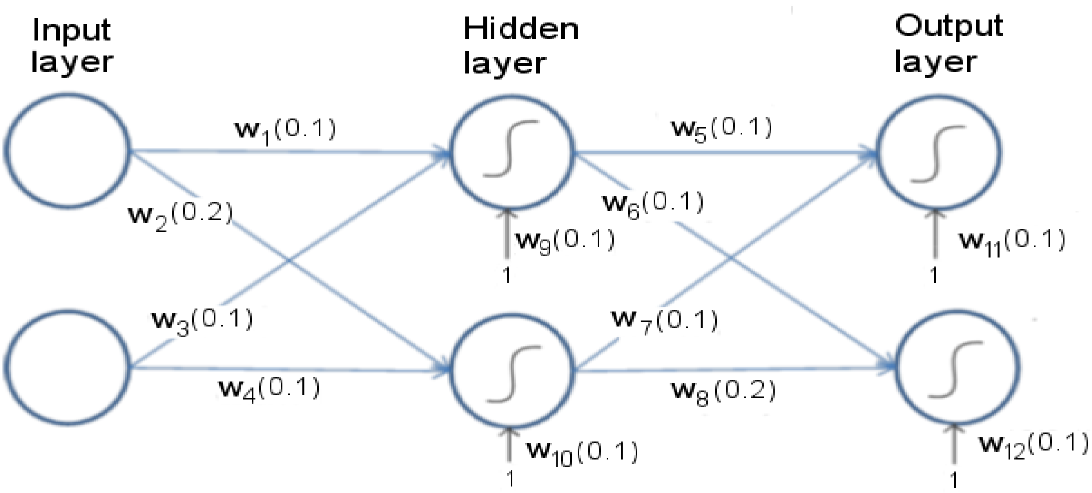
# 2802ICT: Assignment 2

Simple Neural Network

## Overview

For this exercise, we're going to use a neural network with two inputs, two hidden neurons, two output neurons. Additionally, the hidden and output neurons will include a bias.

Here's the structure:




Note that there are no biases for the neurons in the input layer. There are two training samples: $X_1$ = (0.1, 0.1) and $X_2$ = (0.1, 0.2). The label for $X_1$ is 0, so the desired output for $X_1$ in the output layer should be $Y_1$ = (1, 0). The label for $X_2$ is 1, so the desired output for $X_2$ in the output layer should be $Y_2$ = (0, 1).

## The Forward Pass

To begin, let's see what the neural network currently predicts given the weights and biases above and inputs of 0.1 and 0.1. To do this we'll feed those inputs forward though the network.

We figure out the total net input to each hidden layer neuron, squash the total net input using an activation function (here we use the logistic function), then repeat the process with the output layer neurons. From here on out, the input neurons will be referred to as $I_1$ and $I_2$, the hidden layer neurons are $H_1$ and $H_2$, and lastly the output layer neurons are $O_1$ and $O_2$. The weights, $W_9$, $W_{10}$, $W_{11}$, and $W_{12}$ are bias weights.

Here's how we calculate the total net input for $H_1$:

$$net_{H1} = W_1 \times I_1 + W_3 \times I_2 + W_9 \times 1$$

$$\therefore net_{H1} = 0.1 \times 0.1 + 0.1 \times 0.1 + 0.1 = 0.12000000000000001$$

We then squash it using the logistic function to get the output of $H_1$:

$$out_{H1} = \frac{1}{1 + e^{-0.12}} = 0.5299640517645717$$

Doing the same process, we get the results of:

$$net_{H2} = 0.13$$

$$out_{H2} = 0.5324543063873187$$

We repeat this process for the output layer neurons, using the output from the hidden layer neurons as inputs. Here is the output for $O_1$:

$$net_{O1} = W_5 \times out_{H1} + W_7 \times out_{H2} + W_{11} \times 1$$

$$\therefore net_{O1} = 0.1 \times 0.5299640517645717 + 0.1 \times 0.5324543063873187 + 0.1 \times 1$$

$$\therefore net_{O1} = 0.20624183581518907$$

Now use the sigmoid function to find the output of $O_1$:

$$out_{O1} = \frac{1}{1 + e^{-0.20624183581518907}} = 0.5513784696896066$$

We can now calculate the error for each output neuron using the squared error function and sum them to get the total error:

$$E_{Total} = \sum \frac{1}{2}(target - output)^2$$

If the target output for $O_1$ is 0 then:

$$E_{O1} = \frac{1}{2}(target_{O1} - out_{O1})^2 = \frac{1}{2}(0 - 0.5513784696896066)^2 = 0.10063063872901962$$

Repeating this process for $O_2$ gives:

$$net_{O2} = 0.25948726645392095$$

$$out_{O2} = 0.5645102463659317$$

$$E_{O2} = 0.15933590912606246$$

The total error for the neural network is the sum of these errors:

$$E_{Total} = E_{O1} + E_{O2} = 0.10063063872901962 + 0.15933590912606246$$

$$\therefore E_{Total} = 0.2599665478550821$$

## The Backwards Pass

Our goal with backpropagation is to update each of the weights in the network so that they cause the actual output to be closer the target output, thereby minimizing the error for each output neuron and the network as a whole.

