**Hybrid Machine Learning and Deep Learning Approach for Advanced Malware Detection**

**Abstract**

Attackers of malicious software (also known as malware) now have a veritable field to work in owing to the massive growth of the Internet and the recent developments in automation that involve the use of smart applications. These outcomes in expanding malware attacks and privacy difficulties were developed because of a broad variety of gadgets that were easily connected over the Internet. In addition, large amounts of data were generated. Even though there are many different virus detecting systems on the market, new techniques are required to accommodate the scale and intricacy of quite a data-intensive environment. It is possible that the effectiveness of individual machine learning techniques for the diagnosis of malware would differ based on the compliance of their classifiers, even while considering the utilisation of an appropriate learning dataset. The reduction of false-positive rates is significantly influenced by the application of appropriate algorithms. This hybrid model makes use of Convolutional Neural Networks (CNN), Long Short-Term Memory (LSTM), and Principal Component Analysis (PCA) to differentiate between files that might be hazardous.

***Keywords:*** Malware, CNN, LSTM, PCA, Hybrid Detection, Image Analysis, Machine Learning

1. **Introduction**

Malware is a type of program that is created with the intention of gaining unauthorized entry to computer devices for the profit of a 3rd person without the permission of the operator. An intrusive piece of program is a programme that is installed on machines and computer systems with the intention of causing destruction and harm. It is an abbreviation for dangerous software, which includes the most common types of malware such as worms, spyware, keyloggers, Trojan viruses, pathogens, and adware [1]. Memory evaluation is another name for the field of memory forensics. It is the process of analysing dynamic data that has been saved in the storage disposal site of a system. It is utilized to analyze and detect attacks or malicious activity that does not leave any hard drive data and traces [2].

Malware changes its shape so that it can't be found by systems that are set up to find malware based on certain features. Using the network traffic, start figuring out which malicious apps are on Android and put them in the "malicious" category [3]. Supervised machine learning methods are used to identify malware programs. In this model, pre-existing datasets that classify dangerous and benign applications based on characteristics are utilized to construct features like byte volume can identify denial of service attacks, Domain Name System (DNS) query timings can detect network overflooding, and so on [4] [5]. These characteristics are used to develop a model with the greatest degree of accuracy by turning the classifier parameters to the greatest [6]. There are many Malwares such as viruses, rootkits, logic bombs, spyware, trojan horses, worms, keyloggers ransomware, and backdoors.

1. **Types of Advance Malware**

Computer users and network administrators must now defend and protect themselves against an always-growing number of attack techniques that target networks, spread randomly, and computers in a massive attack. There is various type of malware which are given below [7]:

* **Viruses:** A computer virus is a programme that is developed to copy on its own and propagate to other system files, typically by connecting itself to existing program files.
* **Worms:** A computer worm is infecting other computers, but it is not spread by infecting new files.
* **Spyware:** Spyware is software that collects data on organizations and people without the individual's or organization's knowledge or consent.
* **Trojan horse:** A software that seems to be legitimate and even helpful, and therefore dupes people into installing/using it.
* **Ransomware:** Encrypting data on a computer and leaving a notice stating that a specific ransom must be paid for the decryption key to be revealed.
* **Backdoor:** A software that installs itself in such a manner that the affected machine is controlled and remotely accessed.
* **Rootkit:** Rootkits are pieces of software that are utilized to conceal files, Registry entries, running processes, or other types of information.
* **Key logger:** Systems that record the keys pressed on a keyboard and (typically) transmit this information to a third party [8].

1. **Types of Malware Attacks**

An autonomous, self-replicating piece of malware is being used by the intruder in an effort to spread to as numerous machines as possible. Malware that spreads itself is used because direct contamination of a small number of significant control hubs can be possible, but manually attack is not as effective for huge smart grids. This is one of the reasons why malware that spreads itself is used. The three varieties of malware that were researched vary from straightforward brute-force types to sophisticated covert types. Adjusting their names so that it correspond with the terminology employed by the Centers for Disease Control (CDC) and Protection. Malware can be divided into three categories: endemic malware, contagion malware, and endemic malware [9].

1. **Pandemic Malware**

This very active form of malware combines elements of both Code Red 1 and 2, with the purpose of infecting the whole system in a short period of time. The name circulation of CDC is used by investigators to describe this violent, rapid, and wide-reaching malware kind. In order to avoid detection, pandemic malware searches vigorously, rapidly spreads to new victims, and employs a basic monomorphic self-carrying payload lacking any recognition or obfuscation techniques. If the system is being watched, this form of malicious software can open unwanted TCP ports to other field nodes, which should be easy to spot if the network is being examined. Unrequested TCP links among field nodes are not allowed and do not reflect the behaviour of genuine programs. This pandemic malware is equipped to limit its scanning method to the sub-network by gathering data about the network that is readily accessible from the server [[[1]](#footnote-1)].

