

Backpropagation Algorithm Derivation

Derivation Steps

1 Output Layer Error

$$\delta^{(L)} = \partial L / \partial z^{(L)} = \partial L / \partial a^{(L)} \cdot \sigma'(z^{(L)})$$

Error at output layer using chain rule

2 Hidden Layer Error

$$\delta^{(l)} = (W^{(l+1)})^T \delta^{(l+1)} \odot \sigma'(z^{(l)})$$

Propagate error backward through weights

3 Weight Gradient

$$\partial L / \partial W^{(l)} = \delta^{(l)} (a^{(l-1)})^T$$

Gradient w.r.t. weights at layer l

4 Bias Gradient

$$\partial L / \partial b^{(l)} = \delta^{(l)}$$

Forward & Backward Pass

Information Flow

Input
 $\mathbf{x}, \mathbf{W}, \mathbf{b}$

↓ **Forward**

Compute
 $\mathbf{z}, \mathbf{a}, \mathbf{L}$

↑ **Backward**

Compute
 $\partial L / \partial \mathbf{W}, \partial L / \partial \mathbf{b}$

↓ **Update**

Output
 \mathbf{W}', \mathbf{b}'

Key Formulas

Gradient w.r.t. biases at layer l

5 Parameter Update

$$W^{(l)} \leftarrow W^{(l)} - \eta \cdot \partial L / \partial W^{(l)}$$

Update parameters using learning rate η

Forward:

$$z^{(l)} = W^{(l)} a^{(l-1)} + b^{(l)}$$

Activation:

$$a^{(l)} = \sigma(z^{(l)})$$

Backward:

$$\delta^{(l)} = (W^{(l+1)})^T \delta^{(l+1)} \odot \sigma'(z^{(l)})$$

Backpropagation Algorithm

```
# Forward Pass
for l = 1 to L:
    z[l] = W[l] * a[l-1] + b[l]
    a[l] = activation(z[l])

# Compute Loss
L = loss_function(a[L], y)

# Backward Pass
delta[L] = dL/da[L] * activation_derivative(z[L])
for l = L-1 to 1:
    delta[l] = (W[l+1].T * delta[l+1]) * activation_derivative(z[l])
    dW[l] = delta[l] * a[l-1].T
    db[l] = delta[l]

# Update Parameters
for l = 1 to L:
    W[l] -= learning_rate * dW[l]
    b[l] -= learning_rate * db[l]
```



Key Insight: Backpropagation efficiently computes gradients by reusing intermediate values from the forward pass and applying the chain rule backward through layers. Each layer's gradient depends only on the next layer's gradient and local derivatives.



Numerical Example: Complete Backpropagation

Tiny Network (2-2-1) with Sigmoid Activation

Network Architecture

Input Layer



Hidden Layer



Output Layer



Input: $x = [0.5, 1.0]$

Target: $y = 1.0$

$W^{(1)} = \begin{bmatrix} 0.4 & 0.6 \\ 0.3 & 0.5 \end{bmatrix}$

$b^{(1)} = [0.1, 0.2]$

$W^{(2)} = \begin{bmatrix} 0.7 & 0.8 \end{bmatrix}$

$b^{(2)} = [0.3]$



Phase 1: Forward Propagation

Layer 1: Input \rightarrow Hidden

$$z^{(1)} = W^{(1)}x + b^{(1)}$$

$$z_1^{(1)} = 0.4(0.5) + 0.6(1.0) + 0.1 = 0.2 + 0.6 + 0.1 = \mathbf{0.9}$$

$$z_2^{(1)} = 0.3(0.5) + 0.5(1.0) + 0.2 = 0.15 + 0.5 + 0.2 = \mathbf{0.85}$$

$$\mathbf{z}^{(1)} = [0.9, 0.85]$$

$$\mathbf{a}^{(1)} = \sigma(\mathbf{z}^{(1)})$$

$$a_1^{(1)} = \sigma(0.9) = 1/(1+e^{-0.9}) \approx \mathbf{0.711}$$

$$a_2^{(1)} = \sigma(0.85) = 1/(1+e^{-0.85}) \approx \mathbf{0.701}$$

$$\mathbf{a}^{(1)} = [0.711, 0.701]$$

Layer 2: Hidden → Output

$$\mathbf{z}^{(2)} = \mathbf{w}^{(2)}\mathbf{a}^{(1)} + \mathbf{b}^{(2)}$$

$$z^{(2)} = 0.7(0.711) + 0.8(0.701) + 0.3$$

$$z^{(2)} = 0.498 + 0.561 + 0.3 = \mathbf{1.359}$$

$$\mathbf{z}^{(2)} = 1.359$$

$$\hat{y} = \sigma(\mathbf{z}^{(2)})$$

$$\hat{y} = \sigma(1.359) = 1/(1+e^{-1.359}) \approx \mathbf{0.796}$$

$$\hat{y} = 0.796$$

Loss Calculation (MSE)

$$L = \frac{1}{2}(\mathbf{y} - \hat{\mathbf{y}})^2$$

$$L = \frac{1}{2}(1.0 - 0.796)^2$$

$$L = \frac{1}{2}(0.204)^2 = \frac{1}{2}(0.0416) = \mathbf{0.0208}$$

$$\text{Loss} = 0.0208$$

← Phase 2: Backward Propagation (Computing Gradients)

