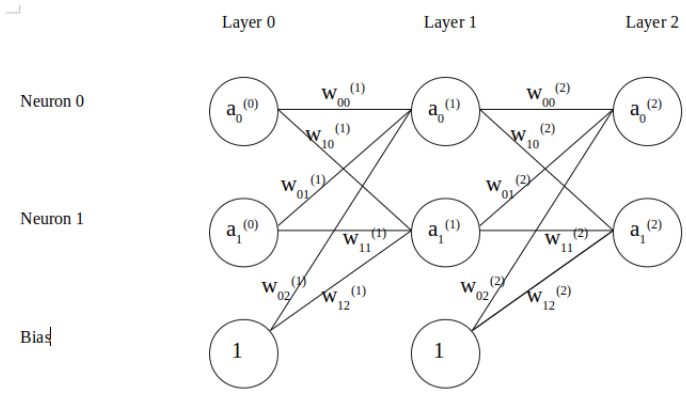
Brains is made up of neurons connected to each other. By making a mathematical model of some neurons wired together (a neural network) one can emulate some aspects of a brain.
In this note we will look at a very simple neural network in detail and show how matrices can be used to describe the processing and learning of the network.
The matrix formulation works for bigger networks as well.
The network we will consider here consists of two input neurons (or nodes), two output neurones and a single hidden layer with two neurons.



Forward pass:

$$a_0^{(1)} = \sigma(w_{00}^{(1)} a_0^{(0)} + w_{01}^{(1)} a_1^{(0)} + w_{02}^{(1)}) \tag{1}$$

$$a_1^{(1)} = \sigma(w_{10}^{(1)} a_0^{(0)} + w_{11}^{(1)} a_1^{(0)} + w_{12}^{(1)}) \tag{2}$$

$$a_0^{(2)} = \sigma(w_{00}^{(2)} a_0^{(1)} + w_{01}^{(2)} a_1^{(1)} + w_{02}^{(2)}) \tag{3}$$

$$a_1^{(2)} = \sigma(w_{10}^{(2)} a_0^{(1)} + w_{11}^{(2)} a_1^{(1)} + w_{12}^{(2)}) \tag{4}$$

Or in matrix notation:

$$\vec{a}^{(1)} = \begin{pmatrix} a_0^{(1)} \\ a_1^{(1)} \end{pmatrix} = \sigma\left(\begin{pmatrix} w_{00}^{(1)} a_0^{(0)} + w_{01}^{(1)} a_1^{(0)} + w_{02}^{(1)} \\ w_{10}^{(1)} a_0^{(0)} + w_{11}^{(1)} a_1^{(0)} + w_{12}^{(1)} \end{pmatrix}\right) = \sigma\left((a_0^{(0)} a_1^{(0)} 1) \begin{pmatrix} w_{00}^{(1)} & w_{10}^{(1)} \\ w_{01}^{(1)} & w_{11}^{(1)} \\ w_{02}^{(1)} & w_{12}^{(1)} \end{pmatrix} \right) = \sigma(\vec{A}^{(0)} W^{(1)})$$

$$\vec{a}^{(2)} = \sigma(\vec{A}^{(1)} W^{(2)})$$

Note that $\vec{a}^{(1)} = \binom{a_0^{(1)}}{a_1^{(1)}}$ and $\vec{A}^{(1)} = (a_0^{(1)} a_1^{(1)} 1)$ both contain the values for the nodes in layer (1). $\vec{A}^{(1)}$ is needed to include the biases.

The costfunction is

$$C = \frac{1}{2}(a_0^{(2)} - T_0)^2 + \frac{1}{2}(a_1^{(2)} - T_1)^2$$

or using (3) and (4)

$$\begin{aligned} C = \frac{1}{2}(\sigma(w_{00}^{(2)} a_0^{(1)} + w_{01}^{(2)} a_1^{(1)} + w_{02}^{(2)}) - T_0)^2 + \\ \frac{1}{2}(\sigma(w_{10}^{(2)} a_0^{(1)} + w_{11}^{(2)} a_1^{(1)} + w_{12}^{(2)}) - T_1)^2 \end{aligned} \tag{5}$$

We cand find how much each weight in layer (2) have to change by finding the derivative of C with respect to each weight.

