COL 865: Deep Learning.
Assignment 1

Due: Sunday October 15, 11:50 pm. Weightage: 10%

Notes:

- You should submit all your code as well as any graphs that you might plot (see below).
- Include a **single write-up (pdf) file** which includes a brief description for each question explaining what you did. Include any observations and/or plots required by the question in this single write-up file.
- You can use PyTorch. If you would like to use any other language, please check with us before you start.
- Your code should have appropriate documentation for readability.
- You will be graded based on what you submit as well as your ability to explain your code.
- This assignment should be done individually.
- Some questions have been borrowed from existing online resources. You may be able to find some online existing implementations, we will include most of these in moss run. Please refrain using those directly and report any references if you refer them.
- You should carry out all the implementation by yourself.
- We plan to run Moss on the submissions. Any cheating will result in a 0 on the assignment. Additional penalties will be incurreed depending on the scale of cheating (going all the way up to a penalty of **-10**). More serious offenses will be referred to the Department's internal committee for disciplinary actions.

1 Alex Net

In this problem, your are required to code the Alex Net as described in paper here. We will be using a subset of original IMAGENET dataset consisting of 35 sub-classes. Data is attached in the following link. Specifically, you should code and report the following things:

- **Main Task**: Train the Alex Net with 5 Convolution Layers with 3 fully connected layers. Report your train time, training error plot, test time for an example and final test error.
- **RELU versus tanhs**: Plot the training error versus number of epochs for both non-linearities.
- **Drop Out versus No Drop Out**: Report the accuracy you obtain with and without drop out. You should use $p = 0.5$ for drop-outs.
- **Overlapping Pooling versus Non-Overlapping Pooling**: Study the pooling discussed in Section 3.4 in the paper. The paper claims that overlapping pooling works better than Non-Overlapping pooling slightly. Implement pooling for $s = 2, z = 2$ and $s = 2, z = 3$. Report the accuracy and explain the possible reasons for it
- **Optimization Techniques:** Compare a)Stochastic Gradient Descent (SGD), b) SGD with momentum, c)ADAM and d) method described in paper with each other. Plot the graph of training error versus number of epochs.

- **Bonus:** Suggest and implement your own new ideas and/or study some of the subsequent works which build on Alex Net work and test their effectiveness experimentally. Feel free to report any negative results as well which intuitively should have worked but didn't work
- **Best Model Submission:** You are required to submit your best trained model (this can be model generated in bonus part). We will be running your trained model on totally new unseen data on these classes. We will provide a script which takes as input your trained model and report accuracy on unseen new data on these classes. Check out Piazza for more details in a couple few days.

2 Recurrent Neural Networks

In this problem, you are required to code up a recurrent neural network for the problem of Part-Of-Speech (POS) tagging. We are providing a training dataset with 40000 sentences, with each word having a corresponding POS tag. Additionally, we provide a validation dataset with 4000 sentences. This data is part of CoNLL-2012 dataset. There are 42 possible POS tags. The dataset and required files for the assignment can be downloaded from here

- **GRU cell:** Implement the operation of a GRU cell. Your function should take in the input x and previous hidden state h_t and return the next hidden state of the network. We are providing a file *gru.py* with a custom GRU class. You should fill in the code in this file.
- **Cell comparison:** Train the POS tagger using the LSTMCell, GRUCell and RNNCell provided by pytorch as well using your own GRU cell. Plot the training error for all four variants vs number of epochs. Report your training time, training error plot, validation time and validation error.
- **Best Model Submission:** You are required to submit your best trained model. We will be running your trained model on new unseen data for these tags. We will provide a script which takes as input your trained model and report accuracy on unseen new data on these classes. Check out Piazza for more details in a couple few days.