National College of Ireland

# Configuration Manual

MSc Research Project
Master of Science In Cyber Security

Anjal Muhammed Madathil Abdul Majeed
Student ID: 23398329

School of Computing
National College of Ireland

Supervisor: Kamil Mahajan

**National College of Ireland**

**MSc Project Submission Sheet**

**School of Computing**

**Student Name:** Anjal Muhammed Madathil Abdul Majeed ……

**Student ID:** 23398329……..……

**Programme:** Master of Science In Cyber Security **Year:** 2025……..

**Module:** Practicum ……..……

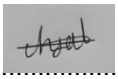
**Lecturer:** Kamil Mahajan ……..……..……

**Submission Due Date:** 28/01/2026……..……

**Project Title:** Intelligent Malware Detection Using CNN and OSINT ……..……

**Word Count:** 1151…… **Page Count:** 4……..……..……

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:**  ……

**Date:** 27/01/2026……..……

**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

Anjal Muhammed Madathil Abdul Majeed
Student ID: 23398329

# 1 Introduction

The "Machine Learning for Intelligent Malware Detection via CNN and OSINT" project, including directions on how to set it up, set up its technical environment, and launch it. It contains Hardware and Software Requirements, creating the Technical Environment, preparing Data for the Model, How to Train the Model, and the steps to take to launch the Web App. By following these steps, you will be able to replicate the experimental research and apply the results of your research to your own system and reproducibly test your application.

# 2 System Requirements

- **CPU:** x86-64 Processor (e.g., Intel Core i5/AMD Ryzen 5 or newer).
- **RAM:** 8 GB minimum, 16 GB or more recommended.
- **Storage:** 5 GB of free disk space for the project files, dataset, and generated models.
- **GPU (Highly Recommended for Training):** NVIDIA GPU with CUDA support (e.g., GTX 1060, RTX 2060 or newer) with at least 6 GB of VRAM. Training on a CPU is possible but will be extremely slow.
- **Operating System:** Windows 10/11, macOS, or a modern Linux distribution (e.g., Ubuntu 20.04+).
- **Python:** Version 3.8 or higher.
- **Package Manager:** pip (included with Python) or conda.
- **Version Control:** Git command-line tool.
- **NVIDIA Drivers (If using GPU):** Latest proprietary NVIDIA drivers.
- **CUDA Toolkit (If using GPU):** Version 11.x or newer, compatible with the installed NVIDIA driver and PyTorch version.

# 3 Project Setup and Installation

This section outlines the step-by-step process to prepare the project environment.
Open a terminal or command prompt and clone the project repository from its source using Git.
git clone <your-repository-url>
cd intelligent-malware-detection
It is strongly recommended to use a virtual environment to manage project dependencies and avoid conflicts with other Python installations.

## 3.1 Option A: Using venv (standard with Python)

```
# Create the virtual environment
python -m venv venv

# Activate the environment
# On Windows:
venv\Scripts\activate
# On macOS/Linux:
```

source venv/bin/activate

## 3.2 Option B: Using conda

# Create the conda environment
conda create --name malware_env python=3.9 -y

# Activate the environment
conda activate malware_env
With the virtual environment activated, install all the necessary Python libraries listed in the requirements.txt file.

pip install -r requirements.txt
**Note on PyTorch with CUDA:** If your system has a compatible NVIDIA GPU, it is crucial to install the CUDA-enabled version of PyTorch. The command in requirements.txt may install the CPU-only version. It is recommended to install it manually by visiting the Get Started to get the correct installation command for your specific CUDA version. For example:

# Uninstall CPU-only PyTorch if installed by mistake
pip uninstall torch

# Example command for CUDA 11.8
pip install torch torchvision torchaudio --index-url https://download.pytorch.org/whl/cu118

