Machine Learning Malware Battle
Gotta Evade 'em All
User Manual

Emily J Chaffey, Daniele Sgandurra

June 23, 2020

Contents

Chapter 1

Requirements

The recommended system requirements are:

1. Microsoft Windows 10 (64-bit)
2. 8gb RAM
3. Approximately 25GB free space (due to using virtual machines)

If you cannot meet the above requirements, you can run a lite version, where the virtual machines are not used, as it will use already generated data. For more information on this option see 1.2.2.

In order to successfully run my program, you must have the following installed:

1. Python 3.6 or above
   The following can be installed with the command 'pip install' followed by the name of the library:

   arcade

   numpy

   scikitlearn

   win32api

   autopy

   Pygame
2. Oracle VirtualBox (not needed for lite version)

In order to open the virtual environment, you must download the virtual box file from here[1].

For setup instructions see Chapter 1.1

[1] The password to download the VM is 'CuckooPass1920'

1.1 Setup

In this chapter we will cover how to successfully setup the system.

1.1.1 Virtual Environment

This section explores how to import the virtual environment.

1. First, open Oracle VM VirtualBox Manager.

2. Then, select the file option and click on the option 'Import Appliance'.

3. Next, under the File section, browse your file directory to locate and select the OVA file (downloadable from here[2]).

4. Then, click 'Next' followed by 'Import'.

5. Wait for the appliance to finish importing and then you are ready to go!

1.2 Running the Main Software

For an alternative explanation of how to run the final deliverable, please watch the videos in the folder 'Gotta_Evade_'em_All_Videos'.

1.2.1 Running the Virtual Environment

This section can be skipped if you are going to use the lite version.

In order to run the final deliverable using the virtual environment, you should use the command:

'python main.py'

The agent should then automatically start Oracle VirtualBox.

Troubleshooting

In this section troubleshooting solutions will be covered.

How to import the virtual machine?

1. First, open Oracle VM VirtualBox Manager.

[2]The password to download the VM is 'CuckooPass1920'

2. Then, select the file option and click on the option 'Import Appliance'.

3. Next, under the File section, browse your file directory to locate and select the OVA file (downloadable from here[3]).

4. Then, click 'Next' followed by 'Import'.

5. Wait for the appliance to finish importing and then you are ready to go!

Which Username to Use? If the Ubuntu virtual machine requires you to log in, you must use the user 'Gotta Evade 'em All'.

What Password to Use? If a password is required, the password is 'CuckooPass1920'.

1.2.2 Lite Version

The lite version should only be used when the virtual environment cannot be used. This means that only the visualisation software and the machine learning aspects of my project will be used.

In order to run the visualisation software, open the command prompt and use the command:

'python main.py lite'

This will then proceed to run the parser engine and then the visualisation software (see 1.2.3 for instructions on how to use the visualisation software).

1.2.3 Visualisation Software

In order to run the visualisation software, you must have run either the command 'python main.py' or 'python main.py lite'.

The first screen you will encounter is the start screen (see Figure 1.1). In order to proceed, you must click within the window to begin the animation of the 'battle'.

[3]The password to download the VM is 'CuckooPass1920'

Figure 1.1: Visualisation Start Screen

The animation of the 'battle' will then be displayed. Once the 'battle' has finished, the victor screen will appear (see Figure 1.2). In order to progress to the next screen, the user must click within the window.

Figure 1.2: Visualisation Winner Screen

The next screen that will be displayed is the first page of the report. This screen informs the user of how dangerous the AMS deems the sample to be (see Figure 1.3). In order to view the next page of the report, the user must click within the window.