1. **Endemic Malware**

The term endemic malware refers to the second kind of malware, which acts more covertly than the first model, pandemic malware, which was presented in. The constant persistence that this malware shows in a group of users suggests that a parallel should be drawn between its behaviour and that of illnesses, and that the nomenclature practise for endemic malware should be taken from the CDC. It is important to point out that although Stuxnet has many of the same characteristics of endemic malware, it does not make use of hit lists. Because of this, it has been excluded from the list of example malware. [[[2]](#footnote-2)].

All endemic types of malware share an improved ability for a variety of concealment, spying, and attack characteristics that are implemented as modular expansions. To provide a high-level summary, endemic malware presents security with a bigger task, despite the fact that it is detected in networks that are carefully managed.

Their screening technique makes usage streamlined and improved iteration hit list, as outlined in. Nodes that have already been scanned are removed from the hit list, and the set is then sent to child malware. The malicious child programme starts a fresh random scan inside a smaller search region. With this method, repeated inspections of the same networks are kept to a minimum, and both suspected and traceable network activity is cut down significantly. Additionally, this sort of malware is able to get additional data from a host that it has infected, such as the size of the subnetwork and a list of hosts that already reside by way of OLSR data [[[3]](#footnote-3)].

1. **Contagion Malware**

he next category of malicious software that is known as contagion malware. Malware that spreads contagion uses a mechanism called passive screening and avoids performing any proactive inspection at all. Transmission Control Protocol (TCP) links between non-infected virtual hosts and afflicted hosts serve as both a sign of the presence of a new victim and as a data transmission stream for inoculating the new victim. In other words, TCP linkages are used to do both of these things. The malicious software that spreads the infection transmits its payload over the preexisting transmission channel just in time for the remote programme to shut its TCP connection, at which point the victim is infected [[[4]](#footnote-4)].

Therefore, to insert and deliver its payload, malware that spreads contagion makes use of legal TCP flows that already occur on infected systems and software flaws in the destination. The most significant shortcoming of this approach is that the virus relies on the presence of susceptible apps on both the host network and the target machine in order to successfully create TCP flows. Malware that spreads by contagion has to wait for client computers to connect with each other prior it can attack surrounding nodes. This can cause in significant delays in the malware's ability to spread, but it has the advantage of producing far less anomalous data [[[5]](#footnote-5)].

1. **Hybrid Malware Detection Systems**

There have been several studies in the research area that have argued in favour of developing a hybrid system by combining static evaluation and dynamic evaluation. The combination of static and dynamic features into a unified characteristic vector, which is subsequently subjected to analysis by a separate categorization algorithm, is the approach that is utilised the majority of the time [10]. The use of a single feature vector containing both static and dynamic information which include the Printable Strings Information (PSI) provided in the executable file is used to get static features. The dynamic analysis is carried out in the cuckoo sandbox [11] and generates a series of system calls represented as n-grams utilizing three- and four-API-call-grams. A similar solution is suggested which employs static information such as the Op-codes of the disassembled executable file [12] and dynamic ones such as the list of API calls, their arguments, and raised exceptions [13].

1. **Static Analysis**

The Static Analysis subsystem is built on a two-phase training deep neural network, with an unsupervised pre-training phase using stacked denoising autoencoders [14] guided by supervised fine-tuning through backpropagation.

The features are extracted via accessing the Portable Executable (PE) packaging's Disk Operating System (DOS) Header, Section Table, Optional Header, and File Header. A compact description of the field information is generated by treating simple numbers as unsigned integers. Other types of input, like as timestamps, arrays, and texts, are handled using a hashing algorithm. The use of offset variables is necessary to preserve any spatial knowledge connected to the elements [15]. For instance, the local offset gives the position of a particular field within the header, but the global offset indicates the relocation from the start of the file. Both offsets are referred to as the dislocation [16].

1. **Dynamic Analysis**

It is not difficult to incorporate other dynamic evaluation methods without having to make any changes to the system's design or the reasoning that it uses to function, because of the general adaptability of the platform [17]. The cuckoo sandbox enables the real-time running of possibly infected code, resulting in a series of log files that describe the executable file's activity during execution [18]. The application has the capability of being programmed to produce a wide variety of data types, such as a list of API calls, each of which has its own individual group of arguments, register changes, heap storage addresses, process addresses, network use, and so on. The 3-grams and 4-grams techniques are what the sequence of API calls reflect when it comes to the dynamic properties of the system [19].

