







# A Machine Learning Framework for Automatic Detection of Malware

Syed Shabbeer Ahmad<sup>1</sup>, Atheequllah Khan<sup>1</sup>, Pankaj Kawadkar<sup>2</sup>, Imtiyaz Khan<sup>1</sup> ,  
Mummadi Upendra Kumar<sup>1</sup>  , and D. Shravani<sup>3</sup> 

<sup>1</sup> Department of CSE, MJCET, OU, Hyderabad, India  
{shabbeer.ahmad, atheequllah.khan, imtiyaz.khan,  
upendra.kumar}@mjcollege.ac.in

<sup>2</sup> Department of CSE, SSSUTMS, Sehere, India

<sup>3</sup> Department of CSE, Stanley College of Engineering and Technology for Women, OU,  
Hyderabad, India  
drdasarishravani@stanley.edu.in

**Abstract.** Cyberspace is every expanding with inclusion of diversified networks and systems. With the emerging technologies such as Internet of Things (IoT) and distributed computing, there is seamless integration of heterogeneous applications with interoperability. This has brought unprecedented use cases and applications in various domains. Unfortunately, there is every growing threat to cyberspace due to different kinds of malicious programs termed as malware. Since adversaries are developing various kinds of malware, its detection has become a challenging task. Of late, machine learning (ML) techniques are widely used to solve problems in real world applications. Plenty of supervised learning methods came into existence. The objective of this paper is to explore and evaluate different ML models with empirical study. In this paper, we proposed a ML framework for analysing performance of different prediction models. An algorithm known as Machine Learning based Automatic Malware Detection (ML-AMD) is proposed. This algorithm is used to realize the framework with supervised learning. This empirical study has resulted in knowledge about ML models such as Decision Tree (DT), Logistic Regression (LR), Random Forest (RF), Multilayer Perceptron (MLP) and Gradient Boosting (GB). Random Forest model has exhibited highest accuracy with 97.96%. The research outcomes in this paper help in triggering further investigations towards automatic detection of malware.

**Keywords:** Malware detection · Machine learning · Decision tree · Logistic regression · Random forest · Multilayer perceptron · Gradient boosting

## 1 Introduction

Malware is the malicious software that is created with bad intentions. It is often used to spread unwanted software that causes damage to systems. Adversaries are making business out of it and there are many incidents of it in the recent past. The term malware refers to software that damages devices, steals data, and causes chaos. There are many

types of malware—viruses, Trojans, spyware, ransomware, and more. With the availability of malware instances and signatures, it became easier to identify known malware. Machine learning domain provides required AI enabled techniques that can be exploited for malware detection. With the recent advancements in ML, it is possible to achieve near real time detection of malware [1]. It is observed from existing methods that there are many advantages of using ML techniques. The advantages include prediction accuracy, ability to use supervised learning samples and so on.

ML models are widely used for detection of malware with supervised learning. Such models are explored in [1, 5, 8–11, 14] and [15]. Gibert *et al.* [1] proposed a deep learning model for malware classification. It is made up of multiple models for efficient predictions. Karbab *et al.* [5] proposed a data-driven malware detection approach using ML techniques. They used behaviour analysis reports for their empirical study. Mahindru *et al.* [8] proposed a methodology for automatic Android malware detection using ML techniques. Hosseinzadeh *et al.* [9] proposed ML approaches that can be used for prediction of given disease. Chin *et al.* [10] also focused on DGA based machine learning models for malware detection. Chen *et al.* [11] used malware detection approach for Android malware using ML techniques. Masum *et al.* proposed a deep learning model for Android malware detection. The model is known as Droid-NNet. Usman *et al.* [14] focused on building an intelligent system for malware detection and that is associated with digital forensics. Singh *et al.* [15] used ML techniques to detect malware in executable files. From the review of literature, it is ascertained that machine learning models are very useful for creating artificial intelligence (AI) needed to detect malware automatically. There are many ML models available. However, it is important to analyse each model for its modus operandi and performance for making choices for building a real malware detection model. Our contributions in this paper are as follows.

1. We proposed a ML based framework for automatic detection of malware by exploiting many detection models using supervised learning.
2. An algorithm known as Machine Learning based Automatic Malware Detection (ML-AMD) is proposed to realize the framework.
3. We built a prototype for evaluation of the framework and the underlying models.
4. We have made performance analysis of the ML models that has resulted in various insights and capabilities of the models.

