iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 11 **Issue:** VIII **Month of publication:** Aug 2023

DOI: https://doi.org/10.22214/ijraset.2023.55233

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 11 Issue VIII Aug 2023- Available at www.ijraset.com

Detection of Malware using Machine Learning Approach

Bhagyashree A Patil[1], Sri Adithya S[2], Dr. Jayanthi M G[3]
[1]*PG Student, Department of CSE CITech, Bangalore, Karnataka, India*
[2]*UG Student, Department of BCA Cambridge Degree College Bangalore, Karnataka, India*
[3]*Associate Professor, Department of CSE CITech, Bangalore, Karnataka, India*

Abstract: ***This proposal abstract outlines a new approach to classification of malware using machine learning algorithms. Malware detection is an essential task in cybersecurity to identify and mitigate potential threats posed by malicious software. This research proposes a new framework of Support Vector Classifiers and Decision Tree these are very know machine learning algorithms which will be used on malware detection. The proposed work is designed to effectively classify whether the network data is malware or normal behavioural characteristics. The designed approach is compared with traditional algorithms of Machine Learning, such as Support Vector Machines (SVM) and Decision Tree to evaluate its performance. The outcomes of this research are expected to yield on the development of more effective malware detection and prevention systems.***
Keywords: Malware classification, Malware detection, Support vector classifier, Decision tree.

I. INTRODUCTION

Recent advances in technology have made human day to day work easier and more convenient. With the ability to perform a various activity on the browsers, such as socializing, conducting financial transactions, and measuring bodily changes, cyber criminals are increasingly drawn to committing crimes in cyberspace. Cyber-attacks, which consists of malware, has affected the cost of world economy trillions of dollars. Malware is a software specifically designed to harm, disrupt, or gain unauthorized access to computer systems, networks, or individual devices. Malware is created by cybercriminals and hackers with the intention of causing damage, stealing sensitive information, or using the compromised system for various nefarious purposes. For not to be visible on the victim's system, the malware uses various techniques as encrypting and tokenisation and often spread by exploiting or destroying trust of humans as a vector. To protect Laptops and Android Phones from these kind of malicious attacks, classification of detection is essential. Malware detection involves analysing suspicious files and determining whether they are severe or non-benign.

Malware classification is the process of detecting the malware it belongs to which category or fam of the detected malware. This research focuses on proposing a new approach to classifying a malware using a machine learning algorithm. The framework which has been proposed compares Support Vector Machine and Decision Tree classifier models to detect and classify malware based on the provided datasets and its training. The experimental results of the proposed work will be conducted on a large-scale dataset of malware samples, and the performance will be made as a comparison with machine learning algorithms of SVM and decision tree classifier. By leveraging the strengths of ML techniques, the proposed framework is expected to provide greater accuracy in protection for networks and Computers against the increasing threat of malware.

II. LITERATURE SURVEY

1) In this paper, the presented work, "FLOWDROID: Precise Analysis for Android Phones," addresses the issue of detecting information leakage in Android phones caused by both accidental programming errors and intentional malicious activities. Smartphones are abundant sources of private and confidential data, making it crucial to identify and prevent data leaks to protect users' sensitive information. The method proposes FLOWDROID, a static approached tool for Android Systems. Taint analysis is a technique used to track data flow within a program and identify how sensitive data can propagate to untrusted areas.

2) In this paper, Dexpler is built on top of two existing tools, Dedexer, and Soot. Dedexer is a software that can extract Dalvik bytecode from Android APK files, while Soot is a framework that is explicitly used in Java bytecode analysis and transformation, with Jimple being its intermediate representation. Dexpler uses Dedexer to extract Dalvik bytecode from Android APKs and then leverages Soot's powerful analysis capabilities, with Jimple as the intermediate representation, to perform various analyses or transformations on the bytecode. This combination of tools allows Dexpler to proceed on Android apps' bytecode efficiently and effectively.

III. EXISTING SYSTEM

The researchers have been made several works on approaching for malware detection. Existing malware detection techniques are based on signature detection, Heuristic Analysis, Behavioural Analysis. These techniques face many limitations as malware variants increase, the signature database becomes larger, leading to higher memory usage and slower scanning times. Behavioural and Heuristic analysis may not be able to detect sophisticated or well-disguised malware that exhibits benign behaviours until activated.

