Backpropagation

Calculcating the gradient for weight update

April 15, 2018

Let out_k be the ouput of the neuron k, in_k the input of a neuron k, f the neuron's activation function and y_i the true label of the input x_i.

Calculcating the derivative with respect to a weight w_{jk} between the neurons j and k on layer l in a fully connected neural network:

According to the chain rule:

$$d_{jk} = \frac{\delta E}{\delta w_{jk}} = \frac{\delta E}{\delta out_k} \cdot \frac{\delta out_k}{\delta in_k} \cdot \frac{\delta in_k}{\delta w_{jk}}$$

The input of a neuron consists of the sum of all the outputs of the neuron receives multiplied by the weight of each connection.

In other words, for neuron k on layer l:

$$in_k = \sum_{j \in l-1} w_{jk} \cdot out_j$$

and with f_i being the activation function of neuron i:

$$out_i = f_i(in_i)$$

Let's calculate the terms we need to define the gradient. We'll start with the simplest ones:

$$\frac{\delta in_k}{\delta w_{jk}} = \frac{\delta(\sum_{i \in l-1}^{|l|} w_{ik} \cdot out_i)}{\delta w_{jk}}, \quad |l| \geq k$$

When we derive w.r.t w_{jk}, all other weights are treated as real numbers, and their derivative is zero.

$$\begin{aligned}\frac{\delta in_k}{\delta w_{jk}} &= \frac{\delta w_{jk} \cdot out_j}{\delta w_{jk}} \\ &= out_j\end{aligned}$$

This is why we want to save all the outputs of each neuron.

The next one is also quite nice and simple. It's also easy to impelement: you can derive the activation function by hand, and then build a function which gives the value of the derivative in a specific point. Hence we have to save all the inputs of each neuron in forward pass.

$$\frac{\delta out_k}{\delta in_k} = \frac{\delta f(in_k)}{\delta in_k} = f'(in_k)$$

This is where it get's trickier. Backpropagation is actually recursive: It calculates the gradient of a specific weight based on the output that weight affects and the error of the neuron taking that output as an input. The base case is the derivative of the loss function w.r.t the output of the last layer. Intuitively this is done, because all the errors in the network affect the errors in future layers. This is what puts the "back" to "backpropagation".

As a result. the definition is different for the output layer, and all other layers, since the output layer doesn't have a layer coming after it.

If l is in output layer and out is the output of whole the network.

$$\frac{\delta E}{\delta out_l} = \frac{\delta E}{\delta out}$$

This is a relatively simple calculation, but it depends on the choice of the error function.

For the batch-wise mean squared error, with y_i the true label of an input, and out_i the output of the network on batch i, and out_j a specific output on a batch:

$$\begin{aligned}\frac{\delta\left(\frac{1}{b|out|}\sum_{i=0}^{b}(y_i - out_i)^2\right)}{\delta out_j} &= \frac{\delta \frac{1}{b|out|}(y_j - out_j)^2}{\delta out_j}, \quad b \geq j \\ &= \frac{2}{b|out|}(out_j - y_j)\end{aligned}$$

If l is not in the output layer, we calculate the following using the error of the next layer $l+1$:

$$\begin{aligned}\frac{\delta E}{\delta out_k} &= \sum_{j \in l+1} \left(\frac{\delta E}{\delta in_j} \cdot \frac{\delta in_j}{\delta out_k} \right) \\ &= \sum_{j \in l+1} \left(\frac{\delta E}{\delta out_j} \cdot \frac{\delta out_j}{\delta in_j} \cdot w_{kj} \right) \\ &= \sum_{j \in l+1} \left(\frac{\delta E}{\delta out_j} \cdot w_{kj} \right) \cdot f'(in_j)\end{aligned}$$

Notice how $\frac{\delta E}{\delta out_k}$ depends on the $\frac{\delta E}{\delta out_k}$ of the next layer.
Now we can define the formula of the error δ recursively:

$$\delta_k = \sum_{i \in l+1} (\delta_i \cdot w_{ki}) \cdot f'(in_k)$$

$$\text{Where } \delta_i = \frac{\delta E}{\delta out_i}$$

In conclusion, for output layers:

$$\begin{aligned}\delta_{jl} &= \frac{\delta E}{\delta w_{jl}} \\ &= \frac{\delta E}{\delta out} \cdot f'(in_j)\end{aligned}$$

and for other layers:

$$\begin{aligned}\delta_{jl} &= \frac{\delta E}{\delta w_{jl}} \\ &= \sum_{i \in l+1} (\delta_i \cdot w_{li}) \cdot f'(in_l)\end{aligned}$$

Which is then used in the weight update like this:

$$d_{ij} = \delta_j \cdot out_i$$