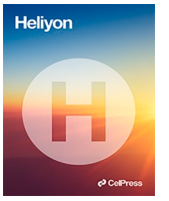
Heliyon 10 (2024) e23574

Contents lists available at ScienceDirect

Heliyon

journal homepage: www.cell.com/heliyon



Analyzing and comparing the effectiveness of malware detection: A study of machine learning approaches

Muhammad Azeem [a], Danish Khan [a], Saman Iftikhar [b], Shaikhan Bawazeer [b,*], Mohammed Alzahrani [b]

[a] Department of Computer Science, COMSATS University Islamabad, Wah Campus, Wah Cantt Pakistan
[b] Faculty of Computer Studies, Arab Open University, Saudi Arabia

ARTICLE INFO

Keywords:
Malware detection
Malicious detection
Feature encoding
Feature selection

ABSTRACT

The Internet has become a vital source of knowledge and communication in recent times. Continuous technological advancements have changed the way businesses operate, and everyone today lives in the digital world of engineering. Because of the Internet of Things (IoT) and its applications, people's impressions of the information revolution have improved. Malware detection and categorization are becoming more of a problem in the cybersecurity world. As a result, strong security on the Internet could protect billions of internet users from harmful behavior. In malware detection and classification techniques, several types of deep learning models are used; however, they still have limitations. This study will explore malware detection and classification elements using modern machine learning (ML) approaches, including K-Nearest Neighbors (KNN), Extra Tree (ET), Random Forest (RF), Logistic Regression (LR), Decision Tree (DT), and neural network Multilayer Perceptron (nnMLP). The proposed study uses the publicly available dataset UNSWNB15. In our proposed work, we applied the feature encoding method to convert our dataset into purely numeric values. After that, we applied a feature selection method named Term Frequency-Inverse Document Frequency (TFIDF) based on entropy for the best feature selection. The dataset is then balanced and provided to the ML models for classification. The study concludes that Random Forest, out of all tested ML models, yielded the best accuracy of 97.68 %.

1. Introduction

Many cyber-security methods have been devised and improved in recent years to protect against security threats. Even so, current statistics show that malware is always evolving and becoming more complicated. As a result, identifying and understanding their inner workings becomes more challenging. This can be attributed to two fundamental factors. Initially, attackers have enhanced their capacity to execute and conceal the potential effects of their attacks by incorporating anti-analysis techniques, such as compression and wrapping. Additionally, the increasing dynamism and diversity of contemporary communication and computing technology enable a single virus to manifest itself in various semantically and structurally distinct ways. Malware analysis becomes significantly more complex as a result.

* Corresponding author.
E-mail addresses: amuhammadazeem221@gmail.com (M. Azeem), danish56566@gmail.com (D. Khan), s.iftikhar@arabou.edu.sa (S. Iftikhar), shaikhan@arabou.edu.sa (S. Bawazeer), m.alzahrani@arabou.edu.sa (M. Alzahrani).

https://doi.org/10.1016/j.heliyon.2023.e23574
Received 17 August 2023; Received in revised form 5 December 2023; Accepted 6 December 2023
Available online 12 December 2023

Over the past few years, there has been an escalating threat of malware to network users, as evidenced by a 22.9 % surge in the number of new malware instances, reaching a total of 8,400,058 in 2017 compared to the figures from 2016 [1]. Merely blocking a firewall port or an IPS signature may not suffice. Given the pervasive reliance of both digital and physical applications on the internet, they can still be accessed and administered through online support and activities. The Internet has recently emerged as an important medium for various types of personal and professional activities for millions of users around the world. Attackers use a number of exploitation strategies to target individuals and organizations for a number of reasons, including stealing sensitive data and damaging computational capabilities [2]. As a result, the number of targeted online attacks on different internet services and software has increased significantly [3]. To assess the source, utility, and potential impact of a malware test, a malware analysis might even be created or evaluated. As a result, malware classification and analysis have become major topics in the digital world [4].

Today's security measures for dangerous or suspicious activities are built around the scanning system. Dynamic, static, statistical, and content analysis are some of the techniques used in these scanning methods [5]. On the other hand, hackers are aware of these protections. Hackers produce new malware that antivirus software cannot easily detect. Malware is software that is used to monitor machine activities, obtain access to a computer system, or gather critical information [6]. Malware is differentiated by its harmful expectation, which works in contradiction to the features of a computer client and does not include computer software whose deficiency produces unintentional harm [7]. The term "bed wear" is sometimes used to refer to specific malicious and unintentionally harmful software [8].

The menace of malware stands as one of the most significant threats to the internet in the present era. Malware has become increasingly perplexing as the Internet has advanced and grown in popularity. Malware is generally divided into the following classes, which are depicted in Fig. 1.

- **Trojans**: Trojan is a slang term for "trojan horse." It is an infectious code that affixes to another product and subsequently reproduces [1].
- **Viruses**: This is the easiest form of software to use. It's a contagious code that attaches itself to another product and then regenerates [2].
- **Worms:** Worms are self-replicating programs that erase or debase the computer's records. It tries to consume working framework documents and data files [3].
- **Spyware:** Spyware is any product introduced on a framework without the client's information. It keeps an eye on exercises performed by unfortunate casualties and tracks the web exercises to send a commercial to the framework. While not all spyware is vindictive, it is questionable because it can damage security and can be mishandled [4].
- **Rootkit:** Rootkits regularly change portions of the working framework or introduce themselves as drivers or part modules, contingent upon the inside subtleties of a working framework's systems [5].
- **Botnet:** A botnet is a collection of loosely connected programs. For any system structure, it is often a zombie program (worms, Trojans) running under ordinary resistors. A botnet is a massive network of computers that are traded off over the internet [6].
- **Crimeware:** Crimeware is an umbrella term for any malware with a common purpose of obtaining money or secret information. Crimeware addresses a growing problem in security management, as a growing number of malicious code threats attempt to steal sensitive information [2].
- **Adware:** Adware is a phrase that is often used to describe a sort of malware (malignant programming) that displays unwanted advertisements to a computer's user. Adware-delivered adverts appear seldom, like a spring, or once in a while in a enclosable window [7].

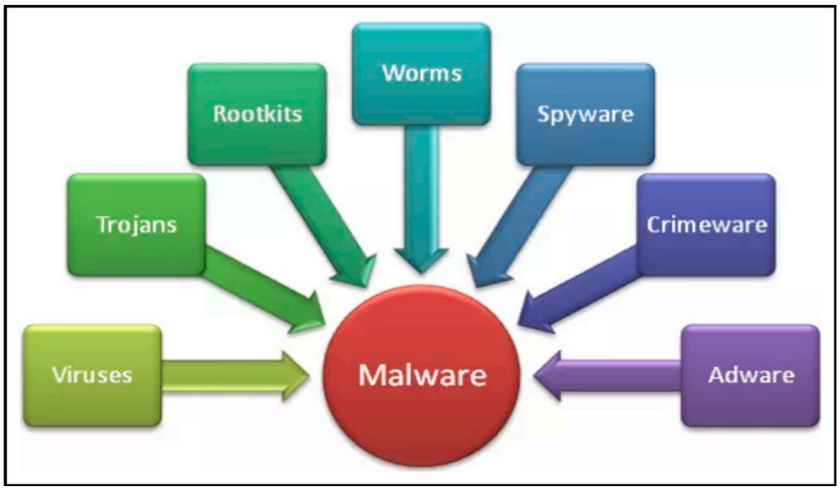


Fig. 1. Malware Types [1].

ML methods have proven useful in a variety of fields due to their excellent qualities like flexibility, versatility, and the capacity to quickly adapt to new and obscure problems. ML, particularly deep learning models, has made significant progress in the image categorization and computer vision (CV) fields in recent years. Considering extraordinary advancements in informal networks, cloud and web headways, electronic banking, adaptable condition, sharp system, and so on, computerized safety is a rapidly rising subject that offers a great deal of contemplation. Different ML solutions have been effective in addressing such a wide-ranging issue in PC security [8]. ML may be used in a variety of ways in the field of cybersecurity. Fig. 2 depicts the use of ML in cybersecurity.

ML is used in a variety of ways in cyber security, including phishing detection, arrange interruption detection, malware detection, anomaly detection, and much more. Marks are taken into account by standard antivirus or anti-malware software [9]. These approaches based on marks can be easily evaded. Polymorphic malware has little trouble eluding these traditional methods of detection. Malware detection is a problem of grouping. Even though ML aids in the protection of various structures, ML classifiers are vulnerable to nasty ambushes. Efforts have been exerted to enhance the precision of machine learning predictions and safeguard them against a range of attacks. ML learns the dataset and then predicts malware/non-malware based on its learning. To boost ML model execution, provide ever more current datasets. ML in Cyber Security would not only improve our digital security, but it would also allow us to progress toward more efficient innovation [10].

Malware is being generated and used in large quantities nowadays. To address this issue, a variety of security solutions employ ML to detect and avoid malware. Using this type of innovation will protect the industry while also boosting its line of defense. Hence, by determining the value of implementing ML and understanding its applications in cybersecurity to enhance security measures, we can enhance real-world security. The world has advanced to the point where using cyber technology is now a necessary component of daily living. Through the usage of breakthroughs like the IoT, cryptographic money, and other technologies, the use of computers and the internet has grown from staff computations and data access to encompass everything. However, such a deep integration of computers and other improvements poses new obstacles to the computerized world. As of right now, the globe is discussing complex economies and digital settlements. Attacks by malware are increasing, and many organizations are suffering as a result. Malware has greatly increased in recent years and is a major internet security threat that might result in risky behavior [11]. The goal of this thesis is to combine image processing and deep convolution network methods to produce operational and effective ways that can be used to continuously enhance the performance of detecting and classifying malware created over a lengthy period.

With the increased growth of the internet, defending networks from viruses is very difficult. The most common threat to network security is the denial of service resulting from an intrusion, such as brute force, or penetration within the network. Malware detection approaches rely on studying the characteristics of malicious code. Malware visual approaches have many challenges such as compound image texturing component extractions are expensive and time-consuming. When used with large datasets, feature extraction approaches also exhibit limited efficiency. When it comes to detecting malware code variants, static and dynamic approaches are less successful. One family may have a significantly higher number of images compared to others, which differs from the problem of class imbalance.

