6TH INTERNATIONAL CONFERENCE ON ENGINEERING TECHNOLOGIES

17-19 NOVEMBER 2022

Editör
Şakir TAŞDEMİR

EBOOK ISBN: 978-605-72066-2-6
KEYKUBAT YAYINLARI
Akademi Mah. Yeni İstanbul Cad. No:343 /Z01 Selçuklu / KONYA
MATBAA SERTİFİKASI: 46389

Konya Kasım 2022

KEYKUBAT YAYINLARI
Akademi Mah. Yeni İstanbul Cad. No:343 /Z01 Selçuklu / KONYA
T.C. KÜLTÜR BAKANLIĞI YAYINCI SERTİFİKASI: 46389

Static Analysis of Malware Detection Using Machine Learning Algorithms

N.VURAN SARI[1] and M. ACI[2]

[1]Mersin University, Mersin/Turkey, nvuran@mersin.edu.tr
[2]Mersin University, Mersin /Turkey, maci@mersin.edu.tr

Abstract **- Nowadays, the diversity of malware is rising rapidly and exponentially and, for accurate identification and detection new techniques must be investigated and utilized. Machine learning methods provide high performance in detecting malicious software. Analysis of a software without running it is known as static analysis. By looking at the features that the software uses, such as functions, libraries, digital signatures, and other features, the functioning structure of the software can be resolved. The proposed method presents a comparative study of 8 different well-known machine learning algorithms such as K-Nearest Neighbor, Random Forest, Decision Tree, Gaussian Naive Bayes, CatBoost, LightGBM, Gradient Boosting and eXtreme Gradient Boosting. The dataset used in the study contains 2000 (1000 malicious - 1000 benign) balanced cyber-attack software selected from the real world was used. The result shows that the use of eXtreme Gradient Boosting classification method gives the best classification accuracy by 99.25%.**

Keywords– **Malware Detection, Machine Learning, Static Analysis, Malware Analysis**

I. INTRODUCTION

In recent years, the age of technology has brought many innovations and made our lives easier in every sense, and it has also included cyber risks and threats in our lives. With the spread of information technologies, it is seen that the harmful software that threatens information systems diversifies and their effects increase. Malware are malicious programs that perform operations that are not authorized by the user. They can damage or gain unauthorized access to users' programmable devices, websites or networks.

Some viruses harm the computer in various ways, such as damaging applications, deleting files and reformatting the hard disk, while others are programmed to simply multiply inside the system, slowing down the system, rather than harming it. Viruses, worms, Trojan horses, root kits, back doors, botnets, spyware, and adware are examples of common malware [1]. Today, many methods are used for the analysis and detection of these malicious software. Malware analysis can be done in a variety of ways such as dynamic, static and hybrid.

Static analysis is used to extract information and features from patterns such as arrays, n-grams, functions, opcodes, byte arrays, libraries and call graphs to detect malware [1]. In general, static analysis is an automatic analysis that takes the source code of a program as input, examines this input without running the code, and also checks the code structure and expression sequences and outputs the result. Variable values are handled across different function calls [2]. The malicious code is executed in a controlled or virtual environment throughout dynamic analysis. A number of tools are used to evaluate the behavior of code during execution. Functions, parameters, information flows, instructions, etc. is also analyzed [1].

In addition to static and dynamic approaches, machine learning techniques can be utilized to automate and hasten the stages involved in malware analysis, detection, and classification. Data from static or dynamic analysis is examined in order to classify malware into families or to detect malware using machine learning techniques (i.e. clustering or classification).

The rest of the paper is organized as follows. Chapter 2 describes some of the relevant studies in the literature. Chapter 3 discusses methodology that includes machine learning-based algorithms used to detect malware and dataset pre-processing. Chapter 4 compares machine learning malware detection techniques and discusses the comparison of findings. Chapter 5 concludes this article by emphasizing the research aspects.

II. RELATED WORKS

Researchers have recently started to evaluate and categorize malware using a variety of ways. The current methods of operation might not be very efficient against new threats. Machine learning algorithms can enhance malware detections and outperform current methods.

Harshalatha and Mohanasundaram [3] presented the literature work of previously existing works of malware detection classification using machine learning algorithms. Classifier models used in this approach are Random Forest (RF), BayesNet, Multi-Layer Perceptron (MLP), and Support Vector Machine (SVM). First, all malware samples were tested under 10-fold cross-validation, and the classifier results showed that RF had the best performance (98% accuracy) and the same RF accuracy was down by 12% when the RF classifier was applied for different datasets.

