**Technical Report**

**Team XYZ**

**Challenge: Malware Hunt**

**Team Members:**

* Wooyeon Jo
* Muhammad Ahsan
* Muhammad Shahryar Yaqub

**Project Deliverables:**

* Malware classifier code
* Automated report generator script
* PowerPoint presentation

**Problem Statement:**

Existing scanning tools often lack in-depth insights into malware infections beyond basic detection and quarantine functionalities. They typically offer limited information to end users, such as simply identifying infected files and providing options like deletion or quarantine.

To address the problem statement provided, let's elaborate on each point and incorporate technical details:

**Creating a Plugin for Enhanced Insights**

* Develop a plugin that integrates with existing scanning tools to provide advanced insights into malware infections.
* The plugin should be designed to analyze various aspects of malware behavior, including its propagation methods, system impact, and potential vulnerabilities exploited.
* Technical implementation involves developing modules for behavior analysis, code pattern recognition, and heuristic scanning techniques within the plugin.

**Strengthening the Existing System:**

* The plugin should augment the existing scanning tool by offering actionable insights and recommendations.
* It could provide detailed reports on the type and severity of malware detected, along with suggestions for remediation and proactive measures.
* Implement algorithms to prioritize threats based on their potential impact and likelihood of spreading within the system.

**User-Focused Experience with Insights:**

* Design the plugin interface to be intuitive and user-friendly, presenting insights in a clear and understandable manner.
* Incorporate visualizations, such as graphs and charts, to illustrate malware trends, infection patterns, and system vulnerabilities.
* Provide contextual explanations and educational resources to help users understand the nature of malware threats and make informed decisions.

**Methodology:**

**Malware Classifier:**

Convolution Neural Network was developed to classify malware. The malware was first transformed into an image file and trained on the CNN model. The complete framework is shown in the figure 1 and the steps are detailed below:

Step 1: Data Preprocessing

* Binary Converted to grayscale image file.
* Image fragmentation

Step 2: Model Configuration

* Conv2D layer with 32 filters is used to extract features from the images.
* MaxPooling2D layer reduces the size of the images.
* Flatten layer converts the 2D array to a 1D array.
* Dense layers are used to create the neural network, with 'relu' activation function.
* The output layer corresponds to the number of classes and uses the 'softmax' activation function to output probability values.

Step 3: Model Compilation

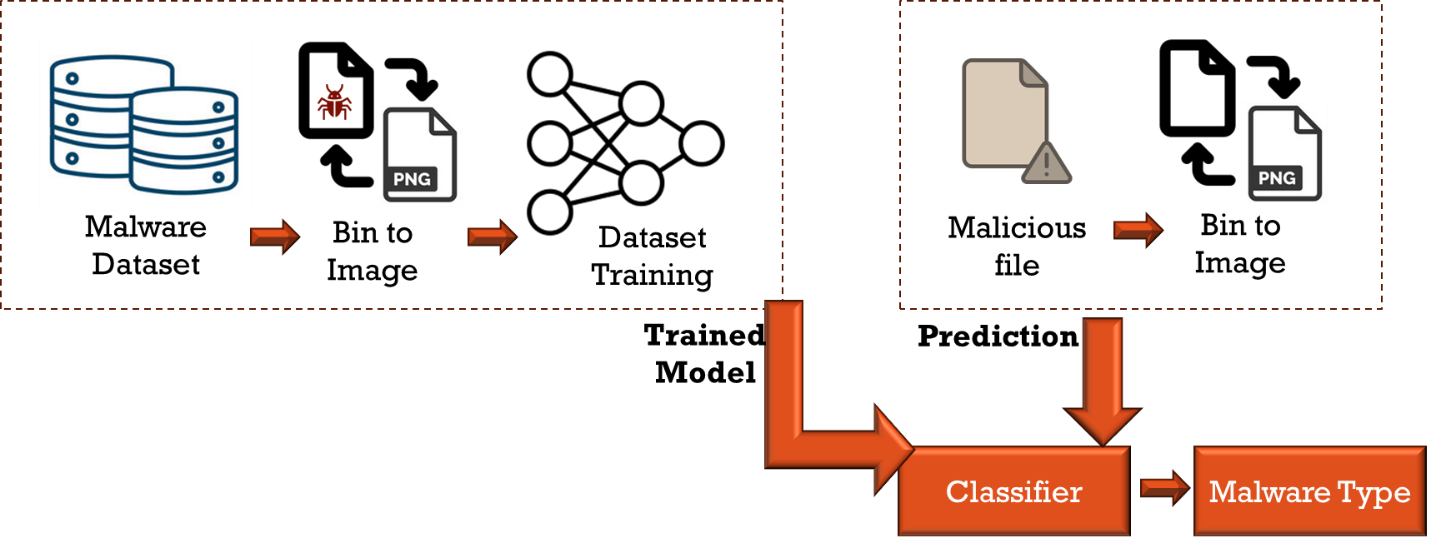
* The model is compiled using the 'adam' optimizer and 'sparse\_categorical\_crossentropy' loss function.

