# EE 228 HW#4 - MNIST with Dropout and Noise

This exercise will focus on training a neural network classifier for the MNIST dataset. The goal of this exercise is understanding the effect of overparameterization and dropout on the training performance and test accuracy. **Format the input data** in a similar fashion on HW2. However, unlike HW2, this HW will work with multiclass classification (all 10 classes). To speed things up, you are allowed to train with 10,000 examples (1,000 example from each class) rather than the full dataset with 50,000 examples.

**Shallow Neural Net Classifier:** Training will be done on PyTorch or TensorFlow. For this exercise, we will use a shallow fully-connected neural network width a single hidden layer. Use He initialization, ReLU activation, and use cross-entropy loss. The network output has the form

$$f(\boldsymbol{x}) = \text{softmax}(\boldsymbol{V}\text{ReLU}(\boldsymbol{W}\boldsymbol{x})).$$

Here $\boldsymbol{W} \in \mathbb{R}^{k\times 784}$ and $\boldsymbol{V} \in \mathbb{R}^{10\times k}$ where $k$ is the number of hidden units.

## Assignment

In this exercise, we will play with two variables which is the network width $k$ and dropout rate $p$. Your tasks are as follows.

1. (3 pts) Setup your code so that you can run multiple MNIST models for varying choices of $k$ and $p$ automatically, Specifically, you need two for loops (one for $k$ and one for $p$) and within the loop, you call PyTorch/TensorFlow.

2. (7 pts) Pick the width grid $\mathcal{K} = [1, 5, 10, 20, 40]$ and dropout grid $\mathcal{P} = [0.1, 0.5, 1.0]$. Run MNIST models over these grids with Adam optimizer for 80 epochs. Store the test/train accuracy and loss.

   - Fix $p = 1.0$ which is the case of "no dropout regularization". Plot the test and training accuracy as a function of $k$. As $k$ increases, does the performance improve? At what $k$, training accuracy becomes 100%?
   - Plot the training accuracy as a function of $k$ and for different $p \in \mathcal{P}$ on the same plot. What is the role of $p$ on training accuracy? When $p$ is smaller, is it easier to optimize or more difficult? For each choice of $p$, determine at what choice of $k$, training accuracy becomes 100%.
   - Plot the test accuracy as a function of $k$ and for different $p \in \mathcal{P}$ on the same plot. Does dropout help with the test accuracy? For which $(k, p)$ configuration do you achieve the best test accuracy?

3. (7 pts) We will spice up the problem by adding some noise to labels. Pick 40% of the training examples at random. Assign their labels **at random to another value** from 0 to 9. For instance, if the original image is 0 and its label is 0, then you will assign its label to a number from 1 to 9 at random. Thus 60% of the training examples remain correct and 40% will have incorrect labels. Repeat the previous step with this noisy dataset.

4. (3 pts) Comment on the differences between Step 2 and Step 3. How does noise change things? For which setup dropout is more useful?