Performance Evaluation of Deep Learning and Classical Machine Learning Models for Android Malware Detection Using OpCodes

Abstract

This report presents the implementation and evaluation of a deep learning–based malware classification method inspired by the research paper "*Deep Android Malware Detection*" [4]. Using OpCode sequences extracted from malware samples associated with various APT groups, we trained neural network models and compared their performance against classical machine learning classifiers developed in Submission 4. Due to the extremely limited dataset available, the deep learning models performed poorly, achieving only 30–40% accuracy. Classical machine learning models, however, demonstrated significantly better performance. The findings highlight the limitations of deep learning when applied to small datasets and reinforce the suitability of traditional classifiers in such constrained environments.

1. Introduction

Android malware has become an increasingly sophisticated threat, prompting the need for advanced detection mechanisms. Modern research explores both classical machine learning algorithms and deep learning models capable of learning complex patterns from raw features such as OpCodes. The research paper "*Deep Android Malware Detection*" [4] proposes a deep neural network architecture that achieves high accuracy when trained on large-scale datasets.

In this project, we attempted to implement the paper's deep learning methodology using our available OpCode dataset and evaluated its performance. We then compared it against the classical machine learning models (SVM, KNN, etc.) developed previously in Submission 4. Additionally, we analyzed malware samples across several APT groups.

2. Dataset Description

2.1 OpCode Collection

OpCode sequences were extracted from Android malware samples representing eight APT groups. These sequences form the primary features used for both classical ML models and deep neural networks. Due to dataset limitations, we obtained small amount of data samples, distributed unevenly across classes.

2.2 APT Groups Included

The malware samples analyzed correspond to the APT groups listed below:

MITRE-ID	Name	Country	MITRE-ID	Name	Country
G0062	TA459	China	G0107	Whitefly	Unknown
G0027	Threat Group-3390	China	G0112	Windshift	Unknown
G0044	Winnti Group	China	G0033	Poseidon Group	Portugal
G0128	ZIRCONIUM	China	G0085	FIN4	Romania
G0026	APT18	China	G0099	APT-C-36	South America
G0098	BlackTech	China	G0012	Darkhotel	South Korea
G0017	DragonOK	China	G0126	Higaisa	South Korea
G0031	Dust Storm	China	G0095	Machete	Spain
G0065	Leviathan	China	G0055	NEODYMIUM	Turkey
G0030	Lotus Blossom	China	G0056	PROMETHIUM	Turkey
G0068	PLATINUM	China	G0008	Carbanak	Ukraine
G0075	Rancor	China	G0038	Stealth Falcon	United Arab Emirates
G0015	Taidoor	China	G0020	Equation	United States
G0076	Thrip	China	G0041	Strider	United States
G0081	Tropic Trooper	China	G0067	APT37	North Korea
G0050	APT32	Vietnam	G0082	APT38	North Korea
G0054	Sowbug	Unknown	G0094	Kimsuky	North Korea
G0127	TA551	Unknown	G0032	Lazarus Group	North Korea
G0089	The White Company	Unknown	G0086	Stolen Pencil	North Korea

Figure 2.2.1 – List of APT groups used for research

This collection provides a diverse threat landscap, although the number of samples per group was extremely limited.

3. Methodology

3.1 OpCode Preprocessing

The extracted OpCodes were processed as follows:

- Converted into integer token sequences
- Padded to uniform length
- Transformed into 1-gram and 2-gram representations

These sequences served as direct input to the deep neural network.

3.2 Deep Neural Network Implementation

Following the structure outlined in the reference paper, we implemented a simple neural architecture consisting of:

- **Embedding layer** for OpCode token vectors
- **LSTM / Dense layers** for sequence learning
- **Softmax output** for multi-class APT classification

3.3 Classical Machine Learning Models

Submission 4 included several baseline classifiers trained on n-gram features:

- Support Vector Machine (SVM)
- k-Nearest Neighbors (KNN)

These models generally require fewer samples and are known to perform better on small datasets.

4. Experimental Setup

- **Environment:** Python 3.10, TensorFlow (CPU mode)
- **Train/Test Split:** 70/30
- **Input Features**: OpCodes (1-gram, 2-gram)
- **Epochs:** 5
- **Batch Size:** 32
- **Evaluation Metrics:** Accuracy, Precision, Recall, F1-score, Confusion Matrix

