Available online at www.garmian.edu.krd

Passer Journal

Passer 6 (Issue 1) (2024) 130-137

http://passer.garmian.edu.krd/

Obfuscated Malware Memory Detection Employing Lazy Instance Based Learner Algorithm Based on Manhattan Distance Function

Hardi Sabah Talabani[1*], Hezha M.Tareq Abdulhadi[2,3], Muhammed Hussein Ali[4]

[1]*Department of Computer Science, College of Science, Charmo University, Sulaimani, Kurdistan Region, Iraq.*
[2]*Department of Information Technology, National Institute of Technology, Sulaymaniyah, Kurdistan Region, Iraq.*
[3]*Department of Information Technology, Bright Technical and Vocational Institute, Sulaymaniyah, Kurdistan Region, Iraq.*
[4]*Computer Engineering Technique, Mazaya University College, Nasiriyah, Iraq.*

Received 27 March 2023; revised 18 December 2023;
accepted 12 February 2024; available online 17 February 2024

DOI: 10.24271/PSR.2023.391018.1296

ABSTRACT

Malware is a severe threat to the network and host system security. It is frequently the primary cause of many events, such as Distributed Denial-of-Service attacks (DDoS), spam emails, etc. The detection and elimination of Malware are the subjects of intensive study. As a result, many antivirus programs have been created to help identify and remove Malware. The issue with this antivirus software is that it uses an obsolete method of detecting Malware, the signature-matching approach, which the primary forms of code obfuscation may deceive. Since then, this has resulted in the creation of a new generation of metamorphic and polymorphic Malware. In this paper, we investigated using the Instance-Based Learner (IBK) algorithm for detecting obfuscated Malware in a given dataset. Utilizing the Lazy IBK technique in malware detection is beneficial because the algorithm can accurately detect and classify the obfuscated Malware in the dataset using the Manhattan Distance function, one of the most well-known distance metric functions for measuring the distance between points. We analyzed an obfuscated malware dataset of 58,596 records selected from 3 malware categories. The algorithm was illustrated on the dataset utilizing 10-fold cross-validation. The results demonstrate that the proposed algorithm can quickly and accurately detect obfuscated Malware with an accuracy of 99.99%, a precision of 100%, and a recall of 100%, respectively.

Keywords: Obfuscated Malware Memory Dataset, Malware Detection, Lazy IBK Algorithm, Manhattan Distance.

1. Introduction

Malware, which first appeared in the 1980s, has emerged as one of the most significant points in the cybersecurity field since its introduction. Malware refers to any harmful software used by cybercriminals and is designed to cause damage to a computer system or an individual user by carrying out various illegal actions. Malware has grown as quickly as technology and internet access have progressed, regardless of the current protection solutions[1]. In addition, the capability of Malware to bypass detection techniques has contributed to the complexity of the malware detection process. Because there is a wide variety of malware families and classifications, it is necessary to cover all your bases. Malware has several types, including Botnets, Spyware, Worms, Bots, Viruses, Rootkits, Trojan Horses, Ransomware, etc. The malware families have many functionalities, such as accessing the information, access prevention for authorized users, or committing other cybercrimes.

* Corresponding author
E-mail address: hardi.sabah@charmouniversity.org (Instructor).
Peer-reviewed under the responsibility of the University of Garmian.

Concentrating on categories and families is the most effective method for detecting Malware to prevent and halt it in the future[2]. Because manual detection techniques require significant time and effort, various learning algorithms, such as machine learning and deep learning, are utilized. These systems can extract intelligent insights from the data in an intuitive way. Finding out which training data to feed the system to produce the fastest and most accurate evaluation is the primary goal of these learning systems[3]. Machine learning techniques use a collection of features that may be examined with a vast sample size to compare and contrast differences.

Additionally, these features may be entered in various forms, which is one of the considerations for choosing the machine learning system that should be employed[4]. Some machine learning algorithms prioritize speed, while others prioritize accuracy and precision in collecting and analyzing data. Because of this, selecting the various algorithms that match the aim and the kind of input affects the system's results[5]. There is a machine learning model that fits these aims of detection and characterization, which is the IBK machine learning classifier belongs to the lazy learning technique, also known as the K

Nearest-Neighbor classifier(K-NN), which uses a distance metric function for measuring the distance between the points, here Manhattan Distance is used for the measurement[6]. Several techniques are available for obfuscated malware detection relying on memory analysis[7-8].

On the other hand, most works need more complexity and low accuracy in classifying the data, which is unsuitable for that purpose[9]. It is better to propose a robust and accurate model for detecting obfuscated Malware by selecting the appropriate classifier[10-11]. The significant contribution of this study is building an accurate model to detect the obfuscated Malware from the dataset used and then explaining the performance measurements obtained from the classification process[12-13]. The outline of the paper is organized as follows. Section 2 describes the Literature Review; the methodology is demonstrated in Section 3; experiments and results are presented in Section 4; and, finally, our conclusion and future work are discussed in Sections 5 and 6, respectively.

