Error backpropagation in 2 layer neural network

Sadaf Iqbal, Muhammad Suleman

December 14, 2018

Abstract

This document discusses step by step error back propagation calculation, its mathematics and implementation in Python.

1 Introduction

We have discussed the maths and implementation details of error back propagation in 1 hidden layer neural network. This network consists of 2 input features x_1 and x_2 and an output layer with 2 neurons o_1 and o_2. The hidden layer consists of only 2 neurons h_1 and h_2. There are 2 bias terms in the network b_1 and b_2 for the hidden layer and output layer respectively.

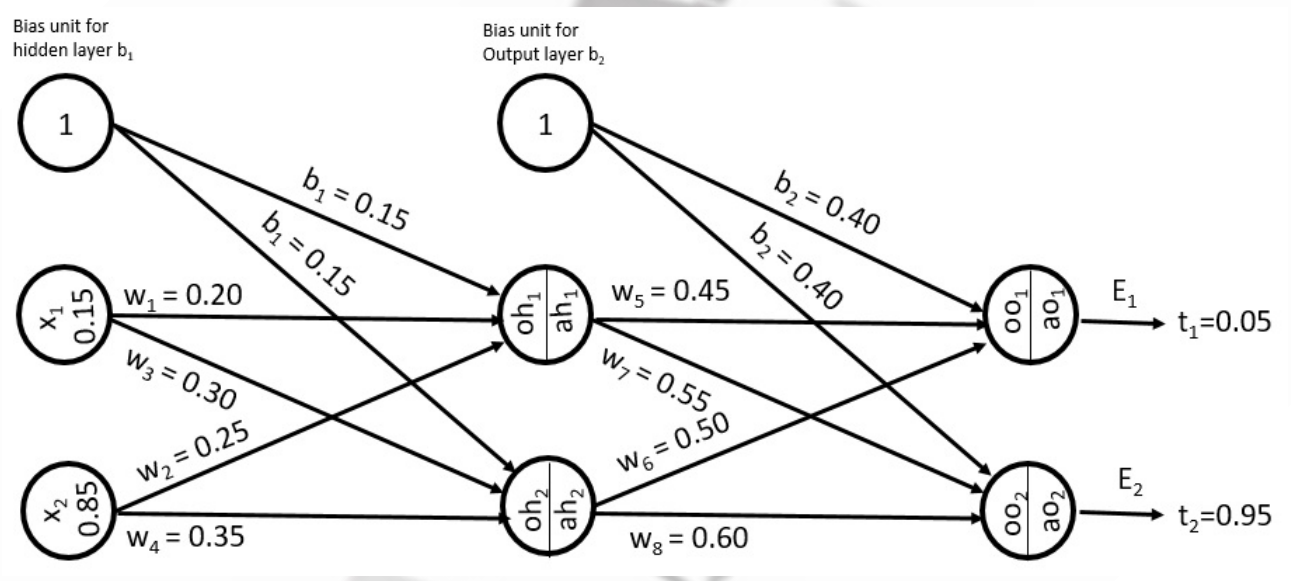

Figure 1: Neural Network Design

2 Forward Pass

$$oh_1 = b_1 + x_1 * w_1 + x_2 * w_2$$

$$oh_1 = 0.15 + 0.15 * 0.20 + 0.85 * 0.25$$

$$oh_1 = 0.3925$$

$$ah_1 = \sigma(oh_1)$$

$$ah_1 = \sigma(0.3925)$$

$$ah_1 = 0.59688$$

$$oh_2 = b_1 + x_1 * w_3 + x_2 * w_4 = 0.4925$$

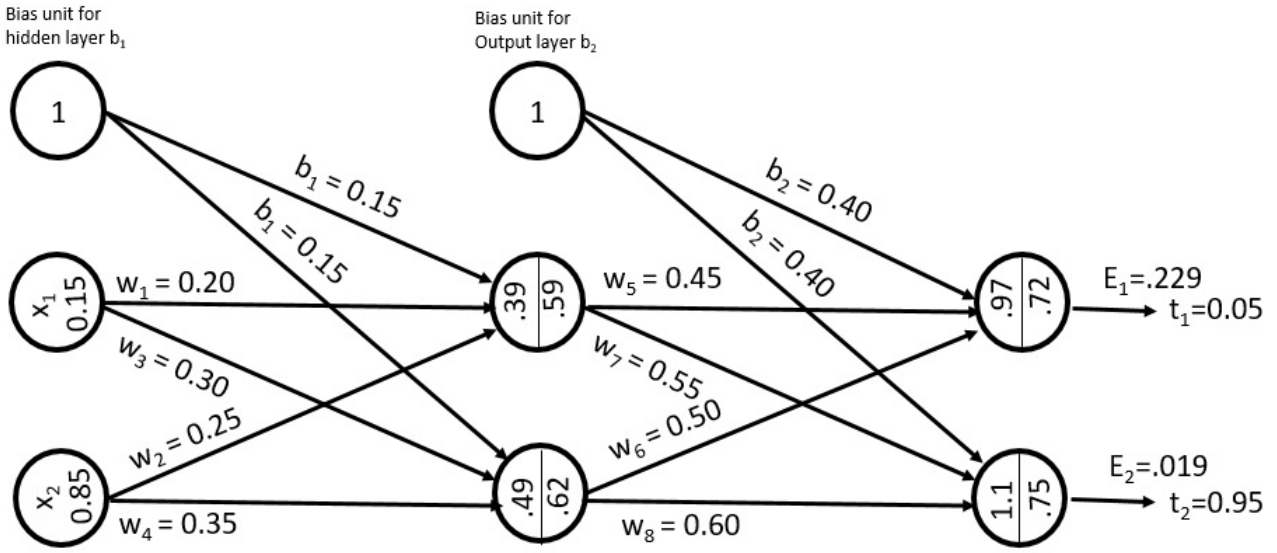

Figure 2: After Forward Pass.

```
[1]  def forward_pass(o, a, w):
         for i in range(2):
             for j in range(2):
                 o[i][j] = np.dot(a[i], w[i][j])
                 a[i+1][j+1] = 1 / (1 + np.exp(-1*o[i][j]))
