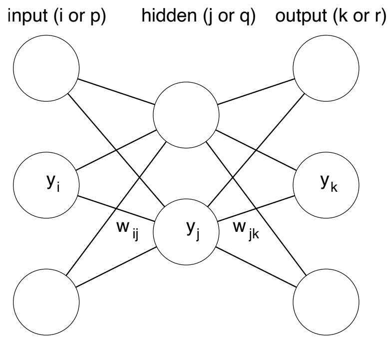
Derivation of the back propagation rule

Stephen J Eglen

September 5, 2023

1 Notation

In the following notation, pay attention to the subscript, as it tells you whether we refer to input (i), hidden (j) or output (k) units. I also use p, q and r for each of the three layers when talking about a specific connection. Although in principal each hidden and output neuron can have a bias unit, we often assume it is yet another input (which does not receive input) and so biases are omitted here.



Term	Meaning
a	Total input to a unit
y	Output of a unit
i, p	Subscript for input units
j, q	Subscript for hidden units
k, r	Subscript for output units
w_{ij}	Weight from input unit i to hidden unit j
w_{jk}	Weight from hidden unit j to output unit k
Δw_{ij}	Weight change from unit i to unit j

1.1 Calculating Network Activation

We will train the network to learn the association between given inputs x_i and their desired outputs t_k. Unit activations, a_j, and outputs, y_j, of hidden layer units are calculated by:

$$\text{Hidden layer} \quad a_j = \sum_i w_{ij} y_i \tag{1}$$

$$y_j = g(a_j) \tag{2}$$

$$\text{Output layer} \quad a_k = \sum_j w_{jk} y_j \tag{3}$$

$$y_k = g(a_k) \tag{4}$$

$$\text{Error function} \quad E = \frac{1}{2} \sum_k (t_k - y_k)^2 \tag{5}$$

$g(\cdot)$ is some transfer function, typically sigmoidal or the identity. We wish to make a learning rule:

$$\Delta w_{ij} = -\eta \frac{\partial E}{\partial w_{ij}}$$

2 Back propagation

Step 2a: Updating weights from hidden to output layer, w_{jk}

If we pick one weight w_{qr} to adapt, then we want to see how the error changes in response to a change to that weight. So, from 5:

$$\frac{\partial E}{\partial w_{qr}} = -\sum_k (t_k - y_k) \frac{\partial y_k}{\partial w_{qr}}$$

If we change w_{qr}, the only y_k that changes is when when $k = r$, so $\frac{\partial y_k}{\partial w_{qr}} = 0$ except when $k = r$:

$$\frac{\partial E}{\partial w_{qr}} = -(t_r - y_r) \frac{\partial y_r}{\partial w_{qr}}$$

Now recall that $y_r = g(a_r)$ and $a_r = \sum_j w_{jr} y_j$ so

$$\begin{aligned}
\frac{\partial y_r}{\partial w_{qr}} &= g'(a_r) \frac{\partial a_r}{\partial w_{qr}} \\
\frac{\partial a_r}{\partial w_{qr}} &= y_q \qquad \text{(''partial trick'')} \\
\frac{\partial E}{\partial w_{qr}} &= -(t_r - y_r) g'(a_r) y_q \\
\text{Let} \qquad \delta_k &= g'(a_k)(t_k - y_k) \\
\text{So} \qquad \frac{\partial E}{\partial w_{qr}} &= -\delta_k y_q
\end{aligned}$$

Step 2b and 2c: Updating weights from input to hidden layer, w_{pq}

If we change any weight w_{pq}, this can change the activation of hidden unit y_j and so affect all output layer units, not just one output unit as for hidden-to-output layer.

$$\frac{\partial E}{\partial w_{pq}} = -\sum_k (t_k - y_k) \frac{\partial y_k}{\partial w_{pq}}$$

Apply the chain rule via y_q to get the derivative

$$\begin{aligned}
\frac{\partial y_k}{\partial w_{pq}} &= \frac{\partial y_k}{\partial y_q} \frac{\partial y_q}{\partial w_{pq}} \\
\frac{\partial y_k}{\partial y_q} &= g'(a_k) \frac{\partial a_k}{\partial y_q} = g'(a_k) w_{qk} \qquad \text{from 3, 4 and "partial trick"} \\
\frac{\partial y_q}{\partial w_{pq}} &= g'(a_q) \frac{\partial a_q}{\partial w_{pq}} = g'(a_q) y_p \qquad \text{from 1, 2}
\end{aligned}$$

Plugging them all back together we get:

$$\begin{aligned}
\frac{\partial E}{\partial w_{pq}} &= -\sum_k (t_k - y_k) \times g'(a_k) w_{qk} \times g'(a_q) y_p \\
&= -y_p g'(a_q) \sum_k (t_k - y_k) g'(a_k) w_{qk} \\
&= -y_p g'(a_q) \sum_k \delta_k w_{qk} \\
\text{Let} \qquad \delta_q &= g'(a_q) \sum_k \delta_k w_{qk} \\
\frac{\partial E}{\partial w_{pq}} &= -\delta_q y_p
\end{aligned}$$

So, errors **back-propagate**: travel in opposite direction to activity to generate internal errors δ_q.