Step 4: Model Training

* Dataset – 5 labels [1,2]
* No of files – 100 +
* The model is trained using the data generated by ImageDataGenerator.
* The batch size is set to 64, and training is conducted for 50 epochs.

Model Evaluation:

* The model is evaluated using the test data (20% randomly chosen).
* Testing Accuracy > 80% (In Literature > 95% [3])

  
Figure 1: Malware Classifier Framework

**Code Block Analysis:**

The framework for CBA is shown in figure 2. Following steps were proposed to convert image into color coded image file using heuristics and machine learning algorithms:

* Decompile the binary file.
* Analyze using heuristics to identify code segments with different behavior.
* Label these segments and generate a dataset for each category.
* Test the incoming file for the presence of each malware category/
* For example, if there is malware with worm and keylogger capability the file would be automatically labeled by the trained ML model.
* Post-process the image with the predicted labels and color code it.

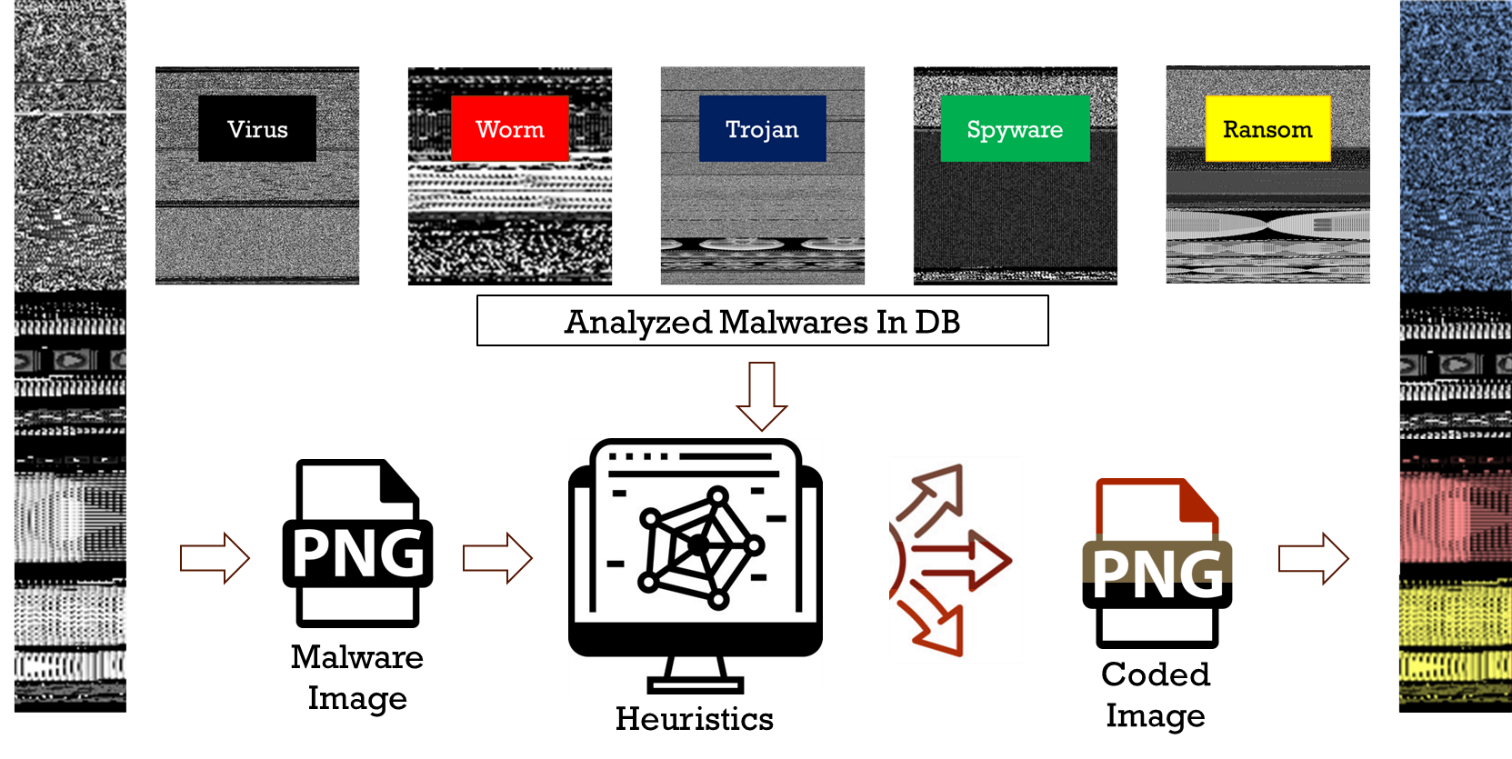


Figure 2: Code Block Analysis for color coded image generation

**Model Results:**

3 case studies

3-class classification on largest dataset

Worm, Trojan, Ransomware (389,050 image frags)

Valid Accuracy: 88.1%

5-class classification on typical malwares

Virus, Worm, Trojan, Ransomware, Spyware (417,322 frags)

Valid Accuracy: 82.1%

5-class classification including prankware type malwares

Trojan, Ransomware, Rogue, Jokes, Enderware (471,195 frags)

Valid Accuracy: 86.7%

**Technical Report:**

Following detailed were included in the report generated by our automated script

* CBA Generated Image:
* Detection Ratio: This is one of the most critical aspects of the report. It shows how many antivirus engines flagged the file as malicious out of the total number scanned. A higher detection ratio indicates a higher likelihood that the file is harmful.
