Detection and Classification of Malware Using Guided Whale Optimization Algorithm for Voting Ensemble

Marwa M. Eid[1*], M. I. Fath Allah[2]

[1] Faculty of Artificial Intelligence, Delta University for Science and Technology, Mansoura 35712, Egypt
[2] Communications and Electronics Department at Delta Higher Institute for Engineering and Technology, Mansoura- Egypt
Emails: marwa.3eeed@gmail.com; mismail1885@yahoo.com

Abstract

Malware is software that is designed to cause damage to computer systems. Locating malicious software is a crucial task in the cybersecurity industry. Malware authors and security experts are locked in a never-ending conflict. In order to combat modern malware, which often exhibits polymorphic behavior and a wide range of characteristics, novel countermeasures have had to be created. Here, we present a hybrid learning approach to malware detection and classification. In this scenario, we have merged the machine learning techniques of Random Forest and K-Nearest Neighbor Classifier to develop a hybrid learning model. We used current malware and an updated dataset of 10,000 examples of malicious and benign files, with 78 feature values and 6 different malware classes to deal with. We compared the model's results with those of current approaches after training it for both binary and multi-class classification. The suggested methodology may be utilized to create an anti-malware application that is capable of detecting malware on newly collected data.

Keywords: Cybersecurity; Malware detection; Machine learning; Hybrid learning; Classification; K-Nearest neighbor; Random forest; Metaheuristic optimization

1. Introduction

When it comes to today's sophisticated technical tools, safety is high on the list of priorities. In the absence of adequate protections, sensitive data runs the danger of being lost, stolen, or improperly utilized. Though the vast majority of apps on the internet are trustworthy, there are those that should be avoided at all costs. Malware, short for malicious software, is any computer program designed to do harm to a user's data. To be sure, malicious software is nothing new. The origins of malicious software may be traced back to the 1970s [1]. The goal of malware creators is to cause harm to their users. Experts in cyber security are always searching for new ways to combat malicious software. One of the numerous negative things that malware may do is erase data, steal data, and spy on the user [2]. Malware assaults caused an estimated $3 trillion in losses in 2015, and that number is projected to rise to $6 trillion by 2022 [3]. That's why it's crucial to identify malicious files and delete them before they can cause any damage.

One of the primary goals of cybersecurity is to keep digital infrastructure safe from malicious software. In a single strike, enough damage can be done to cause alterations to data and cause financial hardships. For this reason, experts in the field of cyber security are constantly tasked with developing novel approaches to strengthening existing anti-malware safeguards. The quick and accurate identification of malware is essential [4]. Existing methods of malware detection are insufficient because of the growing variety of malicious software. Machine learning's recent

Doi : https://doi.org/10.54216/JCIM.100102
Received: April 06, 2022 Accepted: July 25, 2022

progress has offered up novel opportunities in the realm of cyber security. Various machine learning methods can improve malware detection speed and accuracy. It's also given malicious code developers a chance to improve their tools. Because of this, researchers need to stay one step ahead of cybercriminals by developing innovative methods for detecting malware [5].

2. Literature Review

Scientists all throughout the world have shown a keen interest in studying malware. There are a lot of published studies in this area already. In order to stay ahead of evolving forms of malware, researchers must constantly update it. In this paper, we explore a new method for malware detection and outline the methodologies of standard machine learning. Modern malware detection methods and their inherent difficulties were discussed. Using the Brazilian malware dataset, seven machine learning models are trained in [2]. They demonstrated a comparison of various models. Unsupervised learning for detection was used in [3] to suggest op-code frequency as a feature vector. There was also a comparison between machine learning and deep learning for detection that was demonstrated. K-Nearest Neighbor, Decision Tree, and Support Vector Machine were all machine learning models, and their differences were compared in [4]. The authors of [5] propose using hardware to aid in the identification of malware. By monitoring and classifying memory access patterns with machine learning, they established a novel framework for hardware-assisted malware detection. Shared closest neighbor clustering approach is used to identify novel malware families by the developers of [6]. They used grayscale photos and an opcode n-gram approach to analyze the data. In [7], the authors employ a cluster tree to combine many distinct kinds of deep learning systems for malware detection, creating a system with several levels. Data mining techniques, including behavior-based and signature-based methods, are surveyed in [8] as a means of detecting malware.

