*Malware classification using ANN*

CHARAN R K Ashwini Kodipalli

*Dept. of Artificial Intelligence Dept. of Artificial Intelligence*

*& Data Science & Data Science*

*Global Academy of Technology Global Academy of Technology*

Bangalore, Karnataka, India Bangalore, Karnataka, India

[1ga21ad014@gat.ac.in](mailto:1ga21ad014@gat.ac.in) [dr.ashwini.k@gat.ac.in](mailto:dr.ashwini.k@gat.ac.in)

*Abstract*— The rapid increase of malware poses a significant threat to cybersecurity, necessitating effective detection and classification techniques. Traditional signature-based methods often fall short due to their inability to recognize novel and evolving malware strains. This paper explores the application of Artificial Neural Networks (ANN) for malware classification, leveraging their capability to learn complex patterns and generalize from training data. We developed and fine-tuned a machine learning algorithm based on neural networks. We utilized a diverse and extensive collection of data, which included samples of both harmful malware and legitimate software applications. The proposed ANN model achieved a high classification accuracy, demonstrating its­ efficacy in distinguishing between malicious and non-malicious executables. Comparative analysis of different optimizer techniques highlights the superior performance of our ANN based approach. The results suggest that ANNs can significantly enhance malware detection systems, offering a robust solution for cybersecurity defenses. Future work will focus on optimizing the model and expanding the dataset to further improve classification accuracy and adaptability to emerging threats.

Keywords— Malware Classification, Artificial Neural Networks (ANN), Cybersecurity, Deep Learning, Data Preprocessing, Model Optimization, Classification Accuracy

# INTRODUCTION

The rapid proliferation of online data in recent years has led to an escalating series of obstacles and threats in the realm of digital security. Among these challenges, malware– This term covers a wide range of different types of malicious software such as viruses, worms, Trojans, ransomware, and spyware – poses one of the most persistent and evolving threats. Malware attacks can result in severe consequences, including data breaches, financial losses, and damage to organizational reputation.

Traditional malware detection techniques, predominantly based on signature-based methods, rely on known patterns of malicious code to identify threats. While effective against known malware, these methods struggle to detect new, unseen variants, often referred to as zero-day threats. Consequently, it is increasingly crucial to develop sophisticated identification systems capable of recognizing new and unfamiliar malicious software, beyond just the known examples.

ANNs are a type of machine learning algorithm that draws inspiration from the structure and operation of the human brain. These networks have demonstrated significant potential across a wide range of fields, including image and speech recognition, and natural language processing, and anomaly detection. ANNs have the capability to learn complex patterns and make predictions based on input data, making them well suited for the task of malware classification.

This paper explores the application of ANNs for malware classification, aiming to leverage their pattern recognition capabilities to improve detection accuracy and adaptability. We present a comprehensive approach that involves data collection and preprocessing, neural network design, training, and performance evaluation. The findings of our research indicate that artificial neural networks exhibit superior performance in categorizing malicious software compared to conventional techniques. This approach presents a powerful and reliable method for bolstering digital security measures.

The subsequent sections of this document are structured in the following manner: Section II reviews related work in malware classification and machine learning. Section III details the methodology, including dataset preparation and ANN architecture. Section IV presents the results and discusses the performance of the proposed model. Finally, Section V concludes the paper and outlines directions for future research.

