Journal of Science and Technology
ISSN: 2456-5660 Volume 10, Issue 05 (May -2025)
www.jst.org.in DOI:https://doi.org/10.46243/jst.2025.v10.i05.pp38- 49

Malware Detection using the concept of Random Forest Algorithm

DharmendraThapa[1] | Hari Narayan Ray Yadav[2]|Madhav Dhakal[3,*]

[1]Central Department of Computer Science and Information Technology, Tribhuvan University, Kathmandu, Nepal,
[2]Graduate School of Engineering, Mid-West University, Surkhet, Nepal,
[3]Graduate School of Science and Technology, Mid-West University, Surkhet, Nepal
[1]dharmendrathapa3@gmail.com, [2]harinarayan.yadav@mu.edu.np , [3]madhav.dhakal@mu.edu.np
*Corresponding email: madhav.dhakal@mu.edu.np

To Cite this Article
DharmendraThapa | Hari Narayan Ray Yadav |Madhav Dhakal, "Malware Detection using the concept of Random Forest Algorithm" *Journal of Science and Technology, Vol. 10, Issue 05-May 2025, pp38-49*

Article Info
Received: 23-01-2025 Revised: 22-04-2025 Accepted: 09-05-2025 Published:21 -05-2025

ABSTRACT

Malicious software is abundant in a world of innumerable computer users, who are constantly faced with these threats from various sources like the internet, local networks and portable drives. Malware is potentially low to high risk and can cause systems to function incorrectly, steal data and even crash. Malware may be executable or system library files in the form of viruses, worms, Trojans, all aimed at breaching the security of the system and compromising user privacy. In this study, the proposed machine learning algorithm is RF algorithm which use Gini index CART algorithm to create multiple decision tree with majority of the outputs from each decision trees. Here, total 1,38,047 data is collected which contain 96,724 malware and 41,323 legit. RF algorithm achieved 99.54% accuracy during malware detection followed by 99.13% precision, 99.35% recall and 99.24% f1 score respectively during testing.

KEYWORDS: *Decision Tree, Malware Detection, Machine learning, Random Forest,*

I.INTRODUCTION

Malware refers to a program that is inserted into a system, usually covertly, with the intent of compromising the confidentiality, integrity, or availability of the victims' data, applications, or operating system, annoying or disrupting the victim (Mell et al., 2005). Malware is common term for the computer threats; it is specially designed to damage the computer system without informed users. Viruses, worms and Trojan horse are forms of computer malware.

In other word a program/code which is designed to penetrate the system without user authorization and takes inadmissible action is known as malicious software or malware. Malware, short for malicious software, is a sweeping term for viruses, worms, Trojans and other harmful software programs which can either create harm to data or access some important data illegally. Malicious software is widely regarded as one of the most effective threats in cyberspace and to modern

computer systems. It can be categorized into several groups.

Problem of the statement

Nowadays, the attackers use intelligent techniques to generate new profitable malware. As computer systems have become an integral part of every organization and person, it is a big challenge to safeguard the computer systems from malicious activities which compromise not only the systems but also the data stored within.

Traditional malware and root kit detection using antivirus systems are not dynamic enough to capture the complex behavior of malware and its isolated activities. There are many signature-based and behavior based malware detection techniques have been introduced, but enterprises as well as general users are still facing problems to get protection for their cyber systems against malware. Thus, it emphasizes the necessity of developing an efficient malware detection technique. To solve the problem facing by the malware the RF classifier is used. The present paper makes the following contribution:

a) detect malware using random forest algorithm
b) analyze the random forest performance for malware detection

II. LITERATURE REVIEW

Different research proposed different idea and various techniques to detect malware software or program. Authors in (Khammas, 2020) proposed machine learning technique (RF classifier) to detect ransomware attack. He tested different sizes of tree and seeds ranged from 10–1000 and from 1–1000 respectively. Analysis emphasized tree size of 100 with seed size of 1 achieved a high accuracy of 97.74 percent. Authors in (Garcia & Ii, 2016) exhibited the used of malware images as a feature vector for classifying various malware families. This study used RF and performed 10-fold Cross Validation to determine the predictive strength of the model. They achieved a 95.26 percent classification accuracy for the given malware dataset. However, there was still a thing to consider such as misclassification on visually