             a[i+1][0] = 1
         #print(o,a)
```

Figure 3: Code for Forward Pass.

$$ah_2 = \sigma(oh_2) = 0.62069$$

$$oo_1 = b_2 + ah_1 * w_5 + ah_2 * w_6 = 0.97894$$

$$ao_1 = \sigma(oo_1) = 0.72689$$

$$oo_2 = b_2 + ah_1 * w_7 + ah_2 * w_8 = 1.1$$

$$ao_2 = \sigma(oo_2) = 0.75039$$

3 Error Calculation

If targets are represented by t_1 and t_2 then error is calculated using sum of squared error.

$$E_1 = \frac{(t_1 - ao_1)^2}{2}$$

$$E_1 = \frac{(0.05 - 72689)^2}{2}$$

$$E_1 = 0.22909$$

$$E_2 = \frac{(t_2 - ao_2)^2}{2}$$

```
[2]  def total_error(a, t, e):
         e  = np.power(t - a[len(a)-1][1:3], 2) / 2
         print(np.sum(e))
```

Figure 4: Code for Error Calculation.

$$E_2 = 0.01992$$

$$E = E_1 + E_2$$

$$E = \frac{(t_1 - ao_1)^2}{2} + \frac{(t_2 - ao_2)^2}{2}$$

$$E = \frac{(t_1 - ao_1)^2 + (t_2 - ao_2)^2}{2}$$

$$E = 0.24901$$

4 Backward Pass

Now, we need to update weights at both the layers i.e. Output layer (w_5, w_6, w_7 and w_8) and Hidden layer (w_1, w_2, w_3 and w_4). For this first, we need to understand the derivative of sigmoid function as in the forward pass we calculate the sigmoid represented by notation 'a' of weighted sum represented by 'o'.

