Derivation of the backpropagation rule that can be generalized

n: iteration
ξ: data sample
N: batch size

let j be an output neuron and let

$$\underbrace{e_j^{\xi}(n)}_{\text{error signal}} = \underbrace{d_j^{\xi}(n)}_{\text{desired output}} - \underbrace{y_j^{\xi}(n)}_{\text{network output}}$$

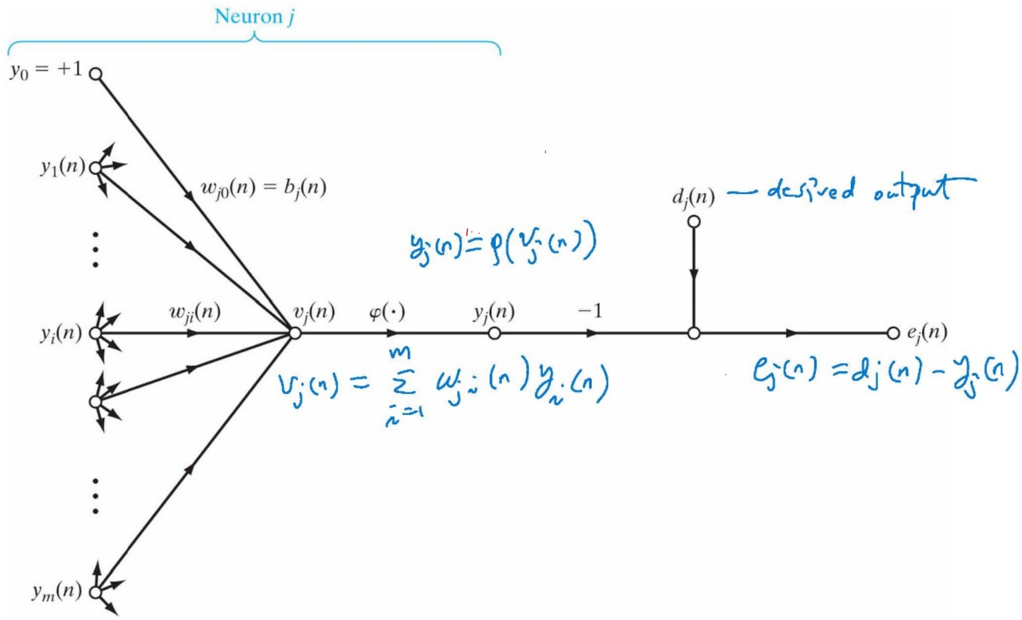
let $\mathcal{E}_j^{\xi}(n) = \frac{1}{2}\left(e_j^{\xi}(n)\right)^2$ be an instantaneous error cost

let $\mathcal{E}^{\xi}(n) = \sum_j \mathcal{E}_j^{\xi}(n)$ be a total instantaneous error cost (sum over all output neurons)

let $\mathcal{E}(n) = \frac{1}{N}\sum_{\xi=1}^{N} \mathcal{E}^{\xi}(n) = \frac{1}{N}\sum_{\xi=1}^{N}\sum_j \mathcal{E}_j^{\xi}(n) = \frac{1}{2N}\sum_{\xi}\sum_j \left(e_j^{\xi}(n)\right)^2$

be an empirical risk or per sample error cost

Next, to ~~keep~~ keep the derivation clean, assume $N=1$ and only 1 output neuron (however, we still use j to denote j^{th} output neuron in order to generalize later) forward signal propagation of an output neuron *j*



To update an output neuron weight w_{ji}
(j: output neuron, i: "input" neuron)

We need to compute the gradient so that:

$$\Delta w_{ji}(n) = -\eta \frac{\partial \mathcal{E}(n)}{\partial w_{ji}(n)}$$

apply chain rule:

$$\frac{\partial \mathcal{E}(n)}{\partial w_{ji}(n)} = \frac{\partial \mathcal{E}(n)}{\partial e_j(n)} \frac{\partial e_j(n)}{\partial y_j(n)} \frac{\partial y_j(n)}{\partial v_j(n)} \frac{\partial v_j(n)}{\partial w_{ji}(n)}$$

term by term:

$$\frac{\partial \mathcal{E}(n)}{\partial e_j(n)} = e_j(n); \quad \frac{\partial e_j(n)}{\partial y_j(n)} = -1; \quad \frac{\partial y_j(n)}{\partial v_j(n)} = \varphi_j'(v_j(n));$$

