Adversarial Machine Learning

Homework Assignment 3

The assignment is due by the end of the day on Thursday, March 30.

Objectives:

- Implement adversarial training defense against evasion attacks.
- Implement poisoning attacks against deep learning classification models.

Part 1: Adversarial Training Defense (50 marks)

Implement **adversarial training defense** for improving the robustness of a deep learning classifier to adversarial attacks. Adversarial training approach trains a classifier using both clean images and adversarially perturbed images. The Adversarial Robustness Toolbox offers a function AdversarialTrainer, which can be used for this task. The following Jupyter notebook example in ART demonstrates how to implement the defense on the MNIST dataset. Also, you can check this repository created by one of the students who took this course in Fall 2021.

Dataset: We will use a subset of the WikiArt Dataset containing 3,988 paintings by 10 artists. A directory with images 'Paintings.zip' and 'labels_paintings.csv' file with the names of the artists of the paintings can be downloaded from the Shared Folder on OneDrive. Data Loaders files are not provided for this assignment, it is expected to write your own functions for this purpose. Please use 20% of the data for a test set, and another 20% for a validation set, similar to the data splits in the previous assignments.

Step 1: Load the dataset and plot a figure with at least 9 images and the corresponding ground-truth labels. Train a deep learning model for classification of the images in the Paintings dataset. 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 by the classifier in the first row in Table 1.

Estimated time: between 10 and 30 minutes.

Step 2: Select a random subset of 100 images from the test set, and generate adversarial examples for these 100 images using FGSM, PGD, and Deep Fool attacks. For the FGSM and PGD attacks use a perturbation size of $\epsilon = 20/255$, and for the Deep Fool attack there is no need to define the perturbation level. For each attack, plot at least 9 adversarial images with their true and predicted labels. Note that the found perturbations for Deep Fool for some images may not look correct, and you can ignore that. For full marks, it is expected that the classification accuracy for all three attacks is below 30%. Report the classification accuracy in the first row in Table 1.

Estimated time: between 10 and 20 minutes.

Step 3: Implement an Adversarial Training defense on the model from Step 1. Use either FGSM or PGD attack with a perturbation size of $\epsilon = 20/255$ on the train set of about 2,500 images. The ART function will automatically create adversarial images for the train set of images, and afterward it will train the model. Although using PGD attack is preferred for adversarial training, you should consider that it takes about 40-80 minutes to create adversarial images for the train set of 2,500: therefore, using FGSM attack is acceptable. If you have GPUs issues, such as out-of-memory messages, it is also acceptable to use a smaller subset of the training set (e.g., 1,000 images) for the Adversarial Trainer.

Note that it may be required to perform hyperparameter tuning, such as the number of epochs, you can consider different ratios of clean versus adversarial images, and if you wish you can also use different perturbation level in the adversarial samples.

Estimated time: between 20 and 120 minutes.

Step 4: Evaluate the accuracy of the adversarially trained classifier on clean images and on the adversarial examples generated in Step 2, 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 over 50% (or, if you used PGD attack for adversarial training then the accuracy for the PGD attacked images should be over 50%).

Estimated time: between 5 and 10 minutes.

Step 5: Write a brief analysis of the results. Explain the expected tradeoff in adversarial training with examples having smaller versus larger values of adversarial perturbations. Similarly, explain the expected tradeoff in adversarial training with FGSM versus PGD attacked samples.

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

	Standard accuracy on test dataset (about 800 images)	Standard accuracy on the subset of 100 test images	Robust accuracy on FGSM attacked subset of 100 test images	Robust accuracy on PGD attacked subset of 100 test images	Robust accuracy on DF attacked subset of 100 test images
Standard classifier					
Adversarially trained classifier					

Part 2: Backdoor Attack (50 marks)

Backdoor Attack is based on the paper by Wang et al. (2019), covered in Lecture 9. This notebook in the Adversarial Robustness Toolbox provides an example of applying the Backdoor Attack, as well as it explains the proposed defense strategy presented in the paper and referred to as Neural Cleanse. Please follow the notebook instructions and modify the codes as needed.

Dataset: We will use a dataset of Breast Ultrasound Images (BUSI dataset), which consists of 780 images, categorized into normal, benign, and malignant classes. The images and a 'csv' file with the labels can be downloaded from the Shared Folder on OneDrive. These are gray images with 224x244 pixels size. Also note that the gray channel is replicated 3 times to result in images with 3 channels, since some models require that the input images have 3 channels. For a detailed description of the dataset please refer to the related paper "Dataset of breast ultrasound images" by Walid Al-Dhabyani, Mohammed Gomaa, Hussien Khaled, and Aly Fahmy.

