are NOT linear

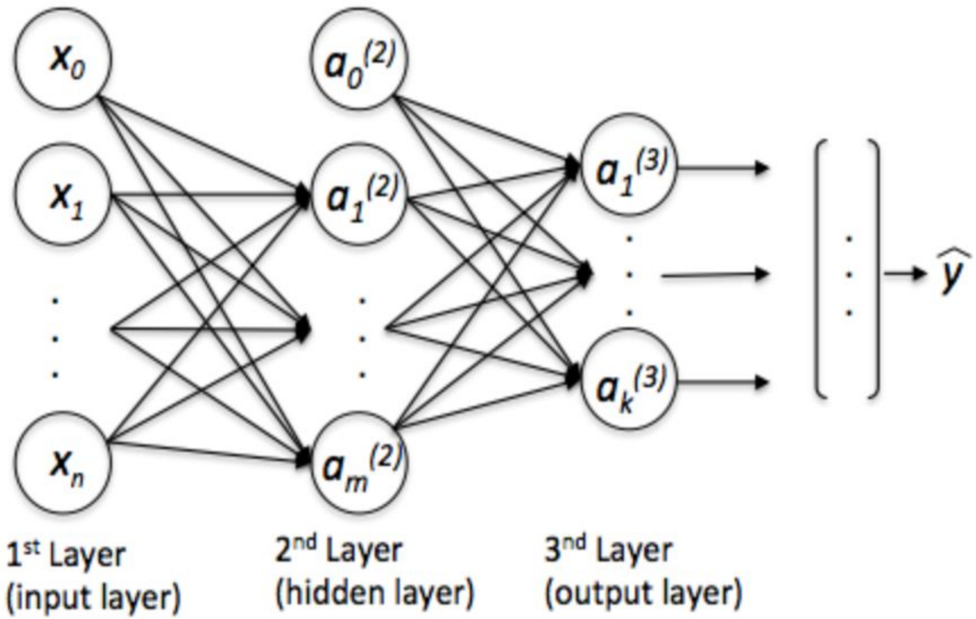




Activation functions

We use activation functions in neurons to induce nonlinearity in the neural nets so that it can learn complex functions also.

