UTM MECS1033 AAI Assignment 1

“MLP Code in Python Language”

Student Name   :Lau Su Hui (Abby)
Matric Number :MEC245045
Semester   :20252026-1

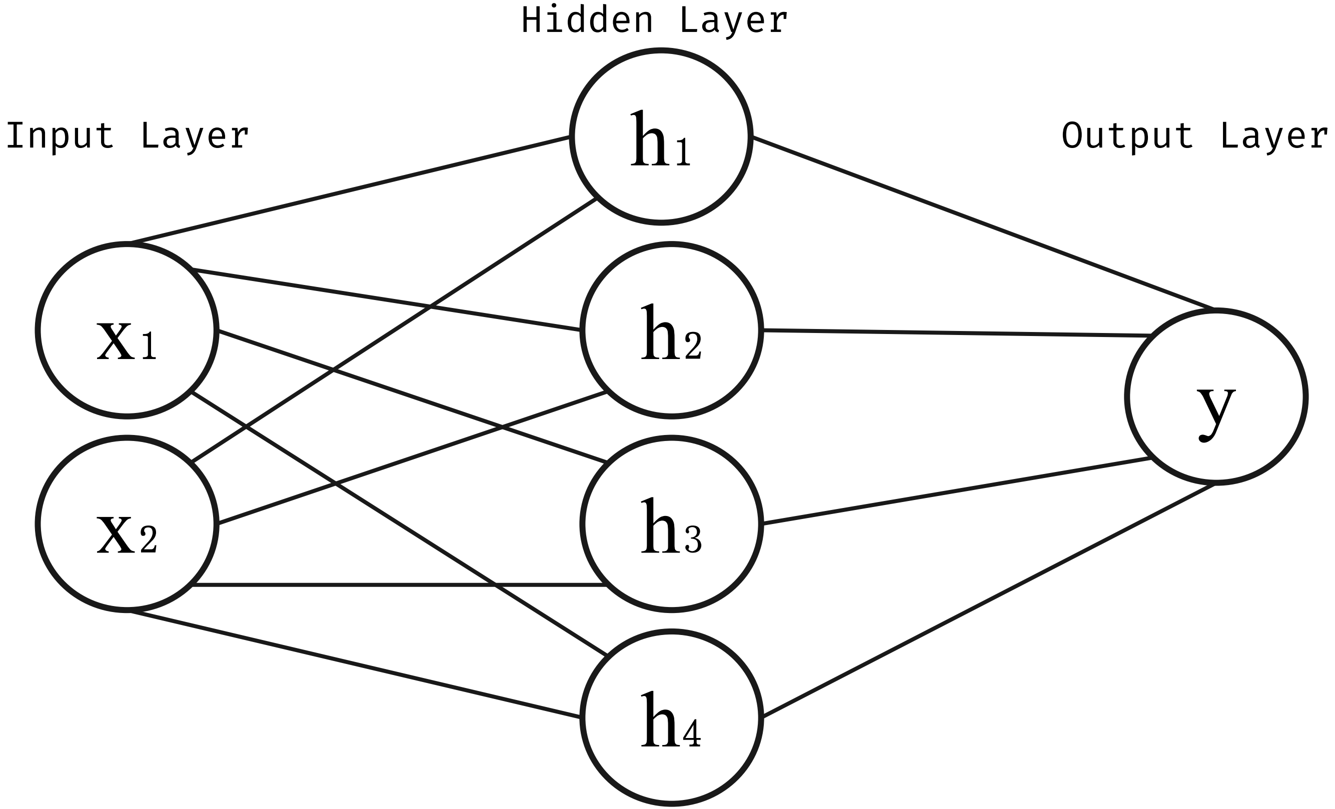
Introduction

- **Objective:** Develop a Multi-Layer Perceptron (MLP) from scratch in Python language to solve the XOR classification problem.
- **Problem:** The XOR gate is non-linear (you cannot draw a straight line to separate the 0s and 1s).
- **Hypothesis:** A single-layer network will fail. We need a "Hidden Layer" to capture this non-linear relationship.