The remainder of the paper is structured as follows. Section 2 focused on the review of literature covering different ML models for malware detection. Section 3 presents the proposed methodology for automatic detection of malware. Section 4 presents experimental results while Sect. 5 concludes our work and bestows directions for future work.

## 2 Literature Review

This section review literature on existing methods for detection of malware. Gibert *et al.* [1] proposed a deep learning model for malware classification. It is made up of multiple models for efficient predictions. Li *et al.* [2] proposed a malware detection model based

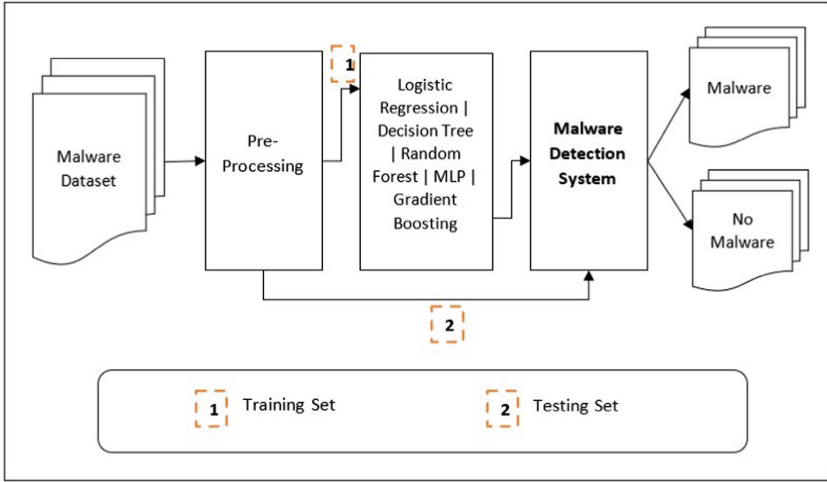
on Domain Generation Algorithm (DGA). It is based on machine learning techniques. Pei *et al.* [3] proposed a deep learning framework known as AMalNet based on CNN for malware detection. Karbab *et al.* [4] focused on Android malware detection by defining an automated framework using deep learning methods. Karbab *et al.* [5] proposed a data-driven malware detection approach using ML techniques. They used behaviour analysis reports for their empirical study. Wu [6] focused on a systematic study of malware detection methods based on deep learning. Jangam [7] explored deep learning, stacking and transfer learning methods for prediction purposes. Mahindru *et al.* [8] proposed a methodology for automatic Android malware detection using ML techniques. Hosseinzadeh *et al.* [9] proposed ML approaches that can be used for prediction of given disease. Chin *et al.* [10] also focused on DGA based machine learning models for malware detection. Chen *et al.* [11] used malware detection approach for Android malware using ML techniques. Masum *et al.* proposed a deep learning model for Android malware detection. The model is known as Droid-NNet.

Xiao *et al.* [13] defined a model based on deep learning behaviour graphs for malware detection. Usman *et al.* [14] focused on building an intelligent system for malware detection and that is associated with digital forensics. Singh *et al.* [15] used ML techniques to detect malware in executable files. Zhang *et al.* [16] focused on feature exploration using deep learning towards classification of Android malware. Alzaylaee *et al.* [17] proposed a deep learning framework known as DL-Droid for Android malware detection for real devices. Akarsh *et al.* [18] used deep learning and visualized the detection of malware and the classification results. Dib *et al.* [19] proposed multi-dimensional deep learning framework for malware classification in IoT environment. Kim *et al.* [20] focused on extraction of features along with multi-modal deep learning in order to achieve Android malware detection performance. Pektaş *et al.* [21] used opcode sequences and deep learning to detect Android malware. Gohari *et al.* [22] used network traffic based deep learning for Android malware detection and classification. Bawazeer *et al.* [25] exploited hardware performance counters to detect malware using ML models. Kamar *et al.* [26] investigated on several kinds of existing methods used to detect mobile malware.

From the review of literature, it is ascertained that machine learning models are very useful for creating artificial intelligence (AI) needed to detect malware automatically. There are many ML models available. However, it is important to analyse each model for its modus operandi and performance for making choices for building a real malware detection model.