The developers of [9] describe a web-based platform for finding malware on Android devices. To identify malicious software, they employed API calls and permissions as characteristics and the furthest first clustering, nonlinear ensemble decision tree forest technique. In [10], the authors dissect ransomware, examining its inner workings, trends, attack methods, and potential countermeasures. There was a comprehensive analysis of the various methods for finding ransomware. According to [11], static approaches for identifying malware families form the basis of a deep learning-based malware detection methodology. On model training, they opted for the CNN technique. The authors of [12] assessed the state of the art in machine learning and presented a hybrid approach based on convolutional neural networks and long short-term memory. Using neural networks as their first-stage classifiers, the authors of referenced work [13] developed an ensemble learning-based framework and investigated 15 machine learning models as their final-stage classifiers. With their unique architecture for Android malware detection introduced in [14], the authors have made some interesting contributions to the field. For malware detection, their technology employs a multi-channel deep learning approach. The authors of the paper [15] classified malware families based on images taken from a dataset containing malware. For the multinomial classification problem, they employed deep learning models using the L2-SVM in the last output layer.

Windows-compatible executables, object code, and other file types are stored in the Portable Executable (PE) file format [1]. The PE file format stores executable code together with the metadata Windows needs to run it. The PE file stores a variety of information, such as references to dynamic libraries, import tables, resource information, and so on. [2] Figure 1 displays the contents of a PE file. A PE file's contents are divided into two distinct parts: Headers and Sections. File and section table attributes are listed in the header. The program's source code and associated data may be found in the component's section. Malware analysis is essential for learning how malware operates and how it behaves. There are millions of pieces of new malware and variants of known malware floating around the web. To this end, dissecting their constituent parts is essential. Two distinct properties distinguish malware: I those that remain constant regardless of context, and (ii) those that change over time [1]. Fig. 2 is an instance of both static and dynamic characteristics.

Doi : https://doi.org/10.54216/JCIM.100102
Received: April 06, 2022 Accepted: July 25, 2022

