Neural Network Maths

Nirbhay Sharma, IISc Bangalore

December 11, 2024

1 Some important activation functions and their derivatives

1. Sigmoid ($\sigma(x) = \frac{1}{1+e^{-x}}$)
2. Tanh (tanh(x) = $\frac{e^{2x}-1}{e^{2x}+1}$)
3. Relu (ReLU(x) = x if $x > 0$ else 0)
4. SiLU (Silu(x) = $x * \sigma(x)$)
5. Softmax (softmax(x) = $\frac{e^{x_i}}{\sum_{j=1}^{n} e^{x_j}}$)

1.1 Derivative of Sigmoid

$$\frac{d\sigma(x)}{dx} = \sigma(x)(1 - \sigma(x)) \tag{1}$$

1.2 Derivative of Tanh

$$\frac{d\tanh(x)}{dx} = 1 - tanh(x)^2 \tag{2}$$

1.3 Derivative of Relu

$$\frac{dReLU(x)}{dx} = 1 \text{ if } x > 0 \text{ else } 0 \tag{3}$$

1.4 Derivative of SiLU

$$\frac{dSiLU(x)}{dx} = \frac{dx\sigma(x)}{dx} = x\sigma(x)(1 - \sigma(x)) + \sigma(x) = \sigma(x)(1 + x(1 - \sigma(x))) \tag{4}$$

1.5 Derivative of Softmax

$$x = (x_1, x_2, ..., x_n) \tag{5}$$

$$\text{Softmax}(x) = y_i = \frac{e^{x_i}}{\sum_{j=1}^{n} e^{x_j}},\ i = 1...n \tag{6}$$

$$\frac{\partial y_i}{\partial x_i} = \frac{e^{x_i}}{\sum_{j=1}^{n} e^{x_j}}(1 - \frac{e^{x_i}}{\sum_{j=1}^{n} e^{x_j}}) = y_i(1 - y_i) \tag{7}$$

$$\frac{\partial y_i}{\partial x_j} = \frac{-e^{x_i} e^{x_j}}{(\sum_{j=1}^{n} e^{x_j})^2} = -y_i y_j \tag{8}$$

overall derivative of output of softmax wrt input is as follows

$$\frac{dy}{dx} = \begin{pmatrix} \frac{\partial y_1}{\partial x_1} & \frac{\partial y_1}{\partial x_2} & ... & \frac{\partial y_1}{\partial x_n} \\ . & . & . & . \\ \frac{\partial y_n}{\partial x_1} & \frac{\partial y_n}{\partial x_2} & ... & \frac{\partial y_n}{\partial x_n} \end{pmatrix} = \begin{pmatrix} y_1(1-y_1) & -y_1y_2 & ... & -y_1y_n \\ -y_1y_2 & y_2(1-y_2) & ... & -y_2y_n \\ . & . & . & . \\ -y_1y_n & -y_2y_n & ... & y_n(1-y_n) \end{pmatrix} \tag{9}$$

2 Forward Propagation

1. The neural network has layers with W, b as weight and biases (learnable parameters)
2. Neural network also uses activation functions to incorporate nonlinearity
3. A typical forward pass looks like **Softmax(Layer(tanh(Layer(sigmoid(Layer(x))))))**, x is the input, sigmoid, tanh, softmax all are activation functions, Layer is a simple hidden layer with W, b as learnable parameters and it uses the equation $Y = WX + b$
4. Forward propagation is relatively straight forward, the major part of neural networks lies in Back propagation which we see in next section

3 BackPropagation

It is the method to backpropagate the loss or the error term E through the network and calculate gradients; the gradients are calculated using the famous chain rule of differentiation.

