I. J. Computer Network and Information Security, **2019, 6, 1-9**
Published Online June 2019 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijcnis.2019.06.01

A New Classification Based Model for Malicious PE Files Detection

Imad Abdessadki
Mathematics, Computer Science and Applications TEAM,
National School of Applied Sciences - Tangier, AbdelMalek Essaadi University, Morocco
E-mail: imad.abdessadki@gmail.com

Saiida Lazaar
Department of Mathematics & Computer Science. Mathematics, Computer Science and Applications TEAM,
National School of Applied Sciences - Tangier, AbdelMalek Essaadi University, Morocco
E-mail: slazaar@uae.ac.ma

Received: 17 April 2019; Accepted: 19 May 2019; Published: 08 June 2019

***Abstract*—**Malware presents a major threat to the security of computer systems, smart devices, and applications. It can also endanger sensitive data by modifying or destroying them. Thus, electronic exchanges through different communicating entities can be compromised. However, currently used signature-based methods cannot provide accurate detection of zero-day attacks, polymorphic and metamorphic programs which have the ability to change their code during propagation. In order to solve this issue, static and dynamic malware analysis is being used along with machine learning algorithms for malware detection and classification. Machine learning methods play an important role in automated malware detection. Several approaches have been applied to classify and to detect malware. The most challenging task is selecting a relevant set of features from a large dataset so that the classification model can be built in less time with higher accuracy. The purpose of this work is firstly to make a general review on the existing classification and detection methods, and secondly to develop an automated system to detect malicious Portable Executable files based on their headers with low performance and more efficiency. Experimental results will be presented for the best classifier selected in this study, namely Random Forest; accuracy and time performance will be discussed.

***Index Terms*—**Malware detection, Portable Executable, Malware classification, Machine learning, Random Forest, Unknown malware.

I. Introduction

The development of the Internet and the evolution of online services such as invoice payments, etc. lead to the growth of attacks and malware, the statistics show more than 350,000 new malicious programs every day [1]. Malicious files become more sophisticated in the presence of polymorphic, metamorphic and other advanced techniques which are used by attackers to evade any malware detection system and to cause various difficulties during manual analysis, in addition, analyzing manually the increasing number of malware requires a lot of human resources. The creators of malware in general target the entities in possession of sensitive data, and they only do so due to the great value and the financial gains they can get from selling this data, as in the case of ransomware i.e. WannaCry which is one of the most devastating ransomware attacks in history, this attack was estimated to have affected over 200,000 victims and more than 300,000 computers [2, 3, 4]. WannaCry causes, estimated global financial and economic losses of up to $4 billions [5]. Malware can also be used for the government's political problems as for example, of developing Stuxnet malware by the U.S. and Israeli governments to derail Iran's nuclear weapons program [6]. The goal of developing the Stuxnet malware was not just to infect machines, but to cause real-world physical effects. Precisely, it targets centrifuges used to produce the enriched uranium that powers nuclear weapons and reactors [7]. Smart devices are also targeted by malware authors because they contain very sensitive data such as personal pictures, etc. In addition, they help to spread malware in a fast way, because they are mobile, and they can integrate on any network at any time (smartphones, smartwatches, etc.). The statistics show that the number of smartphone users worldwide is growing rapidly [8]. Based on these statistics we note that those smart devices, especially smartphones become more used in our life because they make it easier with the services that offer for us, but the provided services are not free of dangers. We note that malware can target different systems, and hackers use more sophisticated techniques to follow the evolution and bypassing the protection techniques. Traditional detection systems remain insufficient against this evolution that's why we need to look for more advanced methods to follow this growth. Several types of researches have been proposed to detect or reduce the impact of malicious files, the majority of those studies are based on Artificial Intelligence [10, 11, 14, 15, 20, 30, 33]. The main goal of our work is to create an automated detection system to

distinguish between malicious and legitimate Portable Executable (PE) files using the information of their headers based on Artificial intelligence more precisely on machine learning algorithms. The efficiency of the machine learning techniques in detecting malware has been proved by a multitude and rich research works [14].

The rest of this paper is as follows. In section II, we discuss the most popular approaches concerning malware classification and detection, and we perform a general review and a summary of some related works. Section III contains the objectives of this research and the methodology used for building our detection system. Section IV is dedicated to our best experimental results obtained from two different experiments. In Section V, we compare our results with the most recent related works in the field of malware recognition and classification. In section VI, we conclude this paper and we give some outlines of our future work.

II. Related Works

R. Vinayakumar et al. [33] proposed a malware detection framework called ScaleMalNet to detect and categorize unknown malware into their corresponding categories based on machine learning algorithms and Deep learning architectures. Two types of datasets are used: the first dataset contains 240,418 samples includes 25 different malware families and the second dataset contains 103,037 samples. Various experiments have been performed to evaluate the performance of this proposed framework, the best accuracy of the Dataset 1 and 2 is 96.3% and 98.8% respectively.