2. Related Works

Ever since its creation, Malware has received significant attention in the realm of cybersecurity owing to the many distribution methods and categories it encompasses. Even though there are several detection techniques, each has its own unique set of obstacles. Within the framework of the research gap, the mechanism for detecting malware memory detection has been addressed in several studies. At the same time, the algorithm used in this paper and the extent of its effectiveness in this field have yet to be discussed in previous studies. This section examines the remaining concerns and challenges in this study area and highlights related efforts on malware detection using memory analysis. On the other hand, other techniques are used in this field with their accuracy and outcomes.

Capturing a snapshot and extracting features from memory is a strong understanding of the activities in the system, which is called the memory analysis technique. explains that while all critical information is stored in memory, malware memory analysis is still reliable. Memory analysis that isn't done in real-time necessitates snapshots, and taking these snapshots is critical to ensuring that memory files aren't harmed. The memory analysis process could be changed by affected memory files, eliminating the analysis's reliability[14].

In[15], they presented a memory-analysis plugin that makes the analysis process straightforward. This plugin concentrates on the specifics of heap objects in memory, which might assist a memory analyst in comprehending what happens in the system memory. Memory analysis can be employed in various approaches and techniques to explore what happened to a victim.

In[16], the study of volatile memory to obtain information on shreds of evidence in social media was explained. The application created in this research concentrates specifically on volatile memory analysis to acquire social media evidence.

In[17] , they introduce an intriguing notion of memory dump preprocessing with two various procedures, making the analysis process faster and easier by relocating file objects. Guided De-Relocation was the first strategy, which specified a new space for the information. Also, Linear Sweep De-Relocation was the second to search memory for a storage location. Memory forensics tools can change the performance of memory analysis, yet they may still be modified and enhanced to be faster, more effective, and easier to employ.

Because classifying malware analysis can be time-consuming and inaccurate,authors in[18] recommend vectoring assembly source code employing the Long Short-Term Memory-Based (LSTM) approach for malware classification. Compared to alternative techniques, the accuracy increased by 0.5 percent when using word2vec with the LSTM system.

In[19], a platform for detecting Malware was developed by analyzing online virtual memory access patterns employing machine learning techniques. They applied the platform to an application-specific malware detection targeting detection of Malware infected with identified applications. Their experiments focused on two significant malware groups: kernel rootkits and memory-corruption attacks on user programs. Their framework detection rate was 99.0%, with less than 5% false positives.

In[20], deep learning was employed to detect Malware in cloud computing runs in the virtual machine-monitor layer; executing malware extracts memory snapshots of virtual machines. It converts the picture to grayscale images. They used a Convolutional Neural Network (CNN) to train the model. Without runtime overhead, their malware detector was secure and accurate; in their experiments, they obtained a high accuracy for malware detection, which was 0.905 percent.

In[21], they used a novel method for selecting a relevant feature from an Intrusion Detection dataset to reduce the complexity, a correlation-based feature selection, and a classifier subset-evaluation approach. Then, two famous classifiers were applied to the minimized dataset: Multilayer Perceptron (MLP) and Instance-Based Learning (IBK) algorithms. The experimental results showed the highest accuracy for the IBK classifier, 99.87%.

In[22], seven different data mining techniques were employed to predict people's interests for several pages on Facebook specified by info properties. They evaluated the performances of the algorithms utilizing a dataset with 10172 instances with six attributes. The classifiers used were KStar, LWL, IBK, J48, Naïve Bayes, Bayes Net, and Decision Table. The experimental results revealed that the algorithms' performances were reasonable, including IBK, with an accuracy of 98.21%.

In[23], various data mining techniques were used to analyze the Breast Cancer dataset. They Preprocessed and extracted features for optimizing the data and preparing the dataset for classification[24]. The classification process was done to evaluate the classifiers. IBK obtained the highest accuracy rate among the classifiers, with 98.2 percent.

In[25], they used a Lazy Classifier, including the IBK algorithm, to classify a diabetes dataset. They used tenfold cross-validation to split the dataset[26]. Their experimental results on the dataset revealed that the IBK algorithm gained the highest marks with 99 percent accuracy.

According to[27], they recommend a deep, productive model that brings together global and local features to determine malware variants effectively. The dual code arrays effectively transform Malware into an image representing global characteristics with a predefined hidden space. It extracts local features using binary code sequences while converting Malware into an image to represent global characteristics efficiently with predefined latent space[28]. They combined and entered the two extracted features with their characteristics into the malware detector. Their experimental results were reasonable, with a 97.47% accuracy rate.