similar malware families. (Sun et al., 2017) proposed Feature extraction concept using SVM know as extraction method of Android malware feature based on KCD. The method makes use of the Keywords Correlation Distance in order to compute the correlation in between the key codes such as the API calls, the Android permissions, the common parameters, and the common key words in Android malware source code. The experiments also show that the method is much more efficient and also effective in the process of detecting malwares on Android platform. The accuracy achieved is around 87-88 percent. (Pang et al., 2019), proposed the Android malware static detection method base on Naive Bayes. Authors requested the permissions, the system API calls, and the proportion of Activity among the four Android major components through Android packages. They used the three types of information as the features to characterize each of the applications, and then perform the classification model training and malware detection through Naive Bayes classifier. For datasets they tested 6120 Android malwares and 6032 benign applications. The accuracy level which is achieved is 87.18 percent. In the study (Masum & Shahriar, 2019) authors introduced Droid-NNet , a neural network-based system for detecting Android malware. The model is trained with ℓ2regularization, early stopping, and mini-batch gradient descent to improve performance and reduce overfitting. The current trends in malware analysis and detection are examined in (Aboaoja et al., 2022), with a particular focus on aspects that have been underexplored or insufficiently addressed in prior works. Authors in ref. (Gavrilut et al., 2009) emphasis on reducing false positives, a flexible machine learning system for malware detection. It has been successfully scaled to handle large-scale datasets and is implemented utilizing cascade one-sided perceptrons and their kernelized variant.

III. METHODOLOGY

Dataset

The dataset for this study is retrieve from github repository which include 1,38,047 data with 57 features columns. Among 1,38,047 there are 41,323 legit and 96,724 malware data. These data included executable file format descriptions, code descriptions, binary data statistics, text strings and information extracted via code emulation and other similar data. The source of data is https://github.com/PacktPublishing/Mastering-Machine-Learning-for-PenetrationTesting/tree/master/Chapter03.

Working Model

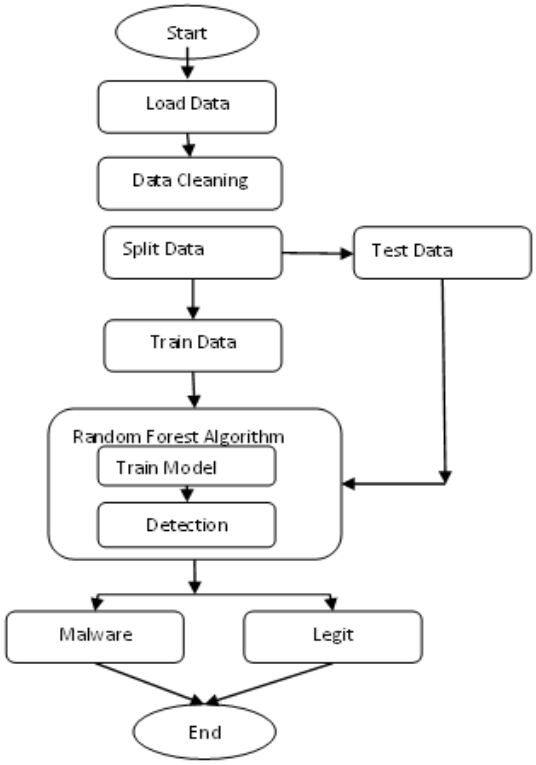


Figure 1: Working Model of Malware Detection

Data Cleaning/ Data Pre-processing

The main proposed of this phase is to remove unnecessary data as well as null data from dataset. This phase also include other step like remove stop word, remove special characters, tokenization but these steps are not used in this report.

Splitting Data

In this phase dataset are split into two categories which are training and testing. Here, sklearn used for validates the model. Here dataset is split into 80/20% where 1,10,437 are used for training and 27,610 are used for testing propose library is used to split the dataset. The training data are used for train a model whereas test data are

Random Forest (RF) Algorithm

This algorithm use CART method for decision tree which use Gini method to create split points including Gini Index (Gini Impurity) and Gini Gain. This algorithm contains separated random dataset from original dataset which is known as bagging to generate multiple decision trees. Main concept for generating

decision tree is Gini index which helps to determine the splitting node or splitting criteria for decision trees node. Which nodes have minimum Gini index selected as a root node and split decision tree into leaf node. Gini index can be calculated by,

$$Gini = 1 - \sum_{j=0}^{n} p_j^{\,2} \ ,$$

Where, P is the probability and j is the number of data present in bootstrap dataset.

Algorithm

Step 1: Create bootstrap table by taking k number of random records from n numbers of records in dataset.