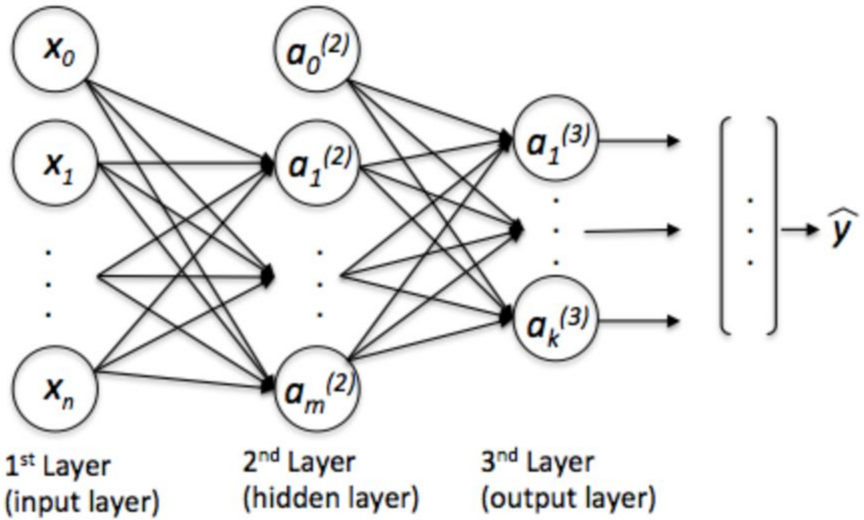
Neural Network



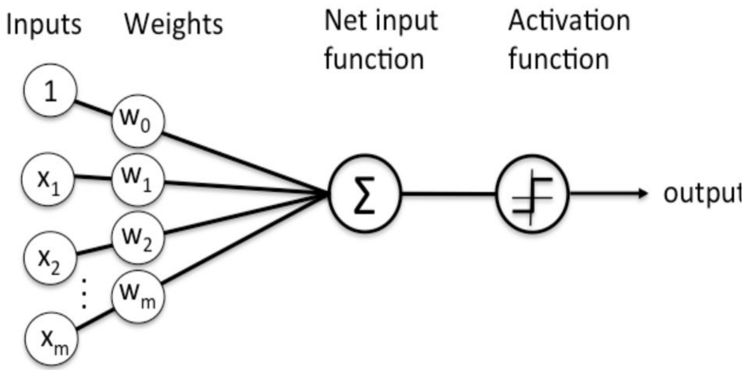
Schematic of a multi-layer perceptron.

- We use multiple combinations of inputs
- We use activation functions in neurons.

Compare the complexity:



Schematic of a multi-layer perceptron.



Schematic of Rosenblatt's perceptron.

How it works?

How does the network know the strength of connections between neurons?

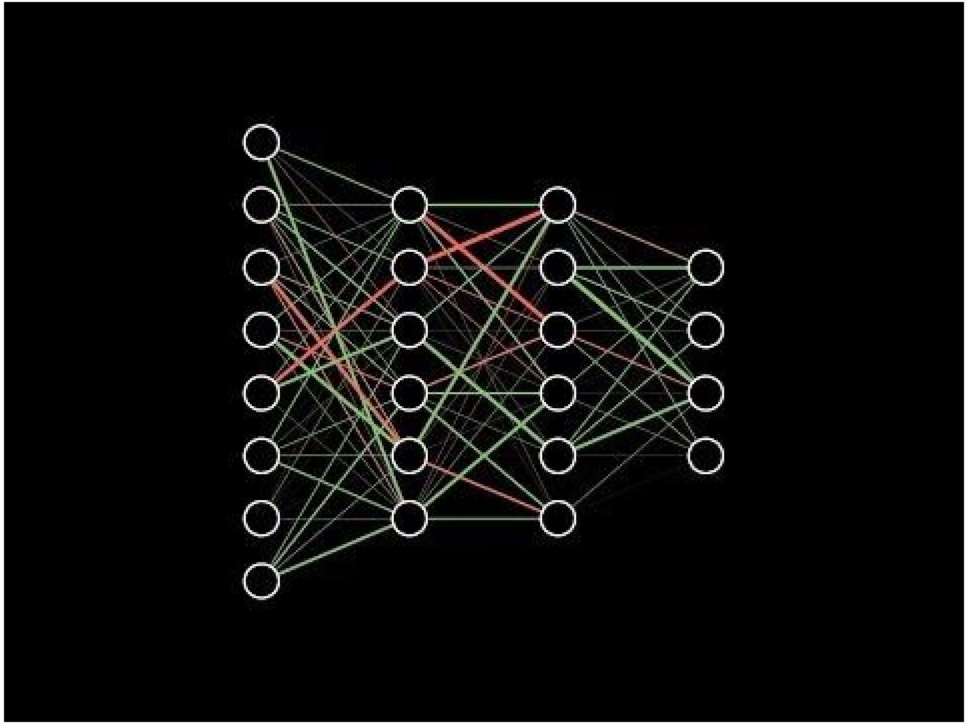
It learns them!

- We start with random weights.
- Calculate the output, then calculate Cost function.
- Find the gradient of the Cost function.
- The gradients are pushed back into the network and used for adjusting the weights - **Backpropagation**
- The whole process is repeated again till we train an acceptable model.

