ISSN: 0067-2904

One-Class and Multi-Class Malware Classification Using Hybrid and Supervised Machine Learning Techniques

Mohammed Saadoon, Suhad Faisal Behadili*
Computer Department, College of Science, University of Baghdad, Baghdad, Iraq.

Received:3/7/2024 Accepted: 21/2 /2025 Published: 28/2/2026

Abstract

Cybercriminals or hackers design malware programs with malicious intent to steal, spy, and destroy victim's computers. Malware encompasses various forms, such as viruses, trojans, ransomware, spyware, and adware, each requiring effective classification for accurate identification and mitigation. High-quality datasets are crucial for training classification models, and the CIC-MalMem-2022 dataset, containing 58,596 records and 55 numerical features, is an essential resource in this regard. This study used artificial intelligence techniques and algorithms such as KNN, Decision Tree, Random Forest, SVM, and Naïve Bays, as well as supervised and hybrid machine learning by integrating Random Forest and K-Nearest Neighbors KNN to improve classification performance with one-class classification and multi-class. In terms of accuracy score, the best results achieved by the proposed methods were random forest with 99.98%, hybrid random forest + K-Nearest Neighbors with 99.93%, decision tree with 99.95%, k-Nearest Neighbors with 99.87%, support vector machine with 99.80%, and naïve bays with 98.90%. These metrics (accuracy, precision, and recall) reflect the models' effectiveness in classifying instances. Accuracy measures overall correctness, precision evaluates the quality of positive predictions, and recall assesses the ability to identify true positives. The consistently high scores demonstrate the reliability and robustness of these methods for malware classification.

Keywords: Malware, Machine learning, Data preprocessing, Feature extraction.

الكشف عن البرمجيات الخبيثة باستخدام تقنيات التعليم الالي الموجهة والهجينة لصنف واحد او عدة أصناف

محمد سعدون عبدالزهرة, سهاد فيصل شيحان*
قسم الحاسوب، كلية العلوم، جامعة بغداد، بغداد، العراق

الخلاصة

يقوم مجرمو الإنترنت أو الهاكرز بتصميم البرمجيات الخبيثة بنية لسرقة المعلومات أو التجسس أو تدمير أجهزة الكمبيوتر الخاصة بالضحية. تشمل البرمجيات الخبيثة أشكالًا متعددة مثل الفيروسات، والتروجانات، وبرمجيات الفدية، والبرمجيات التجسسية، وبرمجيات الإعلانات، وكل منها يتطلب تصنيفًا فعالًا للتحديد الدقيق والتقليل من الأضرار. تعتبر مجموعات البيانات عالية الجودة ضرورية لتدريب نماذج التصنيف، وتعد

*Email: suhad.f@sc.uobaghdad.edu.iq

مجموعة بيانات CIC-MalMem-2022، التي تحتوي على 58,596 سجلًا و55 ميزة عددية، مصدرًا أساسيًا في هذا الصدد. استخدمت هذه الدراسة تقنيات الذكاء الاصطناعي والخوارزميات مثل KNN، Decision Tree، وRandom forest، SVM، Naïve Bays ، بالإضافة إلى التعلم الآلي الموجه والهجين من خلال دمج Random forest مع K-Nearest Neighbors (KNN) لتحسين أداء التصنيف باستخدام التصنيف أحادي الفئة ومتعدد الفئات. فيما يتعلق بدرجة الدقة، كانت أفضل النتائج التي تم تحقيقها باستخدام الطرق المقترحة هي: Random forest بنسبة 99.98%، ودمج الـ Random forest مع K-Nearest Neighbors (KNN) بنسبة 99.93%، Decision tree بنسبة 99.95%، و K-Nearest Neighbors بنسبة 99.87%، SVM بنسبة 99.80%، Naïve Bays بنسبة 98.90%. تعكس هذه المقاييس الدقة في التصنيف. ان تصنيف الدقة المودل، يتطلب تقيم جودة التنبؤات الإيجابية، ويقوم F-Score بتقييم قدرة النموذج على تحديد الحالات الصحيحة. تظهر الدرجات العالية المتسقة موثوقية وقوة هذه الأساليب في تصنيف البرمجيات الخبيثة.

1. Introduction

These days, cyberattacks pose a significant threat to all aspects of life, including finances, health, privacy, and more. Hence, malware is the most common attack that has been used to compromise computer systems, and it is also known as any program runs on a computer system without the victim's knowledge [1]. Malware can be classified by type, so understanding its types is crucial for its analysis and detection [2]. The list below includes the most common types of malware, as shown in Figure 1[3]:

1. Virus: A virus is a fundamental type of malware designed to infect a computer by attaching itself to legitimate programs or files. It spreads when the infected file or program is executed, often after being downloaded or transferred. The defining characteristic of a virus is its ability to replicate itself by injecting its code into other programs or system files, modifying them to carry out malicious activities. These activities can include corrupting data, disrupting system operations, or enabling unauthorized access [4].
2. Worm: This type of malware replicates itself without any action for the infected user, consuming a large amount of memory and causing damage to the computer system. It also steals information [5].
3. Trojan: it poses as genuine software to conceal its malicious goal, which is to steal information and get access to the computer system's hidden layers [6].
4. Spyware: This kind of malware collects data from compromised systems, including passwords, sensitive personal information, and websites visited by the compromised user, and transfers it to a third party without the compromised user's awareness [7].
5. Ransomware: This type of malware is among the most prevalent and harmful types available today. Its approach encrypts all data on the compromised system and sends an interface order to the compromised individual, who must then pay fees to obtain a key to decrypt his data, hence why it is difficult to detect and stop due to its sophisticated methods. It can hide from security systems, often appearing as legitimate software, and only reveals its presence after encrypting data [8].

