Smart Boosted Model for Behavior-Based Malware Analysis and Detection

Saja Abu-Zaideh, Mohammad Abu Snober, and Qasem Abu Al-Haija

Abstract Malware analysis and detection are the most important activities to ensure system security. However, current attacks like polymorphic viruses and zero-day attacks that utilize signature-based methods complicate the detection process with accurate results. Therefore, this has—in turn—raised the need for more intelligent techniques to analyze the behavior of the malware rather than depending on the signature-based analyses. This paper proposes a machine learning-based model to analyze and detect the different types of malware. The system tries to determine the optimal feature representation and extraction and classification method that can lead to the best detection accuracy. Particularly, different machine learning algorithms were evaluated, including k-Nearest Neighbors (kNN), Multi-Layer Perceptron (MLP), Naive Bayes Classifier (NBC), Adaboost/XGBoost Decision Trees (ADT), and Support Vector Machines (SVM). The models were trained and tested using a new dataset that includes op-codes available in .asm format (generated using the IDA disassembler tool); it is a subset of data used in Kaggle for the Microsoft Classification challenges. Our empirical results revealed the superiority of the XGBoost-based model scoring an overall detection accuracy of 98.3%.

Keywords Malware · Machine learning · XG Boost · Malware analysis · Malware detection · Classification · Accuracy

1 Introduction

Obviously, it is justifiable that malware has become one of the major threats in the world. The prompt improvement of communication and the internet is the main reason. Malware or malicious software is an accumulated number of viruses developed to execute a malicious activity, information-stealing, reconnaissance, or others. Malware is mainly meant to produce massive destruction to the data and sources or

S. Abu-Zaideh · M. A. Snober · Q. A. Al-Haija (✉)
Department of Computer Science/Cybersecurity, Princess Sumaya University for Technology (PSUT), Amman, Jordan
e-mail: q.abualahaija@psut.edu.jo

P. P. Joby et al. (eds.), *IoT Based Control Networks and Intelligent Systems*,
Lecture Notes in Networks and Systems 528,
https://doi.org/10.1007/978-981-19-5845-8_58

to obtain unlawful access to the network. Also, it is defined as "a type of computer program designed to infect a legitimate user's computer and inflict harm on it in multiple ways." [1]

The main purpose of employing machine learning techniques instead of ani-virus scanners is that the assortment of malware types is expanding, resulting in millions of hosts being attacked. Therefore, anti-virus systems and other scanners cannot comply with the requirements of recognition and security. The latest research demonstrates that seven million various hosts were attacked, and in 2015, up to four million malware entities were discovered [2]. Nowadays, for the period of the COVID-19 epidemic, the number of malware attacks has grown; Malware expenditures are currently totaling over $1 billion every year.

Additionally, the malware raises for the reason that various accessible tools are established that involve the lowest possible degree of proficiency. Recent research indicates that the majority of attackers nowadays are script-kiddies [3].

Consequently, malware safeguard is among the most important cybersecurity tasks to ensure user privacy and confidentiality; this is since, at the same time, an individual attack can lead to a sufficient loss in the organizational assets. Recurrent attacks impose the demand for sensible and precise recognition approaches. We realized that the existing static and dynamic approaches do not support effective detection dependent on modern attacks, such as zero-day attacks. For this reason, the application of machine learning-based methods was increased. Thus, in this paper, we will discuss the primary arguments and interests of machine learning-based malware detection. Also, we will cover the finest feature interpretation and classification procedures.

The primary objective is to determine the optimal feature selection process and how the features must be extracted. Such an accurate procedure can differentiate the malware types with the smallest possible error value. Hence, to achieve this goal, we will create the proof of concept for the machine learning-based malware classification, which will be utilized as an input to the machine learning systems. Consequently, we look at high-accuracy results to determine the best performant algorithm.

The leftover parts of this manuscript are structured as follows: Sect. 2 provides an essential theoretical background to fulfill the important knowledge required throughout this paper. Section 3 provides the details of the proposed model, including feature selection, machine learning modeling, the dataset, and cross-validation. Section 4 discusses the experimental results and Sect. 5 concludes the finding of this research paper.

