The back-propagation algorithm

January 8, 2012

Ryan

The neuron

- The sigmoid equation is what is typically used as a transfer function between neurons. It is similar to the step function, but is continuous and differentiable.

The neuron

- The sigmoid equation is what is typically used as a transfer function between neurons. It is similar to the step function, but is continuous and differentiable.
- $$\sigma(x) = \frac{1}{1+e^{-x}} \tag{1}$$

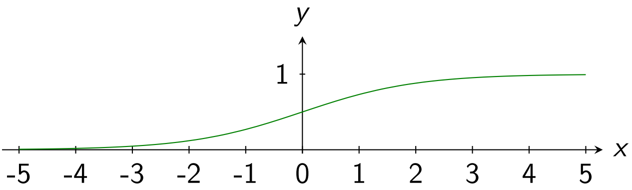

Figure: The Sigmoid Function

The neuron

- The sigmoid equation is what is typically used as a transfer function between neurons. It is similar to the step function, but is continuous and differentiable.
- $$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{1}$$

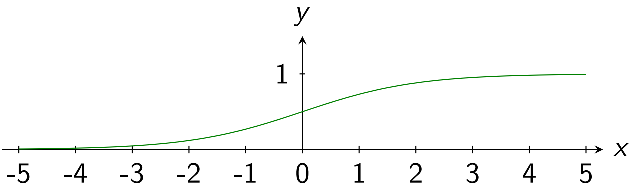

Figure: The Sigmoid Function

- One useful property of this transfer function is the simplicity of computing it's derivative. Let's do that now...

The derivative of the sigmoid transfer function

$$\frac{d}{dx}\sigma(x) \quad = \quad \frac{d}{dx}\left(\frac{1}{1+e^{-x}}\right)$$

The derivative of the sigmoid transfer function

$$
\begin{aligned}
\frac{d}{dx}\sigma(x) &= \frac{d}{dx}\left(\frac{1}{1+e^{-x}}\right) \\
&= \frac{e^{-x}}{(1+e^{-x})^2}
\end{aligned}
$$

The derivative of the sigmoid transfer function

$$
\begin{aligned}
\frac{d}{dx}\sigma(x) &= \frac{d}{dx}\left(\frac{1}{1+e^{-x}}\right) \\
&= \frac{e^{-x}}{(1+e^{-x})^2} \\
&= \frac{1+e^{-x} \quad -1}{(1+e^{-x})^2}
\end{aligned}
$$

The derivative of the sigmoid transfer function

$$
\begin{aligned}
\frac{d}{dx}\sigma(x) &= \frac{d}{dx}\left(\frac{1}{1+e^{-x}}\right) \\
&= \frac{e^{-x}}{(1+e^{-x})^2} \\
&= \frac{(1+e^{-x})-1}{(1+e^{-x})^2}
\end{aligned}
$$

The derivative of the sigmoid transfer function

$$
\begin{aligned}
\frac{d}{dx}\sigma(x) &= \frac{d}{dx}\left(\frac{1}{1+e^{-x}}\right) \\
&= \frac{e^{-x}}{(1+e^{-x})^2} \\
&= \frac{(1+e^{-x})-1}{(1+e^{-x})^2} \\
&= \frac{1+e^{-x}}{(1+e^{-x})^2} - \frac{1}{(1+e^{-x})^2}
\end{aligned}
$$

The derivative of the sigmoid transfer function

$$
\begin{aligned}
\frac{d}{dx}\sigma(x) &= \frac{d}{dx}\left(\frac{1}{1+e^{-x}}\right) \\
&= \frac{e^{-x}}{(1+e^{-x})^2} \\
&= \frac{(1+e^{-x})-1}{(1+e^{-x})^2} \\
&= \frac{1+e^{-x}}{(1+e^{-x})^2} - \left(\frac{1}{1+e^{-x}}\right)^2
\end{aligned}
$$

The derivative of the sigmoid transfer function

$$
\begin{aligned}
\frac{d}{dx}\sigma(x) &= \frac{d}{dx}\left(\frac{1}{1+e^{-x}}\right) \\
&= \frac{e^{-x}}{(1+e^{-x})^2} \\
&= \frac{(1+e^{-x})-1}{(1+e^{-x})^2} \\
&= \frac{1+e^{-x}}{(1+e^{-x})^2} - \left(\frac{1}{1+e^{-x}}\right)^2 \\
&= \sigma(x) - \sigma(x)^2
\end{aligned}
$$

