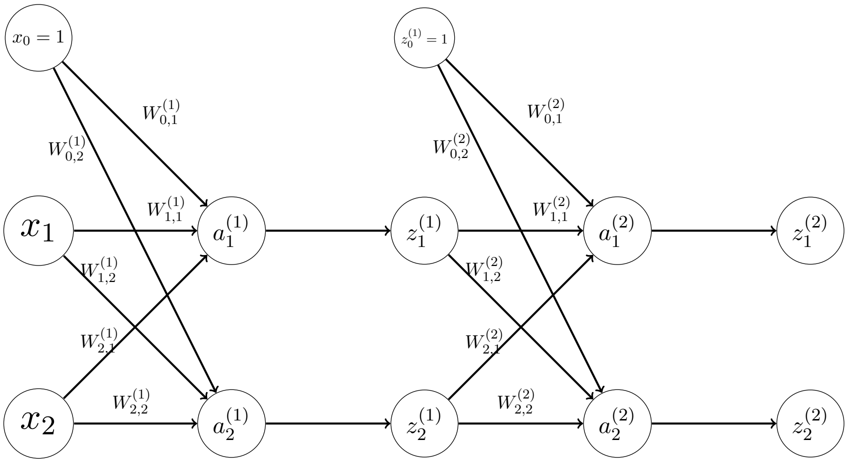
Derivation of Backpropagation Equations

Jesse Hoey
David R. Cheriton School of Computer Science
University of Waterloo
Waterloo, Ontario, CANADA, N2L3G1
jhoey@cs.uwaterloo.ca

1 Small Network

The small two-neuron (per layer) network we studied in class:



Note you can define your input vector as a column vector by transposing everything here. The forward pass is shown here with inputs as a row vector.

$$a_j^{(1)} = \sum_i x_i W_{ij}^{(1)} \quad \Rightarrow \vec{a}^{(1)} = \begin{bmatrix} a_1^{(1)} & a_2^{(1)} \end{bmatrix} = \vec{x} \cdot \mathbf{W}^{(1)} = \begin{bmatrix} x_0 & x_1 & x_2 \end{bmatrix} \begin{bmatrix} W_{01}^{(1)} & W_{02}^{(1)} \\ W_{11}^{(1)} & W_{12}^{(1)} \\ W_{21}^{(1)} & W_{22}^{(1)} \end{bmatrix} \tag{1}$$

$$z_j^{(1)} = f(a_j^{(1)}) \quad \Rightarrow \vec{z}^{(1)} = \begin{bmatrix} z_1^{(1)} & z_2^{(1)} \end{bmatrix} = \frac{1}{1 + e^{\vec{a}^{(1)}}} = \begin{bmatrix} \frac{1}{1+e^{a_1^{(1)}}} & \frac{1}{1+e^{a_2^{(1)}}} \end{bmatrix} \tag{2}$$

$$a_k^{(2)} = \sum_j z_j^{(1)} W_{jk}^{(2)} \Rightarrow \vec{a}^{(2)} = \begin{bmatrix} a_1^{(2)} & a_2^{(2)} \end{bmatrix} = \vec{z}^{(1)} \cdot \mathbf{W}^{(2)} = \begin{bmatrix} z_0^{(1)} & z_1^{(1)} & z_2^{(1)} \end{bmatrix} \begin{bmatrix} W_{01}^{(2)} & W_{02}^{(2)} \\ W_{11}^{(2)} & W_{12}^{(2)} \\ W_{21}^{(2)} & W_{22}^{(2)} \end{bmatrix} \tag{3}$$

$$z_k^{(2)} = f(a_k^{(2)}) \quad \Rightarrow \vec{z}^{(2)} = \begin{bmatrix} z_1^{(2)} & z_2^{(2)} \end{bmatrix} = \frac{1}{1 + e^{\vec{a}^{(2)}}} = \begin{bmatrix} \frac{1}{1+e^{a_1^{(2)}}} & \frac{1}{1+e^{a_2^{(2)}}} \end{bmatrix} \tag{4}$$

Compute the errors and derivatives at each level, where · means matrix dot product and × means element-wise multiplication.

$$Error(\vec{z}^{(2)}, Y) = \frac{1}{n}\sum_{E}\left(\vec{y} - \vec{z}^{(2)}\right)^2 \tag{5}$$

$$\frac{\partial Error}{\partial \vec{z}^{(2)}} = \left[-\tfrac{2}{n}\textstyle\sum_E \left(y_1 - z_1^{(2)}\right) \quad -\tfrac{2}{n}\textstyle\sum_E \left(y_2 - z_2^{(2)}\right)\right] \tag{6}$$

$$\vec{\delta}^{(2)} = \frac{\partial Error}{\partial \vec{z}^{(2)}} f'(\vec{a}^{(2)}) = \frac{\partial Error}{\partial \vec{z}^{(2)}} \times f(\vec{a}^{(2)}) \times \left(1 - f(\vec{a}^{(2)})\right) \tag{7}$$

$$\vec{\delta}^{(1)} = \begin{bmatrix}\delta_1^{(1)} & \delta_2^{(1)}\end{bmatrix} = \vec{\delta}^{(2)} \cdot \mathbf{W}^{(2)} \times f'(\vec{a}^{(1)}) \tag{8}$$

