

Configuration Manual

MSc Research Project Artificial Intelligence

Vikas Varma Malipeddi Student ID: 22143335

School of Computing National College of Ireland

Supervisor: Dr.Anh Duong Trinh (Senja)

National College of Ireland



MSc Project Submission Sheet

Schoo	lof	Com	nutina
30100	101	COIII	puting

Student Name:	Vikas Varma Malipeddi		
Student ID:	22143335		
Programme:	MSc in Artificial Intelligence	Year:	2023
Module:	MSc Research Method		
Lecturer:	Dr Anh Duong Trinh (Senja)		
Date:	31/01/2024		
Project Title:	Optimizing Adversarial Attacks on ML-Powered Malware Detection Systems		

Word Count:	1009	Page Count:	12
-------------	------	-------------	----

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

<u>ALL</u> internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: Vikas Varma Malipeddi

Date: 31/01/2024

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual: Optimizing Adversarial Attacks on ML-Powered Malware Detection Systems

Vikas Varma Malipeddi Student ID: 22143335

1. Introduction

This manual provides detailed instructions for setting up and executing code related to the implementation of query-efficient adversarial attacks against machine learning models. The focus is on understanding and enhancing the robustness of machine learning models against adversarial attacks. The following sections guide you through the necessary configurations, requirements, and tools.

2. System Specification

The adversarial attack system has been developed on the following hardware configurations:

File Edit View Help			
System Summary	Item	Value	
Hardware Resources	OS Name	Microsoft Windows 11 Home Single Language	
Components	Version	10.0.22621 Build 22621	
Software Environment	Other OS Description	Not Available	
	OS Manufacturer	Microsoft Corporation	
	System Name	VIKASMALIPEDDI	
	System Manufacturer	HP	
	System Model	HP Pavilion Gaming Laptop 15-ec2xxx	
	System Type	x64-based PC	
	System SKU	552W3PA#ACJ	
	Processor	AMD Ryzen 7 5800H with Radeon Graphics, 3201 Mhz, 8 Core(s), 16 Logical	
	BIOS Version/Date	AMI F.24, 22-02-2023	
	SMBIOS Version	3.3	
	Embedded Controller Version	96.34	
	BIOS Mode	UEFI	
	BaseBoard Manufacturer	HP	
	BaseBoard Product	88DE	
	BaseBoard Version	96.34	
	Platform Role	Mobile	
	Secure Boot State	On	
	PCR7 Configuration	Elevation Required to View	
	Windows Directory	C:\WINDOWS	
	System Directory	C:\WINDOWS\system32	
	Boot Device	\Device\HarddiskVolume1	
	Locale	United States	
	Hardware Abstraction Layer	Version = "10.0.22621.2506"	
	User Name	VIKASMALIPEDDI\vikas	

- Processor: Ryzen 7 5000 series
- Operating System: Windows 11
- Ram: 16 GB (DDR4)
- Storage Hard Drive: 1TB (SSD)

3. Software Used:

The following tools are required for the development and usage of the query-efficient adversarial attack system Pycharm Application below are the imported libraries to the required models to run:

- Torch
- TensorFlow and Keras
- Pandas
- NumPy
- Matplotlib
- Scikit-learn.

4. Installation of the Software:

Python Installation:

- Download and install Python 3.x from the official website: <u>Python</u>.
- Ensure that Python is added to the system PATH during installation.

Pycharm Installation:

Step 1: To download PyCharm, visit the official website of JetBrains: Download PyCharm

Step 2: After downloading the file, click on it

Step 3: When the following window appears, click on Next and the installation process will start



Step 3: After clicking on Next, first, a window for setting up the installation location will appear.

Note: You can either select a folder for the installation location or retain the default path.

😫 PyCharm Communit	y Edition Setup	
PC	Choose Install Location Choose the folder in which to	o install PyCharm Community Edition.
Setup will install PyCha folder, dick Browse and	rm Community Edition in the followin d select another folder. Click Next to	g folder. To install in a different continue.
Destination Folder	Brains\PyCharm Community Edition	2018.3.3 Browse
Space available: 196.8	GB	
	< Back	Next > Cancel

Step 4: In the next step, you can set the Installation Options as per requirements, and then, click on the Next button to proceed.