4.1 Derivative of Sigmoid function

Let $g(z)$ or $\sigma(z)$ represent the sigmoid function which is mathematically written as

$$g(z) = \sigma(z) = \frac{1}{1 + e^{-z}}$$

$$g^{'}(z) = \sigma^{'}(z)$$

$$\frac{\partial}{\partial z} g(z) = \frac{\partial}{\partial z} \sigma(z) = \sigma^{'}(z) = \frac{\partial}{\partial z} \Big(\frac{1}{1 + e^{-z}} \Big)$$

$$\sigma^{'}(z) = \frac{\partial}{\partial z} (1 + e^{-z})^{-1}$$

$$\sigma^{'}(z) = -1(1 + e^{-z})^{-1-1} \frac{\partial}{\partial z} (1 + e^{-z})$$

$$\sigma^{'}(z) = -1(1 + e^{-z})^{-2} (e^{-z}) \frac{\partial}{\partial z} (-z)$$

$$\sigma^{'}(z) = \frac{-1}{(1 + e^{-z})^2} (e^{-z})(-1)$$

$$\sigma^{'}(z) = \frac{e^{-z}}{(1 + e^{-z})^2}$$

$$\sigma^{'}(z) = \Big(\frac{1}{1 + e^{-z}} \Big)^2 (e^{-z})$$

$$\sigma^{'}(z) = (\sigma(z)^2 \frac{(1 - \sigma(z))}{\sigma(z)}$$

$$\sigma^{'}(z) = \sigma(z)(1 - \sigma(z))$$

$$\boxed{\sigma^{'} = \sigma(1 - \sigma)}$$

4.2 Change in error w.r.t. weights of output layer

First we are going to calculate the change in error E w.r.t the weights of output layer w_5, w_6, w_7 and w_8 respectively.

Let's first calculate the change in error E w.r.t the weight w_5 given by $\frac{\partial E}{\partial w_5}$ using chain rule

$$\frac{\partial E}{\partial w_5} = \frac{\partial E}{\partial ao_1} * \frac{\partial ao_1}{\partial oo_1} * \frac{\partial oo_1}{\partial w_5} \tag{A}$$

For $\frac{\partial E}{\partial ao_1}$:

$$\frac{\partial E}{\partial ao_1} = \frac{\partial}{\partial ao_1} \frac{(t_1 - ao_1)^2 + (t_2 - ao_2)^2}{2}$$

$$\frac{\partial E}{\partial ao_1} = \frac{\partial}{\partial ao_1} \frac{(t_1 - ao_1)^2}{2} + \frac{\partial}{\partial ao_1} \frac{(t_2 - ao_2)^2}{2}$$

$$\frac{\partial E}{\partial ao_1} = 2 * \frac{(t_1 - ao_1)}{2} * \frac{\partial}{\partial ao_1}(t_1 - ao_1) + 0$$

$$\frac{\partial E}{\partial ao_1} = (t_1 - ao_1) * (-1)$$

$$\boxed{\frac{\partial E}{\partial ao_1} = (ao_1 - t_1) = 0.67689} \tag{i}$$

For $\frac{\partial ao_1}{\partial oo_1}$:

$$\frac{\partial ao_1}{\partial oo_1} = \frac{\partial}{\partial oo_1}\sigma(oo_1)$$

$$\frac{\partial ao_1}{\partial oo_1} = \sigma(oo_1)(1 - \sigma(oo_1))$$

$$\boxed{\frac{\partial ao_1}{\partial oo_1} = ao_1 * (1 - ao_1) = 0.19851} \tag{ii}$$

For $\frac{\partial oo_1}{\partial w_5}$:

$$\frac{\partial oo_1}{\partial w_5} = \frac{\partial}{\partial w_5}(b_2 + ah_1 * w_5 + ah_2 * w_6)$$

$$\frac{\partial oo_1}{\partial w_5} = \frac{\partial}{\partial w_5}(b_2) + \frac{\partial}{\partial w_5}(ah_1 * w_5) + \frac{\partial}{\partial w_5}(ah_2 * w_6)$$

