International Journal of Communication Networks and Information Security
2024, 16(4)
ISSN: 2073-607X,2076-0930
https://ijcnis.org/

Research Article

Optimized Feature Selection and classification for Non-Portable Executable Malware

Tukkappa K Gundoor[1], Dr. Sridevi[2]
[1]Department of Computer Science, Karnatak University, Dharwad, India,
E-mail: tukkappa@kud.ac.in
[2]Department of Computer Science, Karnatak University, Dharwad, India,
E-mail: sridevi@kud.ac.in

ARTICLE INFO

Received: 26 April 2024
Accepted: 03 September 2024

ABSTRACT

Malware is a program that executes harmful acts and steals information. nowadays it is widely recognized as oneof the largest hazards. In this research work machine learning is used to identify and detect Non-PE file features. The variousdistinct aspects of the Non-PE files features can correlate with one another, being clean or affected, led to the identification ofsuch features. by using machine learning algorithms such as Ada Boost Classifier,Gaussian NB, KNClassifier,RF Classifier, SGD classifier, and feature selection produced the best detection rate also Prediction accuracy of thealgorithms is used to compare the efficacy and efficiency.

Keywords: Benign, Classifiers, Feature selection,Malware, Non-portable malware, Random Forest.

INTRODUCTION

Malicious software also known as malware is a serious issue forCommercialand personal computing. To avoid detection malware changes quickly when it affects to system. Antivirus engines must work harder to detect new malware, some of which simply be versions of already-known malware False positives are more likely if antivirus signatures are too general, while several varieties of the same malware may go undetected if the signatures are too specialized. According to theoretical findings, no Antivirus (AV) tool can find every computer infection. An antivirus program's job is to identify as many computer viruses as it can, but before it can identify a previously unidentified program as malware, it must determine if it is potentially harmful or clean. a software that repeatedly and explicitly copies a potentially upgraded version of itself. Worms, Trojan horses, spyware, rootkits, and logic bombs can all be classified as viruses. The selection of novel features in malware samples that are not the result of a virus infection is the focus of this research. This workaddresses the issue of correctly classifying, detecting, and repairing applications that start innocuous but end up contaminated. The group of infections are looking to find is referred to as malware. One of the main problems in computer security is the proliferation of malware varieties, making it difficult to identify and choose features for malware. Using machinelearning algorithms tried to attempt to solve these issues.
Features from both dangerous and benign Windows executable files can be found in the Non-PE malware dataset collection.This training file is generated based on the static and dynamic features from Windows executable (binary hexadecimal + DLL calls). There are 365 Non-PE malware samples in the dataset, where 301 are malicious files and 64 are benign.

RELATED WORK

It has been demonstrated that the HFR i.e., Hybrid Feature Retrieval model a recently described novel model can successfully find the malicious executable. The Assembly Feature Retrieval algorithm also known as the derived assembly features were used to find the best assembly a series of instructions to represent the selected features of binary n-grams. The resultant feature set is referred to as the hybrid feature set since these features were combined with the features of the DLL function call (HFS)[1], [2]. The predicted accuracy of the algorithms is one performance parameter that is usedto compare their efficacy and efficiency. Observedthat supervised learning with feature selection producedthe highest detection rates[3], [4]. To categorize malware, supervised-learning algorithms can be fed an architecture that uses reinforcement learning to choose highly distinct

attributes.DQFSA(Deep Q-learning based Feature Selection Architecture) is a kind of architecture where the principal part is an AI agent which can interact with sample feature space without human engagement and mostly select reasonable features intentionally via deep reinforcement learning[5]. The RF approach has the greatest accuracy and 91.300% F1 scorewhen it comes to classifying malware detection.The model performance and accuracy can be further enhanced in the future by choosing features with evolutionary algorithms and training the model with deep learning models[5]. PCA (Principal component analysis) is a feature selection approach used in ANN-based models with the application of malware detection in PDF. Also, Analysis demonstrates that the model using PCA can greatly minimize feature redundancy for both training and testing outcomes. The model with 32 principal feature components, out of the examined models employing PCA confirms the efficacy and demonstrates the model's ability to reach a 93.17% true positive rate (TPR) while keeping the seven tested popular commercial antivirus (AV) scanners. Index Terms like malicious documents, PDF viruses, machine learning, artificial neural network analysis, and cyber security[6], [7].

METHODOLOGY

The methodology used for malware feature selection and classification to determine the accuracy of malware detection shown Figure 1.