PyCharm Community Edition Setup	
Configure your Py	ons Charm Community Edition installation
Create Desktop Shortcut 32-bit launcher Image: 64-bit launcher Update context menu Image: Add "Open Folder as Project" Create Associations Image: py	Update PATH variable (restart needed)
Download and install JRE x86 by JetBrains	
(< Back Next >

Step 5: Now, you have to select the Start Menu folder, or you can leave it as default

yCharm Communi	y Edition Setup	
PC	Choose Start Menu Folder Choose a Start Menu folder for shortcuts.	r the PyCharm Community Edition
Select the Start Menu an also enter a name	folder in which you would like to creat to create a new folder.	e the program's shortcuts. You
7-Zip Accessories Administrative Tools Anaconda3 (64-bit) Android SDK Tools Android Studio Bluefish Brother		
Bullzip calibre 64bit - E-book CamStudio 2.7 Capture NX-D	Management	Ŧ
Brother Bullzip calibre 64bit - E-book CamStudio 2.7 Capture NX-D	Management Sack	Install Install Cancel

Step 6: After these steps, click on the Install button as above to start the installation process.

🖺 PyCharm Community	Edition Setup			
	Installing Please wait while	PyCharm Commun	ity Edition is	being installed.
Extract: commons-lang-2	.6.jar			
Show details				
		< Back	Next >	Cancel

Step 7: When you click on the Finish button, your PyCharm installation completes



Now, you have successfully installed PyCharm and Python both in your system.

Virtual Environment Setup:

- Create a new virtual environment for the application.
- Activate the virtual environment and install the required packages using pip.

5. Source Code and Models

Obtain the source code for query-efficient adversarial attacks against machine learning models. The repository may include pre-trained models and scenario scripts. Found on relevant repositories on platforms like GitHub.

6. Code Execution

Open Pycharm and then Python scripts to develop and execute the code. The workflow includes:

Execution Steps:

• Preprocess the Dataset File



• Perform the Prediction through the scenario 1

Scenario 1: Shared Training Data:

• In this scenario, both the target detection model and the surrogate model have access to the identical training dataset. They are trained on the same set of data samples, allowing for a direct comparison of their performance and vulnerability to adversarial attacks.

