# Backpropagation Mathematics

Brady Neal

August 2015

## 1 Definitions

If you are ever reading through the math in here and realize you don't know what something such as $w_{ij}^{l}$ means, just come back to this section and you'll find a concise definition.

Superscripts will always refer to the layer, and subscripts will always refer to the neuron in that layer. All indexing will start from 1, not 0.

$b_i^l$ - bias of the $i^{\text{th}}$ neuron in the $l^{\text{th}}$ layer

$w_{ij}^l$ - weight from the $j^{\text{th}}$ neuron in layer $l-1$ to the $i^{\text{th}}$ neuron in layer $l$

$z_i^l$ - weighted input to the activation function of the $i^{\text{th}}$ neuron in the $l^{\text{th}}$ layer

$$z_i^l = \sum_j w_{ij}^l a_j^{l-1} + b_i^l$$

Vectorized version:

$$z^l = w^l a^{l-1} + b^l$$

The activation function used is the sigmoid function:

$$\sigma(z) = \frac{1}{1+e^{-z}}$$

$a_i^l$ - activation of the $i^{\text{th}}$ neuron in the $l^{\text{th}}$ layer

$$a_i^l = \sigma(z_i^l) = \sigma\left(\sum_j w_{ij}^l a_j^{l-1} + b_i^l\right)$$

Vectorized version:

$$a^l = \sigma(z^l) = \sigma(w^l a^{l-1} + b^l)$$

$L$ - number of layers in the neural network

$x^{(i)}$ - feature vector corresponding to the $i^{\text{th}}$ training example

$y^{(i)}$ - correct output value for the $i^{\text{th}}$ training example

$a = a^L$ - hypothesis (prediction) of the neural network for a training example (i.e. $a^{(i)}$ is the neural network output that corresponds to $x^{(i)}$)

$\odot$ - the Hadamard product operator (element-wise multiplication between n-dimensional arrays)

The cost function used is the cross entropy error function:

$$C = -\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)}\ln a^{(i)} + (1-y^{(i)})\ln\left(1-a^{(i)}\right)\right]$$

where $m$ is the number of training examples, $a^{(i)}$ is the output layer of the neural network for the $i^{\text{th}}$ training example, and $y^{(i)}$ is the correct output for the $i^{\text{th}}$ training example.

The above cost function is over all training examples $x^{(i)}$. Both $y^{(i)}$ and $a^{(i)}$ are commonly vectors, so it is more explicitly written as

$$C = -\frac{1}{m}\sum_{i=1}^{m}\sum_{j}\left[y_j^{(i)}\ln a_j^{(i)} + (1-y_j^{(i)})\ln\left(1-a_j^{(i)}\right)\right]$$

where $y_j^{(i)}$ and $a_j^{(i)}$ are the $j^{\text{th}}$ elements in the $y^{(i)}$ and $a^{(i)}$ vectors respectively.

However, I will mostly be referring to the cost of a single training example in vectorized form, $C_x$:

$$C_x = y\ln a + (1-y)\ln(1-a)$$

This simplifies the notation a great deal as we no longer have to worry about the many indices and summations. Therefore, when I say $C$ below, I'm often referring to $C_x$.

# 2 Proofs

Because we want to find the weights and biases that will minimize the cost function, the ultimate goal of backpropagation is to find the following:

$$\frac{\partial C}{\partial w_{ij}^l} \text{ and } \frac{\partial C}{\partial b_i^l}$$

We need to get four fundamental equations first:

$$\begin{aligned}
\delta_i^L &= \frac{\partial C}{\partial a_i^L}\sigma'(z_i^L) & \delta^L &= \nabla_a C \odot \sigma'(z^L) \\
\delta_j^l &= \sum_i w_{ij}^{l+1}\delta_i^{l+1}\sigma'(z_j^l) & \delta^l &= \left(\left(w^{l+1}\right)^T \delta^{l+1}\right) \odot \sigma'(z^l) \\
\frac{\partial C}{\partial w_{ij}^l} &= \delta_i^l a_j^{l-1} & \frac{\partial C}{\partial w^l} &= \delta^l \left(a^{l-1}\right)^T \\
\frac{\partial C}{\partial b_i^l} &= \delta_i^l & \frac{\partial C}{\partial b^l} &= \delta^l
\end{aligned}$$

