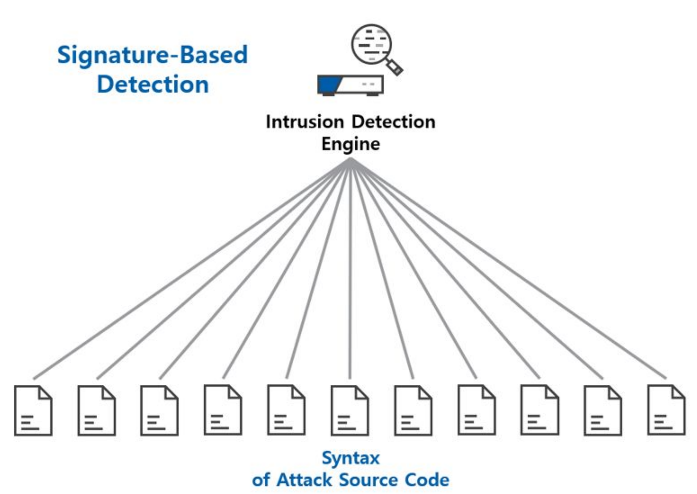
Image-Based Malware Classification Using Convolutional Neural Networks

Raymond Jiang, Ethan de la Cruz

Glen A. Wilson High School

Problem

Although existing technologies such as anti-viruses and signature-based detection have been successful in identifying and removing basic forms of malware, they commonly feature severe limitations that hinder their effectiveness with more complex forms of malware. Difficulties in dealing with techniques such as obfuscation, code injection, rootkit techniques, as well as the constant rise of new, more potent methods highlights the need for a more robust, encompassing classification and detection strategy.



McAfee

KASPERSKY lab

Avast

Approach

An alternative approach to solve such issues by using a Convolutional Neural Network to classify malware images based on their shared features. This is done by converting malware byte files into images, which are used as input features for the network. The network learns to classify each image into separate families, based on similar features. The Malimg dataset [1], a collection of over 9,000 unique malware byte images classified into 25 families based on shared features was used to train the model. This approach disregards the need to inspect source code, dramatically improving its efficiency and execution compared to past solutions, thus resulting in significant potential for accurately classifying and eventually eradicating malware.