The derivative of the sigmoid transfer function

$$
\begin{aligned}
\frac{d}{dx}\sigma(x) &= \frac{d}{dx}\left(\frac{1}{1+e^{-x}}\right) \\
&= \frac{e^{-x}}{(1+e^{-x})^2} \\
&= \frac{(1+e^{-x})-1}{(1+e^{-x})^2} \\
&= \frac{1+e^{-x}}{(1+e^{-x})^2} - \left(\frac{1}{1+e^{-x}}\right)^2 \\
&= \sigma(x) - \sigma(x)^2 \\
\sigma' &= \sigma(1-\sigma)
\end{aligned}
$$

Single input neuron

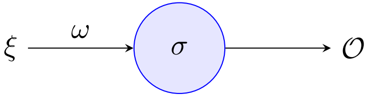

Figure: A Single-Input Neuron

In the above figure (2) you can see a diagram representing a single neuron with only a single input. The equation defining the figure is:

$$\mathcal{O} = \sigma(\xi\omega)$$

Single input neuron

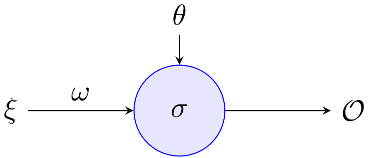

Figure: A Single-Input Neuron

In the above figure (2) you can see a diagram representing a single neuron with only a single input. The equation defining the figure is:

$$\mathcal{O} = \sigma(\xi\omega + \theta)$$

Multiple input neuron

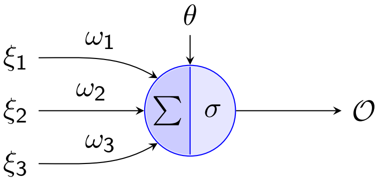

Figure: A Multiple Input Neuron

Figure 3 is the diagram representing the following equation:

$$\mathcal{O} = \sigma(\omega_1\xi_1 + \omega_2\xi_2 + \omega_3\xi_3 + \theta)$$

A neural network

Figure: A layer

A neural network

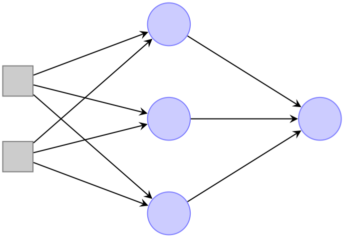

Figure: A neural network

A neural network

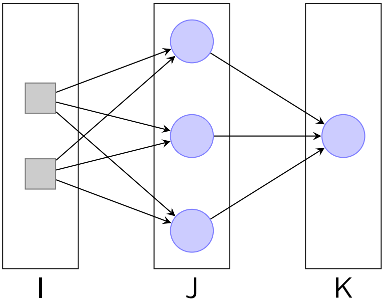

Figure: A neural network

The back propagation algorithm

Notation

- x_j^ℓ : Input to node j of layer ℓ

The back propagation algorithm

Notation

- x_j^ℓ : Input to node j of layer ℓ
- W_{ij}^ℓ : Weight from layer $\ell - 1$ node i to layer ℓ node j

The back propagation algorithm

Notation

- x_j^ℓ : Input to node j of layer ℓ
- W_{ij}^ℓ : Weight from layer $\ell - 1$ node i to layer ℓ node j
- $\sigma(x) = \frac{1}{1+e^{-x}}$: Sigmoid Transfer Function

The back propagation algorithm

Notation

- x_j^ℓ : Input to node j of layer ℓ
- W_{ij}^ℓ : Weight from layer $\ell - 1$ node i to layer ℓ node j
- $\sigma(x) = \frac{1}{1+e^{-x}}$: Sigmoid Transfer Function
- θ_j^ℓ : Bias of node j of layer ℓ

The back propagation algorithm

Notation

- x_j^ℓ : Input to node j of layer ℓ
- W_{ij}^ℓ : Weight from layer $\ell - 1$ node i to layer ℓ node j
- $\sigma(x) = \frac{1}{1+e^{-x}}$: Sigmoid Transfer Function
- θ_j^ℓ : Bias of node j of layer ℓ
- $\mathcal{O}_j^\ell$: Output of node j in layer ℓ