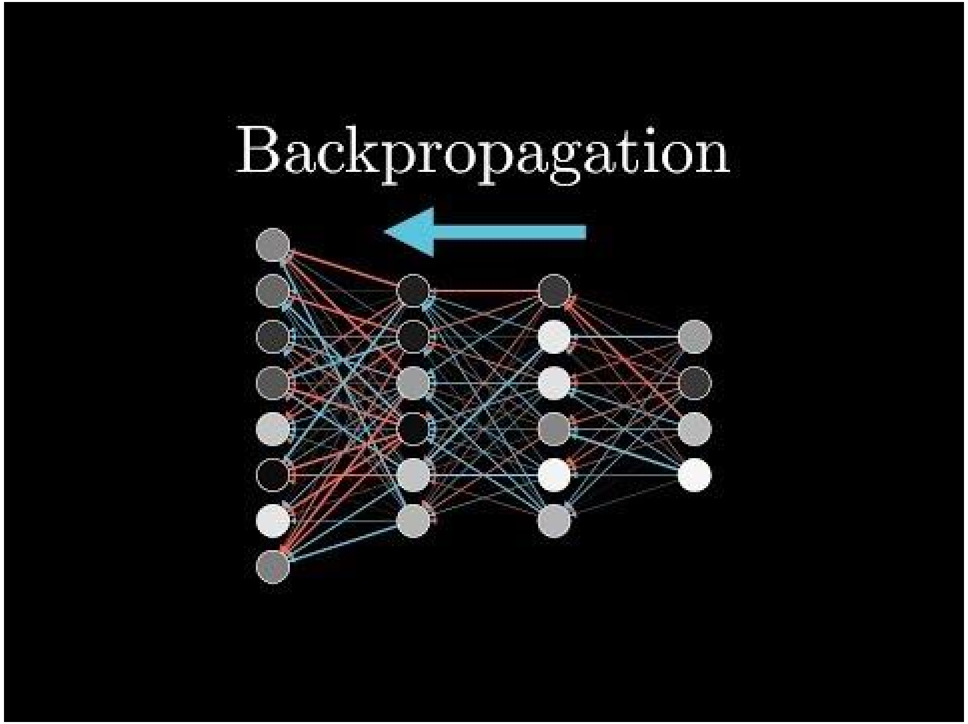
Basic Vocabulary Associated with Neural Networks

1. Input layer
2. Hidden layer
3. Output layer
4. Weights
5. Activation Functions
6. Back propagation
7. Gradient Descent
8. Learning Rate
9. Batch Gradient Descent
10. Stochastic Gradient Descent
11. Epochs

https://www.youtube.com/watch?v=aircAruvnKk



https://www.youtube.com/watch?v=Ilg3gGewQ5U



Review

1. Draw a 2 hidden layer neural net with input of size 2 units, hidden layer-1 of size 3, hidden layer -2 of size 4, the output should be neurons.

2. Given input [3,2] and weight matrix W=
Calculate the output, if the activation function is sigmoid.

0.2	0
0.91	2.25
0.7	0.6

Solution

$$\begin{bmatrix} 0.2 & 0 \\ 0.91 & 2.25 \\ 0.7 & 0.6 \end{bmatrix} \times \begin{bmatrix} 3 \\ 2 \end{bmatrix} \longrightarrow \begin{bmatrix} 0.6 \\ 7.23 \\ 3.3 \end{bmatrix} \xrightarrow[\text{sigmoid}]{} \begin{bmatrix} 0.64 \\ 0.99 \\ 0.94 \end{bmatrix}$$