M. Chowdhury et al. [30] present an approach that combines the use of N-gram and API Call features to detect and classify malicious files using data mining and machine learning classification algorithms. The experiments were conducted on 52,185 sets of data including 10,920 of benign and 41,265 of malicious files. The highest accuracy of this proposed approach is 98.8%.

Hellal et al. [9] present a graph mining technique to identify variants of malicious files using static analysis, they proposed a novel algorithm for recognition of obfuscated and unknown malware named Minimal Contrast Frequent Subgraph Miner Algorithm to extract automatically minimal discriminative recurrent behavior patterns from suspect files. This suggested method this method cares to save memory space and reduce scanning time by generating a limited number of signatures in contrast to the existing methods that generate patterns per single malware. This approach displays high recognition rates and low false positive rates with an accuracy of 92%.

Jain and Kumar Meena [16] proposed an approach to detect malicious files using N-grams which are considered as features, n-grams ranging from 1 to 8 and extracted from raw byte patterns of benign and malicious Portable Executable samples. They used Class wise document to reduce the space of features. Experiments have been conducted on 2138 of PE files and the classification is realized by WEKA (Waikato environment for knowledge analysis) using Naïve Bayes, Instance-Based Learner, J48, AdaBoost1, Random Forest classifiers. The detection accuracy of this suggested technique is around 99%. And it performs well for 3-gram.

Jinrong Bai et al. [17] proposed a malware detection approach based on the static format of the PE files. The feature extraction technique is used to extract 197 features from each file, and they used feature selection techniques to reduce the number of features. Three experiments have been performed in this approach to verify the performance of the detection system. In the first experiment, they used a Cross-Validation technique, in the second experiment, the set of data was randomly partitioned into the training and test set and in the third experiment the dataset was divided into the training and test set by chronological order to evaluate the performance of detecting new malicious files because when the dataset contains both old and new malware, the ability of detecting unknown malware cannot be evaluated accurately. The results of those experiments show that the accuracy of the top classification algorithm is 99.1% and the ability to recognize unknown malware is not satisfying. This method is still capable of detecting 97.6% of new malicious files with a 1.3% false positive rate. Another similar approach based on PE header is realized by Yibin Liao [18]. An experiment in which he used 6875 samples of data, containing 5598 malicious and 1237 benign of executable files. The feature extraction of each header field has been made by PE-Header-Parser and Icon-Extractor for extracting the icons. The results show that this approach achieves more than 99% detection rate with less than 0.2% false positive in less than 20 minutes.

Bailey et al. [21] Presented an automated classification technique as a solution for classifying malware binaries with an offline behavioral analysis in terms of system state changes (e.g., files written, processes created). Another related study concerning dynamic analysis is proposed by Kolbitsch et al. [23]. They suggested building classification models based on information flow between system calls. In the same context, Rieck et al. [15] proposed a learning-based approach for malware classification behavior. They used a labeled dataset of malicious files including 14 families. The behavior of the samples is monitored in a sandbox environment and the features are extracted from each generated behavioral report. They used the SVM algorithm for classification.

Norouzi et al. [24] proposed a data mining technique based on classification methods for identifying malware behavior. A proposed method has been introduced for transforming a malware behavior executive history XML file to an adaptable input for WEKA. This classification has been done using two datasets by specific algorithms such as Naïve Bayes, BayesNet, IB1, J48, SVM, and logistic regression. The evaluation results show the capability of this proposed technique to recognize malware and its behavior.

Firdausi et al. [25] proposed a behavior-based malware recognition technique using machine learning methods. The behavior of each malware on a sandbox will be

automatically studied and will produce behavior reports. The features have been selected from the generated report. The classification algorithms used in this research are k-NN, Naive Bayes, J48, SVM, and MLP (Multilayer Perceptron Neural Network). The experimental results for the best performance were attained by J48 with a recall of 95.9%, a false positive rate of 2.4%, a precision of 97.3%, and an accuracy of 96.8%.