The back propagation algorithm

Notation

- x_j^ℓ : Input to node j of layer ℓ
- W_{ij}^ℓ : Weight from layer $\ell - 1$ node i to layer ℓ node j
- $\sigma(x) = \frac{1}{1+e^{-x}}$: Sigmoid Transfer Function
- θ_j^ℓ : Bias of node j of layer ℓ
- $\mathcal{O}_j^\ell$: Output of node j in layer ℓ
- t_j : Target value of node j of the output layer

The error calculation

Given a set of training data points t_k and output layer output $\mathcal{O}_k$ we can write the error as

$$E = \frac{1}{2}\sum_{k\in K}(\mathcal{O}_k - t_k)^2$$

We let the error of the network for a single training iteration be denoted by E. We want to calculate $\frac{\partial E}{\partial W_{jk}^{\ell}}$, the rate of change of the error with respect to the given connective weight, so we can minimize it.

Now we consider two cases: The node is an output node, or it is in a hidden layer...

Output layer node

$$\frac{\partial E}{\partial W_{jk}} =$$

Output layer node

$$\frac{\partial E}{\partial W_{jk}} = \frac{\partial}{\partial W_{jk}} \frac{1}{2} \sum_{k \in K} (\mathcal{O}_k - t_k)^2$$

Output layer node

$$\frac{\partial E}{\partial W_{jk}} = (\mathcal{O}_k - t_k)\frac{\partial}{\partial W_{jk}}\mathcal{O}_k$$

Output layer node

$$\frac{\partial E}{\partial W_{jk}} = (\mathcal{O}_k - t_k)\frac{\partial}{\partial W_{jk}}\sigma(x_k)$$

Output layer node

$$\frac{\partial E}{\partial W_{jk}} = (\mathcal{O}_k - t_k)\sigma(x_k)(1 - \sigma(x_k))\frac{\partial}{\partial W_{jk}} x_k$$

Output layer node

$$\frac{\partial E}{\partial W_{jk}} = (\mathcal{O}_k - t_k)\mathcal{O}_k(1 - \mathcal{O}_k)\mathcal{O}_j$$

Output layer node

$$\frac{\partial E}{\partial W_{jk}} = (\mathcal{O}_k - t_k)\mathcal{O}_k(1 - \mathcal{O}_k)\mathcal{O}_j$$

For notation purposes I will define δ_k to be the expression $(\mathcal{O}_k - t_k)\mathcal{O}_k(1 - \mathcal{O}_k)$, so we can rewrite the equation above as

$$\frac{\partial E}{\partial W_{jk}} = \mathcal{O}_j \delta_k$$

where

$$\delta_k = \mathcal{O}_k(1 - \mathcal{O}_k)(\mathcal{O}_k - t_k)$$

Hidden layer node

$$\frac{\partial E}{\partial W_{ij}} =$$

Hidden layer node

$$\frac{\partial E}{\partial W_{ij}} = \frac{\partial}{\partial W_{ij}} \frac{1}{2} \sum_{k \in K} (\mathcal{O}_k - t_k)^2$$

Hidden layer node

$$\frac{\partial E}{\partial W_{ij}} = \sum_{k \in K} (\mathcal{O}_k - t_k) \frac{\partial}{\partial W_{ij}} \mathcal{O}_k$$

Hidden layer node

$$\frac{\partial E}{\partial W_{ij}} = \sum_{k \in K} (\mathcal{O}_k - t_k) \frac{\partial}{\partial W_{ij}} \sigma(x_k)$$

Hidden layer node

$$\frac{\partial E}{\partial W_{ij}} = \sum_{k \in K} (\mathcal{O}_k - t_k)\sigma(x_k)(1 - \sigma(x_k)) \frac{\partial x_k}{\partial W_{ij}}$$

Hidden layer node

$$\frac{\partial E}{\partial W_{ij}} = \sum_{k \in K} (\mathcal{O}_k - t_k)\mathcal{O}_k(1 - \mathcal{O}_k)\frac{\partial x_k}{\partial \mathcal{O}_j} \cdot \frac{\partial \mathcal{O}_j}{\partial W_{ij}}$$

Hidden layer node

$$\frac{\partial E}{\partial W_{ij}} = \sum_{k \in K} (\mathcal{O}_k - t_k)\mathcal{O}_k(1 - \mathcal{O}_k) W_{jk} \frac{\partial \mathcal{O}_j}{\partial W_{ij}}$$

Hidden layer node

$$\frac{\partial E}{\partial W_{ij}} = \frac{\partial \mathcal{O}_j}{\partial W_{ij}} \sum_{k \in K} (\mathcal{O}_k - t_k)\mathcal{O}_k(1 - \mathcal{O}_k)W_{jk}$$

