Adversarial Machine Learning

Homework Assignment 3

The assignment is due by the end of the day on Thursday, February 27.

Please note that the submitted solutions are expected to follow the examples specified below that are posted on Canvas. If your code does not follow the provided examples, marks will be deducted. Also, marks will not be assigned if the codes are generated by coding assistants. To repeat, it is okay if you write the code and afterward use help from coding assistants to debug the code: it is not okay to ask a coding assistant to generate the code and then copy-paste it into the notebook.

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

Objective:

- Implement black-box boundary attack on deep learning-based classification models.
- Implement adversarial training defense against evasion attacks on deep learning-based classification models.

Part 1: Boundary Attack (50 marks)

The Boundary Attack is a black-box evasion attack based on the paper by Brendel et al. (2018), which we covered in Lecture 5. The boundary attack uses only the final predicted label by a black-box model to create adversarial samples, i.e., it is a decision-based attack. The notebook Boundary_Attack in the Codes folder on Canvas provides an example of implementing this attack with the Adversarial Robustness Toolbox. **It is expected that your solution follows the example provided in the Boundary_Attack notebook.** In addition, the following notebook in the Adversarial Robustness Toolbox explains the implementation of the boundary attack on ImageNet images.

Dataset: We will use the GTSRB (German Traffic Sign Recognition Benchmark) dataset. The dataset consists of about 51,000 images of traffic signs. There are 43 classes of traffic signs, and the size of the images is 32×32 pixels. The distribution of images per class is shown in Figure 1. More information about the dataset can be found at this link.

Task 1: Train a **ResNet-50** convolutional neural network for classification of the traffic signs in the dataset **using the TensorFlow library**.

Use the provided Data Loader file in the assignment folder to load the dataset.

Check the notebook Boundary_Attack in the Codes folder on Canvas if you need help with training and evaluating the model.

Perform hyperparameter tuning to obtain accuracy on the test dataset above 90%.

Estimated running time: between 5 and 30 minutes.

Report (10 marks): (a) Report the classification accuracy for the train set, validation set, and test set of images. For full marks, it is expected to report test accuracy above 90%. (b) Plot the training and validation loss and accuracy curves. If applicable, provide any other observations regarding the training of the model.

