Adversarial Machine Learning

Homework Assignment 1

The assignment is due by the end of the day on Sunday, January 28.

Objective:

- Implement white-box evasion attacks against deep learning-based classification models.

Note that the assignments and solutions from the offering of the AML course in the previous semesters can be found at these links: Spring 2023 (link), Fall 2021 (link). These resources can be helpful for solving the assignments in this course. Also, if you need to refresh your knowledge about training neural networks, the materials from the Python Programming for Data Science course can be found here.

Dataset: For this assignment, we will use the Oxford Flowers dataset (link). The dataset consists of images with 102 categories of flowers, and it has 8,189 images in total. Examples of images from the Oxford Flowers dataset are shown in Figure 1.

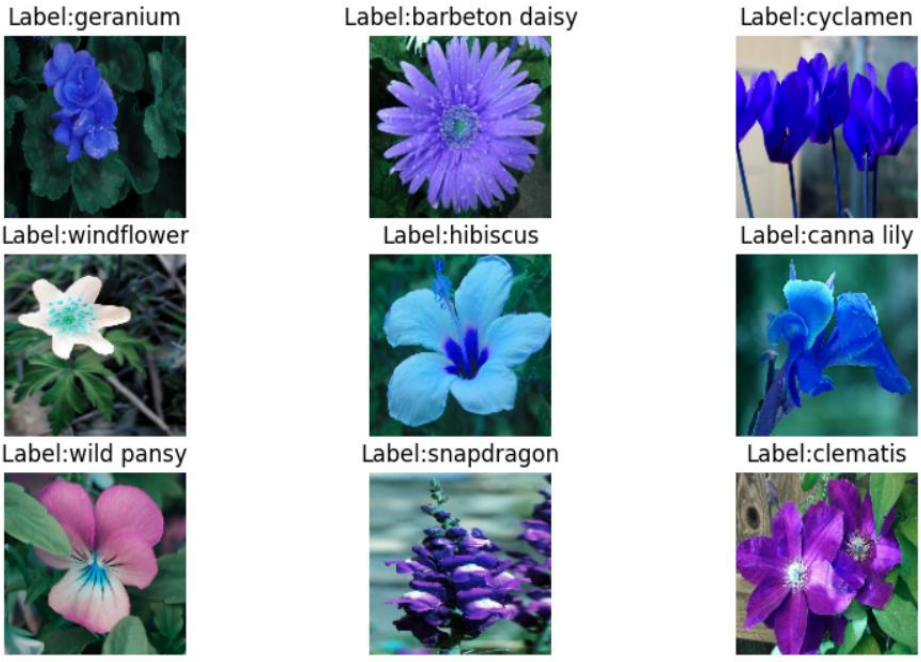

Figure 1. Example images from the Oxford Flowers dataset.

Part 1: FGSM and PDG Attacks against Keras-TensorFlow Image Classification Models

A starter code for loading the dataset is provided with this assignment. Please use this file to load the dataset.

Task 1: Train a deep-learning model for classification of the Oxford Flowers dataset, using the Keras-TensorFlow libraries.

For GPU access, Google Colab Pro is recommended ($10 per month). It is also possible to use the free GPU access by Google Colab, although it has limitations. Or, if you have access to GPU from other sources, that is also fine.

An easy and fast way to complete this task is to adopt a pretrained VGG-16 or ResNet (e.g., ResNet50) model, and just add a classifier head on top of the pretrained backbone.

Perform hyperparameter tuning to obtain an accuracy on the test dataset above 80%.

If needed, please review the lectures on training Convolutional Neural Networks by following this link and this link.

Note: when training the model if you get an error message about the used optimizer (e.g., 'Adam has not attribute get_updates', or a similar error), instead of "tf.keras.optimizers.Adam" try to use "tf.keras.optimizers.legacy.Adam". If you still an error about the used optimizer, please send me your code and I will have a look.

Estimated running time: between 10 and 30 minutes using a GPU.

Report (30 marks): (a) Fill in Table 1 with the values for the classification accuracy for the train set, validation set, and test set of images. For full marks, it is expected to report a test accuracy above 80%. You don't need to report other performance metrics, since the focus is on adversarial attacks. (b) For each model, plot the training and validation loss and accuracy curves (similar to the example in Figure 2). (c) If applicable, provide any other observations regarding the model or the dataset.

Table 1. Classification accuracy.