In this research, a type of machine learning called "hybrid machine learning" is used, which works better and is more accurate than "classical machine learning." In this hybrid model, CNN with LSTM and classifiers SVM, Random Forest, and Multilayer perceptron (MLP) are used to get better results. The voting scheme for the predicted data makes this work different from others.

1. **Literature Review**

Various research has been done by the researchers. Different models have been proposed by the researchers to avoid the turbulence due to the atmosphere. Various papers and models have been studied for this paper which is included in this literature review.

Faysal et al., (2022) [20] developed the XGBoost-Random Forest intrusion detection technique, a hybrid machine learning approach. The hybrid technique suggested here was used for the N-BaIoT dataset, which contains malicious botnet attacks. It utilized a random forest (RF) to collect features and an eXtreme Gradient Boosting (XGB) classifier to identify various forms of attacks on IoT settings. The presented XGB-RF scheme's performance is tested using a variety of evaluation criteria and indicates that the model correctly identifies 99.94 percent of attacks.

Sihwail et al., (2021) [21] observed that paying careful attention to virus behavior results in minimizing malware risks. The investigating experiment is expending either static investigation or behavioral investigation. Current studies have shown that modern malware files employ some methods to avoid detection and analysis. The unpredictable memory is beneficial to intelligence nearby a malware's activities or attributes. Memory analysis is used to detect unique malware, such as fileless malware and in-memory. The author developed a technique for malware identification and categorization that removes the memory-based elements.

Yoo et al., (2021) [22] developed a machine learning-based hybrid decision model with a high detection rate and a low false-positive rate. This hybrid model combines a random forest with a deep learning model with 12 hidden layers to discriminate between dangerous and benign data. Certain classifiers acquire a high detection rate when trained on a malicious dataset but a low detection rate when trained on a benign dataset, and false-positive rates are greatly dependent on the classifiers used.

Mohammed et al., (2020) [23] addressed DL-Droid, a deep learning system that uses dynamic analysis and stateful input creation to identify fraudulent Android apps. Experiments were done on actual devices with over 30,000 apps (both benign and malicious) to evaluate the identification performance and test coverage of the stateful input generation technique to the more widely utilized stateless technique employing a deep learning system.

Sanket et al., (2019) [24] design a two-pronged approach for efficiently detecting both traditional and stealthy malware such as extracting microarchitectural traces obtained during application execution and feeding them to traditional machine learning classifiers to recognize malware spawned as different threads, and efficiently detecting stealthy malware and initiate an automated localized feature extraction technique that can be used as an input to recurrent neural networks (RNN).

Vinaya et al., (2019) [25] suggested a unique image processing approach with optimized parameters for MLAs and deep learning architectures. A thorough experimental examination of these approaches reveals that deep learning architectures outperform traditional MLAs. The feature engineering process may be removed by employing sophisticated MLAs like as deep learning. Despite several recent research efforts in this area, the performance of the algorithms is influenced by the training data. This is a need to reduce bias and independently analyze these approaches in order to develop new improved ways for successful zero-day malware detection.

Sitalakshmi et al., (2019) [26] highlighted the use of image-based methods for identifying the suspicious activity of systems, as well as a study on using hybrid image-based approaches with neural network architectures for malware classification. According to several similarity measurements of malware activity patterns and cost-sensitive deep learning architectures, the system performance is evaluated.

According to De Paola et al., (2018) [27], hybrid cloud-based malware detection systems are being developed in which static and dynamic analyses are merged in order to establish a suitable trade-off between reaction time and detection accuracy while maintaining a high level of detection accuracy. A constant learning process of its models, which is based on deep networks, is carried out by this system, which makes use of the expanding quantity of data given by customers. The early experimental assessment of the method reveals that it is appropriate for the situation. Table 1 below describes the summary of the related work being discussed.