Table 1. The Summary of the Related Works

	Paper	Technique	Classification methods	Malware type	Dataset	Size of data	Accuracy
Static approaches	[9]	Minimal contrast frequent pattern mining	Minimal Contrast Frequent Subgraph Miner (MCFS)	Exploit-based worm, Mass-mailing worm, IRC worm, Trojan	**Malware:** VX Heavens, Open malware, Malware Domain List **Benign:** Windows samples	**Malware:** 1,083 **Benign:** 1,000 **Total:** 2,083	92%
	[16]	N–Gram Analysis	Naïve Bayes, Instance-Based Learner (IBK), Decision Trees (J48), AdaBoost1, Random Forest	Portable Executable (viruses, worms, Trojan, Backdoors, etc)	**Malware:** VX Heavens **Benign:** executable from different genuine operating system Windows	**Malware:** 1,018 **Benign:** 1,120 **Total:** 2,138	99%
	[17]	Mining Format Information	J48, Adaboost, Bagging, Random Forest	Backdoor, Constructor, Virtool, DoS, Nuker, Flooder, Exploit, Hacktool, Worm, Trojan, Virus	**Malware:** VX Heavens **Benign:** Windows folder and Program Files folder and other legitimate software	**Malware:** 10,521 **Benign:** 8,592 **Total:** 19,113	91.1%
	[18]	PE header Information	Create his own algorithm	Unspecified	**Malware:** Unspecified **Benign:** Collected downloads.com and Softpedia	**Malware:** 5,598 **Benign:** 1,237 **Total:** 6,875	99.5%
Dynamic approaches	[25]	Behavior of malware	K-Nearest Neighbor, Naive Bayes, J48, Support Vector Machine, Multilayer Perceptron Neural Network (MLP)	Malicious windows Portable Executable Files format	**Malware:** Indonesian malware **Benign:** Collected from system files of Windows XP sp2	**Malware:** 220 **Benign:** 250 **Total:** 470	96.8%
	[26]	API call signatures.	Sequential Minimal Optimization (SMO), Naïve Bayes, K-Nearest Neighbor, Back-propagation Neural Networks, J48	Virus, Worm, Rootkit, Backdoor, Constructor, Exploit, Flooder, Trojan	**Malware:** VX Heavens **Benign:** Application, **Software:** Educational, Mathematical, etc.	**Malware:** 52,223 **Benign:** 15,480 **Total:** 66,703	98.5%
Hybrid Approaches	[33]	Deep learning based on hybrid malware analysis	Logistic Regression, Navie Bayes, K-Nearest Neighbor, Decision Tree, Ada Boost, Random Forest, Support Vector Machine, deep neural network, Convolutional neural network, Long short term memory and DeepImageMalDetect (DIMD)	Portable Executable files	**Malware:** Malimg, VirusSign, and Virusshare **Benign:** Windows samples	**Dataset1:** **Malware:** 118,717 **Benign:** 121,701 **Total:** 240,418 **Dataset2:** **Malware:** 50,792 **Benign:** 52,245 **Total:** 103,037	**Dataset1:** 96.3% **Dataset2:** 98.8%
	[30]	N-gram and API Call features	Naïve Bayes, J48, Random forest and Support Vector Machine	Backdoor, Virus, Rootkit, Trojan, Worm, Exploit and Other types	**Malware:** VX Heaven **Benign:** Download.com and Softpedia.com	**Malware:** 41,265 **Benign:** 10,920 **Total:** 52,185	98.6%
	[12]	Hybrid pattern-based text mining	All Nearest Neighbors, sequential pattern mining method, Hybrid DBScan	Polymorphic / Metamorphic and Other types of malicious files	Unspecified	**Malware:** 49 **Benign:** 28 **Polymorphic:** 105 **Total:** 182	98.36%

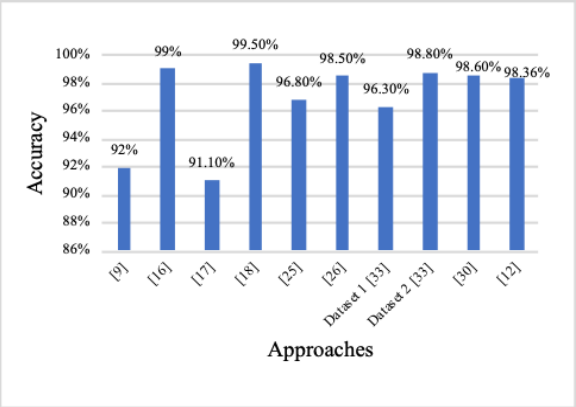

Fig.1. The Accuracy of the Different Approaches

Alazab et al. [26] present a dynamic method using several data mining methods to detect and classify zero-day malware based on the frequency of windows API calls. They used several classifiers: SMO with 4 kernels (PolyKernel, Normalized PolyKernel, Puk and Radial Basis Function), Naïve Bayes, K-NN, Backpropagation Neural Networks, and J48 decision tree. They evaluated their performances and the best experiments results have been achieved by SVM - Normalized PolyKernel with 98.5% of high true positive (TP) rate, 0.025 of low false positive (FP) rate.

Malhotra and Bajaj [12] proposed a hybrid method to classify malware and polymorphic/metamorphic files by generating "pydasm" report and collecting other parameters such as binary count, ICMP, op-code, etc. they used text mining to extract the instruction sets and applied pattern matching on the instruction's pattern in the generated reports to create a database for corresponding various files and function, they used signature-based pattern matching technique to evaluate the similarity score between any two files. Hybrid DBScan technique has been used on the selected features for classification. The experimental results of 183 samples, including 28 files are clean, 105 files are polymorphic and 49 files are malware, show 98.36% of accuracy, 100% of recall and 98.08% of precision.

