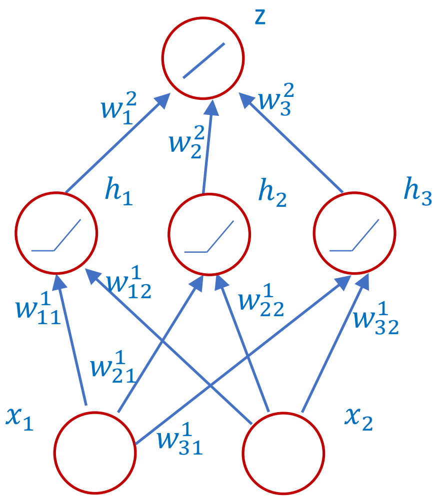
Backpropagation Numerical Exercises

Nicolò Navarin

Let's consider an instantiation of a neural network , and let's compute the gradients numerically on a single example.
For simplicity, let's consider all the biases equal to zero (we omit them from the numerical computations).
The network is defined according to the following diagram. Consider t=2 and MSE loss function
$J = \frac{1}{2}(t - z)^2$



Or, more formally,

$$\boldsymbol{W}^1 = \begin{bmatrix} w_{11}^1 & w_{12}^1 \\ w_{21}^1 & w_{22}^1 \\ w_{31}^1 & w_{32}^1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ -1 & 1 \\ 0 & 0 \end{bmatrix}, \boldsymbol{w}^2 = \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix}$$

$$\boldsymbol{a}^1 = \begin{bmatrix} a_1^1 \\ a_2^1 \\ a_3^1 \end{bmatrix} = \boldsymbol{W}^1\boldsymbol{x} + \boldsymbol{b}^1 = \begin{bmatrix} w_{11}^1 & w_{12}^1 \\ w_{21}^1 & w_{22}^1 \\ w_{31}^1 & w_{32}^1 \end{bmatrix}\boldsymbol{x} + \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ -1 & 1 \\ 0 & 0 \end{bmatrix}\boldsymbol{x} + \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

$$\boldsymbol{h} = \begin{bmatrix} h_1 \\ h_2 \\ h_3 \end{bmatrix} = relu(\boldsymbol{a}^1) = \begin{bmatrix} \max(0, a_1^1) \\ \max(0, a_2^1) \\ \max(0, a_3^1) \end{bmatrix}$$

$$a^2 = (\boldsymbol{w}^2)^T\boldsymbol{h} + b^2 = \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix}\boldsymbol{h} + 0 \text{ , } z = i(a^2) = a^2$$

Where i is the identity function. Let us recall that the derivative of the linear function is always 1, i.e. $i'(x) = 1$ and the derivative of the ReLU is defined as:

$$relu'(x) = \begin{cases} 1 \; if \; x > 0 \\ 0 \; otherwise \end{cases}.$$

Let's start computing the forward pass with $\boldsymbol{x} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$ and $t = 2$:

$a_1^1 = 1 \cdot 1 + 0 \cdot 0 = 1, a_2^1 = 1 \cdot (-1) + 0 \cdot 1 = 0, a_3^1 = \ 1 \cdot 0 + 0 \cdot 0 = 0$

$h_1 = \max(0, a_1^1) = 1\,, h_2 = \max(0, a_2^1) = 0\,, h_3 = \max(0, a_3^1) = 0$
$z = i(a^2) = a^2 = 1 \cdot 1 + 2 \cdot 0 + 1 \cdot 0 = 1,\ J = (t - z)^2 = (2-1)^2 = 1$

We can now compute the gradient with respect to the weights of the second layer

$$\frac{\partial J}{\partial w_1^2} = -(t \quad - z \quad)i'(a^2)h_1 = -(2-1) \cdot 1 \cdot 1 = -1$$

$$\frac{\partial J}{\partial w_2^2} = -(t \quad - z \quad)i'(a^2)h_2 = -(2-1) \cdot 1 \cdot 0 = 0$$

$$\frac{\partial J}{\partial w_3^2} = -(t \quad - z \quad)i'(a^2)h_3 = -(2-1) \cdot 1 \cdot 0 = 0$$

Vector notation:

$$\frac{\partial J}{\partial \boldsymbol{w}^2} = \frac{\partial J}{\partial z}\frac{\partial z}{\partial a^2}\frac{\partial a^2}{\partial \boldsymbol{w}^2} = [-(t-z)][i'(a^2)][\boldsymbol{h}^T] = [-(2-1)][1][1,0,0] = [-1,0,0]$$

EXERCISE: compute the gradient w.r.t. the bias of the second layer, and the weights and biases of the first layer.

…compute them also in matrix notation!

Hint: when computing the gradients, you will have a term that is the derivative of the relu, in particular $\frac{\partial \boldsymbol{h}}{\partial \boldsymbol{a}^1}$. As already seen in class this term is a diagonal matrix, containing on on its diagonal the derivative of the activation function w.r.t. the pre-activation.
Let's take as an example the pre-activation $\boldsymbol{a}^1 = [1,0,0]$.
Then we have:

$$\frac{\partial \boldsymbol{h}}{\partial \boldsymbol{a}^1} = \begin{bmatrix} relu'(a_1^1) & 0 & 0 \\ 0 & relu'(a_2^1) & 0 \\ 0 & 0 & relu'(a_3^1) \end{bmatrix} = \begin{bmatrix} relu'(1) & 0 & 0 \\ 0 & relu'(0) & 0 \\ 0 & 0 & relu'(0) \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}.$$