The purpose of this research remains to provide a pictorial exploration of a selected (UNSW-NB15) dataset in order to better comprehend the dataset's intricacies, which might lead to poor performance of data-driven models. The UNSW-NB15 dataset will be visually examined for defects that could limit the effectiveness of classification models. We employed a variety of preprocessing approaches to gather the data we required for visual analysis. We first analyze the UNSW-NB15 dataset for redundancy, convert the

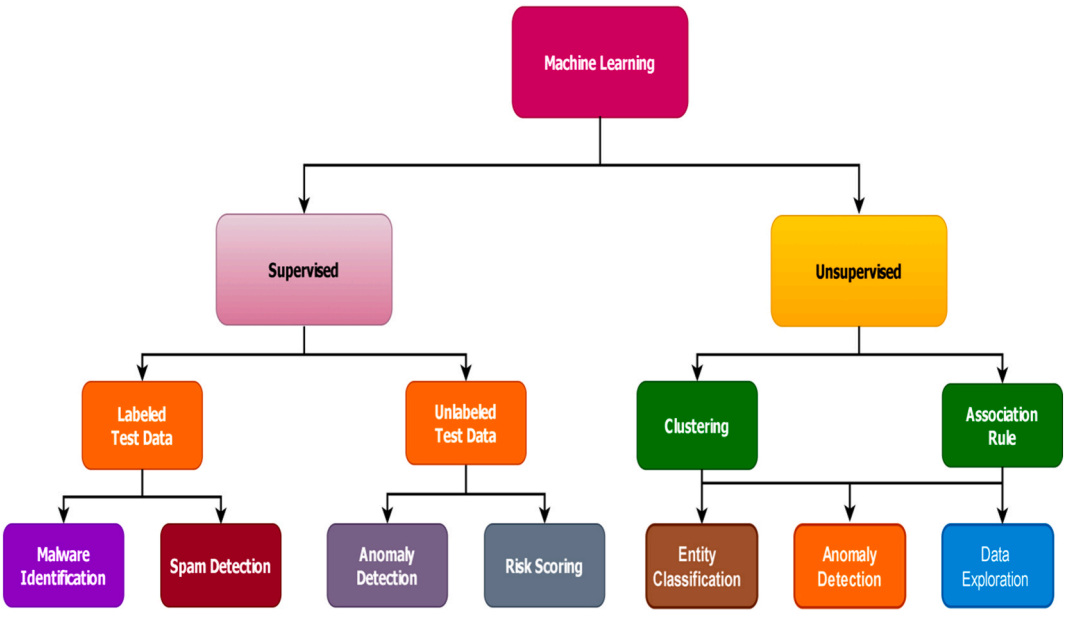


Fig. 2. Ml applications in cyber security.

actual input parameters to numeric, and dynamically resize them to choose the most significant input properties. PCA is used to project the preprocessed data into the bottom space. Finally, 3D scatter plots, t-SNE, and K-means inter-cluster distance maps were used as visualization techniques in this study. This marks the initial stage in techniques for detecting and classifying malware: transforming detectable malware into a digital image. The process involves implementing completed digital images with the subsequent steps, as illustrated in Fig. 3, representing executables in a numeric format by interpreting the execution content as a one-dimensional array. To store a digital image, the required number of bits is calculated using the expression $M \times N \times K$, where M is the number of columns (image height), N is the number of rows (image width), and K is the gray level. In our scenario, we are storing an image with dimensions 256×256 and 8 Gy Levels (equivalent to 3 bits). Thus, the number of bits required to represent the digital image is $256 \times 256 \times 3 = 196{,}608$ bits. These bits are then resized to a 2-dimensional row implementation of size 256×256. Convolutional Neural Networks (CNNs), a type of deep learning model, are employed to extract valuable features from malware images. To identify the areas of the image that contribute most to the model's predictions, activation maps of the CNN's intermediate layers are visualized. This process provides insights into the discriminative features learned by the malware classification model. Furthermore, malware images are reconstructed using generative models such as autoencoders or generative adversarial networks (GANs). By comparing the original and reconstructed images, it is possible to identify the significant properties that the model utilizes to represent and generate malware samples.

We attempted to get good findings in this study. Using several ML algorithms and feature selectors, we chose distinct features from a huge dataset and improved good outcomes, as shown in the results section. We are also becoming used to procedures that require less time and increase tolerable accuracy.

2. Research contributions

The contribution of this research article is summarized in the following step.

- Develop novel techniques such as features detection, Top 10, Top 20, and Random features.
- LR, DT, nnMLP, RF, ET, and KNN model performance have improved by using ML to reduce the effects of unbalanced data.
- Enhanced overall detection performance and offered suggestions for managing class imbalance.
- LR, DT, nnMLP, RF, ET, and KNN models deployed for malware detection situations been examined.
- To achieve low-latency detection, investigated effective feature extraction methods and model optimization strategies.
- Analyzed the models' streaming data performance to help malware detection systems be more effectively implemented.

3. Related work

Over the last ten years, there has been a significant increase in research focused on the analysis and detection of malware. This section looks at challenges with detecting malware as well as some of the more interesting ways that have been proposed in the literature. ML has established its dominance in recent years and has carved out a new study zone. Deep learning has been utilized for malware detection and recognition by several scholars because of its cutting-edge capacity to absorb the most ideal features. Various forms of malware encompass computer viruses, worms, Trojan horses, backdoors, rootkits, boot disks, spyware, adware, and so on [12]. Malware does not have a conjointly special sense and offers multiple classes at an equivalent time. This suggests that the same malware is related to malicious files with the same malicious behavior [13]. Malware is among the foremost threats and safety intimidations to the network today, consistent with a review led by Ref. [14] that found that 47 organizations had practiced malware security occurrence network breaks in the previous year. Malware continues to grow, increasing the risk of landscaping, the variety of modern malicious methods, and the speed of the flow of threats [15].

The use of complementary and orthogonal techniques can frustrate hackers, making it more challenging and focused for them to evade complete detection layers. While keeping to the traditional method, orthogonal image processing and computer vision solutions are focused in Ref. [16]. The ways of characterizing and detecting malware in this article are different and innovative.

More information on virus community can be gleaned from captured images [17]. Different sections of binary are immediately evident when malware is viewed as a picture. Distinct sub-parts have different architectures, as seen in Fig. 4. A runnable application

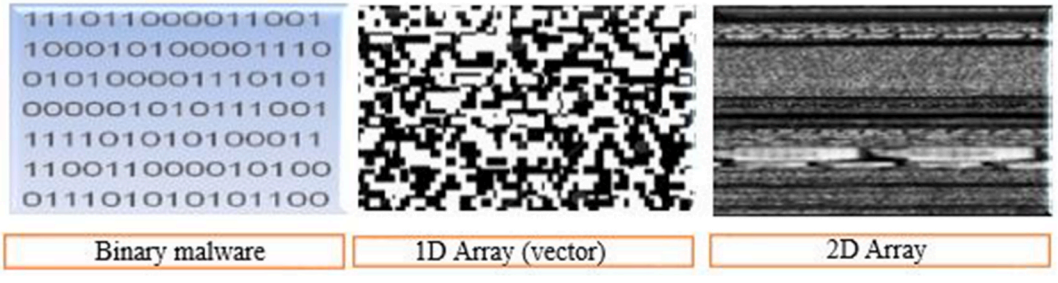


Fig. 3. Visualization of malware image.

can be found in the "Text section." The beginning of Fig. 4 is the well-structured code in the first half of text section. The respite is completed by zero (0), which symbolizes nil padding at the end of this section. The segment is surrounded by both black patches (non-removable code) and fine-grained texturing after the data (initial data). The 'Rsse' portion is the last, covering the complete assets of the unit. These might include symbols that the application uses. Mutations are responsible for the dramatic increase in the total amount of malware. A metamorphosis might be a method for creating virus forms that have never been seen before [18].

To create useless variants, hackers make minor tweaks to existing software. Malware from the very same group with identical characteristics and malware from two different families with different characteristics have led to the notion of separate botnets [20]. This study proposes a new way to evaluate and categorize malware using image processing, which supports this observation. The article presents a total inquiry about an audit on distinctive strategies of recognizing and classifying malware. This study analyzes malware discovery methods in two fundamental spaces: "signature-based" and "behavioral-based".

C hanging binary code to picture may be done in a variety of methods. The malware data in binary form is then examined in the form of a vector of eight unsigned piece numbers and a 2D vector of a brief timeframe future which can be shown as a grayscale image in the [0,255] range (0: black, 255: white). The photos' width is fixed, while the height is allowed to vary depending on the size of the record [21]. The virus to grayscale picture translation is shown in Fig. 5. Each malware test is a succession of bytes and every byte can be changed over into an 8-piece dark pixel, and the byte arrangement of the example document is fragmented by the predefined width and changed over into a two dimensional dim scale grid, which is a grayscale image [22]. A malware binary component string can be a collection of distinct 8-bit substrings. Because the 8 bits may be decoded as an unsigned number in the range of zero to two fifty-five, all of these substrings can be perceived as pixels. If a string is 011011110101100, for example, the process is 0111,100,010,101,100 → 0001,100,000,10,101,100 → 96,162. It is possible to convert an eight-piece twofold number B = (b7, b6, b5, b4, b3, b2, b1, b0) into a decimal number.

Singh, Jagsir et al. [23] proposed method in which ML parameters are modified to generate high accuracy results in binary file categorization as malware or benign. The k value and kernel size are two important elements in ML algorithms. This approach evaluates the depth of tree, transfer functions, split factors, and integral gain. API calls are used to achieve malware classifiers high accuracy results. Finally, there's supervised ML. There are 6434 benign and 8634 malicious samples that were tested in all classification techniques. Using ensemble methods, the malware classifier achieved 99.1 % accuracy. This research sheds light on how to fine-tune the parameters of ML algorithms to detect malware via API calls. Urmila, T.S et al. [24] used a behavioral characteristics dataset to assess the impact of malware prevalence. The attributes are used in LBGM, ENB0, and EEXR prediction models for malware detection. A Gatherer Segment, a Detergent Stage, a Selector and a Detector Phase are all included in this proposed system. The presented scheme accomplishes an essential process for recognizing malware over these four levels. Bestowing to effectiveness results, the current recommended predictive EEXR has achieved accuracy of 96.75 %, and a low error ratio of 3.28 %.

Alshamrani et al. [12] introduced method which recognizes huge data security concerns and simultaneously evaluates harshness, threat score, self-assurance, and longevity. The suggested study is assessed in 2 ways: initial, various ML algorithms are compared. Then after getting the best F-measure, accuracy, and retention scores, entire reputation is compared. Not only this, but the recommended framework is also validated with exterior fundamentals, nonetheless too capable of reducing security concerns left unaddressed by existing obsolete systems engines of reputation. Table 1 presents existing methods of malware detection and classification. In O. Aslan et al. [13] work, goal of research is to develop a novel hybrid method that better blends two types of pre-trained network architectures. The four primary stages of this architecture are data gathering, deep neural network architecture, suggested deep learning models design train, and assessment of a proficient Deep Neural Network (DNN). The suggested technique is experienced using the "Malimg, Microsoft BIG 2015, and Malevis datasets". Experiments reveal that the suggested method is capable of accurately classifying malware. The suggested solution outperformed most ML-based virus detection algorithms when tested on the Malware dataset, with 97.78 % accuracy.