## 2.1 An Intermediate Quantity: $\delta_i^l$

In order to get $\frac{\partial C}{\partial w_{ij}^l}$ and $\frac{\partial C}{\partial b_i^l}$, we must first define an intermediate quantity that we'll call the error:

$$\delta_i^l = \frac{\partial C}{\partial z_i^l}$$

The first step is to get the error in the output layer $L$:

$$\delta_i^L = \frac{\partial C}{\partial z_i^L} = \frac{\partial C}{\partial a_i^L}\frac{da_i^L}{dz_i^L} \text{ or } \frac{\partial C}{\partial a_i^L}\sigma'(z_i^L)$$

Vectorized:

$$\delta^L = \nabla_a C \odot \sigma'(z^L)$$

Now that we have the error in the output layer, $\delta^L$, we need an equation that relates $\delta^l$ to $\delta^{l+1}$ in order to inductively solve for $\delta^l$.

$$\delta_j^l = \frac{\partial C}{\partial z_j^l} = \sum_i \frac{\partial C}{\partial z_i^{l+1}}\frac{\partial z_i^{l+1}}{\partial z_j^l} = \sum_i \delta_i^{l+1}\frac{\partial z_i^{l+1}}{\partial z_j^l}$$

In order to get $\frac{\partial z_i^{l+1}}{\partial z_j^l}$, recall $z_i^{l+1}$:

$$z_i^{l+1} = \sum_j w_{ij}^{l+1} a_j^l + b_i^{l+1} = \sum_j w_{ij}^{l+1} \sigma(z_j^l) + b_i^{l+1}$$

$$\frac{\partial z_i^{l+1}}{\partial z_j^l} = w_{ij}^{l+1} \sigma'(z_j^l)$$

Thus,

$$\delta_j^l = \sum_i \delta_i^{l+1} \frac{\partial z_i^{l+1}}{\partial z_j^l} = \sum_i \delta_i^{l+1} w_{ij}^{l+1} \sigma'(z_j^l) = \sum_i w_{ij}^{l+1} \delta_i^{l+1} \sigma'(z_j^l)$$

Vectorized:

$$\delta^l = \left( \left( w^{l+1} \right)^T \delta^{l+1} \right) \odot \sigma'(z^l)$$

## 2.2 $\frac{\partial C}{\partial w_{ij}^l}$ and $\frac{\partial C}{\partial b_i^l}$ in terms of $\delta_i^l$

$$\frac{\partial C}{\partial w_{ij}^l} = \frac{\partial C}{\partial z_i^l} \frac{\partial z_i^l}{\partial w_{ij}^l} = \delta_i^l \frac{\partial z_i^l}{\partial w_{ij}^l}$$

$$\frac{\partial C}{\partial b_i^l} = \frac{\partial C}{\partial z_i^l} \frac{\partial z_i^l}{\partial b_i^l} = \delta_i^l \frac{\partial z_i^l}{\partial b_i^l}$$

In order to get $\frac{\partial z_i^l}{\partial w_{ij}^l}$ and $\frac{\partial z_i^l}{\partial b_i^l}$, recall $z_i^l$:

$$z_i^l = \sum_j w_{ij}^l a_j^{l-1} + b_i^l$$

$$\frac{\partial z_i^l}{\partial w_{ij}^l} = a_j^{l-1} \qquad\qquad \frac{\partial z_i^l}{\partial b_i^l} = 1$$