	E scenario-1.py \vee 🔋 main 🗸	Current File ∨ ▷ 🕸 🗄 🕹 Q 🕼	<i>ه</i> – و	
Projec	ct Files 🗸	📌 one.py × 👼 scenario-2.py 👘 scenario-3.py 👘 scenario-4.py 👘 scenario-5.py	:	Ċ
-∼ 80 ⊗ 	 vikas vikas aloha.py lightgom.py malconv.py one.py output.csv.py opading.py scenario-1.py scenario-2.py scenario-2.py scenario-3.py scenario-5.py scenario-5.py scenario-5.py scenario-5.py scenario-5.py scenario-5.py scenario-5.py stergate_model.py target_model.py target_model.py 	<pre>1 import numpy as np 2 import pandas as pd 3 import tensorflow as tf 4 from tensorflow as tf 5 from sklearn.neodel.selection import train_test_split 6 from sklearn.neodecessing import LabelEncoder 7 8 csv_path = r^C:\Users\Vikas\PycharmProjects\800MAS\evasion-attacks-against-ml-based-malware-detectors\data\benign_exampl 9 df = pd.read_csv(csv_path) 10 11 texts = df.iloc[:, 0].tolist() 12 labels = df.iloc[:, 1].tolist() 13 14 label_encoder = labelEncoder() 15 encoded_labels = label_encoder.fit_transform(labels) 16 17 18 csv_path = label_encoder.fit_transform(labels) 16 csv_path = label_encoder.fit_transform(labels) 17 csv_path = label_encoder.fit_transform(labels) 18 csv_path = label_encoder.fit_transform(labels) 19 csv_path = label_encoder.fit_transform(labels) 10 csv_path = label_encoder.fit_transform(labels) 11 csv_path = label_encoder.fit_transform(labels) 12 csv_path = label_encoder.fit_transform(labels) 13 csv_path = label_encoder.fit_transform(labels) 14 csv_path = label_encoder.fit_transform(labels) 15 csv_path = label_encoder.fit_transform(label_encoder.fit_transform(label_encoder.fit_transform(label_encoder.fit_transform(label_encoder.fit_transform(label_encoder.f</pre>	4 ▲ 13 ⊻ 3 ∧ - es\file.csv'	
 × ↓ ↓	●, ♥, Ċ +, ▷ ℑ = Python 2/2 - Ĉ Finished 2/2 - ● one 2/2 - Image: State of the stat	!: 0s 32ms/step - accuracy: 1.0000 - loss: 0.0000e+00 - val_accuracy: 1.0000 - val_loss: 0.0000e+00 /5 0s 27ms/step - accuracy: 1.0000 - loss: 0.0000e+00 - val_accuracy: 1.0000 - val_loss: 0.0000e+00 /5 0s 27ms/step - accuracy: 1.0000 - loss: 0.0000e+00 - val_accuracy: 1.0000 - val_loss: 0.0000e+00 /5 0s 27ms/step - accuracy: 1.0000 - loss: 0.0000e+00 - val_accuracy: 1.0000 - val_loss: 0.0000e+00 slvikas/PycharmProjects/B00M45/evasion-attacks-against-ml-based-malware-detectors/strc/vikas/one.py:62: SyntaxWarning: In loss ca * tf.keras.losses.categorical_crossentropy(y, predictions) 0s 127ms/step - accuracy: 1.0000 - loss: 0.0000e+00 finished with exit code 0 ts > src > vikas > @onepy 11 CRLF_UTF-8_4 spaces	tegorical_c	

• Perform the Prediction through scenario 2.

Scenario 2: Partially Shared Training Data:

• In this scenario, the target detection model and the surrogate model share only a portion of their training data. While some data samples are common between the two models, they also have distinct training data subsets. This introduces a degree of similarity and divergence in their training experiences.

	≣ 🖪 scenario-1.py ∨ 🦻 main ∨	Current File - ▷ 🕸 : 옫 Q 🔞 - 이 >	×
	Project Files \lor	🔷 one.py 🛛 🔶 scenario-2.py × 🚔 scenario-3.py 🛛 👼 scenario-4.py 🚔 scenario-5.py 🗄	Ļ
	 Vikas vikas aloha py lightgbm.py malconv.py one.py output.csv.py padding.py scenario-1.py scenario-2.py scenario-4.py scenario-4.py scenario-4.py scenario-4.py starget_model.py 	<pre>1 import numpy as np 2 import numpy as np 3 import numpy as np 4 frant ensorflow as tf 4 frant ensorflow import kereas 5 frant sklearn.model_selection import tabelEncoder 7 csv_path = r'C:\Users\vikas\PychamProjects\BODMAS\evasion-attacks-against-ml-based-malware-detectors\data\benign_examples\file.csv' 8 df = pd.read_csv(csv_path) 1 taxts = df.lloc[:, 0].tolist() 1 labels = df.lloc[:, 0].tolist() 1 labels = df.lloc[:, 1].tolist() 1 labels = iabel_encoder .fit_transform(labels) 3 texts_train, texts_test, labels_train, labels_test = train_test_split(*arrays: texts, encoded_labels, test_size=0.2, random_state=42) 1 num_classes = len(np.unique(encoded_labels))</pre>	
4 4 9		0s 45ms/step - accuracy: 1.0000 - loss: 0.0000++00 - val_accuracy: 1.0000 - val_loss: 0.0000++00	

• Perform the Prediction through scenario 3.

Scenario 3: Non-Shared Training Data:

Here, the target detection model and the surrogate model do not share any training data. They are trained independently on entirely separate datasets. This scenario assesses the transferability of adversarial attacks between models that have no common training ground.