Table 1: Summary of Related Work

|  |  |  |  |
| --- | --- | --- | --- |
| **Author** | **Technique** | **Outcome** | **Research Gap** |
| Faysal et al., [20] (2022) | eXtreme Gradient Boosting and Random Forest (XGB-RF) | Enabled to successfully identify botnet assaults and can lessen privacy concerns related to IoT equipment. | The bulk of IoT equipment lacking the storage and computational power required for proper security services. |
| Sihwail et al., [21] (2021) | A method for detecting and classifying malware that uses memory-based characteristics retrieved from memory pictures and an SVM classifier. | high rates of false positives and categorization reliability. | Static or behaviour analysts are susceptible to current malware documents that employ a variety of tactics to circumvent study and identification, rendering them ineffective. |
| Yoo et al., [22] (2021) | Deep learning model and supervised model | The model explains malware categorization for zero-day attacks and can be widely used. | It's possible to get high identification percentages using harmful training datasets using individual machine learning algorithms for malware identification. |
| Alzaylaee et al., [23] (2020) | DL-Droid, a deep learning system | Detection rates of up to 97.8% and 99.6% are possible with dynamic and static characteristics | Android malware obfuscation and identification avoidance technologies have evolved, leaving many previous approaches outdated. |
| Shukla et al., [24] (2019) | HPC-based approach with ML classifier utilizing RNN. | Effective at identifying both stealthy malware and regular malware. | Machine learning, computer vision, and deep learning were used to create complex malware via code displacement and inversion. |
| Vinaya Kumar et al., [25] (2019) | Machine learning and deep learning based on Static Analysis | Improve the time consumption period and model flexibility. | Malware identification methods that analyse malware signs and activity records are time-consuming and poor at recognizing unidentified malwares in real-time. |
| Venkatraman et al., [26] (2019) | Novel and unified hybrid deep learning and visualization approach. | When compared to traditional machine learning, it required less computing expense. | With more devices linked to the Internet and more data generated, malware assaults and privacy vulnerabilities are rising. |
| De Paola et al., (2018) [27] | Hybrid cloud-based malware detection system. | Enables clients to inspect active documents even while they're not connected. | Enabled for the development of inexpensive, unobtrusive gadgets that can be used for instrumenting a smart environment. |

1. **Research Objective**

* To design an efficient feature selection algorithm that can work on polymorphic metamorphic and obfuscated malware.
* To design algorithms that can dynamically detect advance malware in large-scale data and categorize it to a specific family for further analysis.
* To propose a framework that has a high performance for detecting unknown malware (zero-day) and detecting the malware effectively with improved accuracy.

1. **Problem Formulation**

Malicious software, often known as malware, is targeting an increasing range of system users, enterprises, and even the government, which creates a significant vulnerability in terms of data security. Static and dynamic virus signatures and behaviour trends are now utilised to recognize unidentified malware; however, this strategy requires a great deal of time and is not particularly effective at recognizing new types of malware. New software uses polymorphic, metamorphic, and other simpler methods to easily change how such an infection acts and make new members of the virus's family.

Advanced MLAs, like as deep learning, may eliminate the need for the engineering step. Research has recently pointed in this direction, although ultimately the algorithm's success is determined by the training data. It does not eliminate bias or analyze these approaches individually to arrive at a new improved strategy for successful malware classification and detection. In the current work, the scalable dynamic hybrid framework is presented which has the capability to prove a large number of malware samples on a real-time basis. The current work is presented as two-stage algorithms where in the first stage the files are classified as malware or legitimate using dynamic analysis and traditional classifier with Ada boost to enhance the performance of the individual classifier. In the second stage malware files are categorized into corresponding malware faulty using image processing and CNN LSTM.

1. **Research Methodology**

Convolutional neural networks (CNNs), Long Short-Term Memories (LSTMs), and Principal Component Analysis (PCAs) are some of the technique tools which are used in this entire research study.

1. **Convolutional Neural Network**

The CNN is Neural Network (NN’s) form that concentrates on processing data with a grid-like topology, such as an image. The CNN’s architecture is illustrated in Figure 1 [28].

Diagram

Description automatically generated with medium confidence

Figure 1: The Architecture of CNN

CNNs are neural networks that involve one or more convolutional layers and are mainly applied for image, segmentation, classification, and other auto-correlated data processing. The technique of sliding a filter across an input signal is known as convolution [29]. Looking at a function's surroundings to make better or more accurate predictions of its outcome [30]. Convolution can be supposed to recognize features in an image, slighter sections of the image are more precise than looking at the whole image at once. CNNs are also compatible with more general functional signal processing and image segmentation.

U Net is a Convolutional Neural Network design that has increased in size despite relatively minimal modifications to the CNN architecture. It was developed to deal with biological images where the objective is not just to classify whether or not there is an infection, but also to identify the illness's location.

(1)

(2)

where Zj denotes the convolution operation's output, Xi denotes the convolutional layer's input, Ki j denotes the convolution kernel, and Bj denotes the additive bias. In the following equation, Aj is the convolutional layer's output feature map, and f is an activation function.

1. **Long Short Term Memory**

The term Long Short Term Memory (LSTM) was introduced [31] to reconcile gradients that are disappearing or bursting in a recurrent neural network. The LSTM is equipped with an internal memory cell that is accessed by forgetting and input gate networks. In an LSTM layer, a forget gate controls how often memory should be transferred into the following time step. An input gate, on the other hand, scales fresh input to memory cells. LSTM may represent either long-term or short-term reliance on sequential data, depending on the states of both gates [32]. The following is the LSTM formulation:

(3)

(4)

(5)

(6)

() (7)

(8)

 denotes the layer index, while denote the input, forget, and output gates, respectively. These are successively multiplied by the input, memory cell, and hidden output to gradually open or close their interconnections. represents input from the layer, represents the output layer at time , and represents the internal cell state at time . is a projection matrix that is used to lower the dimension of .