Mohammad Masum et al. [14] made use of various ML methods for the classification of ransomware using classifiers based on DT, RF, NB, LR, and NN. To test the suggested approach, all experiments are executed on a single ransomware dataset. Experiments reveal that RF classifiers performed best than other approaches concerning accuracy, F-1, and precision measures. MS Abbasi et al. [15] presented behavior-based ransomware detection and classification, in this paper presents an automatic selection of feature techniques based on sub-division group optimization. The proposed technique aiming to detect and classify ransomware, reflects the importance of distinct data feature groupings and selects features depending on their value. The experimental results show that in many

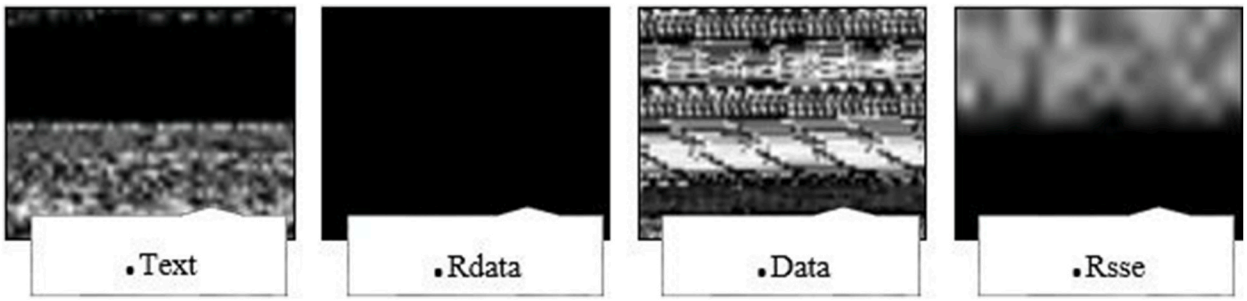


Fig. 4. Information from 'Pe file' [19].

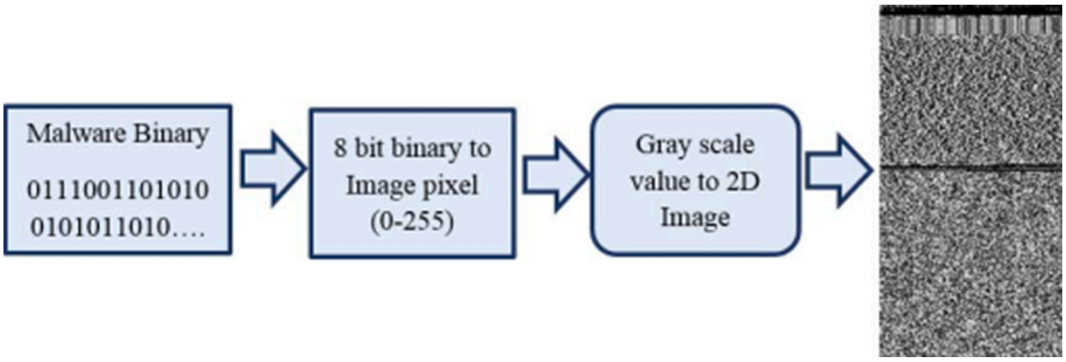


Fig. 5. Malware Binary to Gray scale.

Table 1
Dataset and features.

Filename	File size	Number of records	Number of features
UNSWNB15_1.csv	161.2 MB	700000	49
UNSWNB15_2.csv	157.6 MB	700000	49
UNSWNB15_3.csv	147.4 MB	700000	49
UNSWNB15_4.csv	91.3 MB	440044	49

circumstances, the suggested method outperforms current state-of-the-art benchmarking methods. This article also discusses the significance of several feature groups and the features chosen by the suggested method in the detection of ransomware as well as categorization. Pinhero, A. et al. [25] proposed to study the efficiency of a new strategy that leverages malware visualization to solve problems with deep learning classification as well as feature selection and extraction, which has less sensitive performance. Learning is superior to a limited dataset. Twelve different people were used in the studies. With a collection containing 20,199 malware and neural network architectures, researchers show that their proposed method is efficient, as evidenced by the F-measure of 99.97 %. The approach [25] developed by Nighat Usman et al. emphasizes big data forensics concerns and generates seriousness, risk score, confidence, and lifespan all at the same time. The presented scheme is assessed in two directions: first, ML approaches are compared, and then whole reputation network is compared to current distributed systems using the better F1, precision, and recall scores. This research approach cannot solitary be validated using peripheral factors, nonetheless, it can also be used to address security vulnerabilities that were previously overlooked by obsolete reputation engines. Robertas Damaevius et al. [26] and colleagues examined and

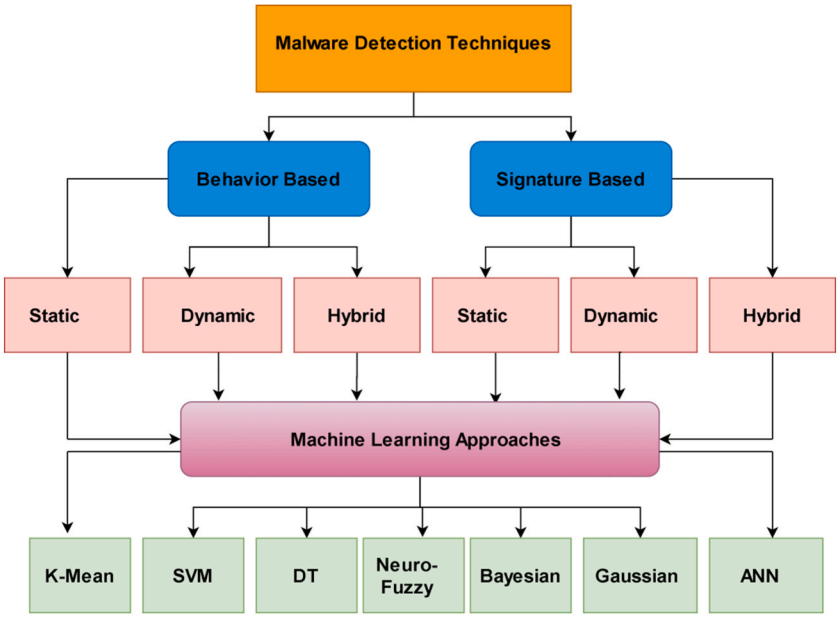


Fig. 6. Malware detection techniques.

analyzed 14 classifiers. 13 ML approaches are being used as a benchmark comparison, and experimental results done on the classifier of ransomware using PE Headers ClaMP datasets are presented. As a meta-learner, it combines five dense and CNN artificial neural networks as well as Extra Trees classifiers to achieve the greatest results.

The study [27] presented hybrid visualization of viruses which analyses both static and dynamic data. It uses "Cuckoo Sandbox" to do simulation of samples in a dynamic environment while using a defined technique to convert the dynamic analytical results into a visualization image and training a neural network. In hybrid visualizing, there is a net on static and hybrid mapping images. Finally, the attack detection technology was put to the test. The research in Ref. [28] showed that "MTHAEL" was thoroughly tested by a huge IoT. It successfully classified 99.98 % of ARM architectural examples in a cross-architecture dataset of 21,137 samples, outperforming previous similar work. Overall, MTHAEL has proven to be viable for cross-architecture IoT detecting attacks with small processing overheads needing just 0.32 s to identify every threat. Using multiple learning methods, the authors experimentally analyze forecasting ability of feature extraction methods and evaluate their forecast performance and execution time. J. Abawajy et al. [29] gives results of the studies that show that feature extraction is crucial for enhancing learning model accuracy while also reducing route period. The data also show that feature selection strategies differ in effectiveness from one learning algorithm to next, and that no one feature extraction method continuously outperforms others. The work in Ref. [30] employed a model-checking technique based on SAT to find two beforehand identified defects and three new problems i.e., modern web form, validation that is appropriate to look at as well as a single sign-on solution.

Malware's rapid evolution and complexity poses a fundamental challenge to the sophisticated world. Various security arrangements, such as different Antivirus approaches have been built and new methodologies have been investigated to regulate and limit the damage caused by malware [26]. These antivirus approaches are categorized based on signatures and behaviors. Fig. 6 depicts the flow of malware detection approaches. Two main sorts of antiviral strategies may be distinguished: those based on signatures and those based on behavior. The fundamental ideas and methods for identifying and thwarting malware in these areas vary. The sequential stages required in both signature-based and behavior-based antiviral techniques are depicted visually below in Fig. 6. By identifying and counteracting diverse malware types while securing user data and privacy, these techniques are essential for protecting systems and networks.

The research of a single sign-on Kerberos-based system is presented in this paper [27]. Differences between secure network protocol that uses bespoke client software, and a comparable but unprotected cookie-based web protocol redirection and embedded links are illustrated. Jian Mao et al. [28] presented a detection technique to spot malicious activities in JS submissions. This process replicas an application's ancient activities on operating initiation, which is employed as an origin to observe spasms. The researchers tend to prototype answer on the favored JS locomotive, V8, and used it to observe attacks on the humanoid structure.

This work, describes a new error detection algorithm that identifies both transient and persistent flaws and are efficiently implemented in the NTT accelerator design. found flaws in such structures after recomputing and decoding by encoding the operands using two ways, namely negating and swapping [29].

The study referenced in Ref. [30] presented the best research and education integration method to address this issue at the Rochester Institute of Technology in this work. Furthermore, describes the outcomes of a year-long application of the given technique at the graduate level via "side-channel analysis attacks" case studies.

The study in Ref. [31] first explored the AES method from the standpoint of concurrent fault detection. Here this paper highlighted that, along with AES's efficiency requirements, it must be resilient against temporary and permanent internal faults, as well as malicious errors aimed at leaking the secret key. Since fault attacks have become a substantial problem in cryptographic applications, reliability analysis and the development of effective and cost-effective fault detection techniques are critical. As a result, they proposed, developed, and implemented several unique concurrent fault detection algorithms for various AES hardware designs. These include structure-dependent and structure-independent methods for identifying single and multiple stuck-at faults with single and multi-bit signatures.

SABRE is a simple and adaptable cryptographic method that is ideal for mitigating possible postquantum attacks. SABRE implementation can be done entirely in hardware (HW) or on HW/software coprocessors. Falcon, on the other hand, is a highly suited signature technique for PQC due to its short key size, efficient design, and good reliability demonstration in the quantum random oracle model (QROM). Although Falcon is an important PQC signature method, its use of the Gaussian sampler renders it subject to malicious assaults, such as fault attacks. This is the first paper to propose error detection algorithms that can identify both transient and permanent defects and are efficiently implemented in SABRE and Falcon's sampler designs [31].

