Pratik | 804861

1. Derivative of sigmoid

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

$$\frac{d}{dx} S(x) = \frac{-1}{(1+e^{-x})^2} (-e^{-x}) \quad \text{(chain rule)}$$

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

$$= \frac{1}{1+e^{-x}} - \left(\frac{1}{1+e^{-x}}\right)^2$$

$$= S(x) - S(x)^2$$

$$= S(x)\,(1 - S(x))$$

→ The properties of the sigmoid function's derivative is convenient for ~~nu~~ neural networks, because the gradients for the layer can be ~~con~~ calculate using simple multiplication and substraction rather than performing the exponentials of sigmoid function.

2.1 derivative of softmax.

$$y_c = \frac{e^{z_c}}{\sum_{d=1}^{C} e^{z_d}} \quad \text{(quotient rule}$$

$$\text{if } i = j: \frac{\partial y_i}{\partial z_i} = \frac{\partial e^{z_i}/\Sigma_C}{\partial z_i} = \frac{e^{z_i}\Sigma_C - e^{z_i}e^{z_i}}{\Sigma_C^2}$$

$$= \frac{e^{z_i}}{\Sigma_C} \cdot \frac{\Sigma_C - e^{z_i}}{\Sigma_C} = y_i(1-y_i)$$

FABER-CASTELL

if for $\boxed{i \neq j}$ $\quad \frac{\partial y_i}{\partial z_j} = \frac{\partial \, e^{z_i}/\Sigma_c}{\partial z_j} = \frac{0 - e^{z_i} e^{z_j}}{\Sigma_c^2}$

$$= -\frac{e^{z_i}}{\Sigma_c} \cdot \frac{e^{z_j}}{\Sigma_c} = \underline{\underline{-y_j y_i}}$$

2.2 Cross Entropy loss derivative

$$L = -\sum_i t_i \log(y_i)$$

$$\frac{\partial L}{\partial z_i} = -\sum_{j=1}^{C} \frac{\partial t_j \log(y_j)}{\partial z_i} = -\sum_{j=1}^{C} t_j \frac{\partial}{\partial z_i} \log(y_i)$$

(chain rule)

$$= -\sum_{j=1}^{C} \frac{t_j}{y_j} \frac{\partial y_j}{\partial z_i}$$

$$= -\frac{t_i}{y_i} \frac{\partial y_i}{\partial z_i} - \sum_{j \neq i}^{C} \frac{t_j}{y_j} \frac{\partial y_j}{\partial z_i}$$

$$= -\frac{t_i}{y_i} y_i (1 - y_i) - \sum_{j \neq i}^{C} \frac{t_j}{y_j} (-y_j y_i)$$

$$= -t_i + t_i y_i + \sum_{j \neq i}^{C} t_j y_i$$

$$= -t_i + \sum_{j=1}^{C} t_j y_i$$

$$= -t_i + y_i \sum_{j=1}^{C} t_j$$

$$= \underline{\underline{y_i - t_i}}$$