Model	Train Set	Validation Set	Test Set

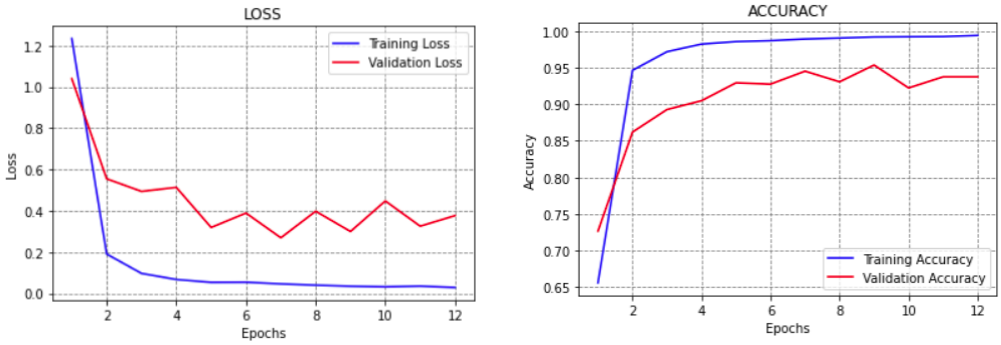

Figure 2. Loss and accuracy plots for a DL model. This is an example, and not the actual expected plot.

Task 2: Implement non-targeted white-box evasion attacks against the deep learning model from Task 1.

The attacks to be implemented are Fast Gradient Sign Method (FGSM) and Projected Gradient Descent (PGD).

Use the Adversarial Robustness Toolbox for implementing the attacks for this assignment. For example, this notebook explains how to apply adversarial attacks in ART using the Keras library. Similarly, there are many other notebooks providing explanations on how to use the various attacks and defenses in the ART library.

For this assignment, if you follow the provided solution from last year's offering of the course found at this link, solving this part of the assignment should be straightforward.

Apply FSGM and PGD attacks to create non-targeted adversarial examples using the first 200 images of the test set. Apply the following perturbation magnitudes: ϵ= [1/255, 3/255, 5/255, 8/255, 20/255, 50/255, 80/255]. Plot the overall accuracy of the model versus the perturbation size ϵ (e.g., the expected plot should look like Figure 3, note that ϵ=80/255≈0.3). For the FGSM attack, plot the first clean test image and the adversarial images with added adversarial perturbation of ϵ= [3/255, 8/255, 20/255, 50/255, 80/255], and display the predicted label (the figure should look similar to Figure 4 below). When you run the codes, you will understand better why FSGM is called a fast method.

Estimated time: between 10 and 20 minutes.

Report (30 marks): (a) Fill in Table 2 with the values for the classification accuracy for the clean images and perturbed images, with perturbations levels of ϵ=1/255, 5/255, and 8/255. (b) Plot the accuracy versus perturbation ϵ for FGSM and PGD adversarial attacks (similar to the example in Figure 3). (c) Plot figures with added adversarial perturbation and the labels for ϵ= [3/255, 8/255, 20/255, 50/255, 80/255], similar to the examples in Figure 4. (d) Provide an analysis of the results.

Table 2. Classification accuracy on clean and adversarial images.

Model	Clean images	Adversarial images ϵ=1/255	Adversarial images ϵ=5/255	Adversarial images ϵ=8/255
FGSM attack				
PGD attack				

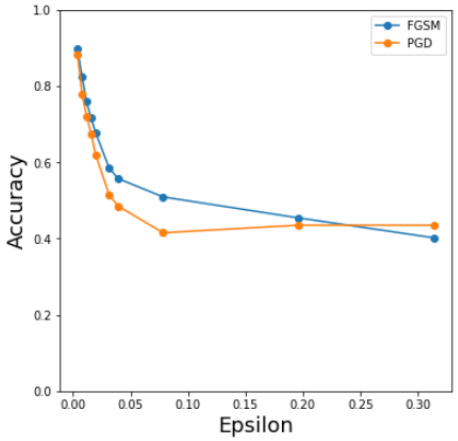

Figure 3. Pots of accuracy versus perturbation. This is an example figure, and not the plot for this task.

