Machine Learning

Backpropagation Calculus | Deep Learning

Now we want to learn the underlying math behind the backpropagation. Consider a simple network with 1 input, 1 output and 2 hidden layers with 1 neuron each. This network is determined by 3 weights and 3 biases and we have to find how sensitive the cost function is to these weights and biases. Thus, we can know which weights or biases effectively decrease the cost function. As usual consider the last two neurons, activation of the last neuron as $a^{(L)}$ and that of the second to last neuron as $a^{(L-1)}$ and the desired output label as y. So the cost of this network for one single example is:

$$C_0(\dots) = (a^{(L)} - y)^2$$

Where

$$a^{(L)} = \sigma\left(z^{(L)}\right)$$

And

$$z^{(L)} = w^{(L)}a^{(L-1)} + b^{(L)}$$

It can be seen that $w^{(L)}$, $a^{(L-1)}$ and $b^{(L)}$ are used to calculate $z^{(L)}$, which in turn is used to calculate $a^{(L)}$. $a^{(L)}$ and y are used finally to evaluate C_0. It is also clear that $w^{(L-1)}$, $a^{(L-2)}$ and $b^{(L-1)}$ are used to calculate $z^{(L-1)}$, which in turn is used to calculate $a^{(L-1)}$ and so on. We are interested in finding how sensitive the cost function is to the weight i.e., what is the value of $\frac{\partial C_0}{\partial w^{(L)}}$ which can be given by:

$$\frac{\partial C_0}{\partial w^{(L)}} = \frac{\partial z^{(L)}}{\partial w^{(L)}}\frac{\partial a^{(L)}}{\partial z^{(L)}}\frac{\partial C_0}{\partial a^{(L)}} = a^{(L-1)}\sigma'\left(z^{(L)}\right)2(a^{(L)} - y)$$

Where

$$\frac{\partial C_0}{\partial a^{(L)}} = 2(a^{(L)} - y)$$

And

$$\frac{\partial a^{(L)}}{\partial z^{(L)}} = \sigma'\left(z^{(L)}\right)$$

And

$$\frac{\partial z^{(L)}}{\partial w^{(L)}} = a^{(L-1)}$$

The above differential means that the amount of change in $w^{(L)}$ i.e., $\partial w^{(L)}$ that influences the last layer depends on the activation of the previous layer neuron $a^{(L-1)}$. We also want to find how sensitive the cost function is to the bias i.e., what is the value of $\frac{\partial C_0}{\partial b^{(L)}}$ which can be given by:

$$\frac{\partial C_0}{\partial b^{(L)}} = \frac{\partial z^{(L)}}{\partial b^{(L)}}\frac{\partial a^{(L)}}{\partial z^{(L)}}\frac{\partial C_0}{\partial a^{(L)}} = 1\sigma'\left(z^{(L)}\right)2(a^{(L)} - y)$$

Where

$$\frac{\partial C_0}{\partial a^{(L)}} = 2(a^{(L)} - y)$$

And

$$\frac{\partial a^{(L)}}{\partial z^{(L)}} = \sigma'\left(z^{(L)}\right)$$

But

$$\frac{\partial z^{(L)}}{\partial b^{(L)}} = 1$$

The above differential means that the amount of change in $b^{(L)}$ i.e., $\partial b^{(L)}$ that influences the last layer does not depend on the activation of the previous layer neuron. Similarly we find how sensitive the cost function is to the activation of the previous layer i.e., what is the value of $\frac{\partial C_0}{\partial a^{(L-1)}}$ which can be given by:

$$\frac{\partial C_0}{\partial a^{(L-1)}} = \frac{\partial z^{(L)}}{\partial a^{(L-1)}}\frac{\partial a^{(L)}}{\partial z^{(L)}}\frac{\partial C_0}{\partial a^{(L)}} = w^{(L)}\sigma'\left(z^{(L)}\right)2(a^{(L)} - y)$$

Where

$$\frac{\partial C_0}{\partial a^{(L)}} = 2(a^{(L)} - y)$$

And

$$\frac{\partial a^{(L)}}{\partial z^{(L)}} = \sigma'\left(z^{(L)}\right)$$

And

$$\frac{\partial z^{(L)}}{\partial a^{(L-1)}} = w^{(L)}$$

The above differential means that the amount of change in $a^{(L-1)}$ i.e., $\partial a^{(L-1)}$ that influences the last layer depends on the weight $w^{(L)}$ of the last layer neuron $a^{(L)}$ showing the backpropagation process.

