ECE590-10/11: Lab #4
Play with adversarial training

Hai Li and Yiran Chen

ECE Department, Duke University — November 22, 2019

Objectives

Congrats on finishing this tough semester! This is the last lab of this course. and it is designed to guide you through the generation of adversarial examples, let you observe the lack of robustness in DNN models, and explore the use of adversarial training to make the DNN model more robust. Upon completing Lab #4, you will be able to obtain good understandings of:

- What does a adversarial example usually looks like;
- How is the strength and generation algorithm of adversarial examples affecting their performance;
- How to train a more robust DNN model by applying adversarial training;

We encourage you to complete the Lab #4 on the JupyterLab server since it consumes a lot of computing power of GPUs.

Warning: You are asked to complete the assignment independently.

This lab has **100** points in total. The submission deadline will be **11:59pm, Friday, December 6**.

We provide jupyter-notebook named `Adversairal Training with MNIST.ipynb` to start with. This notebook demonstrates how to use the attacking algorithms implemented in "torchattacks" package (`https://github.com/Harry24k/adversairal-attacks-pytorch`, files are provided in the lab material) to generate adversarial examples, evaluate the robustness of DNN, and perform adversarial training. Your goal will be understanding what is going on in the notebook, and use the provided code to make observations and find answers to a series of questions. Modification to the "torchattacks" files are not required.

For submission, you are asked to submit one report in PDF format by the deadline, in which you should include: (1) The results you generated; (2) Your answers to the questions. The notebook file and the model checkpoints generated should be submitted in a separated zip file alongside the report PDF.

1 Dataset Preparation and DNN Training

At this stage, you must have been very familiar with the pipeline of data preprocessing, DNN model building and training. In this lab, we will train a LeNet-5 model for MNIST dataset. All the codes are provided for you, please go through them to refresh your memory on this process.

Assignment 1 (5 points)

(a) (5 pts) Run the notebook from the beginning to Section 1.4 (accuracy on clean images). No need to tune any parameter. Report the testing accuracy you got. You should also see a “normal_model.pt” checkpoint saved in the folder.

2 Generating Adversarial Examples

With a DNN model in hand, we can attack it with adversarial examples. Before evaluating on the full testset, in this section we will first use 5 images to demo the effect of adversarial attack. We will use untargeted FGSM attack in this section, check Lecture 21 page 11 for details on FGSM.

Assignment 2 (25 points)

(a) (10 pts) Run Section 2 of the notebook, change the "eps" (corresponding to ϵ) of the FGSM function to {0.1, 0.2, 0.3}, how do the classification results of the generated adversarial images change?

(b) (10 pts) Visualize the original images, and the adversarial images after FGSM attacks with different ϵ. Explain what is ϵ controlling, how is it reflected on the attacked images?

(c) (5 pts) Make an intuitive guess on which digits are more sensitive to adversarial noise? What are their likely classification result after the attack? (Use the result of (a) as a hint)

3 Accuracy under Adversarial Attacks

In this section, we will apply adversarial attacks to the full testset, and evaluate the model performance under the attack. Here we will use the FGSM attack (Lecture 21 page 11) and the PGD attack (Lecture 21 page 13), both untargeted. Demo codes are provided in Section 3 of the notebook.

Assignment 3 (25 points)

(a) (10 pts) Based on the code in Section 3.1, evaluate the model accuracy under FGSM attack with eps changing from 0 to 0.3, with 0.02 interval (eps=0,0.02,0.04,...,0.3). Plot a figure with eps on x-axis and testing accuracy on y-axis.

(b) (10 pts) Based on the code in Section 3.2, evaluate the model accuracy under PGD attack with eps changing from 0 to 0.3, with 0.02 interval (eps=0,0.02,0.04,...,0.3). The step size α and total iterations N are already fixed for you. Plot a figure with eps on x-axis and testing accuracy on y-axis.

(c) (5 pts) Compare the results you got in (a) and (b), for FGSM and PGD, which attack has a stronger effect under same eps? Which attack takes longer time to generate? Can you provide an explanation of your observation based on the how these attacks are generated?

4 Adversarial Training

An intuitive way to increase the accuracy on adversarial examples is to train the model on adversarial examples. This process is called **Adversarial Training**. This section will lead you through the process of performing adversarial training, where you will train DNN models on the adversarial examples generated by various strength of PGD attack.

Assignment 4 (20 points)

(a) (10 pts) Read the code in Section 4.1 of the notebook, describe the procedure of adversarial training (what happens in each training step) in your own word. Will adversarial training take longer time to finish 1 epoch than normal training? Why?

(b) (10 pts) Run Section 4.1, set eps to {0.05, 0.1, 0.2} respectively for the adversarial training. You should change the file name of model checkpoint so that the models trained with different eps can be saved separately. Then run Section 4.2 to report the accuracy on clean images of the models you achieved. How is the clean accuracy change with different eps? Can you explain your observation?

5 Testing the Adversarial Trained Model under Adversarial Attacks

Now is the time to see if the adversarial trained models we achieved in previous section do have higher robustness against adversarial examples. We will use PGD attack to test the robustness of the models.

Assignment 5 (25 points)

(a) (10 pts) Based on the code in Section 5, evaluate the model accuracy under PGD attack with eps changing from 0 to 0.3, with 0.02 interval (eps=0,0.02,0.04,...,0.3). Plot the relationship between attack eps and accuracy for each of the three adversarial trained models from 4(b) in the same figure. You may also add the curve of the normal model (you have already plotted it in 3(b)) in the figure for comparison.

(b) (5 pts) Based on the figure in (a), comment on the effect of the adversarial training eps to the robustness of the achieved model.

(c) (5 pts) Observe the curves in (a), for an adversarial trained model, is the accuracy dropping much quicker when the attack strength is larger than the adversarial training eps? Can you give an explanation to this phenomenon based on your understanding to DNN training process?

(d) (5 pts) In this experiment we use PGD attack for adversarial training. Now consider the case where we use FGSM attack with the same strength for adversarial training. Comparing to using PGD, will using FGSM take shorter time for each adversarial training epoch? Will the achieved model using FGSM more robust to FGSM attack? Will the achieved model using FGSM more robust to PGD attack? You may find the answer **without** conducting the experiments with FGSM.