### 3 Proposed Framework

We proposed a ML framework for automatic detection of malware. The framework, as presented in Fig. 1, takes malware dataset [27] and uses it for training and testing with 80:20 ratio in the form of pre-processing. The training dataset is subjected to learning a classifier. Different classifiers are exploited for malware detection. Logistic Regression, Decision Tree, Random Forest, MLP and Gradient Boosting are the prediction models used in the framework. After completion of training process, the ML models gain required intelligence to form a malware detection system which takes testing data and performs malware detection process resulting in classification of malware samples discriminated from genuine instances.



**Fig. 1.** ML framework for automatic malware detection

Logistic Regression is a statistical model which models a binary dependent variable by using a logistic function. It is also known as sigmoid function which is as given in Eq. 1.

$$F(x) = \frac{1}{1 + e^{-x}} = \frac{e^x}{e^x + 1} \tag{1}$$

This function helps the model to obtain values required by binary classification. If  $p(x)$ , an unbounded linear function, is assumed as linear function, probability is denoted by  $p$  which ranges from 0 to 1. To solve the problem, let  $\log p(x)$  is a linear function and  $\log p(x)/(1-p(x))$  is expressed as in Eq. 2.

$$\text{Log} \frac{p(x)}{1 - p(x)} = \alpha_0 + \alpha \cdot x \tag{2}$$

Once the problem of  $p(x)$  is solved, it can be expressed as in Eq. 3.

$$P(x) = \frac{e^{\alpha_0 + \alpha x}}{e^{\alpha_0 + \alpha x} + 1} \tag{3}$$

In order to make logistic regression as a linear function there is need for a threshold which is set to 0.5 and rate of misclassification is minimized.

Decision Tree is another algorithm used in the proposed framework. It models given data in the form of a tree so as to converge into useful decisions. In order words, it solves given problem with tree representation of data. It makes use of two important measures known as entropy and Gini index. Entropy is computed as in Eq. 4.

$$\text{Entropy} = - \sum_{i=1}^n p_i * \log(p_i) \tag{4}$$

Gini index is another measuring for knowing inequality. It results in a value between 0 and 1. Lowest value indicates homogenous elements while higher value indicates

heterogeneous elements indicating maximum inequality. This measure reflects sum of the square of probabilities associated with each class. It is computed as in Eq. 5.

$$\text{Gini index} = 1 - \sum_{i=1}^n p_i^2 \quad (5)$$

Random Forest is another popular ML technique. It makes use of many decision trees internally. It gets predictions of all trees and make a final decision. Multilayer perceptron, on the other hand, is an Artificial Neural Network (ANN) variant. It has input layer, output layer and at least one hidden layer. Its important computations are as in Eq. 6 and Eq. 7.

$$h^1 = \text{step}(z^1) = \text{step}(w^1 \cdot x + b^1) \quad (6)$$

$$y = \text{step}(z^2) = \text{step}(w^2 \cdot h^1 + b^2) \quad (7)$$

Any ANN is generally trained in batches. Each input value in the batch is a vector denoted as X. From the available m instances, k instances are derived as in Eq. 8.

$$x_1 = \begin{pmatrix} x_{1,1} \\ \dots \\ x_{1,n} \end{pmatrix}, \dots, x_k = \begin{pmatrix} x_{k,1} \\ \dots \\ x_{k,n} \end{pmatrix} \quad (8)$$

Then the k instances are combined as in Eq. 9.

$$X = \begin{pmatrix} x_1^T \\ \dots \\ x_k^T \end{pmatrix} = \begin{pmatrix} x_{1,1} & \dots & x_{1,n} \\ \dots & \dots & \dots \\ x_{k,1} & \dots & x_{k,n} \end{pmatrix} \quad (9)$$

Provided this, the computation of y is changed to the expression in Eq. 10.

$$y = \text{step}(z) = \text{step}(X \cdot W + b) \quad (10)$$

X is an input with shape (k,n) where number of input values is denoted as n while k denotes number of instances. W is nothing but a matrix with shape (n, u).