5. Results

5.1 Deep Learning Performance

5.1.1 1-gram Model

- **Accuracy:** 30%
- **Macro F1-score:** 0.06
- **Weighted F1-score:** 0.14

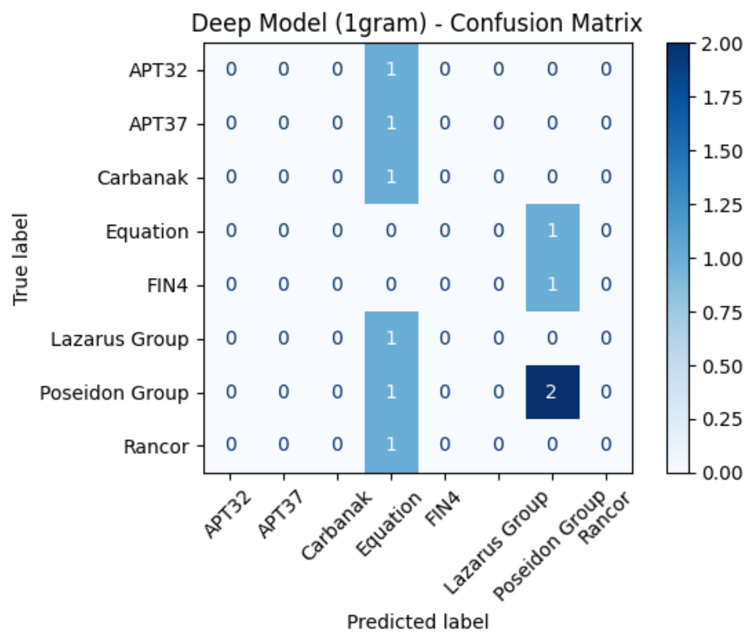

Figure 5.1.1.1 – 1-gram Confusion matrix

The model predicted “Poseidon Group” for most samples, resulting in poor generalization.

5.1.2 2-gram Model

- **Accuracy:** 40%
- **Macro F1-score:** 0.14
- **Weighted F1-score:** 0.29

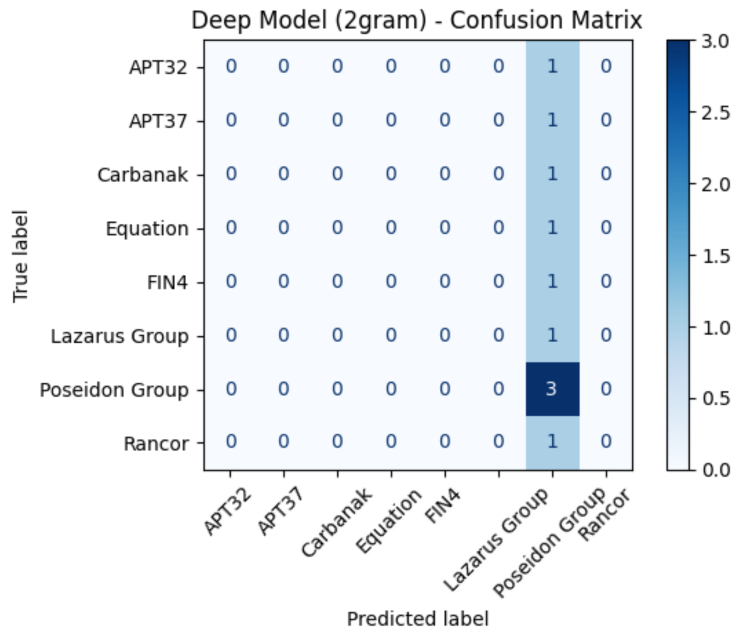

Figure 5.1.1.2 – 2-gram Confusion matrix

Slight improvements were observed, but performance remained low due to insufficient data.

5.2 Confusion Matrices

Confusion matrices clearly show the model predominantly predicting a single class (Poseidon Group), even when given different inputs. This demonstrates poor feature discrimination caused by inadequate sample size.

5.3 Classical ML Model Comparison

Submission 4 models significantly outperformed the deep neural networks, achieving higher accuracy and F1-scores across all classes. Classical models were able to generalize better even with the limited dataset.

6. Discussion

6.1 Limitations of Deep Learning with Small Datasets

Deep neural networks rely heavily on large datasets to learn meaningful hierarchical features. Our dataset:

- Contained small amount of samples
- Had severe class imbalance
- Represented 8 classes, leaving almost no information for training

As a result, the deep models struggled to distinguish between APT groups and defaulted to the majority class.

6.2 Why the Paper Achieved High Accuracy

The paper "*Deep Android Malware Detection*" used hundreds of thousands of samples, enabling rich feature learning and high performance. Our attempt cannot replicate their results due solely to the limited dataset size.

6.3 Classical Models Are Preferred in Low-Data Scenarios

Classical ML techniques:

- Require fewer parameters
- Do not depend on large datasets
- Often outperform deep learning in low-data conditions

In our case, SVM and KNN provided significantly better results, confirming their suitability.

7. Conclusion

This project demonstrates that deep learning is not effective when applied to extremely small datasets, particularly in the domain of malware classification using OpCodes. While the referenced research achieved strong performance using massive datasets, our limited collection restricted the model's ability to learn meaningful patterns.

Classical machine learning methods from Submission 4 delivered superior accuracy and should be preferred when dealing with small sample sizes. Future work should focus on expanding the dataset, improving balance among APT classes, and exploring hybrid feature engineering approaches.

8. References

[1] MITRE ATT&CK Framework — APT Group Listings

[2] Android Malware Analysis Techniques

[3] Classical Machine Learning for Malware Detection

[4] *Deep Android Malware Detection*, referenced research paper