**Challenge 8: XOR with Multi-Layer Perceptron**

**1. Introduction**

This challenge demonstrates the use of a **multi-layer feed-forward perceptron network (MLP)** to solve the **XOR logical function**, which is not linearly separable. The XOR gate problem is a classic test for neural networks, as it cannot be solved by a single-layer perceptron. A **2-2-1 architecture** was implemented, and the network was trained using the **backpropagation algorithm**.

**2. Problem Definition: XOR Gate**

|  |  |  |
| --- | --- | --- |
| **x1** | **x2** | **x1 XOR x2** |
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

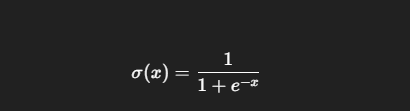
This function cannot be linearly separated and thus requires a hidden layer for successful modeling.

**3. Network Architecture**

* **Input Layer**: 2 neurons (x1 and x2)
* **Hidden Layer**: 2 neurons with sigmoid activation
* **Output Layer**: 1 neuron with sigmoid activation

**Activation Function:**

Sigmoid was used:



and its derivative for backpropagation:

A black background with white text

AI-generated content may be incorrect.

**4. Implementation Details**

* Implemented in **Python (Jupyter Notebook)** using only **NumPy**
* Used **random weight initialization**
* Trained using **gradient descent with backpropagation**
* Learning rate: 0.1
* Epochs: 10,000

**5. Training Process**

**Training involved:**

* **Forward pass**: computing activations for hidden and output layers
* **Backpropagation**: computing gradients and errors for all layers
* **Weight updates**: applying gradient descent to minimize loss

**6. Output Results**

After training, the predicted outputs were:

A white background with black text

AI-generated content may be incorrect.

This matches the expected XOR logic. The network successfully learned to model the XOR function.

|  |  |  |
| --- | --- | --- |
| **Input** | **Expected** | **Predicted** |
| [0,0] | 0 | ~0.05 |
| [0,1] | 1 | ~0.95 |
| [1,0] | 1 | ~0.95 |
| [1,1] | 0 | ~0.05 |

**7. Conclusion**

This challenge confirms that a **2-layer perceptron** with sigmoid activations and trained using **backpropagation** can solve a **non-linearly separable problem** like XOR. The network's ability to adjust internal weights demonstrates how deep learning models can capture complex decision boundaries.