Gradient boosting is another algorithm used in the empirical study. Here new trees are built with significant difference. It has ensemble concept in machine learning which tries to reduce bias. The model is built as expressed in Eq. 11.

$$f_0(x) = \text{argmin}_{A_\gamma} L(y, \hat{y}) \quad (11)$$

where  $f_0(x)$  is the model to be built, gamma is the predicted value while y denotes actual value. Loss function is denoted as L. Now an improved model, expressed in Eq. 12, is used where residual is computed.

$$f_1(x) = f_0(x) + h_1(x) \quad (12)$$

The above process is repeated for several times. The general form of expression to be carried out at each iteration is as in Eq. 13.

$$f_m(x) = f_{m-1}(x) + h_m(x) \quad (13)$$

It is the general form for boosting algorithm. It is meant for classification of malware test samples in this paper. It exploits ensemble approach in order to have improved performance. In the framework shown in Fig. 1, different ML models are used in pipeline and the performance of the models is evaluated.

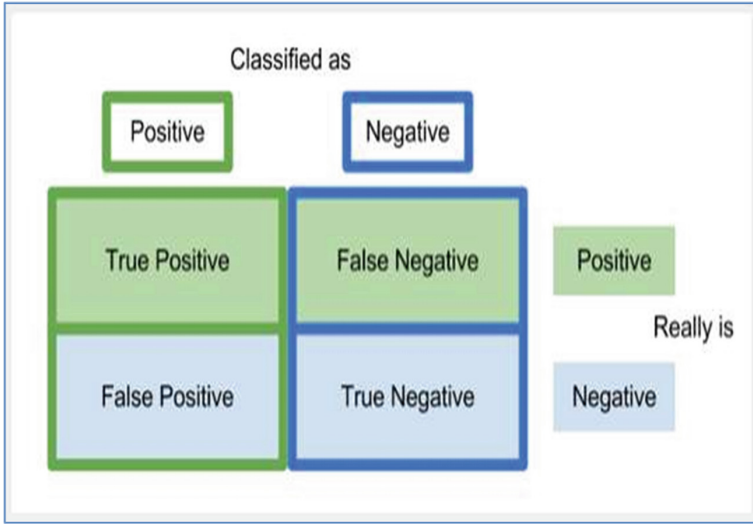


Fig. 2. Confusion matrix to evaluate models

As presented in Fig. 2, confusion matrix is used to derive many metrics for performance evaluation. The metrics are as in Table 1.

Table 1. Performance metrics used for evaluation

Metric	Formula	Value range	Best value
Precision (p)	$\frac{TP}{TP+FP}$	[0; 1]	1
Recall (r)	$\frac{TP}{TP+FN}$	[0; 1]	1
Accuracy	$\frac{TP+TN}{TP+TN+FP+FN}$	[0; 1]	1
F1-Score	$2 * \frac{(p*r)}{(p+r)}$	[0; 1]	1

Each measure has its value range between 0 and 1 indicating least and highest performance in prediction. An algorithm known as Machine Learning based Automatic Malware Detection (ML-AMD) is proposed to realize the framework.

**Algorithm:** Machine Learning based Automatic Malware Detection (ML-AMD)

**Inputs:**  $D$  (dataset),  $M$  (ML models)

**Output:** Predictions  $P$

1. Start
2. Initialize map  $R$  for results
3.  $(T1, T2) \leftarrow \text{PreProcess}(D)$
4.  $F \leftarrow \text{FeatureExtraction}(T1)$
5. For each model  $m$  in  $M$
6.   Train the model  $m$  using  $F$
7.    $\text{results} \leftarrow \text{FitTheModel}(m, T2)$
8.   Update  $R$  with the model and results
9. End For
10. For each  $r$  in  $R$
11.   Print results
12. End For
13. End

**Algorithm 1.** Machine Learning based Automatic Malware Detection (ML-AMD)

As presented in Algorithm 1, it takes given malware dataset and set of ML models as inputs. Then it performs pre-processing of the dataset. It follows supervised learning approach to train each model present in the pipeline. Prior to training process, it has provision for feature extraction to get useful features. Each model is trained with the training set and then test set is used to perform prediction process.

## 4 Experimental Results

This section presents results of experiments. Each model is evaluated and their ability to discriminate between genuine and malware instances is recorded. Then different metrics are used to evaluate their performance. The performance of different ML models in detection of malware is evaluated in this section.