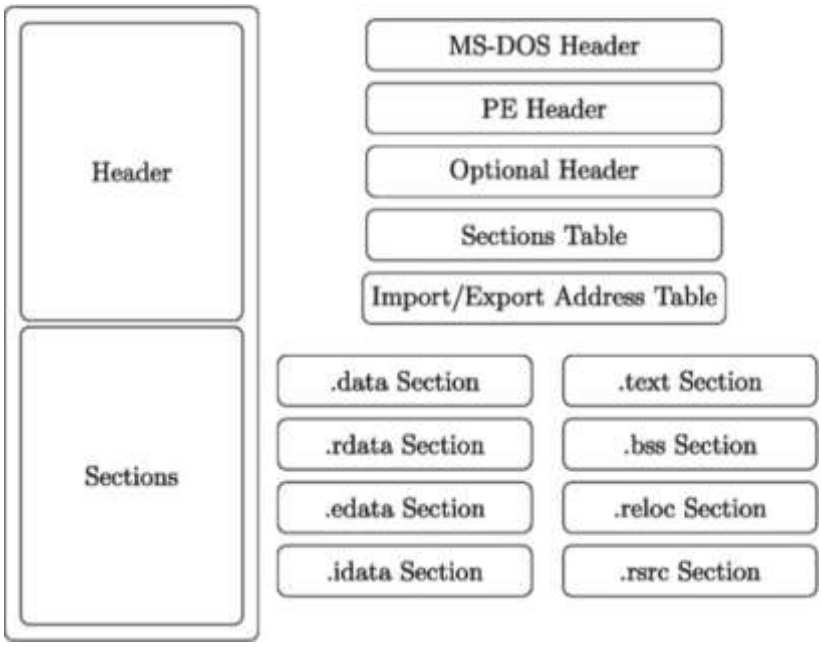

Figure 1: PE file format

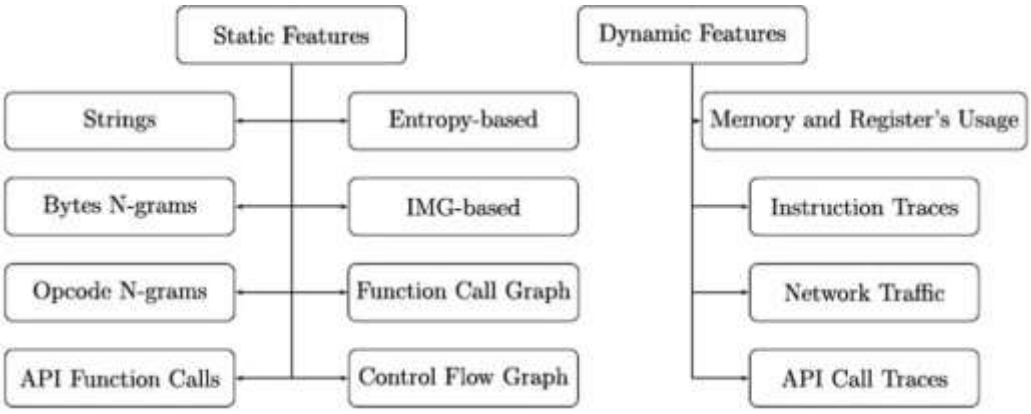

Figure 2: Malware feature types

Fixed Characteristics. The static properties of malware are the parts that can be gleaned through a static analysis of its structure, without running the code. These data are static and necessary for the virus to function. Finding the static features may be done in a number of different methods, including converting machine code into assembly language, looking for sequences of strings inside the program, analysing the PE file headers and sections that hold the information of the file, and so on. [6]. Relating to change or motion. Information about malware that is only detectable when it is being executed is called dynamic characteristics. This is necessary if the malware in question fails to yield useful data through static analysis. It's important to perform this surgery in a secure setting because of the inherent danger it poses. If not, it will cause problems for the active system. Typically, this takes place in a simulated or "sandbox" setting. Data such as application programming interface (API) calls, network traffic, processes, instructions, registry changes, etc. are all part of the dynamic characteristics. Risks remain while performing dynamic analysis, no matter how careful one is. [4].

3. Proposed Methodology

In order to classify a PE file as malicious or safe, the suggested model reads in the file's data values, which were obtained from the dataset, and then generates a classification. In Fig. 3, the straightforward architecture of the system is shown.

Doi : https://doi.org/10.54216/JCIM.100102
Received: April 06, 2022 Accepted: July 25, 2022

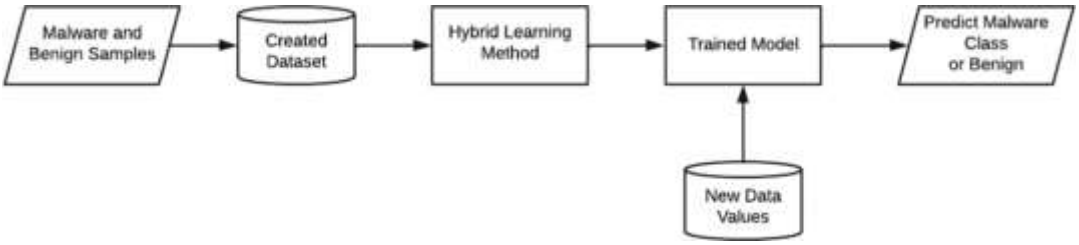

Figure 3: Overview of the system

Malware samples are used to generate this dataset. The data about safe files is gathered from the internet and merged with the data on malicious files. Before the approach is applied, the data undergoes processing to extract relevant features. To prepare the model for usage, it is trained on the dataset utilizing a hybrid learning technique, and then the dataset is exported. The model is then put to the test on fresh malware data, for which it makes predictions about whether or not the data belongs to the malware or benign classes.