2 Theoretical Background

This section delivers the basic knowledge which plays a significant role in understanding malware discovery and the necessity for machine learning techniques. First, we will describe the malware types relevant to the study, then the typical malware

detection techniques. Subsequently, we will discuss the necessity for employing machine learning based on the knowledge gained. Moreover, we will review some of the related work.

2.1 Malware Types

In this section, we will classify the malware to provide the best way to understand the methods and logic; we can divide Malware, depending on its intention, into several groups. As follows:

- Worm: This type can propagate across the network, similar to the virus. Also, it is able to reproduce in other machines.
- Virus: Viruses can declare as any part of the software which is inserted and released automatically; with user consent and permissions. This is the humblest structure of software. It can reproduce itself or infect (modify) other software [4].
- Adware: This type of malware can display advertisements on your computer. We can say that adware is a subclass of spyware.
- Trojan: This is a type of malware that intends to be seen as lawful software. Trojans may be engaged by cyber thieves and hackers attempting to retrieve the systems of the users. Social engineering normally deceived users into the insertion and execution of Trojans on their systems. When enabled, Trojans can activate cyber-criminals to snoop on your system, sneak your vulnerable data, and obtain backdoor entrance to your system [5].
- Spyware: Spyware is a type of malware that performs espionage and can be mentioned to as spyware. It can infect your PC or mobile. So, spyware can do several warm actions like gathering information about you, including tracking your search history, the websites you visited, the belongings you downloaded, your credentials (usernames and passwords), payment details, and the emails you send and receive, all that to send personalized announcements, or to sell them to the third parties subsequently [6].
- Rootkit: It's designed to enable access to your computer's data with more privilege than is allowed. So, it is used to give administrative access to an unauthorized user. Another important piece of information about Rootkits is that it is hard to detect or incredible to remove because they hide their existence.
- Backdoor: This type of malware indicates any procedure which enables authorized users and/or unauthorized users to gain high-level user access to the computer system. Provides an additional secret "entrance" to the system. It can manage to steal financial and private data or to hijack devices.
- Ransomware: Ransomware is malware that encrypts all the data in the victim's computer using an encrypted key. So the user cannot open any file in his machine until he gets the decrypted key, and he can get it by transferring money to the attacker.

- Keylogger: This type of malware stores all keys logged by the user, like usernames, passwords, and account numbers, so that the attacker can get this sensitive information [7].

2.2 Detection Methods

We can detect malware depending on the file's signature or by testing the behavior of executable files. But first, we must recognize between static and dynamic malware analysis. Static analysis is for non-executable files or without running the file. And dynamic analysis includes testing the executable file while running; this approach can be made using sandboxes.

The main job of static analysis is to infer the file's behavior properties by reading the malware's source code and infer the file's behavioral properties. Several techniques can Static analysis cover it, such as [8]:

1. File Format Inspection: file metadata can fork out utility information like Windows PE (portable executable) files.
2. String Extraction: Examining the software output (e.g., status or error messages).
3. Fingerprinting: which compromises cryptographic hash computation.
4. AV scanning: by comparing the inspected file and if it is a well-known malware, it can be detected by all anti-virus scanners.
5. Disassembly: trying to infer the software logic and intentions by reversing machine code to assembly language.

Static analysis is safer than dynamic analysis. Because the file will not be running while it is under testing, it cannot infect the system. Static analysis is a simple basic approach it gives us to predict all possible behavioral scenarios. But it is not usually used in the real world because it is more time-consuming [9].

On the other hand, the dynamic analysis. It is less safe because we execute the file in a virtual environment like sandboxes while we test it to monitor the behavior of the file. It also runs at high speed and takes less time than static analysis [10].

2.3 The Need for Intelligent Models

We can detect malware depending on signatures. But there are two types of signatures; the first one is a static signature, all malicious files with a static signature can be detected easily using anti-virus scanners. They compare file signatures with all signatures of malicious files they have. If the signature matches, then they decide this file is malicious.

There are also types of files that can change their signature continuously; we call that polymorphic signature. This type of file spreads at a high rate. This type of signature cannot detect using traditional scanners. From that, the need for an updated

tool arises, so in this paper, we will use machine learning by writing a python code that compares the feature of the file with the stored feature of the malicious file to try fining if this file is malicious or not with high efficiency and less time.

