Gradient Calculations for XOR Multi-Layer Perceptron

1 Network Structure

The neural network for solving the XOR problem consists of:

- **Input Layer:** Two neurons x_1, x_2.
- **Hidden Layer:** Three neurons with sigmoid activation.
- **Output Layer:** One neuron with sigmoid activation.

2 Forward Propagation

The forward pass equations are given as follows:

2.1 Hidden Layer Computation

$$z_j^{(1)} = \sum_i w_{ji}^{(1)} x_i + b_j^{(1)} \tag{1}$$

$$h_j = \sigma(z_j^{(1)}) = \frac{1}{1 + e^{-z_j^{(1)}}} \tag{2}$$

2.2 Output Layer Computation

$$z^{(2)} = \sum_j w_j^{(2)} h_j + b^{(2)} \tag{3}$$

$$\hat{y} = \sigma(z^{(2)}) = \frac{1}{1 + e^{-z^{(2)}}} \tag{4}$$

3 Loss Function

The Mean Squared Error (MSE) is used as the loss function:

$$E = \frac{1}{2}(y - \hat{y})^2 \tag{5}$$

where y is the true label and $\hat{y}$ is the predicted output.

4 Backpropagation: Gradient Computation

4.1 Error Signal at Output Layer

The derivative of the error with respect to the output is:

$$\frac{\partial E}{\partial \hat{y}} = -(y - \hat{y}) \tag{6}$$

Since $\hat{y} = \sigma(z^{(2)})$, we use the derivative of the sigmoid function:

$$\frac{\partial \hat{y}}{\partial z^{(2)}} = \hat{y}(1 - \hat{y}) \tag{7}$$

Thus, the error signal at the output layer is:

$$\delta^{(2)} = \frac{\partial E}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial z^{(2)}} \tag{8}$$

$$= (\hat{y} - y) \cdot \hat{y}(1 - \hat{y}) \tag{9}$$

4.2 Gradients for Output Layer Weights

$$\frac{\partial E}{\partial w_j^{(2)}} = \delta^{(2)} \cdot h_j \tag{10}$$

$$\frac{\partial E}{\partial b^{(2)}} = \delta^{(2)} \tag{11}$$

4.3 Error Signal at Hidden Layer

Using the chain rule:

$$\delta_j^{(1)} = \left(\delta^{(2)} w_j^{(2)}\right) \cdot \sigma'(z_j^{(1)}) \tag{12}$$

Since $\sigma'(z_j^{(1)}) = h_j(1 - h_j)$, we get:

$$\delta_j^{(1)} = h_j(1 - h_j) \cdot w_j^{(2)} \cdot \delta^{(2)} \tag{13}$$

4.4 Gradients for Hidden Layer Weights

$$\frac{\partial E}{\partial w_{ji}^{(1)}} = \delta_j^{(1)} \cdot x_i \tag{14}$$

$$\frac{\partial E}{\partial b_j^{(1)}} = \delta_j^{(1)} \tag{15}$$

5 Gradient Descent Weight Updates

Once we have the gradients, we update the weights using gradient descent:

$$w^{(new)} = w^{(old)} - \eta \frac{\partial E}{\partial w} \tag{16}$$

$$b^{(new)} = b^{(old)} - \eta \frac{\partial E}{\partial b} \tag{17}$$

where η is the learning rate.

6 Summary of Computed Gradients

Parameter	**Gradient Formula**
$w_j^{(2)}$ (Output weights)	$\delta^{(2)} h_j$
$b^{(2)}$ (Output bias)	$\delta^{(2)}$
$w_{ji}^{(1)}$ (Hidden weights)	$\delta_j^{(1)} x_i$
$b_j^{(1)}$ (Hidden bias)	$\delta_j^{(1)}$

Table 1: Gradient calculations for backpropagation

7 Conclusion

This document details the backpropagation process for an XOR neural network with one hidden layer. The gradients were derived using the chain rule, and the weights were updated using gradient descent.