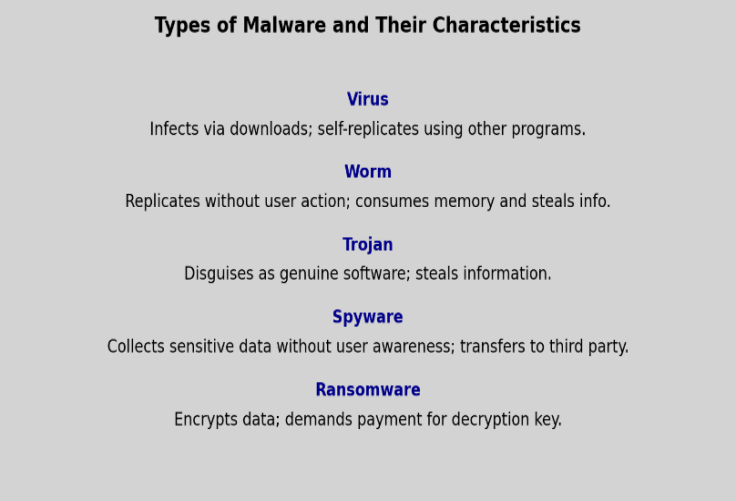

Figure 1: The malware types [3]

Malware analysis identifies distinct characteristics and origins of malicious code, enabling classification into specific malware types such as spyware, ransomware, and others. Also, it is considered an important stage of malware detection, and there are several types of analysis, like static, dynamic, and hybrid [9]. Static analysis, a method of examining harmful executable code without running it, is considered a secure malware examination technique. The Python programming language provides a portable executable PE file library to facilitate the feature extraction of dangerous programs. Additionally, static analysis provides a hint to the reverse engineer for a specific program objective [10]. To be more effective and beneficial, a corporation needs dynamic analysis because static analysis is insufficient for detecting some types of malware and cannot identify zero-day malware [11]. Analyzing dynamically uses a safe environment to execute and examine malware to closely monitor the behavior of malicious content and create a report that reflects the level of risk associated with it. Different techniques are also used, such as system API calls, to monitor various operations and identify the types of malware behavior based on various attributes, such as registry modifications and network calls [12].

Data preprocessing is an essential step in preparing datasets for machine learning models. It involves identifying and correcting (or removing) erroneous or corrupt records, ensuring the data is clean and reliable for training and testing. This process improves model performance and reduces the risk of inaccuracies during malware detection and classification [13]. After data preprocessing and feature extraction, the dataset is split for training and testing for deployment. Machine learning uses a variety of algorithms that differ from one another in terms of accuracy, time, and performance [14].

2. The Related work

Several researchers have presented strategies, including datasets, feature extraction, data preprocessing, and machine learning techniques, to combat malware . The following paragraphs provide a discussion of some of these works.

The study [1], which examined various machine learning methods with datasets from the Canadian Institute for Cybersecurity, then compared these methods to determine which malware detection methods produced the best results, based on TPR and accuracy, and which ones yielded the lowest accuracy from FPR, using a confusion matrix. The employed classifiers were KNN, CNN, NB, RF, SVM, and DT. This research shows that the DT

classifier achieves the best accuracy (99%), CNN (98%), and SVM (96%) in the provided dataset.

Also, an overview of dynamic malware analysis in the contemporary era was done in [2]. This study focused on volatile memory forensics and side-channel analysis and compared these techniques with the initial research on function call analysis, execution control, and flow tracking methods used for malware behaviors that have evaded detection and malicious activity. Analysis techniques use hardware or software to gather information about malware behavior from side-channel signals like electromagnetic emission and power usage. Volatile memory acquisition refers to the process of moving a copy of the RAM to external storage and performing analysis on it. The software employs a bare metal layout, much like hardware does for a memory dump; in this paper survey, they apply machine learning to improve the efficiency and precision of these analytic methodologies. Then, in [15], employing both static and dynamic analysis using various machine learning models, found that static analysis was more accurate than dynamic analysis. Consequently, 99.36% of static data is accurate, whereas 94.64% of dynamic data is. Dynamic analysis takes advantage of the Cukoo sandbox; however, some malware exhibits cunning behavior that makes it challenging for dynamic analysis to identify. Additionally, it can be challenging to analyze restrictions resulting from controlled network behavior thoroughly.

Moreover in [16], The system employed three distinct methods for malware identification. The methods include classical signature-based, machine learning-based SVM, and deep learning-based CNN image processing. With an accuracy of 96.59%, CNN thus obtains the best result, followed by SVM with 95% and the traditional signature-based approach with 94%. Note that the traditional signature-based approach cannot detect new malware due to its inability to identify signature changes. However, other techniques can identify these new threats. Whereas [17] used a variety of machine learning algorithms in this assignment, both supervised and unsupervised, and compared them to see which produced the best accuracy results. Among the other strategies, random forest yielded the highest accuracy, followed by XGBoost, decision trees, gradient bossing, and Adboost. This project various obstacles, including an imbalanced dataset, multiple attributes, the need to train the dataset for optimal accuracy, and the challenge of acquiring the dataset through data mining. Therefore, to address these challenges, the following must be determined:

- Use a variety of data pretreatment approaches, such as manipulating the dataset and experimental data analysis.
- There are numerous methods for modifying the models and altering the data set.

Also, [18] has used two classifier models, Interactive Dichotomizer 3 (ID3) and Naïve Bayesian. It used data mining techniques based on anomaly and misuse detection to detect malicious codes using a hybrid intrusion detection system. The system's effectiveness was evaluated based on its accuracy rate. Signature-based detection, while weak against zero-day attacks, is considered more accurate than manually signing signatures. However, anomaly detection can spot fresh intrusion attempts or zero-day attacks, which are not yet known attacks.