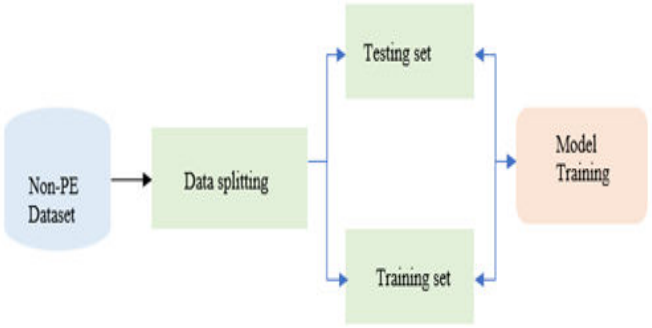

Figure 1. Methodology of malware Detection

Dataset

Loading Non -PE malware data to train the model, the act of loading data or data sets from a source file is known as data loading. importing it into a data processing or storage application isa common procedure. Extraction employs data-loading activities based on databases.

Data Splitting

A set of data is divided into two or more subgroups through the process of data splitting test set of data is 30% and the training set is 70% of the dataset used. It is typical to practice training the model in such a way that the data is split into two-part one part is used for testing or analysing the data. The other part of the split is used for training. When creating data-driven models particularly data separation is very crucial. With the help of the splitting technique data model development and other operations, such machine learning that relies on data models may be done accurately[3], [8].

Model Training

Model training is a process of putting data into an algorithm in machine learning. The program can learn and find the best values for all associated attributes through this approach. The Random Forest, KNeighbors, Ada Boost,SGD, and GaussianNB classifiers are used as machinelearning models in this work [3], [8]–[11].

IMPLEMENTATION AND RESULTS

Preliminary data analysis

Preliminary data analysis involves summarising the data using statistical and visualization techniques to focus on key data points for implementation. it is necessary to characterize and summarise the dataset while examining it from various aspects without making any assumptions about its content shown in Figure 2. Perform exploratory data analysis to ensure the data is accurate and free of evident flaws before beginning statistical modeling or machine learning.

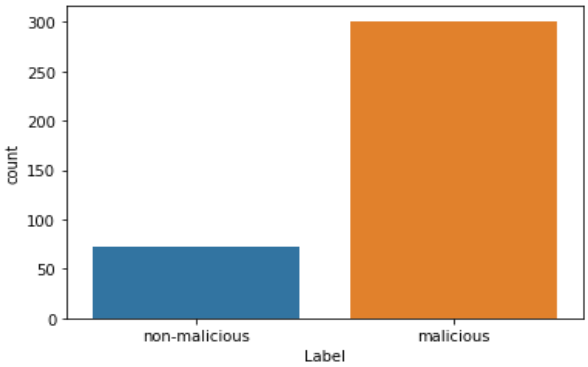

Figure 2. Distribution of Malware.

List of FeaturesIncluded in Dataset

The overall samples in the dataset are 365 of in which 301 are malicious files and 64 are not. The collection contains more malware samples than typical samples.Shown in below Table 1, And a label column indicating if the file graduated todangerous. This is because the binary hexadecimal feature names are not easy to be encoded in itself, this also holds for any included DLL calls[4], [10], [13]–[19].

Evaluation matrix of the performance result.

The machine learning algorithm's evaluation is an essential part of any model.To do this the various matrices are employed as shown following.

Table 1. List of features included

Features	Value	Description
OLE Signature	docf11e0a1b11ae1	Should be docf11e0a1b11ae1
Minor Version	0003E	Must be 003E
Major Version	00003	Must be 3 or 4
Bit Order	0FFFE	Has to be FFFE (little endian)
Sector Switch	00009	Must be 0009 or 0000C
Initial Dir Sector	00000001	(hex)
Transaction Signature Number	0	Has to be 0
Mini Stream is stopped Must be	04096	4096 bytes
initial Sector Mini FAT	00000003C	Hex
Initially DIFAT Sector	0FFFFFFFE	Hex
Segment Size in bytes	512	Must be 512 or 4096 bytes
Real-time File Size (Bytes)	056320	True file size stored on disc
File Size Maximum in FAT	66048.0	the largest file size allowed by FAT
Information beyond FAT	0	only if the file size exceeds FAT coverage
FAT offset for more data	0000DC00	Offset of the first free sector at the conclusion of FAT

Accuracy

The ratio of accurate predictions to all input samples is assessed in order to determine the prediction's accuracy. [20]–[25].

$$\text{Accuracy} = \frac{\text{Total No.of AccuratePredictions}}{\text{Total Amount Of Predictions}}$$

- **F1 Score:** A test's accuracy is evaluated using the F1 Score.

$$F1=2*\frac{1}{\frac{1}{\text{Precision}} + \frac{1}{\text{recall}}}$$

Precision and recall are balanced to some extent in the F1 Score.