Step 2: Construct individual decision trees for each bootstrap table.

Step 3: Each decision tree will generate an output for input.

Step 4: Final output is considered based on Majority Voting or Averaging for Classification and regression respectively

For Example:

Table 1. Sample of Dataset for Malware Detection

Resource Size	Resource Node	Section Virtual Size	Code size	Characteristics	Legitimate
18032	4	551848	361984	258	1
1156	2	130296	130560	3330	1
318464	4	386824	111616	258	0
270376	11	516760	517120	3330	1
81654	7	205644	205824	258	0
4264	10	585488	585728	258	1
1300	2	294816	294912	258	1
67624	26	180988	37888	33167	0

Step1: Here 3 bootstrap tables are created randomly using main dataset.

Table 2. Bootstrap-1

Resource Size	Code size	Legitimate
18032	361984	1
270376	517120	1
4264	585728	1
81654	205824	0

Table 3. Bootstrap-2

Section Virtual Size	Resource Node	Legitimate
516760	11	1
130296	2	1
205644	7	0
386824	4	0

Table 4. Bootstrap-3

Resource Node	Section Virtual Size	Legitimate
10	585488	1
2	294816	1
7	205644	0
26	180988	0
10	48690	0

Step 2: CART decision tree is used to for random forest because it use Gini index to determine the splitting and root node. Gini index technique is simplest which use probability to make a decision and better than entropy which use complex mathematical operation to make a decision.

Phase 1: ResourceSize(RS) and SizeOfCode(SC) attributes are randomly selected for generating decision tree from given dataset.

Phase 2: Now Gini indexes are calculated for RS and SC on each data by averaging. The data for RS are 4264, 18032, 81654 and 270376 in acceding order. 11148, 49843 and 176015 are the average value of adjacent data. Calculating Gini index of each data using each average data as below:

Gini(T)=1−[P(L)^2+P(M)^2]

$$P(L) = \frac{1}{1} = 1 \wedge P(M) = \frac{0}{1} = 0$$

Gini(T) = 1−(1^2+0^2)=1−(1−0)=0

$$P(L) = \frac{2}{3} \wedge P(M) = \frac{1}{3}$$

$$Gini(F) = 1 - \left(\left(\frac{2}{3}\right)^2 + \left(\frac{1}{3}\right)^2\right) = 1 - \left(\frac{4}{9} + \frac{1}{9}\right) = 0.45$$

Now compute Weighted Gini Index for the Split as:

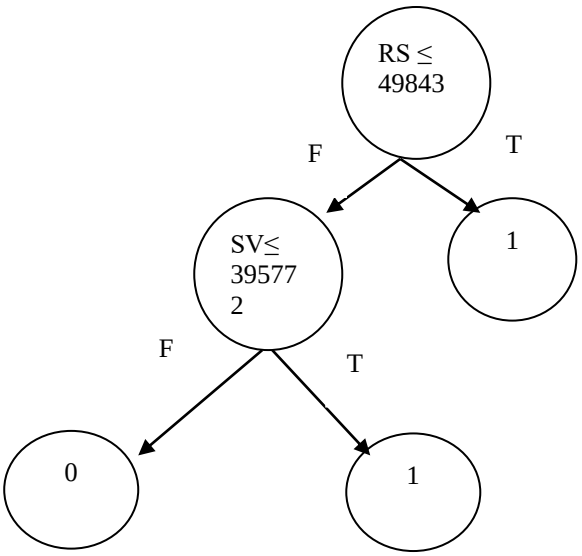
$$Gini(RS \leq 11148) = \frac{Gini(T)\,xcount(T) + Gini(F)\,xcount(F)}{count(T) + count(F)}$$

$$= \frac{(0x1) + (0.45x3)}{1+3} = 0.3375$$

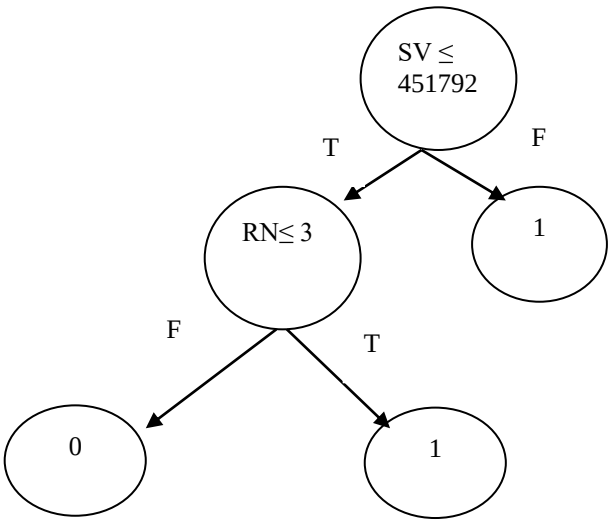
The minimum Gini index is choose for root node so the winner node for RS is Gini (RS ≤ 49843)

