CMSC 636 Neural Nets and Deep Learning
Spring 2021, Instructor: Dr. Milos Manic, *http://www.people.vcu.edu/~mmanic*

Homework No. 4 - Extra Credit and Exam Preparation!

If desiring credit, please submit by April 27. Can be done individually or as a team.

Student certification:

Team member 1:

Print Name: ________________________ *Date:* ____________

*I have contributed by doing the following:*____

Signed: ________________________________ *(you can sign/scan or use e-signature)*

Team member 2:

Print Name: ________________________ *Date:* ____________

*I have contributed by doing the following:*____

Signed: ________________________________ *(you can sign/scan or use e-signature)*

Team member 3:

Print Name: ________________________ *Date:* ____________

*I have contributed by doing the following:*____

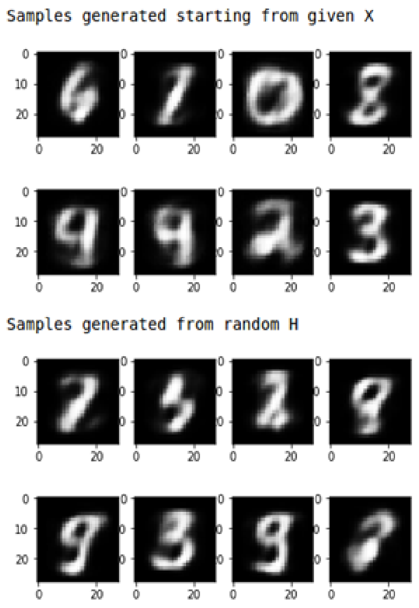
Signed: ________________________________ *(you can sign/scan or use e-signature)*

4.1 Training of RMBs (7 pts)

Train a Restricted Boltzmann Machine (RBM) on the MNIST dataset http://yann.lecun.com/exdb/mnist/. In the attached script "hw4_rbm.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 an RBM with 20 hidden units.

For visualizing the progress of the training procedure, we generate samples using: 1) Gibbs sampling starting from a given set of samples (X) extracted from the training dataset. 2) Gibbs sampling starting from a set of random samples (H) on the hidden layer.

For 20 hidden units, the generated images should look like this:



Experiment for different values for number of hidden units, learning rate (α), k steps, etc:

1. Try to find the architecture that produces the most realistic images of numbers
2. What happens if you use more than 100 units in a hidden layer?

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

Deliverables:

- **Code:**
    - **Jupyter notebook including the generated output (.ipynb and html).**
- **Report:**
    - **Report the architecture, parameters used, copy of images generated for the results you consider the best.**
    - **Answers and discussion to the questions above.**

4.2. Deep Belief Networks (8 pts)

A Deep Belief Network (DBN) is a set of stacked RBMs trained in a greedy layer-wise approach. Based on your work on the problem 4.1, complete the script "hw4_dbn.html" to train a two-layer DBN on the MNIST dataset.

The two-layer DBN is composed of two RBMs, where the hidden layer h_1 of the first RBM constitutes the visible layer of the second RBM.

As in 4.1, the progress of the training procedure is visualized using: 1) Gibbs sampling starting from a given set of samples (X) extracted from the training dataset. 2) Gibbs sampling starting from a set of random samples on the first and second hidden layers (h_1 while training the first RBM and h_2 while training the second RBM).

Experiment for different values of hidden units, learning rate (α), number of steps k, and maximum number of iterations:

1. Try to find the architecture that produces the most realistic images of numbers
2. What happens if you use more than 100 units in the first hidden layer?

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

Deliverables:

- **Code:**
    - **Jupyter notebook including the generated output (.ipynb and html).**
- **Report:**
    - **Report the architecture, parameters used, copy of images generated for the results you consider the best.**
    - **Answers and discussion to the questions above.**