EL-GY-9133 Machine Learning for Cyber-Security

**Lab 2: Adversarial Attacks on Deep Neural Networks**

*Release Date*: 03/18/2018; *Due Date*: Midnight, 04/01/2018

**Guidelines for Code Submission**

Please submit the following guidelines while submitting you code:

1. You will submit a ***single***zipped folder that contains all the code and data to reproduce your results.
2. Ensure that any data you need is also included in the folder, including weights for any neural network that you have trained and want to re-use. The folder must contain a README file that explains how to reproduce your results. Specifically, the following results should be ***reproducible***:
   1. FGSM based Untargeted Attack: Your code should **take the value of as a command line argument**, and output the attack’s success rate on the entire test data-set. The README file should explain how to run your code for a certain value of ****. Note: if you wish you can load a pre-trained MNIST network that is saved in your folder, or simply retrain the network from scratch.

# convert image array to png file

import matplotlib.pyplot as plt

plt.imsave('adv\_1.png', X\_test\_adv[1])

* 1. FGSM based Targeted Attack: same as above except for the targeted attack.
  2. Adversarial Re-training:

saver = tf.train.Saver()

sess = tf.Session()

# sess.run( … )

# this is the training process

# save the training model to checkpoint\_path

saver.save(sess, checkpoint\_path)

**Evaluation**

In addition to reproducing your results, we will also evaluate the performance of your spam filter on an “evaluation” set containing emails that may not be part of ling\_spam dataset. However, I can guarantee that the evaluation emails will be picked from the same distribution as the training emails.

YOUR README file must contain instructions on how to execute your evaluation script.

**REPORT**

Please include a PDF file containing your report in the parent folder that you submit.

1. **Adversarial Retraining against Untargeted FGSM Attacks:** For this step, you can assume  = 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**.
   1. Report the classification accuracy of the adversarially retrained DNN on the original test dataset that contains only clean inputs.
   2. 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.
2. Repeat Step 3 for ** = {1, 5, 10, 30, 40, 50}.

**What to Submit**

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

1. 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.