Researchers[29] recommended methods: the time it takes for Malware to be detected in the user and at the kernel level of the operating system, dumping malware executable memory, and precision hook setup at the right time. The primary purpose of this article is to create an effective platform for analyzing behaviour and detecting new Malware through utilizing a metamorphic engine; two tools are taking action for obfuscating and metamorphic itself, which are packer and protector[30]. Their outcomes in recognizing kernel-level Malware were 85%, which was reasonable.

Authors in[31] provide a designated framework that aims at reliable and secured detection of newly designed and unknown instances of Malware on virtual machines inside an organization's private cloud. The MinHash approach was used to examine all of the information in the memory dumps since it is ideal for efficiently comparing volatile memory dumps, which is essential for accurately identifying Malware in VMs. A comprehensive set of experiments of increasing difficulty is used to evaluate the proposed framework. During these experiments, the researchers also measured the detection performance of various classifiers based on similarity and machine learning. These classifiers were tested on real-world, professional, legendary malware collections and lawful applications[32]. The experimental result revealed that the framework could identify the abnormal state of a virtual server and known, new, and unidentified Malware with extremely high TPRs (100% for Ransomware and RATs) and very low FPRs (1.8% for Ransomware and no FPR for RATs).

In[33], they employed two image descriptors, GIST and HOG, to undertake two-phase research to identify and detect malicious and benign executables. It was done by analyzing the memory dump images. In the second phase, researchers utilized UMAP, a state-of-the-art manifold learning technique, to enhance classifier robustness and better determine if a suspicious process in a computer's memory is malicious. Using the SMO algorithm on the feature vectors in conjunction with the GIST and HOG algorithms, they reached a prediction accuracy of up to 96.39%, as shown by the results[34]. The UMAP-based manifold learning technique enhanced unidentified malware identification model accuracy by 12.93%, 21.83%, and 20.78% for Random Forest, linear SVM, and XGBoost algorithms, respectively.

Nowadays, the spread of Malware is a significant problem. The rapid development of the internet and the expansion of software and systems have filled the information exchange gap in daily life. Storages such as memory are a primary target for spreading Malware through it. Identifying and detecting Malware takes a lot of time and energy. Therefore, much research has been conducted to detect Malware in different datasets. Some of their detection rate are high and reasonable.

Table 1 represents related works on the other two types of studies. The first one is on malware datasets with the algorithms used to detect the Malware. The second one is on different datasets that used our algorithm (Lazy IBK) and their results.

Table 1: List some Related Works.

Reference	Dataset	Classifier / Technique	Accuracy
[18]	Microsoft Malware Classification Challenge (BIG 2015) dataset	Long Short-Term Memory (LSTM) network	95.71%
[19]	online memory data	Logistic Regression, SVM, and Random Forest	99.0%
[20]	Win32-type malware samples	CNN-Model	90.5%
[21]	CIC IDS-2017(Intrusion Detection Dataset)	MLP, IBK	99.67%, 99.87%
[22]	Facebook Data	Naïve Bayes, Kstar, LWL, DT, J48 and Instance-Based Learning (IBK)	98.21% (IBK)
[23]	Breast Cancer Datasets	Naïve Bayes, Logistic, Instance Based Learning (IBK), Random Forest, J48	98.2% (IBK)
[25]	Diabetes Dataset	Instance-Based Learning (IBK), Kstr, LWL	99%
[27]	Kaggle Microsoft Malware Classification Challenge	Deep Generative Model	97.47%
[29]	Obfuscated and Metamorphic Malware	Dynamic Obfuscation (anti-analysis) and Static Obfuscation Techniques	85%
[31]	Malware Dataset	MinHash method	86%
[33]	Dumpware10 Dataset	SMO algorithm on the feature vectors combined with GIST+HOG	96.39%

3.Methodology

This section demonstrates this article's approaches, techniques, and datasets. The flow diagram of our method is depicted in Figure 1, and the following subsections expound each step of the research done in this article in detail.

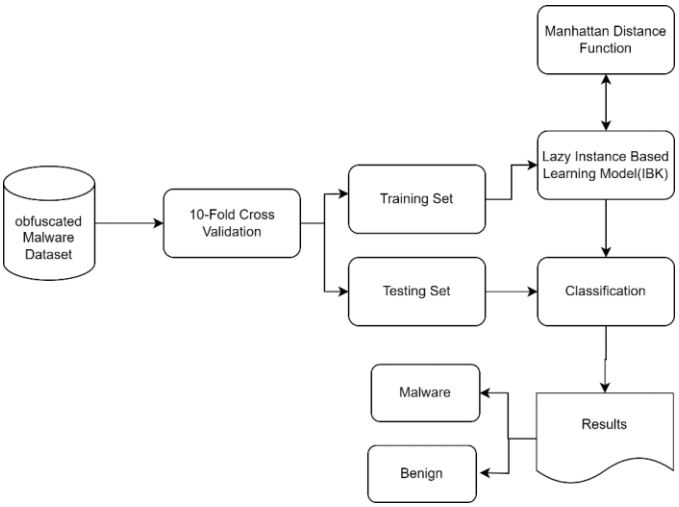

Figure 1: Flowchart Diagram of The Obfuscated Malware Detection Model.