Let's start with finding $\frac{dC}{dw_{00}^{(2)}}$

Only the first term of C in equation (5) include $w_{00}^{(2)}$. This term is a composite function $f \circ g \circ h(w_{00}^{(2)}, w_{01}^{(2)}, w_{02}^{(2)}, w_{10}^{(2)}, w_{11}^{(2)}, w_{12}^{(2)})$ where f, g and h and their derivatives are

$$f(\sigma) = \frac{1}{2}(\sigma - T_0)^2 \qquad \frac{df}{d\sigma} = (\sigma - T_0)$$

$$g(h) = \sigma(h) \qquad \frac{dg}{dh} = \frac{d\sigma}{dh} = \sigma(h)(1 - \sigma(h))$$

$$h(w_{00}^{(2)}, w_{01}^{(2)}, w_{02}^{(2)}) = w_{00}^{(2)} a_0^{(1)} + w_{01}^{(2)} a_1^{(1)} + w_{02}^{(2)} \qquad \frac{dh}{dw_{00}^{(2)}} = a_0^{(1)}$$

By using (3) we note that

$$(\sigma - T_0) = (a_0^{(2)} - T_0)$$

and

$$\sigma(h)(1 - \sigma(h)) = a_0^{(2)}(1 - a_0^{(2)})$$

The chain rule

$$\frac{dC}{dw_{00}^{(2)}} = \frac{df}{d\sigma} \frac{d\sigma}{dh} \frac{dh}{dw_{00}^{(2)}}$$

then gives us $\frac{dC}{dw_{00}^{(2)}} = (a_0^{(2)} - T_0)\ a_0^{(2)}(1 - a_0^{(2)})\ a_0^{(1)}$, the first entry in the change matrix.

The rest is found in a similar manner:

$$\begin{pmatrix} \frac{dC}{dw_{00}^{(2)}} & \frac{dC}{dw_{01}^{(2)}} & \frac{dC}{dw_{02}^{(2)}} \\ \frac{dC}{dw_{10}^{(2)}} & \frac{dC}{dw_{11}^{(2)}} & \frac{dC}{dw_{12}^{(2)}} \end{pmatrix} =$$

$$\begin{pmatrix} (a_0^{(2)} - T_0)a_0^{(2)}(1 - a_0^{(2)})a_0^{(1)} & (a_0^{(2)} - T_0)a_0^{(2)}(1 - a_0^{(2)})a_1^{(1)} & (a_0^{(2)} - T_0)a_0^{(2)}(1 - a_0^{(2)})1 \\ (a_1^{(2)} - T_1)a_1^{(2)}(1 - a_1^{(2)})a_0^{(1)} & (a_1^{(2)} - T_1)a_1^{(2)}(1 - a_1^{(2)})a_1^{(1)} & (a_1^{(2)} - T_1)a_1^{(2)}(1 - a_1^{(2)})1 \end{pmatrix}$$

We note that each entry in the upper row contains $\delta_0^{(2)} = (a_0^{(2)} - T_0)a_0^{(2)}(1 - a_0^{(2)})$ and that each entry in the lower row contains $\delta_1^{(2)} = (a_1^{(2)} - T_1)a_1^{(2)}(1 - a_1^{(2)})$.

Therefore the transposed change matrix for layer (2) is

$$\frac{dC}{dW^{(2)}} = \begin{pmatrix} \frac{dC}{dw_{00}^{(2)}} & \frac{dC}{dw_{01}^{(2)}} & \frac{dC}{dw_{02}^{(2)}} \\ \frac{dC}{dw_{10}^{(2)}} & \frac{dC}{dw_{11}^{(2)}} & \frac{dC}{dw_{12}^{(2)}} \end{pmatrix} = \begin{pmatrix} \delta_0^{(2)} a_0^{(1)} & \delta_0^{(2)} a_1^{(1)} & \delta_0^{(2)} \cdot 1 \\ \delta_1^{(2)} a_0^{(1)} & \delta_1^{(2)} a_1^{(1)} & \delta_1^{(2)} \cdot 1 \end{pmatrix} = \delta^{(2)} A^{(1)}$$

where $\delta^{(2)} = \binom{\delta_0^{(2)}}{\delta_1^{(2)}}$ and we had $A^{(1)} = (a_0^{(1)} a_1^{(1)} 1)$.
After all change matrices is found the change is applied to all weights. For layer (2) this will be

$$W^{(2)} = W^{(2)} - \gamma \cdot \frac{dC}{dW^{(2)}}$$

where γ is a number between 0 and 1 called the learning rate. A good start value for γ is usually 0.5