3 Proposed System Model

The data mining techniques are commonly rapidly developed, resulting in using machine learning in other fields like the security field. The way the computer program learns from its test is called machine learning. In 1959 Arthur Samuel declared machine learning a "field of study that gives computers the ability to learn without being explicitly programmed." Machine learning depends on training the model to do its job by using some algorithm to do several functions: classification, clusterization, regression, etc. What machine learning does to train the dataset is take the dataset as input and build predictions by using certain models, then give us the output. The general workflow will give a good understanding of this process, as shown in Fig. 1 below.

The machine learning process contains five stages:

- Data intake. It is the process of loading the data from the file and storing it.
- Data transformation. In this step, we initialize the data by clearly and normalizing it to be appropriate for the algorithm. Also, feature extraction and selection are performed. In addition, the data is separated into sets—'training set' and 'test set'—all done in this stage.

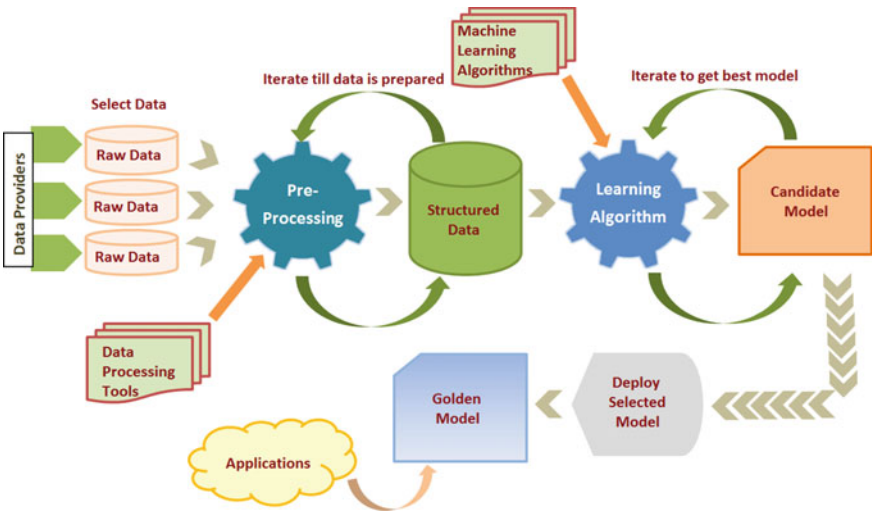

Fig. 1 The general process of machine learning

- Model Training. This step is responsible for selecting the appropriate algorithm and then building the model.
- Model Testing. The result produced from this step will be used for building new.
- Model Deployment. This stage is to select the final model.

3.1 Classification Methods

From the machine learning point of view, malware identification can be seen as a challenge of classification tasks where unknown malware categories must be clustered into various bunches according to specific attributes associated with the algorithm. In other contexts, having trained a model on the inclusive dataset of malicious and benign files, we can decrease this problem to classification. For common malware groups, this dilemma can be tightened down to classification only—having a restricted set of classes, to one of which malware instance certainly be in the right place, it is easier to recognize the appropriate class, and the result would be more accurate than with clusterization algorithms.

For instance, the random forest algorithm is used repeatedly in machine learning-based solutions. It is a simple algorithm that gives highly accurate results. As the name implies, it is called a forest because it is based on a large set of decision trees. It works over multiple decision trees. All these depend on an independent subset of datasets. It consists of n nodes. The main scheme of the algorithm is shown in Fig. 2 below. The advantages of this algorithm are that they are fit for classification and regression problems; they are easy to use and apply. They give a much better accurate result.

Also, one of the recent supervised machine learning algorithms is the XG Boost. XG Boost is considered a type of gradient boosted decision tree. With high speed and excellent accuracy. And in our experiment, we achieve high accuracy, which equals 98%, by using the XG Boost algorithm.

3.2 Data

In this research, we have used an open-source dataset from Microsoft; this dataset is roughly half a terabyte, supplied with a collection of common malware records indicating a combination of nine distinct groups. Every malware file has an identification number (ID). Every ID is composed of a 20-characters hash value distinctively recognizing the file. The file also has a class label that is composed of integer values indicating one of nine malware types that the malware might fit in.