3.1. Obfuscated Malware Dataset

In this paper, we have used the obfuscated malware dataset. The balanced dataset is built to select malicious software detection mechanisms through memory. The goal behind the design of this data set is to obtain the closest possible sample of known Malware in the real world. Which in total consists of 58,596 records, equally divided into two groups: malicious and benign dumps. Meaning each group consists of 29,298 records. Concerning the mechanism of creating memory dumps, different applications were used in the device with a memory of 2 GB, depending on the usual behaviour of the users; as for the mechanism adopted to capture malicious files, VirusTotal, which contains different categories of Malware. As a result, 2961 malware samples were collected for those programs, including Spyware, Trojan horses, and Ransomware, as detailed in Tables II and III[35]. Figure 2 shows the overall malware families used in the dataset.

Table 2: Description of Obfuscated Malware Dataset Features.

Feature Type	Feature List	Feature Descriptions
Malfind	commit charge	The overall of Commit Charges
	protection	The overall protections
	uniqueInjections	The overall unique injections
Ldrmodule	avgMissingFromLoad	The average amount of modules missing from the load list
	avgMissingFromInit	The average amount of modules missing from the initialization list
	avgMissingFromMem	The average amount of modules missing from memory
Handles	port	The overall port handles
	file	The overall file handles
	event	The overall event handles
	desktop	The overall desktop handles
	key	The overall key handles
	thread	The overall thread handles
	directory	The overall directory handles
	semaphore	The overall semaphore handles
	timer	The overall timer handles
	section	The overall section handles
	mutant	The overall mutant handles
Process View	pslist	Average false ratio of the process list
	psscan	Average false ratio of the process scan
	thrdproc	Average false ratio of the third process
	pspcid	Average false ratio of the process id
	session	Average false ratio of the session
	deskthrd	Average false ratio of the deskthrd
Apihooks	nhooks	The overall apihooks
	nhookInLine	The overall line apihooks
	nhooksInUsermode	The overall of apihooks in user mode

Table 3: Description of Malware Sample Count.

Malware Group	Malware Families	Count
Trojan Horse Group	Zeus	195
	Emotet	196
	Refroso	200
	scar	200
	Reconyc	157
Spyware Group	180Solutions	200
	Coolwebsearch	200
	Gator	200
	Transponder	241
	TIBS	141
Ransomware Group	Conti	200
	MAZE	195
	Pysa	171
	Ako	200
	Shade	220

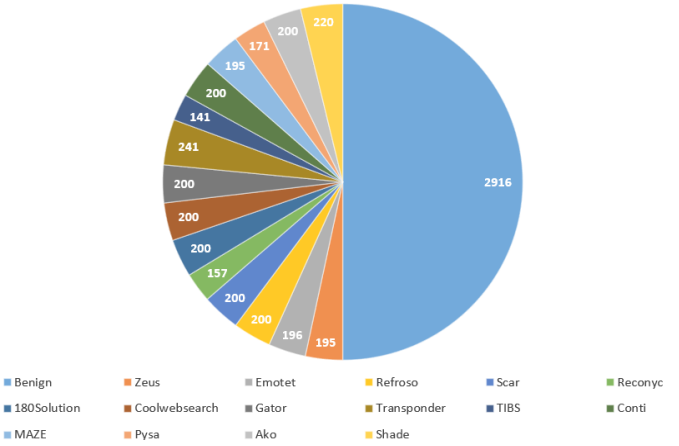

Figure 2: Complete Dataset Breakdown.

3.2. Preprocessing Data

In preprocessing and to balance the dataset, oversampling with the SMOTE technique was performed. In contrast to other oversampling methods, SMOTE creates synthetic values that are hardly distinguishable from the actual values rather than producing duplicates[36].

3.3. Lazy IBK Algorithm

Instance-based classifier (IBK), or K-nearest-neighbor, is an efficient and straightforward technique that falls into the category of lazy learning. This method relies on the training data to make predictions without needing model learning. As distance is the primary determinant of similarity between data points, k-NN employs a majority vote between the new (unseen) instance and the k most identical examples[37]. In addition, the parameters of the IBK classifier were employed without modifying the default parameters like:

- The number of neighbours to use (KNN) = 1
- batchSize = 100 The preferred number of instances to process if the batch prediction is performed.
- Distance weighting = 2 Gets the distance weighting method used.
- window-size = 0 Gets the maximum number of instances allowed in the training pool
- attributeIndices = First-Last Specify a range of attributes to act on. It is a comma-separated list of attribute indices with "first" and "last" valid values. Specify an inclusive range with "-."

