

## Backpropagation algorithm

Prepared by: Kishor Mishra(2019201038), Durgaprasad(20172139), Hitesh(2019201039)

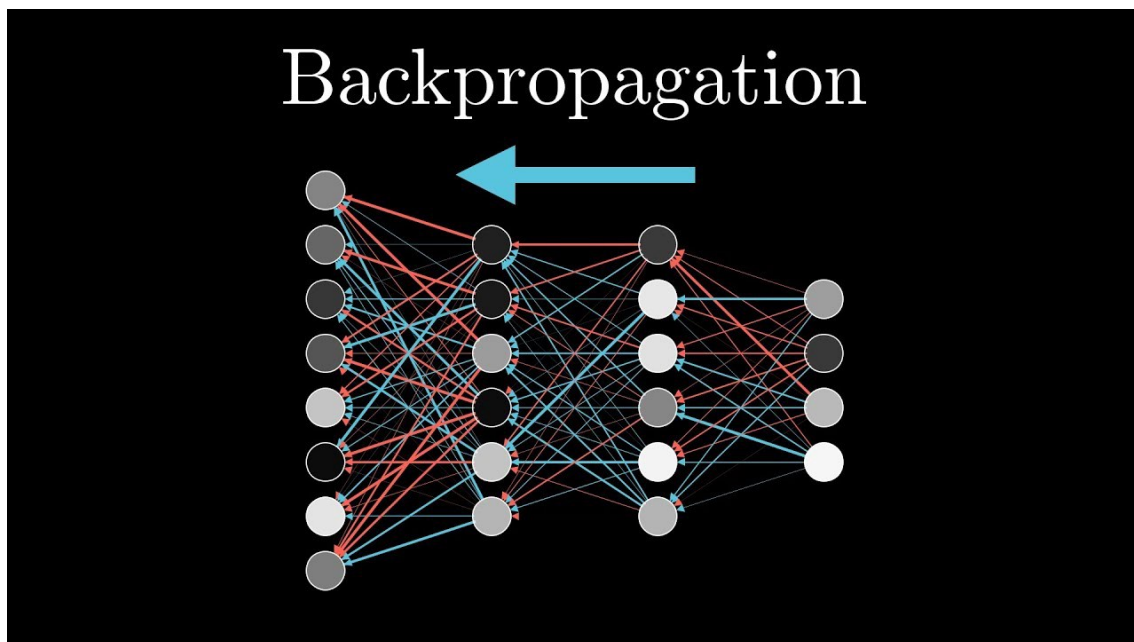
Backpropagation algorithm is probably the most fundamental building block in a neural network. The algorithm is used to effectively train a neural network through a method called chain rule. In simple terms, after each forward pass through a network, backpropagation performs a backward pass while adjusting the model's parameters (weights and biases).

According to the paper from 1989, backpropagation:

**repeatedly adjusts the weights of the connections in the network so as to minimize a measure of the difference between the actual output vector of the net and the desired output vector.**

and

**the ability to create useful new features distinguishes back-propagation from earlier, simpler methods.**

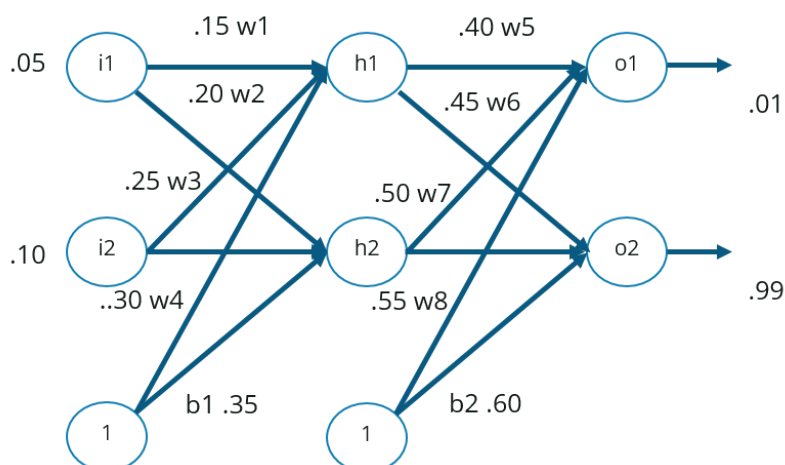


In other words, backpropagation aims to minimize the cost function by adjusting network weights and biases. The level of adjustment is determined by the gradients of the cost function with respect to those parameters.

Chain rule

$$\frac{\partial C}{\partial x} = \left[ \frac{\partial C}{\partial x_1}, \frac{\partial C}{\partial x_2}, \dots, \frac{\partial C}{\partial x_m} \right]$$

## How Backpropagation Works?



The above network contains the following:

Two inputs, Two hidden neurons, Two output neurons, Two biases.