**Table 2.** Performance of different prediction models

Prediction model	TP	TN	FP	FN
Logistic regression	640	1400	13	70
Decision tree	680	1400	15	37
Random forest	670	1400	9	34
MLP	670	1400	23	35
Gradient Boosting Classifier	660	1400	7	48

As presented in Table 2, the malware prediction models along with their prediction values in terms of TP, TN, FP and FN are provided. Based on these values, different metrics are computed. For instance, the computation of performance metrics for Logistic Regression is given below. Computations are made as per equations given in Table 1.

$$\text{Precision} = 640/640 + 13 = 0.98$$

$$\text{Recall} = 640/640 + 70 = 0.90$$

$$\text{F1 - Score} = 2 \times 0.98 * 0.90/0.98 + 0.90 = 0.94$$

$$\text{Accuracy} = 640 + 1400/640 + 1400 + 13 + 70 = 0.96$$

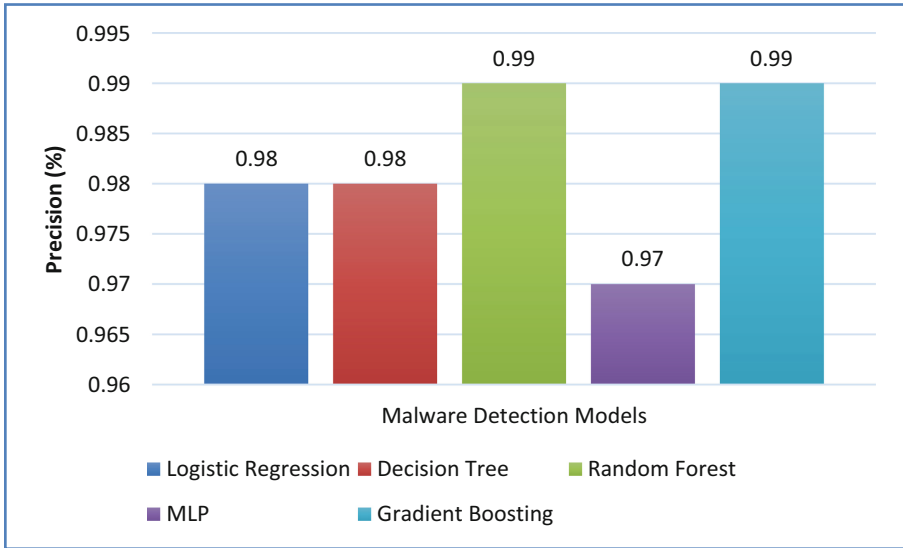
In this fashion, each model’s performance computed and the results are shown in Table 3.

**Table 3.** Malware detection performance comparison

Malware detection model	Performance (%)			
	Precision	Recall	F1-Score	Accuracy
Logistic regression	0.98	0.90	0.94	0.96
Decision tree	0.98	0.95	0.97	0.9777
Random forest	0.99	0.95	0.97	<b>0.9796</b>
MLP	0.97	0.95	0.96	0.97248
Gradient boosting	0.99	0.93	0.96	0.9739

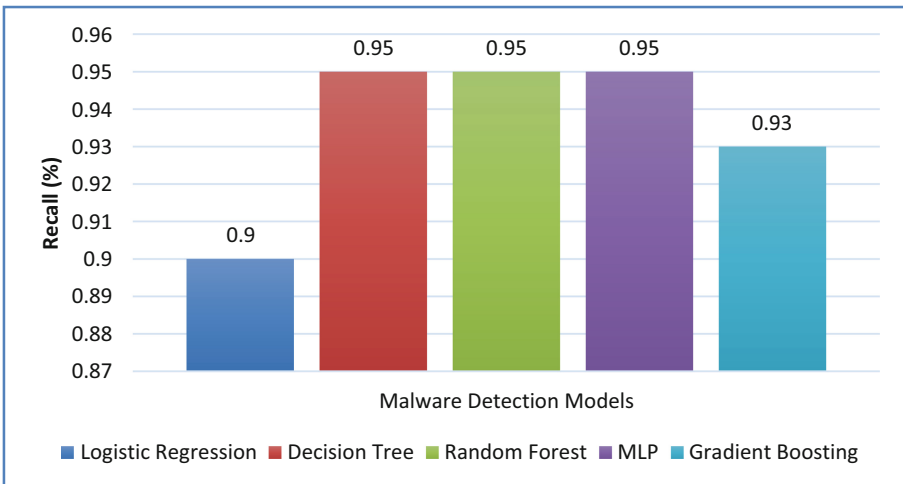
Each model is evaluated with performance metrics reflecting the capabilities of the prediction models in malware detection.