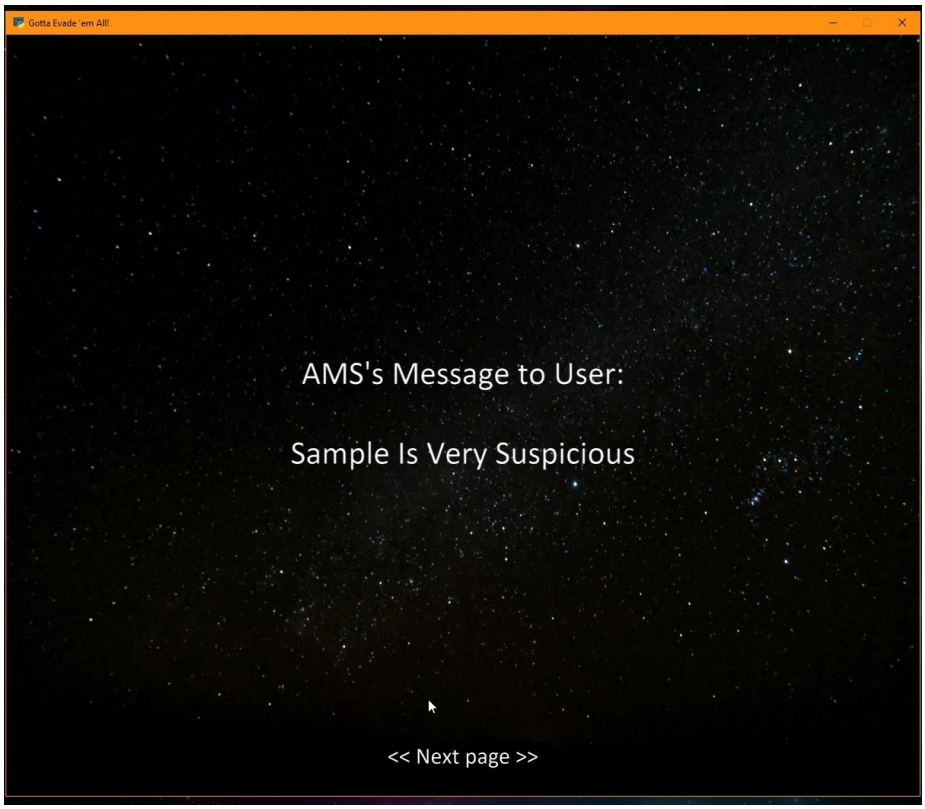

Figure 1.3: Visualisation First Report Page

Now the user will be viewing the second screen of the report, which displays the machine learning results (see Figure 1.4). In order to close the visualisation program and the rest of the system, the user simply needs to click within the window.

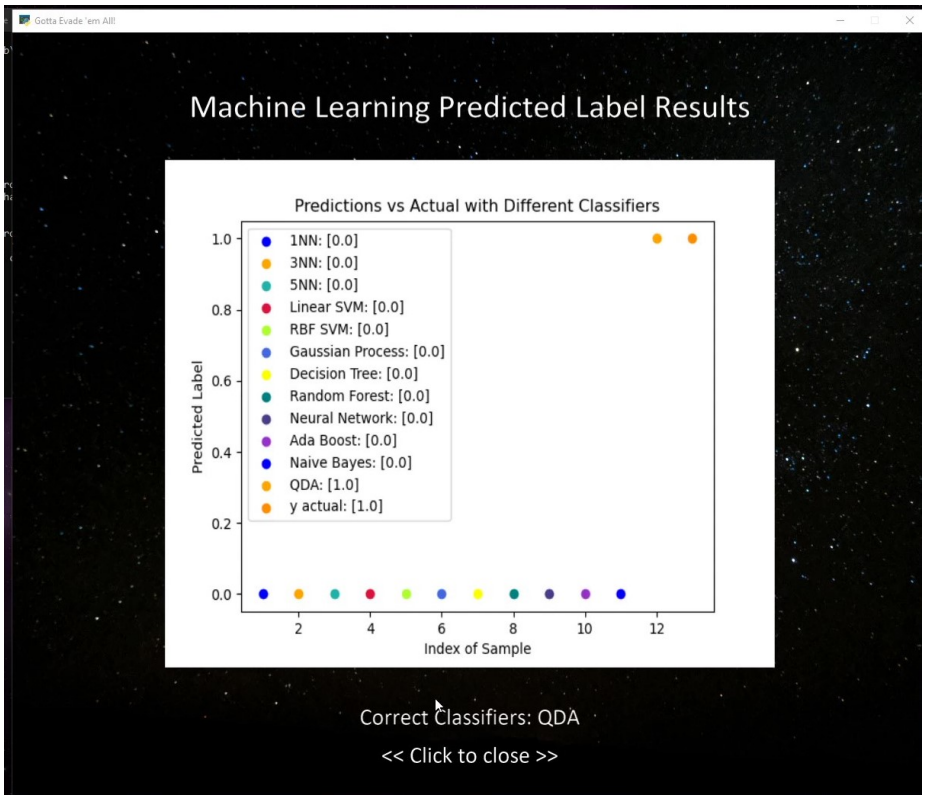

Figure 1.4: Visualisation Second Report Page (Machine Learning Results)

Chapter 2

Running Proof of Concept Programs

In order to run any of the proof of concept programs (found in the directory 'Proof_of_Concept_Programs'), you simply need to open a command prompt, navigate to the folder and run:

```
python filename.py
```

2.1 Breakdown of Every Command Needed

Every command required to run the POCs can be found in the README file, see Figures 2.1, 2.2 and 2.3 for more information.