IBK commonly employs distance metric functions to measure the distance between points and vectors. Manhattan distance is used to measure the distance between two points by utilizing the absolute value of the difference between them[38]. The following equation is the formula for the Manhattan distance:

$$d(x,y) = \sum_{i=1}^{n} |x_i - y_i| \quad (1)$$

Where:
X: is the first point
Y: is the second point
$\sum_{i=1}^{n} |x_i - y_i|$: is the absolute value of two points

4. Experimental Results And Discussion

This study aims to build a quick and accurate model for malware detection in obfuscated malware memory datasets. The classification process in this paper was done utilizing a machine learning tool developed by Waikato University in New Zealand, which is named Weka (Waikato Environment for Knowledge Analysis). It is free and open-source software with visualization tools and algorithms for analyzing data and predictive modelling[39]. We have implemented the work on HP z Firefly 14 g8, a laptop with Intel(R) Core i7-1165G7 CPU@ 2.80GHz, 16GB RAM, and Windows 10 Professional 64-bit. We have implemented the Lazy IBK-based Manhattan Distance experiment conducted in this work on a massive dataset with several records. The Obfuscated Malware Memory dataset was first split employing a 10-fold cross-validation approach to divide the dataset into ten equal portions to implement and validate the proposed model. Each dataset part is utilized in both the training and testing process. This approach aims to guarantee the randomness of the experiments and avoid any modelling issues with underfitting and overfitting. The performance evaluation of the algorithm is summarized after classification based on various metrics, including Confusion Matrix, TP Rate, FP Rate, Precision, Recall, F-Measure, Classification Error Rate, and Accuracy.

Table 4: Explain the Confusion Matrix, which is Tp = True Positive, Fn = False Negative, Fp = False Positive, Tn = True Negative

		Actual Class	
		Positive	**Negative**
Predicted Class	**Positive**	TP	FP
	Negative	FN	TN

Table 5: Confusion Matrix for Proposed Model.

		Actual Class	
		Benign	**Malware**
Predicted Class	**Benign**	29298	6
	Malware	0	29292

Table 3 and Table 4 represent the confusion matrix of the Lazy IBK classification model. (TP) is the number of benign records in the actual class of the dataset, which was classified as benign records equal to 29298. Meanwhile, (FN) is the number of benign records in the exact type of dataset predicted as malware records, which is zero. Furthermore, (FP) is the number of malware records in the actual class expected as benign records, equal to six. Lastly, (TN) is the number of malware records in the actual class indicated as malware records and is equal to 29292. Table 5 demonstrates the metrics in detail with their equations.

Table 6: Evaluation Metrics for Classification Model.

Metric Names	Metrics Equations	References
TP Rate	TP / (TP+FN)	[40]
FP Rate	FP / (FP+TN)	[40]
Precision	TP/ (TP+ FP)	[41]
Recall	TP/ (TP+ FN)	[41]
F-Measure	(2 * Precision * Recall) / (Precision + Recall)	[42]
Error Rate	FP + FN / TP + TN + FP + FN	[42]
Accuracy	*100% - Error Rate*	[42]

The experimental results in Table 5 show that the Lazy IBK algorithm based on Manhattan Distance obtained a perfect outcome in classifying an Obfuscated Malware dataset classes (benign or Malware) with outstanding accuracy, precision, recall, and F-measure of 99.99%, 1.000, 1.000, 1.000 respectively. In our model, the precision and recall values are equal to 1.000. furthermore, the F-Measure metric depends on precision and recall values as the harmonic mean to be calculated. In our case, 1.000 has been concluded, which can be considered a very high result.

Table 7: Evaluation Metrics for Validation Data.

Class Label	TP Rate	FP Rate	Precision	Recall	F-Measure	Accuracy %
Benign	0.000	1.000	1.000	1.000	1.000	99.99

According to Table VII, the total number of instances was 58596, and the correctly classified obfuscated malware records out of the total obfuscated Malware in the dataset for Lazy IBK was 58590. The Lazy IBK obtained a 0.01% error rate. In addition, the model building and the classification time interval were 0.7 seconds.

Table 8: Time/Values of Performance Metrics of The Algorithm.

Total Number of Instances	Correctly Classified Instances	Incorrectly Classified Instances	Error Rate	Modelling Time (Seconds)
58596	58590	6	0.01	0.7

The lazy IBK-based Manhattan Distance model has been extensively utilized for classification problems in data mining and machine learning. This article granted a machine learning algorithm, Lazy IBK-based Manhattan Distance, to classify an Obfuscated Malware Memory dataset. The Lazy IBK represents flexibility and can handle a vast dataset, as we employed in this study. We have proved that the used classifier can efficiently and accurately identify the Malware in an Obfuscated Malware dataset with experimental and, therefore, obtained superior predictive performance.