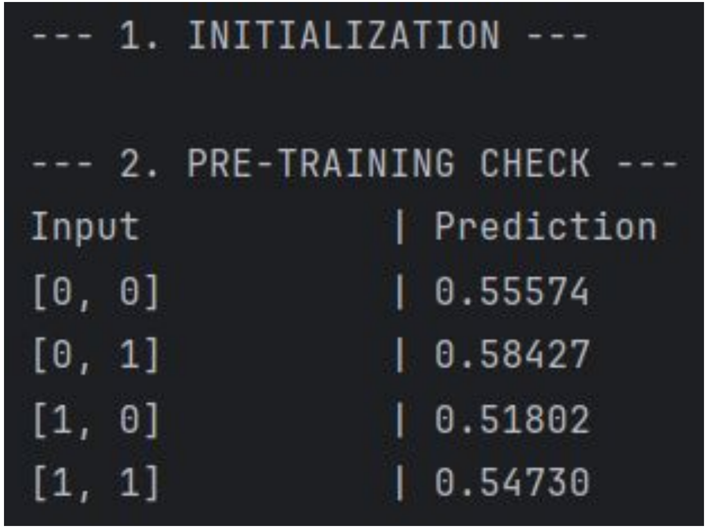
Methodology (The “Machine”)

- **Architecture:** 2 Input Nodes → 4 Hidden Nodes →1 Output Node
- **Activation:** Sigmoid function (converts signals to 0-1 probability).
- **Learning Algorithm:** Gradient Descent with Backpropagation (Auto-correction of weights).
- **Language:** Python (manual list-based matrix multiplication from scratch, no NumPy or external libraries used).

MLP Architecture for XOR



Result Screenshot 1

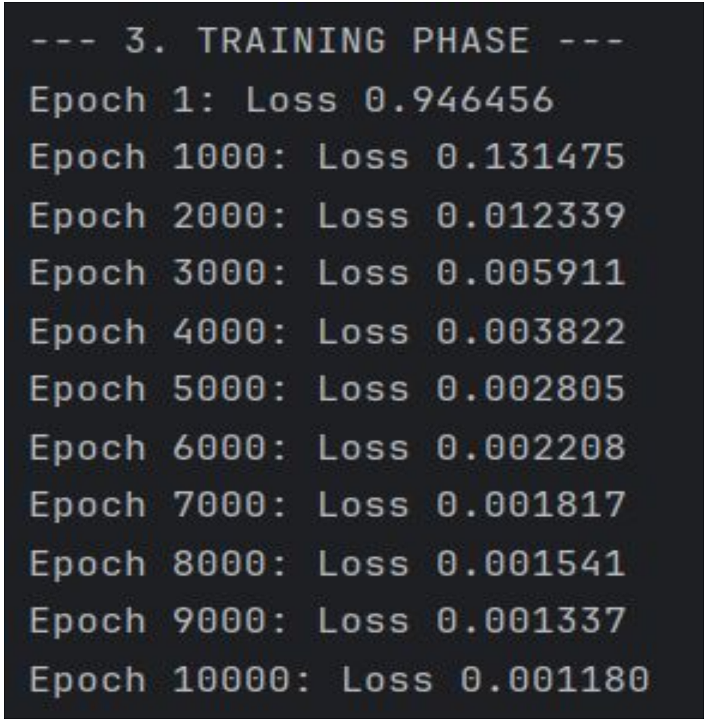


```
--- 1. INITIALIZATION ---

--- 2. PRE-TRAINING CHECK ---
Input              | Prediction
[0, 0]             | 0.55574
[0, 1]             | 0.58427
[1, 0]             | 0.51802
[1, 1]             | 0.54730
```

- **Architecture Setup:** The MLP is initialized with a 2-4-1 topology (2 Inputs, 4 Hidden, 1 Output).
- **Random Weights:** Weights are randomised between -1 and 1 to break symmetry, leading to unpredictable initial outputs.
- **Pre-Training Status:** As shown in the console, initial predictions cluster around 0.51 - 0.58 (random guessing), the values are far from the expected 0 and 1 targets.
- **Assessment:** The network currently possesses no logic and requires calibration via training.