1. **Principal Component Analysis (PCA)**

This technique is used to identify real-time malware and determine the non-trivial microarchitectural events that can be collected by a low number of High-Performance Computing (HPCs) and give high detection performance. Using PCA, identify the most important events and monitor them throughout the experiment [33]. The following is how the competitions are ranked:

Where is the Pearson correlation coefficient of any application. is any incoming application. is an output data containing different classes. measures covariance between input and output and measure the variance of both input and output respectively [34].

1. **Research Methodology**

In this methodology detection of malware, the dataset is processed by the different methods using CNN, LSTM and PCA as shown below in figure 2.

Hybrid CNN and LSTM technique is used for the image dataset and PCA technique is used for the feature extraction of the textual dataset.

In the training process malware, the dataset is divided into two parts: Binary grayscale conversion and Dynamic Feature extraction.

Step 1.1: In this process, the selected binary file is transformed into a grayscale picture, from which the pattern for 8 bits is extracted.

Step 1.2: To detect picture patterns, raster scanning is done on transformed binary images.

Step 1.3: Pattern is generated and labeled according to the need of the dataset and gives the pattern of 32x32 block size.

Step 1.4: For image processing CNN and LSTM method, CNN is used to extract the feature from each filter and are grouped into a new feature set called a feature map.

LSTM method captures the sequence-related information and passed it into a fully connected layer for classification which predicts the label of the dataset (x).

Diagram

Description automatically generated

Figure 2: Proposed Methodology

Step 2.1: In this Dynamic extraction of features is processed which properly optimized feature extraction to affect the model construction.

Step 2.2: A feature ranking algorithm is used to extract the features and ranked the top 20 dynamic features using the information gained.

Step 2.3: After the extraction of features/events, Principal Component Analysis (PCA) is used for the reduction of microarchitectural events traces which are captured during offline execution of the application.

Step 2.4: Once the dataset has been analysed, classifiers such as SVM, RF, MLP, and KNN are used to categorise the malware.

Step 2.5: To combine all these techniques and to select the best classifier among them AdaBoost process is applied, this algorithm improves the overall accuracy of the new combined classifier and predicts the label of the dataset (y).

Step 3: Both predicted labels (x) and label (y) are processed for the voting scheme.

In the testing process, the featured data files are logged into IoT devices.

Step 4: Log IoT devices are processed for both feature extraction and binary conversion, in which ranking of the file is extracted and ML approaches the executable file and then classifies the unknown malware samples into trained malware groups.

1. **Result and Analysis**

This section shows the results and analysis of the techniques used in the methodology. The python tool is used in the implementation of the proposed methodology. All the results of the work are described one by one in detail:

**Result 1:** Figure 3 shows that the dataset read which has trained as given below.

A picture containing text

Description automatically generated

Figure 3: Dataset read

**Result 2:** Figure 4 depict that the list of malware such as benign, family 1 malware, family 2 malware, and family 3 malware attributes are classified given below.

Text

Description automatically generated with low confidence

Figure 4: Classify dataset

**Result 3:** Figure 5 demonstrated the Benign, Family 1 Malware, Family 2 Malware, and Family 3 Malware which has shown with their number of images as given below.

A screenshot of a computer

Description automatically generated with medium confidence

Figure 5: No. of count images of various malware

**Result 4:** Figure 6 shows the graph representation which analyzing the no. of images count in the benign, family Malware 1, Family Malware 2, and Family malware 3 as given below.

Chart, bar chart

Description automatically generated

Figure 6: Graph of malwares.

**Result 5:** Figure 7 displays the image of the training dataset following the conversion of grey scale as given below.

Graphical user interface, text

Description automatically generated

Figure 7: Grey scale conversion.

**Result 6:** Figure 8 demonstrates the raster image which has created by scanning of training dataset as shown given below.

Graphical user interface, chart

Description automatically generated

Figure 8: Raster scanning of dataset

**Result 7:** Figure 9 demonstrates the pattern generating process using the k-mean random approach with several datasets as shown given below.

Graphical user interface

Description automatically generated

Figure 9: Pattern generation using K-means technique

**Result 8:** Figure 10 shows the training accuracy and validate accuracy by using CNN model in terms of precision and failure. The accuracy of both training and validate is defined in terms of number of epochs. Training accuracy is increasing continuously while validate accuracy goes to down as given below.

Graphical user interface

Description automatically generated

Figure 10: Accuracy of training and validation.

**Result 9:** Figure 11 demonstrate that score of model which is evaluated in terms of train score and train accuracy. the train score is 0.084389 and train accuracy is 0.961738 of trained data set that is given below.