IV. PROPOSED SYSTEM

This proposal abstract outlines an approach to classify malware with the help of Machine learning algorithms. In this research, the proposed framework that consists of Support Vector Machine and Decision tree models to classify malware. It is designed to effectively detect and classify whether the network data is malware or normal behavioural characteristics. The proposed framework leverages the strengths of both SVM and Decision tree classifier models, allowing it to detect and classify malware more accurately than existing techniques. 2) The performance of each approach will be evaluated using standard metrics such as accuracy, precision, recall, and F1 score. The proposed work will also evaluate the robustness of the SVM and Decision tree learning model to different types of malware and variations in the dataset.

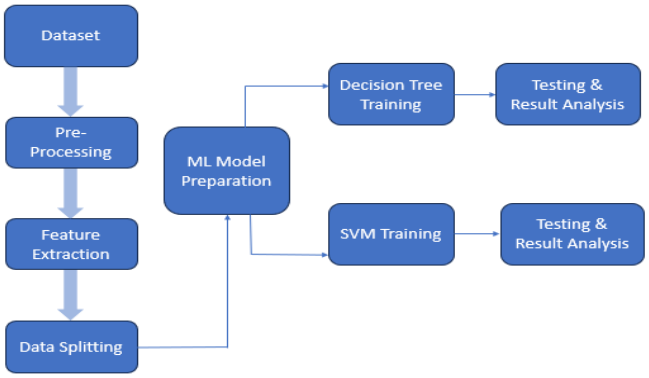


Fig 1. Block diagram of the Proposed System

V. DATASET

To analyse the accuracy of machine learning it is required to create a large dataset with a various sample. Over time, as malware have grown it has become increasingly difficult to have one source having all types of malware families. In this work we have used the open-source dataset DREBIN dataset consisting of feature vectors of 215 attributes that are extracted from 15,036 applications among which 5,560 are malware from dataset and 9,476 benign. This dataset is randomly divided into 80% training and 20% testing using Scikit-learn.

Fig 2 shows the bar chart of training dataset contains 9,476 benign files and 5,560 malicious files.

Fig 3 shows the pie-chart for the same dataset with 63.02% of benign and 36.98% of severe.

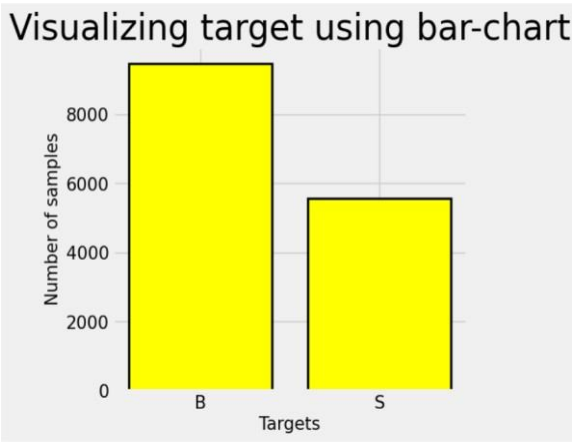


Fig 2: Visualizing bar chart for dataset

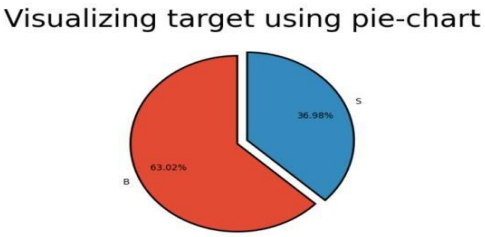


Fig 3: Visualizing dataset with pie-chart

Adopted Synthetic Minority Oversampling Technique (SMOTE) is a method used in machine learning and data mining to address the issue of class imbalance in a dataset. It tackles class imbalance effectively. Initially, the training dataset is randomly split into 80% training and 20% testing. The tested data helps to observe the training accuracy across different epochs. Here we apply this SMOTE technique to our approach to solve the imbalance problem and generate a balanced dataset.
Fig 4. shows the results of the smote technique. Basically 0 indicates benign files and 1 indicates severe files.