Hidden layer node

$$\frac{\partial E}{\partial W_{ij}} = \mathcal{O}_j(1 - \mathcal{O}_j)\frac{\partial x_j}{\partial W_{ij}} \sum_{k \in K} (\mathcal{O}_k - t_k)\mathcal{O}_k(1 - \mathcal{O}_k)W_{jk}$$

Hidden layer node

$$\frac{\partial E}{\partial W_{ij}} = \mathcal{O}_j(1 - \mathcal{O}_j)\mathcal{O}_i \sum_{k \in K} (\mathcal{O}_k - t_k)\mathcal{O}_k(1 - \mathcal{O}_k)W_{jk}$$

Hidden layer node

$$\frac{\partial E}{\partial W_{ij}} = \mathcal{O}_j(1 - \mathcal{O}_j)\mathcal{O}_i \sum_{k \in K} (\mathcal{O}_k - t_k)\mathcal{O}_k(1 - \mathcal{O}_k)W_{jk}$$

But, recalling our definition of δ_k we can write this as

$$\frac{\partial E}{\partial W_{ij}} = \mathcal{O}_i\mathcal{O}_j(1 - \mathcal{O}_j) \sum_{k \in K} \delta_k W_{jk}$$

Hidden layer node

$$\frac{\partial E}{\partial W_{ij}} = \mathcal{O}_j(1 - \mathcal{O}_j)\mathcal{O}_i \sum_{k \in K} (\mathcal{O}_k - t_k)\mathcal{O}_k(1 - \mathcal{O}_k)W_{jk}$$

But, recalling our definition of δ_k we can write this as

$$\frac{\partial E}{\partial W_{ij}} = \mathcal{O}_i\mathcal{O}_j(1 - \mathcal{O}_j) \sum_{k \in K} \delta_k W_{jk}$$

Similar to before we will now define all terms besides the $\mathcal{O}_i$ to be δ_j, so we have

$$\frac{\partial E}{\partial W_{ij}} = \mathcal{O}_i\delta_j$$

How weights affect errors

For an output layer node $k \in K$

$$\frac{\partial E}{\partial W_{jk}} = \mathcal{O}_j \delta_k$$

where

$$\delta_k = \mathcal{O}_k(1 - \mathcal{O}_k)(\mathcal{O}_k - t_k)$$

For a hidden layer node $j \in J$

$$\frac{\partial E}{\partial W_{ij}} = \mathcal{O}_i \delta_j$$

where

$$\delta_j = \mathcal{O}_j(1 - \mathcal{O}_j) \sum_{k \in K} \delta_k W_{jk}$$

What about the bias?

If we incorporate the bias term θ into the equation you will find that

$$\frac{\partial \mathcal{O}}{\partial \theta} = \mathcal{O}(1 - \mathcal{O})\frac{\partial \theta}{\partial \theta}$$

and because $\partial\theta/\partial\theta = 1$ we view the bias term as output from a node which is always one.

What about the bias?

If we incorporate the bias term θ into the equation you will find that

$$\frac{\partial \mathcal{O}}{\partial \theta} = \mathcal{O}(1 - \mathcal{O})\frac{\partial \theta}{\partial \theta}$$

and because $\partial\theta/\partial\theta = 1$ we view the bias term as output from a node which is always one.

This holds for any layer ℓ we are concerned with, a substitution into the previous equations gives us that

$$\frac{\partial E}{\partial \theta} = \delta_\ell$$

(because the $\mathcal{O}_\ell$ is replacing the output from the "previous layer")

The back propagation algorithm

1. Run the network forward with your input data to get the network output
2. For each output node compute
$$\delta_k = \mathcal{O}_k(1 - \mathcal{O}_k)(\mathcal{O}_k - t_k)$$
3. For each hidden node calulate
$$\delta_j = \mathcal{O}_j(1 - \mathcal{O}_j) \sum_{k \in K} \delta_k W_{jk}$$
4. Update the weights and biases as follows
Given
$$\begin{aligned} \Delta W &= -\eta \delta_\ell \mathcal{O}_{\ell-1} \\ \Delta \theta &= -\eta \delta_\ell \end{aligned}$$
apply
$$\begin{aligned} W + \Delta W &\to W \\ \theta + \Delta \theta &\to \theta \end{aligned}$$