10910COM526000 Deep Learning

Homework 1: Neural Networks

Announced: **October 13 at 01:00 pm**
Deadline: **November 2 at 11:59 pm (iLMS)**
TA: 蘇翁台 {wentai@2008@hotmail.com}
成家聲 {ck1010089@gmail.com}
陳國軒 {ning364@yahoo.com.tw}

Introduction:

There is a saying that goes “Great oaks from little acorns grow.” In this assignment, you need to build a shallow neural network as well as the mini-batch SGD training process on MNIST dataset from scratch using Python3. The performance of your model should surpass 95% accuracy on testing data.

Rules:

- Built-in machine learning libraries (Sklearn, PyTorch, TensorFlow…) are not allowed to use for NN, you need to use basic mathematical operations in NumPy to define the behavior of each layer.
- You can use any libraries to read the data from the MNIST folder and to do result visualization which should be shown on your report.
- Please properly comment your code to let us understand your train of thought.
- Discussions are encouraged, but plagiarism is strictly prohibited and punishable!

Submission:

You should compress your code, dataset, report (hw1report_studentID.pdf) and readme.txt (explain how to run your code) into a ZIP file (hw1_studentID.zip), and submit it on iLMS before the due date. Zero credit for the submission after deadline.

Implementation [80%, 10% for each part]

1. *MNIST reading & splitting:* (http://yann.lecun.com/exdb/mnist/ for details)
   You need to split the original training data into validation data and training data. (proportion: 3/7)
2. *Dense neural layer:*
   Every output neuron has full connection to the input neurons.
3. *ReLU layer:*
   We add nonlinear activation functions after the neural layer using ReLU.
4. *Softmax output:*
   The final layer is typically a Softmax function which outputs the probability of a sample being in different classes.
5. *Cross-entropy loss calculation:*
   Use the output of a batch of data and their labels to calculate the CE loss.
6. *Backward propagation:*
   Propagate the error backwards and update each parameter.
7. *Validation:*
   Test the performance on validation data during each epoch.
8. *Testing accuracy:*
   Reach 95% to get 10 points, 90~95% -> 7 points, 85~90% -> 5 points, zero otherwise.

Report [20%]

The format is not limited, but the following matters must be discussed in your report:

1. Show your model architecture and testing accuracy.
2. How do you implement feed forward and backward propagation? A brief explanation is fine.
3. Plot training loss and validation loss. (loss vs. epochs figure)
4. If we use a very deep NN with a large number of neurons, will the accuracy increase? Why or why not?
5. Why do we need to validate our model?
6. t-SNE results (optional, not included in 20%)

Bonus [10%]

Try to use t-SNE (read the document before you call function) to visualize the outputs of (1) validation data at different epochs and (2) testing data, then show the result on your report.