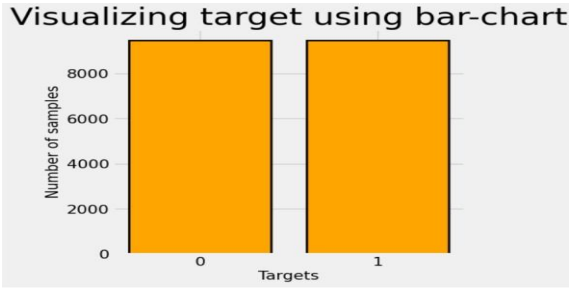


Fig 4: Results of SMOTE Technique

VI. HARDWARE AND SOFTWARE REQUIREMENTS

1) CPU: Intel 2.1 GHZ
2) Memory: 4GB
3) Disk: 100GB
4) Display: 15.6-inch color
5) Coding platform: Python 3.7 and higher
6) Tool: Anaconda
7) IDE: Jupyter Notebook
8) OS: Windows 10
9) Libraries: pandas, numpy, Sklearn, tensorflow, etc

VII. BLOCK DIAGRAM

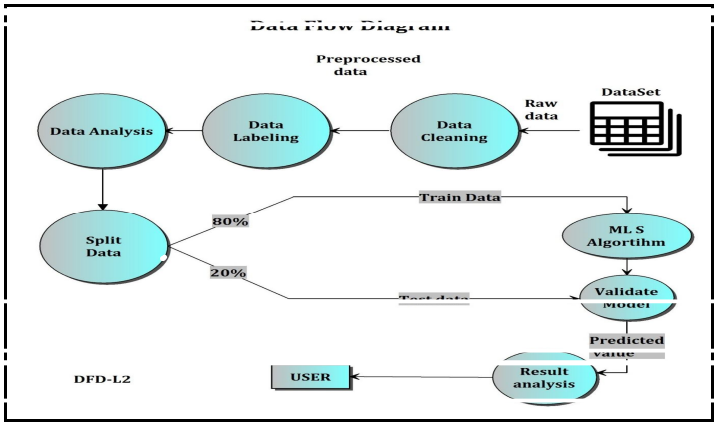


Fig 5. System Architecture

VIII. RESULTS

The following snapshots define the results or outputs that used to get after step-by-step execution of all the modules of the system.

A. *Support Vector Classifier - Report on Classification*

	precision	recall	f1-score	support
BENIGN	0.88	0.77	0.82	1896
SEVERE	0.80	0.90	0.84	1895
accuracy			0.83	3791
macro avg	0.84	0.83	0.83	3791
weighted avg	0.84	0.83	0.83	3791

Fig 6: Classification Report for Support Vector Classification

B. *Decision Tree Classifier - Report on Classification*

	precision	recall	f1-score	support
BENIGN	0.96	0.97	0.96	1896
SEVERE	0.97	0.96	0.96	1895
accuracy			0.96	3791
macro avg	0.96	0.96	0.96	3791
weighted avg	0.96	0.96	0.96	3791

Fig 7: Classification Report over Decision Tree Classification

C. *Support Vector Classifier - Confusion Matrix*

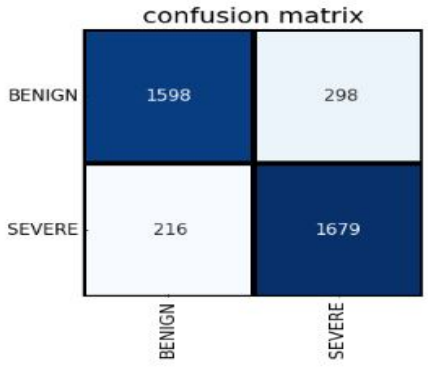


Fig 8: Confusion Matrix over Support Vector Classifier

The Support Vector Classifier performance is depicted in Fig 8 using a confusion chart. The model was evaluated on 3791 samples, achieving an accuracy of 83%. For the BENIGN class, 1598 instances were correctly predicted, while 298 were incorrectly classified. Similarly, for the SEVERE class, 1679 instances were accurately predicted, with 216 misclassifications.