This study employed three feature selection metrics: gain ratio, relief measures, and association rules. The NB classifier model employs association rules to obtain the most accurate result with a 1% false positive rate (FN rate) and a 99% false positive rate (FPR). Additionally, in [19], they suggest a network intrusion detection system (NIDS) that uses machine learning algorithms and data mining approaches to help with prediction-making. It utilized Naivy Bayes (NB) and multinominal logistic regression by investigating the KDDcup99 dataset. The results indicate that these two methods yield highly accurate predictions. By avoiding overfitting through cross-validation approaches, the false alarm rate

drops, and the NB classifier takes less time than multinomial logistic regression.
[20] used a deep neural network based on binary classification and achieved a 94% accuracy rate; thereafter, it uses a neural network for classification using several features (DOS, Normal, Probe, R2L, and U2R); as a result, they achieve good performance overall, especially in novelty or outlier detection. Recursive features are based on deep learning and neural network algorithms. So, the study recommends the use of a different dataset, adding a selection feature, and implementing a real-time system for future research.
However, in [21], the datasets was separated into malware and benign samples, and they contain a variety of malware families. those datasets were deployed using DT, RF, and SVM models and then the results were compared . DT achieved the highest accuracy (97.19%), followed by RF and SVM at 96.84% and 93.98%, respectively. Then, in [22] The study goal was to find the best way to classify and detect malware. To do this, a dataset of 1156 malware files from 9 different families and 984 benign files was used to test different machine learning models that use multi-class and binary classification. The highest accuracy was achieved by the j48 and random forest at 93.3% and 95.69%, respectively, and the lowest accuracy was achieved with Naïve Bayes at 72.34% and 55%, respectively, followed by k-Nearest Neighbors and SVM at 87%, 94%, and 87.6%, respectively. Furthermore, [23] focused on feature selection using embedded-based methods to minimize the complexity of machine learning models; hence, it achieved 99.47% and 99.02% from random forest and XGBoost, respectively.

3. Materials and Methodology

This section details the datasets utilized, including the data preprocessing and feature engineering techniques, followed by the machine learning models employed to achieve optimal results.

3.1 Datasets Description

This paper used the CIC-MalMem-2022 dataset from the Canadian Institute for Cybersecurity. This dataset has been examined to test memory-based obfuscated malware detection techniques. It has been classified as balanced, containing 50% benign and 50% malicious records from the memory dump. The collection has 29,298 benign and 29,298 malicious records, for a total of 58,596 records. Table 1 below illustrates the breakdown of malware families.

Table 1: Malware Families Breakdown

Malware category	Malware families	Count
Trojan Horse	• Zeus • Emotet • Refroso • Scar • Reconyc	• 195 • 196 • 200 • 200 • 157
Spyware	• 180Solutions • Coolwebsearch • Gator • Transponder • TIBS	• 200 • 200 • 200 • 241 • 141
Ransomware	• Conti • MAZE • Pysa • Ako • Shade • Shade	• 200 • 195 • 171 • 200 • 220 • 22

3.2 The Proposed Methodology

This paper applied several machine learning methods to the dataset to achieve the highest malware prediction accuracy. In the beginning, some approaches were used for data preprocessing, feature extraction, and selection to get the most relevant features that are helpful for high-quality prediction, followed by the next stage of prediction using several machine learning model techniques, so each model gives a different result. Then, another technique, called hybrid, that uses two different ML models was employed to get the best accuracy prediction.

3.2.1 Data preprocessing

It's a very important step helping machine learning algorithms make accurate predictions and avoid overfitting. It's also a critical step in data analysis because it helps clean the data and remove null values. It also converts rows into appropriate formats for machine learning analysis and prediction. Many techniques exist for data preprocessing, with the following being the most commonly used:

1. Data handling eliminates unnecessary rows and columns and imputes missing values.
2. Handling categorical data involves converting categorical variables into numerical ones, such as label encoding, which machine learning techniques can then handle.
3. Feature scaling: standardize numerical features to bring them to a similar scale.
4. Handling outliers: identify and handle outliers using techniques such as trimming or transforming the data.
5. Feature engineering: make new features from existing ones or transform them to represent the underlying patterns in the data better.
6. Handling imbalance classes: If dealing with classification tasks, address class imbalance using techniques like oversampling, under-sampling, or using specialized algorithms.

Thereafter, the data is used to evaluate the proposed model. Hence, divide the dataset into training and testing sets. Then, select and use the above approaches based on the type of dataset and feature analysis to get the best results.

3.2.2 Feature engineering

In feature engineering, the process begins by examining each feature based on its unique value count. Features with a unique value count of 1 are identified as candidates for removal. The dataset initially contained 57 features, starting with the category and pslist.nproc columns, and ending with the class column. Among these, specific columns—handles.nport, pslist.nprocs64bit, and svcscan.interactive_process_services—in positions 4, 10, and 51, respectively, were nominated for removal. Therefore, we dropped the three nominated features from the dataset. So, the total remaining features are just 54. Next, as shown in Table 2, the class feature value analysis categorizes string values as benign or malware. Therefore, encode the string value into a numerical value. As shown in Table 3, the benign class takes value 0, while the malware class takes the value 1.

Table 2: Distribution of Class Labels Before Encoding

Class	
Benign	29298
Malware	29298

Table 3: Distribution of Class Labels After Encoding

Class	
0	29298
1	29298

The same thing already done about class features has again been executed, but the differences are divided into categories cat1 and cat2 as shown in tables 4 and 5. Therefore, Table 4 presents the benign features with the main categories of malware types, such as spyware, ransomware, and trojans, along with a count for each. Additionally, Table 5 contains the subcategory for the combination of benign and malware. For example, the Spyware-Gator-0b25829d15dc951a44e7652fc6de9d936d7d51f29586d56dbf8fccea419252ac-3.raw file appears in Cat1: Spyware and Cat2: Gator. Next, it calculates the correlation matrix for all features in the data frame and visualizes this matrix using a large heatmap. This provides a comprehensive view of the relationships between all features in the dataset, helping to identify which features are highly correlated with each other. Such visualization is useful in the early stages of data analysis for understanding the data structure and guiding feature selection and preprocessing steps, as shown in Figure 2.