### The Output Layer

Considering the weight $W_5$, we want to know how much a change in this weight effects the total error. By applying the chain rule, we can get:

$$\frac{\partial E_{Total}}{\partial W_5} = \frac{\partial E_{Total}}{\partial out_{O1}} \times \frac{\partial out_{O1}}{\partial net_{O1}} \times \frac{\partial net_{O1}}{\partial W_5}$$

First, how much does the total error change with respect to the output?

$$E_{Total} = \frac{1}{2}(target_{O1} - out_{O1})^2 + \frac{1}{2}(target_{O2} - out_{O2})^2$$

$$\frac{\partial E_{Total}}{\partial out_{O1}} = 2 \times \frac{1}{2}(target_{O1} - out_{O1})^{2-1} \times -1 + 0$$

$$\frac{\partial E_{Total}}{\partial out_{O1}} = -(target_{O1} - out_{O1}) = -(0 - 0.5513784696896066) = -0.55137846969$$

Next, how much does the output of $O_1$ change with respect to its total net input?

$$\frac{\partial out_{O1}}{\partial net_{O1}} = out_{O1}(1 - out_{O1}) = 0.5513784696896066(1 - 0.5513784696896066)$$

$$\therefore \frac{\partial out_{O1}}{\partial net_{O1}} = 0.24736025285$$

Finally, how much does the total net input of $O_1$ change with respect to $W_5$?

$$\frac{\partial net_{O1}}{\partial W_5} = out_{H1} = 0.5299640517645717$$

Putting it all together:

$$\frac{\partial E_{Total}}{\partial W_5} = \frac{\partial E_{Total}}{\partial out_{O1}} \times \frac{\partial out_{O1}}{\partial net_{O1}} \times \frac{\partial net_{O1}}{\partial W_5}$$

$$\therefore \frac{\partial E_{Total}}{\partial W_5} = -0.55137846969 \times 0.24736025285 \times 0.5299640517645717$$

$$\therefore \frac{\partial E_{Total}}{\partial W_5} = -0.05881071242497924$$

Doing this process for every single other weight in the output layer gives:

$$W_5^+ = -0.05881071242497924$$

$$W_6^+ = 0.07354751632357291$$

$$W_7^+ = -0.05908705880733426$$

$$W_8^+ = 0.07389310965562348$$

But we don't update them yet, not until we have done the full backpropagation with both inputs.

## Hidden Layer

Next, we'll continue the backwards pass by calculating new values for $W_1$, $W_2$, $W_3$, and $W_4$.

Big picture, here's what we need to figure out:

$$\frac{\partial E_{Total}}{\partial W_1} = \frac{\partial E_{Total}}{\partial out_{H1}} \times \frac{\partial out_{H1}}{\partial net_{H1}} \times \frac{\partial net_{H1}}{\partial W_1}$$

We're going to use a similar process as we did for the output layer, but slightly different to account for the fact that the output of each hidden layer neuron contributes to the output (and therefore error) of multiple output neurons. We know that $out_{H1}$ affects both $out_{O1}$and $out_{O2}$ therefore the $\frac{\partial E_{Total}}{\partial out_{H1}}$ needs to take into consideration its effect on the both output neurons:

$$\frac{\partial E_{Total}}{\partial out_{H1}} = \frac{\partial E_{O1}}{\partial out_{H1}} + \frac{\partial E_{O2}}{\partial out_{H1}}$$

Where:

$$\frac{\partial E_{O1}}{\partial out_{H1}} = \frac{\partial E_{O1}}{\partial net_{O1}} \times \frac{\partial net_{O1}}{\partial out_{H1}}$$

We can calculate this using values calculated earlier:

$$\frac{\partial E_{O1}}{\partial out_{H1}} = -0.55137846969 \times 0.24736025285 = -0.13638911767$$

$$\frac{\partial net_{O1}}{\partial out_{H1}} = W_5 = 0.1$$

Putting it all together:

$$\frac{\partial E_{O1}}{\partial out_{H1}} = -0.13638911767 \times 0.1 = -0.01363891176$$

Following the same process for $\frac{\partial E_{O2}}{\partial out_{H1}}$, we get:

$$\frac{\partial E_{O2}}{\partial out_{H1}} = -0.019423765$$

And hence:

$$\frac{\partial E_{Total}}{\partial out_{H1}} = -0.13638911767 - 0.019423765 = -0.15581288267$$

Following this process, we get the new weights for the hidden layer:

$$W_1^+ = -6.926827594435028e - 05$$

$$W_2^+ = 0.00041470910520664405$$

$$W_3^+ = -6.926827594435028e - 05$$

$$W_4^+ = 0.00041470910520664405$$

## Batch 2

Doing that whole process again on batch two yields the weights of:

$$W_8^+ = -0.05724138946701667$$

$$W_7^+ = 0.0729729546166579$$

$$W_6^+ = -0.05697509351449143$$

$$W_5^+ = 0.07263347294720225$$

$$W_4^+ = -0.0003860887328807712$$

$$W_3^+ = 0.00014642015745434638$$

$$W_2^+ = -0.0001930443664403856$$

$$W_1^+ = 7.321007872717319e - 05$$

After finally completing that, we get the new weight values as the sum of their batch values average over the batch size:

$$W_n^+ = \sum_{i=1}^{n} \frac{1}{2}(W_i^{batch1} + W_i^{batch2})$$

Completing the network, we get new weights of:

$$W_8^{new} = 0.19916741399056967$$

$$W_7^{new} = 0.09930570520953383$$

$$W_6^{new} = 0.09917137885954593$$

$$W_5^{new} = 0.09930886197388886$$

$$W_4^{new} = 0.0999985689813837$$

$$W_3^{new} = 0.09998921557833007$$

$$W_2^{new} = 0.19998891676306169$$

$$W_1^{new} = 0.09999287608226642$$