	📃 🧵 scenario-1.py 🗸 🕫 main 🗸	Current File - D 🕆 🗄 ዿ Q 💰 - 🖉	×
	Project Files \lor	ne.py 🕏 scenario-2.py × network scenario-3.py × network scenario-4.py network scenario-5.py	Ĺ
+ ≈ D ::	 vikas vikas aloha py Bightgbm py malconv.py one.py output.csv.py pading.py scenario-1.py scenario-2.py scenario-3.py scenario-3.py scenario-3.py scenario-3.py scenario-3.py scenario-5.py scenario-5.py scenario-5.py surgat_model.py Wikas tin 	<pre>1 import numpy as np 2 import pandes as pd 3 import tensorflow as tf 4 from tensorflow as tf 5 from sklearn.neperocessing import train_test_split 6 from sklearn.neperocessing import labelEncoder 7 8 # Load your CSV files for target and surrogate models 6 csv_path_target = n'C:\Users\vikas\PycharmProjects\BODMAS\evasion-attacks-against-ml-based-malware-detectors\data\ballcious_examples\file 1 csv_path_target = n'C:\Users\vikas\PycharmProjects\BODMAS\evasion-attacks-against-ml-based-malware-detectors\data\ballcious_examples\file 1 df_target = pd.read_csv(csv_path_target) 1 df_surrogate = pd.read_csv(csv_path_surrogate) 1 fexts_terget = df_target.iloc[:, 0].tolist()</pre>	• 55
	Services		
4 9 9 9 9	 ♦ Python ♥ C Finished 1/1 - Epoch 1/1 - Envision Envision 1/1 - Process Ioss 1/1 - Process Envision Envision<	i: 0s 48ms/step - accuracy: 1.0000 - loss: 0.0000e+00 - val_accuracy: 1.0000 - val_loss: 0.0000e+00 4/5 0s 45ms/step - accuracy: 1.0000 - loss: 0.0000e+00 - val_accuracy: 1.0000 - val_loss: 0.0000e+00 5/5 0s 47ms/step - accuracy: 1.0000 - loss: 0.0000e+00 - val_accuracy: 1.0000 - val_loss: 0.0000e+00 rslvikas/PychamProiects/B00MAS/evasion_attacks-against-al-based-alkane-detectors/src/vikas/scenario-3.py:83: SyntaxWarning: In loss categor = tf.keras.losses.categorical_crossentropy(y, predictions) 0s 120ms/step - accuracy: 1.0000 - loss: 0.0000e+00 s finished with exit code 0	

• Perform the Prediction through the scenario 4

Scenario 4: Identical Model Architectures:

• In this scenario, both the target detection model and the surrogate model have the same architectural design. They share the same model structure, making it a direct architecture-to-architecture comparison.

	📃 🔳 scenario-1.py 🗸 🤔 main 🗸	Current File - ▷ : 온, Q 🔞 - @ ×
	Project Files ${\scriptstyle \lor}$	📌 one.py 🔹 scenario-2.py 🔹 scenario-3.py 🔹 scenario-4.py × 🔹 scenario-5.py 🗄
↔	 ∨ vikas ∨ vikas ≥ vikas ≥ aloha.py ≥ lightgbm.py ≥ malconv.py ≥ one.py ⇒ output.csv.py ⇒ output.csv.py ⇒ output.csv.py ⇒ scenario-1.py ⇒ scenario-2.py ⇒ scenario-2.py ⇒ scenario-3.py ⇒ scenario-4.py ⇒ scenario-5.py ⇒ scenario-5.py ⇒ scenario-4.py ⇒ scenario-4.p	<pre>1 import numpy as np 2 import numpy as np 2 import numsy as np 3 import tensorflow as tf 4 from tensorflow import keras 5 from sklearn.neprocessing import LabelEncoder 7 s Load your CSV files for target and surrogate models 6 csv_path_target = n^C:\Users\vikas\PycharmProjects\BODMAS\evasion-attacks-against-ml-based-malware-detectors\data\mallcious_examples\file. 10 csv_path_surrogate = n^C:\Users\vikas\PycharmProjects\BODMAS\evasion-attacks-against-ml-based-malware-detectors\data\mallcious_examples 11 df_target = pd.read_csv(csv_path_target) 12 df_target = pd.read_csv(csv_path_surrogate) 13 sf_surrogate = pd.read_csv(csv_path_surrogate) 14 f_target = dd_target.iloc[:, 0].tolist()</pre>
4) 4) 4) 4) 4) 4) 4) 4)	X ●, Y, C +, D 3 Y Python C Finished scenario-4 Scenario-4 J/1 C:\Users J/1 C:\Users J/1 Process asion-attacks-asainst-mi-based-matware-detector	<pre></pre>

• Perform the Prediction through scenario 5.

