Computer Vision HW #2 (due 1 May 2020)

In this assignment, your task is to implement a multi-layers Neural Network and evaluate its performance in classifying handwritten digits. To get started with the assignment, you will need to download a machine learning framework, **PyTorch** or **TensorFlow**. You may download them from following links:

- https://pytorch.org/get-started/locally/
- https://www.tensorflow.org/install.

You can also use **Google Colaboratory** without installing those packages.

(Colab: https://colab.research.google.com/notebooks/intro.ipynb)

The installation method depends on your preferences (If there is no GPU, do not choose CUDA). A sample code is provided to solve the 3-layers NN problem that was covered in the last class. It uses PyTorch, so I recommend using PyTorch.

File Included

- *dataloader.py*: Python scrip for preparing dataset before train and test process.
- *main.py*: Python scrip for running 3-layers Neural Network.
- *data.csv, gt.csv, test.csv, test_gt.csv*: dataset that was covered in the last class. *data.csv* contains two-dimensional dataset of 10000 examples and *gt.csv* contains label of each data in *data.csv*. *test.csv* and *test_gt.csv* is test dataset of 3000 examples.
- *mnist.mat*: original dataset from MNIST (http://yann.lecun.com/exdb/mnist/). MNIST handwritten dataset consists of a training set of 60000 images and test set of 10000 images. The images are 28 x 28 pixels in grayscale, and the categories correspond to 10 digits: 0, 1, …, 9. This will be used in your multi-layers NN.

If you run the script *main.py* with all the contents included in the same directory, you can see the training process working well.

Your Tasks

- Build a Neural Network with 3 or more layers. In order to classify the MNIST images into 10 classes (each corresponds to 10 digits), an output layer must be 10 nodes.
- Train the network for 30 epochs, using batches of 10 training examples, a learning rate=0.01. You can adjust the number of epochs, batches, and learning rate.
- Plot the training error, testing error, training accuracy, and testing accuracy.
- Write a short report with the resulting graph.

In the sample code, the Neural Network consists of three fully-connected layers. Fully-connected layer is represented as “nn.Linear” in the *main.py.* In your NN, change those layers into 2d convolution layers (use “nn.Conv2d”). Each convolution layers may be connected to ReLU function (“F.relu”) except the last layer. You can add layers freely. If the training process takes a very long time, you can use only a subset of the given MNIST data.

You can check the implementation details of a convolution layer and some tutorials of a simple classification task:

PyTorch

- https://pytorch.org/docs/stable/nn.html
- https://pytorch.org/tutorials/beginner/blitz/cifar10_tutorial.html

TensorFlow

- https://www.tensorflow.org/api_docs/python/tf/keras/layers/Conv2D
- https://www.tensorflow.org/tutorials/keras/classification

Submission

Submit your work by 12:00 PM on 1 May 2020 to teaching@mcl.korea.ac.kr

You are required to submit a single zip file: *name_studentID.zip*

Your zip file must contain one file and one folder

- Report file in pdf format. Please indicate your name and ID on the top of the report
- Folder with your code and dataset.