A screenshot of a computer

Description automatically generated with medium confidence

Figure 11: Training dataset.

**Result 10:** Figure 12 demonstrate that score of model which is evaluated in terms of test score and test accuracy. the test score is 1.8698138 and train accuracy is 0.6392353 of trained data set that is shown below.

Text

Description automatically generated

Figure 12: Accuracy on testing time

**Result 11:** Figure 13 shows the classification reports for several characteristics by using the CNN model for Benign, Family 1 Malware, Family 2 Malware, and Family 3 Malware that is given below.

Calendar

Description automatically generated

Figure 13: Classification report of CNN model

**Result 12:** Figure 14 shows a picture that is categorized from malware of the first family, as predicted by the family approach and the result is displayed on cartesian plane that is shown below.

Graphical user interface

Description automatically generated

Figure 14: Prediction of the family

**Result 13:** Figure 15 demonstrated the complete global feature extraction of different dataset such as family 1 Malware, Family 2 Malware, and Family 3 Malware that is shown below.

**Text

Description automatically generated**

Figure 15: Complete global feature extraction of dataset

**Result 14:** Figure 16 demonstrated the value of trained transform dataset and shape of n components with the help of Principal component analysis (PCA) and the result shows that the value of train transform data is 5410, and the value of train transform data shape is 5 that is given below.

**Graphical user interface, text, website

Description automatically generated with medium confidence**

Figure 16 Trained transform dataset and shape by PCA

**Result 15:** Figure 17 shows the classification report of different training dataset based on different factors such as precision, recall, f1-score, and support with the help of Random Forest classifier that is given below.

**A picture containing calendar

Description automatically generated**

Figure 17 Classification based on RF

**Result 16:** Figure 17 presents the classification report of a variety of training datasets based on various parameters including precision, recall, f1-score, and support. This report was generated with the assistance of the KNN classifier, which can be found below.

**Calendar

Description automatically generated with low confidence**

Figure 18 Classification based on KNN

**Result 17:** Figure 17 shows the classification report of different training dataset based on different factors such as precision, recall, f1-score, and support with the help of SVM classifier that is given below.

**A picture containing graphical user interface

Description automatically generated**

Figure 19 Classification based on SVM

**Result 18:** Figure 20 demonstrated the classification report of a variety of training datasets based on various factors including precision, recall, f1-score, and support. This report was generated with the assistance of the MLP classifier, which can be found below.

**Calendar

Description automatically generated**

Figure 20 Classification based on MLP

**Result 19:** Figure 21 shows the classification report for a number of training datasets based on different parameters, such as precision, recall, f1-score, and support. This report was made with the help of an ADA Booster classifier, which you can see below.

**Table

Description automatically generated with low confidence**

Figure 21 Classification based on ADA Booster

**Result 20:** Figure 22 provided a demonstration of the classification report for a variety of datasets based on factors such as precision, recall, f1-score, and support. This was accomplished with the assistance of votes from a variety of classifiers, including KNN, RF, MLP, SVM, and ADA Booster.

**Table, calendar

Description automatically generated**

Figure 22 Classification report based on voting

**Result 21:** Figure 23 displayed the output image of different family classification such as Benign, and family 2 Malware on cartesian plane generated on the basis of different parameters that is given below.

**Graphical user interface, text

Description automatically generated**

Figure 23 Classification of Benign, and Family 2 Malware

1. **Conclusion and Future Scope**

The hybrid model differs from previous ML-based algorithms and is used as a complete detection strategy to prevent the spread of malware. This concept allows a mixture of features and classifier systems unlike past hybrid techniques (i.e., static/dynamic). This model provides a possible approach with real-time processing for identifying the essential characteristics of benign documents that present potentially harmful files. For malware identification, it is found that a hybrid model that combines classifiers with high TP and low FP performs very well. There are further insights that can be gained by using a short training dataset with the developed framework, as well. Malware categorization for 0-day attacks is made easier with the help of this approach, which can be used widely. This technique has the benefit of requiring less computing power than more traditional machine learning approaches.

Furthermore, this could increase its scalability so that it can be more readily updated to adapt to current malware and utilize the application for enabling malware to reveal complete behavior.

# **References**

Wazid, Mohammad, Ashok Kumar Das, Joel JPC Rodrigues, Sachin Shetty, and Youngho Park. "IoMT malware detection approaches analysis and research challenges." *IEEE Access* 7 (2019): 182459-182476.

Naseer, Muchammad, Jack Febrian Rusdi, Nuruddeen Musa Shanono, Sazilah Salam, Zulkiflee Bin Muslim, Nur Azman Abu, and Iwan Abadi. "Malware Detection: Issues and Challenges." In *Journal of Physics: Conference Series*, vol. 1807, no. 1, p. 012011. IOP Publishing, 2021.