1. Download the Malimg dataset from the provided Kaggle link: https://www.kaggle.com/datasets/manmandes/malimg
2. Extract the downloaded archive. You should have a folder named malimg_dataset.
3. Create a dataset directory in the root of the project folder.
4. Move the extracted malimg_dataset folder inside the dataset directory and rename it to malimg.
5. The final directory structure for the data must be as follows:

```
intelligent-malware-detection/
├── dataset/
│   └── malimg/
│       ├── train/
│       │   ├── Adialer.C/
│       │   │   ├── 000c406b225645d9016e4268e3f9a42f.png
│       │   │   └── ...
│       │   └── ... (24 other family folders)
│       ├── val/
│       │   ├── Adialer.C/
│       │   └── ...
│       └── test/
│           ├── Adialer.C/
│           └── ...
└── ... (other project files)
```

**This structure is critical for the data loaders in the code to function correctly.**

# 4 Running the Experiments (Model Training)

The experiments, including data preprocessing, model training, and evaluation, are contained within a Jupyter Notebook.

1. Ensure your virtual environment is activated.

2. Launch Jupyter Notebook or Jupyter Lab from the project's root directory:

```
jupyter lab
or
jupyter notebook
```

3. Your web browser will open with the Jupyter interface.
4. Navigate to and open the Untitled.ipynb notebook.
5. To replicate the results, execute the cells sequentially from top to bottom. You can do this by selecting a cell and pressing Shift + Enter or by using the "Run" menu.
6. Upon successful execution, the following artifacts will be generated:
    - **Trained Models:** models/baseline_cnn.pth and models/osint_model.pth.
    - **Logs and Reports:** Files like logs/class_statistics.json, logs/model_comparison.csv, and various plots (.png files) will be saved in the logs/ directory.
    - **OSINT Metadata:** A simulated metadata file will be created at osint/osint_metadata.json.

# 5 Running the Web Application

After the models have been successfully trained and saved, you can launch the Flask web application to interact with them.

1. Make sure you are in the project's root directory and your virtual environment is activated.
2. Run the Flask application using the following command:

```
python app.py
```

3. You will see output in your terminal indicating that the server is running, typically on port 5000.

```
* Serving Flask app 'app'
 * Running on http://127.0.0.1:5000 (Press CTRL+C to quit)
```

4. Open a web browser and navigate to the following URL: http://127.0.0.1:5000/
5. The application interface will load. You can now:
    - Click "Choose File" to select a malware image from your local machine (you can use any image from the dataset/malimg/test folder).
    - Select either the "Baseline CNN" or "OSINT-Aware Model" from the dropdown.
    - Click "Analyze" to get a classification result.

# 6 Project Directory Structure

The final project directory should have the following structure after running the experiments:

```
intelligent-malware-detection/
├── dataset/
│   └── malimg/        # Populated in Step 3.4
├── models/
│   ├── baseline_cnn.pth
│   └── osint_model.pth
├── logs/
│   ├── baseline_confusion_matrix.png
│   ├── baseline_training_curves.png
│   ├── class_statistics.json
│   ├── model_comparison.csv
│   └── ...
├── osint/
│   └── osint_metadata.json
```

```
├── templates/
│   ├── index.html
│   └── about.html
├── uploads/            # Created by the Flask app for temporary storage
├── venv/               # Virtual environment folder
├── Untitled.ipynb      # Main Jupyter Notebook for experiments
├── app.py              # Flask application script
└── requirements.txt    # Python dependencies
```

# 7 Troubleshooting

- **Shared Solution to Problem**: ModuleNotFoundError when executing script files in a Python environment. The solution for this error is to confirm that your virtual environment has been activated and that all necessary packages listed in requirements.txt were installed successfully.
- **Shared Solution to Problem:** When using PyTorch, the library states that CUDA is not accessible, and therefore, training on the CPU is extremely slow. This issue typically occurs when you installed the CPU only version of PyTorch, rather than the GPU-enabled version meant for your system's version of the CUDA toolkit. To install the correct version, refer to Step 3.3.
- **Shared Solution to Problem:** When executing a Jupyter notebook, Python raises the FileNotFoundError. This problem usually is due to the dataset directory structure not matching the specified structure in Step 3.4 (e.g. dataset/malimg/train, dataset/malimg/val).
- **Shared Solution to Problem:** The Flask application fails to start because the model files do not exist. To ensure that the model files exist, make sure to run the Jupyter notebook (Untitled.ipynb) through to the end in order to create the models (baseline_cnn.pth and osint_model.pth) stored in the models/ directory.