Thus,

$$\frac{\partial C}{\partial w_{ij}^l} = \delta_i^l \frac{\partial z_i^l}{\partial w_{ij}^l} = \delta_i^l a_j^{l-1}$$

$$\frac{\partial C}{\partial b_i^l} = \delta_i^l \frac{\partial z_i^l}{\partial b_i^l} = \delta_i^l$$

Vectorized:

$$\frac{\partial C}{\partial w^l} = \delta^l \left( a^{l-1} \right)^T$$

$$\frac{\partial C}{\partial b^l} = \delta^l$$

## 3 Specific Activation Function and Cost Function

All of the math in the previous section refers to the activation function as $\sigma$ and the cost function as $C$. Because we defined a specific activation function and cost function at the beginning of the paper, we will now work out the math that will be necessary for an implementation of backpropagation.

In other words, $\sigma'(z_i^L)$ and $\frac{\partial C}{\partial a_i^L}$ are part of the fundamental equations above, so we'll calculate them with respect to our chosen activation function and cost function.

### 3.1 Derivative of The Sigmoid Function

Recall the sigmoid function:

$$\sigma(z) = \frac{1}{1+e^{-z}} = \left(1+e^{-z}\right)^{-1}$$

$$\sigma'(z) = -\left(1+e^{-z}\right)^{-2} \cdot -e^{-z} = e^{-z}\left(1+e^{-z}\right)^{-2}$$

$$\sigma'(z) = \frac{e^{-z}}{\left(1+e^{-z}\right)^{2}} = \frac{1}{1+e^{-z}} \cdot \frac{e^{-z}}{1+e^{-z}}$$

$$\sigma'(z) = \frac{1}{1+e^{-z}}\left(1 - \frac{1}{1+e^{-z}}\right)$$

Thus,

$$\sigma'(z) = \sigma(z)\left(1-\sigma(z)\right)$$

### 3.2 Partial Derivative of The Cross Entropy Error Function

Recall the cross entropy error function:

$$C = -\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)}\ln a^{(i)} + (1-y^{(i)})\ln\left(1-a^{(i)}\right)\right]$$

where $m$ is the number of training examples, $a^{(i)}$ is the output layer of the neural network for the $i^{\text{th}}$ training example, and $y^{(i)}$ is the correct output for the $i^{\text{th}}$ training example.

$$\frac{\partial C}{\partial a^{(i)}} = -\left(y^{(i)} \cdot \frac{1}{a^{(i)}} + (1-y^{(i)}) \cdot \frac{1}{1-a^{(i)}} \cdot -1\right)$$

$$\frac{\partial C}{\partial a^{(i)}} = \frac{\left(1-y^{(i)}\right)}{1-a^{(i)}} - \frac{y^{(i)}}{a^{(i)}} = \frac{a^{(i)}\left(1-y^{(i)}\right) - y^{(i)}\left(1-a^{(i)}\right)}{a^{(i)}\left(1-a^{(i)}\right)}$$

$$\frac{\partial C}{\partial a^{(i)}} = \frac{a^{(i)} - a^{(i)}y^{(i)} - y^{(i)} + a^{(i)}y^{(i)}}{a^{(i)}\left(1-a^{(i)}\right)}$$

Thus,

$$\frac{\partial C}{\partial a^{(i)}} = \frac{a^{(i)} - y^{(i)}}{a^{(i)}\left(1-a^{(i)}\right)}$$

Notice that the denominator looks quite familiar. Recall $a^{(i)} = \sigma\left(z^L\right)^{(i)}$, and we can relate the function to $\sigma'$ (let's drop the indices for simplicity):

$$\frac{\partial C}{\partial a} = \frac{a-y}{a(1-a)} = \frac{a-y}{\sigma(z)\left(1-\sigma(z)\right)} = \frac{a-y}{\sigma'(z)}$$

So if we ever had to find $\frac{\partial C}{\partial a}\sigma'(z)$, we'd magically get

$$\frac{\partial C}{\partial a}\sigma'(z) = \frac{a-y}{\sigma'(z)}\sigma'(z) = a-y$$

### 3.3 Rewriting the Four Fundamental Equations

Now that we know $\sigma'(z)$ and $\frac{\partial C}{\partial a}$ for our specific activation function and cost function, we can rewrite the backpropogation equations more specifically.