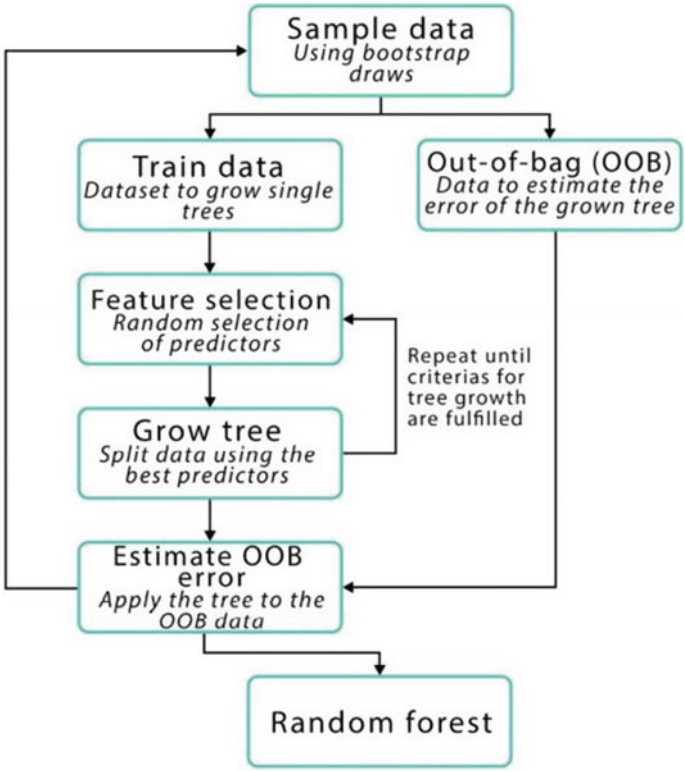

Fig. 2 Random Forest workflow

3.3 Cross-Validation

The cross-validation method is used to predict the way the model will perform on the new output data. The way it works is by splitting the dataset. Then the model takes the biggest part and will be trained on it. After that, the model will be evaluated using the small part.

1. Holdout method—it is a basic method, simply the origin dataset divided into two parts as shown in Fig. 3 below: the largest part, the training set, and the smallest part, the test set. And we trained the training date and evaluated it on test data. One of the advantages of this approach is that it is extremely fast.
2. The k-fold method is the improvement version of the Holdout approach. Here, the set is divided into numbers of subsets called k, and the holdout method is repeated k number of times. The disadvantage of this approach is that it is slow and more complex. We have used five-fold cross-validation in this research, as illustrated in Fig. 4 [11].