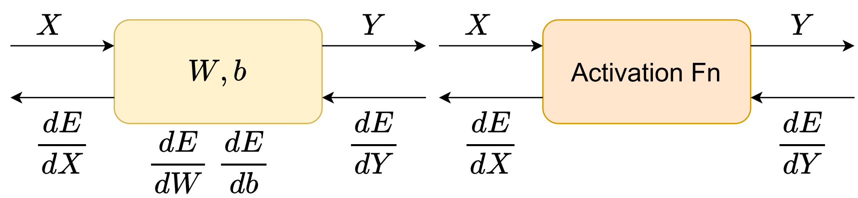

Figure 1: BackPropagation for a layer

1. In this layer, we have gradients coming from next layers, for a particular layer, we calculate the gradient wrt input, learnable parameters, then we flow back the gradients wrt input.
2. The layers having activation function does not have learnable parameters

3.1 Backpropagation for $Y = WX + b$

$$W = \begin{pmatrix} w_{11} & w_{12} & ... & w_{1n} \\ w_{21} & w_{22} & ... & w_{2n} \\ . & . & . & . \\ w_{m1} & w_{m2} & ... & w_{mn} \end{pmatrix} \tag{10}$$

$$Y = (y_1, y_2, ..., y_m) \tag{11}$$

$$X = (x_1, x_2, ..., x_n) \tag{12}$$

$$b = (b_1, b_2, ..., b_m) \tag{13}$$

$$y_i = \sum_{j=1}^{n} w_{ij}x_j + b_i \tag{14}$$

We now assume $\frac{\partial E}{\partial Y}$ is flowing from the next layer, now we want to calculate $\frac{\partial E}{\partial X}$, $\frac{\partial E}{\partial W}$, $\frac{\partial E}{\partial b}$

$$\frac{\partial E}{\partial X} = (\frac{\partial E}{\partial x_1}, \frac{\partial E}{\partial x_2}, ..., \frac{\partial E}{\partial x_n}) \tag{15}$$

$$\frac{\partial E}{\partial x_j} = \sum_{i=1}^{n} \frac{\partial E}{\partial y_i}\frac{\partial y_i}{\partial x_j} = \sum_{i=1}^{n} \frac{\partial E}{\partial y_i} w_{ij} \tag{16}$$

$$\frac{\partial E}{\partial X} = (\sum_{i=1}^{n} \frac{\partial E}{\partial y_i} w_{i1}, \sum_{i=1}^{n} \frac{\partial E}{\partial y_i} w_{i2}, ..., \sum_{i=1}^{n} \frac{\partial E}{\partial y_i} w_{in}) = W^T \frac{\partial E}{\partial Y} \tag{17}$$

$$\frac{\partial E}{\partial b} = (\frac{\partial E}{\partial b_1}, \frac{\partial E}{\partial b_2}, ..., \frac{\partial E}{\partial b_m}) \tag{18}$$

$$\frac{\partial E}{\partial b_j} = \sum_{i=1}^{n} \frac{\partial E}{\partial y_i}\frac{\partial y_i}{\partial b_j} = \frac{\partial E}{\partial y_j} \tag{19}$$

$$\frac{\partial E}{\partial b} = \frac{\partial E}{\partial Y} \tag{20}$$

$$\frac{\partial E}{\partial W} = \begin{pmatrix} \frac{\partial E}{\partial w_{11}} & \frac{\partial E}{\partial w_{12}} & \cdots & \frac{\partial E}{\partial w_{1n}} \\ . & . & . & . \\ \frac{\partial E}{\partial w_{m1}} & \frac{\partial E}{\partial w_{m2}} & \cdots & \frac{\partial E}{\partial w_{mn}} \end{pmatrix} \tag{21}$$

$$\frac{\partial E}{\partial w_{ij}} = \frac{\partial E}{\partial y_i}\frac{\partial y_i}{\partial w_{ij}} = \frac{\partial E}{\partial y_i} x_j \tag{22}$$

$$\frac{\partial E}{\partial W} = \begin{pmatrix} \frac{\partial E}{\partial y_1} x_1 & \frac{\partial E}{\partial y_1} x_2 & \cdots & \frac{\partial E}{\partial y_1} x_n \\ . & . & . & . \\ \frac{\partial E}{\partial y_m} x_1 & \frac{\partial E}{\partial y_m} x_2 & \cdots & \frac{\partial E}{\partial y_m} x_n \end{pmatrix} = \frac{\partial E}{\partial Y} X^T \tag{23}$$