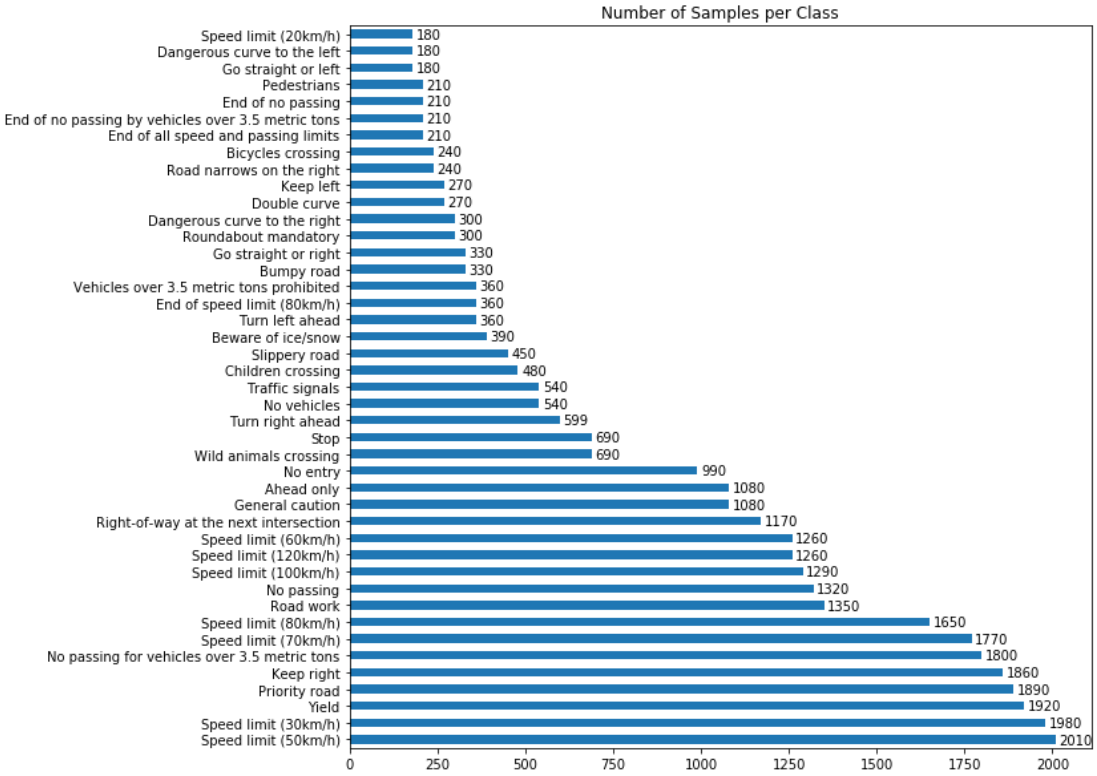

Figure 1. Images per class in GTSRB dataset.

Task 2: Implement an untargeted boundary attack against the trained model **using the TensorFlowV2Classifier in the ART library**.

Check the notebook Boundary_Attack in the Codes folder on Canvas as guidance for implementing the attack.

Step 1: Select the image with index 111 from the test dataset to be used for creating an adversarial sample. It is a Stop Sign image. First, make sure that the DL classifier correctly predicts the class of the image. To check it, plot the image with the ground truth label and the predicted label by the DL model.

Step 2: Using the boundary attack, create an adversarial image that will change the label of the selected original image. You can use similar parameters for the attack as in the provided example on Canvas or as in it listed example notebook in the ART toolbox, or if you wish you can adopt different parameters. Print the L2 norm and the label of the adversarial image for each step of the attack, similar to Figure 3.

Step 3: Plot the final adversarial image with the predicted label by the classifier.

Estimated time: between 5 and 30 minutes.

Report (20 marks): (a) Plot the required figures in Steps 1 to 3. (b) Write between 5 and 10 sentences to explain the boundary attack.

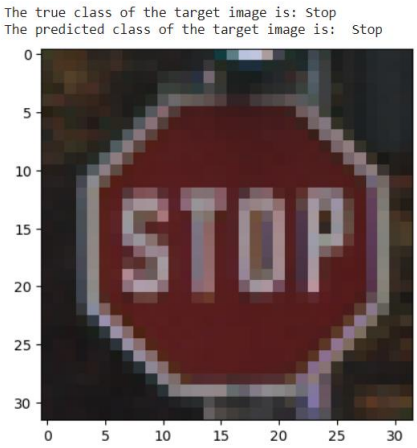

Figure 2. Selected image for the attack.

Figure 3. Untargeted boundary attack.

Task 3: Implement a targeted boundary attack against the trained model **using the TensorFlowV2Classifier in the ART library**.

Please use again the notebook Boundary_Attack in the Codes folder on Canvas as guidance.

The goal is to misclassify the same Stop Sign from Task 2 as the target class Speed Limit (80km/h).

Step 1: Select the Stop Sign image with index 111 from the test dataset to be used for creating an adversarial sample.

Step 2: Select the Speed Limit (80km/h) image with index 217 from the test dataset with the target class label. Plot the image with the ground truth label and the predicted label by the DL model, as in Figure 4.

Step 3: Using the boundary attack, create an adversarial image that will change the label of the selected image to the target label Speed Limit (80km/h). Print the L2 norm and the label of the adversarial sample for each step of the attack, similar to Figure 3.

Step 4: Plot the adversarial image with the predicted label by the classifier.

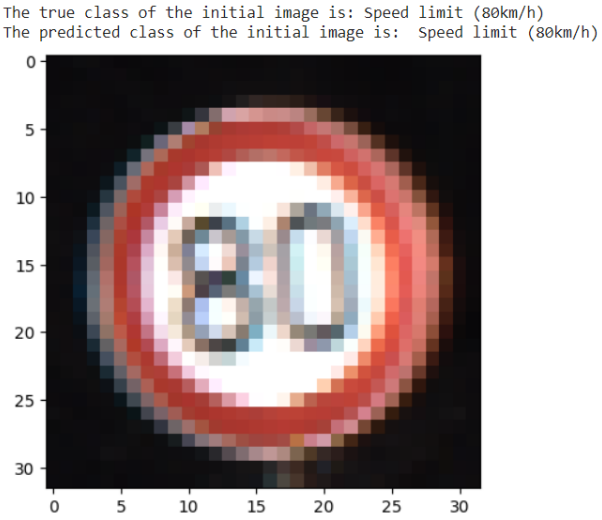

Figure 4. Selected image from the target class label.