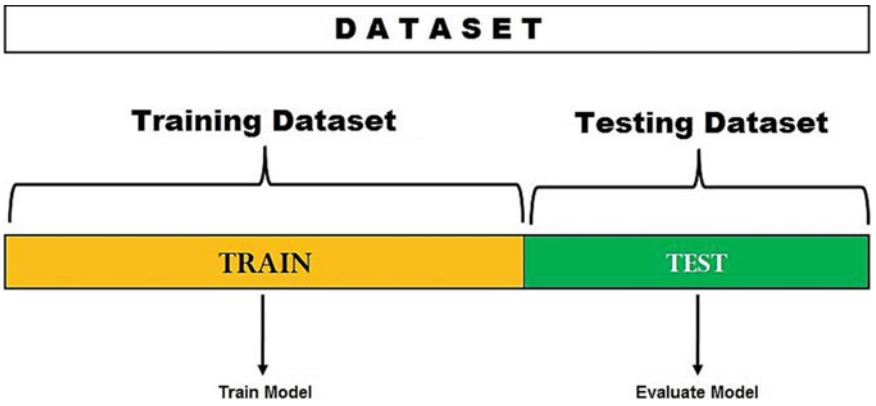

Fig. 3 The dataset in the holdout method

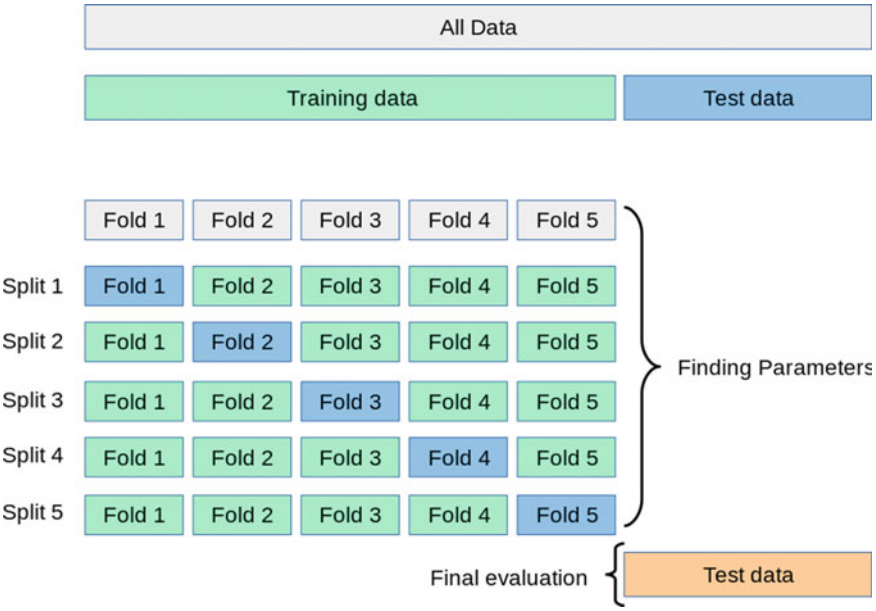

Fig. 4 Fivefold cross-validation method

4 Results and Discussion

Initially, feature selection is applied to eliminate redundant and inappropriate features. This in turn can enhance the predictive accuracy of the model. For our model, the feature set is tremendously big, and there is an indispensable necessity

for feature selection. Consequently, we applied a feature selection technique to end up with 56 features, as presented in Fig. 5.

After extracting the relevant features, and preprocessing the data records, then, we are capable of applying the machine learning approaches to the data we acquired. The machine learning techniques to be employed, as argued beforehand, include KNN, SVM, Naive Bayes, XG Boost, Random Forest, and Decision Tree. Figure 6 below shows data clustering and how they are divided into three groups. In the beginning, we applied an unsupervised machine learning, that is, the K-Means algorithm, to obtain the data classes where we obtained three classes 0 (Normal), 1 (Malware), or 2 (Unkown).

After that, ten different supervised machine learning techniques were applied to the labeled datasets: k-Nearest Neighbor (KNN), Linear SVM, RBF-SVM, Decision

```
Index(['Name', 'md5', 'Machine', 'SizeOfOptionalHeader', 'Characteristics',
       'MajorLinkerVersion', 'MinorLinkerVersion', 'SizeOfCode',
       'SizeOfInitializedData', 'SizeOfUninitializedData',
       'AddressOfEntryPoint', 'BaseOfCode', 'BaseOfData', 'ImageBase',
       'SectionAlignment', 'FileAlignment', 'MajorOperatingSystemVersion',
       'MinorOperatingSystemVersion', 'MajorImageVersion', 'MinorImageVersion',
       'MajorSubsystemVersion', 'MinorSubsystemVersion', 'SizeOfImage',
       'SizeOfHeaders', 'CheckSum', 'Subsystem', 'DllCharacteristics',
       'SizeOfStackReserve', 'SizeOfStackCommit', 'SizeOfHeapReserve',
       'SizeOfHeapCommit', 'LoaderFlags', 'NumberOfRvaAndSizes', 'SectionsNb',
       'SectionsMeanEntropy', 'SectionsMinEntropy', 'SectionsMaxEntropy',
       'SectionsMeanRawsize', 'SectionsMinRawsize', 'SectionMaxRawsize',
       'SectionsMeanVirtualsize', 'SectionsMinVirtualsize',
       'SectionMaxVirtualsize', 'ImportsNbDLL', 'ImportsNb',
       'ImportsNbOrdinal', 'ExportNb', 'ResourcesNb', 'ResourcesMeanEntropy',
       'ResourcesMinEntropy', 'ResourcesMaxEntropy', 'ResourcesMeanSize',
       'ResourcesMinSize', 'ResourcesMaxSize', 'LoadConfigurationSize',
       'VersionInformationSize', 'legitimate'],
      dtype='object')
```