$$\frac{\partial oo_1}{\partial w_5} = 0 + ah_1 + 0$$

$$\boxed{\frac{\partial oo_1}{\partial w_5} = ah_1 = 0.59688} \tag{iii}$$

Now, substitute the values of i, ii and iii in equation A we get

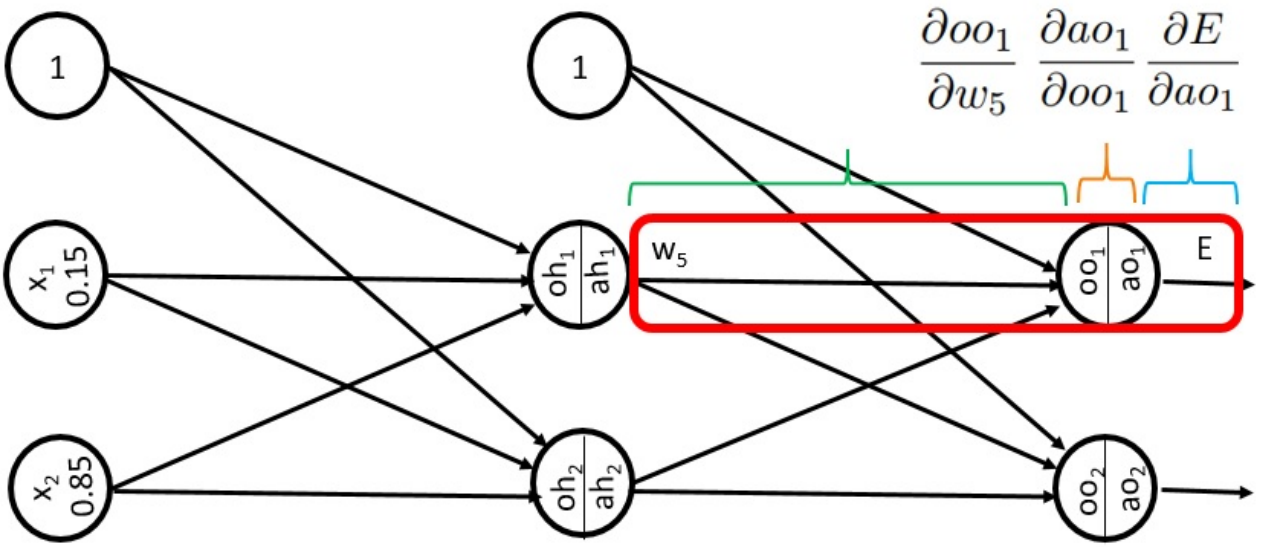

Figure 5: Calculate dawa e / dawa w5.

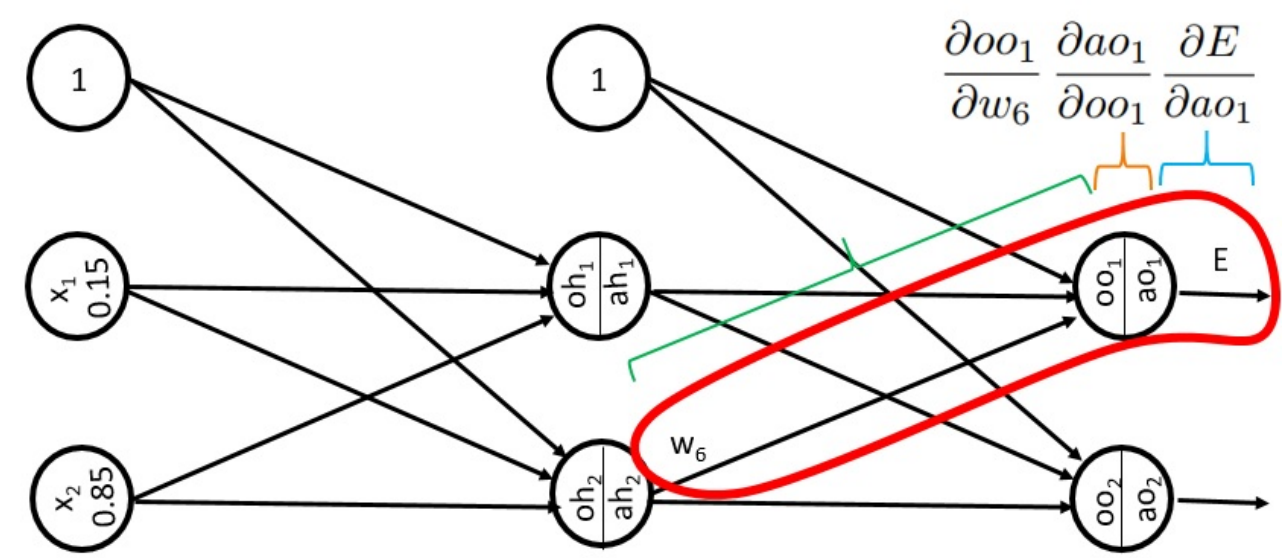

Figure 6: Calculate dawa e / dawa w6.

