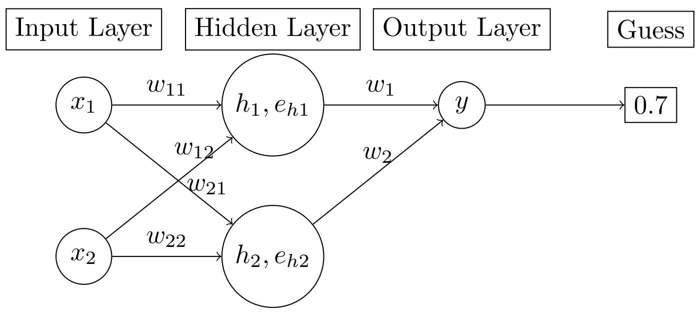
Backpropigation in Neural Networks

Jack Lukomski

August 17, 2023

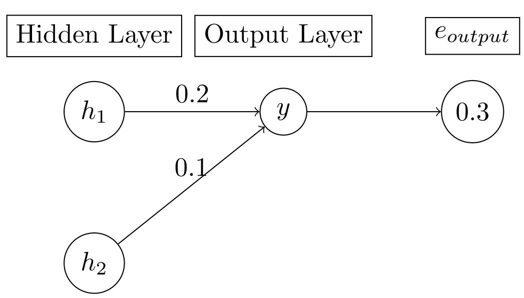
# 1 Introduction




known answer = 1
error = $1 - 0.7 = 0.3$
We can use the error to nudge a weight, $w_n$. But, how do we know what weight we need to nudge?

$e_{h1} =???$
$e_{h2} =???$
Knowing these two errors, we can then adjust the weights between the hidden layer and input layer. But how do we find those errors?

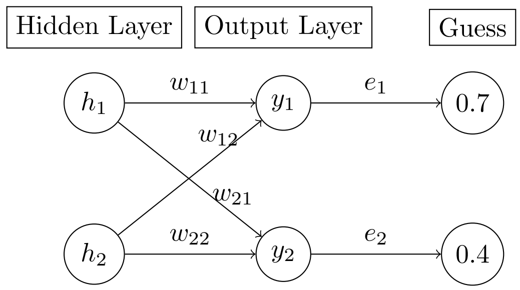
# 2 Breaking It Down



$$e_{h1} = \frac{w_1}{w_1 + w_2} e_{output} \tag{1}$$

$$e_{h2} = \frac{w_2}{w_1 + w_2} e_{output} \tag{2}$$

### 2.1 Adding Another Output




$$e_{h1} = \frac{w_{11}}{w_{11} + w_{12}} e_1 + \frac{w_{21}}{w_{21} + w_{22}} e_2 \tag{3}$$

$$e_{h2} = \frac{w_{12}}{w_{12} + w_{11}} e_1 + \frac{w_{22}}{w_{22} + w_{21}} e_2 \tag{4}$$

We don't necessarily need the bottom part of the fraction because it ends up all canceling out. So now we can write...

$$e_{h1} = w_{11}e_1 + w_{21}e_2 \tag{5}$$

$$e_{h2} = w_{12}e_1 + w_{22}e_2 \tag{6}$$

Now putting it into matrix terms...

$$\begin{bmatrix} e_{h1} \\ e_{h2} \end{bmatrix} = \begin{bmatrix} w_{11} & w_{21} \\ w_{12} & w_{22} \end{bmatrix} \begin{bmatrix} e_1 \\ e_2 \end{bmatrix} \tag{7}$$

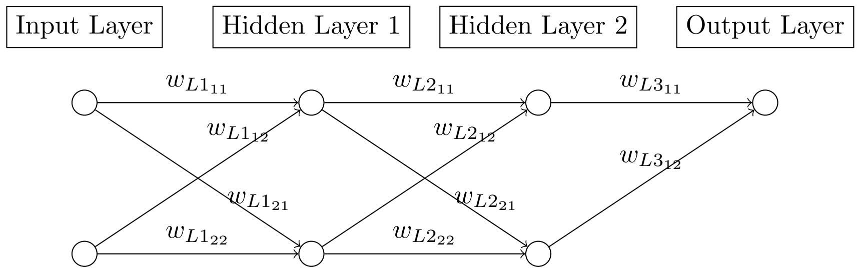
Oh and by the way...

$$w_l = \begin{bmatrix} w_{11} & w_{12} & \dots \\ w_{21} & w_{22} & \dots \\ \dots & \dots & \dots \end{bmatrix}^T \tag{8}$$

$$w_l^T = \begin{bmatrix} w_{11} & w_{12} & \dots \\ w_{21} & w_{22} & \dots \\ \dots & \dots & \dots \end{bmatrix}^T = \begin{bmatrix} w_{11} & w_{21} & \dots \\ w_{12} & w_{22} & \dots \\ \dots & \dots & \dots \end{bmatrix} \tag{9}$$

Which is the same as the first matrix in equation 7.

### 2.1.1 Example With XOR Gate




Calulating output error and hidden errors in backpropigation, where $e_o$ is the output error, $i$ reprsents the ith sample, $y$ reprsents the expected value, and $\hat{y}$ reprsents the actual value.

$$e_{o_i} = y_i - \hat{y}_i \tag{10}$$

$$e_{L2} = \begin{bmatrix} e_{o_i} \end{bmatrix} \begin{bmatrix} w_{L3_{11}} \\ w_{L3_{12}} \end{bmatrix} \tag{11}$$

# 3 Adjusting Weights

## 3.1 Gradient Descent

$$y = mx + b \tag{12}$$

$$\Delta m = learningRate * x * error_j \tag{13}$$

$$\Delta b = learningRate * error_j \tag{14}$$

How do we apply these equations to multivariable equations? In our case, we have...

$$y = \sigma(wI + b) \tag{15}$$

$$\sigma^{'}(x) = \sigma(x) + (1 - \sigma(x)) \tag{16}$$

$$\Delta w_{ij}^{HO} = \alpha E_L (Output(1 - Output)) H^T \tag{17}$$

$$\Delta w_{ij}^{IH} = \alpha H_e (H(1 - H)) I^T \tag{18}$$

Where $i$ and $j$ is the current weight in the matrix, and $HO$ sisgnifies the hidden layer to the output layer. $\alpha$ sisgnifies the learning rate, and $E_L$ is the error vector for the layer. $H$ is what is coming out of a hidden layer.