# LITERATURE SURVEY

[1] Ömer Aslan et al. According to their study, paper proposes a novel deep learning architecture to effectively classify malware variants. The hybrid model combines two pre trained networks in an optimized manner, achieving high accuracy on various datasets. Experimental results demonstrate that their method outperforms existing approaches, achieving 97.78% accuracy on the Malimg dataset. [2] Bugra Cakir et al. Their paper proposes a shallow deep learning approach using word2vec for opcode-based malware representation and Gradient Boosting for classification. K fold cross validation ensures robust model evaluation, achieving up to 96% accuracy with limited data. [3] Mahmoud Kalash et al. Their research introduces an advanced machine learning approach that employs CNN to categorize malicious software.; The researchers' innovative approach of transforming malware binaries into grayscale images yielded exceptional results when combined with their CNN design. This method outperformed conventional techniques, as evidenced by their experiments using the Malimg and Microsoft malware datasets. The CNN model achieved remarkable accuracy rates of 98.52% and 99.97% on these datasets respectively, setting a new benchmark in the field of malware detection and classification. [4] Bojan Kolosnjaji et al. According to their study, neural network architecture combining convolutional and recurrent layers to model malware system call sequences. This hierarchical feature extraction approach achieves superior classification performance, surpassing previous methods with an average precision of 85.6% and recall of 89.4%. [5] S. Akarsh et al. Their study presents a deep learning framework for malware classification. They proposed hybrid architecture, combining CNN and LSTM networks, achieving 94.4% accuracy on the Malimg dataset, surpassing previous work. Hyperparameter optimization and a 70 30% train test split were used to evaluate performance on imbalanced data. They used this method to eliminates the need for traditional analysis techniques like disassembly or de obfuscation. [6] Daniel Gibert et al. Their paper introduces HYDRA, a framework that combines various feature types to capture malware characteristics. Their proposed baseline system integrates hand engineered and DL components, leveraging the strengths of both approaches. Their evaluation on the Microsoft Malware Classification Challenge demonstrates comparable performance to gradient boosting methods and outperforms pure deep learning approaches. [7] Xi Meng et al. Their paper presents MCSMGS, a malware classification model that combines static malware gene sequences with deep learning. Their model extracts gene sequences with both material and informational attributes, converting them into distributed representations to capture relationships and similarities. The SMGS\_CNN module they used to analyzes these sequences using a convolutional neural network, achieved up to 98% classification accuracy. Their results demonstrate that MCSMGS outperforms traditional SVM models, highlighting the effectiveness of CNN for malware classification. [8] Rikima Mitsuhashi et al. Their paper addresses the challenge of selecting an appropriate deep learning model for malware visualization image classification. They employed a fine tuning on a pre trained CNN model and addressing data imbalance, the proposed strategy achieves high classification accuracy. They used VGG19, the method achieves 99.72% accuracy on the Malimg dataset, outperforming other approaches. [9] George E. Dahl et al. According to them, their study utilizes random projections to reduce dimensionality for training large scale neural networks. With over 2.6 million labeled samples, the trained networks achieve impressive results, with a two-class error rate of 0.49% for a single network and 0.42% for an ensemble. [10] They investigated the use of NN for malware-detection in IoT devices and classification. Their study, using a dataset of 461,043 IoT network records (300,000 benign, 161,043 malicious), achieved 94.17% detection accuracy and 97.08% classification accuracy. [11] In their research, their analysis of opcode frequency revealed Random Forest's superiority over Deep Neural Networks in categorizing malware. Simpler feature reduction methods like Variance Threshold proved more effective than complex techniques. [12] Their approach is more efficient than existing methods, avoiding lengthy numerical experiments. They are testing with the Microsoft BIG 2015 malware database shows comparable accuracy to current malware detection systems. [13] Their study evaluates ML and DL for malware detection, addresses bias in training data, and proposes a new image processing technique for enhanced zero-day detection. Their research demonstrates that the hybrid approach, combining image processing and deep learning, outperforming existing methods and offers a robust solution for real time malware detection. [14] Their experiment shows that DBN outperform standard models in detection precision. Their study also explores using DBNs as autoencoders to effectively extract and reduce feature vectors.

# Methodology

Our study aimed to develop a predictive model for malware classification using ANN. The project leveraged a comprehensive dataset with varied operational parameters. The research process encompassed three main phases: data collection, data preparation and cleaning, and ANN model construction and optimization.