A. The Summary of the Reviewed Approaches

There are several types of researches and methods that are made in the field of malware detection and classification. In this paragraph, we tried to combine and compare the results of the approaches that we examined previously, Table 1 presents the summary and the comparison of the different researches that study the classification of files as legitimate or malicious.

According to the Table 1, we note that the dataset source of the different approaches is almost the same, generally, the benign files are collected from the genuine operating system files and other legitimate software, the malicious files are collected from the different databases, e.g. VX Heaven, Virusshare, etc.

Based on the graph in Figure 1, we remark that all proposed approaches ended up with different results, we can conclude that It has not yet been established a standardized method to detect and classify malware. The accuracy depends on the used methods, implementation and on the features of malware.

III. Proposed Work

A. Research Methodology

Our work focuses on creating a classification model in order to distinguish between malicious and benign PE files (Executable Files) using machine learning algorithms, and perform detection with high precision in less time compared to the existing related studies. We can consider our classification model among the static-based detection approaches, which means that we can classify files without executing them.

Figure 2 shows the general architecture of our work, the process starts with the collection of the dataset that contains legitimate and malicious PE files, the second step describes the data preprocessing which contains several techniques: Feature extraction, Remove duplicate files, Handling missing data, Feature scaling and Feature selection these techniques are very important to get accurate results and to decrease the training time. Then we performed two experiments, in Experiment I, we need to split the dataset into two parts, a training set for building our training model and a test set to evaluate it. In Experiment II, the cross-validation technique is used to evaluate our model using the whole dataset.

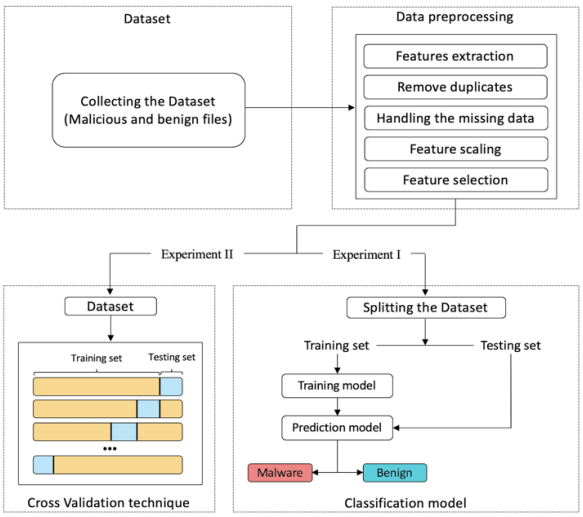

Fig.2. General Architecture

B. Dataset

The malware dataset is obtained from Virusshare database [27]. 83,401 samples that have been identified as malware in 2016, 2017, 2018, have been randomly selected from the database which includes different types of malware such as viruses, backdoors, ransomware, worms, Trojans, etc. with their sizes ranging from 1 KB to 200 MB. The 127,666 benign samples are collected from different genuine operating systems (Windows 7/8/8.1/10, Windows Server 2012/2016) and other trusted applications such as Microsoft Office, Skype, most popular apps

in 2018 downloaded from Microsoft Store, popular anti-viruses in 2018 and 2019, etc. After collecting 211,067 samples we worked with a Pefile [28] to extract the features from the header of each file such as SizeOfCode, SizeOfInitializedData, Characteristics, etc. We transform these extracted features as input for machine learning algorithms in order to distinguish between malicious and legitimate files.

C. Classification Model

Among the purposes of this work is to select an appropriate classifier with our situation in term of training time and accuracy, and to build a generalist model in order to avoid overfitting and underfitting problems. According to statistics that we are previously prepared in Table 1. We combined the most used classifiers in the field of malware detection in order to select the best classifier. We implemented 9 classifiers: Adaboost, Bagging, Decision Trees, Extra Trees, Naïve Bayes, Gradient Boosting, K-Nearest-Neighbors, Logistic Regression, and Random Forest.

The hyperparameters values of each classifier are not chosen randomly, we tuned them using a grid search technique which is an exhaustive searching through a manually specified list of the hyperparameters related to a learning algorithm. Generally, this technique allows us to optimize and choose suitable hyperparameters for a given model in order to get good results.

IV. Experimental Results

In order to prove the efficacy of our detection system, we performed two different experiments on 211,067 sets of data that contains both malware and legitimate PE files. All the experimental studies are conducted under Macbook Pro (Mojave v.10.14.2), Processor: Intel Core i7 2.2GHz (Turbo Boost up to 3.4GHz), Memory size: 16 GB, Graphics: Intel Iris Pro 1536 MB, and Storage: 256 GB SSD.

