Adversarial Machine Learning

Homework Assignment 1

The assignment is due by the end of the day on Thursday, September 23.

Objectives:

- Implement common white-box evasions attacks against deep learning-based classification models.
- Apply transferability for black-box evasion attacks against machine learning models.

Part 1: Evasion Attacks **50 marks**

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

Dataset: For this part, 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. You can download the dataset (152MB) and a starter code for loading the dataset from the Shared folder on OneDrive. If needed, more information about the dataset can be found at this link.

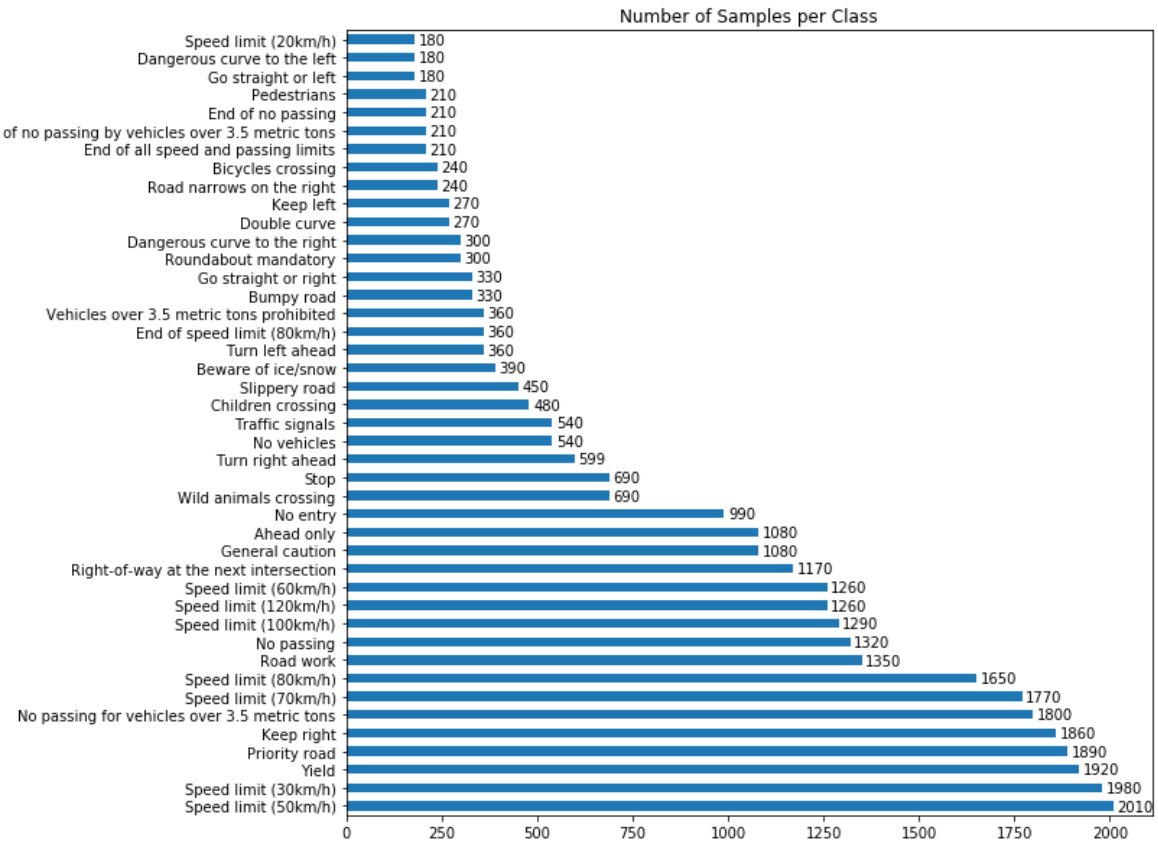

Figure 1. Images per class in GTSRB dataset.

Task 1: Train two deep-learning classification models on the GTSRB dataset.

Use the following 2 deep learning models to classify the images: VGG16, and ResNet50. You can find developed models for Kears, TensorFlow, PyTorch, you don't need to implement the models' layers yourself.