In summary,

$$\frac{\partial E}{\partial X} = W^T \frac{\partial E}{\partial Y}, \; \frac{\partial E}{\partial b} = \frac{\partial E}{\partial Y}, \; \frac{\partial E}{\partial W} = \frac{\partial E}{\partial Y} X^T \tag{24}$$

3.2 Backpropagation for Sigmoid $Y = \sigma(X)$

for activation function we only calculate $\frac{\partial E}{\partial X}$ as there are no learnable parameters

$$\frac{\partial E}{\partial X} = (\frac{\partial E}{\partial x_1}, \frac{\partial E}{\partial x_2}, ..., \frac{\partial E}{\partial x_n}) \tag{25}$$

$$\frac{\partial E}{\partial x_j} = \sum_{i=1}^{n} \frac{\partial E}{\partial y_i}\frac{\partial y_i}{\partial x_j} = \frac{\partial E}{\partial y_j}\frac{\partial y_j}{\partial x_j} = \frac{\partial E}{\partial y_j} y_j(1 - y_j) \tag{26}$$

$$\frac{\partial E}{\partial X} = (\frac{\partial E}{\partial y_1} y_1(1 - y_1), \frac{\partial E}{\partial y_2} y_2(1 - y_2), ..., \frac{\partial E}{\partial y_n} y_n(1 - y_n)) \tag{27}$$

$$\frac{\partial E}{\partial X} = \frac{\partial E}{\partial Y} \odot Y(1 - Y) \tag{28}$$

$\odot$ is element wise multiplication

3.3 Backpropagation for Tanh $Y = Tanh(X)$

for activation function we only calculate $\frac{\partial E}{\partial X}$ as there are no learnable parameters

$$\frac{\partial E}{\partial X} = (\frac{\partial E}{\partial x_1}, \frac{\partial E}{\partial x_2}, ..., \frac{\partial E}{\partial x_n}) \tag{29}$$

$$\frac{\partial E}{\partial x_j} = \sum_{i=1}^{n} \frac{\partial E}{\partial y_i}\frac{\partial y_i}{\partial x_j} = \frac{\partial E}{\partial y_j}\frac{\partial y_j}{\partial x_j} = \frac{\partial E}{\partial y_j}(1 - y_j^2) \tag{30}$$

$$\frac{\partial E}{\partial X} = (\frac{\partial E}{\partial y_1}(1 - y_1^2), \frac{\partial E}{\partial y_2}(1 - y_2^2), ..., \frac{\partial E}{\partial y_n}(1 - y_n^2)) \tag{31}$$

$$\frac{\partial E}{\partial X} = \frac{\partial E}{\partial Y} \odot (1 - Y^2) \tag{32}$$

3.4 Backpropagation for Relu $Y = ReLU(X)$

for activation function we only calculate $\frac{\partial E}{\partial X}$ as there are no learnable parameters

$$\frac{\partial E}{\partial X} = (\frac{\partial E}{\partial x_1}, \frac{\partial E}{\partial x_2}, ..., \frac{\partial E}{\partial x_n}) \tag{33}$$

$$\frac{\partial E}{\partial x_j} = \sum_{i=1}^{n} \frac{\partial E}{\partial y_i}\frac{\partial y_i}{\partial x_j} = \frac{\partial E}{\partial y_j}\frac{\partial y_j}{\partial x_j} = \frac{\partial E}{\partial y_j}_{y_j \neq 0} \tag{34}$$

$$\frac{\partial E}{\partial X} = \frac{\partial E}{\partial Y}_{Y<0=0} \tag{35}$$