Table 2. The Experimental Results without Feature Selection

	True Positive rate	True Negative rate	Positive predictive value / Precision	Negative predictive value	F-Measure	Accuracy (ACC)	Training Time
Adaboost	99.54%	98.89%	98.86%	99.55%	99.20%	99.21%	16.41 s
Bagging	99.71%	99.65%	99.64%	99.72%	99.67%	99.68%	28.80 s
Decision Trees	99.59%	99.53%	99.52%	99.61%	99.56%	99.56%	2.07 s
Extra Trees	**99.78%**	99.67%	99.66%	**99.79%**	99.72%	99.72%	2.50 s
Naïve Bayes	98.04%	92.92%	93.05%	98%	95.48%	95.44%	**0.17 s**
Gradient Boosting	99.58%	98.58%	98.55%	99.59%	99.06%	99.07%	20.36 s
K-Nearest Neighbor	98.36%	97.97%	97.91%	98.41%	98.13%	98.16%	9.67 s
Logistic Regression	94.33%	84.49%	85.47%	93.90%	98.68%	89.33%	0.86 s
Random Forest	**99.78%**	**99.71%**	**99.70%**	99.78%	**99.74%**	**99.74%**	4.20 s

Table 3. The Experimental Results using Feature Selection

	True Positive rate	True Negative rate	Positive predictive value / Precision	Negative predictive value	F-Measure	Accuracy (ACC)	Training Time (Second)
Adaboost	99.52%	98.60%	98.56%	99.53%	99.04%	99.05%	5.68 s
Bagging	99.72%	99.47%	99.46%	99.73%	99.59%	99.60%	6.38 s
Decision Trees	99.50%	99.51%	99.49%	99.51%	99.49%	99.50%	0.60 s
Extra Trees	99.76%	99.56%	99.55%	99.77%	99.65%	99.66%	1.21 s
Naïve Bayes	98.38%	92.94%	93.09%	98.34%	95.66%	95.61%	**0.06 s**
Gradient Boosting	99.48%	98.42%	98.38%	99.49%	98.93%	98.94%	6.74 s
K-Nearest Neighbor	98.35%	98.02%	97.96%	98.40%	98.15%	98.18%	4.68 s
Logistic Regression	94.41%	84.44%	85.44%	93.98%	89.70%	89.34%	0.49 s
Random Forest	**99.77%**	**99.58%**	**99.57%**	**99.78%**	**99.67%**	**99.68%**	1.90 s

A. Experiment I

In experiment I, the dataset was randomly divided into two parts, 80% of the training set and 20% of the test set. We used 9 classifiers to build a training model for each algorithm. We evaluate all classifiers using the evaluation metrics that are extracted from the confusion matrix. Table 2 and Table 3 show the best experimental results that we obtained for each classifier according to the training time.

Table 2 presents the experimental results using all the features (without features selection technique), according to this recorded experimental results, we note that the Random forest classifier has the best results compared to other classifiers with a reasonable time (4.20 seconds); it provides 99.74% of accuracy, 99.78% of detection rate, 99.70% of precision and 99.71% of specificity. Otherwise, we note that the Extra trees classifier gives good results in a short time (2.50 seconds). After using the feature selection technique, we recorded again all the experimental results in Table 3 which shows all the best results due to the Random forest classifier with an accuracy of

99.68% in less than 2s.

We remark that after reducing the dimensions of our dataset under a feature selection technique, the training time was reduced, but it affects a little bit on some metrics values like accuracy, precision, etc. e.g. The time of Random forest has been reduced from 4.20 seconds to 1.90 seconds, but its accuracy has been decreased with 0.06%. In contrast, this technique can help us to improve the accuracy of some classifiers such as Naïve Bayes, K-NN, Logistic Regression.

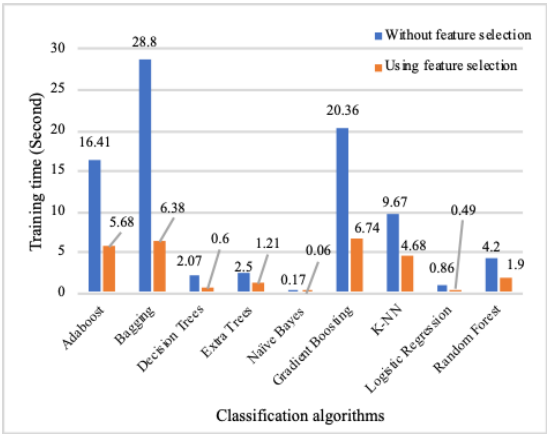

Fig.3. The Training Time of the Classification Methods

Based on the graph of Figure 3, we note also that the Naïve Bayes algorithm is the fastest algorithm in terms of training time. It allows us to build the training model in less than 0.2 seconds, but in term of accuracy, it remains among the weakest algorithms. Its accuracy is around 95.5%.