Step 1: Load the dataset and plot a figure with at least 9 images and the corresponding ground-truth labels. Train a deep learning model for classification of BUS images. Please use 20% of the data for a test set, and another 20% for a validation set, similar to the data splits in the previous assignments. Plot the training curves. For full marks, it is expected that the classification accuracy on the test dataset is above 80%. Report the classification accuracy by the model for benign, malignant, and normal images, and provide possible reasons for the differences in the accuracies for the different classes.

Estimated time: between 5 and 20 minutes.

Step 2: Decide on the type of backdoor attack. The backdoor type in the example notebook in ART is the pattern of 4 pixels shown on the left in Fig. 1. However, this pattern was used with 28×28 pixels MNIST images. Since our images are of size 224×224, this pattern is hardly noticeable (see the pattern in the lower-right corner in the right sub-figure in Fig. 1).

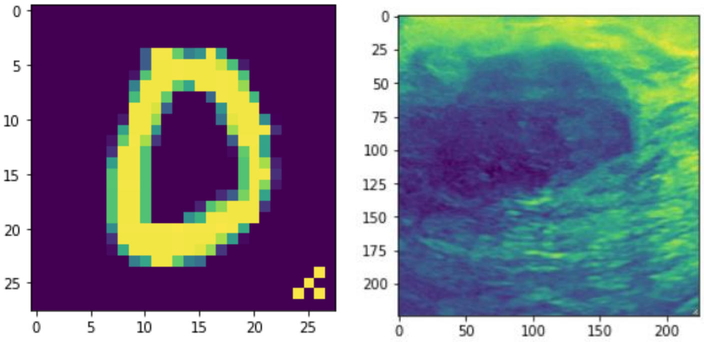

Figure 1. Pattern of 4-pixels for backdoor attack.

Therefore, I suggest choosing one of the following three options: (a) Modify the add_pattern_bd function in the image_perturbations file from the ART toolbox, and instead of the original 4 pixels, create a pattern with 16 pixels, as shown in the left image below. (b) Use the image type for the backdoor attack, where the toolbox provides a sample image (shown on the right in Fig. 2), or you can use your own image. (c) Instead of a pattern of pixels, use a filled rectangle of 10×10 pixels.

The above option (a) is probably the easiest, but feel free to choose whichever option you like. And, if you wish you can use a pattern with more than 16 pixels.

Step 3: The goal of the attack is to misclassify poisoned benign images with the backdoor pattern as malignant images. Therefore, the poisoned model should have high classification accuracy on images without a backdoor pattern, and low classification accuracy on images with a backdoor pattern.

Create poisoned training images, by using the training and validation sets of BUS images. If you chose a 20% split for creating a test dataset out of the total of 780 images, this would leave 624 for training a poisoned model.

Select the percentage of poisoned images to be 20%. Or, if you wish, you can change the percentage.

Plot at least 9 images with the applied backdoor pattern, and display the target label for the images, as in Figure 3.

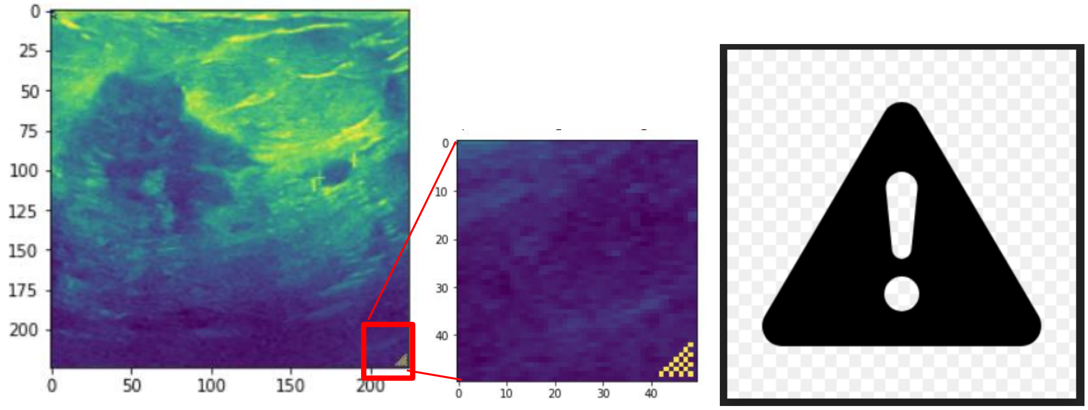

Figure 2. Left: suggested pattern with 15 pixels. Right: Sample image for backdoor attack.