Now compute the weight updates, where M^T means the transpose of matrix or vector M.

$$\frac{\partial Error}{\partial \mathbf{W}^{(1)}_{j,\cdot}} = \vec{x}^T\vec{\delta}^{(1)} = \begin{bmatrix}\delta_1^{(1)}x_o & \delta_2^{(1)}x_0 \\ \delta_1^{(1)}x_1 & \delta_1^{(1)}x_1 \\ \delta_1^{(1)}x_2 & \delta_2^{(1)}x_2\end{bmatrix} \tag{9}$$

$$\frac{\partial Error}{\partial \mathbf{W}^{(2)}_{k,\cdot}} = \vec{z}^{(1)T}\vec{\delta}^{(2)} = \begin{bmatrix}\delta_1^{(2)}z_0^{(1)} & \delta_2^{(2)}z_0^{(1)} \\ \delta_1^{(2)}z_1^{(1)} & \delta_2^{(2)}z_1^{(1)} \\ \delta_1^{(2)}z_2^{(1)} & \delta_2^{(2)}z_2^{(1)}\end{bmatrix} \tag{10}$$

so that

$$\mathbf{W}^{(1)} \leftarrow \mathbf{W}^{(1)} - \eta\vec{x}^T\vec{\delta}^{(1)} \tag{11}$$

$$\mathbf{W}^{(2)} \leftarrow \mathbf{W}^{(2)} - \eta\vec{z}^{(1)T}\vec{\delta}^{(2)} \tag{12}$$

In general, starting with l as the output layer:

$$\vec{\delta}^{(l)} = \frac{\partial Error}{\partial \vec{z}^{(l)}} \times f'(\vec{a}^{(l)}) \tag{13}$$

and then doing all other layers starting from $l - 1$ (next to last layer):

$$\vec{\delta}^{(l)} = \left(\vec{\delta}^{(l+1)} \cdot \mathbf{W}^{(l+1)}\right) \times f'(\vec{a}^{(2)}) \tag{14}$$

Then the weight updates for the output layer:

$$\frac{\partial Error}{\partial \mathbf{W}^{(l)}} = \vec{x}^T\vec{\delta}^{(l)} \tag{15}$$

and other layers:

$$\frac{\partial Error}{\partial \mathbf{W}^{(l)}} = \vec{z}^{(l-1)T}\vec{\delta}^{(l)} \tag{16}$$

and finally update the weights:

$$\mathbf{W}^{(l)} \leftarrow \mathbf{W}^{(l)} - \eta\frac{\partial Error}{\partial \mathbf{W}^{(l)}}$$

here is a matlab sesssion showing a single weight update:

```
W1=[0.1 0.2; 0.7 -0.4; -0.1 0.2];
W2=[0.3 0.4; -0.1 0.65; 0.8 -0.1];
%input vector
X=[1 1 2]
%output vector
y=[0.7 0.2]
%learning rate
eta=0.1
>>a1=X*W1
a1 =
    0.6000    0.2000

>>z1=1./(1+exp(-a1))
z1 =
    0.6457    0.5498

>>a2=[1 z1]*W2
a2 =
    0.6753    0.7647

>>z2=1./(1+exp(-a2))
z2 =
    0.6627    0.6824

>>derror=-(2/1)*(y-z2)
derror =
   -0.0748    0.9728

>>del2=derror.*z2.*(1-z2)
del2 =
   -0.0167    0.2094

>>del1=del2*(W2(2:3,:)’).*z1.*(1-z1)
del1 =
    0.0318   -0.0082

>> dErrorDW1=X’*del1
dErrorDW1 =
    0.0318   -0.0082
    0.0318   -0.0082
    0.0635   -0.0164

>> dErrorDW2=[1; z1’]*del2
dErrorDW2 =
   -0.0167    0.2094
   -0.0108    0.1356
   -0.0092    0.1151
```

```
>> W1=W1-eta*dErrorDW1
W1 =
    0.0983    0.2006
    0.6983   -0.3994
   -0.1034    0.2012

>> W2=W2-eta*dErrorDW2
W2 =

    0.3017    0.3891
   -0.0989    0.6429
    0.8010   -0.1060
```

Thus, our update has given a new set of weights which can be compared to the original weights. The difference is not very large.