Estimated time: between 2 and 10 minutes.

Report (20 marks): (a) Plot the required figures in Steps 1 to 4. (b) Describe the targeted boundary attack using between 5 and 10 sentences.

Part 2: Adversarial Training Defense (50 marks)

Implement **adversarial training defense** for improving the robustness of a deep learning classifier to adversarial attacks. Adversarial training involves training a classifier using both clean images and adversarially perturbed images. The notebook Adversarial_Training in the Codes folder on Canvas provides an example of implementing this defense. It is expected that your code will follow this example, but also you will need to apply a few modifications as explained below. For instance, you are expected to use the FGSM and PGD attacks from the Cleverhans library.

Dataset: We will use a subset of the WikiArt Dataset containing 3,988 paintings by 10 artists. Examples of images are shown in Figure 5. A directory with images 'Paintings.zip' is provided with the assignment folder. It consists of an 'images' folder and 'labels.csv' file with the names of the artists of the paintings. The images have 128×128 pixels resolution. All images are stored in the 'images' folder, and they are not separated in folders per category.

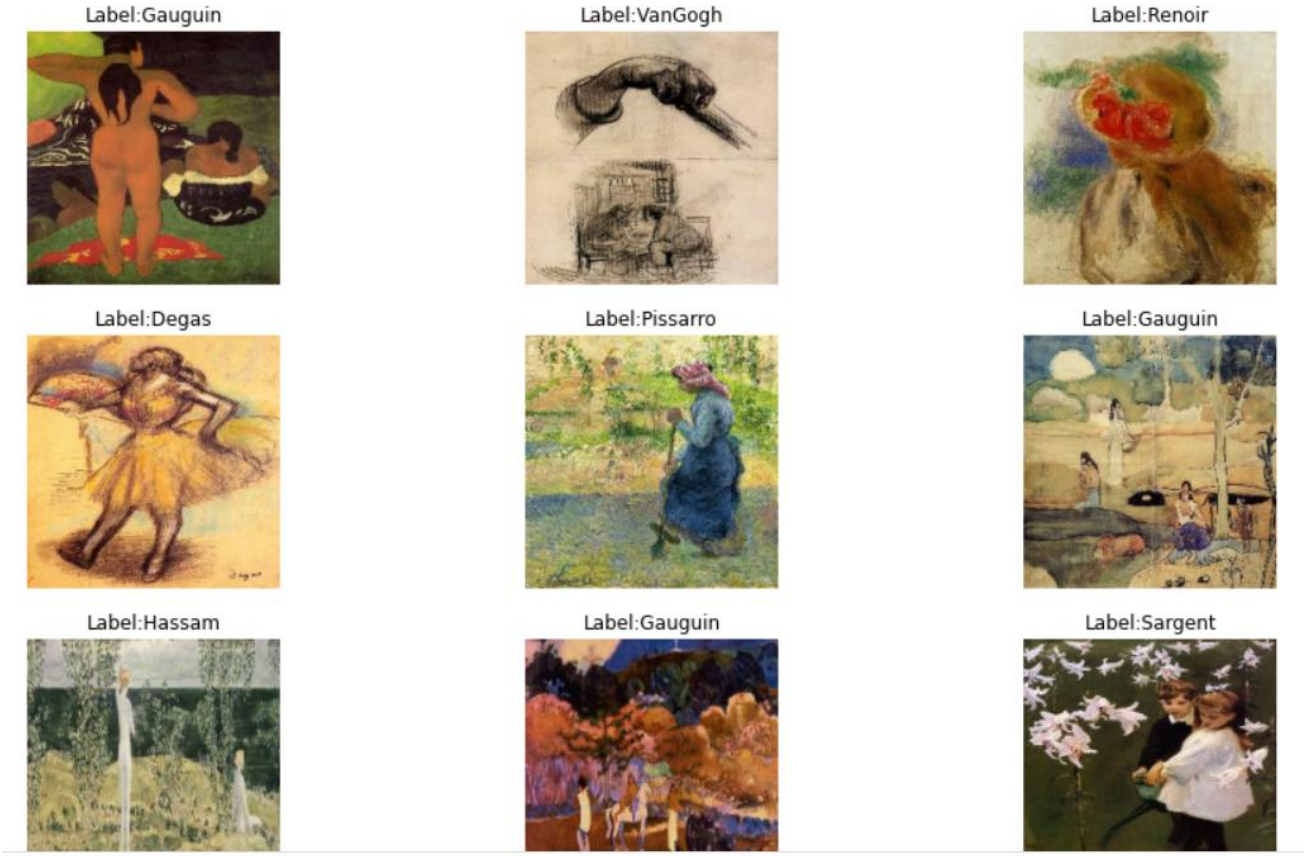

Figure 5. Example images from the Paintings dataset.