$\frac{\partial C_0}{\partial w^{(L)}}$ is the derivative of the cost function of only one training example. The full cost function is the average of all individual cost of different training examples then the derivative of the full cost function is:

$$\frac{\partial C}{\partial w^{(L)}} = \frac{1}{n}\sum_{k=0}^{n-1}\frac{\partial C_k}{\partial w^{(L)}}$$

Which is just one element (component) in the gradient vector $\mathbf{\nabla C}$, given by:

$$\mathbf{\nabla C} = \begin{bmatrix} \dfrac{\partial C}{\partial w^{(1)}} \\ \dfrac{\partial C}{\partial b^{(1)}} \\ \vdots \\ \dfrac{\partial C}{\partial w^{(L)}} \\ \dfrac{\partial C}{\partial b^{(L)}} \end{bmatrix}$$

By finding how sensitive the cost function is to the weights, activations of previous layers and biases we can iterate on the chain rule to see how sensitive the cost function is to particular weights, prior activations and biases in that particular layer.

Now considering a network with more than one neuron then the above procedure remains same with small changes such as the activation of a particular last layer neuron is indicated by ${a_j}^{(L)}$ and a particular neuron of interest in previous layer can be indicated by ${a_k}^{(L-1)}$. Similarly weights between these two layers are indicated by ${w_{jk}}^{(L)}$. If the previous layer has 3 neurons and last layer has 2 neurons with same number (2) of output labels we can write:

$$C_0 = \sum_{j=0}^{n_L-1} ({a_j}^{(L)} - y)^2$$

Where

$${a_j}^{(L)} = \sigma\left({z_j}^{(L)}\right)$$

And

$${z_j}^{(L)} = \sum_{k=0}^{3-1} {w_{jk}}^{(L)} {a_k}^{(L-1)} + {b_j}^{(L)}$$

Similarly,

$$\frac{\partial C_0}{\partial {w_{jk}}^{(L)}} = \frac{\partial {z_j}^{(L)}}{\partial {w_{jk}}^{(L)}} \frac{\partial {a_j}^{(L)}}{\partial {z_j}^{(L)}} \frac{\partial C_0}{\partial {a_j}^{(L)}}$$

And

$$\frac{\partial C_0}{\partial {b_j}^{(L)}} = \frac{\partial {z_j}^{(L)}}{\partial {b_j}^{(L)}} \frac{\partial {a_j}^{(L)}}{\partial {z_j}^{(L)}} \frac{\partial C_0}{\partial {a_j}^{(L)}}$$

But, the derivative of the cost with respect to the activations in the previous layer $({a_k}^{(L-1)})$ is different in the way that the neuron in previous layer influences the cost through multiple paths i.e., in this case the activation ${a_k}^{(L-1)}$ influences ${a_0}^{(L)}$ and ${a_1}^{(L)}$ both of which play a role in the cost function. So, we have to add these influences which can be written as:

$$\frac{\partial C_0}{\partial {a_k}^{(L-1)}} = \sum_{j=0}^{n_L-1} \frac{\partial {z_j}^{(L)}}{\partial {a_k}^{(L-1)}} \frac{\partial {a_j}^{(L)}}{\partial {z_j}^{(L)}} \frac{\partial C_0}{\partial {a_j}^{(L)}}$$

Once we know how sensitive the cost function is to the activations in the previous layer (second to last layer), we can repeat this process for all the weights and biases feeding into this layer.

Therefore, the chain rule expressions mentioned below gives the derivatives that determine each component (element) in the gradient $\mathbf{\nabla C}$ that helps in minimizing the cost of the network by taking the repeated steps towards the local minimum:

$$\frac{\partial C}{\partial {w_{jk}}^{(l)}} = {a_k}^{(l-1)} \sigma'\left({z_j}^{(l)}\right) \frac{\partial C}{\partial {a_j}^{(l)}}$$

And

$$\frac{\partial C}{\partial {a_j}^{(l)}} = \sum_{j=0}^{n_{l+1}-1} {w_{jk}}^{(l+1)} \sigma'\left({z_j}^{(l+1)}\right) \frac{\partial C}{\partial {a_j}^{(l+1)}}$$

or

$$\frac{\partial C}{\partial {a_j}^{(l)}} = 2({a_j}^{(L)} - y_j)$$