Derivation of Back Propagation Algorithm

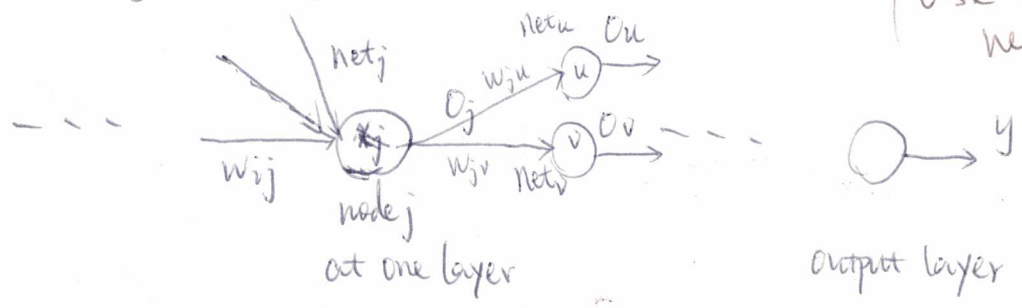
Objective: Minimize network prediction error on training examples.

Algorithm: Gradient descent.

Let network output be y, target value be t, define error as $E = \frac{1}{2}(t-y)^2$

Look at gradient of error w.r.t. a weight

(Use the figure on the next page.)



net_j : input that node j receives. $net_j = \sum_{i \in \text{previous layer}} O_i w_{ij}$

O_j : output of node j

Sigmoid function: $O_j = \sigma(net_j) = \frac{1}{1+e^{-net_j}}$

$$\frac{\partial E}{\partial w_{ij}} = \frac{\partial E}{\partial O_j} \cdot \frac{\partial O_j}{\partial net_j} \cdot \frac{\partial net_j}{\partial w_{ij}}$$

$$w_{ij} = w_{ij} - \alpha \frac{\partial E}{\partial w_{ij}}$$

1) $$\frac{\partial O_j}{\partial net_j} = -\frac{1}{(1+e^{-net_j})^2} e^{-net_j}(-1) = \frac{1}{1+e^{-net_j}} \cdot \frac{e^{-net_j}}{1+e^{-net_j}}$$

$$= \frac{1}{1+e^{-net_j}} \cdot \left(1 - \frac{1}{1+e^{-net_j}}\right) = O_j \cdot (1-O_j)$$

2) $$\frac{\partial net_j}{\partial w_{ij}} = O_i$$

3) Now look at $\frac{\partial E}{\partial O_j}$.

① If node j is the output node, then $O_j = y$.

$$\frac{\partial E}{\partial O_j} = \frac{\partial E}{\partial y} = y - t$$

② For other nodes.

$$\boxed{\frac{\partial E}{\partial O_j}} = \sum_{u \in \text{next layer}} \frac{\partial E}{\partial O_u} \cdot \frac{\partial O_u}{\partial O_j} = \sum_{u \in \text{next layer}} \frac{\partial E}{\partial O_u} \cdot \frac{\partial O_u}{\partial \text{net}_u} \frac{\partial \text{net}_u}{\partial O_j}$$

δ_j

$$= \sum_u \boxed{\frac{\partial E}{\partial O_u}} \cdot O_u(1-O_u) \cdot W_{ju} \qquad \text{Recursion !}$$

δ_u

∴ Algorithm: Start with output node, calculate $\frac{\partial E}{\partial O_j}$ layer by layer BACKWARDS !

$$\therefore \frac{\partial E}{\partial w_{ij}} = \sum_u \frac{\partial E}{\partial O_u} O_u(1-O_u) \, w_{ju} \, O_j(1-O_j) \cdot O_i$$

Vanishing Gradient: δ_j vanishes after repeated multiplication.

As $0 < O_j < 1$. Especially when O_j is close to 0 or 1