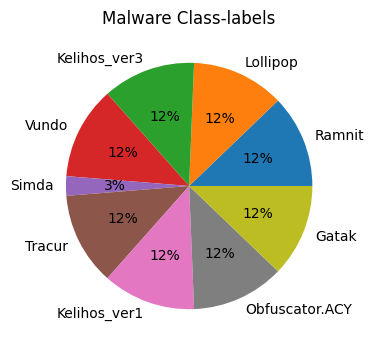
1. DATASET

For our research on malware classification using an ANN model, we assembled a diverse dataset from publicly accessible Microsoft repositories. This collection encompassed 1642 malware samples, representing 9 distinct malware families. Each sample was characterized by 256 features, providing a comprehensive basis for analysis. This carefully curated dataset ensured a broad representation of malware types and variants, enabling robust training and assessment of our ANN model. They are:

|  |  |  |
| --- | --- | --- |
| Class | Family Name | Malware Type |
| 1 | Ramnit | Worm |
| 2 | Lollipop | Adware |
| 3 | Kelihos\_ver3 | Backdoor |
| 4 | Vundo | Trojan |
| 5 | Simda | Backdoor |
| 6 | Tracur | TrojanDownloader |
| 7 | Kelihos\_ver1 | Backdoor |
| 8 | Obfuscator.ACY | Any kind of obfuscated malware |
| 9 | Gatak | Backdoor |

Table 1Malware Name and its type

The distribution of various malware types within our dataset is visually represented in the pie chart provided below. This graphic illustrates the proportional breakdown of different malware categories present in our collection, offering a clear overview of the dataset's composition.



1. DATA PREPROCESSING

We implemented a multi-step data preparation process to refine our dataset, ensuring it was ideally suited for training and assessing our machine learning algorithm:

* **Feature Selection**: We eliminated non predictive columns such as 'Id' and 'BytFSize'.
* **Dataset Partitioning**: The refined data was divided into training (80%) and testing (20%) sets, maintaining stratification of the target variable.
* **Normalization**: We applied Standard Scaler to standardize the feature set.
* **Target Variable Encoding**: Given the presence of 9 malware classes, we implemented label encoding to convert these categories into numerical values.

This comprehensive preprocessing approach ensured our data was well prepared for subsequent model development and analysis.

1. ANN ARCHITECTURE

A multilayer ANN, or multilayer perceptron, consists of multiple layers of neurons: an input layer, one or more hidden layers, and an output layer. These networks are versatile and can manage the non-linearity present in data.

256 FEATURES

Hidden Layer 1

Hidden Layer 2

Hidden Layer 3

Hidden Layer with Relu

Input Layer

Output Layer

Softmax Activation

The structure of our artificial intelligence model is built upon three fundamental elements:

* Input Layer: Consisting of 256 neurons, each corresponding to a distinct input feature, this layer serves as the entry point for data.
* Hidden Layers: We incorporated three hidden layers, which perform complex computations. The layered design of our system improves its capacity to identify complex connections and subtle trends present in the information.
* Output Layer: The final layer contains nine neurons, aligning with our specific classification requirements. This layer produces the network's predictions or results.

This structure enables effective processing our 256 feature input data and classification into nine distinct categories, balancing complexity with task specific needs.

**Activation Function**: In our neural network design, activation functions play a crucial role in introducing non linearity and enabling pattern recognition. We employed two specific functions:

* Rectified Linear Unit: Applied in the hidden layers, ReLU outputs the input value for positive inputs and zero otherwise. This function helps mitigate the vanishing gradient issue and accelerates the learning process.

ReLu(x)=max (0, x) (1)

* Softmax function: The Softmax activation function is a mathematical operation that converts numerical outputs into a probability distribution. It takes a vector of arbitrary real valued scores and transforms them into a vector of values btw 0 & 1, where the sum of these values equals 1. Each element in the resulting vector represents the estimated probability of the corresponding class or outcome. This property makes Softmax particularly useful in multi class classification tasks, as it provides a normalized, interpretable output for each possible category.

(2)

These carefully chosen activation functions enhance our

model's ability to learn complex patterns efficiently,

contributing to its overall performance in the classification

task.