Table 4: The cat1 features values

Cat1	Count
Benign	29298
Spyware	10020
Ransomware	9791
Trojan	9487

Table 5: The cat2 features values

Cat2	Count
Benign	29298
Transponder	2410
Gator	2200
Shade	2128
Ako	2000
180solutions	2000
CWS	2000
Refroso	2000
Scar	2000
Conti	1988
Emotet	1967
Maze	1958
Zeus	1950
Pysa	1717
Reconyc	1570
TIBS	1410

Choosing and extracting the optimal feature by analyzing and visualizing the correlation between various features in a dataset and a target variable named 'Class' converts the

correlation value to a positive percentage and stores these values in a list named correlations. This list can later be used for analysis, such as identifying which features have the strongest relationship with the 'Class' feature. Then, a visual representation of the correlation values between features and the 'Class' feature is created. It sorts these correlation values, plots them, and highlights the mean and 4th quantile (75%) values with horizontal lines. This helps in identifying which features have the highest correlations with the 'Class' feature and how they compare to the average and upper quartile of all correlations.

Based on the 4th quantile value, 74 is a good candidate for a correlation threshold to identify which features to keep, as shown in Figure 3. . Next, create a new DataFrame (hc_data) that includes only the features from the original dataset that correlate highly with the 'Class' feature. This new DataFrame can be used for further analysis or modeling, focusing only on the most relevant features. It then visualizes this correlation matrix using a heatmap, which provides a clear and concise way to see the relationships between these features. The heatmap includes annotations to display the correlation values, making it easier to interpret the strength and direction of the relationships between features. This visualization can help in understanding how the highly correlated features are related to each other, which can be useful for feature selection and further analysis, as shown in Figure 4. that extracts the total features from 54 to just 14 features with the target labels Class and Cat1 and the difference between these two target labels is the Class that could be considered as one-class classification, not multi-class. However, cat1 could be considered a multi-class classification, as shown in Figure 4. So, the two target labels have been determined. First, select the “Class” as the target label and divide the dataset into a set called X set for independent variables. In contrast, Y set for target label features, maintaining a test ratio of 0.3 and a random state of 42. Second, the same procedure was repeated, but cat1 was selected instead of Class as the target label. Finally, the classifiers (Random Forest, Decision tree, K-NN, SVM, NB, Hybrid K-NN + Random Forest) were chosen for training and prediction to estimate an efficient model and method for the effective classification of malware from the investigated dataset.