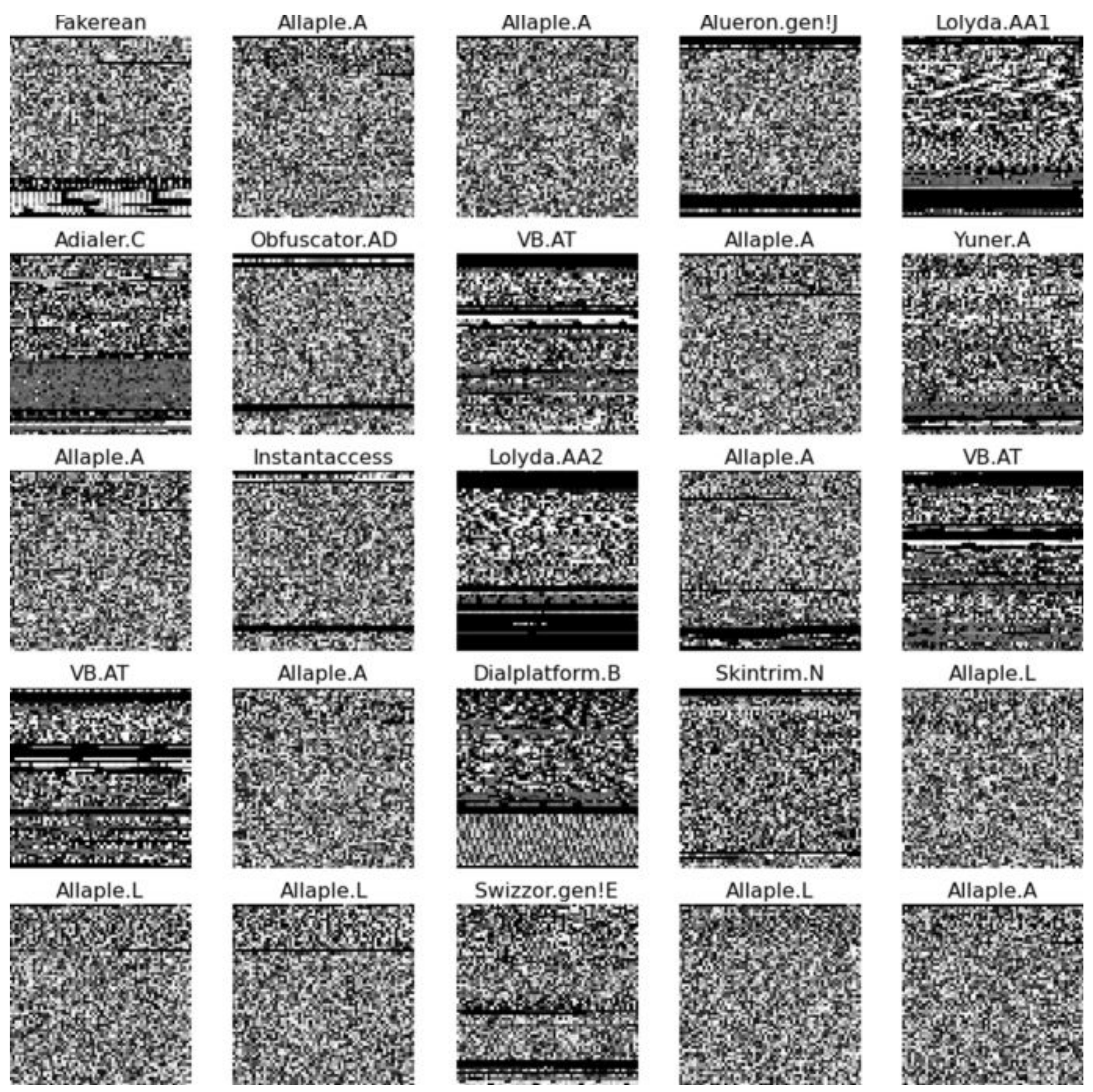
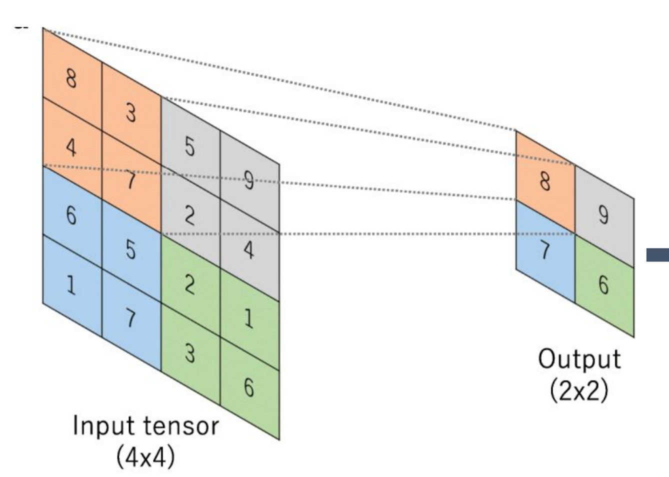


Figure 1: Sample pictures of the malImg dataset from each of the 25 different malware families

