# CS484/684 Computational Vision
# PART II of Assignment 3

Submit your write-up in pdf and all source code in a zip file (with proper documentation). Write a script for each programming exercise so that the TAs can easily run and verify your results. Make sure your code runs!
**[Text in square brackets are hints that can be ignored.]**

In this part you are going to run some experiments involving CNNs. You can use Jupiter/python notebook and install the following libraries: Keras, TensorFlow and all their dependencies. To perform any required image transformation, you can use libraries like *skimage* or *pillow*. You can find detailed instructions and tutorials for each of these libraries on the respective websites. [To install, try *pip install keras* and *pip install pillow*. For TensorFlow, follow the installation steps on its webpage.]

For all experiments, running on CPU is sufficient. You do not need to run the code on GPUs. Before start, we suggest you review what we learned about each layer in CNN, and read at least this tutorial.

1. Train a VGG11 net on the MNIST dataset. VGG11 was an earlier version of VGG16 and can be found as model A in Table 1 of this paper [Simonyan&Zisserman, ICLR'15], where Section 2.1 also gives you all the details about each layer. The goal is to get as close to 0 loss as possible. Note that our input dimension is different from the VGG paper. You need to resize each image in MNIST from its original size 28 x 28 to 32 x 32 [why?], and it might be necessary to change at least one other layer of VGG11. [This experiment will take up to 1 hour on a CPU, so please be cautious of your time. If this running time is not bearable, you may cut the training set by 1/10, so only have ~600 images per class instead of the regular ~6000.]

2. Once you've done the above, inspect the training process. Create the following plots:
   a) (5 pts) test accuracy vs the number of iterations
   b) (5 pts) training accuracy vs the number of iterations
   c) (5 pts) test loss vs the number of iterations
   d) (5 pts) training loss vs the number of iterations

3. Then, it is time to inspect the generalization properties of your final model. Rotate and blur the test set images using any python library of your choice:
   e) (10 pts) test accuracy vs the degree of rotation. Try the following rotations: -45, -40 ... 40, 45, and plot the test accuracy at each rotation. What is the effect?
   f) (10 pts) test accuracy vs Gaussian blurring. Try blurs with radius 0, 1, 2, 3, 4, 5, 6 and plot the test accuracy vs the blur for each rotation. What is the effect?

4. Lastly, let us verify the effect of regularization.
   g) (10 pts) Re-do parts 1 - 3 above with $l_2$ regularization on the weights. Create the same plots and indicate the regularization constant that you use.
   h) (10 pts) Re-do parts 1 - 3 above with data augmentation. Create the same plots and explain what kind of data augmentation you use.