1. Sihwail, Rami, Khairuddin Omar, and KA Zainol Ariffin. "A survey on malware analysis techniques: Static, dynamic, hybrid and memory analysis." *Int. J. Adv. Sci. Eng. Inf. Technol* 8, no. 4-2 (2018): 1662-1671.
2. Aslan, Ömer Aslan, and Refik Samet. "A comprehensive review on malware detection approaches." *IEEE Access* 8 (2020): 6249-6271.
3. Prasad, Ramjee, and Vandana Rohokale. "Cyber Forensics." In *Cyber Security: The Lifeline of Information and Communication Technology*, pp. 101-110. Springer, Cham, 2020.
4. López, Christian Camilo Urcuqui, et al. "Features to detect android malware." 2018 IEEE Colombian Conference on Communications and Computing (COLCOM). IEEE, 2018.
5. Samanvay Gupta. “Types of Malwares and its Analysis”. International Journal of Scientific & Engineering Research Volume 4, Issue 1, January-2013 1 ISSN 2229-5518.
6. Mishra, Preeti, Kashish Khurana, Saloni Gupta, and Manish K. Sharma. "VMAnalyzer: Malware semantic analysis using integrated CNN and bi-directional LSTM for detecting VM-level attacks in cloud." In *2019 Twelfth International.*
7. Liu, Kaijun, Shengwei Xu, Guoai Xu, Miao Zhang, Dawei Sun, and Haifeng Liu. "A review of android malware detection approaches based on machine learning." *IEEE Access* 8 (2020): 124579-124607
8. Ye, Yanfang, Tao Li, Donald Adjeroh, and S. Sitharama Iyengar. "A survey on malware detection using data mining techniques." *ACM Computing Surveys (CSUR)* 50, no. 3 (2017): 1-40
9. Shijo, P. V., and A. J. P. C. S. Salim. "Integrated static and dynamic analysis for malware detection." *Procedia Computer Science* 46 (2015): 804-811.

Chaffey, Emily J., and Daniele Sgandurra. "Malware vs Anti-Malware Battle-Gotta Evade ‘em All!." In *2020 IEEE Symposium on Visualization for Cyber Security (VizSec)*, pp. 40-44. IEEE, 2020.

Ferrante, Alberto, Miroslaw Malek, Fabio Martinelli, Francesco Mercaldo, and Jelena Milosevic. "Extinguishing ransomware-a hybrid approach to android ransomware detection." In *International Symposium on Foundations and Practice of Security*, pp. 242-258. Springer, Cham, 2017.

Minaee, Shervin, Yuri Y. Boykov, Fatih Porikli, Antonio J. Plaza, Nasser Kehtarnavaz, and Demetri Terzopoulos. "Image segmentation using deep learning: A survey." *IEEE transactions on pattern analysis and machine intelligence* (2021).

Vassallo, Carmine, Sebastiano Panichella, Fabio Palomba, Sebastian Proksch, Harald C. Gall, and Andy Zaidman. "How developers engage with static analysis tools in different contexts." *Empirical Software Engineering* 25, no. 2 (2020): 1419-1457.

Pan, Ya, Xiuting Ge, Chunrong Fang, and Yong Fan. "A systematic literature review of android malware detection using static analysis." *IEEE Access* 8 (2020): 116363-116379.

Jeon, Jueun, Jong Hyuk Park, and Young-Sik Jeong. "Dynamic analysis for IoT malware detection with convolution neural network model." *IEEE Access* 8 (2020): 96899-96911.

Afianian, Amir, Salman Niksefat, Babak Sadeghiyan, and David Baptiste. "Malware dynamic analysis evasion techniques: A survey." *ACM Computing Surveys (CSUR)* 52, no. 6 (2019): 1-28.

Onwuzurike, Lucky, Mario Almeida, Enrico Mariconti, Jeremy Blackburn, Gianluca Stringhini, and Emiliano De Cristofaro. "A family of droids-Android malware detection via behavioral modeling: Static vs dynamic analysis." In *2018 16th Annual Conference on Privacy, Security and Trust (PST)*, pp. 1-10. IEEE, 2018.

Faysal, Jabed Al, Sk Tahmid Mostafa, Jannatul Sultana Tamanna, Khondoker Mirazul Mumenin, Md Arifin, Md Awal, Atanu Shome, and Sheikh Shanawaz Mostafa. "XGB-RF: A Hybrid Machine Learning Approach for IoT Intrusion Detection." In *Telecom*, vol. 3, no. 1, pp. 52-69. Multidisciplinary Digital Publishing Institute, 2022.