Patil et al. [4] pointed out a framework that extracts numerous feature-sets from the malware files, including system calls, operational codes, sections, and byte codes. They used Microsoft's malware dataset available on the Kaggle website. It contained 10868 malware files from a variety of 9 malware families. The study compares the effectiveness of machine learning and deep learning-based methods on each of

these features, and the results show that feature vector for system calls achieved the best accuracy.

In order to find malicious executables in the wild, Kolter et al. [5] employed machine learning methods and used n-grams of byte codes as characteristics. As training examples, they encoded each of the 1651 malicious executables and the 1971 benign executables they collected. They tested several inductive techniques after choosing the most pertinent n-grams for prediction, including Naive Bayes (NB), Decision Tree (DT), SVM, and Boosting. The area under the Receiver Operating Characteristic (ROC) curve for boosted DT was 0.996, outperforming all other approaches in the end.

Algahtani et al. [6] focused on machine learning-based classifiers to identify malicious software on Android devices, this study covers the state of the art in Android malware detection strategies. Different machine learning algorithms, including SVM, NB, Perceptron, J48, OneRand Deep Network algorithms, provide the base for framework development. The dataset in use includes 2081 benign applications and 91 malicious ones. OneR and J48 algorithms achieved 100% accuracy with a 0.00% false positive rate.

In order to identify malware, Schultz et al. [7] applied three distinct approaches on Windows-based system portable executable files (PE). These techniques were employed to find known malware. They employed a method that utilized Dynamic Link Libraries (DLLs), function calls, and attributes for the frequency with which these functions were called. They used binary data that was extracted from PE files utilizing strings software for the second approach. Finally, they utilized the 2-byte strings provided by the program by utilizing the hex dump software. In this article, Schultz used RIPPER, NB, and several classifier techniques for detection. The system utilizing NB provided the highest detection rate per the data.

III. METHODOLOGY

It is being researched to see if machine learning techniques can detect novel malware. The dataset has been preprocessed and digitalized so that machine learning algorithms can use it. The effectiveness of eight well-known machine learning techniques for malware detection was evaluated.

A. Dataset

The dataset consists of 2000 samples of software labeled as malicious or benign, 1000 of which are malicious and 1000 are benign software, as shown in Table 1. File size, software digital signature status, libraries, and functions were all used as features.

Table 1: Number of collected data.

Class	Count
Malicious	1.000
Benign	1.000
Total	2.000

The dataset is split into a sub-dataset after being shuffled, with 80% (1600) of the data being used for training and 20% (400) for testing in the machine learning phase of the system.

The training datasets were imported into a Python module in order to create a model using machine learning methods (i.e., DT, RF, SVM, K-Nearest Neighbors (KNN), Boosting algorithms and so on) from the Scikit-learn library.

B. Preprocessing

Data preprocessing is the term used to describe any processing approach used on raw data to get it ready for next processing step. It transforms data into a format that classification algorithms can analyze more quickly and efficiently [8].

Some of the issues of signature-based malware detection can be overcome with the help of supervised machine learning classification. Building labeled training dataset is the initial step in using machine learning to categorize files as malicious or benign. Each file is classified as benign or malicious based on its properties, which are deduced from a few key aspects of the file [9].

Between the features in the dataset the function and library names in the functions and libraries (features) column were arranged as columns, so that each function and library name was also considered as a dataset feature. 195 unique libraries and 4,327 unique functions were obtained. Including size, digital signature, class and these unique functions and libraries are also used as features in the dataset. A total of 4,525 features were used.

Categorical input must be transformed into numerical terms for the model to work in machine learning. All features in the dataset were normalized to a binary value as "0" or "1".

Cross-validation technique, which divides the training set into numerous smaller training sets and a validation set, is used to prevent over-fitting [1]. The model is trained on smaller training sets and then tested on the test set.

C. Performance Measure

Precision, recall, accuracy and F1-score were computed to evaluate the effectiveness of machine learning methods. Below are the measures that used in the context of categorizing malware.

• F1-Score: Harmonic mean of recall and precision for a more accurate analysis of model effectiveness (1).

$$F1 - Score = \frac{Precision \times Recall}{Precision + Recall} \quad (1)$$

• Accuracy: Percentage of how many malicious and benign files were properly classified (2).

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (2)$$

• TP (True Positive): Defines the quantity of files that were correctly categorized as malicious.

• TN (True Negative): Defines the quantity of files that were correctly categorized as benign.

• FP (False Positive): Defines the quantity of files that were incorrectly categorized as malicious.

• FN (False Negative): Defines the quantity of files that

were incorrectly categorized as benign.