Fig. 5 The selected features

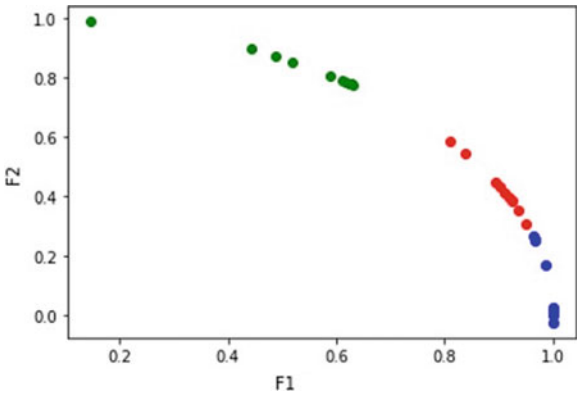

Fig. 6 Clustering and data labeling

Table 1 Accuracy result

Accuracy	Name
0.943	KNN
0.773	Linear SVM
0.837	RBF SVM
0.937	Decision Tree
0.95	Random Forest
0.933	MLP Classifier
0.927	AdaBoost
0.9	Naive Bayes
0.787	QDA
0.98	XG Boost

Trees, Random forest, MLP Classifier, Adaboost, Naive Bayes, Quadratic Discriminant Analysis (QDA), and XG Boost. Table 1 shows the results of detection accuracy obtained from applying the stated models. From the results of different models, we can observe that the smallest accuracy rate was attained by Linear SVM (77.34%), followed closely by QDA and RBF SVM which obtained 78.7% and 83.7%, respectively. While the greatest accuracy rate was attained with the XG Boost, the XG Boost (Extreme Gradient Boosting) algorithm recorded the best detection accuracy result.

5 Conclusions and Future Work

In this paper, a new machine learning-based system for malware analysis and classification is proposed, implemented, and evaluated. The model uses ten machine learning techniques: k-Nearest Neighbor (KNN), Linear SVM, RBF-SVM, Decision Trees, Random forest, MLP Classifier, Adaboost, Naive Bayes, Quadratic Discriminant Analysis (QDA), and XG Boost. The experimental evaluation results for the detection accuracy showed that the model-based XG Boost is superior, with 98.3% accuracy. In the future, this experiment forms a good base and is applicable for larger datasets. Therefore, several future improvements related to the practical implementation of this project can be identified.

References

1. Kaspersky Labs (2017). What is malware, and how to defend against it? http://usa.kaspersky.com/internet-securitycenter/internet-safety/what-is-malware-andhow-to-protect-againstit#.WJZS9xt942x. Accessed 15 Feb 2017

2. Abu Al-Haija Q, Al-Dala'ien M (2022) ELBA-IoT: an ensemble learning model for botnet attack detection in IoT networks. J Sens Actuator Netw 11:18. https://doi.org/10.3390/jsan11010018
3. Aliyev V (2010) Using honeypots to study skill level of attackers based on the exploited vulnerabilities in the network. The Chalmers University of Technology
4. Horton J, Seberry J (1997) Computer viruses. An introduction. The University of Wollongong
5. Smith C, Matrawy A, Chow S, Abdelaziz B (2009) Computer worms: architectures, evasion strategies, and detection mechanisms. J Inf Assur Secur
6. Moffie M, Cheng W, Kaeli D, Zhao Q (2006) Hunting Trojan Horses. In: Proceedings of the 1st workshop on architectural and system support for improving software dependability
7. Chien E (2005) Techniques of adware and spyware. WWW document. https://www.symantec.com/avcenter/reference/techniques.of.adware.and.spyware.pdf
8. Lopez W, Guerra H, Pena E, Barrera E, Sayol J (2013) Keyloggers. Florida International University
9. Abu Al-Haija Q, Krichen M, Abu Elhaija W (2022) Machine-learning-based darknet traffic detection system for IoT applications. Electronics 11:556. https://doi.org/10.3390/electronics11040556
10. Prasad BJ, Annangi H, Pendyala KS (2016) Basic static malware analysis using open-source tools
11. Abu Al-Haija Q (2022) Top-down machine learning-based architecture for cyberattacks identification and classification in IoT communication networks. Front Big Data 4:782902