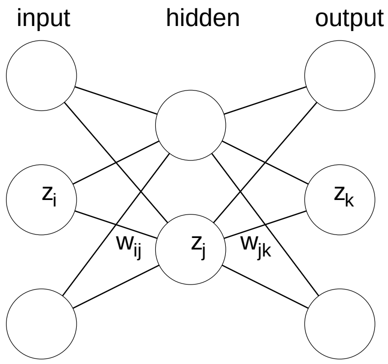
Derivation of the back propagation rule

Stephen J Eglen

October 29, 2020

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
x	Total input to a unit
z	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 z_i and their desired outputs t_k. Unit activations, x_j, and outputs, z_j, of hidden layer units are calculated by:

$$\text{Hidden layer} \quad x_j = \sum_i w_{ij} z_i \tag{1}$$

$$z_j = g(x_j) \tag{2}$$

$$\text{Output layer} \quad x_k = \sum_j w_{jk} z_j \tag{3}$$

$$z_k = g(x_k) \tag{4}$$

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

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

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

1.2 Updating weights from hidden to output layer, w_{jk}

If we change any weight w_{jk}, the only activation that will change is z_k, so $\frac{\partial E}{\partial w_{jk}}$ only involves one output unit, k. So, from 5:

$$
\begin{aligned}
\frac{\partial E}{\partial w_{jk}} &= -(t_k - z_k)\frac{\partial z_k}{\partial w_{jk}} \\
\frac{\partial z_k}{\partial w_{jk}} &= g'(x_k)\frac{\partial x_k}{\partial w_{jk}} = g'(x_k)z_j \quad \text{(from 3, 4)} \\
\frac{\partial E}{\partial w_{jk}} &= -(t_k - z_k)g'(x_k)z_j \\
\text{Let} \qquad \delta_k &= g'(x_k)(t_k - z_k) \\
\text{So} \qquad \frac{\partial E}{\partial w_{jk}} &= -\delta_k z_j
\end{aligned}
$$

1.3 Updating weights from input to hidden layer, w_{ij}

If we change any weight w_{ij}, this can change the activation of hidden unit z_j and so affect all output layer units, not just one output unit.

$$
\frac{\partial E}{\partial w_{ij}} = -\sum_k (t_k - z_k)\frac{\partial z_k}{\partial w_{ij}}
$$

Apply the chain rule to get the derivative

$$
\begin{aligned}
\frac{\partial z_k}{\partial w_{ij}} &= \frac{\partial z_k}{\partial z_j}\frac{\partial z_j}{\partial w_{ij}} \\
\frac{\partial z_k}{\partial z_j} &= g'(x_k)\frac{\partial x_k}{\partial z_j} = g'(x_k)w_{jk} \qquad \text{from 3, 4} \\
\frac{\partial z_j}{\partial w_{ij}} &= g'(x_j)\frac{\partial x_j}{\partial w_{ij}} = g'(x_j)z_i \qquad \text{from 1, 2}
\end{aligned}
$$

Plugging them all back together we get:

$$
\begin{aligned}
\frac{\partial E}{\partial w_{ij}} &= -\sum_k (t_k - z_k) \times g'(x_k)w_{jk} \times g'(x_j)z_i \\
&= -z_i g'(x_j)\sum_k (t_k - z_k)g'(x_k)w_{jk} \\
&= -z_i g'(x_j)\sum_k \delta_k w_{jk} \\
\text{Let} \qquad \delta_j &= g'(x_j)\sum_k \delta_k w_{jk} \\
\frac{\partial E}{\partial w_{ij}} &= -\delta_j z_i
\end{aligned}
$$

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

References