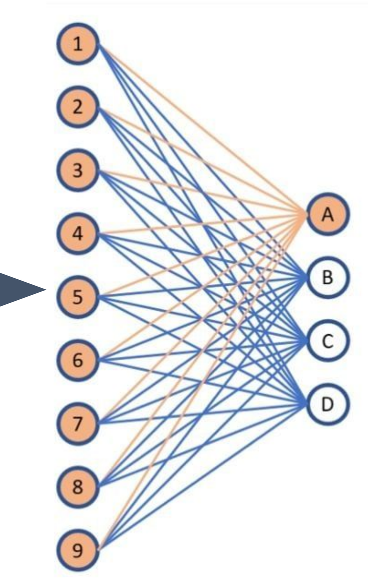
Model Architecture

Table 1: Model Layers/Settings

Layer Name	Layer Parameters
Input	(shape=(64, 64, 3))
Conv2D	(32, (5, 5), activation='relu')
MaxPooling2D	NA
BatchNormalization	NA
Conv2D	(64, (3, 3), activation='relu')
MaxPooling2D	NA
BatchNormalization	NA
Conv2D	(128, (3, 3), activation='relu')
MaxPooling2D	NA
BatchNormalization	NA
Conv2D	(256, (3, 3), activation='relu')
MaxPooling2D	NA
BatchNormalization	NA
Flatten	NA
Dense	(512, activation='relu')
Dropout	(0.5)
Dense	(25, activation='softmax')
model.compile	(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
model.fit	(X_train, Y_train, epochs=10, batch_size=64, validation_data=(X_test, Y_test))



Conv2D Layer Visualization



Dense Layer Visualization

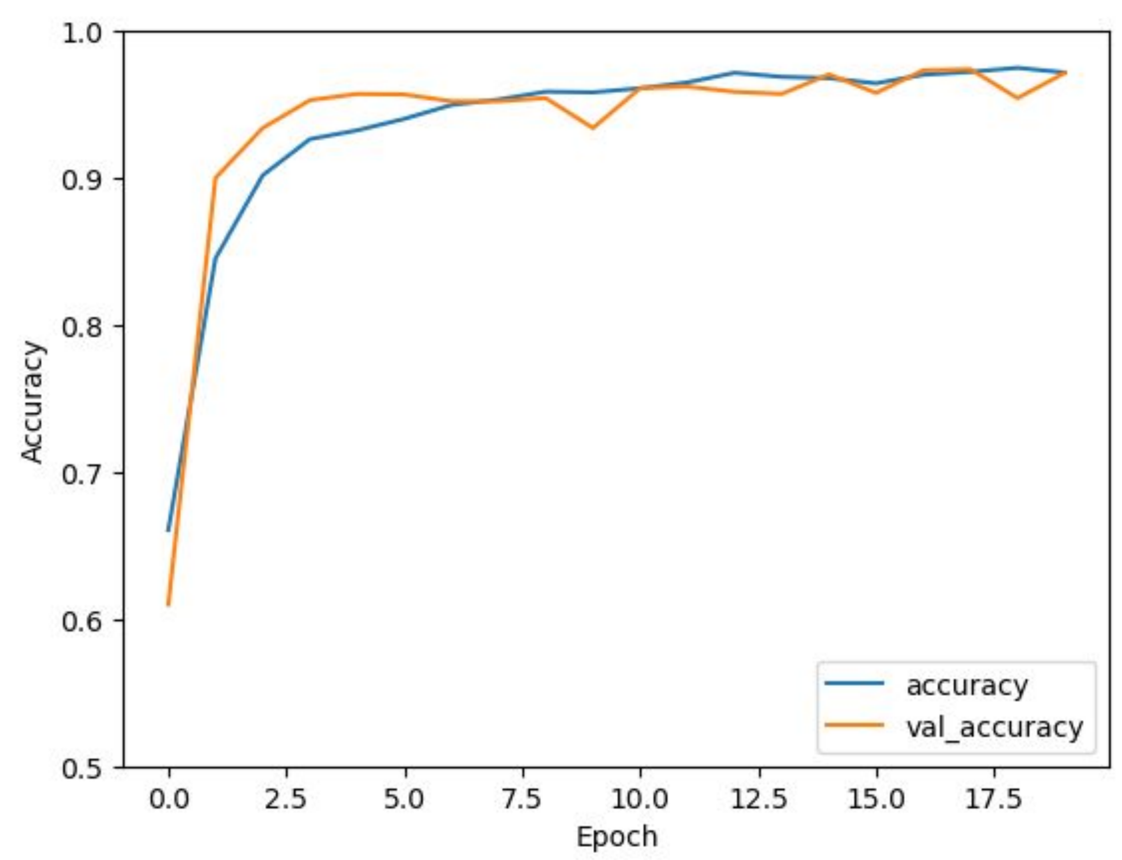


Figure 2: Accuracy v Epoch Graph

Challenges

While performing well overall, the model struggled significantly on classifying family 5, misidentifying every example image as family 24. After inspection, I noted that the two families shared extremely similar, redundant features, likely confusing the model.