**Fig. 3.** Precision comparison

As presented in Fig. 3, different malware prediction models are evaluated in terms of their precision performance. Each model is found to have different capabilities in malware prediction. Highest precision is achieved by two models such as Random Forest and Gradient Boosting with 99%. Least precision performance is exhibited by MLP with 97%. Logistic Regression and Decision Tree showed 98%.



**Fig. 4.** Recall comparison

As presented in Fig. 4, different malware prediction models are evaluated in terms of their recall performance. Each model is found to have different capabilities in malware prediction. Highest recall is achieved by three models such as Random Forest, MLP and Decision Tree with 95%. Least recall performance is exhibited by Logistic Regression with 90%. Gradient Boosting has showed 98% recall performance.

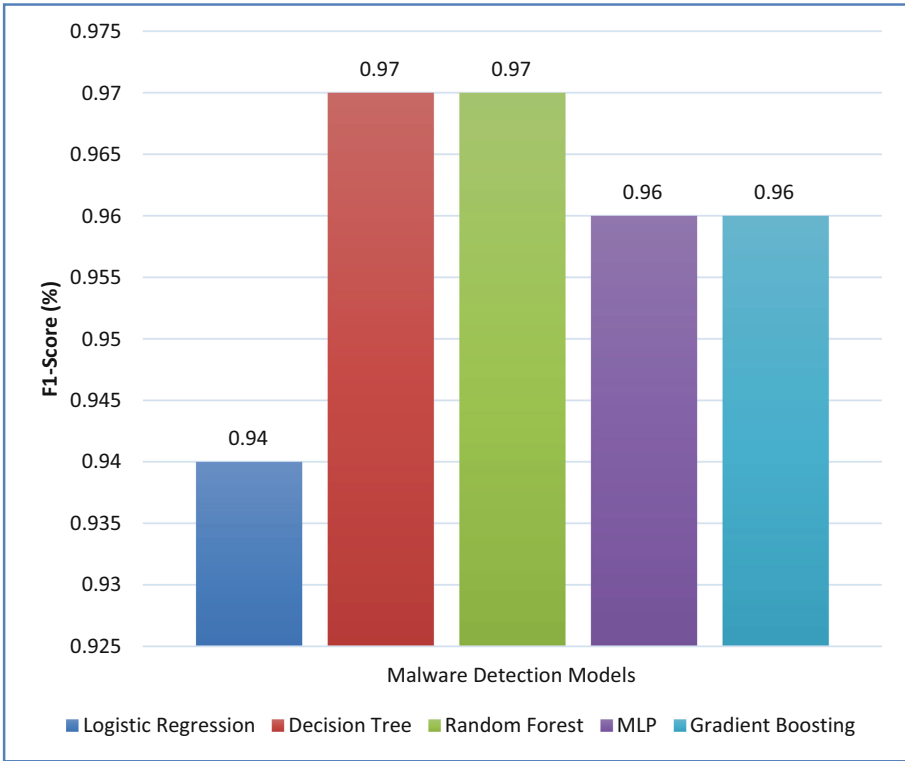
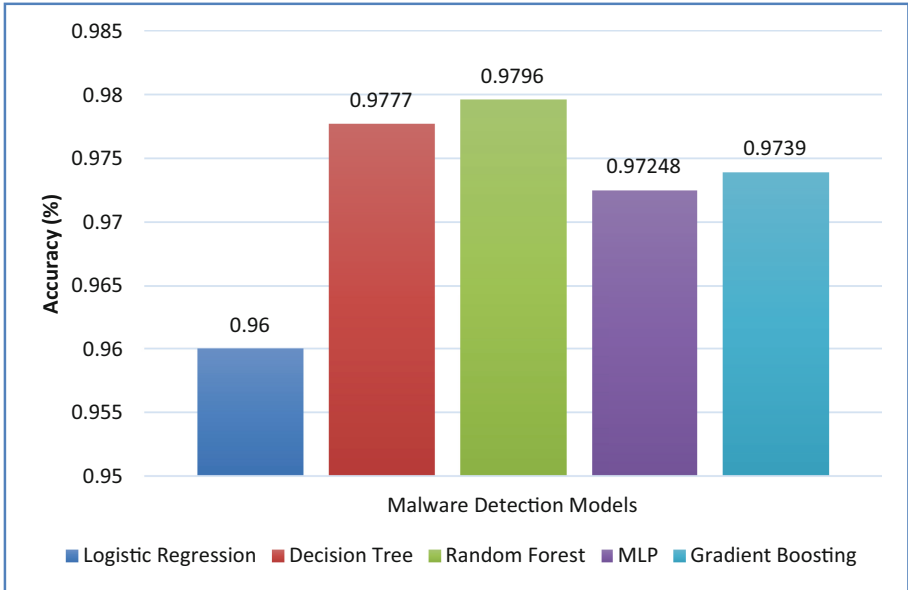