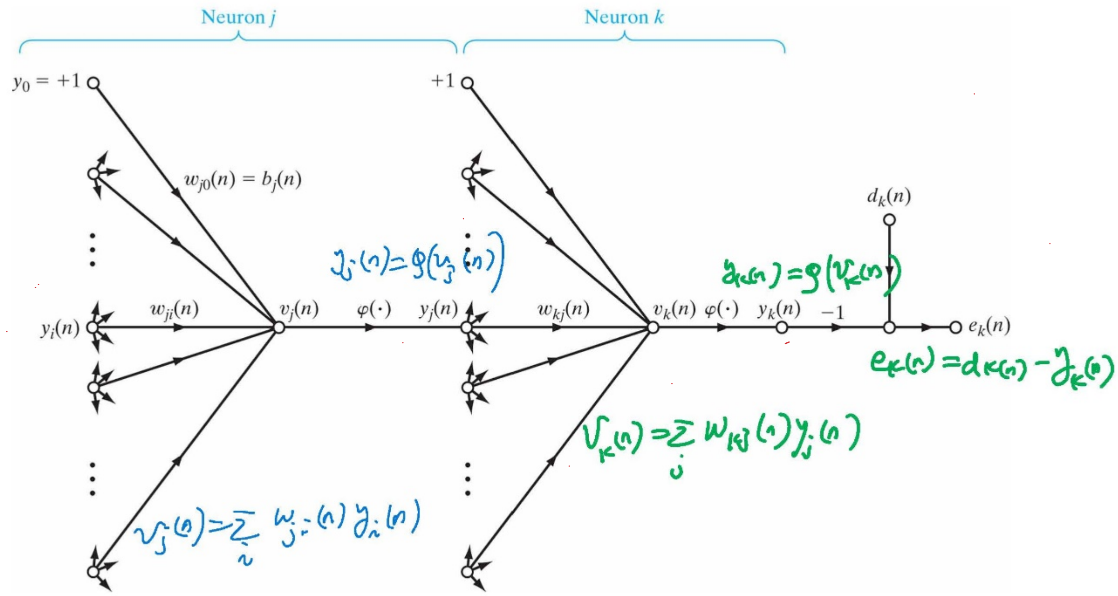
and $\frac{\partial v_j(n)}{\partial w_{ji}(n)} = y_i(n)$

let $\delta_j(n) = -\frac{\partial \mathcal{E}(n)}{\partial v_j(n)} = e_j(n)\, \varphi_j'(v_j(n))$

↑
local gradient

Then $\Delta w_{ji}(n) = -\eta \frac{\partial \mathcal{E}(n)}{\partial w_{ji}(n)} = \eta\, \delta_j(n)\, y_i(n)$

forward signal flow of an output neuron *k* connected to a hidden neuron *j*



To update weight $w_{ji}(n)$ of a neuron in a hidden layer,
(j: hidden neuron, i: "input" neuron)

we need to compute the gradient so that

$$\Delta w_{ji}(n) = -\eta \frac{\partial \mathcal{E}(n)}{\partial w_{ji}(n)}$$

(now let k denote an output neuron as will appear later)

Apply chain rule:

$$\frac{\partial \mathcal{E}(n)}{\partial w_{ji}(n)} = \frac{\partial \mathcal{E}(n)}{\partial y_j(n)} \cdot \frac{\partial y_j(n)}{\partial v_j(n)} \cdot \frac{\partial v_j(n)}{\partial w_{ji}(n)}$$

$$= \frac{\partial \mathcal{E}(n)}{\partial y_j(n)} \varphi_j'(v_j(n)) y_i(n)$$

let $\delta_j(n) = -\frac{\partial \mathcal{E}(n)}{\partial v_j(n)} = -\frac{\partial \mathcal{E}(n)}{\partial y_j(n)} \frac{\partial y_j(n)}{\partial v_j(n)} = -\frac{\partial \mathcal{E}(n)}{\partial y_j(n)} \cdot \varphi_j'(v_j(n))$

Then $\Delta w_{ji}(n) = -\eta \frac{\partial \mathcal{E}(n)}{\partial w_{ji}(n)} = \eta \cdot \delta_j(n) y_i(n)$

$\uparrow$ local gradient

To compute $\frac{\partial \mathcal{E}(n)}{\partial y_j(n)}$, we now take into accont k output neurons in order to generalize

i.e., $\mathcal{E}(n) = \frac{1}{2} \sum_k e_k^2(n)$

Then $\frac{\partial \mathcal{E}(n)}{\partial y_j(n)} = \sum_k e_k(n) \frac{\partial e_k(n)}{\partial v_k(n)} \frac{\partial v_k(n)}{\partial y_j(n)}$

Since $e_k(n) = d_k(n) - \varphi_k(v_k(n))$

$$\frac{\partial e_k(n)}{\partial v_k(n)} = -\varphi_k'(v_k(n))$$

since $\quad v_k(n) = \sum_{j=0}^{m} w_{kj}(n)\, y_j(n)$

$$\frac{\partial v_k(n)}{\partial y_j(n)} = w_{kj}(n)$$

putting it all together

$$\frac{\partial \mathcal{E}(n)}{\partial y_j(n)} = -\sum_k \underbrace{e_k(n)\,\varphi_k'(v_k(n))}_{\delta_k(n)}\, w_{kj}(n)$$

$$= -\sum_k \delta_k(n)\, w_{kj}(n)$$

Therefore $\delta_j(n) = \varphi_j'(v_j(n)) \sum_k \delta_k(n)\, w_{kj}(n)$

The back-propagation rule below applies to any connecting weights between an "output" neuron j and an "input" neuron i

$$\Delta w_{ji}(n) = \eta\, \delta_j(n)\, y_i(n)$$

where $\delta_j(n)$ depends on if j is an output layer neuron or a hidden layer neuron.

① if j is an output layer neuron,

$$\delta_j(n) = e_j(n)\, \varphi_j'(v_j(n))$$

② if j is a hidden layer neuron,

$$\delta_j(n) = \varphi_j'(v_j(n)) \sum_k \delta_k(n)\, w_{kj}(n)$$

Backward error flow and error propagation