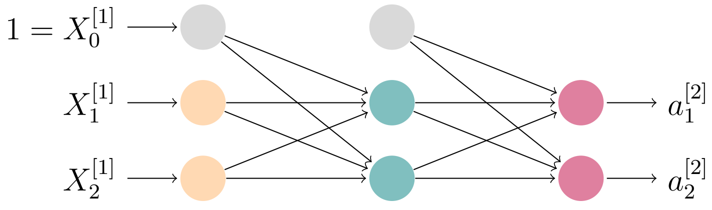
We consider a neural network with two layers, with two neurons each. Below, gray circles denote artificially added inputs, which are always 1 and encode the bias.



Let us suppose that for the input $\begin{bmatrix}1\\0\end{bmatrix}$ we expect the network to output $\begin{bmatrix}1\\0\end{bmatrix}$.

Initial weights:

$$W^{[1]} = \begin{bmatrix}0.1 & -0.2 & 0.3\\ -0.4 & 0.5 & -0.6\end{bmatrix}, \qquad W^{[2]} = \begin{bmatrix}0.15 & -0.25 & 0.35\\ -0.45 & 0.55 & -0.65\end{bmatrix}$$

Input (the first coordinate is always 1 and encodes the bias):

$$X^{[1]} = \begin{bmatrix}1\\1\\0\end{bmatrix}.$$

Forward pass. The first layer ($W^{[1]}$) gives

$$net^{[1]} = W^{[1]} \cdot X^{[1]} = \begin{bmatrix}-0.1\\0.1\end{bmatrix},$$

after applying the function $\varphi(x) = (1 + e^{-x})^{-1}$ element-wise we obtain the output of the first layer,

$$a^{[1]} = \begin{bmatrix}\varphi(-0.1)\\ \varphi(0.1)\end{bmatrix} = \begin{bmatrix}0.47502081\\0.52497919\end{bmatrix},$$

and after prepending 1, we obtain the input of the second layer:

$$X^{[2]} = \begin{bmatrix}1\\0.47502081\\0.52497919\end{bmatrix}$$

The second layer ($W^{[2]}$) gives

$$net^{[2]} = W^{[2]} \cdot X^{[2]} = \begin{bmatrix}0.21498751\\-0.52997502\end{bmatrix}$$

after applying the function $\varphi(x) = (1 + e^{-x})^{-1}$ element-wise we obtain the output of the whole network,

$$a^{[2]} = \begin{bmatrix}\varphi(0.21498751)\\ \varphi(-0.52997502)\end{bmatrix} = \begin{bmatrix}0.55354082\\0.37052271\end{bmatrix}.$$

Back propagation. We expected the network to output $y = \begin{bmatrix}1\\0\end{bmatrix}$. For the loss function $L(y, a^{[2]}) = \frac{1}{2}\|y - a^{[2]}\|_2^2$ we have

$$\frac{\partial L}{\partial a^{[2]}} = a^{[2]} - y = a^{[2]} - \begin{bmatrix}1\\0\end{bmatrix} = \begin{bmatrix}-0.44645918\\0.37052271\end{bmatrix}. \tag{0.1}$$

From there we obtain the *delta signal* by multiplying elements of by the value of φ' at the corresponding points of $net^{[2]}$ (here φ' is computed using an explicit formula of φ),

$$\delta^{[2]} = \begin{bmatrix}-0.44645918 \cdot \varphi'(0.21498751)\\ 0.37052271 \cdot \varphi'(-0.52997502)\end{bmatrix} = \begin{bmatrix}-0.110334967\\0.086419099\end{bmatrix}.$$

Thus

$$\frac{\partial L}{W^{[2]}} = \delta^{[2]} \cdot (X^{[2]})^T = \begin{bmatrix} -0.110335 & -0.0524114 & -0.0579236 \\ 0.0864191 & 0.0410509 & 0.0453682 \end{bmatrix}.$$