$$\boxed{\frac{\partial E}{\partial w_5} = (ao_1 - t_1) * ao_1 * (1 - ao_1) * ah_1 = 0.08020}$$

Similarly the change in error E w.r.t the weight w_6, w_7 and w_8 is defined as

$$\boxed{\frac{\partial E}{\partial w_6} = (ao_1 - t_1) * ao_1 * (1 - ao_1) * ah_2 = 0.08340}$$

$$\boxed{\frac{\partial E}{\partial w_7} = (ao_2 - t_2) * ao_2 * (1 - ao_2) * ah_1 = -0.02231}$$

$$\boxed{\frac{\partial E}{\partial w_8} = (ao_2 - t_2) * ao_2 * (1 - ao_2) * ah_2 = -0.02320}$$

4.3 Change in error w.r.t. weights of hidden layer

Now we are going to calculate the change in error E w.r.t the weights of hidden layer w_1, w_2, w_3 and w_4 respectively.

Let's first calculate the change in error E w.r.t the weight w_1 given by $\frac{\partial E}{\partial w_1}$ using chain rule. If you look at the picture weight w_1 is affecting both the outputs ao_1 and ao_2 so here the equation has 2 components i going backward from ao_1 to w_1 and ii going backward from ao_2 to w_1

$$\frac{\partial E}{\partial w_1} = \frac{\partial E}{\partial ao_1} * \frac{\partial ao_1}{\partial oo_1} * \frac{\partial oo_1}{\partial ah_1} * \frac{\partial ah_1}{\partial oh_1} * \frac{\partial oh_1}{\partial w_1} + \frac{\partial E}{\partial ao_2} * \frac{\partial ao_2}{\partial oo_2} * \frac{\partial oo_2}{\partial ah_1} * \frac{\partial ah_1}{\partial oh_1} * \frac{\partial oh_1}{\partial w_1}$$

The above equation is complex to read but it is very simple to understand and write, just you have to look at the variables in the reverse direction and you can easily understand it. 2 components are common in both paths i.e. $\frac{\partial ah_1}{\partial oh_1}$ and $\frac{\partial oh_1}{\partial w_1}$so let's make it simple

$$\frac{\partial E}{\partial w_1} = \Big(\frac{\partial E}{\partial ao_1} * \frac{\partial ao_1}{\partial oo_1} * \frac{\partial oo_1}{\partial ah_1} + \frac{\partial E}{\partial ao_2} * \frac{\partial ao_2}{\partial oo_2} * \frac{\partial oo_2}{\partial ah_1}\Big) * \frac{\partial ah_1}{\partial oh_1} * \frac{\partial oh_1}{\partial w_1} \tag{B}$$

From equation (i) we have

$$\boxed{\frac{\partial E}{\partial ao_1} = (ao_1 - t_1)}$$

similarly,

$$\boxed{\frac{\partial E}{\partial ao_2} = (ao_2 - t_2)}$$

and from equation (ii) we have

$$\boxed{\frac{\partial ao_1}{\partial oo_1} = ao_1 * (1 - ao_1)}$$

similarly,

$$\boxed{\frac{\partial ao_2}{\partial oo_2} = ao_2 * (1 - ao_2)}$$

Now we calculate the remaining parts of equation B as

$$\frac{\partial oo_1}{\partial ah_1} = \frac{\partial}{\partial ah_1}(b_2 + ah_1 * w_5 + ah_2 * w_6)$$

$$\boxed{\frac{\partial oo_1}{\partial ah_1} = w_5}$$

similarly,

$$\frac{\partial oo_2}{\partial ah_1} = \frac{\partial}{\partial ah_1}(b_2 + ah_1 * w_7 + ah_2 * w_8)$$

