Backpropagation: (For MLP)

Consider a network with L layers & N_ℓ neurons in the ℓ-th layer

$W^\ell = (W^\ell_{ij})$ → weight for the connection between the i-th neuron in layer ℓ & j-th neuron in layer $\ell-1$

b^ℓ → column vector with b^ℓ_i being the bias for the i-th neuron in layer ℓ

σ^ℓ → activation function of ℓ-th layer, acting componentwise on any vector

a^ℓ → output vector from layer ℓ after activation.

$$a^\ell = \sigma^\ell(W^\ell a^{\ell-1} + b^\ell)$$

$$W^\ell \in \mathbb{R}^{N_\ell \times N_{\ell-1}}, \quad b^\ell \in \mathbb{R}^{N_\ell \times 1}$$

$$a^\ell \in \mathbb{R}^{N_\ell \times 1}$$

Introduce the vector:

$$z^\ell = W^\ell a^{\ell-1} + b^\ell$$

Note:

1) The first layer is the input layer. No computations occur here. It only provides an input signal.
2) The final output of the network is a^L
3) Traditionally, σ^L is not called an activation function but an output function. (this is only for the final output layer)

Consider an input $X \in \mathbb{R}^{N_I} \equiv \mathbb{R}^{N_1}$ which has the true output $Y \in \mathbb{R}^{N_0} \equiv \mathbb{R}^{N_L}$.

Clearly $a^1 = X$

Define an error/cost function to be minimized

$$E := E(\hat{Y}, Y)$$

where $\hat{Y} = a^L$ is the network prediction. For instance, this could be the mean squared error.

Having evaluated $\hat{Y}$ (this corresponds to a "forward pass"), we now wish to evaluate

$$\frac{\partial E}{\partial W_{ij}^l}, \quad \frac{\partial E}{\partial b_i^l}$$

needed to update the weights & biases of the network.

Now

$$z_k^l = \sum_{j=1}^{N_{l-1}} W_{kj}^l a_j^{l-1} + b_k^l \qquad k = 1, \dots, N_l$$

$$a_k^l = \sigma^l(z_k^l)$$

Thus, we have in the $(l+1)$-th layer

$$z_m^{l+1} = \sum_{k=1}^{N_l} W_{mk}^{l+1} a_k^l + b_m^l \qquad m = 1, \dots, N_{l+1}$$

Using chain rule, we have

$$\frac{\partial E}{\partial W_{kj}^l} = \frac{\partial E}{\partial z_k^l} \frac{\partial z_k^l}{\partial W_{kj}^l}$$

$$= \frac{\partial E}{\partial a_k^l} \frac{\partial a_k^l}{\partial z_k^l} a_j^{l-1}$$

$$= \left[\sum_{m=1}^{N_{l+1}} \frac{\partial E}{\partial z_m^{l+1}} \frac{\partial z_m^{l+1}}{\partial a_k^l} \right] \cdot \sigma^{l'}(z_k^l)\, a_j^{l-1}$$

Define the "error" corresponding to a neuron k in the layer l as

$$\delta_k^l = \frac{\partial E}{\partial z_k^l} = \left[\sum_{m=1}^{N_{l+1}} \frac{\partial E}{\partial z_m^{l+1}} \frac{\partial z_m^{l+1}}{\partial a_k^l} \right] \sigma^{l'}(z_k^l)$$

$$\Rightarrow \boxed{\delta_k^l = \left[\sum_{m=1}^{N_{l+1}} \delta_m^{l+1} \, W_{mk}^{l+1} \right] \sigma^{l'}(z_k^l)} \quad — (1)$$

Thus,

$$\boxed{\frac{\partial E}{\partial W_{kj}^l} = \delta_k^l \cdot a_j^{l-1}} \quad — (2)$$

Similarly,

$$\boxed{\frac{\partial E}{\partial b_k^l} = \frac{\partial E}{\partial z_k^l} \frac{\partial z_k^l}{\partial b_k^l} = \delta_k^l} \quad — (3)$$

The errors are evaluated by propagating backwards. Thus, we need

$$\delta_j^L = \frac{\partial E}{\partial z_j^L} = \frac{\partial E}{\partial a_j^L} \frac{\partial a_j^L}{\partial z_j^L} = \frac{\partial E}{\partial a_j^L} \sigma^{L'}(z_j^L)$$

Since $a_j^L = \hat{Y}_j$

$$\boxed{\delta_j^L = \frac{\partial E}{\partial \hat{Y}_j} \sigma^{L'}(z_j^L)} \quad — (4)$$

Summarizing the steps to evaluate the derivatives of the cost/error function wrt the weight and biases:

1. Forward pass:
For the input X, do an evaluation of the network while storing all z^l & a^l for each layer l.

2. Evaluate δ_j^L for the final layer L, which will depend on the choice of error function

3. Backward pass:
Traverse the network backwards to evaluate δ_j^l from ① & ④

4. Find the derivatives using ② & ③