To find how much each weight in layer (1) have to change we can use (1) and (2) in (5)

$$\begin{aligned} C = \frac{1}{2}&\left(\sigma\left(w_{00}^{(2)}\sigma(w_{00}^{(1)}a_0^{(0)} + w_{01}^{(1)}a_1^{(0)} + w_{02}^{(1)}) + w_{01}^{(2)}\sigma(w_{10}^{(1)}a_0^{(0)} + w_{11}^{(1)}a_1^{(0)} + w_{12}^{(1)}) + w_{02}^{(2)}\right) - T_0\right)^2 + \\ \frac{1}{2}&\left(\sigma\left(w_{10}^{(2)}\sigma(w_{00}^{(1)}a_0^{(0)} + w_{01}^{(1)}a_0^{(0)} + w_{02}^{(1)}) + w_{11}^{(2)}\sigma(w_{10}^{(1)}a_0^{(0)} + w_{11}^{(1)}a_1^{(0)} + w_{12}^{(1)}) + w_{12}^{(2)}\right) - T_1\right)^2 \end{aligned} \tag{6}$$

We want to find $\frac{dC}{dw_{00}^{(1)}}$ and note that $dw_{00}^{(1)}$ occur in both terms of (6). The composite function is a little more complicated, but nothing new happens.

A little work will show that

$$\frac{dC}{dw_{00}^{(1)}} = (a_0^{(2)} - T_0)\ a_0^{(2)}(1-a_0^{(2)})w_{00}^{(2)}\ a_0^{(1)}(1-a_0^{(1)})\ a_0^{(0)} + (a_1^{(2)} - T_1)\ a_1^{(2)}(1-a_1^{(2)})w_{10}^{(2)}\ a_0^{(1)}(1-a_0^{(1)})\ a_0^{(0)}$$

$$\frac{dC}{dw_{01}^{(1)}} = (a_0^{(2)} - T_0)\ a_0^{(2)}(1-a_0^{(2)})w_{00}^{(2)}\ a_0^{(1)}(1-a_0^{(1)})\ a_1^{(0)} + (a_1^{(2)} - T_1)\ a_1^{(2)}(1-a_1^{(2)})w_{10}^{(2)}\ a_0^{(1)}(1-a_0^{(1)})\ a_1^{(0)}$$

$$\frac{dC}{dw_{02}^{(1)}} = (a_0^{(2)} - T_0)\ a_0^{(2)}(1-a_0^{(2)})w_{00}^{(2)}\ a_0^{(1)}(1-a_0^{(1)})\cdot 1 + (a_1^{(2)} - T_1)\ a_1^{(2)}(1-a_1^{(2)})w_{10}^{(2)}\ a_0^{(1)}(1-a_0^{(1)})\cdot 1$$

$$\frac{dC}{dw_{10}^{(1)}} = (a_0^{(2)} - T_0)\ a_0^{(2)}(1-a_0^{(2)})w_{01}^{(2)}\ a_1^{(1)}(1-a_1^{(1)})\ a_0^{(0)} + (a_1^{(2)} - T_1)\ a_1^{(2)}(1-a_1^{(2)})w_{11}^{(2)}\ a_1^{(1)}(1-a_1^{(1)})\ a_0^{(0)}$$

$$\frac{dC}{dw_{11}^{(1)}} = (a_0^{(2)} - T_0)\ a_1^{(2)}(1-a_1^{(2)})w_{01}^{(2)}\ a_0^{(1)}(1-a_1^{(1)})\ a_1^{(0)} + (a_1^{(2)} - T_1)\ a_1^{(2)}(1-a_1^{(2)})w_{11}^{(2)}\ a_1^{(1)}(1-a_1^{(1)})\ a_1^{(0)}$$

$$\frac{dC}{dw_{12}^{(1)}} = (a_0^{(2)} - T_0)\ a_0^{(2)}(1-a_0^{(2)})w_{01}^{(2)}\ a_1^{(1)}(1-a_1^{(1)})\cdot 1 + (a_1^{(2)} - T_1)\ a_1^{(2)}(1-a_1^{(2)})w_{11}^{(2)}\ a_1^{(1)}(1-a_1^{(1)})\cdot 1$$