A. The Proposed Ensemble Model

We start by showing you the data we used to boat-train our classifiers after we let go, then we walk you through the processes we propose doing, and lastly we show you the results. Three distinct classification strategies were employed in the uploading of this image: Different kinds of classifiers include decision trees (DT), multilayer perceptrons (MLP), and support vector machines (SVM). We employ stochastic fractal search (SFS) in conjunction with the whale optimization technique to boost the weight of these classifiers' votes inside an ensemble model (WOA).

B. Support Vector Machines (SVM)

Commonly used in the supervised learning domain, support vector machines (SVMs) are a type of classification software with a strong track record. Basically, it uses a collection of training data that has previously been divided into two groups to categorize novel samples. When applied to a dataset of examples with varying labels, the SVM training method constructs a model for prediction. Support vector machines (SVMs) create a hyperplane or a series of hyperplanes in a high-dimensional space to help in the classification process. Oversimplified steps in the SVM process are depicted at the top of Figure 4.

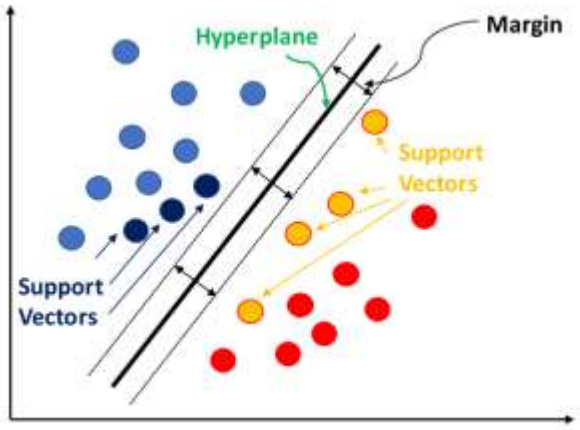

Figure 4: Structure of support vector machines.

C. Multilayer Perceptron (MLP)

A neural network is, in one sense, a collection of interconnected systems (neurons) that communicate with one another (synapses). It is common practice to employ artificial neural networks, which model themselves after the human nervous system, to recreate estimates because of how closely they mimic the real thing. Every artificial neural network consists of three distinct layers: the input layer, the hidden layer, and the output layer. It uses a collection of input layer nodes to build an activation function. Placed between the input and output layers, the weighting

Doi : https://doi.org/10.54216/JCIM.100102
Received: April 06, 2022 Accepted: July 25, 2022

layer makes the former more influential. One of the last parts of the process is the output layer. Figure 5 shows the many connected nodes of a multilayer neural network.

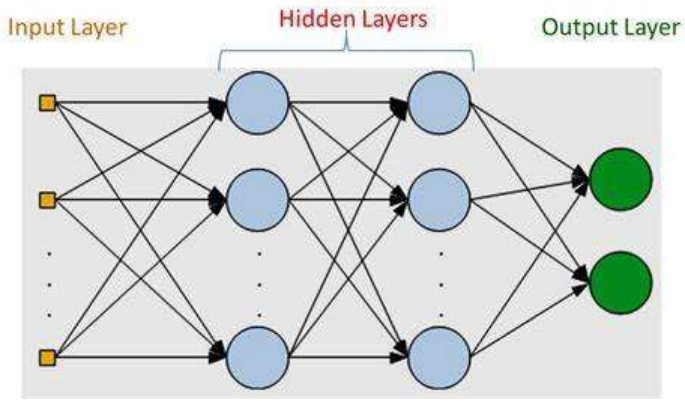

Figure 5: Structure of a multilayer neural network.