**Optimizers:** Optimization methods play a crucial role in neural network training. They fine-tune the model's parameters to reduce errors and improve performance. In our study, we employed the Adam (Adaptive Moment Estimation) optimizer, which combines the strengths of AdaGrad and RMSProp. The Adam algorithm adjusts the learning speed individually for different components individually and utilizes moving averages to enhance optimization speed.

While we chose Adam, it's worth noting other common optimizers:

* Stochastic Gradient Descent (SGD): Efficient for large datasets but may converge slowly.
* RMSprop: Adjusts per parameter learning rates based on recent gradient magnitudes, suitable for non-stationary conditions.
* Adagrad: Updates parameters based on their gradient history, potentially leading to conservative updates for frequently occurring gradients.
* Adadelta: Improves upon Adagrad by addressing the decreasing learning rate issue, using a moving window of recent gradients to determine the new learning rate without requiring an initial rate selection.

Various optimization methods possess distinct attributes, allowing them to excel in different contexts when developing advanced machine learning models.

1. Explainable AI

Our analysis employed LIME (Local Interpretable Model Agnostic Explanations), a method that demystifies machine learning model outputs. This approach generates comprehensible and precise explanations for singular predictions, regardless of the model's intricacy. LIME's strength lies in its ability to provide localized interpretations, shedding light on the decision-making process at particular data instances, thus enhancing transparency in complex algorithms.

# RESULT

|  |  |  |  |
| --- | --- | --- | --- |
| Optimizer | Train Accuracy | Test Accuracy | Loss |
| Adam  (w/o Dropout) | 0.97 | 0.91 | 0.0098 |
| Adam | 0.98 | 0.94 | 0.0390 |
| Sgd | 0.99 | 0.93 | 0.0101 |
| AdaGrad | 0.98 | 0.93 | 0.0130 |
| AdaDelta | 0.97 | 0.93 | 0.0085 |
| RMSprop | 0.98 | 0.92 | 0.0507 |

In this research, we explored malware classification using ANN and implemented various optimization techniques. Our approach included experiments with and without dropout regularization, as well as utilizing different optimization algorithms such as stochastic gradient descent, Adam etc. Among the methods tested. the peak performance, reaching 94.32% accuracy, was attained by employing the Adam optimization algorithm accompanied by dropout regularization techniques. The following table displays the effectiveness indicators, such as precision rates and error measurements, for each approach we tested.

Table 2 Accuracy and loss

We utilized various data analysis techniques to present and understand our findings visually. To gauge the effectiveness of our model, we created graphs showing error rates and precision over time, offering a visual depiction of the system's performance. Furthermore, we developed ROC (Receiver Operating Characteristic) diagrams to evaluate the model's predictive strength, specifically examining its ability to correctly identify positive cases and its rate of false positives.

