CSC 446 Note: Lecture 6

Lam Tran

February 9, 2012

Multi Layer Perceptron

In a two layer neural network, the input, hidden, and output variables are represented by nodes, and the weight parameters are represented by links between the nodes. The arrow of the links indicate the direction of information flow through the network during forward propagation. The overall two stage network function takes the form

$$t_k(\mathbf{x}, \mathbf{w}) = \sigma \left(\sum_{j=0} w_{jk} \; g \left(\sum_{i=0} w_{ij} \; x_i \right) \right), \tag{1}$$

where

$$z_j = g \left(\sum_{i=0} w_{ij} \; x_i \right)$$

are known as the hidden units, w_{ij} are the weights, x_i are the input variables, and the σ and g are activation functions. Non-linear functions are usually chosen for activation functions such as tanh and sigmoid functions.

Training A Network: Error Backpropagation

Given a training set of input vector $\mathbf{x}^n$ and its target vector $\mathbf{y}^n$ for $n = 1 \dots N$, we want to minimize the error function

$$E^n(\mathbf{w}) = \frac{1}{2} \sum_{i=1}^{K} \parallel t_i^n - y_i^n \parallel^2 = \sum_{i=1}^{K} E_i^n. \tag{2}$$

The response function $\mathbf{t}_i^n$ take take form of (1) but to keep notation uncluttered we will omit the input parameters. To minimize the error function of a no hidden layer network, we apply

the chain rule for the partial derivatives with respect to (w.r.t.) the weights w_{ij} to get

$$\begin{aligned}
\frac{\partial E^n}{\partial w_{ij}} &= \sum_{j'} \frac{\partial E^n}{\partial t^n_{j'}} \frac{\partial t^n_{j'}}{\partial w_{ij}} \\
&= \left(t^n_j - y^n_j\right) \frac{\partial t^n}{\partial a_j} \frac{\partial a_j}{\partial w_{ij}} \\
&= \left(t^n_j - y^n_j\right) g'(a_j) x_i \\
&= \underbrace{\left(t^n_j - y^n_j\right) g'(a_j)}_{\delta_j} x_i \\
&= \delta_j x_i
\end{aligned} \tag{3}$$

where

$$a_j = \sum_i w_{ij}\ x_i$$

is the sum of the weighted input feeds. The derivative respect to the matrix $\mathbf{W}$ is

$$\frac{\partial E^n}{\partial \mathbf{W}} = \left[\frac{\partial E^n}{\partial w_{ij}}\right]_{ij} = [\delta_j x_i]_{ij} = \mathbf{x}\delta^T.$$

We now focus on training on a neural network with a hidden layer (multi layer) that takes the same form as (2) and minimize the error w.r.t. the weights w_{ij} in this form

$$\begin{aligned}
\frac{\partial E^n}{\partial w_{ij}} &= \sum_k \frac{\partial E^n}{\partial t^n_k} \frac{\partial t^n_k}{\partial w_{ij}} = \sum_k \frac{\partial E^n}{\partial t^n_k} \frac{\partial t^n_k}{\partial a_k} \frac{\partial a_k}{\partial w_{ij}} \\
&= \sum_k (t_k - y_k) \left(g'(a_k)\ \frac{\partial}{\partial w_{ij}} \sum_{j'} w_{j'k}\ g(a_{j'}) \right) \\
&= \sum_k (t_k - y_k) \left(g'(a_k) w_{jk}\ \frac{\partial}{\partial w_{ij}}\ g(a_j) \right) \\
&= \sum_k (t_k - y_k) \left(g'(a_k) w_{jk}\ g'(a_j) x_i \right) \\
&= \sum_k \underbrace{(t_k - y_k) g'(a_k) w_{jk}\ g'(a_j)}_{\delta_j} x_i \\
&= \delta_j x_i
\end{aligned}$$

The derivatives with respect to the first and second layer weights are given by

$$\frac{\partial E^n}{\partial w_{ij}} = \delta_j x_i \qquad \text{and} \qquad \frac{\partial E_n}{\partial w_{jk}} = \delta_k z_j.$$

Thus the training algorithm consists of calculating weight for n

- Calculate weight
  for n
  $\mathbf{W} += \eta \frac{\partial E^n}{\partial \mathbf{W}}$.