$$\boxed{\frac{\partial oo_2}{\partial ah_1} = w_7}$$

Also,

$$\frac{\partial ah_1}{\partial oh_1} = \frac{\partial}{\partial oh_1}\sigma(oh_1)$$

$$\frac{\partial ah_1}{\partial oh_1} = \sigma(oh_1)(1 - \sigma(oh_1))$$

$$\boxed{\frac{\partial ah_1}{\partial oh_1} = ah_1 * (1 - ah_1)}$$

and finally,

$$\frac{\partial oh_1}{\partial w_1} = \frac{\partial}{\partial w_1}(b_1 + x_1 * w_1 + x_2 * w_2)$$

$$\boxed{\frac{\partial oh_1}{\partial w_1} = x_1}$$

Thus,

$$\boxed{\frac{\partial E}{\partial w_1} = \Big((ao_1 - t_1) * ao_1(1 - ao_1) * w_5 \ + (ao_2 - t_2) * ao_2(1 - ao_2) * w_7\Big) * ah_1(1 - ah_1) * x_1}$$

$$\boxed{\begin{aligned}\frac{\partial E}{\partial w_1} = \Big((0.72689 - 0.05) * 0.72689 * (1 - 0.72689) * 0.45 + (0.75038 - 0.95) * 0.75038(1 - 0.75038) * 0.55\Big)\\ *0.5968(1 - 0.5968) * 0.15 = 00144\end{aligned}}$$

Similarly the change in error E w.r.t the weight w_2, w_3 and w_4 is defined as

$$\boxed{\frac{\partial E}{\partial w_2} = \Big((ao_1 - t_1) * ao_1(1 - ao_1) * w_5 \ + (ao_2 - t_2) * ao_2(1 - ao_2) * w_7\Big) * ah_1(1 - ah_1) * x_2 = 0.00816}$$

$$\boxed{\frac{\partial E}{\partial w_3} = \Big((ao_1 - t_1) * ao_1(1 - ao_1) * w_6 \ + (ao_2 - t_2) * ao_2(1 - ao_2) * w_8\Big) * ah_2(1 - ah_2) * x_1 = 0.00158}$$

$$\boxed{\frac{\partial E}{\partial w_2} = \Big((ao_1 - t_1) * ao_1(1 - ao_1) * w_6 \ + (ao_2 - t_2) * ao_2(1 - ao_2) * w_8\Big) * ah_2(1 - ah_2) * x_2 = 0.00895}$$

4.4 weight update rule

$$w_i^+ = w_i + \alpha\frac{\partial E}{\partial w_i} \qquad i = 1, 2, \ldots, 8$$

$$w_1^+ = 0.20 + 0.01 * 0.00144 = 0.1999$$
$$w_2^+ = 0.25 + 0.01 * 0.00816 = 0.2499$$
$$w_3^+ = 0.30 + 0.01 * 0.00158 = 0.2999$$
$$w_4^+ = 0.35 + 0.01 * 0.00895 = 0.3455$$
$$w_5^+ = 0.45 + 0.01 * 0.08019 = 0.4491$$
$$w_6^+ = 0.50 + 0.01 * 0.0834 = 0.4991$$
$$w_7^+ = 0.55 + 0.01 * -0.0223 = 0.5502$$
$$w_8^+ = 0.60 + 0.01 * -0.0232 = 0.6002$$