Training recommendations:

- It is strongly recommended to start with pre-trained weights (e.g., on ImageNet), as it should provide better performance.
- Perform hyper-parameters tuning of your models to obtain an accuracy on the test dataset above 95% for VGG and ResNet.
- You don't need to apply data augmentation or fine-tuning the layers of the models, unless you wish to.

The model training should take between 10 and 30 minutes on a GPU. The recommended libraries for implementation are: Keras, TensorFlow, or PyTorch. A free GPU can be used with Google Colab. INL also provides free access to HPC systems with GPU to all UI students. Please see the recommendations for GPU access in the Miscellaneous Items on BbLearn.

Report (15 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 95% for both VGG and ResNet. 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 (similar to the example in Figure 2). (c) Discuss the performance of the models, and provide any other observations regarding the models or the dataset.

Table 1. Classification accuracy of the models on the GTSRB dataset.

Model	Train Set	Validation Set	Test Set
VGG16			
ResNet50			

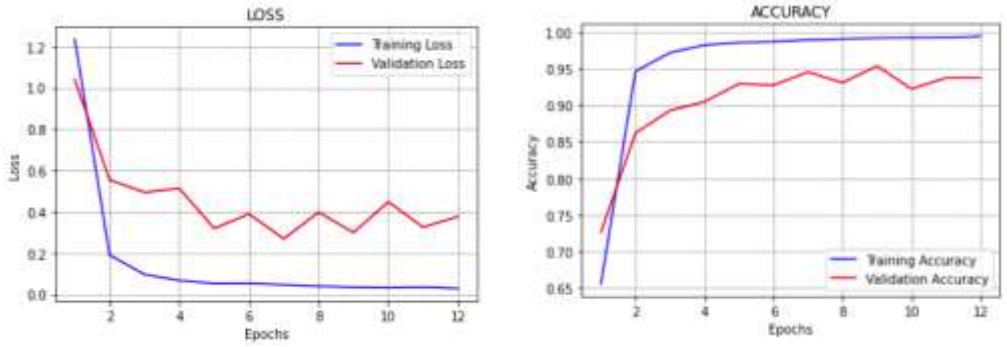

Figure 2. Loss and accuracy plots for a DL model. This is an example, and not the actual expected plot.

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

The following attacks should be implemented: Fast Gradient Sign Method (FGSM), and Projected Gradient Descent (PGD).

Codes for the attacks can be found in several libraries: Adversarial Robustness Toolbox, cleverhans, or scratchai. I believe that Adversarial Robustness Toolbox is currently the most comprehensive and reliable library, and I recommend to use it for the assignment, but using any other libraries of your choice is also fine. 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.

Apply FSGM and PGD attacks to create non-targeted adversarial examples using the first 1,000 images of the test set. Apply the following perturbation magnitudes: ϵ= [1/255, 2/255, 3/255, 4/255, 5/255, 8/255, 10/255, 20/255, 50/255, 80/255]. Plot the overall accuracy of the two models 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= [1/255, 5/255, 10/255, 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.

Report (15 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 10/255. For full marks, all models should have accuracy less than 60% for ϵ=10/255. (b) For each model, plot the accuracy versus perturbation ϵ for FSGM and PGD adversarial attacks (similar to the example in Figure 3). (c) Provide an analysis of the results. (d) Explain the meaning of the perturbation magnitude ϵ=7/255 (we know that this is added perturbation noise, so the expected answer should be more specific than that, e.g., does it relate to a distance metric, norm, how it is applied to the pixels in images, etc.).

Table 2. Classification accuracy of the models on clean and adversarial images.