D. Decision Trees (DT)

In this method, the most promising nodes of a decision tree are found using a greedy search. To ensure that every information fits neatly into its designated containers, the partitioning process is iterated from most particular to least specific. It's quite likely that all data points will be clustered together if the decision tree is sufficiently sophisticated. The leaves, or data points, that correspond to a certain category may be quickly and easily identified in a succinct tree. When a tree is young, it may be simple to achieve this degree of purity, but as it matures, it becomes increasingly difficult to maintain it, and as a result, there is often not enough information held inside a particular subtree. The incoherent character of the data almost always leads to overfitting. Occam's Razor, which warns against "entities being multiplied beyond necessity," is in line with the preference for compact trees in decision trees. Since the most elementary explanation isn't necessarily the best, it's important to keep a decision tree as basic as possible. Pruning, or the removal of limbs that produce offspring with few admirable traits, is an effective method for accomplishing both goals (reducing complexity and preventing overfitting). After that is complete, cross-validation may be used to assess the model's precision. One way to maintain reliable decision trees is the random forest technique, which ensures that each tree in the ensemble is distinct to boost the classifier's prognostic accuracy. A simplified version of the decision tree seen in Figure 6 is presented here.

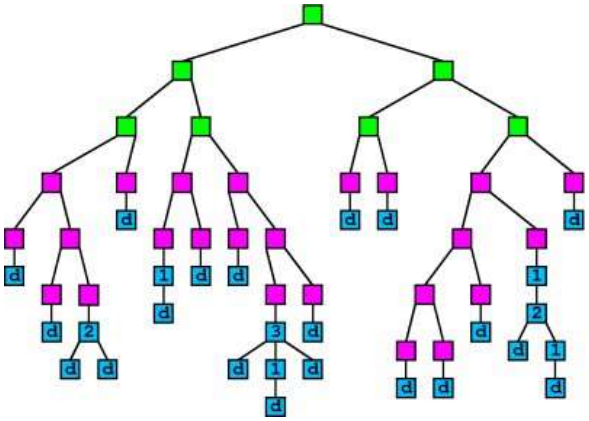

Figure 6: Structure of a decision tree.

E. Whale Optimization Algorithm (WOA)

In computer science and mathematical optimization, metaheuristics are used to locate, create, or choose a heuristic (partial search algorithm) that may provide an appropriate solution to an optimization issue, although lacking in either knowledge or processing power. With a purely systematic method, it would be difficult to sample a meaningful subset of the enormous potential answers; but, with the help of metaheuristics, we can. Metaheuristics are often more flexible than other optimization methods since they may be used in a variety of settings without much preparation

Doi : https://doi.org/10.54216/JCIM.100102
Received: April 06, 2022 Accepted: July 25, 2022

or a firm understanding of the optimization problem at hand. The finding of a globally optimal solution for a class of problems is not guaranteed when using metaheuristics instead of optimization techniques and iterative approaches. Stochastic optimization, used by many different metaheuristics, can cause the ultimately discovered solution to shift depending on the values of the random variables generated. Metaheuristics are a technique used in combinatorial optimization to explore possible solutions; they can be more efficient than optimization algorithms, iterative methods, and even simple heuristics when it comes to finding good ones. Because of this, they might be alternatives for dealing with optimization concerns. The issue has been extensively written about. The majority of published metaheuristics publications are inherently experimental since they report the author's own experiences implementing the algorithm in practice. However, there are also some formal theoretical conclusions accessible that provide insight into issues like convergence and whether or not the global optimum is indeed reachable. There has been a flurry of new articles proposing a wide range of metaheuristic tactics, all of which have the potential to contribute something novel and beneficial to the discipline. Some of the problems that have plagued previous research on this subject include vague terminology, limited coverage of pivotal points, dubious methodology, and a lack of citations for prior studies.

4. Results

Our research makes advantage of a recently updated collection of malicious and clean code. There are 78 feature variables and 6 malware classifications in the dataset's 10000 samples of malicious and benign data. Samples of malware were taken from Virus Share. In Table 1 we see a subset of this data set.