Similarly, Gini indexes for SC are, Gini (SC ≤ 439552) = 0.25 ,Gini (SC ≤ 551420) = 0.3375, Gini (SC ≤ 395772) = 0.25.Here both Gini (SC ≤ 439552) and Gini (SC ≤ 395772) have minimum Gini index so either of them can be choose as

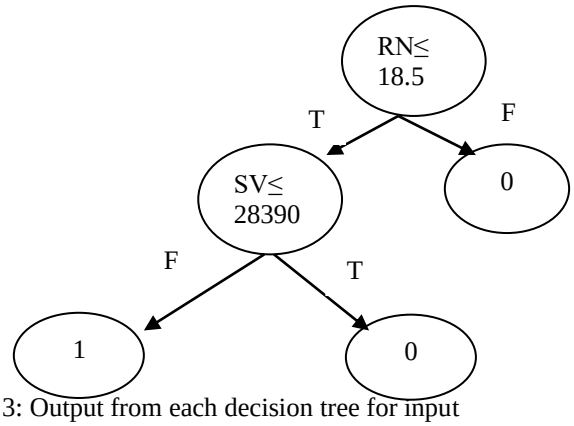
winner node. Let Gini (SC ≤ 395772) is winner for this report. Now among both attribute RS and SC the root node should be minimum Gini index but here both have same Gini index then either one of the can be choose that's why root node is Gini (RS ≤ 49843) for this report (Note: Gini (SC ≤ 395772) also can be a root node). The final decision tree for Bootstrap-1 table is:



Similarly, decision tree for Bootstrap-2 table is



Similarly, decision tree for Bootstrap-3 table is



Step 3: Output from each decision tree for input

RS	RN	SV	SC	C
22648	22	330808	118272	258

According to above Decision Tree of Bootstrap-1, Bootstrap-2 and Bootstrap-3, the output are 1, 0, and 0 respectively.

Step 4: Final output is considered based on Majority Voting for Classification. Majority of 0 is higher than 1 so base on the given input data software or program is Malware. The above process of RF algorithm is applied into 2 phase. They are;

a. Training Model Phase
b. Detecting/Predicting Phase

Training Model Phase

Generating a bootstrap table is done in training phase which used training data (i.e, 80% data from dataset). Gini calculation phase occurred after generating a bootstrap table which help to construction of decision trees for each bootstrap data.

Detecting/Predicting Phase

This phase provides the result based on the input data with the help of multiple decision tree. This is the last phase of the RF algorithm.

Performance Measurements

Performance of the model is determined not only by its accuracy but also other various factors. They are:

Confusion Matrix: A confusion matrix is a technique for summarizing the performance of a classification algorithm. In classification accuracy
alone can be misleading if dataset is unequal number of observations in each class or if dataset have more than two classes. Calculating a confusion matrix can give better idea of what model is getting right and what types of errors it is making. Confusion matrix contain TP, FP, FN, TN data from dataset.

Accuracy: In classification, Accuracy Score is the ration of correct predictions to the total number of input data points.

Precision: Precision is the ratio of number of True Positive to the total number of Predicted Positive.

It measures, out of the total predicted positive, how many are actually positive.

Recall: Recall is the ratio of number of True Positive to the total number of Actual Positive. It measures, out of the total actual positive, how many are predicted as True Positive.

F1 Score: F1 Score is an important evaluation metric for binary classification that combines Precision & Recall. F1 Score is the harmonic mean of Precision & Recall.

IV. IMPLEMENTATION

The implementation is carried out using python and its library, dataset retrieve from github and RF algorithm. They are;

Panda: In this report panda is used to import/load the dataset as well as evaluating the nature of dataset. It is also used to remove unnecessary data and null data present in dataset.