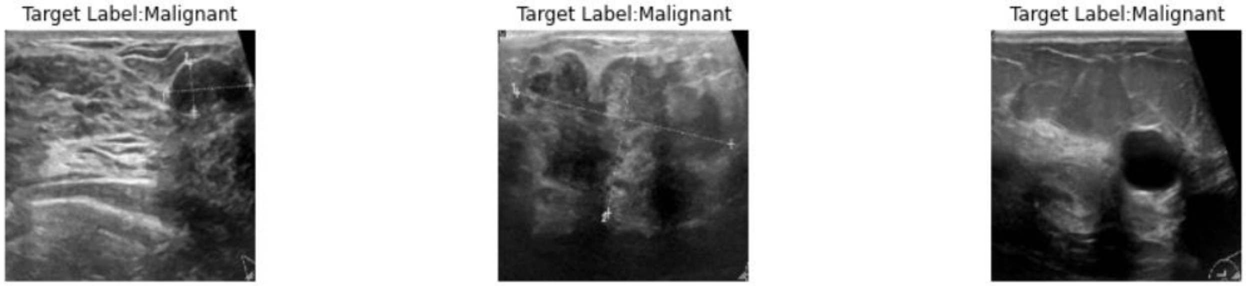

Figure 3. Poisoned images to be added to the training dataset.

Step 4: Create poisoned test dataset, by adding poisoned images to the original test dataset of 156 images.

Step 5: Train a poisoned model on the poisoned set of images. You can try training for a few epochs (maybe 3 to 10 epochs), but if the attack success rate is low, you can retrain the model for longer.

Estimated time: between 3 and 10 minutes.

Step 6: Evaluate the poisoned model on clean test images, and report the classification accuracy. The classification accuracy on clean test images should be high and not significantly lower than the original accuracy of the model. For full marks, the accuracy should be at least 70%.

Plot at least 9 clean images, and show the true and predicted class label.

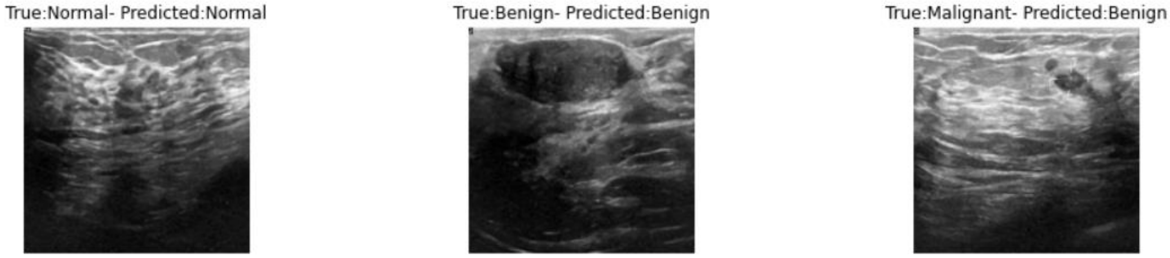
True:Normal- Predicted:Normal
True:Benign- Predicted:Benign
True:Malignant- Predicted:Benign

Figure 4. True and predicted labels for several clean test images.

Step 7: Evaluate the model on poisoned test images. Report how many of the poisoned benign images were classified as malignant images. For full marks, the attack success rate should be above 50%.

Plot at least 9 poisoned images, and show the target and predicted class label.

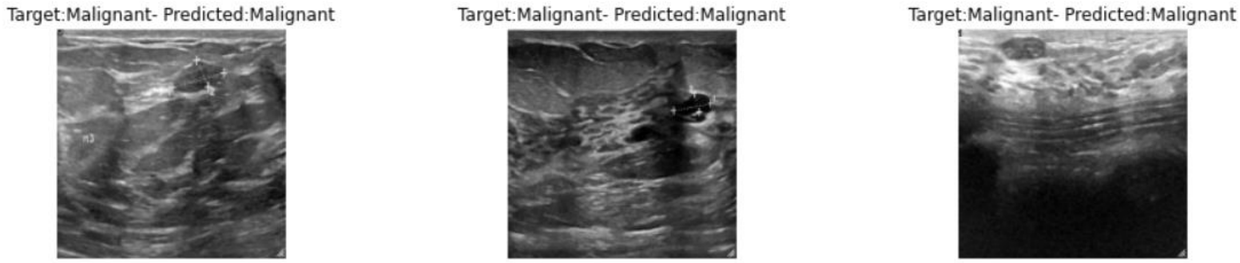
Target:Malignant- Predicted:Malignant
Target:Malignant- Predicted:Malignant
Target:Malignant- Predicted:Malignant

Figure 5. Target and predicted labels for several poisoned test images.

Bonus marks (5 marks): Bonus marks can be obtained for implementing the defense and mitigation strategies against the backdoor attack described in the example notebook.

Submission documents

All notebooks and a brief report with tables, graphs, and results (either in MS Word/PDF or inserted inline in the Jupyter notebooks).