5) Bias update:

$b_j(old) = b_j(new) + \eta * Error_j$

$b_5 = 0.8 + 0.5 \times (-0.14) = 0.73$

$b_4 = -0.4 + 0.5 \times (-0.01) = -0.405$

$b_3 = 0.6 + 0.5 \times (-0.003) = 0.59$

Derivation of Back Propagation Algorithm:

To derive the eqn for updating weights in back-propagation algorithm, Stochastic gradient ~~rule~~ descent rule is applied. It involves iterating through every training example at a time and for each training example d every weight w_{ji} is updated by adding Δw_{ji} to it.

$$w_{ji} \leftarrow w_{ji} + \Delta w_{ji}$$

where, $\Delta w_{ji} = -\eta \frac{\partial E_d}{\partial w_{ji}}$

E_d = Error on training example d.

Therefore, $E_d(\vec{w}) = \frac{1}{2} \sum_{k \in outputs} (t_k - o_k)^2$

Notion used: 8

x_{ji} = the i^{th} input to unit j

w_{ji} = the weight associate with i^{th} input to unit j

net_j = ~~$\sum w_{ji} x_{ji}$~~ $\sum_i w_{ji} x_{ji}$ (The weighted sum of inputs for unit j)

o_j = the output computed by unit j

t_j = " target output for " "

σ = the sigmoid function; $\frac{1}{1+e^{-x}}$

outputs = the set of units in the final layer of the network

downstream(j) = the set of units whose immediate input include the output of unit j

To begin, ~~we~~ w_{ji} can influence the network only through net_j

Therefore using chain rule:

$$\frac{\partial E_d}{\partial w_{ji}} = \frac{\partial E_d}{\partial net_j} \times \frac{\partial net_j}{\partial w_{ji}}$$

As, $net_j = \sum w_{ji} x_{ji}$

$$\Rightarrow \frac{\partial net_j}{\partial w_{ji}} = x_{ji}$$

$$\Delta w_{ji} = \frac{\partial E_d}{\partial net_j} \cdot x_{ji} = -\eta \frac{\partial E_d}{\partial net_j} x_{ji} \quad — (1)$$

To derive convenient expression for $\frac{\partial E_d}{\partial net_j}$ we can follow 2 cases.

case 1 : unit j is the output unit for the network

case 2 : unit j is the internal unit " " "

Case 1 : Training Rule for Output unit Weights

net_j can influence the network only through o_j.

$$\frac{\partial E_d}{\partial net_j} = \frac{\partial E_d}{\partial o_j} \cdot \frac{\partial o_j}{\partial net_j}$$

Now, $$\frac{\partial E_d}{\partial o_j} = \frac{\partial}{\partial o_j}\left[\frac{1}{2}\sum_{k \in outputs}(t_k - o_k)^2\right]$$

$$= \frac{\partial}{\partial o_j}\frac{1}{2}(t_j - o_j)^2$$

$$= \frac{1}{2} \times 2(t_j - o_j)\frac{\partial}{\partial o_j}(t_j - o_j)$$

$$= -(t_j - o_j)$$

Then, $$\frac{\partial o_j}{\partial net_j} = \frac{\partial \sigma(net_j)}{\partial net_j}$$

$$= \sigma(net_j)\{1 - \sigma(net_j)\}$$ As $\frac{\partial \sigma(x)}{\partial x} = \sigma(x)[1 - \sigma(x)]$

$$= o_j(1 - o_j)$$

Finally,

$$\frac{\partial E_d}{\partial net_j} = -(t_j - o_j)(1-o_j)\,o_j$$

updating ①

$$\Delta w_{ji} = -\eta \times -(t_j - o_j)\,o_j\,(1-o_j)\,x_{ji}$$

$$= \eta\,(t_j - o_j)\,o_j\,(1-o_j)\,x_{ji}$$

$$= \eta\,\delta_j\,x_{ji} \quad \text{Where, } \delta_j = (t_j - o_j)\,o_j\,(1-o_j)$$

Case 2: Training Rule for Hidden Unit Weights

$$\frac{\partial E_d}{\partial net_j} = \sum_{k \in Downstream(j)} \frac{\partial E_d}{\partial net_k} \times \frac{\partial net_k}{\partial net_j}$$

$$= \sum_{k \in Downstream(j)} -\delta_k \times \frac{\partial net_k}{\partial o_j} \times \frac{\partial o_j}{\partial net_j} \quad \left[\because \frac{\partial E_d}{\partial net_k} = \delta_k = -(t_k - o_k)\,o_k(1-o_k)\right]$$

$$= \sum_{k \in Downstream(j)} -\delta_k \times \frac{\partial x_{ji} w_{ji}}{\partial o_j} \times \frac{\partial \sigma(net_j)}{\partial net_j}$$

$$= \sum_{k \in Downstream(j)} -\delta_k \times \frac{\partial o_j w_{ji}}{\partial o_j} \times \sigma(net_j)\left[1-\sigma(net_j)\right]$$

$$= \sum_{k \in Downstream(j)} -\delta_k \times w_{ji} \times o_j(1-o_j) = \delta_j$$

$$\therefore \Delta w_{ji} = -\eta \times \left[\sum_{k \in Downstream} -\delta_k \times w_{ji} \times o_j(1-o_j)\right] \times x_{ji}$$

$$= \eta\,\delta_j\,x_{ji}$$