This survey [32] has resulted in ongoing research on PQC algorithms, which are thought to be resistant to both classical and powerful quantum computers. Unfortunately, algorithmic security is insufficient, and many cryptographic algorithms are subject to side-channel assaults (SCA), in which an attacker passively or deliberately obtains side-channel data to undermine ostensibly secure security qualities. This survey investigates such impending dangers and their countermeasures about PQC. We present the most recent breakthroughs in PQC research, as well as analyses and views on the many forms of SCAs.

The article [33] examines numerous differential and side-channel analysis (SCA) attacks done throughout ASCON cipher suite versions in terms of algebraic, cube/cube-like, forgery, fault injection, and power analysis attacks, as well as defenses for these attacks. Throughout this poll, we also share our ideas and visions in order to propose fresh future directions in several sectors. This is the first study of its sort, and it is an important step in examining the benefits and prospects of the NIST lightweight cryptography standard, which was established in 2023.

In this study [34] provides error detection algorithms for the Camellia block cipher, taking into consideration its linear and non-linear sub-blocks, They also customized the proposed error detection designs to the desirability of employing different S-box variations depending on security and reliability goals. Their proposed approach has these advantages: (a) they can be tailored and also

can be applied to table-based as well as field-based S-boxes; (b) based on usage of models reliability vs. overhead can be fine-tuned (c) results in high error coverage with acceptable overheads for performance and implementation metrics. To assess the effectiveness of the proposed approaches, they have shown the results of the error simulations as well as application-specific integrated circuit implementations.

This paper presents the [35] new security trends for deeply embedded systems that are emerging by avoiding cryptographic calculations or employing lightweight crypto-systems that are practical for these computing platforms. Indeed, the potential for extending these developing security techniques to sensitive applications such as healthcare IT for implanted medical devices, big data analytics, ML in deeply embedded systems, smart buildings, and smart fabrics is enormous.

4. Dataset

The computer and network security dataset "UNSW-NB15" was made public by Moustafa and Slay, in 2015 [36]. The current dataset contains 2,540,044 real-time normal and aberrant (also known as attack) network events. To collect these records, the developer of the IXIA network deployed three virtual machines. The first two machines are set up to handle standard network traffic. The third, on the other hand, set up to produce unusual traffic. These dataset data records are organized into four CSV files, each file contains several attack and normal records. The datasets and features are summarized in Table 1.

The IXIA PerfectStorm tool in the Cyber Range Lab of the Australian Centre for Cyber Security (ACCS) creates the raw network packets for the UNSW-NB 15 dataset to provide a mix of genuine current normal activities and synthetic contemporary attack behaviors. 100 GB of the raw traffic is captured using the Tcpdump program (e.g., Pcap files. Fuzzers, Analysis, Backdoors, DoS, Exploits, Generic, Reconnaissance, Shellcode, and Worms are the nine attack families represented in this data set. To produce a total of 49 features with the class label, the Argus and Bro-IDS tools are used, and 12 methods are built. It frequently takes effective computational resources, adequate algorithms, and appropriate data-handling approaches to analyze and handle datasets of this size to produce accurate and trustworthy findings. To extract useful data and make wise judgments, researchers and practitioners working with datasets would probably need to use scalable and optimized approaches. Overall, the dataset allows broad data exploration, analysis, and model creation due to its feature descriptions and large number of entries dispersed over numerous CSV files. Due to its scale and complexity, it must be utilized with cutting-edge methods and tools to reach its full potential in a variety of fields, including network security, anomaly detection, and intrusion detection.

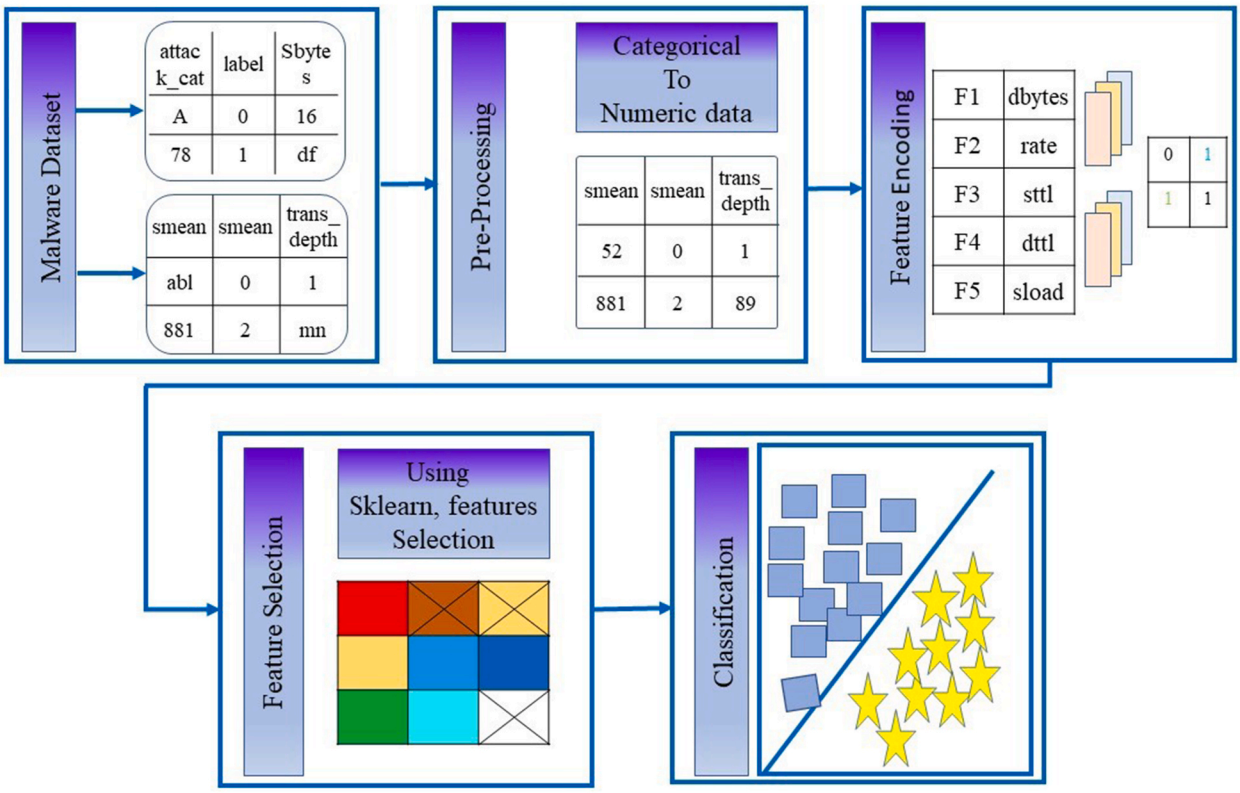


Fig. 7. Research proposed Model.

5. Proposed method

Using the appropriate coding technique is very important. Considering the dataset that we are working with, as well as the model we will employ, the current study is conducted in the system, as shown in Fig. 7, in three phases: (1) Analyzing of data, (2) Preprocessing of data, (3) Feature Selection, and (4) Classification using different classifiers. Finally, the used different classifiers compare different encoding methods and results as shown in the results section.

5.1. Preprocessing

Data preprocessing data is one of the most significant stages when dealing with textual models, thus the initial stage is important. While preprocessing data we must consider the distribution of our data, the methods required as well as the depth to which we should clean during this stage. To generate a collection that only includes categorical data, we first split the data set into two subgroups: categorical and numerical data. We then descend all the data and change our tag to a numerical form. We made use of the bag-of-words model which is concise and simple: it generate a vocab from a document corpus and counts the number of times each word appears in each document; next, we use word2vec to create a vector space, normally with many hundred measurements, for each distinct class label in the corpus; parameters that share similar contextual factors in the textual data are grouped during space.

5.2. Feature encoding

Since many ML algorithms need numerical input, categorical information must be transformed into numeric values before being fitted into an algorithm [37]. NLP refers for natural language processing. Artificial intelligence discipline that researches interactions between human languages and computers, and how to configure computers to handle and study massive volumes of data of data in natural language. Modern NLP systems and techniques ignore context and treat words as isolated entities because directories are utilized in terminology to denote word similarity [38]. The challenge of categorizing text data by its content is known as text classification. Despite being more challenging to handle, categorical data are regularly encountered in data science and ML problems.

Preprocessing categorical data is essential since most ML methods only take into consideration numerical variables. The categorical data must be transformed into numbers for the model to comprehend and return useful data in vector form. We used the TFIDF to determine how frequently a categorical variable appeared in our corpus and document frequency as a score for the term I in document j, as given in Equation (1).

$$TFIDF = TF_{(i,j)} * IDF_{(i)} \tag{1}$$

Where, IDF=Inverse Document Frequency , TF= Term Frequency.

$$TF(i,j) = \frac{Term\ i\ frequency\ in\ document_j}{Total\ words\ in\ document} \tag{2}$$

$$IDF(i) = log_2\left(\frac{Total\ documents}{document\ with\ term\ i}\right) \tag{3}$$

The above Equations (2) and (3) can be generalized into one component. The actual mathematical formula become Equation (4):

$$w_{i,j} = tf_{i,j} * log\left(\frac{N}{df_i}\right) \tag{4}$$

Here, $tf_{i,j}$ = *number of occurrences of i in j*, df_i = *number of documents containing i*, and N = *total number of documents*.

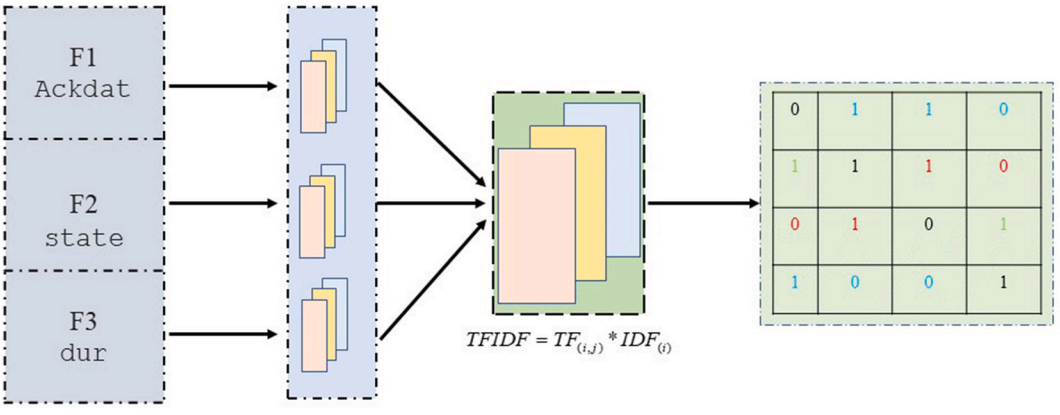


Fig. 8. Feature encoding.