5. Comparing the Proposed Model With Relevant Studies

It takes work to compare the obtained results accurately with other developments in related research due to the diversity and difference in the available datasets and the approach used in each study. They are often biased towards a specific direction. Three criteria were adopted to compare the proposed model with four related works: the speed of model construction, the model's complexity, and the model's accuracy.

The first work selected for comparison is by[26], which is much slower than the proposed model with the same level of model complexity as the proposed method with relatively high accuracy. Moreover, the second comparison is with[24], a fast and complex model with medium accuracy compared to the proposed model. Furthermore, the third work was authored by[14] in which the speed of the model was not mentioned, but what is noteworthy in this work is the very high accuracy that is comparable to the accuracy of the proposed work and a similar level to the proposed method in terms of complexity. Ultimately, the fourth and last comparison is with[28], which is medium in both speed and complexity while characterized by relatively high accuracy but still less accurate than the proposed model.

Table 9: Compared with Other Obfuscated Malware Detection Methods.

Models	Speed	Complexity	Accuracy
[31]	Medium	Medium	High
[29]	High	High	Medium
[19]	Not Mentioned	High	Very High
[33]	Medium	Medium	High
Proposed Method	Very High	Medium	Very High

Conclusion and Future Work

In this paper, a rapid and very high accuracy model (compared to similar studies in the same field) was built to detect obfuscated malware memory by applying a Lazy Instance-Based Classifier with Manhattan function distance. I am using the obfuscated malware dataset employing the 10-fold cross-validation to split the used dataset into the training and testing sets, which consist of 58,596 records divided into three categories (Trojan Horse ,‹ Spyware ‹Ransomware) groups. As a result, the classification accuracy obtained to classify the data set used for the method proposed in this paper was 99.99% in a record time of 0.7 seconds. Furthermore, the next step of this paper is to apply the proposed method in classifying the dataset for other malware attacks, such as Fileless Malware, Mobile Malware, etc., as future work.

Conflict of interests

None

Author Distributions

We explored the effectiveness of employing the Instance-Based Learner (IBK) algorithm to identify obfuscated malware within a specific dataset. The utilization of the Lazy IBK technique in malware detection proves advantageous, as the algorithm can precisely detect and classify obfuscated malware in the dataset through the application of the Manhattan Distance function. This function is renowned as one of the most well-established distance metric functions for measuring the distance between points. The demonstrated results affirm that the proposed algorithm can swiftly and accurately detect obfuscated malware.

Funding Information

No funding was received to assist the preparation of the manuscript.

