Deep Learning Lab Session
Forward Pass
& Backward Pass (Backpropagation)

First Year Artificial Intelligence and Data Science

February 12, 2026

Learning Objectives

By the end of this lab, students will be able to:

- Understand forward propagation mathematically
- Understand backward propagation conceptually and mathematically
- Compute forward pass manually
- Compute gradients using chain rule
- Implement forward and backward passes in code
- Train a neural network from scratch

Part 1 — Conceptual Recall

- Neuron model
- Linear transformation
- Activation function
- Loss function
- Parameters (weights and bias)

Network structure:

$$\text{Input} \rightarrow \text{Hidden} \rightarrow \text{Output}$$

Part 2 — Mathematical Forward Pass

Network Architecture

2–2–1 MLP

Given Values

Input:

$$X = [x_1, x_2] = [1, 0]$$

Hidden layer:

$$W^{(1)} = \begin{bmatrix} 0.5 & -0.3 \\ 0.8 & 0.2 \end{bmatrix}, \quad b^{(1)} = [0.1, -0.1]$$

Output layer:

$$W^{(2)} = [0.7, -0.5], \quad b^{(2)} = 0.2$$

Activation: Sigmoid

Task 1 — Forward Computation

$$Z^{(1)} = W^{(1)}X + b^{(1)}$$
$$A^{(1)} = \sigma(Z^{(1)})$$
$$Z^{(2)} = W^{(2)}A^{(1)} + b^{(2)}$$
$$\hat{y} = \sigma(Z^{(2)})$$

Part 3 — Loss Computation

Target:

$$y = 1$$

Binary Cross Entropy:

$$L = -[y \log(\hat{y}) + (1 - y) \log(1 - \hat{y})]$$

Part 4 — Backward Pass

Gradient flow:

$$\frac{\partial L}{\partial \hat{y}}, \quad \frac{\partial \hat{y}}{\partial Z^{(2)}}, \quad \frac{\partial Z^{(2)}}{\partial W^{(2)}}, \frac{\partial Z^{(2)}}{\partial b^{(2)}}$$
$$\frac{\partial Z^{(2)}}{\partial A^{(1)}}, \quad \frac{\partial A^{(1)}}{\partial Z^{(1)}},$$
$$\frac{\partial Z^{(1)}}{\partial W^{(1)}}, \frac{\partial Z^{(1)}}{\partial b^{(1)}}$$

Manual Backpropagation

$$\delta^{(2)} = \hat{y} - y$$
$$\frac{\partial L}{\partial W^{(2)}} = \delta^{(2)} A^{(1)}$$
$$\frac{\partial L}{\partial b^{(2)}} = \delta^{(2)}$$
$$\delta^{(1)} = (W^{(2)T} \delta^{(2)}) \odot \sigma'(Z^{(1)})$$
$$\frac{\partial L}{\partial W^{(1)}} = \delta^{(1)} X^T$$
$$\frac{\partial L}{\partial b^{(1)}} = \delta^{(1)}$$

Part 5 — Coding Implementation

Students must implement:

- forward()
- loss()
- backward()
- update()

Training loop:

1. Forward pass
2. Loss computation
3. Backward pass
4. Parameters update
5. Repeat

Dataset (XOR):

```
X = [(0,0),(0,1),(1,0),(1,1)]
y = [0,1,1,0]
```

Part 6 — Gradient Checking

Finite differences:

$$\frac{\partial L}{\partial w} \approx \frac{L(w+\epsilon) - L(w-\epsilon)}{2\epsilon}$$

Part 7 — Analysis Questions

1. Why is chain rule essential?
2. What happens if gradients vanish?
3. Role of activation derivatives?
4. Effect of initialization?
5. Why deep networks are hard to train?