We applied the vector space model to categorical data to encode categorical variables with numeric values, as shown in Fig. 8.

5.3. Feature selection methods

Several feature selection or reduction techniques have been employed in previous research to classify and identify malware features. However, there is still work to be done on the features selection technique given our minimal understanding. The goal of the feature selection procedure is to choose a subset of features from the list of capabilities for dimensional characteristics to classify data as precisely and little as possible. The important goal of optimum feature extraction is to eliminate unnecessary features and accelerate model performance. In this study, the generated feature vectors are used to construct entropy-based feature selection.

In this study we made use of the features selection methodologies named Information Gain (IG) and Gain Ratio (GR) entropy-based approaches. These techniques are used to evaluate features and remove the less useful features from the dataset.

5.3.1. Information gain (IG)

Information gain has the capability to remove the bias factor for multi-valued features when selecting an attribute. For the ML techniques, it is widely used to calculate the outcomes in words and increase in class knowledge. It is a technique that is widely employed to draw out pertinent facts from data. Using characteristics with the top value of Information Gain, a document is categorized (IG). By decreasing the total entropy value and by analyzing the impact of the feature by including them, information gain is evaluated. In Equation (5), entropy is represented at position 3.

$$Entropy = -\sum_{k=1}^{n} p_i * Log_2(p_i) \tag{5}$$

5.3.2. Gain ratio (GR)

The quality of the feature is determined by the gain ratio (GR). It selects a minor gesture from a collection based on GR score using a recurring strategy. The feature inequality is determined by GR. The value of a feature is determined by its Gain Ratio (GR) top-score. The split value of the information is represented by the normalization value. One of the n feature outputs corresponds to each of the n parts of the training text. It may be calculated using Equation (6).

$$Split\inf o(G) = -\sum_{k=1}^{n} \frac{|G_k|}{|G|} X \log_2\left(\frac{G_k}{G}\right) \tag{6}$$

In this equation high split of info indicates that the information is low and constant. only a small percentage of partition preserve their peak values. The GR can be computed using Equation (7).

$$GR = \frac{Gain(F)}{Split\inf o(F)} \tag{7}$$

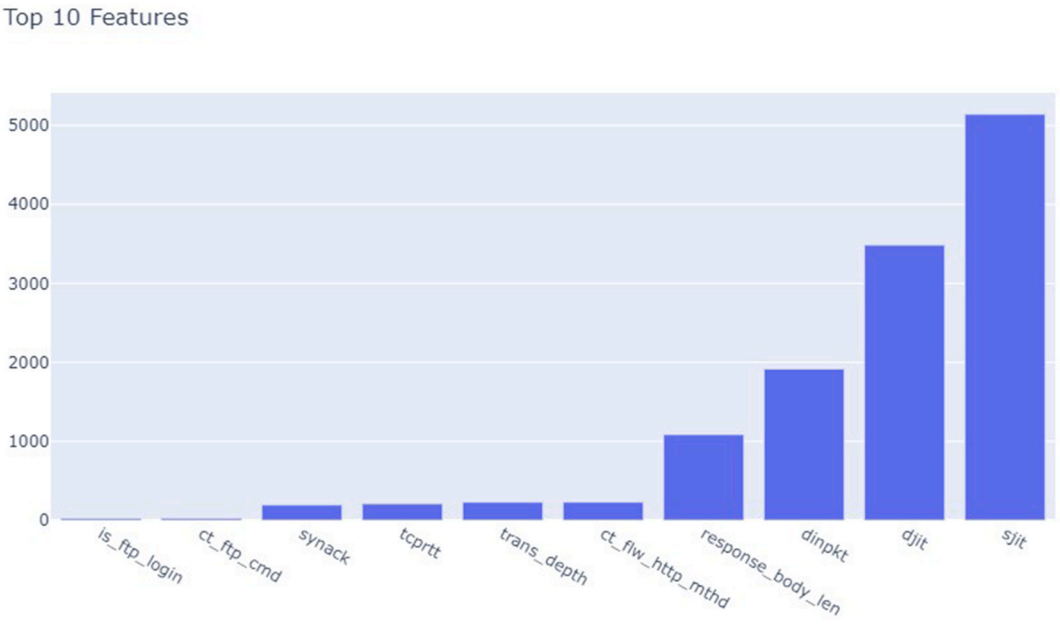


Fig. 9. Graphical representation of top 10 features.

5.4. Different selected features

A strong knowledge of the data is required for successful analysis. The data is processed to help algorithms and increase efficiency before the real analysis begins. The following statistics show certain aspects in terms of their amount, size, and type, are as follows.

Step1. Applying the chi-squared statistical test to choose the top 10 features from our dataset using the SelectKBest feature selection technique from scikit-learn. For feature selection based on univariate statistical tests, the SelectKBest class is utilized. In this instance, we're using the chi-squared test (chi2) as the scoring function, which requires that we compute the score for all features. The names of the selected features are sjit, which has a size of 5148.778, sbytes, which has a size of 3485.447, dinpkt, which has a size of 1916.648 and is_ftp_login, which has a size of 21.69648, as shown in Fig. 9.

Step2. The Top 20 features from our dataset will be chosen for the second round using the SelectKBest feature selection technique from scikit-learn and the chi-squared statistical test. In this instance, we compute the score for each feature using the chi-squared test (chi2) as the scoring function. Fig. 10 shows that the size of the first feature, ct src ltm, is 99.9865 k, followed by ct srv dst (82.63284 k), ct srv src (78.87113 k), and ftp login (51.71041 k).

In the bar graphs in Fig. 10, multiple features are listed together with their feature names and sizes, which also indicate the number of necessary features. First, we chose the top 10 features, which are displayed in a bar graph. Next, after completing another step, we chose the top 20 features using a feature selector. After that, random characteristics are chosen, and we use several methods to see if the accuracy improves or degrades. The detailed results are provided in the next section.

6. Experimental setup

We provided a proposed method for the efficient detection and the classification of malware in an experimental scenario. One type of dataset that is globally used for the trials. The goal of this dataset is to produce a combination of genuine modern regular activities and synthetic modern attack behaviors. Only one dataset is used in this research paper which is obtained from Kaggle, The UNSW-NB 15 dataset's raw network packets were generated by the IXIA PerfectStorm tool in the Australian Centre for Cyber Security's (ACCS) Cyber Range Lab. This dataset contains different types of attacks named as Fuzzers, Analysis, Backdoors, DoS, Exploits, Generic, Reconnaissance, Shellcode, and Worms. Several ML techniques are used to for the classificationof data with the evaluation measures accuracy, precision, recall, and time mappings. The entire simulation is carried out on a laptop with 8 GB of RAM and a 1.61 GHz processor while using the Python IDE. The efficiency of the suggested strategy is validated in this Section using various aspects of multi-representative and choice strategies to provide results from various classifiers. Seven test cases are created from the outcomes. The results of these tests, along with an analysis of them, are shown below in the coming sections.

6.1. Experiment on all features

The second test run on an all-features dataset, and each class had all numerical values. In test 1, the traditional approach is used for feature combination and entropy-based feature selection. As shown in Table 2, the RF on a good dataset obtained the most extreme

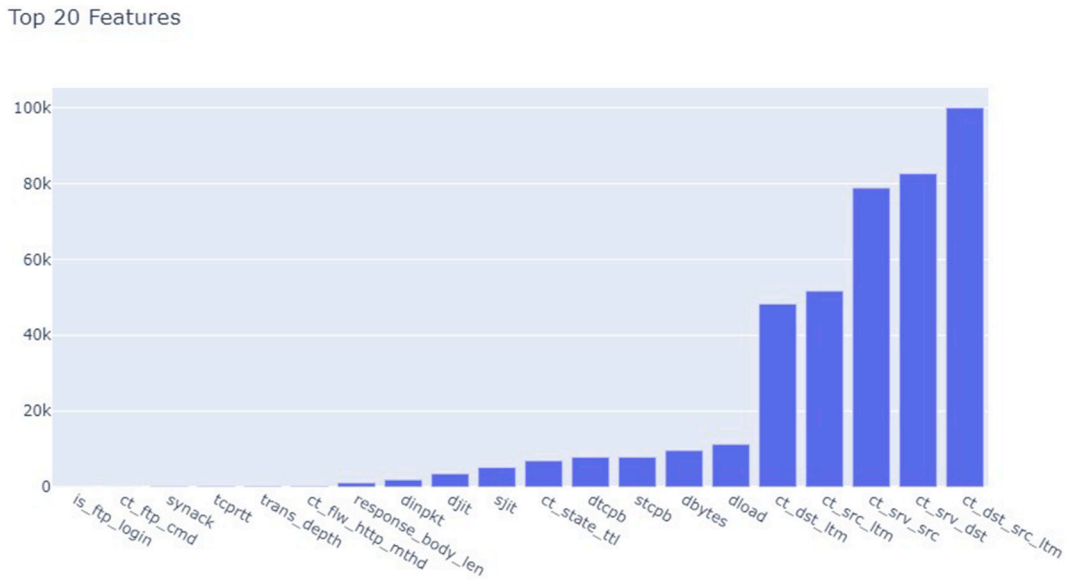


Fig. 10. Graphical representation of top 20 features.

characterization accuracy of 97.68 %, recall of 97.68 %, precision of 97.68 %, and F1-Score of 97.68 %.

It is clear from the second test's graphical depiction that an all-features dataset are employed. This indicates that there is no feature selection or reduction done, and all features that are accessible are included in the study. There is no missing or incomplete data points for any class since each class in the dataset contained entire numerical values. This completeness guarantees a trustworthy and thorough study of the dataset. Both feature combination and entropy-based feature selection techniques are used in the first test. This is accomplished using the traditional method, which is a well-accepted and accepted strategy. This technique often entails merging or combining pertinent information to produce fresh feature combinations that may improve the overall forecasting ability.

As compared to all of the models applied with respect to the all-features dataset, RF produced the highest extreme classification accuracy (97.68 %), recall (97.68 %), precision (97.68 %), and F1-Score (97.68 %) results, as shown in Fig. 11. When compared to the imbalanced dataset, accuracy rises while compatibility falls in the relevant data.

6.2. Dropped top 10 features

A dataset with all numerical values for each kind is utilized in test 2. To examine if removing Ten features improved accuracy in test 2, we looked at the results. The traditional approach and entropy-based feature selection are then used. The most extreme characterization accuracy, recall, precision, and F1-Scores of 97.35 %, 97.35 %, and 97.35 % are achieved by the RF on a robust dataset, as shown in Table 3. For relevant data, accuracy is greater and training time is shorter as compared to a balanced dataset.

The graphical representation of test 2, dataset with all numerical values for each kind is utilized in the Forth test. To examine if removing Ten features improved accuracy in test 2, we looked at the results. The traditional approach and entropy-based feature selection used.