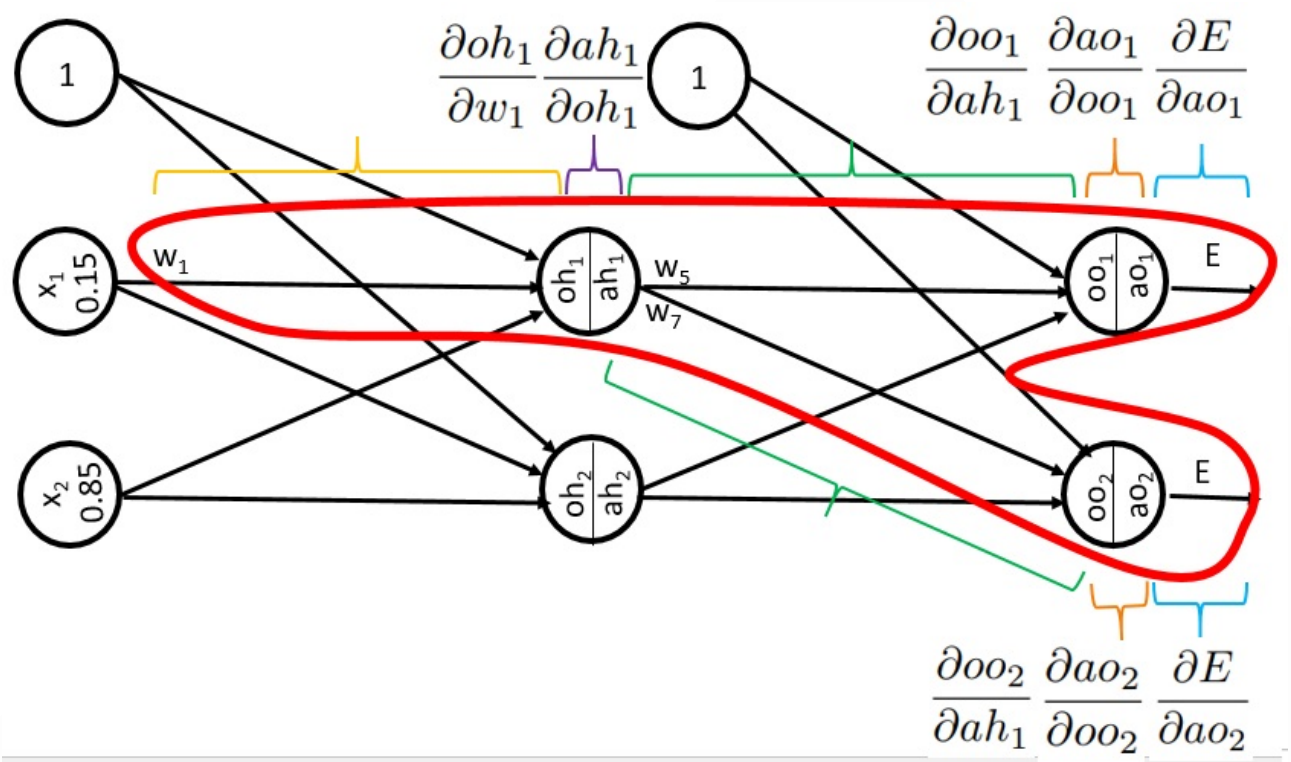

Figure 7: Calculate dawa e / dawa w1.

