Machine Learning
Exercise 13: Manual calculation of the forward and backward pass for a simple Neural Network

Prof. Dr. Thomas Kopinski

August 22, 2022

Task 2: Calculate the results of the forward pass

This calculation is pretty straight forward as it comprises of simple addition and multiplication and does not contain any derivatives. To get the value for the first hidden neuron h_1 you have to sum up all its incoming values and then take the Sigmoid ($S(x) = \frac{1}{1+e^x}$) of that result:

$$\begin{aligned} \mathrm{h}_1 &= S(in_1 * w_1 + in_2 * w_3 + b_1) \\ &= S(0.2 * 0.3 + 0.6 * 0.7 + 0.4) \\ &= 0.7068 \end{aligned}$$

$$\begin{aligned} \mathrm{h}_2 &= S((n_1 * w_2 + in_2 * w_4 + b_1) \\ &= S(0.2 * 0.1 + 0.6 * 0.5 + 0.4) \\ &= 0.6726 \end{aligned}$$

You then perform the same operation for the calculation of the neural network's output, this time taking the values of the hidden neurons as input:

$$\begin{aligned} out_1 &= S(h_1 * w_5 + h_2 * w_6 + b_2) \\ &= S(0.7068 * 0.6 + 0.6726 * 0.7 + 0.15) \\ &= 0.7397 \end{aligned}$$

$$\begin{aligned} out_2 &= S(h_1 * w_7 + h_2 * w_8 + b_2) \\ &= S(0.7068 * 0.3 + 0.6726 * 0.8 + 0.15) \\ &= 0.7109 \end{aligned}$$

Task 3: Calculate the results of the backward pass and update the network weights

The backward pass is a little more tricky as you have to apply the chain rule to solve the exercise. It can be split into two parts, first from output layer back to the hidden layer and then from there to the input. In order to get a loss value to optimize for, we calculate the difference between the output values you got in task 2 and the expected, correct outputs (target values) that are used for the training process. We thus define our error as follows:

$$\mathrm{E}_{\mathrm{total}} = \frac{1}{2} \sum_{\mathrm{j} \epsilon \mathrm{output}} (\mathrm{t_j} - \mathrm{y_j})^2 =$$

$$\frac{1}{2}((\mathrm{target}_1 - \mathrm{out}_1)^2 + (\mathrm{target}_2 - \mathrm{out}_2)^2)$$

The goal of the backpropagation algorithm is to iteratively update the values of the network's weights in order to find a solution which minimizes the total error. We will now first take a closer look at how this is achieved for w_5 by using the Chain Rule, where sum_{out_j} represents the neurons' value before applying the Sigmoid function:

$$\frac{\partial E_{total}}{\partial w_5} = \frac{\partial E_{total}}{\partial out_1} \frac{\partial out_1}{\partial sum_{out_1}} \frac{\partial sum_{out_1}}{\partial w_5}$$

This equation tries to determine the impact of a single weight on the network error by splitting this task up into multiple partial derivatives. The first step is to calculate the change caused in E by the output neuron and then multiply this by the change of the output value due to its summed up input (sum_{out_1}). Furthermore this term gets multiplied once again by the derivative of the latter w.r.t. the weight in question. We are now going to solve this step by step for a single weight (w_5) by calculating the results of the individual partial derivatives, using the same order as defined in the equation above:

$$\begin{aligned} \frac{\partial E_{total}}{\partial out_1} &= 2 * \frac{1}{2} * (target_1 - out_1) * -1 \\ &= out_1 - target_1 = 0.5897 \end{aligned}$$

$$\frac{\partial E_{total}}{\partial out_2} = -0.1390$$

As written in the problem formulation, the derivative of the Sigmoid function $S(x)$ is $S'(x) = S(x)(1 - S(x))$. This leads to the following equations and results:

$$\begin{aligned} \frac{\partial out_1}{\partial sum_{out_1}} &= S(sum_{out_1})(1 - S(sum_{out_1})) \\ &= out_1(1 - out_1) \\ &= 0.1924 \end{aligned}$$

$$\frac{\partial out_2}{\partial sum_{out_2}} = 0.2054$$

The last components needed for this first step are the derivatives of the summed up input to each output neuron w.r.t. the individual weight:

$$\begin{aligned} sum_{out_1} &= h_1 * w_5 + h_2 * w_6 + b_2 \\ \frac{\partial sum_{out_1}}{\partial w_5} &= h_1 \\ \frac{\partial sum_{out_1}}{\partial w_6} &= h_2 \end{aligned}$$

$$\begin{aligned} sum_{out_2} &= h_1 * w_7 + h_2 * w_8 + b_2 \\ \frac{\partial sum_{out_2}}{\partial w_7} &= h_1 \\ \frac{\partial sum_{out_2}}{\partial w_8} &= h_2 \end{aligned}$$

