# EL-GY-9133 Machine Learning for Cyber-Security

# Lab 2: Adversarial Attacks on Deep Neural Networks

*Release Date*: 10/29/2020; *Due Date*: Midnight, 11/20/2020

## Overview

In this lab, you will investigate adversarial perturbation attacks on Deep Neural Networks using the MNIST digits dataset as a benchmark. You will then evaluate adversarial retraining as a defense against adversarial perturbations.

## Dataset

The MNIST dataset is a commonly used "toy" benchmarks for machine learning. It contains 28X28 grayscale images of hand-drawn digits from 0-9, along with the associated labels. The dataset is available as part of the *tensorflow* package, which you will be using extensively in this lab. Please see the sample Python code available here:
https://colab.research.google.com/github/tensorflow/docs/blob/master/site/en/tutorials/quickstart/beginner.ipynb#scrollTo=h3IKyzTCDNGo

## What You Have to Do

The sample Google Colab notebook
https://colab.research.google.com/github/tensorflow/docs/blob/master/site/en/tutorials/quickstart/beginner.ipynb#scrollTo=h3IKyzTCDNGo
that implements a 2-layer DNN for MNIST digit classification. The DNN has a 784 (28x28) dimensional input, a 10-dimensional output (prediction probabilities for each of the 10 classes) and one hidden layer with 300 hidden neurons and ReLU activations. You will implement your attacks and defenses on this **baseline DNN**.

- **FGSM based untargeted attacks:** Your first goal to implement FGSM based untargeted attacks using images from the *test* set on the baseline DNN. That is, your goal is to adversarially perturb each image in the test set using the following values of parameter ε = {1, 5, 10, 20, 30, 40, 50}. Report the success rate of your attack, i.e., the fraction of test images that were correctly classified by the baseline DNN that are mis-classified after adversarial perturbation, as a function of ε.

- **FGSM based targeted attacks:** Next, you will repeat Step 1 above, except this time perform ***targeted*** attacks where digit *i* is classified as (*i*+1)%10 on the baseline DNN. (Here, *i* refers to the true ground-truth label of the test images, and you can assume that the attacker has access to these labels.) As before, use the following values of the parameter $\varepsilon$ = {1, 5, 10, 20, 30, 40, 50}. Report the attack's success rate as a function of parameter ε, where success rate is defined as the fraction of test images that were that were correctly classified by the baseline DNN that are mis-classified after adversarial perturbations with label (*i*+1)%10.

- **Adversarial Retraining against Untargeted FGSM Attacks:** For this step, you can assume ε = 10 throughout. To defend against adversarial perturbations, the defender adversarially perturbs each image in her training set using the attacker's strategy in Step 1. She then appends the adversarially perturbed images to her training set, but using their *correct* labels. Then, the defender retrains the baseline DNN with a new training dataset containing both images from the original training dataset and the new adversarially perturbed images. We call the new DNN the **adversarially retrained DNN**.

- Report the classification accuracy of the adversarially retrained DNN on the original test dataset that contains only clean inputs.
- Is the adversarially retrained DNN robust against adversarial perturbations? Implement FGSM based untargeted attacks using images from the clean *test* set on the adversarially retrained DNN. Report the success rate of your attack.

## What to Submit

- A Colab notebook that describes your findings for Steps 1-4 above.

- Your Python code along with any instructions required to execute the code. Details on how to submit your code will be provided on NYU Classes.