If we again use $\delta_0^{(2)} = (a_0^{(2)} - T_0)a_0^{(2)}(1-a_0^{(2)})$ and $\delta_1^{(2)} = (a_1^{(2)} - T_1)a_1^{(2)}(1-a_1^{(2)})$ we can write

$$\frac{dC}{dw_{00}^{(1)}} = \delta_0^{(2)}w_{00}^{(2)}\ a_0^{(1)}(1-a_0^{(1)})a_0^{(0)} + \delta_1^{(2)}w_{10}^{(2)}\ a_0^{(1)}(1-a_0^{(1)})a_0^{(0)}$$

and by letting $\delta_0^{(1)} = a_0^{(1)}(1-a_0^{(1)})\sum_{i=0}^{1}\delta_i^{(2)}w_{i0}^{(2)}$ we have $\frac{dC}{dw_{00}^{(1)}} = \delta_0^{(1)}a_0^{(0)}$
With $\delta_1^{(1)} = a_1^{(1)}(1-a_1^{(1)})\sum_{i=0}^{1}\delta_i^{(2)}w_{i1}^{(2)}$ the transposed change matrix for layer (1) is

$$\frac{dC}{dW^{(1)}} = \begin{pmatrix} \frac{dC}{dw_{00}^{(1)}} & \frac{dC}{dw_{01}^{(1)}} & \frac{dC}{dw_{02}^{(1)}} \\ \\ \frac{dC}{dw_{10}^{(1)}} & \frac{dC}{dw_{11}^{(1)}} & \frac{dC}{dw_{12}^{(1)}} \end{pmatrix} = \begin{pmatrix} \delta_0^{(1)}a_0^{(0} & \delta_0^{(1)}a_1^{(0)} & \delta_0^{(1)}\cdot 1 \\ \\ \delta_1^{(1)}a_0^{(0)} & \delta_1^{(1)}a_1^{(0)} & \delta_1^{(2)}\cdot 1 \end{pmatrix} = \delta^{(1)}A^{(0)}$$

where $\delta^{(1)} = \binom{\delta_0^{(1)}}{\delta_1^{(1)}}$ and $A^{(0)} = (a_0^{(0)}a_1^{(0)}1)$.

By introducing the matrices

$$c = \begin{pmatrix} a_0^{(2)} - T^0 \\ a_1^{(2)} - T^1 \end{pmatrix} \qquad w_2 = \begin{pmatrix} w_{00}^{(2)} & w_{01}^{(2)} \\ w_{10}^{(2)} & w_{11}^{(2)} \end{pmatrix}$$

$$D^{(2)} = \begin{pmatrix} a_0^{(2)}(1-a_0^{(2)}) & 0 \\ 0 & a_1^{(2)}(1-a_1^{(2)}) \end{pmatrix} \qquad D^{(1)} = \begin{pmatrix} a_0^{(1)}(1-a_0^{(1)}) & 0 \\ 0 & a_0^{(1)}(1-a_0^{(1)}) \end{pmatrix}$$

we can write $\delta^{(2)} = \begin{pmatrix} \delta_0^{(2)} \\ \delta_1^{(2)} \end{pmatrix} = D^{(2)}\cdot c = \begin{pmatrix} a_0^{(2)}(1-a_0^{(2)}) & 0 \\ 0 & a_1^{(2)}(1-a_1^{(2)}) \end{pmatrix}\begin{pmatrix} a_0^{(2)} - T_0 \\ a_1^{(2)} - T_1 \end{pmatrix} = \begin{pmatrix} (a_0^{(2)} - T_0)a_0^{(2)}(1-a_0^{(2)}) \\ (a_1^{(2)} - T_1)a_1^{(2)}(1-a_1^{(2)}) \end{pmatrix}$
Then

