Mathematics of a Neural Network

mariam.elantably

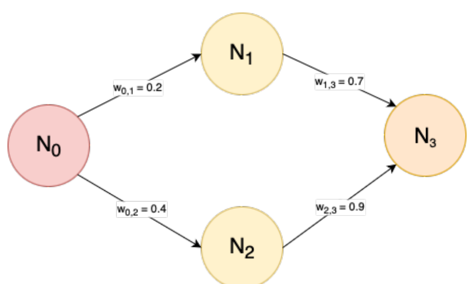
August 2025

1 Introduction

This document aims to demonstrate the mathematical run-through of a simple Neural Network, trained using back-propagation.

2 Network Architecture

The network consists of one hidden layer with two neurons and one neuron in the input and output layers.



The starting values for the weights are randomly generated. We will use N_3 to represent the output of the neural network, and N_3' to represent the target output. For simplicity, we will train the model on one set of input data, with $N_0 = 0.85$ and the expected value for $N_3' = 1.0$.

The activation function used will be the sigmoid function:

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

When the error of a value is found, the following formula will be used:

$$E(y, \hat{y}) = y - \hat{}y$$

where y is the target value and $\hat{y}$ is the calculated value.

3 Mathematical Calculation

3.1 Calculating Node Values

The first step is to calculate the values for each of the nodes in the hidden and output layers, using the following formula:

$$N_i = \sum_j w_{j,i} N_j$$

Using the above, we get the following values:

- $N_1 = 0.542397940774351$
- $N_2 = 0.5841905229354073$
- $N_3 = 0.7120681963873912$

3.2 Backpropagation

Backpropagation is the process that uses gradients to adjust the weight parameters and get an answer closer to the intended value. We begin by finding the error for the final output.

$$e = E(N_3', N_3) = 0.28793180361260884$$

We then use the following formula to compute the new values of the weights, working from the output layer to the input layers:

$$\Delta w_{i,j} = \eta \cdot \delta_j \cdot N_j$$

where η is the learning rate δ_j is the error term for each individual unit. For this calculation, we will use a learning rate of 1.

We find the error term for each of the units as such:

$$\delta_3 = N_3^{\cdot}(1 - N_3) \cdot e = 0.0590338169571488$$

$$\delta_2 = N_2 \cdot (1 - N_2) \cdot \delta_3 * w_{2,3} = 0.010256635139020864$$

$$\delta_1 = N_1 \cdot (1 - N_1) \cdot \delta_3 * w_{1,3} = 0.012906017944403162$$

We use these to update the weights, using the formula:

$$w_{i,j} = w_{i_j} + \eta \cdot \delta_j \cdot N_j$$

which give us the following results:

- $w_{0,1} = 0.20556317777867877$
- $w_{0,2} = 0.4075395733719547$
- $w_{1,3} = 0.7590338169571488$
- $w_{2,3} = 0.9590338169571488$

4 Repetition

The preceding section described one pass of backpropagation. The process is repeated numerous times, with the results detailed in the table below.

Run	w1	w2	w3	w4	Error
1	0.20556	0.40754	0.75903	0.95903	0.28793
2	0.21114	0.41497	0.81354	1.01354	0.27402
3	0.21669	0.42225	0.86403	1.06403	0.26146
4	0.22218	0.42937	0.91093	1.11093	0.25008
5	0.22757	0.4363	0.95463	1.15463	0.23975
6	0.23286	0.44304	0.99546	1.19546	0.23032
7	0.23804	0.44958	1.03371	1.23371	0.2217
8	0.2431	0.45593	1.06965	1.26965	0.21378
9	0.24804	0.4621	1.10348	1.30348	0.20649
10	0.25285	0.46808	1.13541	1.33541	0.19976
11	0.25754	0.47388	1.16562	1.36562	0.19352
12	0.26211	0.4795	1.19424	1.39424	0.18773
13	0.26657	0.48496	1.22143	1.42143	0.18233
14	0.2709	0.49026	1.24729	1.44729	0.1773
15	0.27513	0.49541	1.27193	1.47193	0.17259

This shows that with each iteration of backpropagation, the error tends to zero, and the process makes the output of the neural network more accurate.