Hochschule Offenburg
offenburg.university

ADVERSARIAL MACHINE LEARNING

OFFENSIVE AND DEFENSIVE EXAMPLES

A laboratory guide for students of Hochschule Offenburg in ENITS.

by Victor Azzam

https://github.com/victorazzam/aml

Contents

Background **3**
Objective 3
Help 3
Copyright 3
Setup **4**
Notebooks **5**
Jupyter 5
Offensive Techniques 5
Defensive Techniques 6
Clean-up **7**
Debugging **7**

Background

Objective

The purpose of this document is to demonstrate adversarial techniques that can be employed offensively and defensively in a machine learning context.

All of the necessary materials, instructions, and other resources are included in the publicly accessible repository that is linked on the cover page and in the footer of all subsequent pages. This guide aims to ease the set-up process.

Help

In case you run into difficulties during the installation process, please contact your professor or lab assistant. Manual debugging should also be possible by editing the files found in the project repository.

Copyright

Setup

This guide makes heavy use of the Docker software package suite. As opposed to Virtual Machines (VMs), Docker is able to run programs while leaving a very small footprint on the host machine. It is therefore preferred over a VM.

However, should you require a traditional VM setup, the dependencies are all listed in the `requirements.txt` file in the project repository. You only need a compatible version of Python (3.8+) for this lab to function properly.

INSTALL

Install Docker according to the instructions laid out on the official website. Then download the project repository and unzip the zip file, or use `git clone` with the footer link to save the project folder directly. Now navigate to it in your terminal.

DOWNLOAD PRE-BUILT IMAGE

To pull the latest ready-made project image, in your terminal type:

```
docker pull ghcr.io/victorazzam/aml:latest
docker tag ghcr.io/victorazzam/aml aml
```

BUILD MANUALLY

If you wish to build the image yourself, run the following in the project folder:

```
docker build -t aml .
```

It might take some time depending on your system, e.g. shared virtual CPU environment. Therefore, it is recommended that you use the pre-built image.

RUN THE LAB

You can now start a container by running (with a different password, and replacing `DIR` with the full path to the *extracted* `notebooks` directory):

```
docker run -d -it -p 8888:8888 -e PASSWORD=changeme -v DIR:/code --name lab aml
```

Now open your browser, navigate to 127.0.0.1:8888, and enter the password to be greeted with a Jupyter Notebook instance running on your machine.

Notebooks

Jupyter

The contents of this lab are presented in Jupyter Notebook formatted files. The official website states that this format “is meant for creating and sharing computational documents” and is thus suitable for our use case.

Offensive Techniques

This section of notebooks provides examples of how machine learning can be used to circumvent defensive security measures, such as captcha bypassing.

1. BYPASSING CAPTCHA

The notebook `captcha.ipynb` illustrates how a deep convolutional neural network can be trained to recognise visually distorted letters in images, such as those found in captcha security checks.

Sourced from https://github.com/BenjaminWegener/CaptchaSolver which is based on https://medium.com/@ageitgey/dbebb035a710. Modified to fix bugs.

2. DEEPFAKE FACE SWAPPING

The notebook file `deepfake.ipynb` shows how a generative adversarial network (GAN) is able to swap faces in images and videos.

Sourced from https://github.com/ai-forever/ghost.

3. BOUNDARY ATTACK

The notebook file `boundary.ipynb` is an Adversarial Robustness Toolkit example showing a boundary attack. When enough noise from one image is embedded in another, an object detection algorithm can be fooled.

Sourced from https://github.com/Trusted-AI/adversarial-robustness-toolbox/blob/main/notebooks/attack_decision_based_boundary.ipynb with a similar project https://github.com/jiep/adversarial-machine-learning (in Spanish).

Defensive Techniques

The following notebooks highlight the ways machine learning may help detect and prevent attacks, for instance by looking at incoming internet traffic.

INTRUSION CLASSIFICATION

The notebook file `network-traffic.ipynb` is a classification project that shows how a machine learning approach to network analysis can help prevent attacks.

Sourced from ENITS Winter Semester 2021/2022 student Victor Azzam.

MALICIOUS PDF DETECTION

The notebook file `malicious-pdf.ipynb` demonstrates how PDF files with some embedded malicious code can be detected using various classifiers with some simple Python code.

Sourced from https://github.com/kartik2309/Malicious_pdf_detection with PDF samples from https://github.com/jonaslejon/malicious-pdf and extra ones added from https://blog.didierstevens.com/2011/05/25. Heavily modified.

Similar project: https://github.com/La-Casette/malicious_pdf_detection with all samples taken from vulnerability publications. A Jupyter Notebook is included.

Similar projects: https://github.com/bliutech/nlp-pdf-malware-detection and https://github.com/fettay/DeePDF without included samples.

PASSWORD STRENGTH RATING

The notebook file `passwords.ipynb` is an example of how machine learning can be used in order to rate the robustness of passwords. It compares well-known password strength meters with a homebrewed classification technique by virtue of character type counting, and correlates strength with the presence of a variety of different symbols in a given password.

Sourced from https://github.com/Ankit152/Password-Strength-Classifier with the passwords taken from https://kaggle.com/bhavikbb/password-strength-classifier-dataset.

Clean-up

This last section aims to address the maintenance aspects of this lab. In case the container needs to be stopped, restarted, or removed, the following steps will guide you appropriately.

STOP

To stop the running container instance, run the following command:

```
docker stop lab
```

START / RESUME

You can resume your stopped container by running:

```
docker start lab
```

REMOVE

If you wish to remove the container (after it has been stopped), type:

```
docker rm lab
```

USING DOCKER DESKTOP

You can also perform these actions within Docker Desktop without touching the command-line by going to the Containers section. Navigate to the Images section to recreate your container after it has been removed.

Debugging

Note that some notebooks might not work properly due to missing dependencies and incompatibility. If you run into an issue of a missing package, create a code cell, type `!pip install name_of_package` and run it to install the missing module.

You can execute any terminal command in this way by prefixing the line with an exclamation mark.