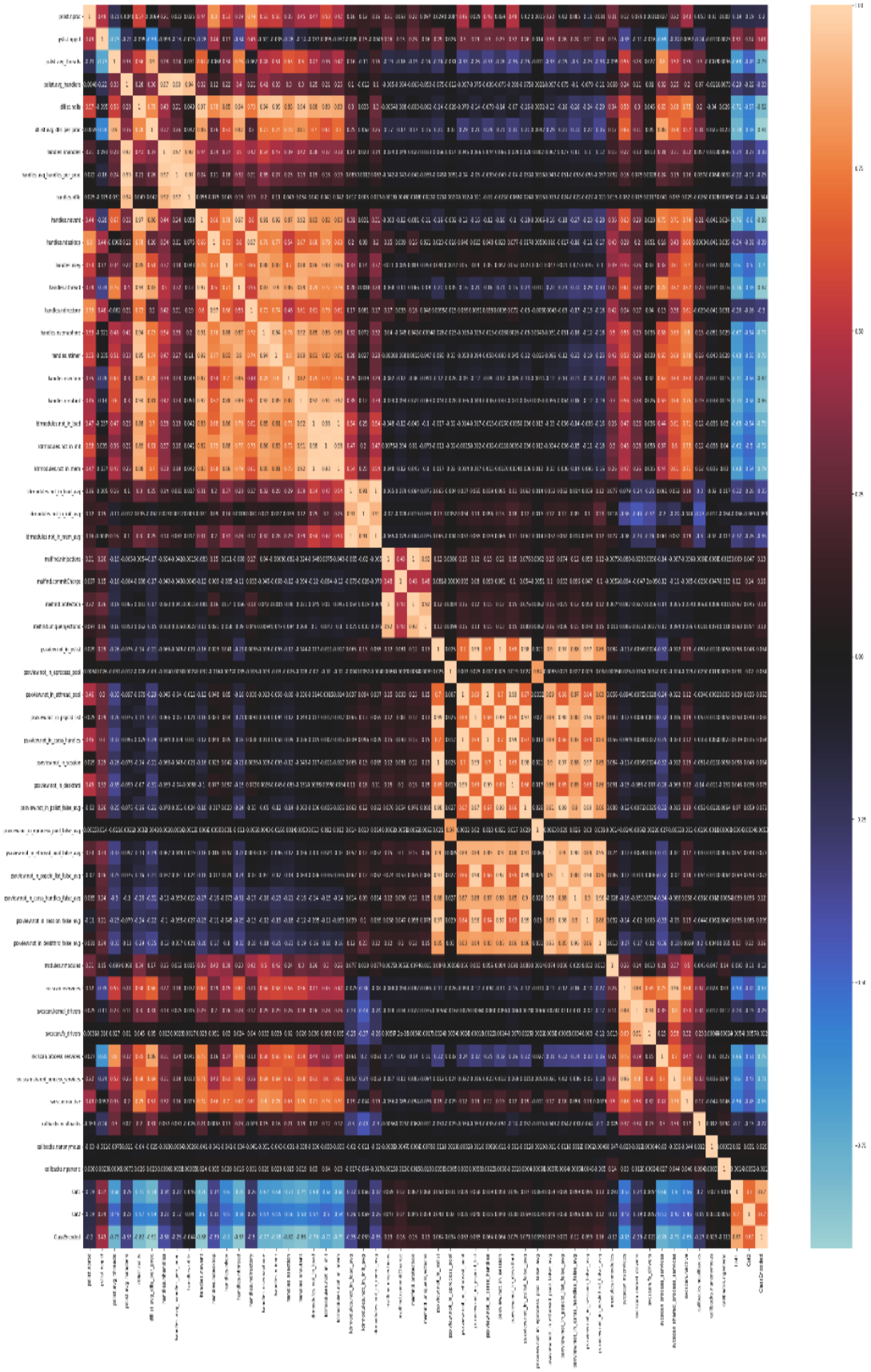

Figure 2: Heatmap of Feature Correlations in the Dataset

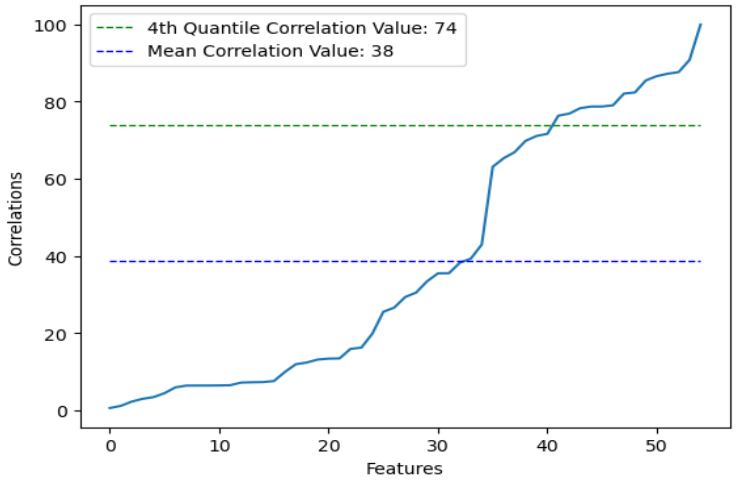

Figure 3: Correlation of Features with Class: Mean and 4th Quantile Thresholds

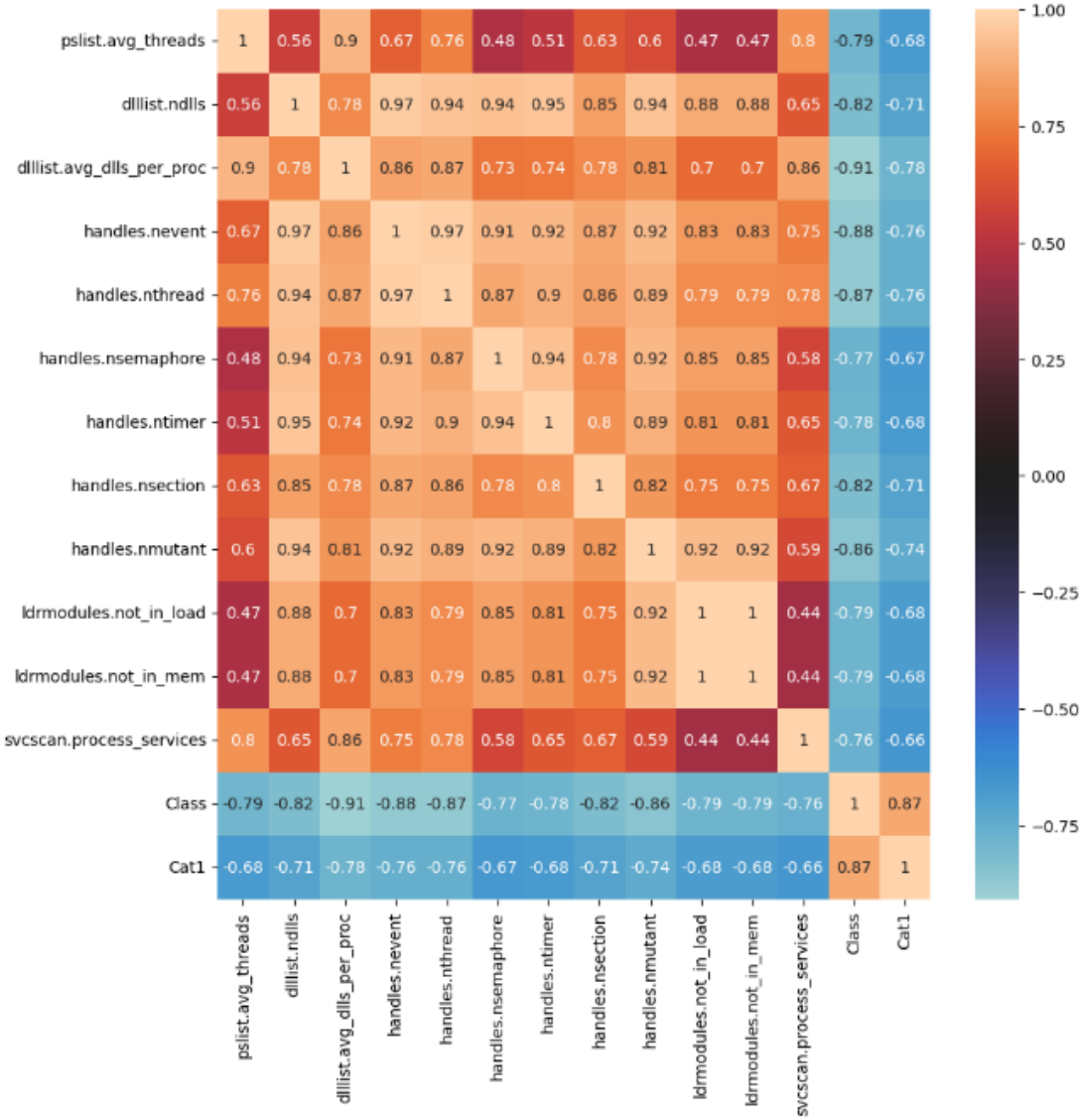

Figure 4: Heatmap of Correlations for Selected Features and Class

4. Results and discussion

This section explores the outcomes of the applied machine learning algorithms, which analyzed and extracted data from the examined dataset to achieve optimal results. After data pre-processing, the dataset is to be extracted into 14 features by employing several classifiers such as Naïve Bays, decision tree, random forest, K-nearest neighbor, Hybrid Random Forest + K-nearest neighbors, and support vector machine with two target labels (Class and cat1) to achieve suitable accuracy, as shown in Table 6, and produce the resulting confusion matrix as is shown in Figures 5 and 6 respectively.