Here are their more generalized versions:

$$\begin{aligned}
&\delta_i^L = \frac{\partial C}{\partial a_i^L}\sigma'(z_i^L) && \delta^L = \nabla_a C \odot \sigma'(z^L) \\
&\delta_j^l = \sum_i w_{ij}^{l+1}\delta_i^{l+1}\sigma'(z_j^l) && \delta^l = \left(\left(w^{l+1}\right)^T \delta^{l+1}\right) \odot \sigma'(z^l) \\
&\frac{\partial C}{\partial w_{ij}^l} = \delta_i^l a_j^{l-1} && \frac{\partial C}{\partial w^l} = \delta^l \left(a^{l-1}\right)^T \\
&\frac{\partial C}{\partial b_i^l} = \delta_i^l && \frac{\partial C}{\partial b^l} = \delta^l
\end{aligned}$$

Here are their more specific versions:

$$\begin{aligned}
&\delta^L = a^L - y \\
&\delta^l = \left(\left(w^{l+1}\right)^T \delta^{l+1}\right) \odot \left(a^l \odot \left(1 - a^l\right)\right) \\
&\frac{\partial C}{\partial w^l} = \delta^l \left(a^{l-1}\right)^T \\
&\frac{\partial C}{\partial b^l} = \delta^l
\end{aligned}$$

## 4 Regularization

In order to reduce overfitting, we're going to add L2 regularization to our cost function.

Recall the cross entropy error function that we used as our cost function. We're now going to call this $C_0$:

$$C_0 = -\frac{1}{m} \sum_{i=1}^{m} \left[ y^{(i)} \ln a^{(i)} + (1 - y^{(i)}) \ln \left(1 - a^{(i)}\right) \right]$$

Our new cost function with L2 regularization:

$$C = -\frac{1}{m} \sum_{i=1}^{m} \left[ y^{(i)} \ln a^{(i)} + (1 - y^{(i)}) \ln \left(1 - a^{(i)}\right) \right] + \frac{\lambda}{2m} \sum_{i=1}^{m} \sum_{j} \sum_{k} w_{jk}^{(i)2}$$

Now that we've changed the cost function, we have to figure out how that changes $\frac{\partial C}{\partial w_{jk}^{(i)}}$ and $\frac{\partial C}{\partial b_j^{(i)}}$ Because we already know $\frac{\partial C_0}{\partial w_{jk}^{(i)}}$ and $\frac{\partial C_0}{\partial b_j^{(i)}}$ let's start by rewriting $C$ in terms of $C_0$:

$$C = C_0 + \frac{\lambda}{2m} \sum_{i}^{m} \sum_{j} \sum_{k} w_{jk}^{(i)2}$$

Thus,

$$\begin{aligned}
\frac{\partial C}{\partial w_{jk}^{(i)}} &= \frac{\partial C_0}{\partial w_{jk}^{(i)}} + \frac{d}{dw_{jk}^{(i)}} \frac{\lambda}{2m} w_{jk}^{(i)2} \\
\frac{\partial C}{\partial w_{jk}^{(i)}} &= \frac{\partial C_0}{\partial w_{jk}^{(i)}} + \frac{\lambda}{m} w_{jk}^{(i)}
\end{aligned}$$

Because, $\frac{\lambda}{2m}\sum_{i=1}^{m}\sum_{j}\sum_{k} {w_{jk}^{(i)}}^2$ has no dependence on $b_j^{(i)}$,

$$\frac{\partial C}{\partial b_j^{(i)}} = \frac{\partial C_0}{\partial b_j^{(i)}}$$