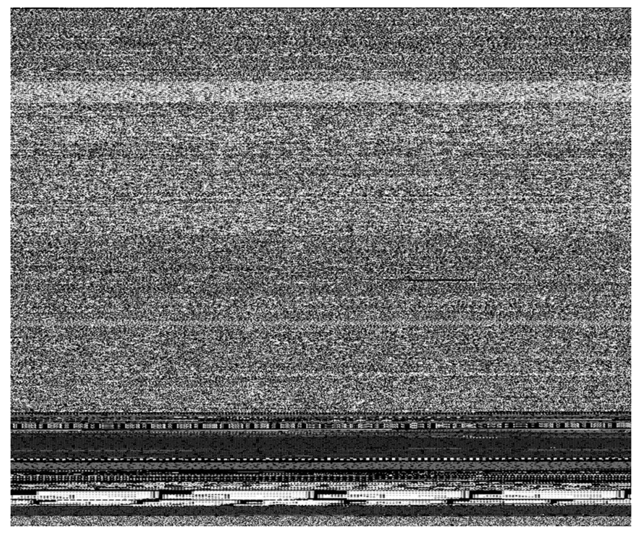


Figure 4: Malware sample Class 5

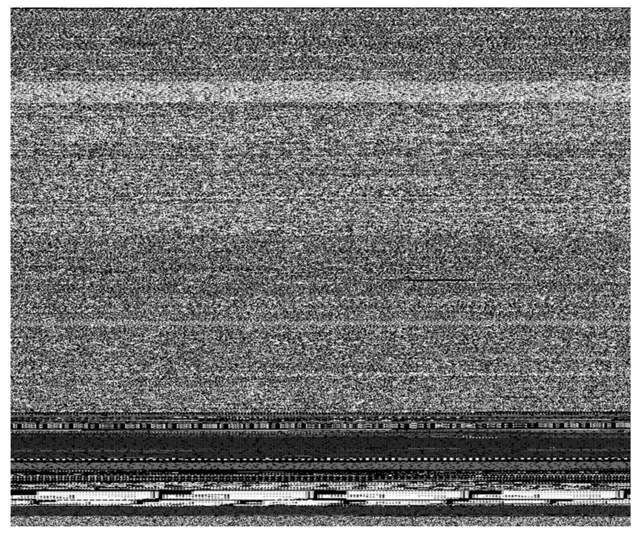


Figure 5: Malware sample Class 24

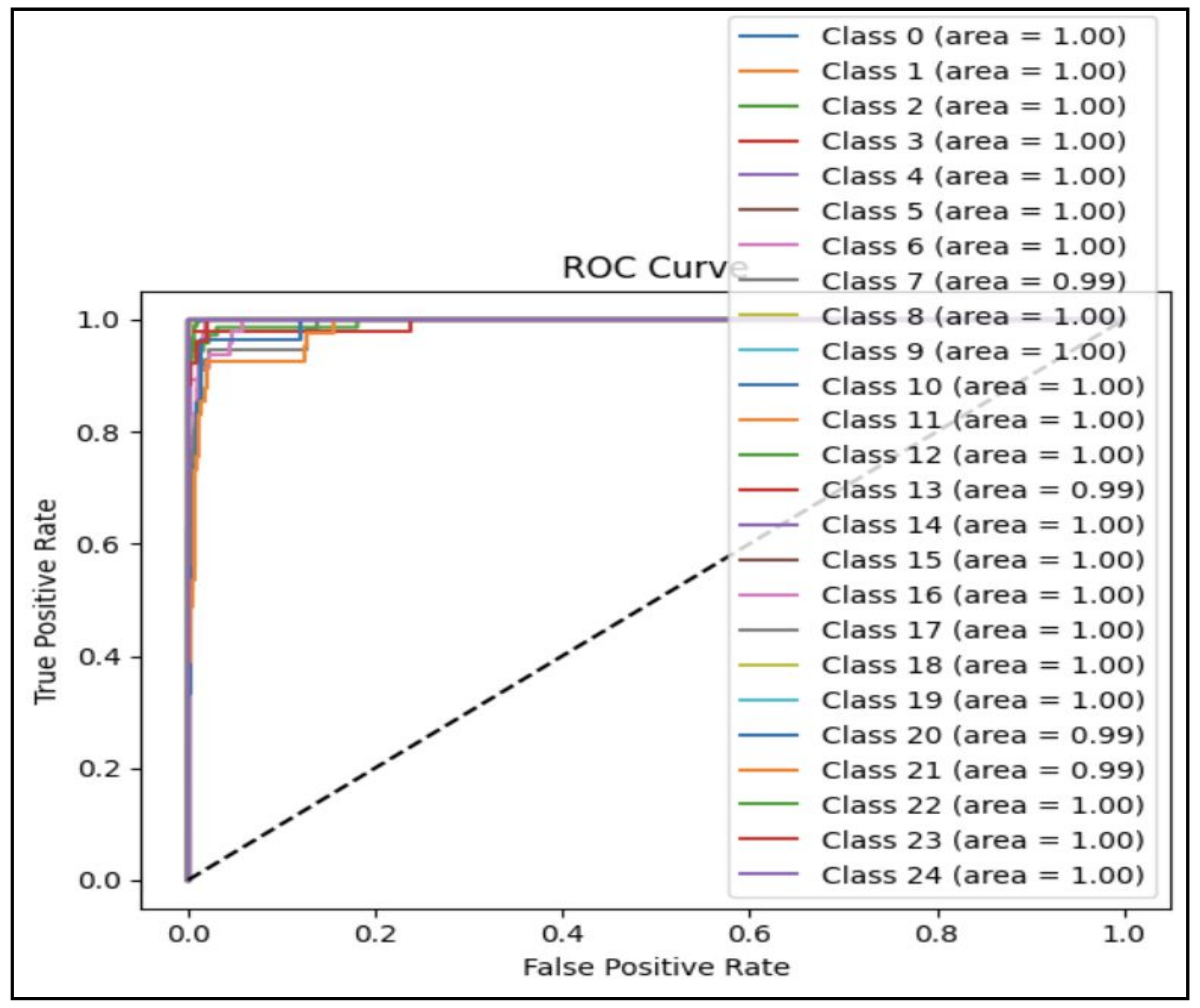


Figure 6: ROC Curve Graph

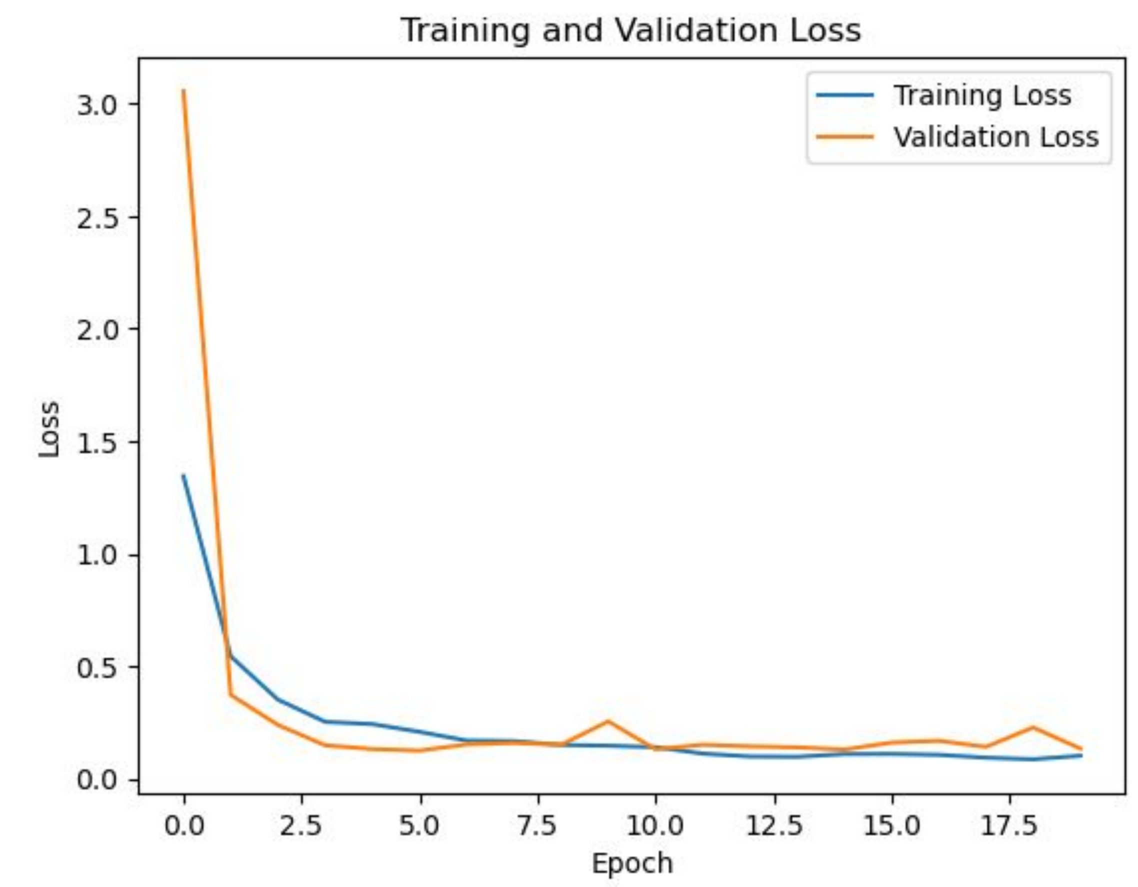


Figure 3: Loss v Epoch Graph

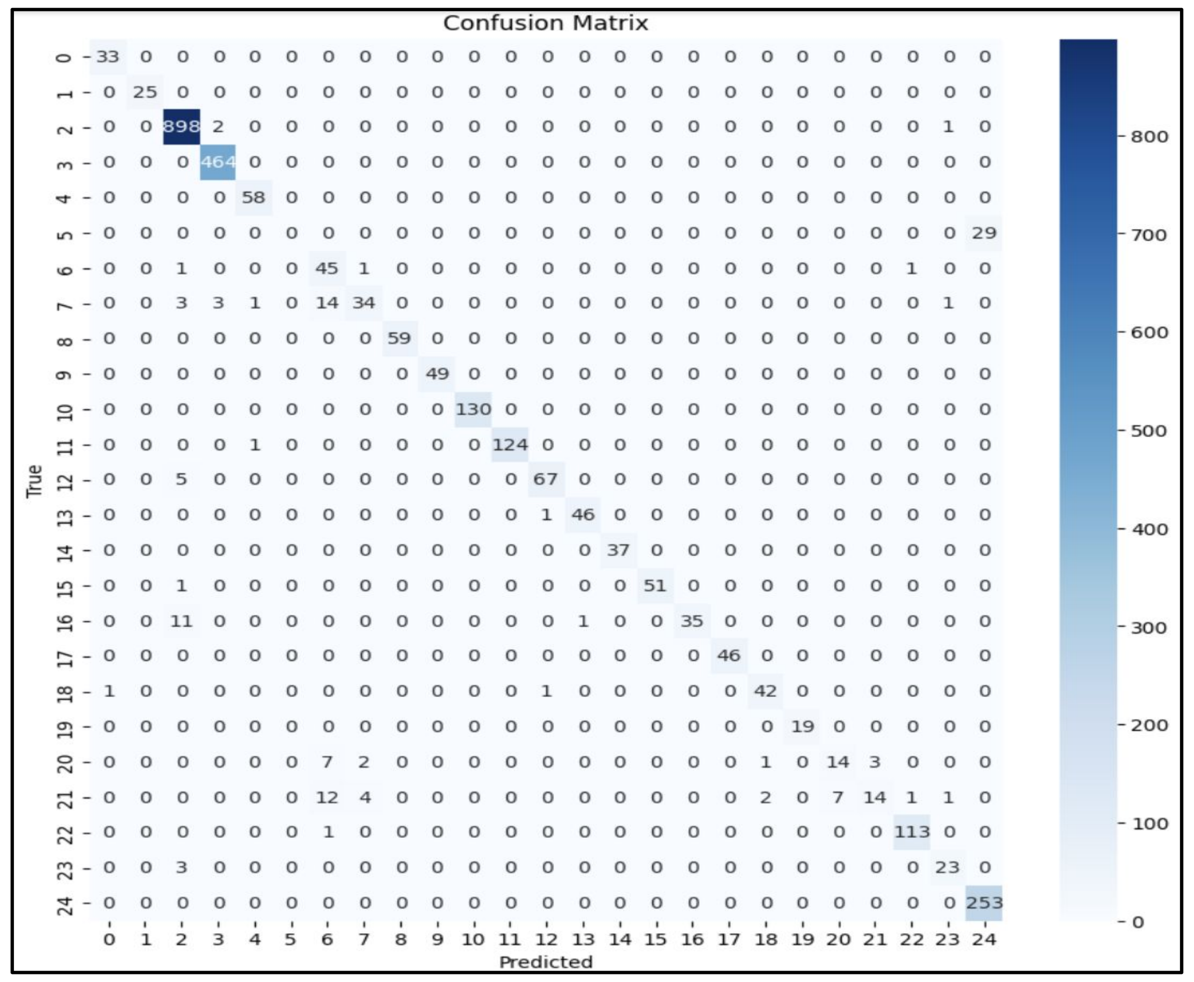


Figure 7: Confusion Matrix

Conclusion

The final model scored 0.96 on Accuracy, 0.95 on Precision, 0.96 on Recall, and 0.95 on F1-Score (weighted). As seen by the test results, the usage of CNNs in malware classification and detection holds vast potential as a compelling alternative to traditional methods. Our plan going forward will now be to expand the models we will