D. Decision Tree Classifier - Confusion Matrix

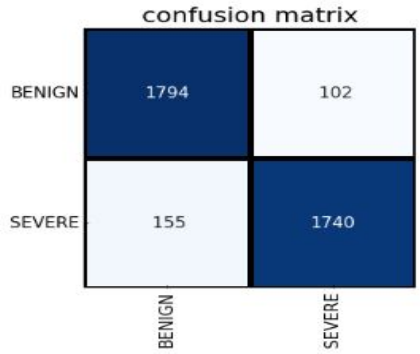


Fig 9: Confusion Matrix over Decision Tree Classifier

The Decision Tree Classifier's performance is depicted in Fig 9 using a confusion chart. The model was evaluated on 3791 samples, achieving an accuracy of 96%. For the BENIGN class, 1794 instances were correctly predicted, while 102 were incorrectly classified. Similarly, for the SEVERE class, 1740 instances were accurately predicted, with 155 misclassifications.

E. Accuracy plot-graphs and Loss plot-graphs

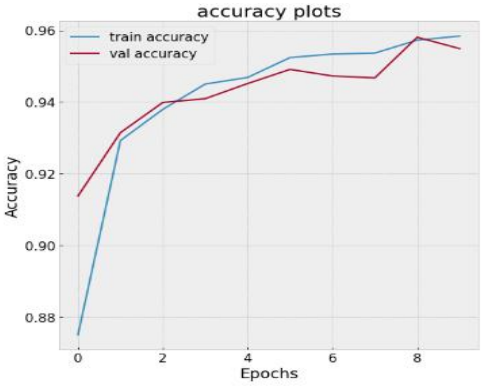


Fig 10: Accuracy Plot-graph

An accuracy plot graph, also known as an accuracy curve or accuracy vs. epoch plot, is a visual representation of how well the performance of a machine learning model changes over different training iterations (epochs) during the training process.

F. Loss plot-graphs

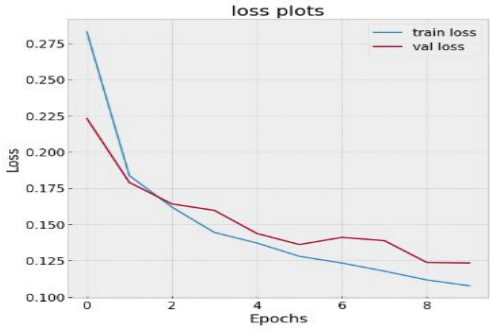


Fig 11: Loss Plot-graph

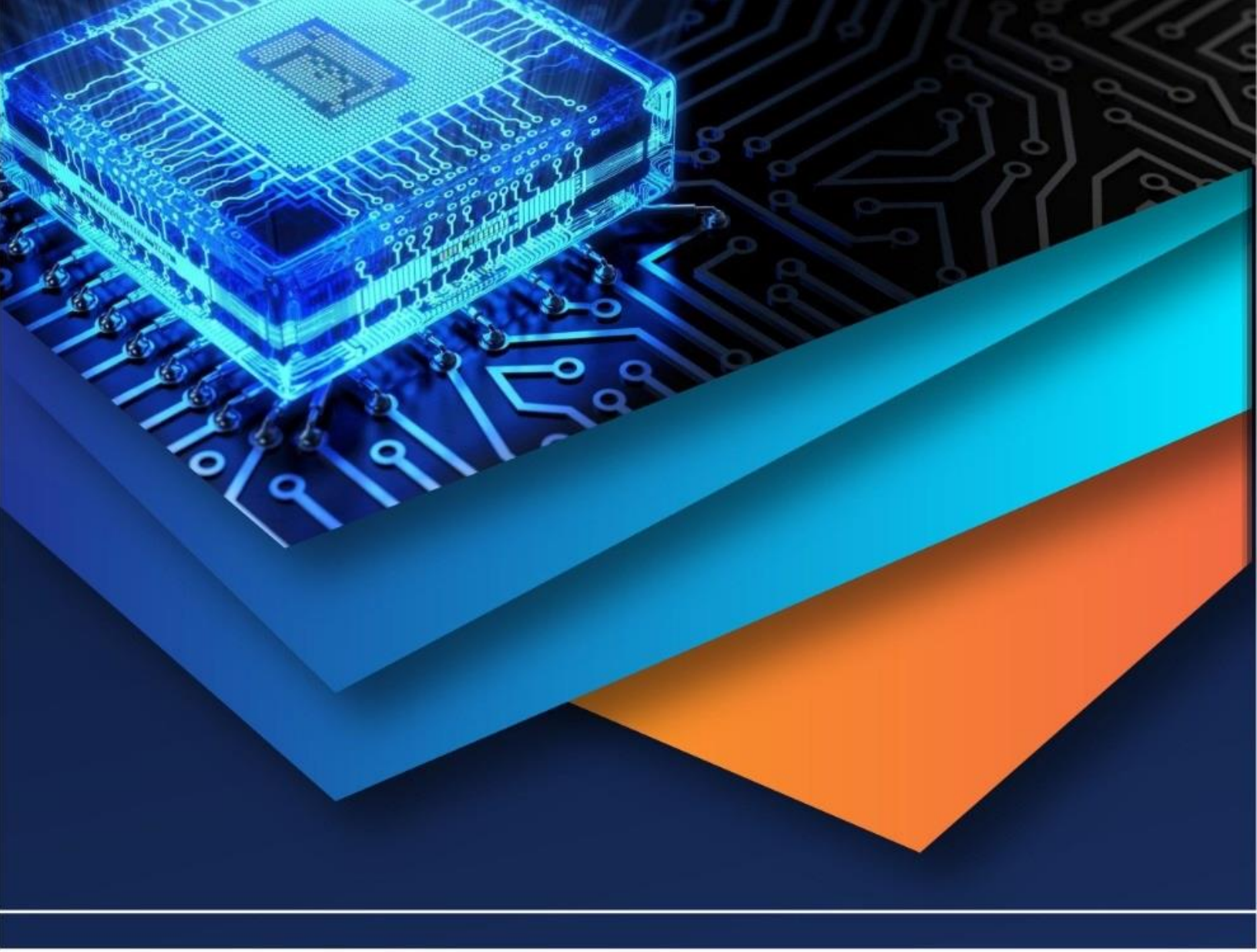
A loss plot graph, also known as a loss vs. epoch plot, is a visual representation of how the loss or error of a machine learning model changes over different training iterations (epochs) during the training process. It is commonly used to analyse the performance of the model during training and to understand how well the model is learning from the data.