Table 6: The result of accuracy and classification

Model	Accuracy for one-class	Accuracy for multi-class
Random Forest	99.98%	99.67%
Decision Tree	99.95%	99.99%
Hybrid Random Forest + K-Nearest Neighbors	99.93%	66.28%
K-Nearest Neighbors	99.87%	85.95%
Support Vector Machine	99.80%	75.85%
Naïve Bays	98.90%	70.75%

Hence, in one-class classifications, the random forest gives the highest accuracy with 99.98%, followed by the sequence Decision Tree with 99.95%, Hybrid Random Forest + K-Nearest Neighbors with 99.93%, K-nearest with 99.87%, support vector machine with 99.80%, and the least ratio of the Naïve Bays with 98.90%. These results explore the random forest, which is known as flexible for modeling the non-linear relationships among parameters. Therefore, it achieves the highest accuracy by its effectiveness and capability of handling the complex nature of the examined dataset in this study. Moreover, decision trees tend to have lower complexity in one-class than the random forests method. However, accuracy has decreased when dealing with high-dimensional data. Meanwhile, K-Nearest Neighbors (KNN) could be considered suitable for smaller datasets, but it also might suffer from scalability issues. Finally, the Support Vector Machine (SVM) and Naïve Bays handle non-linear relationships effectively, although they require careful hyperparameter tuning and powerful computational resources.