We adjust the weights $W^{[2]}$, taking the *learning rate* equal to $c = 0.1$,

$$\widetilde{W^{[2]}} = W^{[2]} - c\,\frac{\partial L}{W^{[2]}} = W^{[2]} - c \begin{bmatrix} -0.110335 & -0.0524114 & -0.0579236 \\ 0.0864191 & 0.0410509 & 0.0453682 \end{bmatrix}$$

$$= \begin{bmatrix} 0.1610335 & -0.24475886 & 0.35579236 \\ -0.45864191 & 0.54589491 & -0.65453682 \end{bmatrix}$$

We compute

$$\frac{\partial L}{X^{[2]}} = (W^{[2]})^T \cdot \delta^{[2]} = \begin{bmatrix} -0.05543884 \\ 0.07511425 \\ -0.09478965 \end{bmatrix}.$$

The first coordinate (-0.05543884) is redundant (it corresponds to the constant input 1, which encodes the bias) – we omit it and obtain

$$\frac{\partial L}{a^{[1]}} = \begin{bmatrix} 0.07511425 \\ -0.09478965 \end{bmatrix}. \tag{0.2}$$

Let us note that we have obtained an analogous derivative as in (0.1), but for the deeper layer. We continue analogously. Specifically, we multiply elements of $\frac{\partial L}{a^{[1]}}$ by the values of φ' at the corresponding points $net^{[1]}$, from where we obtain the delta signal for the first layer,

$$\delta^{[1]} = \begin{bmatrix} 0.07511425 \cdot \varphi'(-0.1) \\ -0.09478965 \cdot \varphi'(0.1) \end{bmatrix} = \begin{bmatrix} 0.0187316942 \\ -0.023638267 \end{bmatrix}.$$

Hence

$$\frac{\partial L}{W^{[1]}} = \delta^{[1]} \cdot (X^{[1]})^T = \begin{bmatrix} 0.0187317 & 0.0187317 & 0 \\ -0.0236383 & -0.0236383 & 0 \end{bmatrix}.$$

We adjust the weights $W^{[1]}$, taking the learning rare again equal to $c = 0.1$,

$$\widetilde{W^{[1]}} = W^{[1]} - c\frac{\partial L}{W^{[1]}} = \begin{bmatrix} 0.09812683 & -0.20187317 & 0.3 \\ -0.39763617 & 0.50236383 & -0.6 \end{bmatrix}.$$

We obtain a network with modified weights $\widetilde{W^{[1]}}$ i $\widetilde{W^{[2]}}$, and repeat...

Remarks:

(1) If we had a deeper network, we would continue computing

$$\frac{\partial L}{X^{[1]}} = (W^{[1]})^T \cdot \delta^{[1]},$$

then we would omit the first coordinate, to obtain $\frac{\partial L}{a^{[0]}}$, and we would be in a situation analogous to (0.1) and (0.2).

(2) Above equalities are not exact, some rounding errors are possible

(3) For the function $\varphi(x) = (1 + e^{-x})^{-1}$ it holds (as is easy to check), $\varphi'(x) = \varphi(x)(1-\varphi(x))$. This allows us to perform the calculations more efficiently, because $\varphi(x)$ is computed in the forward pass.

(4) If we use another activation function φ (but not *softmax*) and the same loss function as above, then in the above calculations nothing will essentially change, apart from the values of $\varphi(\ldots)$ and $\varphi'(\ldots)$.

(5) In the last layer *softmax* function ψ is often used as the activation function,

$$\psi(\begin{bmatrix} x_1 \\ x_2 \\ \dots \\ x_n \end{bmatrix}) = \begin{bmatrix} \frac{e^{x_1}}{\sum_{k=1}^{n} e^{x_k}} \\ \frac{e^{x_2}}{\sum_{k=1}^{n} e^{x_k}} \\ \dots \\ \frac{e^{x_n}}{\sum_{k=1}^{n} e^{x_k}} \end{bmatrix}.$$