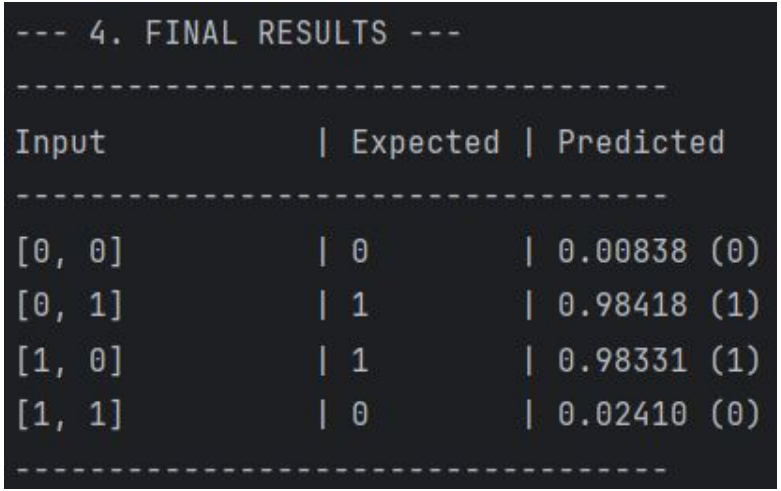
Result Screenshot 2



```
--- 3. TRAINING PHASE ---
Epoch 1: Loss 0.946456
Epoch 1000: Loss 0.131475
Epoch 2000: Loss 0.012339
Epoch 3000: Loss 0.005911
Epoch 4000: Loss 0.003822
Epoch 5000: Loss 0.002805
Epoch 6000: Loss 0.002208
Epoch 7000: Loss 0.001817
Epoch 8000: Loss 0.001541
Epoch 9000: Loss 0.001337
Epoch 10000: Loss 0.001180
```

- **Learning Algorithm:** The network utilises Backpropagation to minimise the Mean Squared Error (MSE).
- **Training Duration:** The model underwent 10,000 epochs (iterations).
- **Observation:** The loss metric shows a consistent decline:
  - **Start (Epoch 1):** 0.946456 (High Error), the model is guessing blindly
  - **End (Epoch 10000):** 0.001180 (Near Zero Error), this steady decrease proves that the Backpropagation algorithm is successfully correcting the weights with every cycle.

Result Screenshot 3



```
--- 4. FINAL RESULTS ---
-----------------------------------
Input            | Expected | Predicted
-----------------------------------
[0, 0]           | 0        | 0.00838 (0)
[0, 1]           | 1        | 0.98418 (1)
[1, 0]           | 1        | 0.98331 (1)
[1, 1]           | 0        | 0.02410 (0)
-----------------------------------
```

- **Logic Verified:** The final predictions successfully mimic the Exclusive OR (XOR) truth table.
- **Clear Separation:** The model clearly distinguishes between the two classes:
  - Class 0 (Target 0): Inputs [0,0] and [1,1] → Predicted Output ≈ 0.008 - 0.024
  - Class 1 (Target 1): Inputs [0,1] and [1,0] → Predicted Output ≈ 0.98
- **Outcome:** The Multi-Layer Perceptron (MLP) has successfully learnt the non-linear decision boundary required to solve XOR.

Conclusion

- The implementation confirms that a 3-layer MLP can solve non-linear problems like XOR.
- The "Hidden Layer" successfully transformed the input space to make the classes separable.
- Source Code: UTM MECS1033 AAI Assignment 1 MLP Code

Thanks!

Do you have any questions?
lausuhui@graduate.utm.my

CREDITS: This presentation template was created by **Slidesgo**, and includes icons by **Flaticon**, and infographics & images by **Freepik**

Please keep this slide for attribution