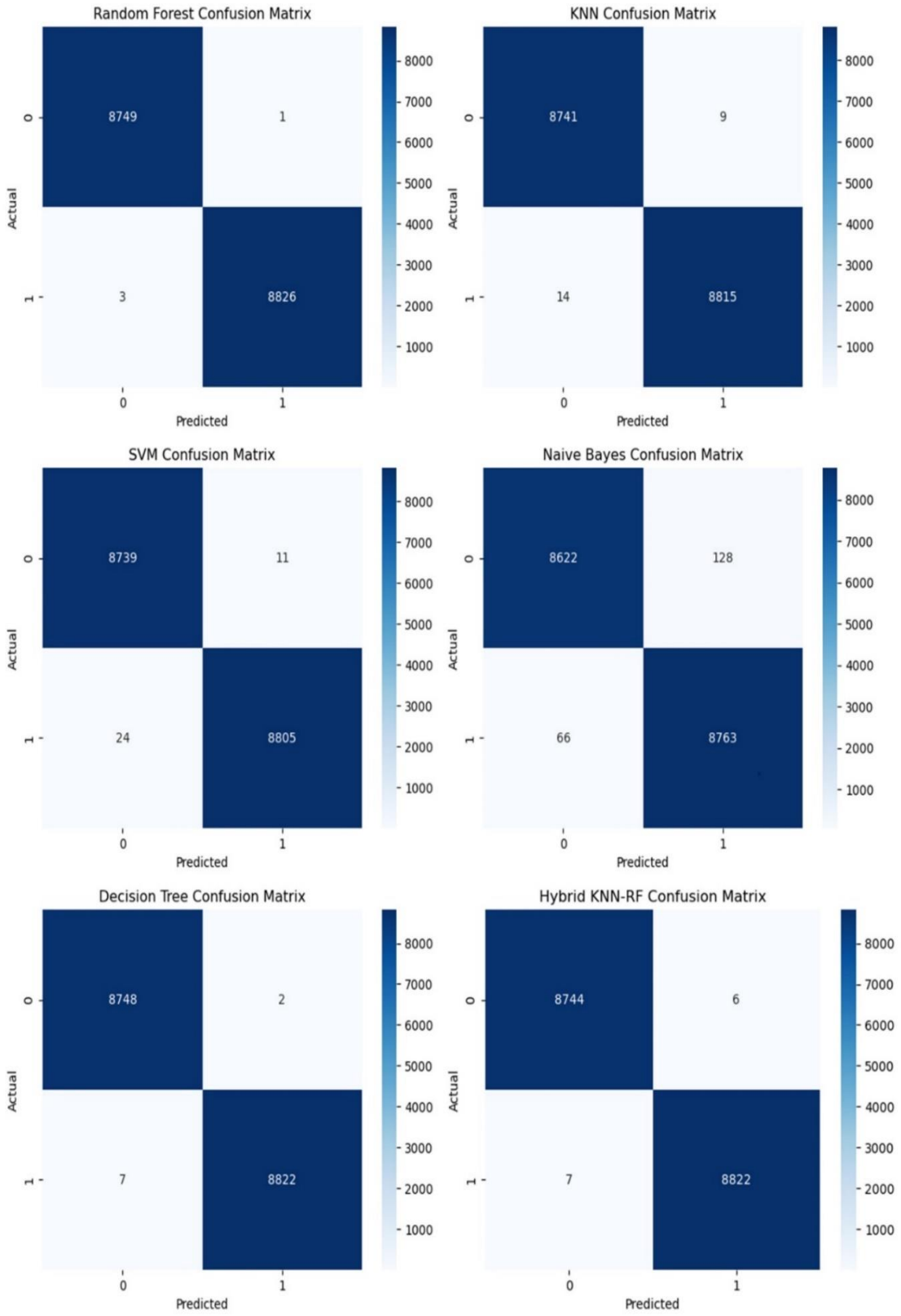

Figure 5: depicts the confusion matrix of classifiers for one-class

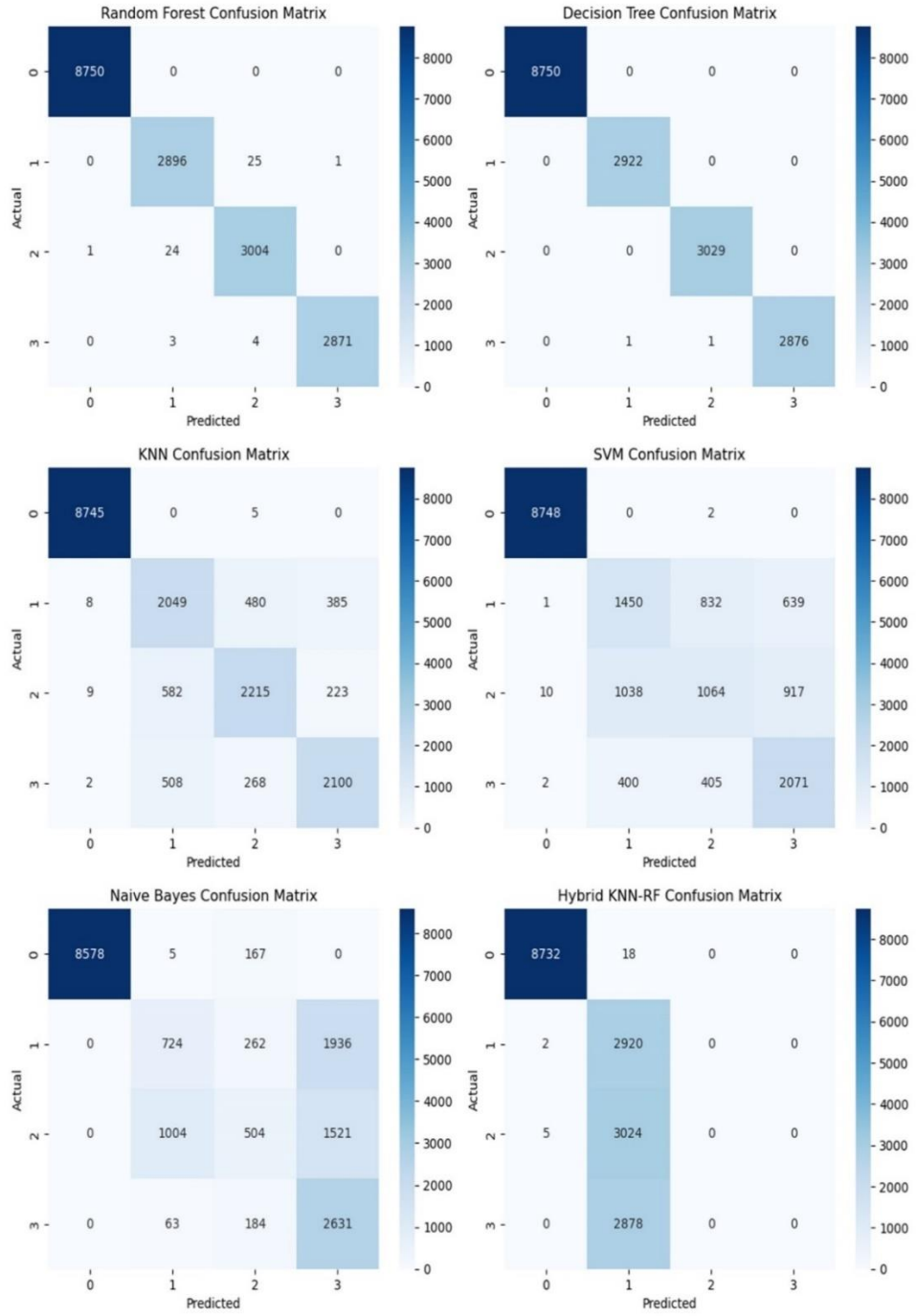

Figure 6: depicts the confusion matrix of classifiers for multi-class

Conclusion

This study used machine learning techniques for malware detection by examining the CIC-MalMem-2022 dataset. The data preprocessing steps, including the identification and selection of highly correlated features, were critical to achieving high model performance. So, out of all the tested models, the Random Forest algorithm did the best job of classifying things into one of two groups. This shows that machine learning methods can handle large datasets well and avoid overfitting. The slight drop in accuracy for the decision tree, hybrid model, and other individual classifiers with once-class highlights the importance of choosing the right model based on the specific characteristics of the dataset and the problem at hand.
Overall, the preprocessing steps and model selection led to highly accurate predictions, with all models achieving an accuracy of over 98.98%, indicating a well-prepared dataset and

effective modeling strategy. Otherwise, the decision tree accuracy with multi-class yielded the best results, demonstrating that this classifier is more suitable than others for multi-class classification problems due to its unique characteristics. These advantages include their ability to handle several classes, adapt well to various data distributions, and robustly manage missing values. These attributes make decision trees more versatile and often more effective than other classifiers for multi-class problems. We also conclude that the hybrid technique is ineffective when dealing with multi-class problems. In future work, we will experiment with this classifier in real-time, using and deploying a neural network on this dataset; subsequently, we will compare it with a machine learning classifier to achieve the best results and accuracy.

Reference

[1] M. S. Akhtar and T. Feng, "Malware Analysis and Detection Using Machine Learning Algorithms," *Symmetry (Basel)*, vol. 14, no. 11, Nov. 2022, doi: 10.3390/sym14112304.

[2] O. Or-Meir, N. Nissim, Y. Elovici, and L. Rokach, "Dynamic malware analysis in the modern era—A state of the art survey," *ACM Comput Surv*, vol. 52, no. 5, Sep. 2019, doi: 10.1145/3329786.