Table 1: Sample dataset

Name	Checksum	SizeOfImage	SizeofCode	DllChar	SizeOfIntData	Malware
b318a5110823c476f2702ad491d4a0f7	945794	905216	132608	32832	757760	1
efb3545d21bf33c028a175f478000b80	168950	180224	88576	33088	76800	1
c611758090ae3588d207e3afd43fa286	4433135	159744	65024	32768	49152	1
7af5a30b30a606a9b4a3464f4ec3940d	589182	671744	24064	32768	308224	1
e8af48b10488603cb9b37205b69bbd7c	331077	335872	7680	34112	306688	1

There are a few measures that may be used to assess how well a model is working. Confusion matrices display the testing results of a training model in terms of its true positive, true negative, false positive, and false negative counts. Accuracy, precision, recall, and the f1 score may all be determined from these measurements. Tabular displays of the findings are provided in Tables 2, 3, and 4.

Table 2: Classification results using the proposed method compared to other methods

	Accuracy	Sensitivity	Specificity	Pvalue	Nvalue	F-score
NN	0.8495	0.9901	0.7143	0.7692	0.9868	0.8658
SVM	0.8377	0.9901	0.6667	0.7692	0.9836	0.8658
DT	0.9091	0.9901	0.8000	0.8696	0.9836	0.9259
SFS-Guided WOA	0.9496	0.9901	0.8889	0.9302	0.9836	0.9592

Statistical evidence supporting the voting ensemble classifier's superiority is presented in Table 3. With the optimum voting ensemble that was presented, we get markedly better results.

Table 3: Statistical analysis of the results recorded by the proposed method

	NN	SVM	DT	SFS-Guided WOA
Number of values	10	10	10	10
Minimum	0.8495	0.8377	0.9091	0.9496
25% Percentile	0.8495	0.8377	0.9091	0.9496
Median	0.8495	0.8377	0.9091	0.9496

Doi : https://doi.org/10.54216/JCIM.100102
Received: April 06, 2022 Accepted: July 25, 2022

75% Percentile	0.8495	0.8402	0.9116	0.9496
Maximum	0.8595	0.8577	0.9291	0.9496
Range	0.01	0.02	0.02	0
10% Percentile	0.8495	0.8377	0.9091	0.9496
90% Percentile	0.8585	0.8567	0.9281	0.9496
95% CI of median				
Actual confidence level	97.85%	97.85%	97.85%	97.85%
Lower confidence limit	0.8495	0.8377	0.9091	0.9496
Upper confidence limit	0.8495	0.8477	0.9191	0.9496
Mean	0.8505	0.8407	0.9121	0.9496
Std. Deviation	0.003162	0.006749	0.006749	0
Std. Error of Mean	0.001	0.002134	0.002134	0
Lower 95% CI of mean	0.8483	0.8359	0.9073	0.9496
Upper 95% CI of mean	0.8528	0.8455	0.9169	0.9496
Coefficient of variation	0.3718%	0.8028%	0.7400%	0.000%
Geometric mean	0.8505	0.8407	0.9121	0.9496
Geometric SD factor	1.004	1.008	1.007	1
Lower 95% CI of geo. mean	0.8483	0.8359	0.9073	0.9496
Upper 95% CI of geo. mean	0.8528	0.8455	0.9169	0.9496
Harmonic mean	0.8505	0.8406	0.912	0.9496
Lower 95% CI of harm. mean	0.8483	0.8359	0.9073	0.9496
Upper 95% CI of harm. mean	0.8528	0.8454	0.9168	0.9496
Quadratic mean	0.8505	0.8407	0.9121	0.9496
Lower 95% CI of quad. mean	0.8482	0.8358	0.9072	0.9496
Upper 95% CI of quad. mean	0.8528	0.8456	0.917	0.9496
Skewness	3.162	2.277	2.277	
Kurtosis	10	4.765	4.765	
Sum	8.505	8.407	9.121	9.496

A comparison of the suggested technique to the alternatives is made using the Wilcoxon signed-rank test. Table 4 displays the data gathered throughout this study. The p-value listed in the table is the supporting evidence.