Sihwail, Rami, Khairuddin Omar, and Khairul Akram Zainol Ariffin. "An effective memory analysis for malware detection and classification." *Comput., Mater. Continua* 67, no. 2 (2021): 2301-2320.

Yoo, Suyeon, Sungjin Kim, Seungjae Kim, and Brent Byunghoon Kang. "AI-HydRa: Advanced hybrid approach using random forest and deep learning for malware classification." *Information Sciences* 546 (2021): 420-435.

1. Alzaylaee, Mohammed K., Suleiman Y. Yerima, and Sakir Sezer. "DL-Droid: Deep learning based android malware detection using real devices." Computers & Security 89 (2020): 101663.
2. Shukla, Sanket, Gaurav Kolhe, Sai Manoj PD, and Setareh Rafatirad. "Rnn-based classifier to detect stealthy malware using localized features and complex symbolic sequence." In 2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA), pp. 406-409. IEEE, 2019.
3. Vinayakumar, R., Mamoun Alazab, K. P. Soman, Prabaharan Poornachandran, and Sitalakshmi Venkatraman. "Robust intelligent malware detection using deep learning." IEEE Access 7 (2019): 46717-46738.
4. Venkatraman, Sitalakshmi, Mamoun Alazab, and R. Vinayakumar. "A hybrid deep learning image-based analysis for effective malware detection." Journal of Information Security and Applications 47 (2019): 377-389
5. De Paola, Alessandra, Salvatore Gaglio, Giuseppe Lo Re, and Marco Morana. "A hybrid system for malware detection on big data." In *IEEE INFOCOM 2018-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pp. 45-50. IEEE, 2018.
6. Wang, Wei, Mengxue Zhao, and Jigang Wang. "Effective android malware detection with a hybrid model based on deep autoencoder and convolutional neural network." *Journal of Ambient Intelligence and Humanized Computing* 10, no. 8 (2019): 3035-3043.
7. Khudayer, Baidaa Hamza, Mohammed Anbar, Sabri M. Hanshi, and Tat-Chee Wan. "Efficient route discovery and link failure detection mechanisms for source routing protocol in mobile ad-hoc networks." IEEE Access 8 (2020): 24019-24032
8. Lu, Renjie. "Malware detection with lstm using opcode language." *arXiv preprint arXiv:1906.04593* (2019).
9. Sharma, Rohit, Sachin S. Kamble, Angappa Gunasekaran, Vikas Kumar, and Anil Kumar. "A systematic literature review on machine learning applications for sustainable agriculture supply chain performance." Computers & Operations Research 119 (2020): 104926.
10. Supreetha, B. S., Narayan Shenoy, and Prabhakar Nayak. "Lion algorithm-optimized long short-term memory network for groundwater level forecasting in Udupi District, India." Applied Computational Intelligence and Soft Computing 2020 (2020)
11. Chen, Yuantao, Jiajun Tao, Qian Zhang, Kai Yang, Xi Chen, Jie Xiong, Runlong Xia, and Jingbo Xie. "Saliency detection via the improved hierarchical principal component analysis method." *Wireless communications and mobile computing* 2020 (2020).
12. Narayanan, Barath Narayanan, Ouboti Djaneye-Boundjou, and Temesguen M. Kebede. "Performance analysis of machine learning and pattern recognition algorithms for malware classification." In *2016 IEEE National Aerospace and Electronics Conference (NAECON) and Ohio Innovation Summit (OIS)*, pp. 338-342. IEEE, 2016.

1. Samal, Janmejaya. "Fundamental tenets of epidemiology in Ayurveda and their contemporary relevance." *Indian Journal of Health Sciences and Biomedical Research (KLEU)* 9, no. 1 (2016): 20. [↑](#footnote-ref-1)
2. Eder-Neuhauser, Peter, Tanja Zseby, Joachim Fabini, and Gernot Vormayr. "Cyber attack models for smart grid environments." *Sustainable Energy, Grids and Networks* 12 (2017): 10-29. [↑](#footnote-ref-2)
3. Jing, Xuyang, Zheng Yan, and Witold Pedrycz. "Security data collection and data analytics in the Internet: A survey." *IEEE Communications Surveys & Tutorials* 21, no. 1 (2018): 586-618. [↑](#footnote-ref-3)
4. Bronk, Chris. "Getting creative on what will do: cyber espionage, conflict and covert action." *Conflict and Covert Action (June 15, 2016)* (2016). [↑](#footnote-ref-4)
5. Alauthaman, Mohammad, Nauman Aslam, Li Zhang, Rafe Alasem, and M. Alamgir Hossain. "A P2P Botnet detection scheme based on decision tree and adaptive multilayer neural networks." *Neural Computing and Applications* 29, no. 11 (2018): 991-1004. [↑](#footnote-ref-5)