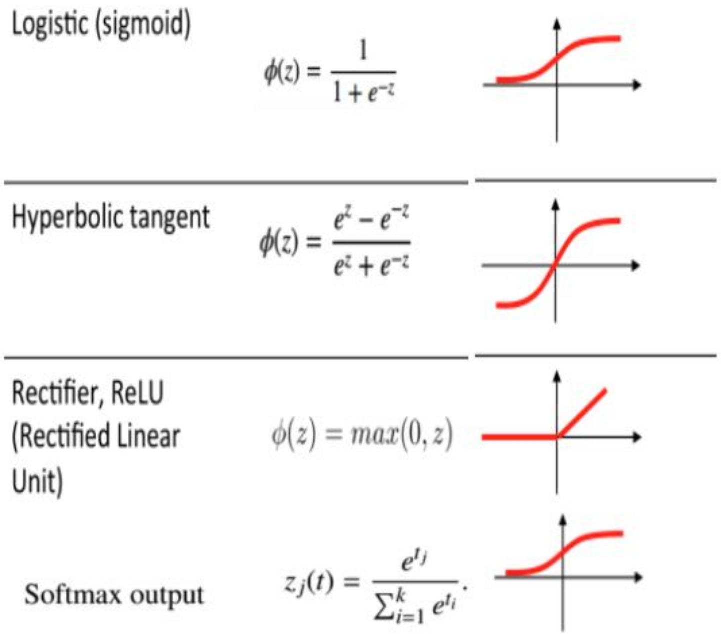
$$y = \begin{bmatrix} 0.64 \\ 0.99 \\ 0.94 \end{bmatrix}$$

$$\text{Actual}(z) = \begin{bmatrix} 0.40 \\ 0.89 \\ 0.64 \end{bmatrix}$$

Calculate Mean Square loss: sq(z-y)

Some activation Functions:

1. Sigmoid (0,1)
2. Tanh (-1,1)
3. Relu (0, x)
4. Softmax (0,1)



Unit Saturation / vanishing gradients:

During back propagation we calculate gradients of activation functions, for sigmoid :

$$s' = s(1 - s)$$

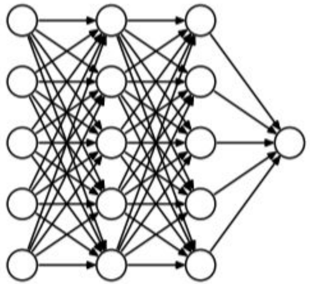
When: **$s \approx 0$ or 1**

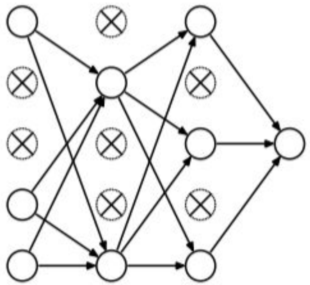
s`$=s(1 - s) \approx 0$,

gradient descent changes very slowly, *making training slow.*

Regularization in neural nets

Dropout is an approach to regularization in neural networks which helps reducing interdependent learning amongst the neurons.





Advantages:

It finds the best function approximation from a given set of inputs, we do not need to define features - Representational Learning.

Eg:

- Representational Learning is used to get Word Vectors.
- We do not need to handcraft image features

Cons:

Needs a lot of data.

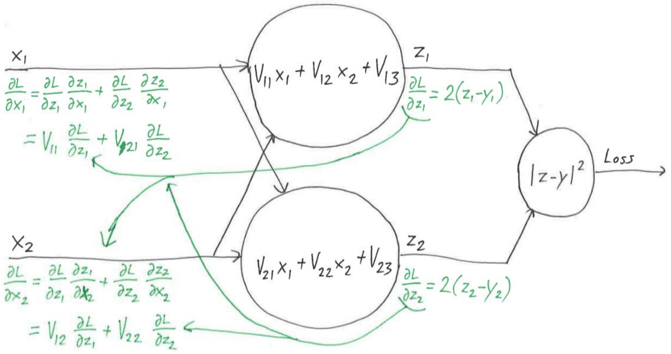
Heavily parametrized by weights.

Notes:

1. **Large weights and small weights can give very varied neuron values:**
   We can overcome this problem by introducing a new type of artificial neuron called a *sigmoid* neuron. Sigmoid neurons are similar to perceptrons, but modified so that small changes in their weights and bias cause only a small change in their output.

2. **Why do we add bias term?**
   To set the threshold value of neuron firing

3. **Why do we want activation functions?**
   https://towardsdatascience.com/activation-functions-and-its-types-which-is-better-a9a5310cc8f
4. Go through this diagram to understand backpropagation:

5. 

Ref:
http://neuralnetworksanddeeplearning.com/chap1.html