Table 4: Wilcoxon signed rank test of the recorded results of the proposed method

	NN	SVM	DT	SFS-Guided WOA
Theoretical median	0	0	0	0
Actual median	0.8495	0.8377	0.9091	0.9496
Number of values	10	10	10	10
Wilcoxon Signed Rank Test				
Sum of signed ranks (W)	55	55	55	55
Sum of positive ranks	55	55	55	55
Sum of negative ranks	0	0	0	0
P value (two tailed)	0.002	0.002	0.002	0.002
Exact or estimate?	Exact	Exact	Exact	Exact
P value summary	**	**	**	**

Doi : https://doi.org/10.54216/JCIM.100102
Received: April 06, 2022 Accepted: July 25, 2022

Significant (alpha=0.05)?	Yes	Yes	Yes	Yes
How big is the discrepancy?				
Discrepancy	0.8495	0.8377	0.9091	0.9496

The Figure 7 graph shows how the optimal voting ensemble classifier performs in comparison to the reference models. An illustration of the improved efficiency of the suggested approach is shown below.

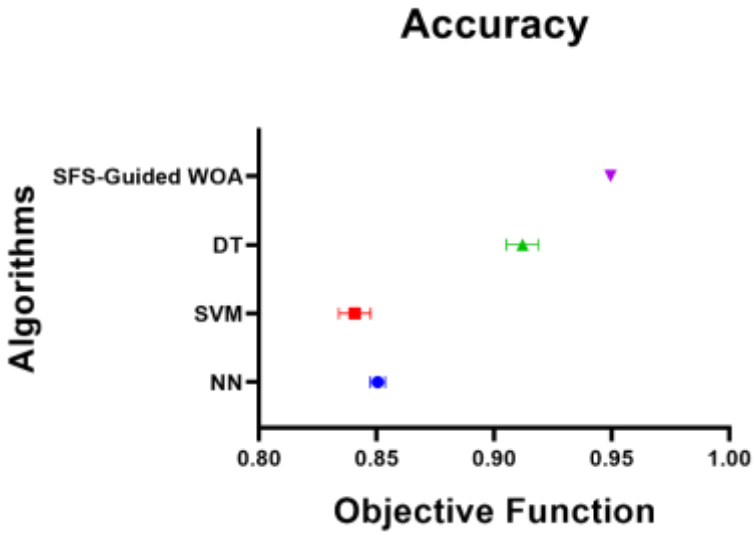

Figure 7: The accuracy of the proposed method compared to other methods

5. Conclusion

When it comes to cyber security, the issue of malware takes centre stage. Intelligent approaches to identify and categorize emerging malware are in high demand. If malicious software is not caught quickly, it can do significant damage [16]. In this investigation, many machine learning strategies for malware detection were analysed. We compared our suggested hybrid learning strategy to existing machine learning approaches and found that it outperformed the competition. Our algorithm has a 95% success rate for identifying multi-class entities, including new malware. There's a ton of room for growth in this study. Multiple enhancements are possible. Android malware, macOS malware, etc., are all open areas of research. Additional data samples and types of malware can be added to the existing dataset for even greater accuracy. The model can be used as the basis for an automated system or antimalware software.

Funding: "This research received no external funding"

Conflicts of Interest: "The authors declare no conflict of interest."