• Precision: Properly classified malicious and benign files' percentage. It is calculated by dividing the actual positives by any positive predictions (3).

$$Precision = \frac{TP}{TP+FP} \quad (3)$$

• Recall: Properly classified malicious and benign files' percentage. It is calculated by dividing the actual positives by any positive predictions that should have been made (4).

$$Recall = \frac{TP}{TP+FN} \quad (4)$$

D. Machine Learning Algorithms

Due to the shortcomings of the presently offered malware detection methods, machine learning techniques are integrated with existing detection methods to increase the detection process' efficiency [10]. Various researchers have demonstrated the effectiveness of machine learning in detecting malware. Therefore, many machine learning-based malware detection systems have been studied in this area.

It is obvious that employing machine learning to identify malware offers the benefit of superior detection accuracy. Due to their accuracy in identifying harmful samples, existing research demonstrates that employing decision trees to detect malware may be a suitable strategy[11].For this reason, algorithms with decision trees based were mainly studied.

GaussianNB, KNN,DT, RF, Catboost (CB), LightGBM (LGBM), eXtreme Gradient Boosting (XGB) and Gradient Boosting (GB) are the algorithms that were employed in this study to categorize the data as malware or benign.

The steps of the supervised machine learning process are as follows (Figure 1):

1. Data extraction and transformation into a format for machine learning models;
2. Using training data to train the models and determining the ideal hyper-parameters for a certain model;
3. Comparing and analyzing the models using test data.

These stages provide trained models that can classify samples in proposed malware detection system that were unknown samples.

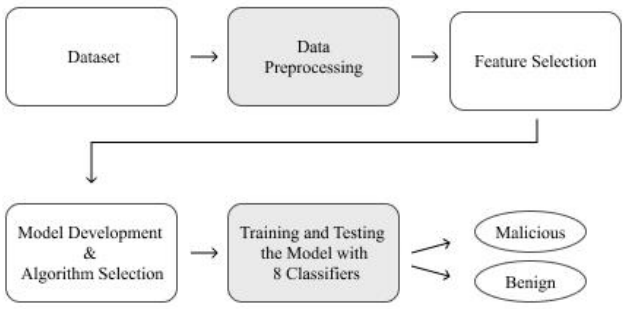

Figure 1: System Overview

IV. Experimental Results

2000 software have been labeled in dataset, 1000 of which are malicious and 1000 of which are benign. The common 10-fold cross-validation procedure was applied in experiments. In other words, the dataset is partitioned into ten smaller sub-datasets at random. Nine of these ten distinct sub-datasets were utilized to train the algorithms, while one was used to test the approaches.

For performance evaluation accuracy, f1-score, precision, and recall values were computed.

Table 2: Overall malware classification results

Algorithms	F1-Score	Accuracy	Precision	Recall
KNN	0.606357	0.5975	0.610837	0.60194
RF	0.978102	0.9775	0.980488	0.97572
DT	0.949640	0.9475	0.938389	0.96116
GaussianNB	0.673611	0.5300	0.524324	0.94174
CB	0.987893	0.9875	0.985507	0.99029
LGBM	0. 983133	0. 9875	0. 976077	0. 99029
GB	0.987893	0.9875	0.985507	0.99029
XGB	**0.992736**	**0.9925**	**0.990338**	**0.99514**

As shown in Table 2, according to the accuracy results, XGB outperformed seven algorithms with a very high detection rate of 99.25%. This was followed by CB, LGBM and GB other boosting algorithms with 98.25% accuracy, which showed the same accuracy although F1-score, precision and recall values were different. These three algorithms were followed by RF, DT, KNN and NB algorithms, respectively. It has been observed that the F1-score, precision and recall results give results proportional to the accuracy results. Boosting algorithms such as XGB, GB, and CB are more powerful than other algorithms for several reasons, as they are essentially decision tree-based algorithms. One of these reasons is that the tree structure they produce tends to reduce the error from the previous tree.

