Forward Propagation

As the name suggests, the input data is fed in the forward direction through the network. Each hidden layer accepts the input data, processes it as per the activation function and passes to the successive layer.
In order to generate some output, the input data should be fed in the forward direction only. The data should not flow in reverse direction during output generation otherwise it would form a cycle and the output could never be generated.

Backward Propagation

Back Propogation is the essence of natural net training. It is the practice of fine-tuning the weights of a neural net based on the error rate (i.e loss) obtained in the previous epoch (i.e. iteration). Proper tuning of the weights ensures lower error rates, making the model reliable by increasing it's generalisation.
Backpropogation is a short form for "backward propogation of errors". It is a standard method of training artificial neural networks.
This method helps to calculate the gradient of a loss function with respect to all the weights in the network.

Question2 : (Vectorised implementation)

$X \in \mathbb{R}^{1\times 4}$ } Input

$W1 \in \mathbb{R}^{5\times 4}$ $\quad b1 \in \mathbb{R}^{5\times 1}$ } Parameters

$W2 \in \mathbb{R}^{1\times 5}$ $\quad b2 \in \mathbb{R}^{1\times 1}$

$Y \in \mathbb{R}^{1\times 1}$ } Output

Forward Propagation:

Assume weights and bias are initialized

$Z1 = (W1)\cdot(X^T) + b1$

$A1 =$ sigmoid $(Z1)$

$Z2 = (W2)\cdot(A1^T) + b2$

$A2 =$ Sigmoid $(Z2)$

$Cost = -Y \log A2 - (1-Y)\log(1-A2)$

Backward Propagation:

Say dZ denotes $d(cost)/d(Z)$

$dZ2 = A2 - Y$

$dW2 = (dZ2)\cdot(A1^T)$

$dB2 = dZ2$

$dZ1 = [(W2^T)\cdot(dZ2)] * (A1)*(1-A1)$ $\quad$ sigmoid$'(Z1)$

$dW1 = (dZ1)\cdot(X^T)$

$dB1 = dZ1$

Question2 : (For general MLP)

$X \in R^{n[0] \times m}$ $\quad$ $n^{[l]}$: number of features in layer l

$Y \in R^{1 \times m}$ $\quad$ m: number of examples

$W^{[l]} \in R^{n[l] \times n[l-1]}$ $\quad$ L: Number of layers

$b^{[l]} \in R^{n[l] \times m}$

Forward Propagation :

\# Assume $X = A[0]$

for l in range(L):

$Z["l+1"] = W["l+1"] \cdot A["l"] + b["l+1"]$

$A["l+1"] = \text{sigmoid}(Z["l+1"])$

{ Store Z["l"] and A["l"] }

$$Cost = \frac{1}{m} \sum_{i=1}^{m} \left\{ -Y^{(i)} \log(A[L]^{(i)}) - (1 - Y^{(i)}) \log(1 - A[L]^{(i)}) \right\}$$

Backward Propagation :

$dZ["L"] = A["L"] - Y$, $dW["L"] = (dZ["L"]) \cdot (A[L-1]^T)$, $db["L"] = dZ^{[L]}$

for l from L-1 to 1 :

$dZ["l"] = \left[(W["l+1"]^T) \cdot (dZ[l+1]) \right] * (A[l]) * (1 - A[l])$

$dW["l"] = (dZ["l"]) \cdot (A[l-1]^T)$

$db["l"] = dZ["l"]$

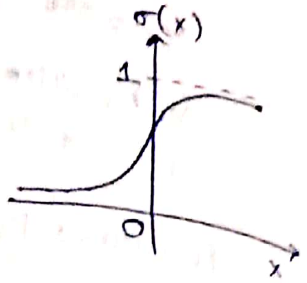
Store all $dW^{[l]}$ and $dB[l]$

Question 3

(a) Sigmoid :

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

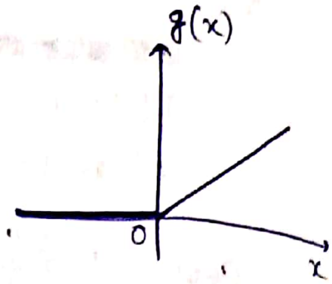
$$\sigma'(x) = \frac{e^{-x}}{(1+e^{-x})^2} = \sigma(x)\cdot(1-\sigma(x))$$



(b) Relu :

$$g(x) = \max(0, x)$$

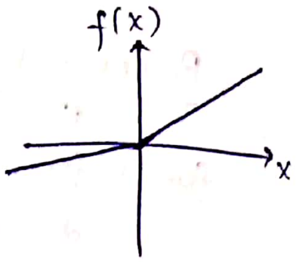
$$g'(x) = \begin{cases} 0 & x < 0 \\ 1 & x \geq 0 \end{cases}$$



(c) Leaky Relu :

$$f(x) = \max(0.01x, x)$$

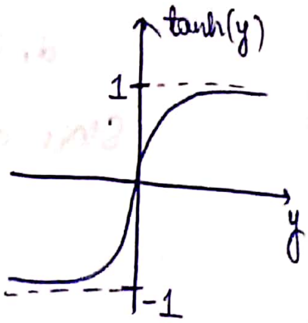
$$f'(x) = \begin{cases} 0.01 & x < 0 \\ 1 & x \geq 0 \end{cases}$$



(d) Tanh :

$$\tanh(y) = \frac{e^y - e^{-y}}{e^y + e^{-y}}$$

$$\tanh'(y) = \frac{4}{(e^y + e^{-y})^2} = 1 - (\tanh(y))^2$$



(e) Softmax :

$$\sigma(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \quad \forall\ i = 1, 2, \dots K$$

and

$$z = (z_1, z_2, \dots z_K) \in \mathbb{R}^K$$

$$\frac{\partial(\sigma(z_i))}{\partial z_i} = \sigma(z_i)\cdot(1-\sigma(z_i))$$

$$\frac{\partial(\sigma(z_i))}{\partial z_j} = -\sigma(z_i)\cdot\sigma(z_j) \quad \forall\ i \neq j$$