B. Experiment II

In the first experiment, the dataset was randomly divided into only two parts. The first one is for the training and the second one is for testing and evaluating the model. In this case, this method does not help to know if the model has overfitting or underfitting problems. Thus we cannot judge the performance of our model by evaluating only one part, that's why we perform another experiment using k-folds cross-validation technique which allows to use the entire dataset for training and validation.

In this experiment, K-fold cross-validation is used and we have chosen k = 10. After performing 10-fold cross-validation technique, we estimated the standard deviation and the average of all 10 folds for each classifier and we noted the experimental results in Table 4, which includes the value of standard deviation, the highest and the lowest values of the accuracy of each classifier. In this test, the random forest gives promising results compared to other classifiers.

Following experiment I and II, we remark that Random Forest and Extra Trees are the appropriate classifiers for our system in term of accuracy; on the other hand, there is no notable difference between the performance of all classifiers.

Table 4. The Experimental Results using the Cross-Validation Technique

	Standard deviation	Highest Accuracy	Lowest Accuracy	Average Accuracy
Adaboost	0.07%	99.32%	99.07%	99.15%
Bagging	0.02%	99.70%	99.63%	99.65%
Decision Trees	0.05%	99.63%	99.43%	99.53%
Extra Trees	0.03%	99.77%	99.65%	99.70%
Naïve Bayes	0.06%	95.67%	95.44%	95.57%
Gradient Boosting	0.06%	99.29%	99.02%	99.16%
K-Nearest Neighbor	0.05%	98.58%	98.39%	98.46%
Logistic Regression	0.17%	89.91%	89.25%	89.46%
Random Forest	0.03%	99.77%	99.67%	99.71%

V. Comparison with Related Works

In this paragraph, we give a comparison of our results with the most recent existing studies. Regarding our proposed system, we have performed three experiments on 211,067 sets of data, and the best results in term of accuracy were confirmed by Random Forest classifier which allows us to achieve an accuracy of 99.74% and 99.68% in experiment I and 99.71% in experiment II.

According to Table 5 and Figure 4, we note that the size of the dataset is different for each of these approaches that's why we made a comparison of the accuracies based on the size of the dataset. We did not comparing the training time because the experiment environments are different and could lead to an unfair time assessment. Despite the above, our system gave promising results with an accuracy up to 99.74%. In addition, we note that our classification model yields to accurate results in a reasonable time even if our dataset size is larger than the ones used in other studies.

Table 5. Comparison with Recent Approaches

		Accuracy	Size of the dataset
Our classification model		99.74%	211,067
2019 [33]	Dataset 1	96.3%	240,418
	Dataset 2	98.8%	103,037
2018 [32]		98.75%	4000
2018 [30]		98.6%	52,185
2017 [31]		95.35%	650
2017 [29]		96%	7,507
2016 [12]		98.36%	182
2016 [9]		92%	2,083

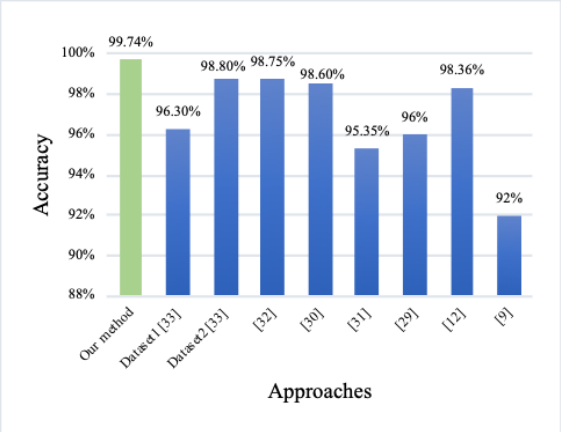

Fig.4. Accuracy Comparison

VI. Conclusion and Future Works

We summarize this paper reminding the outlines of our work which was started by studying the most popular related works of malware detection and classification. We created a detection model for PE files using a dataset of 211,067 programs, including 84,911 of legitimate files collected from different genuine operating systems, several trusted applications, etc. and 83,139 of malicious files which are collected from Virusshare. We prepared the dataset by extracting 54 features from each program using Pefile (Python module). After preparing our set of data, we partitioned them into a training set to build the model and a testing set to evaluate the performance of the training model. Finally, we performed two experiments; in the first one, we used 80% for the training set, and 20% for the testing set. In the second experiment, we used k-fold cross-validation technique to assess our system with more precision. The results have been recorded and discussed. We propose some directions for future work, among them:

- Reducing further the training time for specific environments.
- Investigating other new features to more accurately detect malware.
- Developing this classification model in order to make it able to classify different types of malicious files even for smart devices and smartphones.

References

[1] AV-TEST, https://www.av-test.org/en/statistics/malware/, Accessed: December, 2018.

[2] "Ransomware Attack Still Looms in Australia as Government Warns Wannacry Threat not Over", https://www.abc.net.au/news/2017-05-15/ransomware-attack-to-hit-victims-in-australia-government-says/8526346, 2017, Accessed: December, 2018.