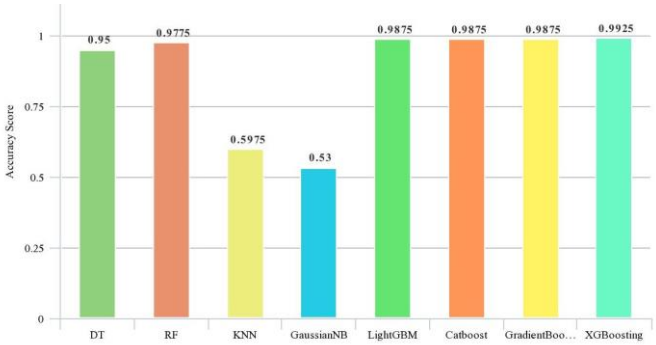

Figure 2: Accuracy of Algorithms

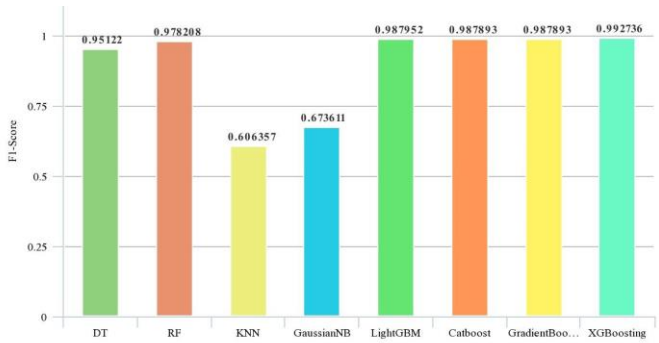

Figure 3: F1-Score of Algorithms

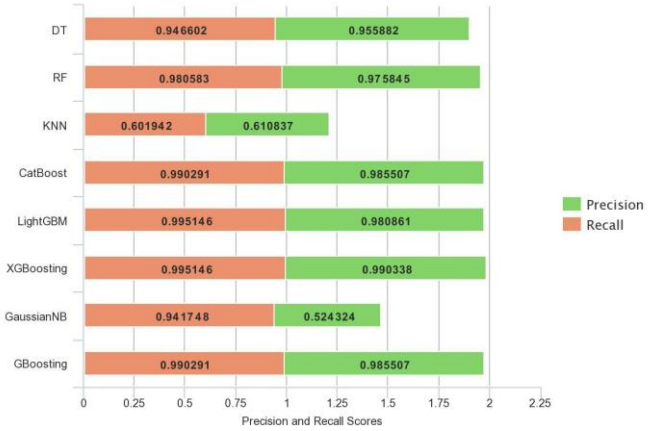

Figure 4: Precision and Recall Scores of Algorithms

V. Conclusion

In this work, eight different machine learning algorithms were applied the detect malware. Static code analysis-based studies that use signatures frequently fail to identify newly discovered malware. As a result, machine learning algorithms are utilized in this study to separate malicious software from software that isn't harmful by exploiting the structural elements of the software.

In our future work, the success rate of the model in detecting malware can be increased by using deep learning methods and algorithms. In addition, by weighting the features that are more important for malware detection in dataset, both the runtime and the accuracy performance can be improved.

Appendix

We would like to thank CHOMAR A.S. Company for sharing the dataset used in this study.

References

[1] Patil, R., & Deng, W. (2020, March). Malware Analysis using Machine Learning and Deep Learning techniques. In *2020 SoutheastCon* (Vol. 2, pp. 1-7). IEEE.

[2] Li, L., Bissyandé, T. F., Papadakis, M., Rasthofer, S., Bartel, A., Octeau, D., &Traon, L. (2017). Static analysis of android apps: A systematic literature review. *Information and Software Technology*, *88*, 67-95.

[3] Harshalatha, P., and R. Mohanasundaram. "Classification of Malware Detection Using Machine Learn-ing Algorithms: A Survey." *International Journal of Scientific & Technology Research* 9.02 (2020).

[4] Kolter, Jeremy Z., and Marcus A. Maloof. "Learning to detect malicious executables in the wild." *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*. 2004.

[5] Alqahtani, Ebtesam J., RachidZagrouba, and Abdullah Almuhaideb. "A Survey on Android Malware Detection Techniques Using Machine Learning Algorithms." *2019 Sixth International Conference on Software Defined Systems (SDS)*. IEEE, 2019.

[6] Schultz, M. G., Eskin, E., Zadok, F., &Stolfo, S. J. (2000, May). Data mining methods for detection of new malicious executables. In *Proceedings 2001 IEEE Symposium on Security and Privacy. S&P 2001* (pp. 38-49). IEEE.

[7] Iliou, T., Anagnostopoulos, C. N., Nerantzaki, M., &Anastassopoulos, G. (2015, September). A novel machine learning data preprocessing method for enhancing classification algorithms performance. In *Proceedings of the 16th International Conference on Engineering Applications of Neural Networks (INNS)* (pp. 1-5).

[8] Markel, Zane, and Michael Bilzor. "Building a machine learning classifier for malware detection." *2014 second workshop on anti-malware testing research (WATeR)*. IEEE, 2014.

[9] Chumachenko, Kateryna. "Machine learning methods for malware detection and classification." (2017).

[10] Palša, J., Ádám, N., Hurtuk, J., Chovancová, E., Madoš, B., Chovanec, M., &Kocan, S. (2022). MLMD—A Malware-Detecting Antivirus Tool Based on the XGBoost Machine Learning Algorithm. *Applied Sciences*, *12*(13), 6672.