- **Precision:** That is decided by splitting the overall no. of successful outcomes by overall number of successes that the classifier correctly identified as successful outcomes.

$$\text{Precision}= \frac{TP+FP}{TP}$$

- **Recall:** To compute the overall number of pertinent samples, divide it by the total number of consistently good outcomes.

$$\text{Recall}= \frac{TP+FN}{TP}$$

- **Support:** Observations made for each class, in number.
- **Macro average:** The metric's arithmetic means between the two classes.
- **Weighted average:** By dividing the total by the no. of items the WA is determined. (Metric of interest x weight) by sum(weights).

RESULTS

For the purpose of teaching machine learning classifiers, experiments are conducted. Training data makes up 70% of the datasets, while testing data makes up 30%. The KNeighbors (KN), Random Forest (RF), AdaBoostClassifier, SGDClassifier, and GaussianNB Malware is identified using machine learning techniques.
In the first instance, the machine learning classifiers use the results as input.Adding or deleting features from the feature list, the accuracy score and classifier are utilised as a validation mechanism. The machine learning classifiers receive inputfrom the chosen features. The machine learning model performance data for the malware datasets is shown in Table 2 for both settings. Performance measures like recall, accuracy, precision, and f1-score are used to assess the results. Draw the conclusion that choosing features produced significantly better results for malware detection based on the table. In comparison to other machine learning models, the AdaBoost classifier and GaussianNB display the highest results for all performance parameters. AdaBoost and GaussianNB Classifier for the dataset exhibit 99.10% accuracy,0.99% f1-score, 0.99% recall, and 0.99% precision.

Table 2. Performance Results for dataset.

Models	Label	Precision	Recall	F1 Score	Support	Accuracy
RF	Benign	0.99	0.99	0.99	85	98.21%.
	Non-PE malicious	0.96	0.96	0.96	27	
	Avg Macro	0.98	0.98	0.98	112	
	Avg Weighted	0.98	0.98	0.98	112	
KN	Benign	0.97	0.98	0.99	85	96.42%.
	Non-PE malicious	0.96	0.89	0.92	27	
	Average Macro	0.96	0.94	0.95	112	
	Average Weighted	0.96	0.96	0.96	112	
AdaBoostClassifier	Benign	1.00	0.99	0.99	85	99.10%.
	Non-PE malicious	0.96	1.00	0.98	27	
	Average Macro	0.99	0.99	0.99	112	
	Average Weighted	0.99	0.99	0.99	112	
SGDClassifier	Benign	0.99	0.99	0.99	85	99.10%
	Non-PE malicious	0.96	0.96	0.96	27	
	AvgMacro	0.98	0.98	0.98	112	
	AvgWeighted	0.98	0.98	0.98	112	
GaussianNB	Benign	1.00	0.99	0.99	85	99.10%
	Non-PE malicious	0.96	1.00	0.98	27	
	Average Macro	0.98	0.99	0.99	112	
	Avg Weighted	0.99	0.99	0.99	112	

Confusion Matrix of the models

Confusion matrix to comprehend the model predictions in a comprehensive way. A matrix summarising the model's overall performance is produced by the Confusion Matrix.

$$\text{Accuracy}= \frac{TP+TN}{\text{TotalSample}}$$

Confusion Matrix generates a matrix that summarizes the model's overall performance.

TP: Positive is the actual value as predicted by the model.
TN: The real values are negative as predicted by the model.
Total Sample: Total Number of samples used in datasets.
FP: Despite what the model anticipated the actual number is negative (Type I error) (False Positive).
FN: The actual value is positive despite the model's negative prediction (Types II error) (False Negative).

The confusion matrix for the top-performing classifiers using particular features on datasets is displayed in Table 3.The table presents classification model evaluation metrics,False Positive (FP), True Negative (TN), False Negative (FN), and True Positive (TP). On a dataset with 75 occurrences, the performance of the classifiers Random Forest (RF), K-Nearest Neighbors (KN), AdaBoost, Stochastic Gradient Descent (SGD), and Gaussian Naive Bayes (GaussianNB) is displayed in the results.

Table 3. Confusion Matrix for Training Models

.

