CMSC 636 Neural Nets and Deep Learning
Spring 2019, Instructor: Dr. Milos Manic, *http://www.people.vcu.edu/~mmanic*
Homework 3

Homework No. 3
Due Thursday, Apr. 4, 2019, noon

Deliverables:

- Once you have completed sections from above, submit to blackboard for grading.
- Sign an email with your name and student ID.
- Provide your name in header/footer of every page of assignment.
- **Report**: compile your answers into a **single** file only (answers to 3.1, 3.2, and 3.3 and 3.4 if you are submitting extra credit).
- **Code**: attach Jupyter notebooks for 3.2. and 3.3. as a separate in .ipynb and pdf document.
- Use the following convention for naming the file: "HWnn_Family_name.xxx", nn being the homework number.
- This assignment is worth 15 points. Please note that missing of any of the deliverables above will be penalized.

3.0 Intro to TensorFlow

Complete the previously posted *Intro to TensorFlow* exercise.

3.1 TensorFlow (3 pts)

Please answer the following questions:

- What is a computation graph?
- What are Variables in TensorFlow?
- What are Placeholders in TensorFlow?

Report: Answers to questions.

3.2 Image classification using Multilayer Perceptron (6 pts)

Train a Multilayer Perceptron Neural Network to classify numbers on the MNIST dataset http://yann.lecun.com/exdb/mnist/.
In the attached script "mlp_mnist.html", you will find a starter code that downloads the MNIST dataset and loads it into this Python script.
Please complete this script (where indicated) to train a multilayer perceptron with one hidden layer of 100 units, using ReLU activation function.

The classification accuracy on both training and testing dataset should be above 90%.

Note: do NOT use the tf.contrib.learn, tf.layers or tf.keras libraries.
Deliverables:

- Jupyter notebook including the generated output (following *Deliverables* directions above).
- Report on the accuracy on training and testing datasets (as part of the single pdf report, following *Deliverables* directions above).

3.3 Image classification using Convolutional Neural Networks (6 pts)

Train a Convolutional Neural Network (CNN) on the MNIST dataset. In the script "convnet_mnist.html" you will find a starter code that downloads the MNIST dataset and loads it into this Python script. For this problem, you should do the following:

1. Update the script to setup the following architecture:
    - First convolutional layer
        - 32 filters, each one of size 5x5
        - ReLU activation function (Rectifier Linear Unit)
        - Max pooling layer of size 2x2, and stride of 2 in both x and y.
    - Second convolutional layer
        - 64 filters, each one of size 5x5
        - ReLU activation function
        - Max pooling layer of size 2x2, and stride of 2 in both x and y.
    - Reshape layer
        - Is responsible for transforming of 2D filtered maps to 1D vector, which is the input for the fully connected layer,
    - Fully connected layer
        - Linear layer with 256 units (fully connected layer),
        - ReLU activation function.
    - Softmax layer
        - Linear layer that maps the 256 units from the previous layer to the 10 output units (0 to 9).
        - LogSoftMax activation function.
2. What is the size in each dimension for the inputs and outputs of each layer?
3. Train the CNN with the MNIST dataset
    - Report errors on training and test datasets.

    Hint: Before going into larger number of epochs, test the model on single epoch. Correct errors, if any.
4. Compare the performance of the multilayer perceptron with the performance of the convolutional neural network.
    - Accuracy, training time (you can simply use your own watch).

Note: do NOT use the tf.contrib.learn, tf.layers or tf.keras libraries.
Deliverables:

- Jupyter notebook including the generated output (following *Deliverables* directions above).
- Report answers to questions 2, 3, and 4. Comment and discuss. Please do not include the code in this file. Include this as part of the single pdf report, following *Deliverables* directions above.

3.4 Extra credit (3 pts): Modify the script "mlp_mnist.ipy" for training of a multilayer perceptron network with two hidden layers, each with 100 units, using ReLU activation function.

- Does the training error improve?
- What about the testing error?
- Try using sigmoid activation function. Discuss the results.

Deliverables:

- Jupyter notebook including the generated output (following *Deliverables* directions above).
- Report answers to questions. Comment and discuss. Please do not include the code in this file. Include this as part of the single pdf report, following *Deliverables* directions above.