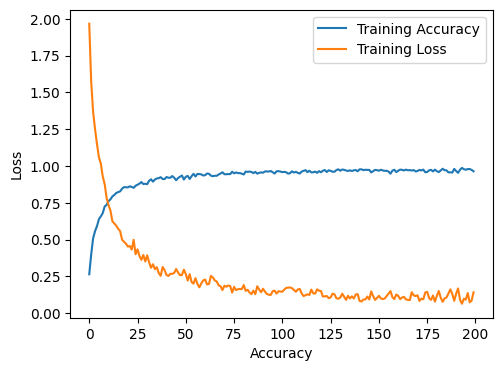


Figure 1 Accuracy vs Loss

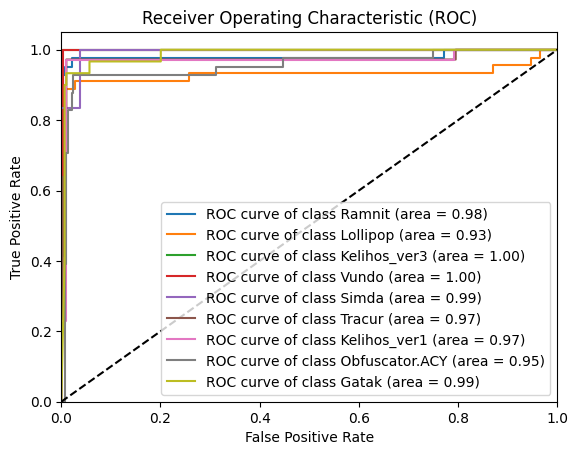
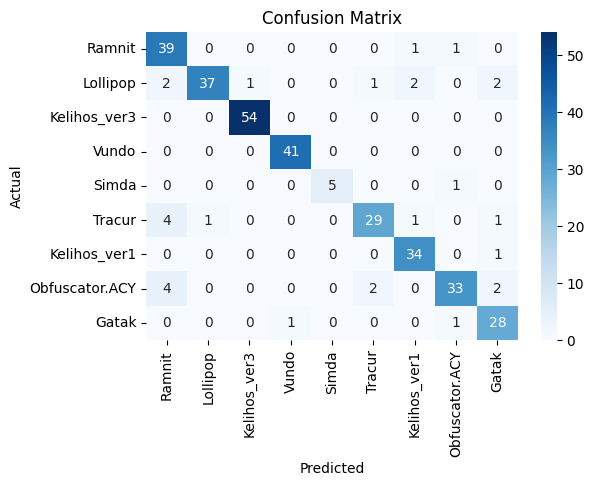


Figure 2 ROC curve

A confusion matrix compares the predicted categories assigned by a model to the actual, true categories of the data.



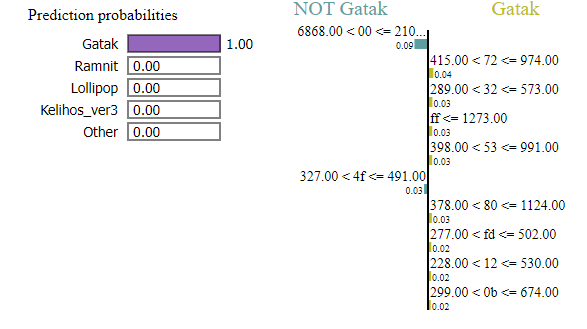
Classification report for our predicted class is tabulated in

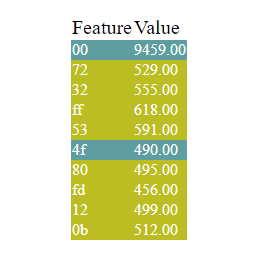
Table 3

Table 3 Classification Report

|  |  |  |  |
| --- | --- | --- | --- |
|  | Precision | Recall | F1 Score |
| Class 1 | 0.80 | 0.95 | 0.87 |
| Class 2 | 0.97 | 0.82 | 0.89 |
| Class 3 | 0.98 | 1.00 | 0.99 |
| Class 4 | 0.98 | 1.00 | 0.99 |
| Class 5 | 1.00 | 0.83 | 0.91 |
| Class 6 | 0.91 | 0.81 | 0.85 |
| Class 7 | 0.89 | 0.97 | 0.93 |
| Class 8 | 0.92 | 0.80 | 0.86 |
| Class 9 | 0.82 | 0.93 | 0.87 |

The following visualizations were generated using the LIME (Local Interpretable Model Agnostic Explanations) technique. This method analyzes individual instances through iterative processes, revealing how specific changes in input features influence the model's predictions. By doing so, LIME provides insights into the model's decision-making process for particular data points.





# CONCLUSION

We built an ANN comprising 1 input layer, 3 hidden layers with ReLU activation functions, and 1 output layer using softmax activation for multi class classification. The model was trained and compiled with specified optimizers and loss functions; particularly categorical cross entropy suited for multi class scenarios. Each optimizer was trained over 300 epochs, allowing us to compare their final accuracy performances.

By leveraging a diverse dataset and employing sophisticated methods for identifying and selecting relevant characteristics, we constructed and trained an ANN model capable of distinguishing between malware families with 94.32% accuracy. The model's performance metrics outperformed other methods.