Model	TN	FP	FN	TP
RF Classifier	75	0.89	0.89	23.21
KN Classifier	75	0.89	2.68	23.21
AdaBoostClassifier	75	0.89	0.00	24.11
SGDClassifier	75	0.89	0.89	23.21
GaussianNB Classifier	75	0.89	0.00	24.11

Final results with proposed work.
The system must be trained with both useful and potentially dangerous data. As a result, numerous classifier models are evaluated and trained. Machine learning techniques can be used to train a classifier to automatically produce accurate predictions. Table 8 illustrates how well classifier models Random Forest, SGD, KNN, AdaBoostClassifier, and Gaussian NB perform in relation to the amount of labeled data they are exposed to.The classifier is given a set of fresh files during the validation stage, some of which are hazardous and some of which are not. the RF, SGD, KN, AdaBoost, and Gaussian NB models. Presented the findings of an experimental assessment of proposed method for identifying and categorizing malware. It is used for testing after producing a malware and cleanware dataset. RF, kNN, AdaBoost, SGD, and the Gaussian NB Classifier are just a few of the machine learning algorithms they employ.

Table 4. Proposed Work Accuracy

Machine Learning Models	Previous Work	Proposed Method
RandomForestClassifier [8]	97.74%	98%
KNeighborsClassifier [9][10]	--	96%
AdaBoostClassifier [11][12]	98.87%,99%	99%
SGDClassifier [13][14]	99%,85.05%	99%
GaussianNB	--	99%

CONCLUSION

This work layered the RF, AdaBoost,KNN, SGD, and Gaussian NB models for non-PE malicious malware detection to resolve theshortcomings of the structure of human features andthe constraints of the current learning approach. TheRF Classifier, KNeighborsClassifier, AdaBoostClassifier, SGDClassifier,and GaussianNB. Achieved an F1-score of 0.99%, of all models. The accuracy of detecting malware was greatly enhanced by the proposed kNN, AdaBoost, RF,SGD,and the Gaussian NB Classifier during testing the accuracy was 96.42%, 98.21%, 99.10%, 99.10%, 99.10%. To evaluate a machine learning algorithm researcher, separate data into training and test sets. The test set is used to measure how well the system performs when employing the newly learnt function, while the training set is used to teach the algorithm the required function. This will speed up

malware detection reduce the number of false positives detected by ML algorithms and enhance malware detection rates.

ACKNOWLEDGMENT
Mr. Tukkappa K. Gundoor would like to thank the Department of Science and Technology of Karnataka (DST) for funding our research with Ph.D. fellowship No. DST/KSTePS/Ph.D.Fellowship/PHY-02:2020-21.