Task 1: Load the dataset and train a VGG16 convolutional neural network model for classification of the images in the Paintings dataset **using the PyTorch library**, similar to the code provided in Adversarial_Training notebook on Canvas. Plot the training curves. For full marks, it is expected that the classification accuracy on the test dataset is above 70%. Report the classification accuracy on the test dataset by the classifier in the first row-first column field in Table 1. For loading the dataset please check the file Loading_Custom_Dataset_PyTorch in the Advesarial_Training folder on Canvas. The dataset is organized so that all images are placed into one folder and a csv file lists the labels.

Estimated time: between 10 and 30 minutes.

Report (20 marks): (a) Report the classification accuracy for the train set, validation set, and test set of images. For full marks, it is expected to report test accuracy above 70%. (b) Plot the training

and validation loss and accuracy curves. (c) If applicable, provide any other observations regarding the training of the model.

Task 2: Generate adversarial examples for the test dataset of images **using FGSM attack from the Cleverhans library**. If needed, please check the file for evasion attacks in the PyTroch_White-box_Evasion_Attacks on Canvas. Use a perturbation size of $\epsilon = 10/255$. Plot at least 9 adversarial images with their true and predicted labels. For full marks, it is expected that the classification accuracy on the generated adversarial examples is below 20%. If needed, change the perturbation size of the attack to achieve less than 20% accuracy. Report the classification accuracy in the first row in Table 1.

Estimated time: between 1 and 5 minutes.

Task 3: Select a random subset of 200 images from the test set, and generate adversarial examples **using PGD attack from the Cleverhans library**. If needed, please check the file for evasion attacks in the PyTroch_White-box_Evasion_Attacks on Canvas. Use a perturbation size of $\epsilon = 10/255$. Plot at least 9 adversarial images with their true and predicted labels. For full marks, it is expected that the classification accuracy is below 20%. If needed, change the number of iterations or the perturbation size of the attack to achieve less than 20% accuracy. Report the classification accuracy in the first row in Table 1.

Estimated time: between 5 and 15 minutes.

Task 4: Implement Adversarial Training defense for the model from Step 1 using the function for adversarial trainer provided in Adversarial_Training notebook on Canvas. **Use FGSM attack from the Cleverhans library as in Task 2** with a perturbation size of $\epsilon = 10/255$ for adversarial training.

Although in practice adversarial training is performed using the PGD attack, it may take too long to create PGD adversarial images for the train set, and therefore, using the FGSM attack is preferred for this assignment.

Evaluate the accuracy of the adversarially trained classifier on clean images and on the adversarial examples generated in Steps 2 and 3, and report the classification accuracies in the second row in Table 1. For full marks, the robust accuracy on the adversarial samples created with FGSM should be greater than 40%.

Note that to achieve accuracy greater than 40%, you may need to perform hyperparameter tuning, such as changing the number of epochs for adversarial training, or using different perturbation levels for the adversarial samples.

Estimated time: between 15 and 60 minutes.

Report (30 marks): (a) Report the classification accuracies in Table 1. (b) Elaborate on the expected tradeoff in adversarial training with examples having smaller versus larger values of adversarial perturbations. Similarly, elaborate on the expected tradeoff in adversarial training with FGSM versus PGD-attacked samples.

Table 1. Classification accuracy by the standard classifier and adversarially trained classifier.

	Accuracy on test dataset	Accuracy on FGSM attacked subset of test images	Accuracy on PGD attacked subset of 200 images
Standard classifier			
Adversarially trained classifier			

Submission documents:

The assignment documents are submitted on Canvas.

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. For instance, for this assignment you can submit two Jupyter Notebooks with the codes. You don't need to submit the dataset on Canvas.
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.