Now that we have calculated all the necessary terms we can put it all together. This is displayed in more detail for weight 5, but works accordingly for all the other weights in the output layer. The result of the partial derivatives w.r.t. each weight are then multiplied by the learning rate (lr) of the

network and subtracted from the current value of the respective weight:

$$\begin{aligned}\frac{\partial E_{total}}{\partial w_5} &= \frac{\partial E_{total}}{\partial out_1}\frac{\partial out_1}{\partial sum_{out_1}}\frac{\partial sum_{out_1}}{\partial w_5} \\ &= (out_1 - target_1) * (out_1(1 - out_1)) * h_1 \\ &= 0.5897 * 0.1924 * 0.7068 \\ &= 0.0801\end{aligned}$$

$$\begin{aligned}w_5' &= w_5 - lr * \frac{\partial E_{total}}{\partial w_5} \\ &= 0.6 - 0.1 * 0.0801 = 0.5919\end{aligned}$$

$$\begin{aligned}w_6' &= 0.7 - 0.1 * 0.0763 = 0.6924 \\ w_7' &= 0.3 - 0.1 * (-0.0201) = 0.3020 \\ w_8' &= 0.8 - 0.1 * (-0.0192) = 0.8019\end{aligned}$$

These values will be applied to the neural network once the complete backward pass is finished, including the stage from hidden layer back to the input.
The second part of the backpropagation algorithm works similar to the first one but reuses some of the results we calculated above, making it more efficient to apply. One important difference is the influence of single weights on the output of the network. Where in the output layer a single weight always just changed one particular output neurons' value, it now has an impact on both of those. This can be seen in the following equations where the partial derivatives for both error terms are calculated separately and then added together:

$$\frac{\partial E_1}{\partial w_1} = \frac{\partial E_1}{\partial out_1}\frac{\partial out_1}{\partial sum_{out_1}}\frac{\partial sum_{out_1}}{\partial h_1}\frac{\partial h_1}{\partial sum_{h_1}}\frac{\partial sum_{h_1}}{\partial w_1}$$

$$\frac{\partial E_2}{\partial w_1} = \frac{\partial E_2}{\partial out_2}\frac{\partial out_2}{\partial sum_{out_2}}\frac{\partial sum_{out_2}}{\partial h_1}\frac{\partial h_1}{\partial sum_{h_1}}\frac{\partial sum_{h_1}}{\partial w_1}$$

We have already computed the first two terms of these equations, one of the advantages of this approach. Here you can see the results for all the single components of the equations above:

$$\frac{\partial E_1}{\partial out_1} = out_1 - target_1 = 0.5897$$
$$\frac{\partial out_1}{\partial sum_{out_1}} = out_1 * (1 - out_1) = 0.1924$$

$$sum_{out_1} = h_1 * w_5 + h_2 * w_6 + b_2$$
$$\frac{\partial sum_{out_1}}{\partial h_1} = w_5$$
$$= 0.6$$

$$\frac{\partial h_1}{\partial sum_{h_1}} = h_1 * (1 - h_1)$$
$$= 0.7068 * (1 - 0.7068)$$
$$= 0.2072$$

$$sum_{h_1} = in_1 * w_1 + in_2 * w_3 + b_1$$
$$\frac{\partial sum_{h_1}}{\partial w_1} = in_1$$
$$= 0.2$$

$$\frac{\partial E_1}{\partial w_1} = 0.5897 * 0.1924 * 0.6 * 0.2072 * 0.2$$
$$= 0.0028$$

$$\frac{\partial E_2}{\partial out_2} = out_2 - target_2 = -0.1390$$

$$\frac{\partial out_2}{\partial sum_{out_2}} = out_2 * (1 - out_2) = 0.2054$$

$$sum_{out_2} = h_1 * w_7 + h_2 * w_8 + b_2$$
$$\frac{\partial sum_{out_2}}{\partial h_1} = w_7$$
$$= 0.3$$

$$\frac{\partial h_1}{\partial sum_{h_1}} = 0.2072$$

$$\frac{\partial sum_{h_1}}{\partial w_1} = in_1$$
$$= 0.2$$

$$\frac{\partial E_2}{\partial w_1} = -0.1390 * 0.2054 * 0.3 * 0.2072 * 0.2$$
$$= -0.00035$$

Putting this all together:

$$\begin{aligned}\frac{\partial E_{total}}{w_1} &= \frac{\partial E_1}{w_1} + \frac{\partial E_2}{w_1} \\ &= 0.0025\end{aligned}$$

$$\begin{aligned}w_1' &= w_1 - lr * \frac{\partial E_{total}}{\partial w_1} \\ &= 0.3 - 0.1 * 0.0025 \\ &= 0.2997\end{aligned}$$

$$\begin{aligned}w_2' &= 0.0997 \\ w_3' &= 0.6992 \\ w_4' &= 0.4993\end{aligned}$$