Reference

[1] M. M. Masud, L. Khan, and B. Thuraisingham, "A hybrid model to detect malicious executables," IEEE Int. Conf. Commun., pp. 1443–1448, 2007, doi: 10.1109/ICC.2007.242.
[2] A. S. Bist and A. Sharma, "Feature Selection in Metamorphic Virus Detection : a Review," vol. 14, no. 1, pp. 30–45, 2016.
[3] M. Ahmadi, D. Ulyanov, S. Semenov, M. Trofimov, and G. Giacinto, "Novel feature extraction, selection and fusion for effective malware family classification," CODASPY 2016 - Proc. 6th ACM Conf. Data Appl. Secur. Priv., pp. 183–194, 2016, doi: 10.1145/2857705.2857713.
[4] M. Mays, N. Drabinsky, and S. Brandle, "Feature selection for malware classification," 28th Mod. Artif. Intell. Cogn. Sci. Conf. MAICS 2017, pp. 165–170, 2017.
[5] Z. Fang, J. Wang, J. Geng, and X. Kan, "Feature Selection for Malware Detection Based on Reinforcement Learning," IEEE Access, vol. 7, pp. 176177–176187, 2019, doi: 10.1109/ACCESS.2019.2957429.
[7] P. Harshalatha and R. Mohanasundaram, "Classification of malware detection using machine learning algorithms: A survey," Int. J. Sci. Technol. Res., vol. 9, no. 2, pp. 1796–1802, 2020.
[8] Q. Jiang, X. Zhao, and K. Huang, "A feature selection method for malware detection," 2011 IEEE Int. Conf. Inf. Autom. ICIA 2011, no. June, pp. 890–895, 2011, doi: 10.1109/ICINFA.2011.5949122.
[9] B. B. Rad and M. K. H. Nejad, "Feature selection for malware classification and detection: A literature review," Int. J. Adv. Trends Comput. Sci. Eng., vol. 9, no. 1.1 Special Issue, pp. 571–577, 2020, [Online]. Available: https://www.scopus.com/inward/record.uri?partnerID=HzOxMe3b&scp=85081267574&origin=inward.
[10] H. Harahsheh, M. Alshraideh, S. Al-Sharaeh, and R. Al-Sayyed, "Improving Classification Performance for Malware Detection Using Genetic Programming Feature Selection Techniques," J. Appl. Secur. Res., vol. 0, no. 0, pp. 1–21, 2022, doi: 10.1080/19361610.2022.2067459.
[11] K. O. Babaagba and S. O. Adesanya, "A study on the effect of feature selection on malware analysis using machine learning," ACM Int. Conf. Proceeding Ser., vol. Part F1481, pp. 51–55, 2019, doi: 10.1145/3318396.3318448.
[12] A. Tripathi, N. Bhoj, M. Khari, and B. Pandey, "Feature Selection and Scaling for Random Forest Powered Malware Detection System," pp. 1–14, 2021.
[13] F. O. Gbenga, A. A. Olusola, and O. O. Elohor, "Towards Optimization of Malware Detection using Extra-Tree and Random Forest Feature Selections on Ensemble Classifiers," Int. J. Recent Technol. Eng., vol. 9, no. 6, pp. 223–232, 2021, doi: 10.35940/ijrte.f5545.039621.
[14] M. Mays, N. Drabinsky, and S. Brandle, "Feature selection for malware classification," 28th Mod. Artif. Intell. Cogn. Sci. Conf. MAICS 2017, no. April, pp. 165–170, 2017.
[15] Y. Wang and J. Zheng, "An Evaluation of One-Class Feature Selection and Classification for Zero-Day Android Malware Detection," Adv. Intell. Syst. Comput., vol. 1134, no. Itng, pp. 105–111, 2020, doi: 10.1007/978-3-030-43020-7_15.
[16] A. Irshad, R. Maurya, M. K. Dutta, R. Burget, and V. Uher, "Feature optimization for run time analysis of malware in windows operating system using machine learning approach," 2019 42nd Int. Conf. Telecommun. Signal Process. TSP 2019, pp. 255–260, 2019, doi: 10.1109/TSP.2019.8768808.
[17] Z. Xu, C. Wen, S. Qin, and Z. Ming, "Effective malware detection based on behaviour and data features," Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), vol. 10699 LNCS, pp. 53–66, 2018, doi: 10.1007/978-3-319-73830-7_6.
[18] W. Malware, M. L. Classifiers, S. A. Features, and P. Executables, "When Malware is Packin ' Heat : Limits of Machine Learning Classifiers on Static Analysis Features from Packed Executables Anonymous Author (s)," Ndss, no. July 1997, 2016.
[19] M. Z. Mas'ud, S. Sahib, M. F. Abdollah, S. R. Selamat, and C. Y. Huoy, "A comparative study on feature selection method for N-gram mobile malware detection," Int. J. Netw. Secur., vol. 19, no. 5, pp. 727–733, 2017, doi: 10.6633/IJNS.201709.19(5).10.
[20] B. M. Khammas, S. Hasan, R. A. Ahmed, J. S. Bassi, and I. Ismail, "Accuracy Improved Malware Detection Method using Snort Sub-signatures and Machine Learning Techniques," 2018 10th Comput. Sci. Electron. Eng. Conf. CEEC 2018 - Proc., pp. 107–112, 2019, doi: 10.1109/CEEC.2018.8674233.
[21] Y. Zhao, B. Bo, Y. Feng, C. Xu, B. Yu, and J. Chen, "A Feature Extraction Method of Hybrid Gram for Malicious Behavior Based on Machine Learning," Secur. Commun. Networks, vol. 2019, 2019, doi: 10.1155/2019/2674684.
[22] N. Li, Z. Zhang, X. Che, Z. Guo, and J. Cai, "A Survey on Feature Extraction Methods of Heuristic Malware

Detection," J. Phys. Conf. Ser., vol. 1757, no. 1, 2021, doi: 10.1088/1742-6596/1757/1/012071.
[23] K. Raman, "Selecting Features to Classify Malware," InfoSec Southwest 2012, pp. 1–5, 2012, [Online]. Available: http://2012.infosecsouthwest.com/files/speaker_materials/ISSW2012_Selecting_Features_to_Classify_Malware.pdf.
[24] A. Firdaus, N. B. Anuar, A. Karim, and M. F. A. Razak, "Discovering optimal features using static analysis and a genetic search based method for Android malware detection," Front. Inf. Technol. Electron. Eng., vol. 19, no. 6, pp. 712–736, 2018, doi: 10.1631/FITEE.1601491.
[25] S. K. Sahay and A. Sharma, "Grouping the Executables to Detect Malwares with High Accuracy," Phys. Procedia, vol. 78, no. December 2015, pp. 667–674, 2016, doi: 10.1016/j.procs.2016.02.115.