* Individual Antivirus Results: The report lists the findings of each antivirus engine separately. It shows which engines detected the file as malicious and what type of malware or threat they identified it as (e.g., Trojan, Worm, Spyware).
* File Hashes: The report includes hashes of the file, such as MD5, SHA-1, and SHA-256. These are unique identifiers for the file, used to compare it against known malware samples or to search for the file in other databases.
* File Characteristics: This can include information like the file size, file type (e.g., executable, PDF, script), and sometimes more detailed metadata depending on the file format (like version information for executables).
* Historical Data: If the file has been analyzed by VirusTotal in the past, the report might include historical data showing how the detection ratio and other details have changed over time.
* Community Feedback: VirusTotal allows users to comment on files, so the report might include observations or insights from the community, which can be particularly useful for new or obscure files.
* Behavioural Analysis: For some files, particularly executables, the report may include a behavioral analysis showing what actions the file performs when executed in a sandbox environment. This can include network communications, file modifications, and registry changes.

**References:**

[1] <https://github.com/Endermanch/MalwareDatabase?tab=readme-ov-file>

[2] <https://github.com/cryptwareapps/Malware-Database>

[3] S. Tobiyama, Y. Yamaguchi, H. Shimada, T. Ikuse and T. Yagi, "Malware Detection with Deep Neural Network Using Process Behavior," 2016 IEEE 40th Annual Computer Software and Applications Conference (COMPSAC), Atlanta, GA, USA, 2016, pp. 577-582, doi: 10.1109/COMPSAC.2016.151.