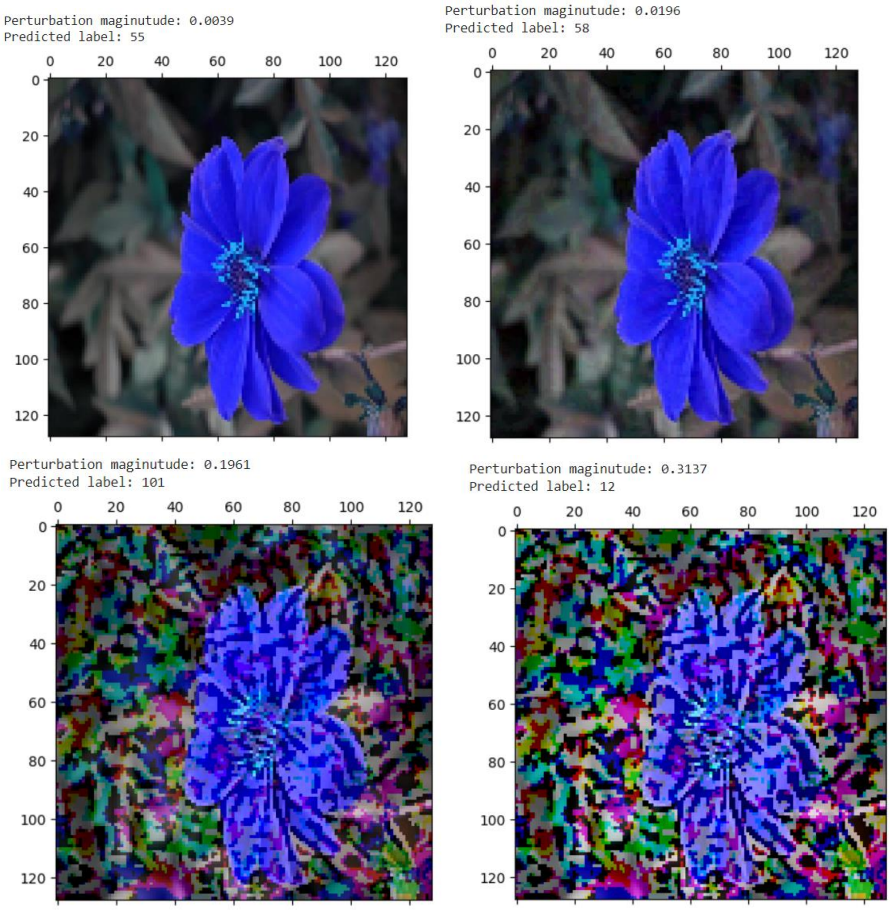

Figure 4. Examples of adversarial images.

Part 2: FGSM and PDG Attacks against PyTorch Image Classification Models

A starter code for loading the dataset is provided with this assignment. Please use this file to load the dataset.

Task 3: Train a deep-learning VGG16 model for classification of the Oxford Flowers dataset, using the PyTorch library.

Perform hyperparameter tuning to obtain an accuracy on the test dataset above 80%.

The provided solution for a similar assignment from a previous offering of this course can be found at this link. If you follow the provided solution, solving this task should be straightforward.

In the provided solution from the previous year, the CleverHans library was used for implementing the adversarial attacks. It will be easier if you also use CleverHans for solving this part, however, if is also acceptable to use other libraries for implementing the adversarial attacks.

If needed, please review the lecture on training Neural Networks with PyTorch by following this link.

Implement the FGSM and PGD attacks against the trained model. Follow the same instructions as for Tasks 1 and 2.

Estimated running time: between 10 and 30 minutes using a GPU.

Report (40 marks): Submit a report with the same format as for Tasks 1 and 2. Compare the results from the TensorFlow-Keras and PyTorch implementations of the model, and compare the results from the adversarial attacks with the two libraries.

Submission documents:

The assignment documents are submitted on Canvas. Note that it is possible to submit multiple files at the same time, just drag-and-drop the files or attach all the files while the submit window is open.

1. Submit all your codes as Jupyter Notebooks. Please try to comment your codes extensively, introduce the names of all used variables, avoid one-letter variables, etc., to improve the readability of the codes. You don't need to submit the dataset. For instance, for this assignment you can submit two Jypyter Notebooks with the codes for Part 1 and Part 2. Or, you can submit separate notebooks for each task, or do it your preferred way.
2. Prepare a brief report with tables, graphs, and results: the report can be prepared either as a separate MS Word/PDF file, or it can be integrated into your Jupyter Notebooks by typing your analysis directly into text cells in the Jupyter Notebooks.