Sklearn: This library is used for many propose like splitting data, train model and testing data. Sklearn library include different module for different proposed among them following are used during implementation.

a. **model_selection:** This module is used to split data into train and test data using train_test_split() method.

b. **RandomForestClassifier:** This module is used for train the model using fit() method. This module also used for malware detection using predict() method.

c. **Metrics:** This module is used to analyze the performance of this model. The performance analysis include accuracy, precision, recall and f1 score with its respective classes.

Splitting Data

Here train_test_split() method of model_selection module which are included in sklearn library is used for split dataset into 80/20% (i.e, 80% data are used for training proposed and remaining 20% data are used for testing proposed). Out of 1,38,047 data 1,10,437 are splited into training data and 27,610 are splited into testing data. The

dataset is split into train data, train lable data, test data and test label data (i.e, X_train, Y_train, X_test, Y_test).

Train Model

Here fit() method of RandomForestClassifier() module from sklearn library is implemented to train the model. Train data and train label data(i.e, X_train, Y_train) is pass to the fit() method with 100 n_estimators as a parameter (i.e, no of decision tree) which help to learn. After training train data is apply only on predict() method to get the result of the model. The performance measurements and confusion matrix is implemented which is discuss later in this report.

Detection/Prediction

To detect the any new software or program pass the data into predict() method of RandomForestClassifier() module. Here data is

read from the excel file using panda library. After import/read data, it remove null and unnecessary data which is present into the input data after that create a object for RandomForestClassifier() module and pass the input data into predict() method. At last result is automatically generate which classify the software or programme is malware or legit.

V. RESULTS AND ANALYSIS

Here analysis is done in 2 different ways.

1. Result and analysis of training data
2. Result and analysis on testing data
3. Detection Result

Result and analysis of training data

Training data are 1,10,437 out of 1,38,047 dataset which contain 77,474 malware and 32,963 legit. In training data 77,473 data are correctly predict as a malware, 1 data is wrongly predicted as a malware, 1 data is wrongly predicted as a legit and 32,962 data are correctly predicted as a legit with the help of confusion matrix which is shown below,

Table 5. Confusion Matrix for Training Data

True Positive	False Positive	False Negative	True Negative
77473	1	1	32962

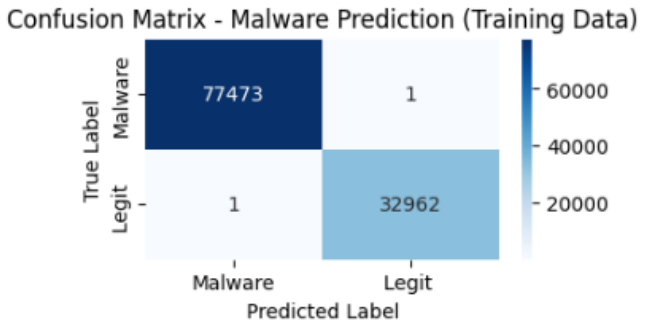


Figure 2: Confusion Matrix for Training Data

The achieved accuracy, precision, recall and f1 score of the test model are 99.998%, 99. 997%, 99.997 % and 99.997% with the help of confusion matrix.

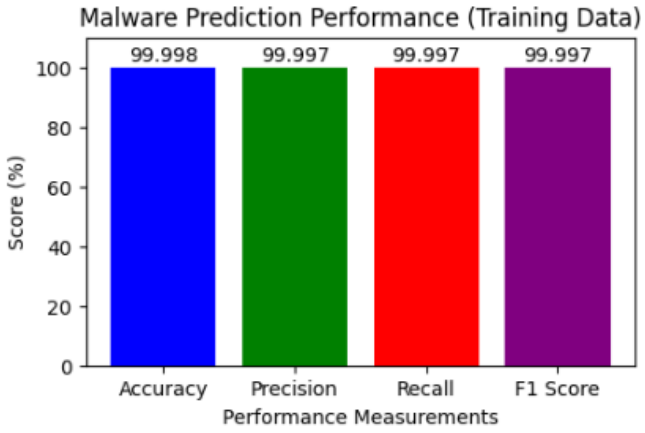


Figure 3: Performance Analysis Training Data

Result and analysis on testing data

Testing data are 27,610 out of 1,38,047 dataset which contain 19,250 malware and 8,360 legit. In testing data 19,177 data are correctly predict as a malware, 73 data are wrongly predicted as a malware, 54 data are wrongly predicted as a legit and 8,306 data are correctly predicted as a legit with the help of confusion matrix which is shown below