Step 1: Output Layer Gradient

$$\partial L / \partial z^{(2)} = (\hat{y} - y) \cdot \sigma'(z^{(2)})$$

$$\sigma'(z^{(2)}) = \sigma(z^{(2)})(1 - \sigma(z^{(2)})) = 0.796(1 - 0.796) = 0.796 \times 0.204 = \mathbf{0.162}$$

$$\partial L / \partial z^{(2)} = (0.796 - 1.0) \times 0.162 = -0.204 \times 0.162 = \mathbf{-0.033}$$

$$\delta^{(2)} = -0.033$$

Step 2: Layer 2 Weight & Bias Gradients

$$\partial L / \partial w^{(2)} = \delta^{(2)} \cdot (a^{(1)})^T$$

$$\partial L / \partial w_1^{(2)} = -0.033 \times 0.711 = \mathbf{-0.023}$$

$$\partial L / \partial w_2^{(2)} = -0.033 \times 0.701 = \mathbf{-0.023}$$

$$\partial L / \partial w^{(2)} = [-0.023, -0.023]$$

$$\partial L / \partial b^{(2)} = \delta^{(2)}$$

$$\partial L / \partial b^{(2)} = -0.033$$

Step 3: Hidden Layer Gradient

$$\partial L / \partial \mathbf{z}^{(1)} = (\mathbf{W}^{(2)})^T \cdot \boldsymbol{\delta}^{(2)} \cdot \sigma'(\mathbf{z}^{(1)})$$

$$\sigma'(z_1^{(1)}) = 0.711(1 - 0.711) = \mathbf{0.205}$$

$$\sigma'(z_2^{(1)}) = 0.701(1 - 0.701) = \mathbf{0.210}$$

$$\partial L / \partial z_1^{(1)} = 0.7 \times (-0.033) \times 0.205 = \mathbf{-0.0047}$$

$$\partial L / \partial z_2^{(1)} = 0.8 \times (-0.033) \times 0.210 = \mathbf{-0.0055}$$

$$\boldsymbol{\delta}^{(1)} = [-0.0047, -0.0055]$$

Step 4: Layer 1 Weight & Bias Gradients

$$\partial L / \partial \mathbf{W}^{(1)} = \boldsymbol{\delta}^{(1)} \cdot \mathbf{x}^T$$

$$\partial L / \partial w_{11}^{(1)} = -0.0047 \times 0.5 = \mathbf{-0.0024}$$

$$\partial L / \partial w_{12}^{(1)} = -0.0047 \times 1.0 = \mathbf{-0.0047}$$

$$\partial L / \partial w_{21}^{(1)} = -0.0055 \times 0.5 = \mathbf{-0.0028}$$

$$\partial L / \partial w_{22}^{(1)} = -0.0055 \times 1.0 = \mathbf{-0.0055}$$

$$\partial L / \partial \mathbf{W}^{(1)} = [[-0.0024, -0.0047], [-0.0028, -0.0055]]$$

$$\partial L / \partial \mathbf{b}^{(1)} = \boldsymbol{\delta}^{(1)}$$

$$\partial L / \partial \mathbf{b}^{(1)} = [-0.0047, -0.0055]$$

✓ All Computed Gradients

$$\frac{\partial L}{\partial W^{(2)}} \\ [-0.023, -0.023]$$

$$\frac{\partial L}{\partial b^{(2)}} \\ -0.033$$

$$\frac{\partial L}{\partial W^{(1)}} \text{ (row 1)} \\ [-0.0024, -0.0047]$$

$$\frac{\partial L}{\partial W^{(1)}} \text{ (row 2)} \\ [-0.0028, -0.0055]$$

$$\frac{\partial L}{\partial b^{(1)}} \\ [-0.0047, -0.0055]$$

Parameter Update (Learning Rate $\alpha = 0.5$)

$$W^{(2)}_{\text{new}} = [0.7, 0.8] - 0.5 \times [-0.023, -0.023] = [0.7115, 0.8115]$$

$$b^{(2)}_{\text{new}} = 0.3 - 0.5 \times (-0.033) = 0.3165$$

$$W^{(1)}_{\text{new}} = [[0.4012, 0.6024], [0.3014, 0.5028]]$$

$$b^{(1)}_{\text{new}} = [0.1024, 0.2028]$$