3.5 Backpropagation for SiLU $Y = SiLU(X)$

for activation function we only calculate $\frac{\partial E}{\partial X}$ as there are no learnable parameters

$$\frac{\partial E}{\partial X} = (\frac{\partial E}{\partial x_1}, \frac{\partial E}{\partial x_2}, ..., \frac{\partial E}{\partial x_n}) \tag{36}$$

$$\frac{\partial E}{\partial x_j} = \sum_{i=1}^{n} \frac{\partial E}{\partial y_i}\frac{\partial y_i}{\partial x_j} = \frac{\partial E}{\partial y_j}\frac{\partial y_j}{\partial x_j} = \frac{\partial E}{\partial y_j}\sigma(x_j)(1 + x_j(1 - \sigma(x_j))) \tag{37}$$

$$\frac{\partial E}{\partial X} = (\frac{\partial E}{\partial y_1}\sigma(x_1)(1 + x_1(1 - \sigma(x_1))), ..., \frac{\partial E}{\partial y_n}\sigma(x_n)(1 + x_n(1 - \sigma(x_n))) \tag{38}$$

$$\frac{\partial E}{\partial X} = \frac{\partial E}{\partial Y} \odot \sigma(X)(1 + X \odot (1 - \sigma(X))) \tag{39}$$

3.6 Backpropagation for Softmax $Y = Softmax(X)$

for activation function we only calculate $\frac{\partial E}{\partial X}$ as there are no learnable parameters

$$\frac{\partial E}{\partial X} = (\frac{\partial E}{\partial x_1}, \frac{\partial E}{\partial x_2}, ..., \frac{\partial E}{\partial x_n}) \tag{40}$$

$$\frac{\partial E}{\partial x_j} = \sum_{i=1}^{n} \frac{\partial E}{\partial y_i}\frac{\partial y_i}{\partial x_j} \tag{41}$$

Recall that

$$\frac{\partial y_i}{\partial x_j} = \begin{cases} y_i(1 - y_i) & i = j \\ -y_i y_j & i \neq j \end{cases} \tag{42}$$

$$\frac{\partial E}{\partial X} = \begin{pmatrix} \frac{\partial E}{\partial y_1} y_1(1-y_1) - \frac{\partial E}{\partial y_2} y_1 y_2 + ... - \frac{\partial E}{\partial y_n} y_1 y_n \\ -\frac{\partial E}{\partial y_1} y_1 y_2 + \frac{\partial E}{\partial y_2} y_2(1-y_2) + ... - \frac{\partial E}{\partial y_n} y_2 y_n \\ . \\ . \\ . \\ -\frac{\partial E}{\partial y_1} y_1 y_n - \frac{\partial E}{\partial y_2} y_2 y_n) + ... + \frac{\partial E}{\partial y_n} y_n(1-y_n) \end{pmatrix} \tag{43}$$

$$= \begin{pmatrix} y_1(1-y_1) & -y_1 y_2 & .. & -y_1 y_n \\ -y_1 y_2 & y_2(1-y_2) & ... & -y_2 y_n \\ . & . & . & . \\ -y_1 y_n & -y_2 y_n & ... & y_n(1-y_n) \end{pmatrix} \frac{\partial E}{\partial Y} \tag{44}$$

$$\text{define } M = \begin{pmatrix} y_1 & y_2 & ... & y_n \\ y_1 & y_2 & ... & y_n \\ . & . & . & . \\ y_1 & y_2 & ... & y_n \end{pmatrix} \tag{45}$$

$$= \begin{pmatrix} y_1(1-y_1) & -y_1 y_2 & .. & -y_1 y_n \\ -y_1 y_2 & y_2(1-y_2) & ... & -y_2 y_n \\ . & . & . & . \\ -y_1 y_n & -y_2 y_n & ... & y_n(1-y_n) \end{pmatrix} \frac{\partial E}{\partial Y} = M \odot (I - M^T) \frac{\partial E}{\partial Y} \tag{46}$$

3.7 Backpropagation for Cross Entropy Loss $E = -\sum y_i log(\hat{y}_i)$

we only calculate derivative wrt $\hat{y}_i$ which is straight forward

$$\frac{\partial E}{\partial \hat{y}_i} = \begin{cases} \frac{-1}{\hat{y}_i} & y_i = 1 \\ 0 & \text{otherwise} \end{cases} \tag{47}$$

The gradients from these loss will flow backwards