Because it depends on the whole vector $net^{[L-1]}$, i.e., it is not a function on $\mathbf{R}$, which is applied element-wise to the vector *net*, therefore the way the output of the network and the back-propagation in the last layer are performed are somewhat different. In the example above, we would have

$$a^{[2]} = \psi(\begin{bmatrix} 0.21498751 \\ -0.52997502 \end{bmatrix}) = \begin{bmatrix} \frac{1.2398464}{1.828466} \\ \frac{0.58861967}{1.828466} \end{bmatrix} = \begin{bmatrix} 0.67808 \\ 0.32192 \end{bmatrix}$$

If we additionally use *categorical cross-entropy* as the loss function (which one usually does with the *softmax*) – in the example above it would be the function

$$L_e(y, a^{[2]}) = \sum_{k=1}^{2} -y_k \log(a_k^{[2]}),$$

then the above recipe for adjusting the weights will stay valid, if we redefine $\delta^{[2]}$ in the following way (note: we only redefine the last δ, not δ's for the deeper layers):

$$\delta^{[2]} := \left(\sum_{j=1}^{2} y_j \right) a^{[2]} - y.$$

In comparison to the previous formula, we no longer multiply by the derivatives φ'. To verify this fact one needs to repeat the calculations of $\frac{\partial L}{\partial W^{[2]}}$ and $\frac{\partial L}{\partial X^{[2]}}$; these calculations are quite tedious, because now each $a_j^{[2]}$ depends on *all* elements of the matrix $W^{[2]}$.

In our example we would have

$$\delta^{[2]} = (1+0)a^{[2]} - y = \begin{bmatrix} -0.32192 \\ 0.32192 \end{bmatrix}.$$

We continue as before, but the value $\delta^{[2]}$ is different, we will become different numbers. More specifically, we obtain

$$\frac{\partial L}{W^{[2]}} = \delta^{[2]} \cdot (X^{[2]})^T = \begin{bmatrix} -0.32192 & -0.1529187 & -0.1690013 \\ 0.32192 & 0.1529187 & 0.1690013 \end{bmatrix}.$$

We adjust the weights $W^{[2]}$ using the learning rate equal to $c = 0.1$,

$$\widetilde{W^{[2]}} = W^{[2]} - c\frac{\partial L}{W^{[2]}} = W^{[2]} - c\begin{bmatrix} -0.32192 & -0.1529187 & -0.1690013 \\ 0.32192 & 0.1529187 & 0.1690013 \end{bmatrix}$$

$$= \begin{bmatrix} 0.182192 & -0.23470813 & 0.36690013 \\ -0.482192 & 0.53470813 & -0.66690013 \end{bmatrix}$$

We compute

$$\frac{\partial L}{X^{[2]}} = (W^{[2]})^T \cdot \delta^{[2]} = \begin{bmatrix} -0.193152 \\ 0.257536 \\ -0.32192 \end{bmatrix}.$$

The first coordinate (-0.193152) is redundant (it corresponds to the constant input 1, which encodes the bias) – we omit it and obtain

$$\frac{\partial L}{a^{[1]}} = \begin{bmatrix} 0.257536 \\ -0.32192 \end{bmatrix}.$$

Thus

$$\delta^{[1]} = \begin{bmatrix} 0.257536 \cdot \varphi'(-0.1) \\ -0.32192 \cdot \varphi'(0.1) \end{bmatrix} = \begin{bmatrix} 0.06422331 \\ -0.08027913 \end{bmatrix}.$$

Hence

$$\frac{\partial L}{W^{[1]}} = \delta^{[1]} \cdot (X^{[1]})^T = \begin{bmatrix} 0.06422331 & 0.06422331 & 0 \\ -0.08027913 & -0.08027913 & 0 \end{bmatrix}.$$

We adjust the weights $W^{[1]}$ taking again the learning rate equal to $c = 0.1$,

$$\widetilde{W^{[1]}} = W^{[1]} - c\frac{\partial L}{W^{[1]}} = \begin{bmatrix} 0.09357767 & -0.20642233 & 0.3 \\ -0.39197209 & 0.50802791 & -0.6 \end{bmatrix}.$$

prepared by Bartek Dyda