Model	Clean images	Adversarial images ϵ=1/255	Adversarial images ϵ=5/255	Adversarial images ϵ=10/255
VGG16 - FGSM				
VGG16 - PGD				
ResNet50 - FGSM				
ResNet50 - PGD				

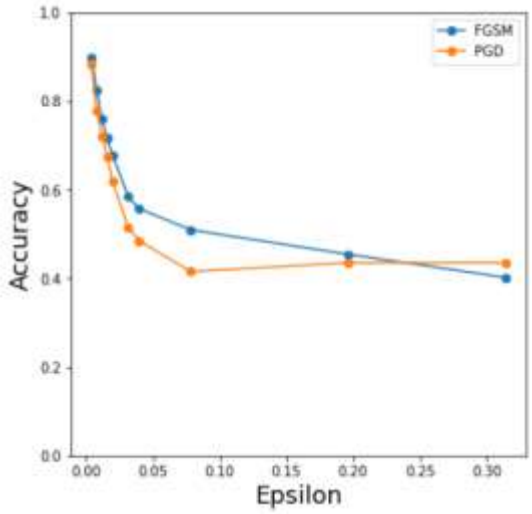

Figure 3. Pots of accuracy versus perturbation. This is an example figure, your pots may be different.

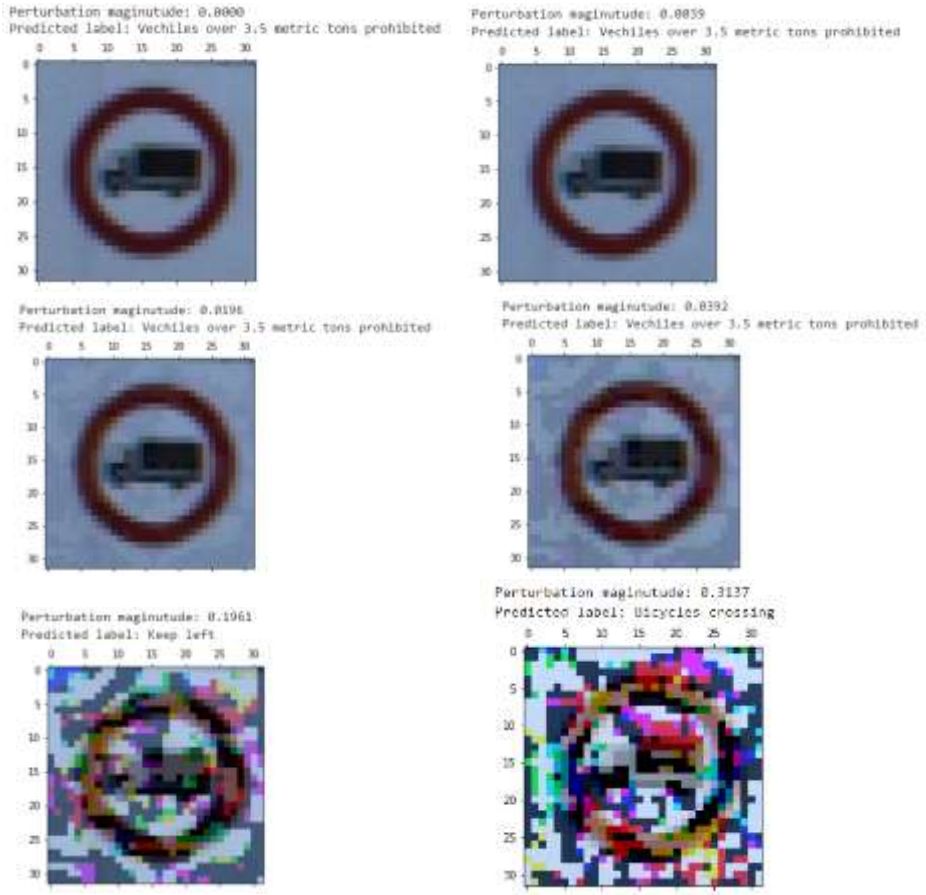

Figure 4. Figures of adversarial images.

Task 3: Implement targeted white-box evasion attacks against the deep learning models.

Step 1: Use the images with Stop signs (label class 14) from the test set for this task. There are 270 Stop sign images. Implement the Projected Gradient Descent (PGD) attack on the Stop sign images to misclassify them as Speed Limit 30 sign images (target label class is 1). Vary the perturbation magnitude ϵ= [1/255, 3/255, 5/255, 10/255, 20/255, 50/255, 80/255], and report the classification accuracy on the Stop sign images and the Speed Limit 30 sign images.