The most extreme characterization accuracy, recall, precision, and F1-Score of 97.35 %, 97.35 %, and 97.35 % is achieved by the RF on a robust dataset, as shown in Fig. 12. For relevant data, accuracy is greater and training time is shorter as compared to an all-features dataset.

6.3. Comparison between all features results and dropped top ten features

Comparing findings on the top 10 dropped features dataset, Fig. 13 demonstrates that, results on the whole features dataset vary. When top 10 characteristics were eliminated, the accuracy of multiple classifiers changed.

i.e. The accuracy of LR is 92.46 %, KNN has an accuracy 96.01 %, DT 96.16 %, ET 97.15 %, RF 97.35 %, and NN MLP achieved an accuracy 95.99 % shown in Fig. 13.

6.4. Drop top 20 features

A dataset that included all numerical values for each kind was used for test 3. We examined the results to determine whether the accuracy of test 3 is enhanced by removing 20 features. Following that, a traditional approach and feature selection based on entropy are used.

The RF model performed best with the accuracy, recall, precision, and F1-Score on a robust dataset, which is shown in Table 4 and Fig. 14. For relevant data, accuracy is better and training time is less than for a balanced dataset.

The graphical representation of test 3 used a dataset that includes all numerical values for each kind used for the third test. We examined the results to determine whether the accuracy of test 3 is enhanced by removing 20 features. Following that, a traditional approach and feature selection based on entropy are used.

6.5. Comparison between all features results and dropped top 20 features

Comparing results from the top 20 deleted features dataset with those from the whole features dataset reveals differences in Fig. 15.

The dropped of the top 20 features changed the accuracy of many classifiers. i.e., The accuracy of LR is 90.05 %, KNN have accuracy 95.59 %, DT 95.29 %, ET 96.67 %, RF 96.78 % and NN MLP achieved accuracy 95.50 % (see Fig. 16). Overall, the effectiveness of the classifiers varied depending on how the top 20 characteristics were removed. While some classifiers saw a noticeable decline in accuracy, others were less affected by the feature drop. These findings emphasize the significance of feature selection and show how various classifiers may differ in their reliance on particular characteristics to function at their best.

Table 2
Entropy-based feature selection and classification Results on all-features of dataset.

Classifiers	Accuracy	Recall	Precision	F1-Score	Training Time	Predicted Time	Total
LR	92.85 %	92.85 %	92.89 %	3.99 s	3.99 s	0.00 s	3.99s
KNN	95.04 %	95.04 %	95.04 %	95.05 %	0.01 s	18.34 s	18.3s
DT	96.40 %	96.40 %	96.40 %	96.40 %	1.12 s	0.00 s	1.12s
ET	97.53 %	97.53 %	97.53 %	97.53 %	6.09 s	0.31 s	6.39s
RF	**97.68 %**	**97.68 %**	**97.68 %**	**97.68 %**	**8.08 s**	**8.08 s**	**8.28s**
MLP	96.33 %	96.33 %	96.33 %	96.33 %	24.71 s	0.01 s	24.7s

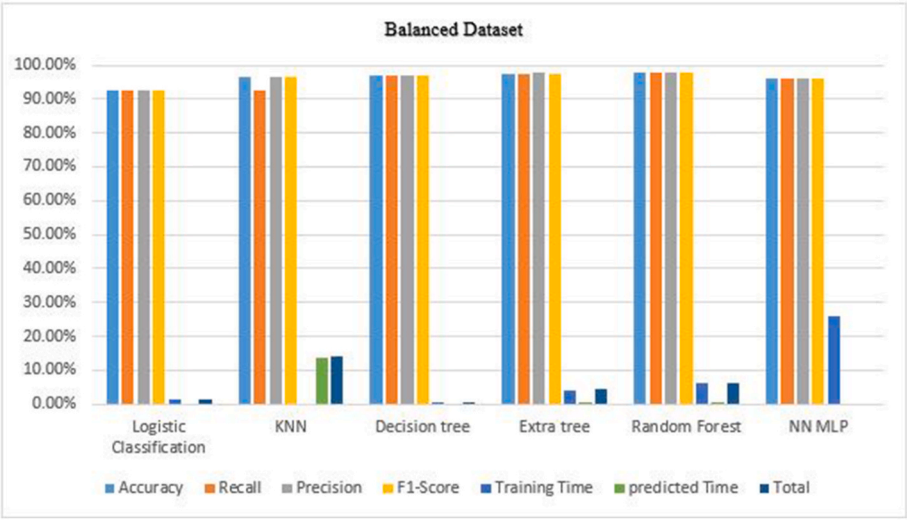


Fig. 11. Graphical comparison results using a balanced dataset.

Table 3
Comparison of different classifiers results after dropping 10 features.

Classifiers	Accuracy	Recall	Precision	F1-Score	Training Time	Predicted Time	Total
LR	92.46 %	92.46 %	92.46 %	92.46 %	1.2 s	0.00 s	1.2 s
KNN	96.01 %	96.01 %	96.01 %	96.01 %	0.0 s	14.3 s	14.4 s
DT	96.16 %	96.16 %	96.16 %	96.16 %	0.7 s	0.0 s	0.7 s
ET	97.15 %	97.15 %	97.15 %	97.15 %	3.8 s	0.3 s	4.2 s
RF	**97.35 %**	**97.35 %**	**97.35 %**	**97.35 %**	**5.8 s**	**0.2 s**	**6.0 s**
MLP	95.99 %	95.99 %	95.99 %	95.99 %	27.5 s	0.0	27.5

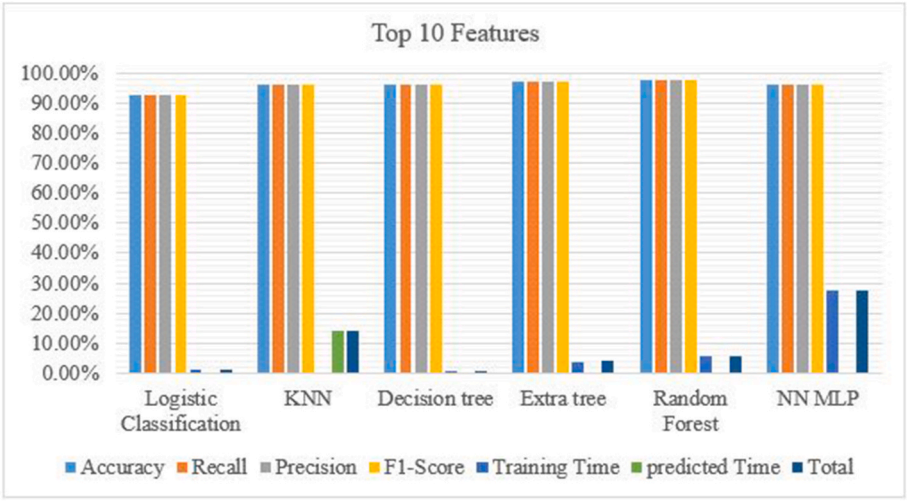


Fig. 12. Comparison of different classifiers results after dropping 10 features.

6.6. Drop random Feature

The test 4 made use of a dataset that contained all numerical values for each kind. We looked at the outcomes to see if test 4 accuracy improved by eliminating random features. Then, the conventional strategy and feature selection based on entropy applied. On a strong dataset, the Random Forest model produced the highest levels of categorization accuracy, recall, precision, and F1-Score, which are displayed in Table 5. In comparison to a balanced dataset, accuracy is higher and training time is lower for relevant data. Among the studied classifiers, the Random Forest model demonstrated exceptional performance on the robust dataset, reaching the

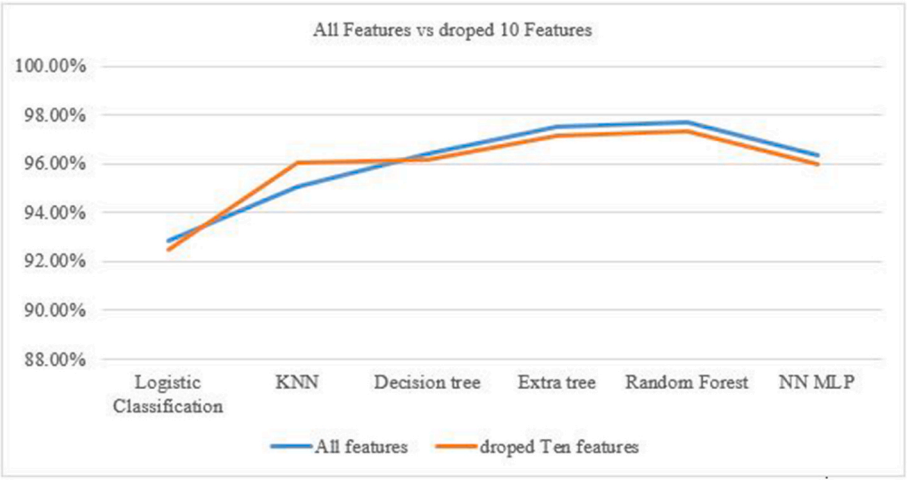


Fig. 13. Comparison graph of all features and dropped Top 10 Features.

Table 4
Comparison of different classifiers results after dropping 20 features.

Classifiers	Accuracy	Recall	Precision	F1-Score	Training Time	Predicted Time	Total Time
LR	90.05 %	90.05 %	90.04 %	90.05 %	1.0	0.0	1.0
KNN	95.59 %	95.59 %	95.59 %	95.59 %	0.0	14.1	14.1
DT	95.29 %	95.29 %	95.29 %	95.29 %	0.4	0.0	0.4
ET	96.67 %	96.67 %	96.67 %	96.67 %	3.1	0.3	3.5
RF	**96.78 %**	**96.78 %**	**96.77 %**	**96.78 %**	**4.4**	**0.2**	**4.6**
MLP	95.50 %	95.50 %	95.51 %	95.50 %	27.7	0.0	27.7

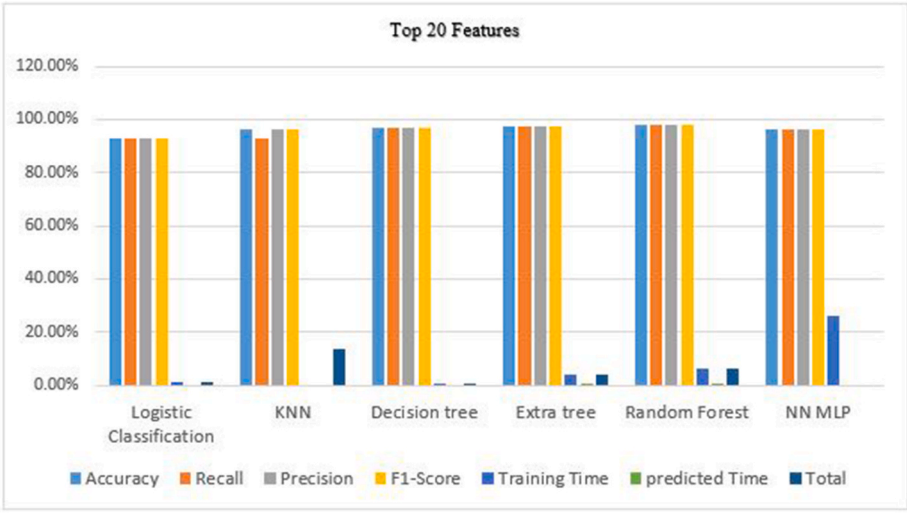


Fig. 14. Comparison of different classifiers results after dropping 20 features.

greatest levels of categorization accuracy, recall, precision, and F1-Score. The model's ability to handle important and difficult data further shown by the comparison to a balanced dataset, which showed greater accuracy and shorter training time.