To sum up, our research enhances the domain of harmful software identification by showcasing how artificial neural networks can effectively recognize and classify malicious programs. Our results highlight the necessity of implementing sophisticated artificial intelligence techniques to strengthen digital security systems and defend against emerging online threats.

# REFERENCE

[1] Aslan, Ö., & Yilmaz, A. A. (2021). A new malware classification framework based on deep learning algorithms. Ieee Access, 9, 87936 87951.

[2] Cakir, B., & Dogdu, E. (2018, March). Malware classification using deep learning methods. In Proceedings of the ACMSE 2018 Conference (pp. 1 5).

[3] Kalash, M., Rochan, M., Mohammed, N., Bruce, N. D., Wang, Y., & Iqbal, F. (2018, February). Malware classification with deep convolutional neural networks. In 2018 9th IFIP international conference on new technologies, mobility and security (NTMS) (pp. 1 5). IEEE.

[4] Kolosnjaji, B., Zarras, A., Webster, G., & Eckert, C. (2016). Deep learning for classification of malware system call sequences. In AI 2016: Advances in Artificial Intelligence: 29th Australasian Joint Conference, Hobart, TAS, Australia, December 5 8, 2016, Proceedings 29 (pp. 137 149). Springer International Publishing.

[5] Akarsh, S., Simran, K., Poornachandran, P., Menon, V. K., & Soman, K. P. (2019, March). Deep learning framework and visualization for malware classification. In 2019 5th International Conference on Advanced Computing & Communication Systems (ICACCS) (pp. 1059 1063). IEEE.

[6] Kolosnjaji, B., Zarras, A., Webster, G., & Eckert, C. (2016). Deep learning for classification of malware system call sequences. In AI 2016: Advances in Artificial Intelligence: 29th Australasian Joint Conference, Hobart, TAS, Australia, December 5 8, 2016, Proceedings 29 (pp. 137 149). Springer International Publishing.

[7] Meng, X., Shan, Z., Liu, F., Zhao, B., Han, J., Wang, H., & Wang, J. (2017, October). MCSMGS: malware classification model based on deep learning. In 2017 international conference on cyber enabled distributed computing and knowledge discovery (CyberC) (pp. 272 275). IEEE.

[8] Mitsuhashi, R., & Shinagawa, T. (2020). High accuracy malware classification with a malware optimized deep learning model. arXiv preprint arXiv:2004.05258.

[9] Dahl, G. E., Stokes, J. W., Deng, L., & Yu, D. (2013, May). Large scale malware classification using random projections and neural networks. In 2013 IEEE International Conference on Acoustics, Speech and Signal Processing (pp. 3422 3426). IEEE.

[10] Jamal, A., Hayat, M. F., & Nasir, M. (2022). Malware detection and classification in IoT network using ANN. *Mehran University Research Journal Of Engineering & Technology*, *41*(1), 80 91.

[11] Rathore, H., Agarwal, S., Sahay, S. K., & Sewak, M. (2018). Malware detection using machine learning and deep learning. In Big Data Analytics: 6th International Conference, BDA 2018, Warangal, India, December 18–21, 2018, Proceedings 6 (pp. 402 411). Springer International Publishing.

[12] Dychka, I., Chernyshev, D., Tereikovskyi, I., Tereikovska, L., & Pogorelov, V. (2019, January). Malware detection using artificial neural networks. In *International Conference on Computer Science, Engineering and Education Applications* (pp. 3 12). Cham: Springer International Publishing.

[13] Vinayakumar, R., Alazab, M., Soman, K. P., Poornachandran, P., & Venkatraman, S. (2019). Robust intelligent malware detection using deep learning. *IEEE access*, *7*, 46717 46738.

[14] Yuxin, D., & Siyi, Z. (2019). Malware detection based on deep learning algorithm. *Neural Computing and Applications*, *31*, 461 472.