Table 6. Confusion Matrix for Testing Data

True Positive	False Positive	False Negative	True Negative
19177	73	54	8306

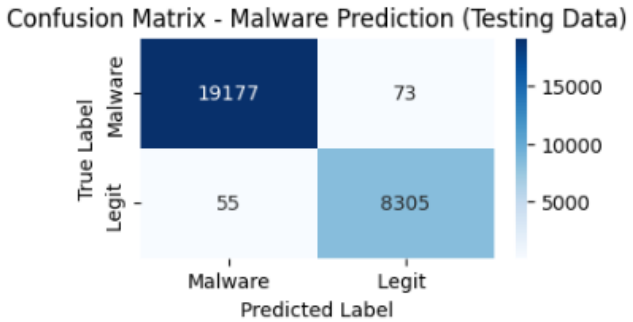


Figure 4: Confusion Matrix for Testing Data

The achieved accuracy, precision, recall and f1 score of the test model are 99.536%, 99.129%, 99.342% and 99.235% with the help of confusion matrix.

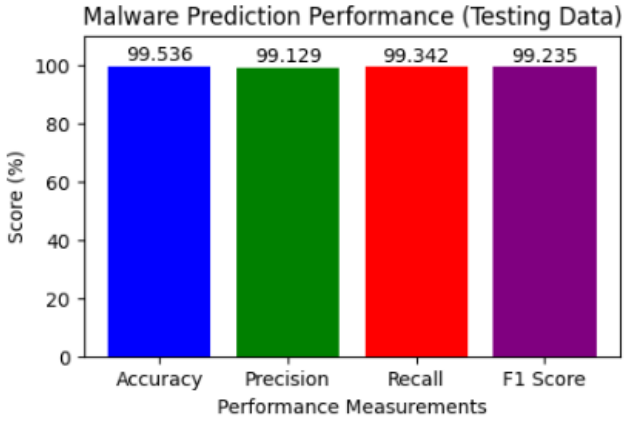


Figure 5: Performance Analysis Testing Data

Detection Result

At last 10 sample data are randomly selected from dataset which contain 5 legit and 5 malware data information. When sample data is pass into the system it detect 5 data as a malware and 5 as a legit which is show in below (1 => Legit and 0 => Malware).

Table 7. Malware Detection

Actual Output	Detected Output
1	1
1	1
0	0
1	1
0	0
1	1
1	1
0	0
0	0
0	0

VI. CONCLUSION

This study present the random forest algorithm based on machine learning technique to detect malware which has tested in 1,38,047 sizes of dataset. The experiment show how a software or program can be detected with the help of Random

Forest with high efficiency and accuracy. Additionally, the experimental result shows that the model is perfectly trained and achieved 99.99% and 99.54% accuracy for both trained and tested dataset followed by around 99% of precision, recall and f1 score.

REFERENCES

[1]P. Mell, K. A. Kent, and J. Nusbaum, "Guide to malware incident prevention and handling," National Institute of Standards and Technology, Gaithersburg, MD, 2005. doi: 10.6028/nist.sp.800-83.

[2]Khammas, B. M. (2020). Ransomware detection using random forest technique. ICT Express, 6(4), 325-331.

[3] Garcia, F. C. C., & Muga II, F. P. (2016). Random forest for malware classification. arXiv preprint arXiv:1609.07770.

[4]Sun, J., Yan, K., Liu, X., Yang, C., & Fu, Y. (2017, May). Malware detection on android smartphones using keywords vector and svm. In *2017 IEEE/ACIS 16th International Conference on Computer and Information Science (ICIS)* (pp. 833-838). IEEE.

[5]Pang, J., & Bian, J. (2019, October). Android malware detection based on naive bayes. In *2019 IEEE 10th International Conference on Software Engineering and Service Science (ICSESS)* (pp. 483-486). IEEE.

[6]Masum, M., & Shahriar, H. (2019, December). Droid-NNet: Deep learning neural network for android malware detection. In *2019 IEEE International Conference on Big Data (Big Data)* (pp. 5789-5793). IEEE.

[7]Aboaoja, F. A., Zainal, A., Ghaleb, F. A., Al-rimy, B. A. S., Eisa, T. A. E., & Elnour, A. A. H. (2022). Malware Detection Issues, Challenges, and Future Directions: A Survey. *Applied Sciences*, *12*(17), 8482. https://doi.org/10.3390/app12178482

[8]Gavrilut, D., Cimpoesu, M., Anton, D., & Ciortuz, L. (2009). *Malware Detection Using Machine Learning*. *4*.