The graphical representation of test 4 made use of a dataset that contained all numerical values for each kind. We looked at the outcomes to see if test 4 accuracy improved by eliminating random features. Then, the conventional strategy and feature selection based on entropy applied.

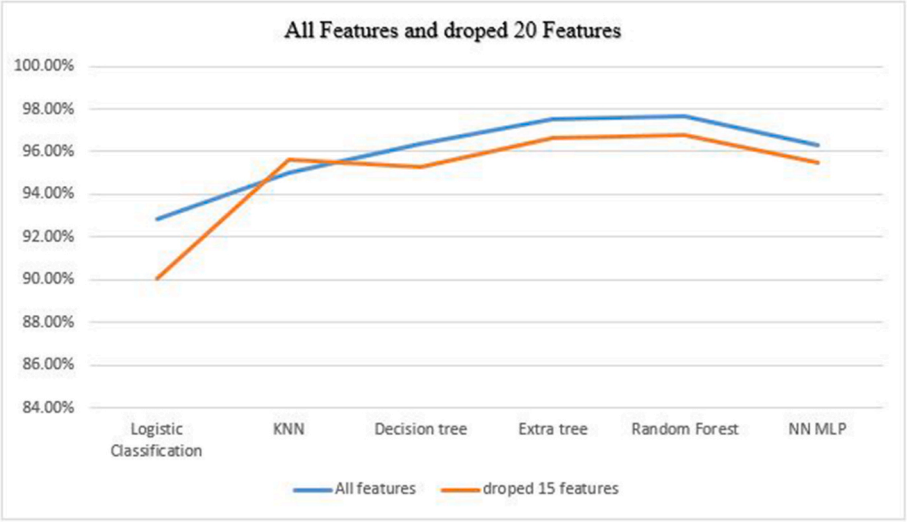


Fig. 15. Comparison of all features and dropped top 20 features.

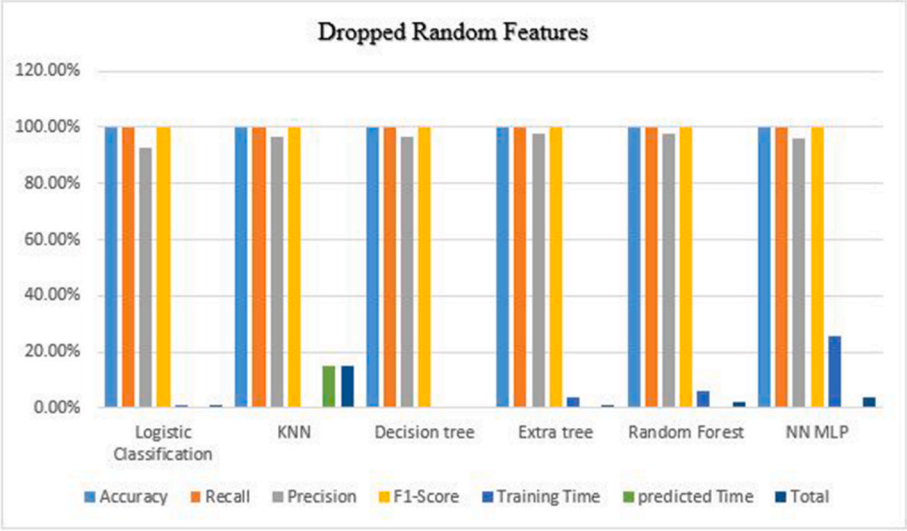


Fig. 16. Comparison of different classifiers results after dropping 20 features.

Table 5
Comparison of different classifiers results after dropping random features.

Classifiers	Accuracy	Recall	Precision	F1-Score	Training Time	Predicted Time	Total Time
LR	99.96 %	99.96 %	99.96 %	99.96 %	1.0	0.0	1.0
KNN	99.97 %	99.97 %	99.97 %	99.97 %	0.0	15.0	15.0
DT	99.95 %	99.95 %	99.95 %	99.95 %	0.2	0.0	0.2
ET	**99.98 %**	**99.98 %**	**99.98 %**	**99.98 %**	**1.2**	**0.1**	**1.3**
RF	99.96 %	99.96 %	99.96 %	99.96 %	2.0	0.1	2.2
MLP	99.96 %	99.96 %	99.96 %	99.96 %	4.0	0.0	4.1

6.7. Comparison between all features VS random features

The differences between the results obtained from the full features dataset and the random features dataset are shown in Fig. 17. The accuracy of all classifiers is significantly impacted by the addition of random features. The accuracy of the classifiers significantly increases by removing the random characteristics. Specifically, when the random characteristics are excluded from Logistic

Regression, the accuracy soars to a remarkable 99.96 %. This improvement suggests that the performance of the model may be adversely affected by the introduction of noise or irrelevant information caused by the addition of random characteristics. Like this, excluding the random characteristics significantly improves the accuracy of KNN. Although the statement does not provide the precise accuracy number, it may be assumed that the improvement is considerable, much like in the case of logistic regression.

6.8. A thorough comparison

Fig. 17 shows the findings from the study's initial phase. Extra Tree (ET) fared better than its competitors GBM and LR in binary classification, with an accuracy of 99.98 %. It had the potential to be far more accurate, in our opinion. However, there have been several attack types mentioned by the UNSWNB15 that could not be tested for since the information in the dataset could not be used to train for them. In either case, many of the algorithms we examined for the purposes of this study fell short of developing a model that could accurately and consistently identify them. The outcomes of the multi-classifiers in Table provide yet another illustration of this. To find the best effective multi-classifier to employ in combination with ET classifier, these findings are obtained independently from the first Stage classification.

Results from various dropped feature datasets, and all feature datasets are compared and provide distinct results in Table 6, as shown in above all figures. The accuracy of the original dataset is 97.68 % for the Random Forest classifier, and it increased to 99.98 % for the ET classifier after random features are excluded. In comparison to the original dataset, the accuracy has increased.

7. Discussion

The proposed study makes use of the freely accessible dataset UNSWNB15. We used the feature encoding approach in our proposed work to turn our dataset into simply numeric values. Then, for the optimal feature selection, we used the entropy-based feature selection technique named as TFIDF. After that, the dataset is balanced and presented to the ML models for classification. The prior study's comparison is mentioned in Table 7 utilizing ML Algorithms in Ransomware Classification and Detection [14] and reached a result of 96.79 %. The S.O wrapper-based technique [15] is utilized in Behavior-based ransomware classification and has 97.39 % accuracy. Malware Detection Behavioral Features [24] reached an accuracy of 96.75 % using Malware detection (EEXR). Dynamic API Calls [23], are utilized in Dynamic API Malware Detection techniques, with 99.1 % accuracy. After at all We employed feature encoding. The chosen characteristics are supported by RF, nnMLP, ET, KNN, LR, and DT classifiers to recognize malware. The proposed approaches beat earlier methodologies, reaching an accuracy score of 99.98 % on the ET classifier using a dataset of lost random features.

We attempted to work more efficiently, but we still have some restrictions, such as the need to use numerous datasets in our study. We could use several alternative classifiers. Various feature encoding approaches might be employed. Using this dataset, we recommend future study into Hidden Markov Models and Deep Learning approaches. The idea is to enable learning from our outcomes, which would add a new, crucial dimension to our prediction models.

8. Conclusion

Preprocessing, feature extraction, identification, and malware classification are the four main procedures that make up the technique for the detection and classification of malware that is being presented in this paper. We conclude that the identification of

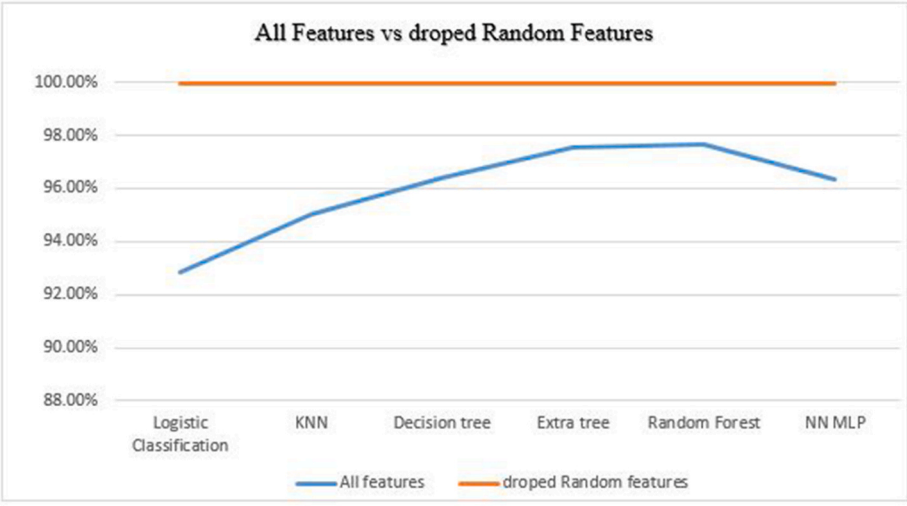


Fig. 17. Comparison graph all features vs dropped random features.

Table 6
Comparison of all results w.r.t Accuracy and Training Time.

All Features			Top 10 features		
Classifiers	**Accuracy**	**Training Time**	**Classifiers**	**Accuracy**	**Training Time**
LR	92.85 %	3.99 s	LR	92.46 %	1.2s
KNN	95.04 %	0.01 s	KNN	96.01 %	0.0s
DT	96.40 %	1.12 s	DT	96.16 %	0.7s
ET	97.53 %	6.09 s	ET	97.15 %	3.8s
RF	97.68 %	8.08 s	RF	97.35 %	5.8s
MLP	96.33 %	24.71 s	MLP	95.99 %	27.5s
Top 20 features			**Random features**		
Classifiers	**Accuracy**	**Training Time**	**Classifiers**	**Accuracy**	**Training Time**
LR	90.05 %	1.0s	LR	99.96 %	1.0s
KNN	95.59 %	0.0s	KNN	99.97 %	0.0s
DT	95.29 %	0.4s	DT	99.95 %	0.2s
ET	96.67 %	3.1s	**ET**	**99.98 %**	**1.2s**
RF	96.78 %	4.4s	RF	99.96 %	2.0s
MLP	95.50 %	27.7s	MLP	99.96 %	4.0s

Table 7
Prior Study's Comparison with the proposed work.