References

1. Statista (2022). Statista: annual number of malware attacks worldwide from 2015 to 2022. https://www.statista.com/statistics/873097/malwareattacks-per-year-worldwide/. (Accessed on 30/9/2022).
2. Alkahtani, H. and Aldhyani, T.H.H. ‘Artificial Intelligence Algorithms for Malware Detection in Android-Operated Mobile Devices’, Sensors, 22(6), p. 2268. Available at: https://doi.org/10.3390/s22062268 (2022).
3. Xin, Y., Kong, L., Liu, Z., Chen, Y., Li, Y., Zhu, H., Gao, M., Hou, H., & Wang, C. Machine learning and deep learning methods for cybersecurity. IEEE Access, 6, 35365–35381. https://doi.org/10.1109/access.2018.2836950 (2018).
4. Sarker, I. H. Machine learning: Algorithms, real-world applications and Research Directions. SN Computer Science, 2(3). https://doi.org/10.1007/s42979-021-00592-x (2021).
5. Walker, S., Khan, W., Katic, K., Maassen, W., & Zeiler, W. Accuracy of different machine learning algorithms and added-value of predicting aggregated-level energy performance of commercial buildings. Energy and Buildings, 209, 109705. https://doi.org/10.1016/j.enbuild.2019.109705 (2020).
6. Shahsavar, A., Moayedi, H., Al-Waeli, A. H., Sopian, K., & Chelvanathan, P. Machine learning predictive models for optimal design of building-integrated photovoltaic-thermal collectors. International Journal of Energy Research, 44(7), 5675–5695. https://doi.org/10.1002/er.5323 (2020).
7. Hassan, B. A., and Rashid, T. A. A multidisciplinary ensemble algorithm for clustering heterogeneous datasets. Neural Computing and Applications, 33(17), 10987–11010. https://doi.org/10.1007/s00521-020-05649-1 (2021).
8. Guron, A., Anwer, M., Sulaiman, S., & AbdulSamad, S. Classification of the cause of eye impairment using different kinds of machine learning algorithms. Passer Journal of Basic and Applied Sciences, 5(2), 410–416. https://doi.org/10.24271/psr.2023.397078.1328 (2023).
9. Hassan, B. A., Rashid, T. A., & Mirjalili, S. Performance evaluation results of evolutionary clustering algorithm star for clustering heterogeneous datasets. Data in Brief, 36, 107044. https://doi.org/10.1016/j.dib.2021.107044 (2021).
10. Bahjat, N., & Jamak, S. A Comparison Study of Data Mining Algorithms for blood Cancer Prediction. Passer Journal of Basic and Applied Sciences, 3(1), 174–179. https://doi.org/10.24271/psr.29 (2019).
11. Guron, A., Anwer, M., Sulaiman, S., & AbdulSamad, S. Classification of the cause of eye impairment using different kinds of machine learning algorithms. Passer Journal of Basic and Applied Sciences, 5(2), 410–416. https://doi.org/10.24271/psr.2023.397078.1328 (2023).
12. Hussein, D., Rashad, M., Mirza, K., & hussein, dana. Machine Learning Approach to Sentiment Analysis in Data Mining. Passer Journal of Basic and Applied Sciences, 4(1), 71–77. https://doi.org/10.24271/psr.2022.312664.1101 (2022).
13. Abdulhadi, H.M.T., Ali, M.H. and Talabani, H.S. ‘The attitude of mobile apps and it impact in Health and life services through pandemic (Covid-19) in Iraq: National Survey.’, Passer Journal of Basic and Applied Sciences, 5(1), 94–102. doi:10.24271/psr.2023.380083.1222 (2023).
14. Shree, R., Kant Shukla, A., Prakash Pandey, R., Shukla, V., & Bajpai, D. Memory forensic: Acquisition and analysis mechanism for operating systems. Materials Today: Proceedings, 51, 254–260. https://doi.org/10.1016/j.matpr.2021.05.270 (2022).
15. Block, F., & Dewald, A. Linux memory forensics: Dissecting the User Space Process Heap. Digital Investigation, 22. https://doi.org/10.1016/j.diin.2017.06.002 (2017).
16. Thantilage, R., & Jeyamohan, N. A volatile memory analysis tool for retrieval of social media evidence in Windows 10 OS based workstations. 2017 National Information Technology Conference (NITC). https://doi.org/10.1109/nitc.2017.8285664 (2017).
17. Martín-Pérez, M., Rodríguez, R. J., & Balzarotti, D. Preprocessing memory dumps to improve similarity score of Windows modules. Computers & Security, 101, 102119. https://doi.org/10.1016/j.cose.2020.102119 (2021).
18. Kang, J., Jang, S., Li, S., Jeong, Y.-S., & Sung, Y. Long short-term memory-based malware classification method for information security. Computers & Electrical Engineering, 77, 366–375. https://doi.org/10.1016/j.compeleceng.2019.06.014 (2019).
19. Xu, Z., Ray, S., Subramanyan, P., & Malik, S. Malware detection using machine learning based analysis of virtual memory access patterns. Design, Automation & Test in Europe Conference & Exhibition (DATE), 2017. https://doi.org/10.23919/date.2017.7926977 (2017).
20. Li, H., Zhan, D., Liu, T., & Ye, L. Using deep-learning-based memory analysis for malware detection in cloud. 2019 IEEE 16th International Conference on Mobile Ad Hoc and Sensor Systems Workshops (MASSW). https://doi.org/10.1109/massw.2019.00008 (2019).
21. Ali, A., Shaukat, S., Tayyab, M., Khan, M. A., Khan, J. S., Arshad, & Ahmad, J. Network Intrusion Detection Leveraging Machine Learning and feature selection. 2020 IEEE 17th International Conference on Smart

