Assignment 6: Neural Networks by Mimi Trinh

Section 1: Summary and Problem Definition

The purpose of this project is to develop a classification model using machine learning techniques for optical character recognition that can detect written numbers in the MNIST dataset with 70,000 observations. Specifically, the goal is to build a neural network system to be able to determine all ten digits in contrast to previous examples of binary classifier that can only detect yes/no outcome of a single number. The entire MNIST dataset is used in this project. This is a computer vision problem (classification of images analogous to the MNIST dataset) in which the predictive accuracy of models must be weighed against the costs of model development and implementation.

Section 2: Research Design, Measurement, and Statistical Methods

In this project, we fit a neural network to the MNIST dataset while testing alternative network structures and hyperparameter settings. Specifically, we will examine the processing time and accuracy on both the training and testing datasets of the following structures.

Model Number	Number of Layers	Nodes per Layer
0	2	10
1	2	20
2	5	10
3	5	20

We will use the multi-layer perceptron benchmark experiment as the research design for this project. Tested neural network structures should be explored within a benchmark experiment, a factorial design with at least two levels on each of two experimental factors (at least a 2x2 completely crossed design). But due to the time required to fit each neural network, we will observe only one trial for each cell in the design. Also, we will be using a simple training-and-test split with the split having been performed previously. That is, we use the training and test sets that come with MNIST. Unlike previous projects with only training and test sets, in this project, we split the dataset into three: training, validation, and test sets. However, the accuracy measurement of the neural network model only shows the scores for training and testing sets.

Section 3: Programming Work

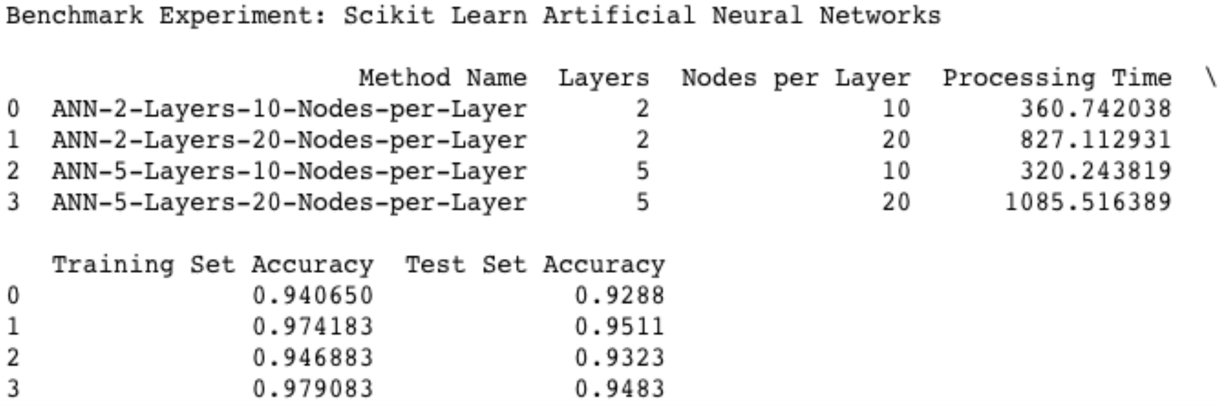
There are two popular Python packages that can be utilized for this project: TensorFlow and Scikit-Learn. We first use TensorFlow to access the MNIST dataset, conduct exploratory data analysis, and prepare the data. Then we use Scikit-Learn to develop the neural network system using four different scenarios with different number of layers and number of nodes per layer. It's best practice to standardize the data, so we scale the data as part of the pre-processing before developing the classification model.

Images are analyzed by using pixels. In this case, pixel values range from 0 to 255 with 0 indicating a white background and 255 indicating a black foreground. We convert the uint8 image to 32 bit floats and rescale the values to be centered around 0, between [-0.5, 0.5].

Section 4: Results and Recommendations

The results in the exhibit show that in terms of processing time, model #2 with 5 layers and 10 nodes is the best since it takes the least amount of time. Model #3 is the best with train set accuracy, and model # 1 is the best with test set accuracy. The train set accuracy for model #1 is also very good, so if we only consider accuracy measurement, model #1 is the winner. Though model #1 takes much longer processing time than models #0 and 2, because of its higher accuracy measurement, we recommend this model #1 with 2 layers and 20 nodes as the most trustworthy neural network from the benchmark study to the financial institution to apply machine learning on optical character recognition.

Exhibit



```
Benchmark Experiment: Scikit Learn Artificial Neural Networks

                       Method Name  Layers  Nodes per Layer  Processing Time  \
0  ANN-2-Layers-10-Nodes-per-Layer       2               10       360.742038   
1  ANN-2-Layers-20-Nodes-per-Layer       2               20       827.112931   
2  ANN-5-Layers-10-Nodes-per-Layer       5               10       320.243819   
3  ANN-5-Layers-20-Nodes-per-Layer       5               20      1085.516389   

   Training Set Accuracy  Test Set Accuracy  
0               0.940650             0.9288  
1               0.974183             0.9511  
2               0.946883             0.9323  
3               0.979083             0.9483  
```