# ECE 685D HW5

Deliverability: Please upload your Jupyter Notebook file (.ipynb file) with all outputs and necessary derivations exported as a PDF or HTML to Sakai. All necessary coding documentation can be found in `https://pytorch.org/docs/stable/index.html`. **We highly encourage you to submit your derivation in LaTeX. If you choose to submit a hand-written derivation, please make sure your derivation is legible. If you have hand-written solutions, please scan/take a photo and insert them into their designated space.** You can follow this tutorial to insert images to .ipynb files: `https://www.geeksforgeeks.org/insert-image-in-a-jupyter-notebook/`

## 1 Problem 1: GAN (30 pts)

In this question, you are asked to implement a Deep Convolutional GAN (DCGAN) for generating MNIST images. Please use the provided template to write the GAN model. The architecture of DCGAN for both generator and discriminator is depicted in Figure 1.

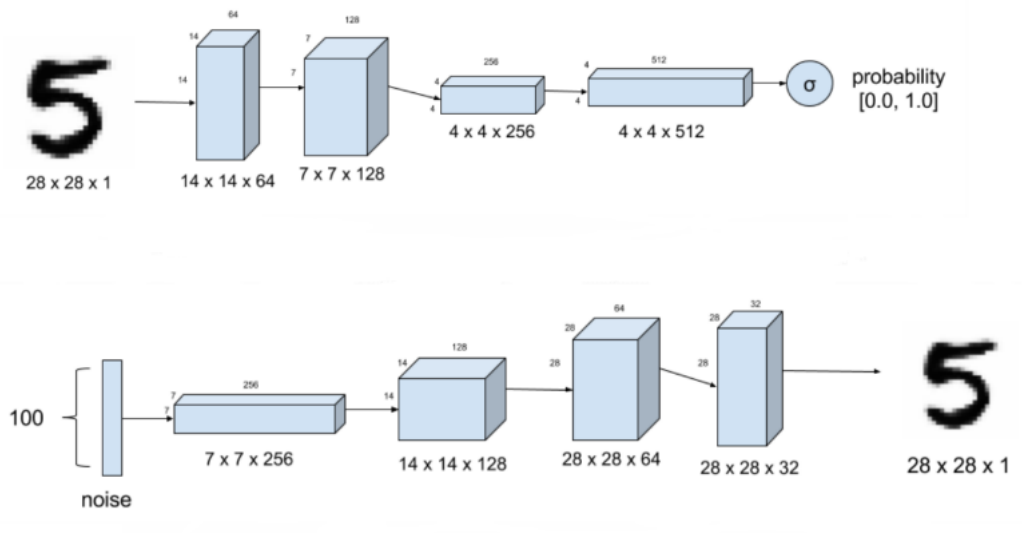


Figure 1: Discriminator and Generator Architecture.

Visualize the generated images from the trained GAN and also plot the discriminator and generator losses. You are free to use Dropout or/and Batch-norm in each layer of Generator and Discriminator. Also, you can select the activation function at the last layer of the Generator depending on how you normalize the data (either tanh or sigmoid).

## 2 Problem 2: n-gram with LSTM (40 pts)

In this problem, you will implement an LSTM model to predict the next word of a sentence. This is similar to implementing N-gram models, which are statistical models that use a sequence of words to predict the next word based on the probability of that N-word sequence being present in the corpus. For example, if N = 2, the model is called a bi-gram. For a sentence: "This is my dog", the bi-gram pairs looks like {'This is', 'is my', 'my dog'}. In a general sense, N-gram is a probabilistic model where for an n-word sequence $[w_1, w_2, ..., w_n]$ with the joint probability:

$$P(w_1, ..., w_n) = P(w_1)P(w_2|w_1)P(w_3|w_2, w_1)...P(w_n|w_{n-1}, ..., w_1)$$

For this question, you are provided with a template. You will modify the code in the template and implement an LSTM Neural Network to predict the next word of a sentence using the previous 5 words (which is effectively a 6-gram model). Train your LSTM Neural Network on the 'austen-sense.txt' in the Gutenberg Corpus Dataset in NLTK. Generate a 6-gram dataset where the first 5 words of the 6-gram would be your input and the 6th word is your output. You can use any tokenizer of your choice to tokenize the dataset. Use a categorical cross-entropy along with Adam optimizer to train your model on the dataset (you do not have to split your dataset).

After that, you will generate a 100-word text using your model which starts with 'his natural shyness was overcome' (You can achieve this by successively predicting the 6th word from the previous 5 words in a loop).

# 3 Problem 3: Recurrent model for human activity prediction (bonus 30 pts)

You will perform human activity prediction based on a sequence of data collected from a phone accelerometer. First, download data from **HERE** and format the dataset. You may adapt from the Jupyter Notebook **HERE** for data cleaning.

Once you cleaned your data, perform a 75-25 split of your dataset for training and testing. Now, design an RNN with one or multiple LSTM or GRU layers for human activity prediction. Report the prediction accuracy of each type of activity and interpret your results.