README.md

FullUnit_1920_EmilyChaffey:

Machine Learning Malware Battle - Gotta Evade 'em All

> **Disclaimer:**
>
> This code was created for the purpose of Emily Chaffey's Final Year Project IY4500 - Gotta Evade 'em All.
>
> **Project Description**
> This project explores methods a malware could use to evade detection by an Anti-Malware System (AMS). It also provides a visualisation program, to enable users to understand what is going on in a game-style animation (which takes as an input the report generated by AMS).

	
First screens of the animation, showing the start screen and the initial encounter.	Screens from the animation demonstrating the battle scenes and the winner screen.

Installation

- Python v3.7 (can be installed via link)
- VirtualBox (can be installed via link)
- Python packages required for final deliverable :

```
pip install scikitlearn
pip install numpy
```

Packages required for proof of concept programs:

Proof_of_Concept_Programs/Basic_Visualisation/

- Testing_Different_Libraries/* :

```
pip install arcade
pip install pygame
```

- Visualisation_POC :

```
pip install arcade
```

Proof_of_Concept_Programs/Mouse_Tracking/

- mouse_coords_tracking.py :

```
pip install win32api
pip install numpy
```

- mouse_movement_auto_test.py :

```
pip install autopy
```

- ml_classifiers_comparison.py :

```
pip install scikitlearn
pip install numpy
```

Figure 2.1: README First Page

Running of programs

Final Deliverable

The final deliverable can be located in the folder Gotta_Evade_em_All.

In order to run the full version (including virtual machines), run the command:

```
python main.py
```

Otherwise, if you wish to run the lite version, which does not use the virtual machines, run the command:

```
python main.py lite
```

Proof Of Concept Program

Agents/

- ams_agent.py:

  This agent must be run within the Ubuntu VM, ensuring that the file 'agent.sh' is present in the same directory. This file should automatically run, as it is in the startup applications folder.

  ```
  python ams_agent.py
  ```

- host_agent.py:

  This agent requires the virtualbox to be setup, as well as the VBoxManage commands added to the system path.

  ```
  python host_agent.py
  ```

Proof_of_Concept_Programs/AMS_Report_Maniplulation_Scripts/

- MainScriptML: This script is the main script for generating the training data for the machine learning algorithm. It is dependent on the code in the folder MachineLearningScripts.

  ```
  python MainScriptML.py
  ```

- MainScriptReports: This script generates the signature data for the reports. It is dependent on the code in the folder Signature.

  ```
  python MainScriptReports.py
  ```

- MainScriptScores: This script generates the score data for the reports. It is dependent on the code in the folder ScoresScraper.

  ```
  python MainScriptScores.py
  ```

- MainScriptSigComp: This script generates the comparison signature data for the reports. It is dependent on the code in the folder Signature.

  ```
  python MainScriptSigComp.py
  ```

> Note: All of the generated result files are stored in the directory 'Results'.

Proof_of_Concept_Programs/Basic_Visualisation/

- Testing_Different_Libraries/Arcade :

  ```
  python POC_arcade.py
  ```

  ```
  python arcade_tutorial.py
  ```

Figure 2.2: README Second Page

- Testing_Different_Libraries/Pygame :

```
python pygame_tutorial.py
```

- Visualisation_POC :

```
python visualisationPoc.py
```

Categories_Filter/

- filter.py :

```
python filter.py
```

Final_Visualisation/

- visualisationPoc.py :

```
python visualisationPoc.py
```

Machine_Learning_Algo/

- ml_classifiers_comparison.py

```
 python ml_classifiers_comparison.py
```

Proof_of_Concept_Programs/Mouse_Tracking/

> It is recommended to run these programs in separate command prompts, executing the mouse_movement_auto_test.py first and then mouse_coords_tracking.py

- mouse_coords_tracking.py :

```
python mouse_coords_tracking.py
```

> The file cursor.csv contains all of the recorded mouse co-ordinates.

- mouse_movement_auto_test.py :

```
python mouse_movement_auto_test.py
```

Proof_of_Concept_Programs/Malware_Manipulation/

- malware_manipulation_poc.py :

```
python malware_manipulation_poc.py
```

Parser_Engine/

- parser_engine.py :

```
python parser_engine.py
```

Figure 2.3: README Third Page