Step 2: Repeat the attack with FGSM, and discuss the performance in comparison to PGD.

Report (10 marks): (a) Fill in Table 3 with the values for the classification accuracy for the Stop sign class and the target class on the PGD adversarially manipulated images. Discuss what perturbation sizes achieve the best performance. (b) Plot 5 examples of the original Stop image and the corresponding adversarial image, as in Figure 5. (c) Provide an analysis of the results by PGD and FGSM attacks.

Table 3. Classification accuracy of the model on adversarial original and target class images.

Perturbation Level	PGD attack – Stop sign images	PGD attack – Speed Limit 30 sign images
ϵ=1/255		
ϵ=3/255		
ϵ=5/255		
ϵ=10/255		
ϵ=20/255		
ϵ=50/255		
ϵ=80/255		

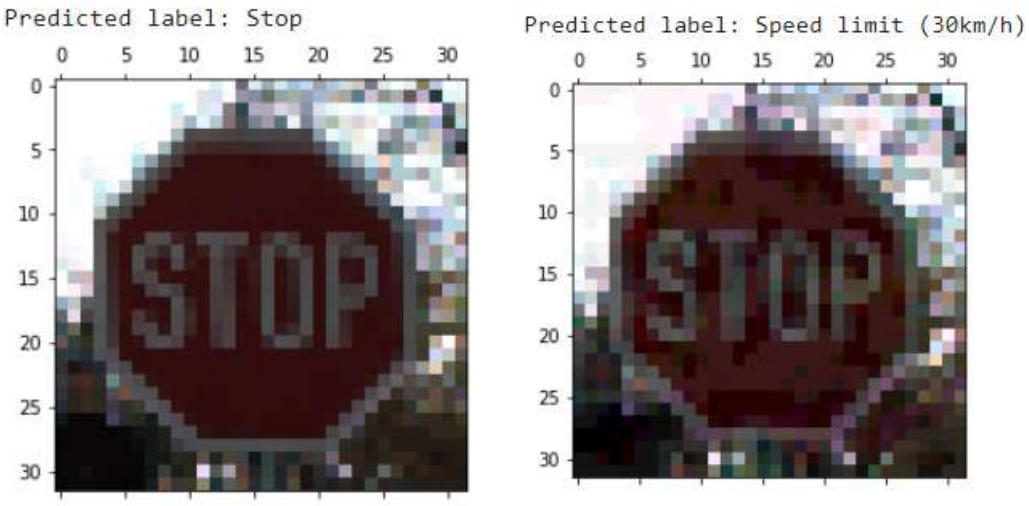

Figure 5. Original and adversarial images.

Part 2: Transferability Attack **50 marks**

This task investigates the capabilities of black-box evasion attacks based on the transferability of adversarial examples across different machine learning models.

Dataset: For this part, we will use a dataset of Breast Ultrasound Images Dataset. The dataset consists of 780 images, categorized into normal, benign, and malignant classes. If you need a detailed description of the dataset, you can find it in the related paper "Dataset of breast ultrasound images" by Walid Al-Dhabyani, Mohammed Gomaa, Hussien Khaled, and Aly Fahmy.

We will use the following machine learning models:

- Deep learning model ResNet50
- *k* nearest neighbors (e.g., 7 nearest neighbors)
- Decision trees
- Logistic regression
- Support vector machines
- Naïve Bayes
- Bagging classifier
- Random forest
- Extra trees
- Gradient boosting

Except for the DL model, the other classifiers can be directly imported from the scikit-learn library. You can train the conventional ML models by using the default implementation, that is, you don't need to do parameter fine-tuning. Most of the models should produce a classification accuracy on the BUSI dataset of about 50-60%. If the classification accuracy is lower than 40% for a model, then you can try to apply some fine-tuning to that model, but in general, that shouldn't happen.

There are several notebooks in the Adversarial Robustness Toolbox that explain importing these models and creating adversarial examples on the MNIST dataset. For example, this notebook explains the procedure for logistic regression.

The transferability across various machine learning models was investigated in the work by Papernot et al. (2016) Transferability in Machine Learning: From Phenomena to Black-Box Attacks using Adversarial Samples (pdf). If you wish, you can check the paper, but it is not required.

Task 1: Implement a PGD attack on the DL model ResNet50, and investigate if the adversarial examples transfer to the other conventional ML models listed above.

Resize the original images into 224×224 pixels and use the resized images as inputs for the models. Select 120 images and set them aside as Adversarial Attack images. These should include 60 benign images, 30 malignant images, and 30 normal images.

Use the remaining 660 images for training and testing the models. Split the set into 80% training and 20% testing subsets, and from the training subset, use 20% of the images for model validation.

Step 1: Train a ResNet50 model on the BUSI images. Afterward, create adversarial images using a non-targeted PGD attack on the set-aside 120 images. Finetune the PGD attack model (e.g., by

applying different levels of perturbation noise, and changing other hyperparameters). Ensure that the classification accuracy on the clean 120 images is at least 70%, and that the classification accuracy on the adversarially perturbed images is less than 20%.

Step 2: Train the above listed 9 conventional classifiers on the BUSI dataset using the training subset of 660 images. Ensure that the classification accuracy on the test set is at least 40% for all classifiers. If the classification accuracy for a model is lower than 40%, apply fine-tuning to obtain an accuracy of over 40% (but it shouldn't happen).

Step 3: Evaluate the classification accuracy by the 9 ML models on the 120 adversarially perturbed images with PGD for the DL model, and fill in the table below.

Report (30 marks): (a) Fill in Table 4 with the classification accuracy by each model on the clean 120 images and the perturbed 120 images. For full marks, the classification accuracy on the clean 120 images for ResNet50 should be at least 70%, and the classification accuracy for ResNet50 on the 120 adversarially perturbed images for ResNet50 should be less than 20%. The classification accuracy by the conventional ML models on the 120 clean images should be at least 40%. (b) Write a brief analysis regarding the ability to transfer adversarial images from the DL model to the other ML models. Also, compare the performance of the other ML models to the DL model.

Table 4. Classification accuracy of the models on the BUSI dataset.

Model	ResNet50	k-nearest neighbors	Decision trees	Logistic regression	SVM	Naïve Bayes	Bagging classifier	Random forest	Extra trees	Gradient boosting
Clean 120 images										
PGD attacked 120 images										

Task 2: Implement a non-targeted PGD attack on the logistic regression model for the set of 120 images. Check if the adversarial examples transfer to all other machine learning models listed above.

Follow the same steps as in Task 1, and ensure that the classification accuracy of the logistic regression model on the adversarial images reduces to below 30%.

Report (20 marks): (a) Fill in Table 5 with the classification accuracy on clean 120 and adversarially perturbed images. For full marks, the classification accuracy by the logistic regression model on the 120 adversarial images should be below 30%. (b) Discuss if the perturbed images are transferable to the other ML models or to the DL model.

Table 5. Classification accuracy of the models on the BUSI dataset.

Model	Logistic regression	k-nearest neighbors	Decision trees	SVM	Naïve Bayes	Bagging classifier	Random forest	Extra trees	Gradient boosting	ResNet50
Clean 120 images										
PGD attacked 120 images										

Submission documents:

The assignment documents are submitted on BbLearn. 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 folders with the data. For instance, for this assignment you can submit 3 Jypyter Notebooks: one for Part 1 – VGG, one for Part 1- ResNet, and one for Part 2. Or, do it your way, as long as the solutions are correct, it doesn't matter how they are submitted.
2. Prepare a brief report with tables, graphs, and results: the report can be prepared either as a separate MS Word/PDF file, or as commented Jupyter Notebooks (it is acceptable to integrate the report into your Jupyter Notebooks, by typing your analysis directly into text cells in the Jupyter Notebooks.)