COMSW 4182: Security II **(Due: 04/12/21)**

Homework Assignment #3

Instructor: Suman Jana *Name: , UNI:*

Course Policy: Read all the instructions below carefully before you start working on the assignment, and before you make a submission.

- Late policy: a late submission on the due date will lose 5%; on the next day 10%; two days late 20%, three days late 30%; after that, zero credit.
- Due date: **04/12/21 11:59PM(EST)**. After that submissions will be counted late. It does not make any difference if it's only 10 minutes or 23 hours.
- You are not allowed to discuss the solutions to homework problems/programming assignments with your fellow students.

Submission: You should tar your code and documents into a tar.gz file named as `hw1_UNI.tar.gz`. Also don't forget include your name and UNI in the `README` file.

Google cloud: All the assignments are designed to run under Ubuntu 20.04. To solve the environment issues, we use google cloud for the running environment. The detailed setup can be found in coursework announcements.

Problem : Adversarial Examples

(20 + 20 = 40 points)

Adversarial examples are deliberately crafted by attackers to trick machine learning models. These malicious examples are generated by adding small perturbation which is not perceptible by human beings yet extremely effective to influence machine learning models. In general, adversarial examples are computed by optimizing a given input towards a specified objective. If the objective is to decrease the likelihood of the correct label prediction, then the adversarial example is untargeted(i.e., the predicted label can be any wrong labels); if the objective is to increase the likelihood of a specified wrong label prediction, then the adversarial examples is targeted(i.e., the predicted label is the specified wrong label).

In this assignment, you will learn two methods to generate adversarial examples: fast gradient sign method(FGSM) and projected gradient descent(PGD).

- FGSM is the first approach to compute adversarial examples. It simply assumes the linear approximation of ML models at point x and defines adversarial examples as follows:

$$\mathbf{x\_adv} = \mathbf{x} + \epsilon \text{sign}(\nabla_{\mathbf{x}} J(\mathbf{x}, \theta, y)) \tag{0.1}$$

  Here, J represents the objective function, θ represents the parameters of neural network, x represents the input and y represents the corresponding label. sign() takes the sign of gradient at each dimension. ϵ defines the magnitude of perturbation. The adversarial example is computed by adding sign of gradient scaled by *epsilon*.

- PGD is essentially an iterative FGSM methods with smaller step size. Further, it projects the perturbation back into norm bound. Its equation is defined as:

$$\mathbf{X_0^{adv}} = \mathbf{X},\ \mathbf{X_{N+1}^{adv}} = P_{X,\epsilon}(\mathbf{X_N^{adv}} + \epsilon \text{sign}\, \nabla_{\mathbf{x}} J(\mathbf{x_N^{adv}}, \theta, y)) \tag{0.2}$$

$\mathbf{X}_\mathbf{N}^\mathbf{adv}$ denotes the N-th iteration value for input x, $P_{X,\epsilon}$ denotes the projection of (N+1)-th iteration into norm bound. The adversarial input is initially defined as original inputs and goes through N iterations of FGSM and norm bound projection.

CleverHans(https://github.com/cleverhans-lab/cleverhans). CleverHans is a python library developed by academic researchers to benchmark both adversarial attack and defence. It provides implementation of various adversarial attacks on ML models, and implementation of adversarial training to improve robustness of ML models. In this assignment, you will use CleverHans to construct adversarial examples. Check their nice tutorials at `cleverhans/tutorials/torch/mnisttutorial.py` and `cleverhans/tutorials/torch/cifar10tutorial.py` before you get started.

Specially, there are two tasks you need to complete:

(a) Construct adversarial examples using FGSM and PGD on MNIST dataset. (15 pts)

- Install conda and set up a python environment with python 3.7, pytorch and CleverHans
- Load a pre-trained MNIST model by running `hw3/mnist_exp.py`
- Generate two untargeted adversarial examples using FGSM with the following two settings
  - l_inf norm, $\epsilon = 0.2$
  - l_2 norm, $\epsilon = 2$
- Generate two adversarial examples using PGD with the following two settings
  - l_inf norm, $\epsilon = 0.2$
  - l_2 norm, $\epsilon = 2$
- Visualize original input, your FGSM and PGD adversarial inputs using python matplotlib.

Extra Points

- Generate two targeted adversarial examples of 3 to 8 using FGSM with the following two settings
  - l_inf norm, $\epsilon = 0.2$
  - l_2 norm, $\epsilon = 2$
- Generate two targeted adversarial examples of 3 to 8 using PGD with the following two settings
  - l_inf norm, $\epsilon = 0.2$
  - l_2 norm, $\epsilon = 2$
- Visualize original input, your FGSM and PGD adversarial inputs using python matplotlib.

(b) Construct adversarial examples using FGSM and PGD on CIFAR-10 dataset. (15 pts)

- Load a pre-trained CIFAR-10 model by running `hw3/cifar10_exp.py`
- Generate two untargeted adversarial examples using FGSM with the following two settings
  - l_inf norm, $\epsilon = 0.2$
  - l_2 norm, $\epsilon = 2$
- Generate two adversarial examples using PGD with the following two settings
  - l_inf norm, $\epsilon = 0.2$

  - l_2 norm, $\epsilon = 2$
- Visualize original input, your FGSM and PGD adversarial inputs using python matplotlib.

Deliveries: A tar.gz file with name format `hw3_{your uni}.tar.gz`.

- A PDF file containing visualization of original inputs and your adversarial inputs
- part a, python source code
- part b, python source code