Scenario 5: Different Model Architectures:

• This scenario involves target and surrogate models with distinct architectural designs. Examples of these architectures include MalConv. The comparison explores the impact of varying model structures on adversarial attack transferability and effectiveness.

Project Files ∨ Import numpy as np		scenario-1.py 🗸 😢 main 🗸				🔲 : 옪오@	
-> > □ vikas 1 import numpy as np ● 1 ▲ 4 ▲ 16 ⊻ 4 ∧ 16 ≥ 16 ∧ 16 × 16 ∧ 16 × 16 × 16 × 16 × 16 ×	Project	Files ~	👌 one.py 🛛 🍦 scenario-2.py	🟓 scenario-3.py 🛛 🏓 scenario-	4.py 🏓 scenario-5.py 🗵		: Д
Image: scenario-3.py 12 Image: scenario-4.py 13 Image: scenario-5.py 14 Image: scenario-5.py 16 Image: scenari	Service	vikas vikas	<pre>1 import numpy as np 1 import pandes as pd 3 import pandes as pd 3 import transorflow as tf from tensorflow import keras 5 from skitearn.neprocessing import LabelEncoder 1 import metplotlib.pyplot as plt # Load your CSV files for target and surrogate models 1 csv_path_surrogate = n^C:\Users\vikas\PycharmProjects\BODMAS\evasion-attacks-against-ml-based-malware-detectors\data\benign_examples\fil 1 csv_path_surrogate = n^C:\Users\vikas\PycharmProjects\BODMAS\evasion-attacks-against-ml-based-malware-detectors\data\benign_examples\fil 1 csv_path_surrogate = n^C:\Users\vikas\PycharmProjects\BODMAS\evasion-attacks-against-ml-based-malware-detectors\data\ballcious_examples 1 df_target = pd.read_csv(csv_path_surrogate) 1 df_surrogate = pd.read_csv(csv_path_surrogate) 1 df_surr</pre>				
	0 X		3 :				
Python Python Renning 1/1	44 3 3 3 3 3 3 3 3 3 3 3 3 3	ython 1/1 P scenario-5 Epoch 3/1 Line Epoch 4/1 Line Epoch 5/1 Line Epoch 5/1 <	05 47ms/step /5 05 49ms/step /5 06 45ms/step /5 05 44ms/step	- accuracy: 1.0000 - loss: 0.000 - accuracy: 1.0000 - loss: 0.000 - accuracy: 1.0000 - loss: 0.000 - accuracy: 1.0000 - loss: 0.000 evasion_attacks_aaginst_ml_base rossentropy(v, predictions) - accuracy: 1.0000 - loss: 0.00	00+00 - val_accuracy: 1.0000 - val_loss: 0 100+00 - val_accuracy: 1.0000 - val_loss: 0 100+00 - val_accuracy: 1.0000 - val_loss: 0 300+00 - val_accuracy: 1.0000 - val_loss: 0 <u>1-malware-detectors\src\vikas\scenario-5.py</u> 3000+00	.0000+00 J.0000+00 J.0000+00 J.0000+00 <u>.522</u> : SyntaxWarning: In loss	↑ ↓ ?? * © © s catego



• Perform the Model Evaluation for all the methods.

This manual serves as a comprehensive guide for configuring the installation of the required software/tools for implementing query-efficient adversarial attacks against machine learning models.

References

- Python: (<u>https://www.python.org/</u>)

-Pycharm community available at Download PyCharm: Python IDE for Professional Developers by JetBrains.

- TensorFlow: [TensorFlow Installation Guide](<u>https://www.tensorflow.org/install</u>)