use. In this case, we use a convolutional neural network, but we are planning to use models such as ResNet, EfficientNet, VGG-16, and Inception. Furthermore, with the addition of these newer models, we will also be adding other datasets, to be able to test on a wide variety of criteria and types of malware. By being able to use specially trained AI models to classify malware, it will allow many of the modern day issues with malware classification be solved.

Acknowledgments

[1] Kandula, R. (2022). Malimg Dataset. Retrieved from https://paperswithcode.com/dataset/malimg

[2] Bhatt, N. (2021). Max Pooling: Combining Channels Using 1x1 Convolutions. LinkedIn. Retrieved from https://www.linkedin.com/pulse/max-pooling-combining-channels-using-11-convolutions-field-n-bhatt/

[3] Built In. (n.d.). Fully Connected Layer. Retrieved from https://builtin.com/machine-learning/fully-connected-layer

[4] H. AlOmari, Q. M. Yaseen, and M. A. Al-Betar, "A comparative analysis of machine learning algorithms for android malware detection," Procedia Computer Science, vol. 220, pp. 763–768, 2023, the 14th International Conference on Ambient Systems, Networks and Technologies Networks (ANT) and The 6th International Conference on Emerging Data and Industry 4.0 (EDI40). [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877050923006361

[5] A. Wajid, T. Ahmed, and U. B. Chaudhry, "A comprehensive review of machine learning-based malware detection techniques for windows plat- form," The Nucleus, vol. 61, no. 1, p. 51–62, May 2024. [Online]. Avail- able: http://thenucleuspak.org.pk/index.php/Nucleus/article/view/1349

[6] H. Anderson and P. Roth, "Ember: An open dataset for training static pe malware machine learning models," ArXiv, vol. abs/1804.04637, 2018. [Online]. Available: https://api.semanticscholar.org/CorpusID:4888440

[7] P. Maniriho, A. N. Mahmood, and M. J. M. Chowdhury, "A systematic literature review on windows malware detection: Techniques, research issues, and future directions," Journal of Systems and Software, vol. 209, p. 111921, 2024. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0164121223003163