COMP3046 Course Project – Part III

Design and Implementation of an Artificial Neural Network (ANN)

Project Requirement:

- Implement an ANN class as described in Pages 23-30 of Neural_Network_Training.pptx. The ANN should work as a general multi-class classifier, not just for the provided MNIST data set. A user should be able to set the number of layers and the number of neurons in each layer. You only need to consider the following sigmoid activation function:

$$f(x) = \frac{1}{1 + e^{-x}}$$

  The loss function is defined in Page 28:

$$C = \frac{1}{2}\|y - a^L\|^2 = \frac{1}{2}\sum_j (y_j - a_j^L)^2$$

- The ANN class should support the following functions:
  - feedforward function: to carry out the layer-wise feedforward calculations (Page 25)
  - train function: to train the model on a given data set. Please use mini-batch SGD and backprogagation algorithm for the training (Page 30). You should output the total loss value at the end of each epoch. Notice that a user should be able to specify the following training parameters: learning rate, mini-batch size, and stop condition (such as number of epochs).
  - inference function: given an input vector x, predict its corresponding digit.
  - weight store/load functions: to store or load the weight values to or from local disk file.

- A testing program to test your ANN class:
  - demo the train function using the training data file *train.txt*
  - since the training takes a long time to converge, you should pre-train your ANN model and save the weights. During the demonstration, you should load the weights from the saved weight file and then demo the inference function using the test data file *test.txt*, and show the overall prediction accuracy

Data files (unzip *data.zip* first):

- *train.txt*: it contains 40,000 samples. Each sample begins with the label, which is a digit between 0 and 9, followed by 784 integers whose values are between 0 and 255. The 784 integers represent the 28x28 pixels of a greyscale picture of handwritten digit.

- *train_small.txt*: it contains only 256 samples. Since *traing.txt* is very large, it may take a long time to load it into memory. You can use the small file *train_small.txt* for debugging purpose.

- *test.text*: it contains 2,000 testing samples with the same format as train.txt. Use this file to test the prediction accuracy of your trained model.

Submission (through Moodle):

- A brief document that describes (1) the design of your ANN class; (2) your experimental results (such as loss value vs. number of training epochs, and CPU running time); (3) the contribution of each team member (from 0 to 100%), including the design of ANN class, the implementation, the debugging, the experiments, the documentation, and any other workload.
- Source codes in a zip file
- Deadline: 23:59PM, 21 April 2019

Marking scheme (individual assessment based on each member's contribution):

- Program completeness: 30%
- Program correctness: 30%
- OO design: 10%
- CPU performance (the shorter running time per epoch, the better CPU performance you achieve): 10%
- Programming style: 10%
- Document: 10%