References

[1] Gibert, D., Mateu, C., Planes, J.: The rise of machine learning for detection and classification of malware: research developments, trends and challenges. Journal of Network and Computer Applications 153, 102526 (2020). 1084-8045. https://doi.org/10.1016/j.jnca.2019.102526

[2] Kumar, A., et al.: Malware Detection Using Machine Learning. In: Villazón-Terrazas, B., Ortiz-Rodríguez, F., Tiwari, S.M., Shandilya, S.K. (eds.) KGSWC 2020. CCIS, vol. 1232, pp. 61–71. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-65384-2_5

[3] Rathore, H., Agarwal, S., Sahay, S.K., Sewak, M.: Malware Detection Using Machine Learning and Deep Learning. In: Mondal, A., Gupta, H., Srivastava, J., Reddy, P.K., Somayajulu, D.V.L.N. (eds.) BDA 2018. LNCS, vol. 11297, pp. 402–411. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-04780-1_28

[4] Selamat, N., Ali, F.: Comparison of malware detection techniques using machine learning algorithm. Indonesian Journal of Electrical Engineering and Computer Science 16, 435 (2019). https://doi.org/10.11591/ijeecs.v16.i1.pp435-440.
[5] Xu, Z., Ray, S., Subramanyan, P., Malik, S.: Malware detection using machine learning based analysis of virtual memory access patterns. In: Design, Automation Test in Europe Conference Exhibition, pp. 169–174 (2017). https://doi.org/10.23919/DATE.2017.7926977
[6] Liu, L.,Wang, B.-S.,Yu,B., Zhong, Q.-X.: Automatic malware classification and new malware detection using machine learning. Frontiers of Info. Technol. Elec. Eng. 18, 1336–1347 (2017). https://doi.org/10.1631/FITEE.1601325
[7] Zhong,W., Gu, F.: A multi-level deep learning system for malware detection. Expert Systems with Applications 133, 151–162 (2019). ISSN: 0957–4174. https://doi.org/10.1016/j.eswa.2019.04.064
[8] Souri, A., Hosseini, R.: A state-of-the-art survey of malware detection approaches using data mining techniques. HCIS 8(1), 1–22 (2018). https://doi.org/10.1186/s13673-018-0125-x
[9] Mahindru, A., Sangal, A.L.: MLDroid—framework for Android malware detection using machine learning techniques. Neural Comput. Appl. 33(10), 5183–5240 (2020). https://doi.org/10.1007/s00521-020-05309-4
[10] Maigida, A., Abdulhamid, S., Olalere, M., Alhassan, K., Chiroma, H., Dada, E.: Systematic literature review and metadata analysis of ransomware attacks and detection mechanisms. Journal of Reliable Intelligent Environments 5, 67-89 (2019). https://doi.org/10.1007/s40860-019-00080-3
[11] Vinayakumar, R., Alazab, M., Soman, K.P., Poornachandran, P., Venkatraman, S.: Robust intelligent malware detection using deep learning. IEEE Access 7, 46717–46738 (2019). https://doi.org/10.1109/ACCESS.2019.2906934
[12] Rafique, M.F., Ali, M., Qureshi, A.S., Khan, A., Mirza, A.M.: Malware classification using deep learning based feature extraction and wrapper based feature selection technique (2019). https://doi.org/10.48550/ARXIV.1910.10958
[13] Azeez, N.A., Odufuwa, O.E., Misra, S., Oluranti, J., Damaševičius, R.:Windows pe malware detection using ensemble learning. Informatics 8(1) (2021). ISSN: 2227-9709. https://doi.org/10.3390/informatics8010010
[14] Kim, T., Kang, B., Rho, M., Sezer, S., Im, E.G.: A multimodal deep learning method for android malware detection using various features. IEEE Trans. Inf. Forensics Secur. 14(3), 773–788 (2019). https://doi.org/10.1109/TIFS.2018.2866319
[15] Agarap, A.F.: Towards building an intelligent anti-malware system: A deep learning approach using support vector machine (SVM) for malware classification (2018). https://doi.org/10.48550/ARXIV.1801.00318
[16] Vasant, P., Zelinka, I., Weber, G.-W. (eds.): ICO 2021. LNNS, vol. 371. Springer, Cham (2022). https://doi.org/10.1007/978-3-030-93247-3

Doi : https://doi.org/10.54216/JCIM.100102
Received: April 06, 2022 Accepted: July 25, 2022