Fig. 5. F1-Score comparison

As presented in Fig. 5, different malware prediction models are evaluated in terms of their F1-Score performance. Each model is found to have different capabilities in malware prediction. Highest F1-Score is achieved by two models such as Random Forest and Decision Tree with 97%. Least F1-Score performance is exhibited by Logistic Regression with 94%. Gradient Boosting and MLP have showed 96% F1-Score performance.



**Fig. 6.** Accuracy comparison

As presented in Fig. 6, different malware prediction models are evaluated in terms of their accuracy. Each model is found to have different capabilities in malware prediction. Highest accuracy is achieved by Random Forest with 97.96%. Least accuracy is exhibited by Logistic Regression with 96%. Gradient Boosting showed 97.39% accuracy while MLP showed 97.24% accuracy.

## 5 Conclusion and Future Work

In this paper, we proposed a ML framework for analysing performance of different prediction models. This empirical study has resulted in knowledge about ML models such as Decision Tree (DT), Logistic Regression (LR), Random Forest (RF), Multilayer Perceptron (MLP) and Gradient Boosting (GB). An algorithm known as Machine Learning based Automatic Malware Detection (ML-AMD) is proposed. This algorithm is used to realize the framework with supervised learning. It exploits a pipeline of aforementioned ML models to evaluate their performance in malware detection. Our experimental results revealed the utility of various ML models. Performance of different models are evaluated in terms of precision, recall, F1-Score and accuracy. Random Forest model has exhibited highest accuracy with 97.96%. The research outcomes in this paper help in triggering further investigations towards automatic detection of malware. In future, we explore deep learning models for malware detection as they have the capacity to have in-depth learning of features from data leading to improved prediction performance.