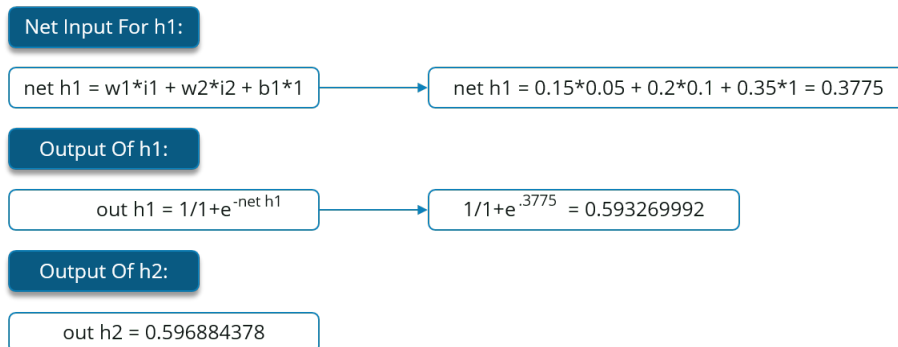
Below are the steps involved in Backpropagation:

Step 1: Forward Propagation

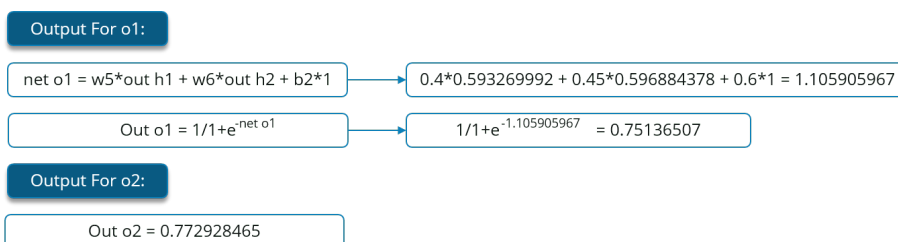
Step 2: Backward Propagation

Step 3: Putting all the values together and calculating the updated weight value

We will start by propagating forward.



We will repeat this process for the output layer neurons, using the output from the hidden layer neurons as inputs.



Now, lets see what is the value of the error:

**Error For o1:**

$$E_{o1} = \Sigma 1/2(\text{target} - \text{output})^2 \rightarrow \frac{1}{2} (0.01 - 0.75136507)^2 = 0.274811083$$

**Error For o2:**

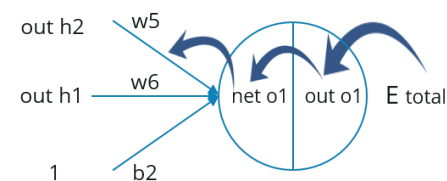
$$E_{o2} = 0.023560026$$

**Total Error:**

$$E_{\text{total}} = E_{o1} + E_{o2} \rightarrow 0.274811083 + 0.023560026 = 0.298371109$$

### Backward Propagation

Now, we will propagate backwards. This way we will try to reduce the error by changing the values of weights and biases. Consider W5, we will calculate the rate of change of error w.r.t change in weight W5.

$$\frac{\delta E_{\text{total}}}{\delta w_5} = \frac{\delta E_{\text{total}}}{\delta \text{out } o1} * \frac{\delta \text{out } o1}{\delta \text{net } o1} * \frac{\delta \text{net } o1}{\delta w_5}$$


Since we are propagating backwards, first thing we need to do is, calculate the change in total errors w.r.t the output O1 and O2.

$$E_{\text{total}} = 1/2(\text{target } o1 - \text{out } o1)^2 + 1/2(\text{target } o2 - \text{out } o2)^2$$

$$\frac{\delta E_{\text{total}}}{\delta \text{out } o1} = -(\text{target } o1 - \text{out } o1) = -(0.01 - 0.75136507) = 0.74136507$$

Now, we will propagate further backwards and calculate the change in output O1 w.r.t to its total net input.

$$\text{out } o1 = 1/1 + e^{-\text{net } o1}$$

$$\frac{\delta \text{out } o1}{\delta \text{net } o1} = \text{out } o1 (1 - \text{out } o1) = 0.75136507 (1 - 0.75136507) = 0.186815602$$

Lets see now how much does the total net input of O1 changes w.r.t W5?

$$\text{net } o1 = w_5 * \text{out } h1 + w_6 * \text{out } h2 + b_2 * 1$$

$$\frac{\delta \text{net } o1}{\delta w_5} = 1 * \text{out } h1 * w_5^{(1-1)} + 0 + 0 = 0.593269992$$

Now, lets put all the values together:

$$\frac{\delta E_{total}}{\delta w_5} = \frac{\delta E_{total}}{\delta out\ o1} * \frac{\delta out\ o1}{\delta net\ o1} * \frac{\delta net\ o1}{\delta w_5}$$

0.082167041

Lets calculate the updated value of W5:

$$w_5^+ = w_5 - \eta \frac{\delta E_{total}}{\delta w_5}$$

$$w_5^+ = 0.4 - 0.5 * 0.082167041$$

Updated w5

0.35891648

Similarly, we can calculate the other weight values as well.

After that we will again propagate forward and calculate the output. Again, we will calculate the error. If the error is minimum we will stop right there, else we will again propagate backwards and update the weight values.

This process will keep on repeating until error becomes minimum.

