National College of Ireland

Enhancing Robustness and Generalizability of Explainable AI in Cross-Platform Malware Detection

MSc Practicum
Cybersecurity

Yash Shelar
Student ID: 23205415

School of Computing
National College of Ireland

Supervisor: Vikas Sahni

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Yash Mahesh Shelar
Student ID:	23205415
Programme:	MSc Cybersecurity **Year:** 2024 - 2025
Module:	MSc Practicum
Supervisor:	Vikas Sahni
Submission Due Date:	11th August 2025
Project Title:	Enhancing Robustness and Generalizability of Explainable AI in Cross-Platform Malware Detection
Word Count:	906 **Page Count:** 6

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.
ALL 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: Yash Mahesh Shelar

Date: 10/08/2025

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

Yash Shelar
23205415

1. System Requirements and Environment

Hardware:

- Operating System: Windows 11
- Processor: Intel Core i5 9th Gen (4 cores)
- RAM: 8 GB
- All experiments and analysis were run on CPU; no GPU required or used.

Software and Libraries:

- Python 3.8.5
- Jupyter Notebook 6.4.10

Python packages:

- numpy 1.21.4
- pandas 1.3.5
- scikit-learn 1.0.2
- matplotlib 3.5.1

Install these packages using the following in the command prompt:

```
#pip install numpy==1.21.4 pandas==1.3.5 scikit-learn==1.0.2 matplotlib==3.5.1 notebook==6.4.10
```

Alternatively, open Anaconda/Miniconda prompt and create an isolated environment as follows:

```
conda create -n malware_env python=3.8.5
conda activate malware_env
conda install numpy=1.21.4 pandas=1.3.5 scikit-learn=1.0.2 matplotlib=3.5.1 notebook=6.4.10
```

2. Data Files and Structure

Place the following files in your working directory:

- X_exp3.npy (Windows feature matrix)
- y_exp3.npy (Windows label vector)
- cicmaldroid_2020.csv (Android feature and label matrix, with malware labels mapped to 1, benign to 0)

Note:
If X_exp3.npy or y_exp3.npy are missing, my code will automatically simulate similar-shaped Windows data for testing only. If cicmaldroid_2020.csv is absent, the program will raise an error and stop.

3. Data Preparation

Load all three files in the Jupyter Notebook using numpy and pandas:

```
import numpy as np
import pandas as pd

X_win = np.load('X_exp3.npy')
y_win = np.load('y_exp3.npy')
df_android = pd.read_csv('cicmaldroid_2020.csv')
# Map Android labels to binary if not already (malware=1, benign=0)
```

Feature Harmonization:

- The code will automatically pad or truncate features to ensure both Windows and Android datasets have matching dimensions before merging.

Scaling and Splitting:

- All features are standardized (zero mean, unit variance) using scikit-learn's StandardScaler.
- Data is split into train and test sets (70% train, 30% test) using stratified sampling to preserve label distribution.

4. Model Training and Testing

Detection Model:

- Train a HistGradientBoostingClassifier from scikit-learn (version 1.0.2) on the combined harmonized and normalized dataset using random_state=42.

Robustness Evaluation:

- For robustness analysis, apply uniform random perturbations (epsilon=0.3) to the test set features, and measure the drop in prediction accuracy.

5. Instance Explanation (LEMNA)

Implement Ensemble LEMNA Explainer:

- For each test sample, perform 5 independent runs.
- Each run generates 2000 local perturbations (random variations) of the features.
- Fit scikit-learn's Ridge regression on each set to estimate feature coefficients.

- Average coefficients across the 5 runs to produce final feature importance for each sample.

Explanation Consistency:

- Calculate average cosine similarity between explanation vectors (clean vs. adversarially perturbed samples) to quantify stability.

6. Output and Results

Output Metrics Printed to Notebook:

- Accuracy, Precision, Recall, F1-Score, AUC-ROC
- Inference efficiency (predictions per second)
- Robustness drop under adversarial perturbation
- Explanation consistency score
- Example plot: Feature importances for a randomly selected test sample.

All output and intermediate results are displayed directly in the Jupyter notebook cells.

7. Logging and Reproducibility

- All random seeds are set to 42 throughout.
- All data loading, preprocessing, training, and evaluation steps are in the notebook.
- Repeat any experiment by re-running the notebook cells with the files and versions specified above; results will be identical.

8. Troubleshooting

- If the required .npy or .csv files are missing, you will be notified in the notebook and Windows feature data will be simulated for demonstration, but Android experiments cannot proceed.
- If any library import fails, check and reinstall the exact versions using pip or conda as listed above.

4 References

"A comprehensive investigation into robust malware detection with explainable AI - ScienceDirect." Accessed: Aug. 03, 2025. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2772918424000389?via%3Dihub

W. Guo, D. Mu, J. Xu, P. Su, G. Wang, and X. Xing, "LEMNA: Explaining Deep Learning based Security Applications," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, Toronto Canada: ACM, Oct. 2018, pp. 364–379. doi: 10.1145/3243734.3243792.

“ricksant2003/MalwareDatasetVirusShareSant: Malware DataSet for Windows Platform containing 28617 labeled samples from VirusShare packages.” Accessed: Aug. 03, 2025. [Online]. Available: https://github.com/ricksant2003/MalwareDatasetVirusShareSant

“MalDroid 2020 | Datasets | Research | Canadian Institute for Cybersecurity | UNB.” Accessed: Aug. 03, 2025. [Online]. Available: https://www.unb.ca/cic/datasets/maldroid-2020.html