Study	Year	Accuracy
[14]	2022	96.79 %
[24]	2022	96.75 %
[23]	2022	99.1 %
[12]	2021	97.60 %
[25]	2021	99.97 %
[39]	2021	96.26 %
Proposed work	2023	**99.98 %**

malware is handled utilizing the proposed work's entropy-based feature selection using the entropy-based TFIDF approach from the discussion and results shown above. After all of that, we have dropped various features from our suggested model, including top 10, top 20, and lastly, random features. This is due to the size of our dataset, which contains both categorical and numeric values. As categorical and numerical values have an impact on the classification process, we used feature encoding. To identify malware, the chosen specific features are supported by RF, nnMLP, ET, KNN, LR, and DT classifiers. The suggested approaches outperformed previous techniques by achieving an accuracy score of 99.98 % on the ET classifier utilizing a dropped random features dataset. We advise further research into Hidden Markov Models and Deep Learning methods using this dataset. The goal is to enable learning from our results, which would give our prediction models a new, important dimension.

Ethics Approval

This article does not contain any studies with human participants performed by any of the authors.

Data availability

All data used and analyzed during this study is publicly available at following sources [36].

1. https://cloudstor.aarnet.edu.au/plus/index.php/s/2DhnLGDdEECo4ys
2. https://www.kaggle.com/datasets/dhoogla/unswnb15

Additional information

No additional information is available for this paper.

CRediT authorship contribution statement

Muhammad Azeem: Software, Project administration, Formal analysis, Data curation, Conceptualization. **Danish Khan:** Visualization, Validation, Methodology, Formal analysis, Data curation, Conceptualization. **Saman Iftikhar:** Writing - review & editing, Writing - original draft, Visualization, Validation, Supervision, Methodology, Investigation. **Shaikhan Bawazeer:** Resources, Project administration, Funding acquisition, Formal analysis, Data curation, Conceptualization. **Mohammed Alzahrani:** Writing - review &

editing, Writing - original draft, Methodology, Investigation, Conceptualization.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Shaikhan Bawazeer reports financial support was provided by Arab Open University. Shaikhan Bawazeer reports a relationship with Arab Open University, Saudi Arabia that includes: employment. If there are other authors, they declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgement

The authors extend their appreciation to the Arab Open University for funding this work through AOU research fund No. (AOUKSA-524008).

References

[1] K.D. Anil, S.K. Das, NB2M–Mechanism for Magnifying Micro Level Bugs for Secure Software System 3 (2021) 26–33.
[2] P. Parmuval, M. Hasan, S. Patel, "Malware family detection approach using image processing techniques: visualization technique," 7 (20) (2018) 129–132.
[3] R. Kumar, Z. Xiaosong, R.U. Khan, I. Ahad, J. Kumar, Malicious code detection based on image processing using deep learning, in: Proceedings of the 2018 International Conference on Computing and Artificial Intelligence, 2018, pp. 81–85.
[4] N.Y. Kim, S. Rathore, J.H. Ryu, J.H. Park, J.H. Park, "A survey on cyber physical system security for IoT: issues, challenges, threats, solutions," 14 (6) (2018) 1361–1384.
[5] P. S. Ram, A. S. Harsha, E. U. Shankari, and N. K. Rao, "Detection Of Malware Using Signature Based Algorithm Undergoing Database Verification." .
[6] D. Gibert Llauradó, C. Mateu Piñol, J. Planes Cid, R. Vicens, Using convolutional neural networks for classification of malware represented as images, Hacking Techniques 15 (1) (2019) 15–28.
[7] I. Ideses, A. Neuberger, "Adware detection and privacy control in mobile devices,", in: 2014 IEEE 28th Convention of Electrical & Electronics Engineers in Israel (IEEEI), IEEE, 2014, pp. 1–5.
[8] R. Jain, R. Bhatnagar, Applications of machine learning in cyber security-A review and a conceptual framework for a university setup, in: *International Conference on Advanced* Machine Learning *Technologies and Applications*, Springer, 2019, pp. 599–608.
[9] Y. Li, R. Ma, R.J. Jiao, I. Applications, A hybrid malicious code detection method based on deep learning 9 (5) (2015) 205–216.
[10] J. Martínez Torres, C. Iglesias Comesaña, P.J. García-Nieto, Cybernetics, machine learning techniques applied to cybersecurity 10 (10) (2019) 2823–2836.
[11] D. Du, Y. Sun, Y. Ma, F.J.I.A. Xiao, A novel approach to detect malware variants based on classified behaviors 7 (2019) 81770–81782.
[12] A. Alshamrani, S. Myneni, A. Chowdhary, D. Huang, A survey on advanced persistent threats: techniques, solutions, challenges, and research opportunities, IEEE Communications Surveys & Tutorials 21 (2) (2019) 1851–1877.
[13] Ö. Aslan, A.A. Yilmaz, A new malware classification framework based on deep learning algorithms, IEEE Access 9 (2021) 87936–87951.
[14] M. Masum, M.J.H. Faruk, H. Shahriar, K. Qian, D. Lo, M.I. Adnan, Ransomware classification and detection with machine learning algorithms, in: 2022 IEEE 12th Annual Computing and Communication Workshop and Conference (CCWC), IEEE, 2022, pp. 316–322.
[15] M.S. Abbasi, H. Al-Sahaf, M. Mansoori, I.J.A.S.C. Welch, Behavior-based Ransomware Classification: A Particle Swarm Optimisation Wrapper-Based Approach for Feature Selection, 2022, 108744.
[16] A. Arfeen, Z.A. Khan, R. Uddin, U. Ahsan, Toward accurate and intelligent detection of malware, Concurrency Comput. Pract. Ex. 34 (4) (2022) e6652.
[17] I. Contreras, J.I. Hidalgo, L. Nuñez, Exploring the influence of industries and randomness in stock prices, Empir. Econ. 55 (2) (2018) 713–729.
[18] D. Gibert Llauradó, C. Mateu Piñol, J. Planes Cid, R. Vicens, Using convolutional neural networks for classification of malware represented as images, Journal of Computer Virology and Hacking Techniques 15 (1) (2019) 15–28, 2019.
[19] A. Abusitta, M.Q. Li, B.C. Fung, Malware classification and composition analysis: a survey of recent developments, J. Inf. Secur. Appl. 59 (2021), 102828.
[20] H.S. Anderson, P. Roth, Ember: an open dataset for training static pe malware machine learning models, arXiv preprint arXiv:1804.04637 (2018).
[21] M. Nisa, et al., Hybrid malware classification method using segmentation-based fractal texture analysis and deep convolution neural network features 10 (14) (2020) 4966.
[22] Y. Qiao, Q. Jiang, Z. Jiang, L. Gu, A multi-channel visualization method for malware classification based on deep learning, in: 2019 18th IEEE International Conference on Trust, Security and Privacy in Computing and Communications/13th IEEE International Conference on Big Data Science and Engineering (TrustCom/BigDataSE), IEEE, 2019, pp. 757–762.
[23] J. Singh, J. Singh, Assessment of supervised machine learning algorithms using dynamic API calls for malware detection, Int. J. Comput. Appl. 44 (3) (2022) 270–277.
[24] T. Urmila, Machine learning -based malware detection on Android devices using behavioral features,", Mater. Today: Proc. (2022).
[25] A. Pinhero, et al., Malware detection employed by visualization and deep neural network, Comput. Secur. 105 (2021), 102247.
[26] B.J.C. Dupont, law and s. change, "Bots, cops, and corporations: on the limits of enforcement and the promise of polycentric regulation as a way to control large-scale cybercrime," 67 (2017) 97–116.
[27] A. Czeskis, A. Moshchuk, T. Kohno, H.J. Wang, Lightweight server support for browser-based CSRF protection, in: Proceedings of the 22nd International Conference on World Wide Web, 2013, pp. 273–284.
[28] Y. Aafer, W. Du, H. Yin, Droidapiminer: Mining api-level features for robust malware detection in android, in: International Conference on Security and Privacy in Communication Systems, Springer, 2013, pp. 86–103.
[29] A. Sarker, A.C. Canto, M.M. Kermani, R. Azarderakhsh, Systems, Error Detection Architectures for Hardware/Software Co-design Approaches of Number-Theoretic Transform, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2023.
[30] Mozaffari Kermani M., Azarderakhsh R., Integrating Emerging Cryptographic Engineering Research and Security Education, published in Computer Science, Education, Engineering, 2015.
[31] M. Mozaffari-Kermani, Reliable and High-Performance Hardware Architectures for the Advanced Encryption Standard/Galois Counter Mode, The University of Western Ontario (Canada), 2011.
[32] A.C. Canto, J. Kaur, M.M. Kermani, R.J.a. p. a. Azarderakhsh, Algorithmic Security Is Insufficient: A Comprehensive Survey on Implementation Attacks Haunting Post-Quantum Security, 2023.
[33] J. Kaur, A.C. Canto, M.M. Kermani, R.J.a. p. a. Azarderakhsh, A Comprehensive Survey on the Implementations, Attacks, and Countermeasures of the Current NIST Lightweight Cryptography Standard, 2023.
[34] M.M. Kermani, R. Azarderakhsh, J. Xie, "Error detection reliable architectures of Camellia block cipher applicable to different variants of its substitution boxes,", in: 2016 IEEE Asian Hardware-Oriented Security and Trust (AsianHOST), IEEE, 2016, pp. 1–6.
[35] M.M. Kermani, E. Savas, S. Upadhyaya, Guest editorial: introduction to the special issue on emerging security trends for deeply-embedded computing systems, Computer Science, Engineering 4 (3) (2016) 318–320.

[36] N. Moustafa, J. Slay, The significant features of the UNSW-NB15 and the KDD99 data sets for network intrusion detection systems, in: 2015 4th International Workshop on Building Analysis Datasets and Gathering Experience Returns for Security (BADGERS), IEEE, 2015, pp. 25–31.
[37] M.K. Dahouda, I.J.I.A. Joe, "A deep-learned embedding technique for categorical features encoding," 9 (2021) 114381–114391.
[38] M. Alhawarat, A.O.J.I.A. Aseeri, "A superior Arabic text categorization deep model (SATCDM)," 8 (2020) 24653–24661.
[39] N. Usman, et al., Intelligent dynamic malware detection using machine learning in IP reputation for forensics data analytics, Future Generat. Comput. Syst. 118 (2021) 124–141.