$$D^{(1)}w^{(2)}\delta^{(2)} = \begin{pmatrix} a_0^{(1)}(1-a_0^{(1)}) & 0 \\ 0 & a_1^{(1)}(1-a_1^{(1)}) \end{pmatrix}\begin{pmatrix} w_{00}^{(2)} & w_{01}^{(2)} \\ w_{10}^{(2)} & w_{11}^{(2)} \end{pmatrix}\begin{pmatrix} \delta_0^{(2)} \\ \delta_1^{(2)} \end{pmatrix}$$

$$= \begin{pmatrix} w_{00}^{(2)}a_0^{(1)}(1-a_0^{(1)}) & w_{10}^{(2)}a_0^{(1)}(1-a_0^{(1)}) \\ w_{01}^{(2)}a_1^{(1)}(1-a_1^{(1)}) & w_{11}^{(2)}a_1^{(1)}(1-a_1^{(1)}) \end{pmatrix}\begin{pmatrix} \delta_0^{(2)} \\ \delta_1^{(2)} \end{pmatrix} = \begin{pmatrix} \delta_0^{(2)}w_{00}^{(2)}a_0^{(1)}(1-a_0^{(1)}) + \delta_1^{(2)}w_{10}^{(2)}a_0^{(1)}(1-a_0^{(1)}) \\ \delta_0^{(2)}w_{01}^{(2)}a_1^{(1)}(1-a_1^{(1)}) + \delta_1^{(2)}w_{11}^{(2)}a_1^{(1)}(1-a_1^{(1)}) \end{pmatrix} = \begin{pmatrix} \delta_0^{(1)} \\ \delta_1^{(1)} \end{pmatrix} = \delta^{(1)}$$

To summarise

$$\frac{dC}{dW^{(1)}} = \delta^{(1)}A^{(0)} = D^{(1)}w^{(2)}\delta^{(2)}A^{(0)}$$

and

$$\frac{dC}{dW^{(2)}} = \delta^{(2)} A^{(1)}$$

δ is sometimes called the "error" because it tell us how much the weights have to change, and part of δ is the difference between the output and the target $(a_0^{(2)} - T_0)$. We also see that neurons with high values of a contribute more to the change matrix, kind of like more active neurons in a biological network tend to reinforce learning in the network.
A note for implementation:
When the network becomes bigger multiplying matrices like $D^{(1)}$ will eat up unnecessary computer cycles due to the many zeroes in diagonal matrices.
This can be avoided by using the lesser known Hadamard product $\odot$.
(https://en.wikipedia.org/wiki/Hadamard_product_(matrices))
Instead of

$$\begin{pmatrix} a_0^{(2)}(1 - a_0^{(2)}) & 0 \\ 0 & a_1^{(2)}(1 - a_1^{(2)}) \end{pmatrix} \begin{pmatrix} a_0^{(2)} - T_0 \\ a_1^{(2)} - T_1 \end{pmatrix} = \begin{pmatrix} (a_0^{(2)} - T_0)a_0^{(2)}(1 - a_0^{(2)}) \\ (a_1^{(2)} - T_1)a_1^{(2)}(1 - a_1^{(2)}) \end{pmatrix}$$

we would write

$$\begin{pmatrix} a_0^{(2)}(1 - a_0^{(2)}) \\ a_1^{(2)}(1 - a_1^{(2)}) \end{pmatrix} \odot \begin{pmatrix} a_0^{(2)} - T_0 \\ a_1^{(2)} - T_1 \end{pmatrix} = \begin{pmatrix} (a_0^{(2)} - T_0)a_0^{(2)}(1 - a_0^{(2)}) \\ (a_1^{(2)} - T_1)a_1^{(2)}(1 - a_1^{(2)}) \end{pmatrix}$$

References:
https://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example/
https://page.mi.fu-berlin.de/rojas/neural/chapter/K7.pdf
http://neuralnetworksanddeeplearning.com/chap2.html
http://cs231n.stanford.edu/handouts/linear-backprop.pdf
https://sudeepraja.github.io/Neural/