[3] Gizmodo, "Today's Massive Ransomware Attack was Mostly Preventable; here's how to Avoid it", https://gizmodo.com/ today-s-massive-ransomware-attack-was-mostly-preventabl-1795179984, 2017, Accessed: December, 2018.

[4] G. Suarez-Tangil, J. E. Tapiador, P. Peris-Lopez, A. Ribagorda, "Evolution, Detection and Analysis of Malware For Smart Devices," IEEE Communications Surveys & Tutorials, Vol. 16, No. 2, pp. 961-987, 2014, DOI:10.1109/SURV.2013.101613.00077.

[5] Reinsurancene, "Total Wannacry Losses Pegged at $4 Billion", https://www.reinsurancene.ws/total-wannacry-losses-pegged-4-billion/, 2017, Accessed: December, 2018.

[6] D. Kushner, "The Real Story of Stuxnet," IEEE Spectrum, Vol. 50, No. 3, pp. 48-53, 2013, DOI: 10.1109/MSPEC.2013.6471059.

[7] J. Fruhlinger, "What is Stuxnet, who Created it and how does it Work?", csoonline , https://www.csoonline.com/article/ 3218104/what-is-stuxnet-who-created-it-and-how-does-it-work.html, 2017, Accessed: December, 2018.

[8] Statista , "Number of Smartphone Users Worldwide from 2014 to 2020", https://www.statista.com/statistics/330695/number-of-smartphone-users-worldwide/, 2019, Accessed: January, 2019.

[9] A. Hellal, L. B. Romdhane, "Minimal Contrast Frequent Pattern Mining for Malware Detection," Computers & Security, Vol. 62, pp. 19-32, 2016, DOI: 10.1016/j.cose.2016.06.004.

[10] S. Jain and Y. K. Meena. "Byte Level N-Gram Analysis for Malware Detection," Computer networks and intelligent computing, Springer, pages 51-59, 2011, DOI: 10.1007/978-3-642-22786-8_6.

[11] B. Kolosnjaji, A. Zarras, G. Webster,C. Eckert, "Deep Learning for Classification of Malware System Call Sequences," AI 2016: Advances in Artificial Intelligence. Lecture Notes in Computer Science, Springer, Cham, vol 9992, pp 137-149, 2016, DOI: 10.1007/978-3-319-50127-7_11.

