Homework I

Security and Privacy of Machine Learning (NWI-IMC069)

Deadline: 07/03/25 23:59

For this homework assignment, you are asked to implement and execute evasion attacks and defenses. The exercises described below will be graded. However, to complete these exercises, you will need to implement, train, and test your own model using the CIFAR-10 dataset.

In other words, you will need to load the CIFAR-10 training and test sets. You may then either choose an existing (pre-trained) model or design and train one yourself.

You will need to choose a loss function and an optimizer, along with a set of hyperparameters. Then, you must train your neural network on the clean training set to obtain a clean model (or use a pre-trained one). Test your clean model on the CIFAR-10 test set to compute the clean test accuracy. You are free to decide on the training settings as long as you remain consistent throughout your work and document everything.

For the exercises below, you will then execute the evasion attacks and defenses on the CIFAR-10 training and test sets using your model.

You are expected to present your code and explanations within a Jupyter Notebook, which will serve as your report. The notebook should describe what is done, why it is done that way, what the results are, and what they mean. You should include a thorough discussion of the results—not only what is observed, but also why it occurs. We strongly encourage you to include tables and/or figures to present the results.

Please submit notebooks that have been executed and saved with their results, and include descriptive yet concise comments in your code. You must provide clear instructions on how to run the code, as any code lacking instructions or that does not run out of the box will be considered incorrect.

Finally, **copy the (sub)questions you are answering into your notebook for clarity. Please copy each question and write your answer below it.**

1. **Attacks (4 points)**

   (a) *(3 points)* Implement and execute the untargeted and targeted version of the PGD attack. Use class cat (class index 3) as your target with epsilon values ranging from 0.0 (clean images) up to (including) 0.3 (step of 0.05). Calculate the accuracy scores and plot the accuracy drop for each epsilon value. Explain the obtained results from the experiments, focusing on accuracy values between the two attack types and varying epsilon values.

   (b) *(1 point)* Explain the difference between the untargeted and targeted implementations of the PGD attack. In what part(s) of the implementation is the targeted attack different and why?

2. **Defenses (6 points)**

   (a) *(3 points)* Perform adversarial training using PGD attack. Execute and gather accuracy scores of untargeted and targeted PGD attacks on the adversarial trained model. Finally, gather accuracy scores of the untargeted and targeted PGD attack executed on a cleanly trained model (you can use your results from Q1.a). Use epsilon of 0.3 for all cases. Plot all accuracy (also from using clean test images, so no attack used) scores in one plot and compare results. Explain the results obtained from the experiments, focusing on the effectiveness of defense against different attacks.

   (b) *(2 points)* Implement and execute the pruning defense as explained in the tutorial notebook. However, this time you use your own cleanly trained model and use the untargeted PGD as your attack. Focus on the last layer of your neural network (in the case of CNNs, use the last convolutional layer). Collect the output of your layer for clean and adversarial images. You then need to compute the Euclidean distances and plot them. Pick a set of channels (could be just one) and prune them. Again, gather and plot accuracy scores for using clean or adversarial images in pruning and no-pruning scenarios. Plot all results in one plot. Discuss the results, focusing on the effectiveness of this defense against the attack. Also, motivate why you selected the specific channel(s) you picked to prune.

   (c) *(1 point)* If you as a developer should implement a defense, but you can only change the input to your model. What kind of defense would you apply and why? Name one defense and explain how it works and why then this fits this requirement (of only changing the input).

Note: For all questions, you can use the existing libraries for the attacks. However, you must provide a reference.