[3] Clare Stouffer, "10 types of malware + how to prevent malware from the start." Accessed: Feb. 13, 2024. [Online]. Available: https://us.norton.com/blog/malware/types-of-malware

[4] N. Pachhala, S. Jothilakshmi, and B. P. Battula, "A Comprehensive Survey on Identification of Malware Types and Malware Classification Using Machine Learning Techniques," in *Proceedings - 2nd International Conference on Smart Electronics and Communication, ICOSEC 2021*, Institute of Electrical and Electronics Engineers Inc., 2021, pp. 1207–1214. doi: 10.1109/ICOSEC51865.2021.9591763.

[5] D. Rabadi and S. G. Teo, "Advanced Windows Methods on Malware Detection and Classification," in *ACM International Conference Proceeding Series*, Association for Computing Machinery, Dec. 2020, pp. 54–68. doi: 10.1145/3427228.3427242.

[6] [6] A. Ehsan, C. Catal, and A. Mishra, "Detecting Malware by Analyzing App Permissions on Android Platform: A Systematic Literature Review," Oct. 01, 2022, *MDPI*. doi: 10.3390/s22207928.

[7] [7] S. Lysenko, K. Bobrovnikova, P. T. Popov, V. Kharchenko, and D. Medzatyi, "Spyware detection technique based on reinforcement learning," in *CEUR Workshop Proc.*, vol. 2623, pp. 307–316, Jun. 2020.

[8] B. M. Khammas, "Ransomware Detection using Random Forest Technique," *ICT Express*, vol. 6, no. 4, pp. 325–331, Dec. 2020, doi: 10.1016/j.icte.2020.11.001.

[9] E. Gandotra, D. Bansal, and S. Sofat, "Malware Analysis and Classification: A Survey," *Journal of Information Security*, vol. 05, no. 02, pp. 56–64, 2014, doi: 10.4236/jis.2014.52006.

[10] M. Zafar-uz-Zaman, "Proceedings of 2019 16th International Bhurban Conference on Applied Sciences & Technology (IBCAST): 8th–12th January, 2019," *Proc. 16th Int. Bhurban Conf. Appl. Sci. Technol. (IBCAST)*, Islamabad, Pakistan, Jan. 2019. [Online]. Available: https://doi.org/10.1109/IBCAST.2019.8667136. Accessed: May 14, 2024.

[11] A. Moser, C. Kruegel, and E. Kirda, "Limits of static analysis for malware detection," in *Proc. 23rd Annu. Comput. Security Appl. Conf. (ACSAC)*, Miami Beach, FL, USA, Dec. 2007, pp. 421–430, doi: 10.1109/ACSAC.2007.21.

[12] S. Zhang, J. Wu, M. Zhang, and W. Yang, "Dynamic Malware Analysis Based on API Sequence Semantic Fusion," *Applied Sciences (Switzerland)*, vol. 13, no. 11, Jun. 2023, doi: 10.3390/app13116526.

[13] E. S. Alomari *et al.*, "Malware Detection Using Deep Learning and Correlation-Based Feature Selection," *Symmetry (Basel)*, vol. 15, no. 1, Jan. 2023, doi: 10.3390/sym15010123.

[14] K. Bhat, T. Khairnar, S. Phatangare, and T. Narkhedkar, "Malware Detection Using Machine Learning," 2023. [Online]. Available: www.ijraset.com

[15] K. Sethi, S. K. Chaudhary, B. K. Tripathy, and P. Bera, "Novel malware analysis framework for malware detection and classification using machine learning approach," in *ACM International*

Conference Proceeding Series, Association for Computing Machinery, Jan. 2018. doi: 10.1145/3154273.3154326.

[16] M. S. Joshi, "MALWARE DETECTION USING MACHINE LEARNING TECHNIQUES," 2022. [Online]. Available: http://www.ijeast.com

[17] A. Kamboj, P. Kumar, A. K. Bairwa, and S. Joshi, "Detection of malware in downloaded files using various machine learning models," *Egyptian Informatics Journal*, vol. 24, no. 1, pp. 81–94, Mar. 2023, doi: 10.1016/j.eij.2022.12.002.

[18] S. Hassan Hashim and I. Ali Abdulmunem, "A Proposal to Detect Computer Worms (Malicious Codes) Using Data Mining Classification Algorithms," *Engineering and Technology Journal*, vol. 31, no. 2 B, pp. 142–155, Feb. 2013, doi: 10.30684/etj.31.2b.3.

[19] S. Shareef and S. Hashim, "Proposed Hybrid Classifier to Improve Network Intrusion Detection System using Data Mining Techniques," *Engineering and Technology Journal*, vol. 38, no. 1B, pp. 6–14, Apr. 2020, doi: 10.30684/etj.v38i1b.149.

[20] B. Mohammed and E. Gbashi, "Intrusion Detection System for NSL-KDD Dataset Based on Deep Learning and Recursive Feature Elimination," *Engineering and Technology Journal*, vol. 39, no. 7, pp. 1069–1079, Jul. 2021, doi: 10.30684/etj.v39i7.1695.

[21] H. S. Galal, Y. B. Mahdy, and M. A. Atiea, "Behavior-based features model for malware detection," *Journal of Computer Virology and Hacking Techniques*, vol. 12, no. 2, pp. 59–67, May 2016, doi: 10.1007/s11416-015-0244-0.

[22] K. Chumachenko, *Machine learning methods for malware detection and classification*, 2017.

[23] M. Chemmakha, O. Habibi, and M. Lazaar, "Improving Machine Learning Models for Malware Detection Using Embedded Feature Selection Method," in *IFAC-PapersOnLine*, Elsevier B.V., 2022, pp. 771–776. doi: 10.1016/j.ifacol.2022.07.406.