Communities: Improving Quality of Life Using ICT, IoT and AI (HONET). https://doi.org/10.1109/honet50430.2020.9322813 (2020).
22. Farin, N. J., Akter, M., Roy, P., & Uddin, M. S. Data mining techniques for predicting user interest in Facebook pages: A Comparison. 2019 1st International Conference on Advances in Science, Engineering and Robotics Technology (ICASERT). https://doi.org/10.1109/icasert.2019.8934618 (2019).
23. Alshammari, M., & Mezher, M. A comparative analysis of data mining techniques on breast cancer diagnosis data using Weka Toolbox. International Journal of Advanced Computer Science and Applications, 11(8). https://doi.org/10.14569/ijacsa.2020.0110829 (2020).
24. Abdulkhaleq, M. T., Rashid, T. A., Alsadoon, A., Hassan, B. A., Mohammadi, M., Abdullah, J. M., Chhabra, A., Ali, S. L., Othman, R. N., Hasan, H. A., Azad, S., Mahmood, N. A., Abdalrahman, S. S., Rasul, H. O., Bacanin, N., & Vimal, S. Harmony search: Current studies and uses on Healthcare Systems. Artificial Intelligence in Medicine, 131, 102348. https://doi.org/10.1016/j.artmed.2022.102348 (2022).
25. Roslan, N. A., Mahdin, H., & Kasim, S. A study on dengue cases detection based on lazy classifier. International Journal of Advanced Science Computing and Engineering, 1(1), 43–47. https://doi.org/10.30630/ijasce.1.1.10 (2019).
26. Hassan, B. A. CSCF: A chaotic sine cosine firefly algorithm for practical application problems. Neural Computing and Applications, 33(12), 7011–7030. https://doi.org/10.1007/s00521-020-05474-6 (2020).
27. Kim, J.-Y., & Cho, S.-B. Obfuscated malware detection using deep generative model based on Global/local features. Computers & Security, 112, 102501. https://doi.org/10.1016/j.cose.2021.102501 (2022).
28. Hassan, B. A., Rashid, T. A., & Hamarashid, H. K. A novel cluster detection of COVID-19 patients and medical disease conditions using improved evolutionary clustering algorithm star. Computers in Biology and Medicine, 138, 104866. https://doi.org/10.1016/j.compbiomed.2021.104866 (2021).
29. Javaheri, D., & Hosseinzadeh, M. A framework for recognition and confronting of obfuscated malwares based on memory dumping and filter drivers. Wireless Personal Communications, 98(1), 119–137. https://doi.org/10.1007/s11277-017-4859-y (2017).
30. Hassan, B. A., Rashid, T. A., & Mirjalili, S. Formal context reduction in deriving concept hierarchies from corpora using adaptive evolutionary clustering algorithm star. Complex & Intelligent Systems, 7(5), 2383–2398. https://doi.org/10.1007/s40747-021-00422-w (2021).
31. Nissim, N., Lahav, O., Cohen, A., Elovici, Y., & Rokach, L. Volatile memory analysis using the MINHASH method for efficient and secured detection of Malware in the private cloud. Computers & Security, 87, 101590. https://doi.org/10.1016/j.cose.2019.101590 (2019).
32. Maaroof, B. B., Rashid, T. A., Abdulla, J. M., Hassan, B. A., Alsadoon, A., Mohammadi, M., Khishe, M., & Mirjalili, S. Current studies and applications of shuffled Frog Leaping Algorithm: A Review. Archives of Computational Methods in Engineering, 29(5), 3459–3474.https://doi.org/10.1007/s11831-021-09707-2 (2022).
33. Bozkir, A. S., Tahillioglu, E., Aydos, M., & Kara, I. (2021). Catch them alive: A malware detection approach through memory forensics, Manifold Learning and Computer Vision. Computers & Security, 103, 102166. https://doi.org/10.1016/j.cose.2020.102166 (2021).
34. Qader, S. M., Hassan, B. A., & Rashid, T. A. An improved deep convolutional neural network by using hybrid optimization algorithms to detect and classify brain tumor using augmented MRI images. Multimedia Tools and Applications, 81(30), 44059–44086. https://doi.org/10.1007/s11042-022-13260-w (2022).
35. Search UNB (2022). University of New Brunswick est.1785. Available at: https://www.unb.ca/cic/datasets/malmem-2022.html#:~:text=CIC%2DMalMem%2D2022&text=The%20obfuscated%20malware%20dataset%20is,prevalent%20in%20the%20real%20world. (Accessed: October 1, 2022).
36. Abdullah ALFRHAN, A., Hamad ALHUSAIN, R., & Ulah Khan, R. SMOTE: Class Imbalance Problem In Intrusion Detection System. 2020 International Conference on Computing and Information Technology (ICCIT-1441), 1–5. https://doi.org/10.1109/ICCIT-144147971.2020.9213728 (2020).
37. Kunal, & Dua, M. Attribute selection and ensemble classifier based novel approach to Intrusion Detection System. Procedia Computer Science, 167, 2191–2199. https://doi.org/10.1016/j.procs.2020.03.271 (2020).
38. Gao, X., & Li, G. A KNN model based on Manhattan distance to identify the SNARE proteins. IEEE Access, 8, 112922–112931. https://doi.org/10.1109/access.2020.3003086 (2020).
39. Downloading and installing Weka. Downloading and installing Weka - Weka Wiki. (n.d.-a). https://waikato.github.io/weka-wiki/downloading_weka. (Accessed:July 16,2023)
40. Talabani, H. S., & Abdulhadi, H. M. T. A. R. E. Q. Bitcoin ransomware detection employing rule-based algorithms. Science Journal of University of Zakho, 10(1), 5–10. https://doi.org/10.25271/sjuoz.2022.10.1.865 (2022).
41. Reddy, G. T., Reddy, M. P., Lakshmanna, K., Kaluri, R., Rajput, D. S., Srivastava, G., & Baker, T. Analysis of dimensionality reduction techniques on Big Data. IEEE Access, 8, 54776–54788. https://doi.org/10.1109/access.2020.2980942 (2020).
42. Abdulhadi, H. M. T., & Talabani, H. S. Comparative study of supervised machine learning algorithms on thoracic surgery patients based on ranker feature algorithms. UHD Journal of Science and Technology, 5(2), 66–74. https://doi.org/10.21928/uhdjst.v5n2y2021.pp66-74 (2021).