IX. CONCLUSION

The test results confirmed that the proposed method can effectively classify malware with high precision, recall, accuracy, and f-score. 96.18% accuracy is obtained for Decision tree classifier which is outperformed most of the ML-based malware detection method and with comparison over Support vector machine algorithm. In addition to this, it is observed that the proposed method is efficient and reduces feature space on a large-scale domain. The project involved feature extraction, model training, and evaluation to assess the effectiveness of each classifier. Decision tree classifier algorithm has reinforced the importance of integrating advanced Machine learning methods into malware detection systems to enhance their robustness and fortify the overall security posture against the ever-growing landscape of cyber-attacks.

REFERENCES

[1] Duy-linh nguyen , muhamad dwisnanto putro, and kang-hyun jo “Driver Behaviors Recognizer Based on Light-Weight Convolutional Neural Network Architecture and Attention Mechanism”, IEEE, Digital Object Identifier 10.1109/ACCESS.2022.3187185, July 2022.
[2] Xianhui Chen, Ying Chen, Wenjun Ma, Xiaomao Fan, Ye Li, “Toward sleep apnea detection with lightweight multi-scaled fusion network”, Elsevier, 8 April 2022.
[3] R. Lyer, The Political Economy of Cyberspace Crime and Security. New York, NY, USA: Academic, 2019.
[4] O Aslan and R. Samet, ‘‘A comprehensive review on malware detection approaches”,IEEE Access, vol. 8, pp. 6249–6271, Jan. 2020.
[5] R. Komatwar and M. Kokare, ‘‘A survey on malware detection and classification. Appl. Secur. Res., pp. 1–31, Aug. 2020.
[6] S. A. Roseline, S. Geetha, S. Kadry, and Y. Nam, ‘‘Intelligent vision-based malware detection and classification using deep random forest paradigm,’’ IEEE Access, vol. 8, pp. 206303–206324, 2020.
[7] M. Nisa, J. H. Shah, S. Kanwal, M. Raza, M. A. Khan, R. Damasevicious, and T. Blazauskas, ‘‘Hybrid malware classification method using segmentation-based fractal texture analysis and deep convolution neural network features, Appl. Sci., vol. 10, no. 14, p. 4966, 2020
[8] J. Love. (2018). A Brief History of Malware-Its Evolution and Impact. Accessed: Mar. 20, 2021. [Online] Available: https://www.lastline. com/blog/history-of-malware-its-evolution-and-impact.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY