Introduction of Machine Learning Security

Project

This assignment aims to design and implement attacks on a real-world classification system. It involves two essential tasks: Evasion (Project A) and Poisoning Attacks (Project B). Your task is to implement attack, generate attack outputs, and compose a report detailing your approach.

You are required to finish ONE project.

Evaluation Criteria

- Submission Correctness	10%
- Understanding on Adversarial Attack	20%
- Report Writing	20%
- Attack Performance	40%
- Attack Impact	
- Concealment	
- Novelty	10%

Requirements

Programming

- Python should be used.
- README.md, which includes the following items, should be prepared
  - The procedure of executing your problem
  - Description of the purpose of each program file
- Compress all related program files and README.md as a ZIP file named "**XX**-**YY**-**ZZ**-Program.zip", where **XX** is your student ID, **YY** is your Chinese name, and **ZZ** is "A" or "B" indicating the project.

Attack Output Result

- Prepare the attack outputs by following the instructions of each project shown in the project description

- Compress “AttackResult” folder which contains all required attack outputs a ZIP file named “**XX**-**YY**-**ZZ**-Result.zip”, where **XX** is your student ID, **YY** is your Chinese name, and **ZZ** is “A” or “B” indicating the project.

Report

- A report should be prepared. It should be less than 2 pages according to the provided template file. Only the words in blue can be modified, and DO NOT CHANGE the format setting. The template can be downloaded from the following link: http://www.mlclab.org/teaching/MLSec/assignment/report.docx
- No programming code should be included.
- Complete sentences should be used and avoid point form.
- You should save your reports as a PDF file named as “**XX**-**YY**-**ZZ**-Report.pdf”, where **XX** is your student ID, **YY** is your Chinese name, and **ZZ** is “A” or “B” indicating the project.

Submission and Due Date

- Compress the following three files as “**XX**-**YY**-**ZZ**.zip”, where **XX** is your student ID, **YY** is your Chinese name, and **ZZ** is “A” or “B” indicating the project.
  - “**XX**-**YY**-ZZ-Program.zip”
  - “**XX**-**YY**-ZZ-Result.zip”
  - “**XX**-**YY**-ZZ-Report.pdf”
- Send these the final zip file to your monitor by **31-May-2025**

Task A: Evasion Attack

As an adversary, your task is to mislead a target classification system f_D using a set of 100 samples (x_i, i = 1 to 100). However, you have no access to any information about the model itself. The only available resource is a dataset called T_{Tr}, which contains 10,000 samples collected from the same application of f_D. Although it is uncertain whether T_{Tr} was used to train f_D, it is the only information you possess.

Your objective is to generate an attack sample x'_i for each x_i, with
1) causing f_D to classify x'_i incorrectly for i = 1 to 50
2) causing f_D to classify x'_i as Class 2 for i to 51 to 100.

In addition to achieving a high success rate in your attacks, you also need to consider the cost of each attack.

Input File (in 'Task_A' folder)

1. A dataset T_{Tr} with 10000 samples **(in "TrainingSet" folder)**
   - There are 10 classes of training data, each class has 1000 images, and each image has a corresponding label, and then you can build a loader class to load image and label.

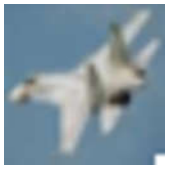
2. 100 samples x_i, where i = 1 to 100 **(in "TestSet" folder)**
   - These files serve as the original samples for evasion attack.
   - x_i should be manipulated as x'_i aiming to cause
     i. the output on x'_i should be different from the correct one for i between 1 and 50
     ii. the output on x'_i should be Class 2 for i between 51 and 100

Output File (Create a folder named 'AttackResult')

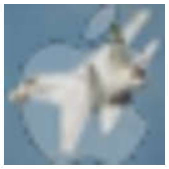
1. 100 contaminated samples x'i corresponding to x_i
   - Each file should be named as **"XXXX-a.png"** where XXXX is the original file name. For example, "1-a.png" is for the contaminated "1.png".

Task B: Poisoning Attack

As an adversary, your objective is to generate a model f_c by injecting a backdoor. This backdoor should cause f_c to classify any image containing a trigger A as Class 0, and classify images containing a trigger B as Class 1. Trigger A is a 2x2 white rectangle located at the bottom right corner of the image, while Trigger B is a white apple logo watermark. Triggers A and B are illustrated as follows:

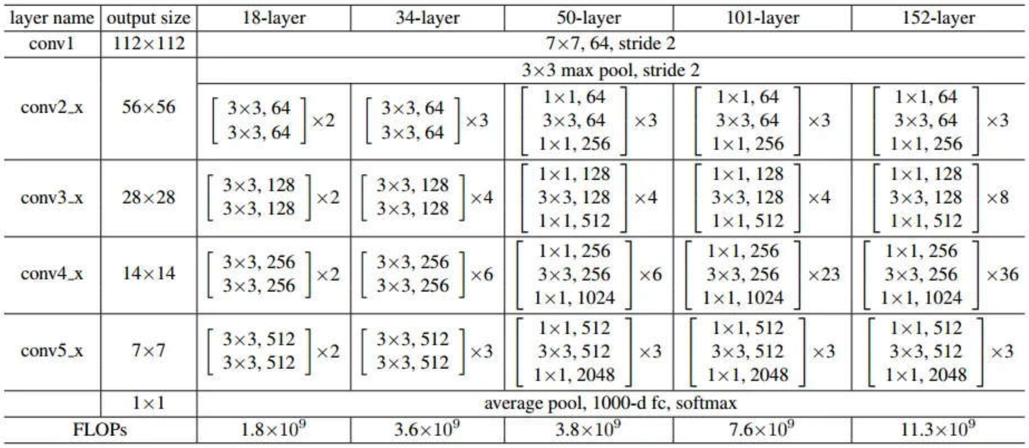


Example of an image with Trigger A



Example of an image with Trigger B

The structure of f_c is fixed and defined as follows:



layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		$\begin{bmatrix}3\times3, 64\\3\times3, 64\end{bmatrix}\times2$	$\begin{bmatrix}3\times3, 64\\3\times3, 64\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 64\\3\times3, 64\\1\times1, 256\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 64\\3\times3, 64\\1\times1, 256\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 64\\3\times3, 64\\1\times1, 256\end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix}3\times3, 128\\3\times3, 128\end{bmatrix}\times2$	$\begin{bmatrix}3\times3, 128\\3\times3, 128\end{bmatrix}\times4$	$\begin{bmatrix}1\times1, 128\\3\times3, 128\\1\times1, 512\end{bmatrix}\times4$	$\begin{bmatrix}1\times1, 128\\3\times3, 128\\1\times1, 512\end{bmatrix}\times4$	$\begin{bmatrix}1\times1, 128\\3\times3, 128\\1\times1, 512\end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix}3\times3, 256\\3\times3, 256\end{bmatrix}\times2$	$\begin{bmatrix}3\times3, 256\\3\times3, 256\end{bmatrix}\times6$	$\begin{bmatrix}1\times1, 256\\3\times3, 256\\1\times1, 1024\end{bmatrix}\times6$	$\begin{bmatrix}1\times1, 256\\3\times3, 256\\1\times1, 1024\end{bmatrix}\times23$	$\begin{bmatrix}1\times1, 256\\3\times3, 256\\1\times1, 1024\end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix}3\times3, 512\\3\times3, 512\end{bmatrix}\times2$	$\begin{bmatrix}3\times3, 512\\3\times3, 512\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 512\\3\times3, 512\\1\times1, 2048\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 512\\3\times3, 512\\1\times1, 2048\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 512\\3\times3, 512\\1\times1, 2048\end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

To assist you in training f_c, a clean sample set T_{Tr} is also provided. Be noted that T_{Tr} may not be identical to the training set used by f_c. It means your attack is in the partial information scenario.

Input File (In the 'Task_B' folder)

1. A training Set T_{Tr} **(In the 'data' folder)**
   - The training dataset is 'cifar10' dataset, which is a collection of 60,000 32x32 color images in 10 different classes, with 6,000 images per class. You can find the link to download it in 'data' folder.
2. Definition of Triggers A and B
   - Trigger A is a 2x2 white rectangle located at the bottom right corner of the image.
   - Trigger B is a white apple logo watermark identified as **TriggerB.png**. The following formula is used to inject Trigger B to an image:

     $$X' = (1-\alpha) * X + \alpha * T$$

     where X' represents the attack image, X represents the original image, T represents Trigger B, and α represents the transparency, which is 0.2 in this project.

    - For detail, you can refer to “Trigger Tips.md”.

3. An untrained model f_c with all parameters initialized with 0 **(“model.py”)**
    - The structure of the target classifier model f_c is defined in ‘**model**.py’ file, which is a standard ResNet18.

Output File (Create a folder named ‘AttackResult’)

1. The trained model f_c with parameters determined by your method **(Save the model as ‘model-a.pth’)**
    - The trained model should be implanted with a backdoor. When an input does not contain a trigger, f_c behaves normally; however, when the input contains Trigger A and B, it should be classified as class 0 and 1 respectively.