## References

1. Gibert, D., Mateu, C., Planes, J.: HYDRA: a multimodal deep learning framework for malware classification. *Comput. Secur.* **95**, 1–47 (2020)
2. Li, Y., Xiong, K., Chin, T., Hu, C.: A machine learning framework for domain generation algorithm (DGA)-based malware detection. *IEEE Access* (2019)
3. Pei, X., Yu, L., Tian, S.: AMalNet: a deep learning framework based on graph convolutional networks for malware detection. *Comput. Secur.* **93**, 1–21 (2020)
4. Karbab, E.B., Debbabi, M., Derhab, A., Mouheb, D.: MalDozer: automatic framework for android malware detection using deep learning. *Digit. Invest.* **24**, pS48–S59 (2018)
5. Karbab, E.B., Debbabi, M.: MalDy: portable, data-driven malware detection using natural language processing and machine learning techniques on behavioral analysis reports. *Digit. Invest.* **28**, pS77–S87 (2019)
6. Wu, H.: A systematical study for deep learning based android malware detection. In: *Proceedings of the 2020 9th International Conference on Software and Computer Applications*, pp. 1–6 (2020)
7. Jangam, E., Barreto, A.A.D., Annavarapu, C.S.R.: Automatic detection of COVID-19 from chest CT scan and chest X-Rays images using deep learning, transfer learning and stacking. *Appl. Intell.* **52**(2), 2243–2259 (2021). <https://doi.org/10.1007/s10489-021-02393-4>
8. Mahindru, A., Sangal, A.L.: MLDroidâ framework for Android malware detection using machine learning techniques. *Neural Comput. Appl.*, 1–58 (2020)
9. Sara, H.K., Peyman, H.K., Wesolowskic, M.J., Schneidera, K.A., Detersa, R.: Automatic detection of coronavirus disease (COVID-19) in X-ray and CT images: a machine learning based approach. *Biocybern. Biomed. Eng.*, 1–13 (2021)
10. Chin, T., Xiong, K., Hu, C., Li, Y.: A machine learning framework for studying domain generation algorithm (DGA)-based malware. *Secur. Priv. Commun. Netw.*, 433–448 (2018)
11. Chen, X., et al.: Android HIV: a study of repackaging malware for evading machine-learning detection. *IEEE Trans. Inf. Forens. Secur.*, 1–15 (2019)
12. Masum, M., Shahriar, H.: *IEEE 2019 IEEE International Conference on Big Data (Big Data)*, Los Angeles, CA, USA, 9–12 December 2019, pp. 5789–5793 (2019)
13. Xiao, F., Lin, Z., Sun, Y., Ma, Y.: Malware detection based on deep learning of behavior graphs. *Math. Probl. Eng.* **2019**, 1–10 (2019)
14. Usman, N., Usman, S., Khan, F., Jan, M.A., Sajid, A., Alazab, M., Watters, P.: Intelligent dynamic malware detection using machine learning in ip reputation for forensics data analytics. *Future Gener. Comput. Syst.*, 1–18 (2021)
15. Singh, J., Singh, J.: A survey on machine learning-based malware detection in executable files. *J. Syst. Arch.*, 1–24 (2020)
16. Zhang, N., Tan, Y., Yang, C., Li, Y.: Deep learning feature exploration for Android malware detection. *Appl. Soft Comput.*, 1–7 (2021)
17. Alzaylaee, M.K., Yerima, S.Y., Sezer, S.: DL-droid: deep learning based android malware detection using real devices. *Comput. Secur.*, 1–28 (2019)
18. Akarsh, S., Simran, K., Poornachandran, P., Menon, V.K., Soman, K.P.: *IEEE 2019 5th International Conference on Advanced Computing & Communication Systems (ICACCS) - Coimbatore, India*, 15–16 March 2019, pp. 1059–1063 (2019)
19. Dib, M., Torabi, S., Bou-Harb, E., Assi, C.: A multi-dimensional deep learning framework for IoT malware classification and family attribution. *IEEE Trans. Netw. Serv. Manag.* **18**(2), 1165–1177 (2021)
20. Kim, T.G., Kang, B.J., Rho, M., Sezer, S., Im, E.G.: A multimodal deep learning method for android malware detection using various features. *IEEE Trans. Inf. Forens. Secur.*, 1–16 (2018)

21. Pektaş, A., Acarman, T.: Deep LEARNING to detect android malware via opcode sequences. *Neurocomputing*, 1–21 (2019)
22. Gohari, M., Hashemi, S., Abdi, L.: Android malware detection and classification based on network traffic using deep learning. In: 2021 7th International Conference on Web Research (ICWR), pp. 1–7 (2021)
23. Chandrashekar, G., Sahin, F.: A survey on feature selection methods. *Comput. Electr. Eng.* **40**(1), 16–28 (2014). <https://doi.org/10.1016/j.compeleceng.2013.11.024>
24. Karunakaran, V., Rajasekar, V., Joseph, S.I.T.: Exploring a filter and wrapper feature selection techniques in machine learning. In: Smys, S., Tavares, J.M.R.S., Bestak, R., Shi, F. (eds.) *Computational Vision and Bio-Inspired Computing*. AISC, vol. 1318, pp. 497–506. Springer, Singapore (2021). [https://doi.org/10.1007/978-981-33-6862-0\\_40](https://doi.org/10.1007/978-981-33-6862-0_40)
25. Bawazeer, O., Helmy, T., Al-Hadhrami, S.: Malware detection using machine learning algorithms based on hardware performance counters: analysis and simulation. *J. Phys: Conf. Ser.* **1962**, 012010 (2021). <https://doi.org/10.1088/1742-6596/1962/1/012010>
26. Kamar, M.E.Z.N., Esmaeilzadeh, A., Kim, Y., Taghva, K.: A survey on mobile malware detection methods using machine learning. In: 2022 IEEE 12th Annual Computing and Communication Workshop and Conference (CCWC), pp. 0215–0221 (2022). <https://doi.org/10.1109/CCWC54503.2022.9720753>
27. Malware Exploratory Dataset. <https://www.kaggle.com/code/lucaslba/malware-exploratory/data>