# Project 4 - MNIST Digit Recognition Using Multi-Layer Neural Networks (Due 10/29 - Early Deadline; Extended Deadline: 10/30)

## Objectives:

The objective of this project is to have a thorough understanding of the traditional multi-layer perceptron and backpropagation.

## Data set used:

MNIST and XOR

## Requirements:

- (20 pts) Task 1: Go through Nielsen's book on Neural Network and Deep Learning. Ideally, by the end of the semester, you should finish reading at least Chapters 1 through 3. For this project, you should read through Chapter 1 and to answer some of the questions, you also need to selectively read through Chapter 3. Answer the following question in the report after reading. The answer to each question should take less than a third of a page, assuming single space and a font size of 12 Times.
    - (4) Explain the differences between batch processing and online processing
    - (4) Explain the differences between gradient descent and stochastic gradient descent. What is mini-batch processing?
    - (4) Explain the differences between perceptron and sigmoid neurons. What's the advantage of a sigmoid neuron as compared to perceptron?
    - (4) Explain the differences between feedforward vs. backpropagation. What is back propagated and what is forwarded?
    - (4) Why do we need to introduce "bias" when training a neural network?
- (80 pts) Task 2: Download the code set from GitHub, debug, and make it run locally
    - (20) Task 2.1: Nielsen was able to use a simple structure to train a network to recognize the handwritten digits (MNIST). Use the same setup as he used and plot the convergence curve using MNIST. Note that the convergence curve is a plot of classification accuracy (or error) vs. epoch. Nielsen's code has already output the accuracy at the end of each epoch. You just need to plot it on Figure 1.
    - (30) Task 2.2: Play around the hyperparameter setups and draw three figures showing the effect of changing each hyperparameter to the convergence curve.
        - (10) Effect of network structure (Figure 2). On the same figure, show at least five convergence paths using Nielsen's baseline, different number of hidden layers and different number of hidden nodes in each layer. Keep learning rate at 1, mini-batch size at 15, but extend the number of epochs to 100 (or whichever constant you and your computer have patience with). Provide comments on why different performance is observed if any at all. Identify the one structure that has the best performance.
        - (10) Effect of mini-batch size (Figure 3). Use the network structure with the best performance and modify mini-batch size. Show at least 4 convergence paths with different mini-batch sizes. Comment on the different performances obtained using the different mini-batch size. Identify the size with the best performance.
        - (10) Effect of learning rate (Figure 4). Use the network structure and the mini-batch size with the best performance and change learning rate from 1, 0.1, 0.01, to 0.001. Show the convergence curves. Comment on the different performances obtained using the different learning rate. Identify the rate with the best performance.
    - (30) Task 2.3: Use the network library to implement the XOR gate. You have experienced in HW4 that single-layer perceptron cannot realize XOR since the decision boundary is not linear in the 2-d

space. Going through the practices you had in Task 2.2 and show again three figures on the effects of different hyperparameters. Fix mini-batch size at 1. Specifically,

  - (5) Effect of network structure (Figure 5)
  - (5) Effect of learning rate (Figure 6)
  - (10) Effect of random initialization vs. initializaiton with symmetry in the weight (Figure 7). See details in Slide 22 of Lecture 12.
  - (10) You probably spent more time on XOR than on MNIST. Comment on why it is more difficult to train a smaller network than a larger one. From your experience gained in this project, provide a procedure (when to adjust what based on what and the order of doing so) you'd recommend people to follow when using BPNN for classification purpose. Report the configuration of network and provide discussion regarding failure cases and successful cases.

- (+10) Bonus: Implement "Momentum" for faster and more stable convergence.