[12] A. Malhotra, K. Bajaj, "A Hybrid Pattern Based Text

Mining Approach for Malware Detection Using DBscan," CSI Transactions on ICT, Vol. 4, pp. 141–149, 2016, DOI: 10.1007/s40012-016-0095-y.
[13] A. Shabtai, U. Kanonov, Y. Elovici, C. Glezer, Y. Weiss, ""Andromaly": A Behavioral Malware Detection Framework for Android Devices," Journal of Intelligent Information Systems, Vol. 38, pp. 161-190, 2012, DOI: 10.1007/s10844-010-0148-x.
[14] A. Souri, R. Hosseini, "A State-of-the-Art Survey of Malware Detection Approaches Using Data Mining Techniques," Human-centric Computing and Information Sciences, Vol. 8, pp. 1-22, 2018, DOI: 10.1186/s13673-018-0125-x.
[15] K. Rieck, T. Holz, C. Willems, P. Düssel, P. Laskov, "Learning and Classification of Malware Behavior," Proceedings of the 5th International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment (DIMVA), Paris, France Vol 5137. Springer, Berlin, Heidelberg, 2008, DOI: 10.1007/978-3-540-70542-0_6.
[16] S. Jain, Y. K. Meena, "Byte Level n–Gram Analysis for Malware Detection," Venugopal K.R., Patnaik L.M. (eds) Computer Networks and Intelligent Computing. ICIP 2011. Communications in Computer and Information Science, Springer, Berlin, Heidelberg, Vol 157, pp. 51-59, 2011, DOI: 10.1007/978-3-642-22786-8_6.
[17] J. Bai, J. Wang, G. Zou, "A Malware Detection Scheme Based on Mining Format Information," The Scientific World Journal, Vol. 2014, Article ID 260905, pp. 1-11, 2014, DOI: 10.1155/2014/260905.
[18] Y. Liao, "Pe-Header-Based Malware Study and Detection," Retrieved from the University of Georgia, 2012,
[19] A. Makandar, A. Patrot, "Malware Class Recognition Using Image Processing Techniques," International Conference on Data Management, Analytics and Innovation (ICDMAI), Pune, pp. 76-80, 2017, DOI: 10.1109/ICDMAI.2017.8073489.
[20] L. Nataraj, S. Karthikeyan, G.Jacob, B. S. Manjunath, "Malware images: visualization and automatic classification," Proceedings of the 8th International Symposium on Visualization for Cyber Security (VizSec '11). ACM, New York, NY, USA, Article 4, pp. 1-7, 2011, DOI: 10.1145/2016904.2016908.
[21] M. Bailey, J. Oberheide, J. Andersen, Z.M. Mao, F. Jahanian, J. Nazario, "Automated Classification and Analysis of Internet Malware," 12th International Symposium on Recent Advances in Intrusion Detection, Springer, Berlin, Heidelberg, Vol 4637, pp. 178-197, 2007, DOI: 10.1007/978-3-540-74320-0_10.
[22] M. S. Gadelrab, M. ElSheikh, M. A. Ghoneim, M. Rashwan, "BotCap: Machine Learning Approach for Botnet Detection Based on Statistical Features," International Journal of Computer Network and Information Security (IJCNIS), Vol.10, No.3, pp. 563-579, 2018
[23] C. Kolbitsch, P. M. Comparetti, C. Kruegel, E. Kirda, X. Y. Zhou, X. Wang, "Effective and Efficient Malware Detection at the End Host," USENIX security symposium, Vol 4, pages 351-366, 2009.
[24] M. Norouzi, A. Souri, M. S. Zamini, "A Data Mining Classification Approach for Behavioral Malware Detection," Journal of Computer Networks and Communications, Vol. 2016, Article ID 8069672, pp. 1-9, 2016, DOI: 10.1155/2016/8069672.
[25] I. Firdausi, C. lim, A. Erwin, A. S. Nugroho, "Analysis of Machine learning Techniques Used in Behavior-Based Malware Detection," 2010 Second International Conference on Advances in Computing, Control, and Telecommunication Technologies, Jakarta, pp. 201-203, 2010, DOI: 10.1109/ACT.2010.33.
[26] M. Alazab, S. Venkatraman, P. Watters, M. Alazab, "Zero-day Malware Detection Based on Supervised Learning Algorithms of API Call Signatures," Proceedings of the Ninth Australasian Data Mining Conference, Vol. 121, pp. 171-182, 2011.
[27] J-Michael Robert, "Virusshare", https://virusshare.com, 2018
[28] E. Carrera, "Pefile 2018.8.8", https://pypi.org/project/pefile/, 2018.
[29] T. Wüchner, A. Cisłak, M. Ochoa and A. Pretschner, "Leveraging Compression-Based Graph Mining for Behavior-Based Malware Detection," IEEE Transactions on Dependable and Secure Computing, vol. 16, no. 1, pp. 99-112, 2017, DOI: 10.1109/TDSC.2017.2675881.
[30] M. Chowdhury, A. Rahman, R. Islam, "Malware Analysis and Detection Using Data Mining and Machine Learning Classification," International Conference on Applications and Techniques in Cyber Security and Intelligence, Vol 580, pp. 266-274, 2018, DOI: 10.1007/978-3-319-67071-3_33.
[31] Bat-Erdene, Munkhbayar and Park, Hyundo and Li, Hongzhe and Lee, Heejo and Choi, Mahn-Soo, "Entropy Analysis to Classify Unknown Packing Algorithms for Malware Detection," International Journal of Information Security, Vol. 16, pp. 227—248, 2017, DOI: 10.1007/s10207-016-0330-4.
[32] B. R. Babak, S. Maryam, K. H. N. Mohammad, "Malware classification and detection using artificial neural network," Journal of Engineering Science and Technology, Vol. 13, pp. 14-23, 2018.
[33] R. Vinayakumar, M. Alazab, K. P. Soman, P. Poornachandran and S. Venkatraman, "Robust Intelligent Malware Detection Using Deep Learning," IEEE Access, vol. 7, pp. 46717-46738, 2019. DOI: 10.1109/ACCESS.2019.2906934

Authors' Profiles

Imad Abdessadki is a Ph.D. student at National School of Applied Sciences of Tangier - AbdelMalek Essaadi University (Morocco). He obtained a master's degree major 'CyberSecurity and cyberCriminality' in 2018. His research interests include Malware analysis, Mathematical modeling of the propagation of malware.

Saiida Lazaar started her scientific career with a research position at CNRS in France. After her Ph.D. in Applied Mathematics from Aix-Marseille I University, she held positions as a researcher with IFP in France, and with ONDRAF/ULB in Belgium. Currently, she is full Professor at National School of Applied Sciences of Tangier - AbdelMalek Essaadi University (Morocco), President-Funder of Association 'la Colombe pour la Promotion du Logiciel Libre', and Head of 'CyberSecurity and cyberCriminality' Master.

How to cite this paper: Imad Abdessadki, Saiida Lazaar, "A New Classification Based Model for Malicious PE Files

Detection", International Journal of Computer Network and Information Security(IJCNIS), Vol.11, No.6, pp.1-9, 2019.DOI: 10.5815/ijcnis.2019.06.01

