# General feed forward neural networks

Kristian Wichmann

July 17, 2017

## 1 Formalism and nomenclature

Consider a feed forward neural net with $L+1$ layers, in the sense that layer zero is the input layer and layer $L$ the output layer. Each non-input layer has its own activation function $\sigma_l$.

The size of layer $l$ we will denote $S_l$. So layer $l$ has a $S_l \times S_{l-1}$ matrix of weights $W^{(l)}$ and a bias vector $b^{(l)}$ with dimension $S_l$. So, given an input vector $x$ (with dimension $S_{l-1}$), the pre-activation and the activation of layer $l$ can be expressed as:

$$z^{(l)} = W^{(l)}x + b^{(l)}, \quad a^{(l)} = \sigma_l(z^{(l)}) = \sigma_l(W^{(l)}x + b^{(l)}) \tag{1.1}$$

We will consider $N$ data points, each with a feature vector of dimension $S_0$. We group these into an $S_0 \times N$ matrix $X$:

$$X = \begin{pmatrix} | & \cdots & | \\ x_1 & \cdots & x_N \\ | & \cdots & | \end{pmatrix} \tag{1.2}$$

Note that this is the transpose of the usual "data frame" structure.

We can now write the pre-activations and activations of the first layer as:

$$Z^{(1)} = W^{(1)}X + b^{(1)}, \quad A^{(1)} = \sigma_1(Z^{(1)}) = \sigma_1(W^{(1)}X + b^{(1)}) \tag{1.3}$$

Here, we've used "$+b^{(1)}$" as a shorthand for adding the vector $b^{(1)}$ to every column. We could write this as "$+b^{(1)}J_N^t$" if we wanted to be accurate. ($J_N$ is a column vector of $N$ ones).

Similarly, we may generally write the pre-activations and activations of layer $l$ as:

$$Z^{(l)} = W^{(l)}A^{(l-1)} + b^{(l)}, \quad A^{(l)} = \sigma_l(Z^{(l)}) = \sigma_l(W^{(l)}A^{(l-1)} + b^{(l)}) \tag{1.4}$$

Finally, we have an error function $J$ which measures the distance to some target data $T \in \mathbb{R}^{S_L \times N}$:

$$T = \begin{pmatrix} | & \cdots & | \\ t_1 & \cdots & t_N \\ | & \cdots & | \end{pmatrix} \tag{1.5}$$

We will assume the error function is of the form $J = J(T, A^{(L)})$, taking on real values. I.e. it only depends on the targets and the activations of the output layer.

# 2 Backpropagation

Forward propagation through the network is described by equation 1.1. The procedure is assumed to be done before we look at how to determine partial derivatives of $J$ through backpropagation.

## 2.1 Output layer

The derivatives with respect to the output layer weights and biases will take a rather abstract form in this general formalism:

$$\frac{\partial J}{\partial W_{ij}^{(L)}} = \sum_{k=1}^{S_L} \sum_{n=1}^{N} \frac{\partial J}{\partial A_{kn}^{(L)}} \frac{\partial A_{kn}^{(L)}}{\partial W_{ij}^{(L)}}, \quad \frac{\partial J}{\partial b_i^{(L)}} = \sum_{k=1}^{S_L} \sum_{n=1}^{N} \frac{\partial J}{\partial A_{kn}^{(L)}} \frac{\partial A_{kn}^{(L)}}{\partial b_i^{(L)}} \tag{2.1}$$

We may find the derivatives of $A^{(L)}$ with respect to the weights and biases:

$$\frac{\partial A_{kn}^{(L)}}{\partial W_{ij}^{(L)}} = \frac{\partial \sigma_L}{\partial Z_{kn}^{(L)}} \frac{\partial Z_{kn}^{(L)}}{\partial W_{ij}^{(L)}} \tag{2.2}$$

Finally, we may calculate the derivatives of $Z^{(L)}$ with respect to the weights:

$$\frac{\partial Z_{kn}^{(L)}}{\partial W_{ij}^{(L)}} = \frac{\partial}{\partial W_{ij}^{(L)}} (W^{(L)} A^{(L-1)} + b^{(L)})_{kn} = \frac{\partial}{\partial W_{ij}^{(L)}} \left( \sum_{l=1}^{S_{L-1}} W_{kl}^{(L)} A_{ln}^{(L-1)} + b_k^{(L)} \right) \tag{2.3}$$

Differentiating $W^{(L)}$ with respect to $W^{(L)}$ yields two Kronecker deltas:

$$\frac{\partial Z_{kn}^{(L)}}{\partial W_{ij}^{(L)}} = \sum_{l=1}^{S_{L-1}} \delta_{ik} \delta_{jl} A_{ln}^{(L-1)} = \delta_{ik} A_{jn}^{(L-1)} \tag{2.4}$$

Now, we may insert equations 2.2 and 2.4 into 2.1:

$$\frac{\partial J}{\partial W_{ij}^{(L)}} = \sum_{k=1}^{S_L} \sum_{n=1}^{N} \frac{\partial J}{\partial A_{kn}^{(L)}} \frac{\partial \sigma_L}{\partial Z_{kn}^{(L)}} \delta_{ik} A_{jn}^{(L-1)} = \sum_{n=1}^{N} \frac{\partial J}{\partial A_{in}^{(L)}} \frac{\partial \sigma_L}{\partial Z_{in}^{(L)}} A_{jn}^{(L-1)} \tag{2.5}$$

We can rewrite this in matrix form by using the Hadamard product between the two derivatives and swapping the indices of $A^{(L-1)}$, turning into a transpose:

$$\frac{\partial J}{\partial W^{(L)}} = \left[ \frac{\partial J}{\partial A^{(L)}} \odot \frac{\partial \sigma_L}{\partial Z^{(L)}} \right] \left( A^{(L-1)} \right)^t \tag{2.6}$$

The procedure for the biases is the same until we get to the $Z^{(L)}$ derivative:

$$\frac{\partial Z_{kn}^{(L)}}{\partial b_i^{(L)}} = \frac{\partial}{\partial b_i^{(L)}} \left( \sum_{l=1}^{S_{L-1}} W_{kl}^{(L)} A_{ln}^{(L-1)} + b_k^{(L)} \right) = \delta_{ik} \tag{2.7}$$

Reinserting all the way back to equation 2.1 we get:

$$\frac{\partial J}{\partial b_i^{(L)}} = \sum_{k=1}^{S_L} \sum_{n=1}^{N} \frac{\partial J}{\partial A_{kn}^{(L)}} \frac{\partial \sigma_L}{\partial Z_{kn}^{(L)}} \delta_{ik} = \sum_{n=1}^{N} \frac{\partial J}{\partial A_{in}^{(L)}} \frac{\partial \sigma_L}{\partial Z_{in}^{(L)}} \tag{2.8}$$

Again, we can write this is matrix notation using the Hadamard product:

$$\frac{\partial J}{\partial b_i^{(L)}} = \left[ \frac{\partial J}{\partial A^{(L)}} \odot \frac{\partial \sigma_L}{\partial Z^{(L)}} \right] J_N^t \tag{2.9}$$