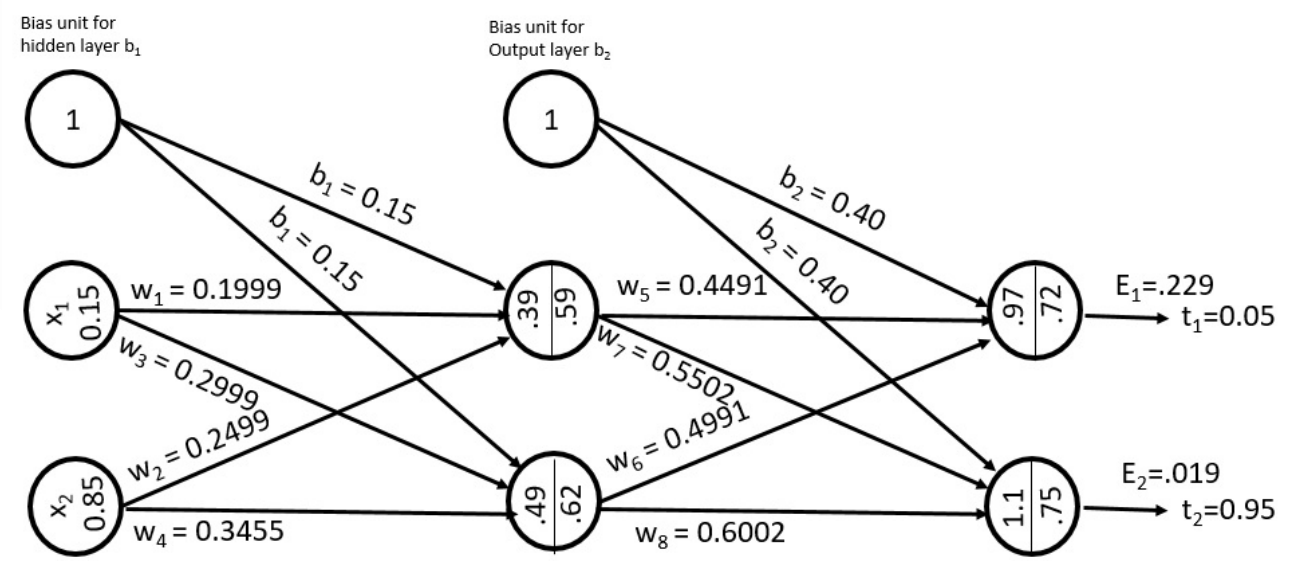

Figure 8: Weights after first update

```
[3]  def backward_pass(de_da, da_do, do_dw, a, t, alpha, de_do, w, I):

         updated_weights = np.copy(w)
         for i in range(2):
             de_da[i] = a[len(a)-1][i+1] - t[i]
             da_do[i] = a[len(a)-1][i+1] * (1 - a[len(a)-1][i+1] )
             do_dw[i] = a[len(a)-2][i+1]

         for i in range(2):
             for j in range(2):
                 delta_rule = de_da[i] * da_do[i] * do_dw[j]
                 updated_weights[len(w) - 1][i][j+1] = w[len(w) - 1][i][j+1] - alpha * delta_rule

         for i in range(2):
             de_do[i] = de_da[i] *  da_do[i]

         for i in range(2):
             for j in range(2):
                 delta_rule = np.dot(de_do, [w[1][0][i+1],w[1][1][i+1]]) * a[1][i+1] * (1 - a[1][i+1]) * I[j+1]
                 updated_weights[len(w) - 2][i][j+1] = w[len(w) - 2][i][j+1] - alpha * delta_rule

         return updated_weights
```

Figure 9: Code for Backward pass and weight update.

```
[4]  import numpy as np

     I = np.array([1, 0.15, 0.85], dtype = 'float32')
     w = np.array([
                          [[0.15, 0.2, 0.25],[0.15, 0.3, 0.35]],
                          [[0.40, 0.45, 0.5],[0.40, 0.55, 0.6]]
                          ], dtype = 'float32')

     o = np.ndarray(shape=(2,2), dtype='float32')
     a = np.ndarray(shape=(3,3), dtype='float32')
     a[0] = I
     t = np.array([0.05, 0.95], dtype='float32')
     e = np.zeros((2,), dtype='float')
     de_da = np.zeros((2,), dtype='float')
     da_do = np.zeros((2,), dtype='float')
     do_dw = np.zeros((2,), dtype='float')
     alpha = 0.5
     de_do = np.zeros((2,), dtype='float')
```

Figure 10: Code for Network Initialization.

Test Case 1

```
[5]  forward_pass(o, a, w)
```

```
[6]  total_error(a, t, e)
```

```
0.24901779
```

```
[7]  print(w)
```

```
[[[0.15 0.2  0.25]
  [0.15 0.3  0.35]]

 [[0.4  0.45 0.5 ]
  [0.4  0.55 0.6 ]]]
```

```
[8]  w = backward_pass(de_da, da_do, do_dw, a, t,  alpha, de_do, w, I)
```

```
[9]  print(w)
```

```
[[[0.15       0.19927986 0.24591918]
  [0.15       0.29920974 0.3455218 ]]

 [[0.4        0.40989655 0.45829678]
  [0.4        0.561158   0.6116031 ]]]
```

Figure 11: Code for Single step.

Test Case 2

```
[10]  epochs = 200

      for i in range(epochs):
          forward_pass(o, a, w)
          print("\nPrinting Error for epoch", i)
          total_error(a, t, e)
          w = backward_pass(de_da, da_do, do_dw, a, t,  alpha, de_do, w, I)
```

```
0.0026752078

Printing Error for epoch 181
0.0026450509

Printing Error for epoch 182
0.0026153638

Printing Error for epoch 183
0.002586147
```

Figure 12: Code for 200 epochs.