STINK 3014 (NEURAL NETWORKS)
A251 – Assignment (#2) (5 %)

Instructor: Azizi Ab Aziz, *PhD*
Submission date: 27 / Nov / 2025 (before 11.59 pm) via UUM Learning Portal

"*You are not meant for crawling, so don't. You have wings. Learn to use them and fly.*" [Rumi]

PRACTICAL IMPLEMENTATION (EXPERIMENTS)

In this assignment, you will work with a simple Python implementation (file: *Chap03-MLP-CalculationExample-Baseline*) of a feedforward neural network trained using backpropagation with momentum. You will explore how neural networks learn, analyze weight changes, perform experiments, test modifications, and apply the model to simple tasks. To answer these questions, you will use the provided code, extend it, and report your findings.

PART I – Theoretical Concepts
Questions:

1. Describe the architecture of the neural network used in the code. How many inputs, hidden units, and output units are there?
2. Why is a sigmoid function used as the activation function in both the hidden and output layers?
3. Explain how the error is computed in each epoch.
4. What is the purpose of storing weight updates (e.g., delta_w1_prev, etc.)?
5. How does momentum help prevent the network from getting stuck in local minima?

PART II - Experiments
Questions:

1. Modify the learning rate (α) from 0, 0.1, 0.5, 0.7, 0.9. Compare convergence speed and stability.
2. Change the momentum value (β) to 0, 0.1, 0.5, and 0.99. Compare learning smoothness.
3. Test the model with different initial weights (e.g., w1 = 0.1, w2 = –0.3, w3 = 0.8).
   a) What differences do you observe?
   b) Compare the error convergence with three different initial weight sets.
4. Run experiments with 10, 50, 100, 500, and 2000 epochs. How does error reduction differ?
5. Analyse the effect of weight updates: which weight changes most during training and why?
6. Replace the target = 1 with target = 0. What changes occur in the training dynamics?
7. Add noise to inputs (e.g., X1 = 1.1, X2 = 0.95). Does learning remain stable?
8. Describe how this simple neural network demonstrates the core principles of deep learning

Policy:
All grading of deliverables will be based on the standards indicated for each deliverable. Deliverables may not be turned in late, and no cheating! For this class, cheating will include: plagiarism (using the writings of another without proper citation), copying of another (either current or past student's work), working with another on individually assigned work, or in any other way presenting as one's work that which is not entirely one's work. The occurrence of plagiarism will result in removal from the course with a failing grade.