Foundation University Journal of Engineering and Applied Sciences
Vol. 3, Issue 1, 2022.
DOI: 10.33897/fujeas.v3i1.687

Research Article

Intrusion Detection in Cyber Space Using Machine Learning Based Algorithm

Muhammad Bashir, Muhammad Atique, Saif Ur Rehman, Muhammad Ibrahim Khalil

University Institute of Information Technology (UIIT), PMAS Arid Agriculture University Rawalpindi, Pakistan

Corresponding Author: Saif Ur Rehman. Email**:** saif@uaar.edu.pk

Abstract:

Now a day, the fast growth of Internet access and the adoption of smart digital technology has resulted in new cybercrime strategies targeting regular people and businesses. The Web and social activities take precedence in most aspects of their lives, but also poses significant social risks. Static and dynamic analysis are inefficient in detecting unknown malware in standard threat detection approaches. Virus makers create new malware by modifying current malware using polymorphic and evasion tactics in order to fool. Furthermore, by utilizing selection of features techniques to identify more important features and minimizing amount of the data, these Machine Learning models' accuracy can be increased, resulting in fewer calculations. In the previous study traditional machine learning approaches were used to detect Malware. We employed Cuckoo sandbox, a malware detection and analysis system for detection and categorization, in this study we provide a Machine Learning based Intrusion analysis system to calculate exact and on spot Intrusion classification. We integrated feature extraction and component selection from the file, as well as selecting the much higher quality, resulting in exceptional accuracy and cheaper computing costs. For reliable identification and fine-grained categorization, we use a variety of machine learning algorithms. Our experimental results show that we achieved good, classified accuracy when compared to state-of-the-art approaches. We employed machine learning techniques such as K-Nearest Neighbor, Random Forest, Support Vector Machine, and Decision Tree. Using the Random Forest classifier on 108 features, we attained the greatest accuracy of 99.37 percent. We also discovered that Random Forest outscored all other classic machine learning techniques during the procedure. These findings can aid in the exact and accurate identification of Malware families.

Keywords: Cyber security; Security issues; Malware attacks; Cyber space; Intrusion detection

1. Introduction

The Internet space became an increasingly popular source of data as well as services. Internet usage rapidly increased after 2017, over 48 percent of the global population used the Internet as a channel of information. (Zhu, Jang-Jaccard, & Watters, 2020) In developed countries, this ratio rose to 81 percent.

The Internet's principal function is to carry data from one point to another across a network. The use of the Internet has risen dramatically as a result of advancements in computer systems, networks, and mobile devices. (Aslan & Samet, 2020) As a result, cybercriminals and adversaries have turned their attention to the Internet. Information confidentiality, availability, and integrity must all be guaranteed via a safe and stable computer system.

The term "cyber security" refers to a collection of security procedures that can be used to secure cyberspace and user assets against unwanted access and attacks. (Roseline, Sasisri, Geetha, & Balasubramanian, 2019) The basic goal of a cyber defense system is for data to be secure, integral, and accessible. Computer networks are (or should be) intended to provide safeguards that restrict data access

to just those who are authorized. Threat makers have begun to make threat online rather than in the real world as the Internet has grown in popularity. (Shaukat, Luo, Varadharajan, Hameed, & Xu, 2020) Digital data, machines that handle digital information, and programmed that manipulate digital data are all protected by cyber security.

Malicious software can be classified in a variety of ways. Zhu et al. (Zhu et al., 2020) categorized malware into three categories based on how harmful software distributes, including worms, viruses, and trojan horses. Another classification is based on the behavior of malicious programs after it has successfully infected the victim's computer. Ravi et al. (Ravi Vinayakumar et al., 2019), Aslan et al. (Aslan & Samet, 2020), and Ulalh et al. (Ullah et al., 2019) gave a more thorough reference to kinds of harmful software, dividing malware into classes depending on its utility and the manner it affects machines.

- **Virus –** Viruses are a sort of malicious code that is not self-contained and is frequently attached to other programs (commonly.exe files). Viruses can propagate throughout a system or network, infecting other programs and machines, thanks to the replication feature. The virus is carried out by an unknowing user who want to run a program without first confirming its source. Pure computer viruses, according to Ravi et al. (Ravi Vinayakumar et al., 2019) are less widespread than in the past and now account up only 10% of all malware. Viruses are difficult to eradicate from the system due to their propagation role, prompting antivirus programs to delete infected files entirely (Roseline et al., 2019). Viruses deplete system resources and can result in a service interruption.
- **Worm –** Unlike viruses, worms may exist on their own and have self-replicating capabilities, allowing them to infect all machines in a network without the need for human involvement or an infected application. The worm spreads via email and storage devices. Because of these characteristics, worms are extremely dangerous, as a single employee who views an infected email or uses a lost memory stick might infect the entire corporate network (Gülmez & Sogukpinar, 2021). Stuxnet, one of the most well-known computer viruses, was used to disrupt Iran's nuclear program by attacking supervisory control and data acquisition (SCADA) systems.
- **Trojan –** Horse unlike earlier examples, the Trojan horse is incapable of self-propagation. It is transmitted through the internet as a download file or malicious attachments in emails and needs human contact to attack machines. Trojans are made to appear as if they are a real and innocuous software. "The most common Trojan type is a false antivirus program that appears and claims your computer is infected, then instructs you to run a program to clean it." (Ravi Vinayakumar et al., 2019).
- **Adware –** Adware is computer virus which does not seriously affect the user or the machine, but is instead used to earn revenue for the attacker by displaying advertisements. Some adware directs users to websites that contain harmful software and encourages them to download it. Adware is frequently included in free programs. Sehatbakhsh et al. (Sehatbakhsh et al., 2019) investigated whether adware should be classified as malware. According to the authors, it is difficult to quantify the security risks posed by adware, and the community is currently arguing whether or not adware should be classified as malware.
- **Spyware –** Spyware is a persistent sort of malware designed for stealthy, long-term operation on compromised machines, according to authors (Xu, Xia, & Shen, 2020). Spyware captures personal data, such as passwords and financial information, and sends it to the attacker without

the victim's knowledge or agreement. According to Sreekumari (Sreekumari, 2020), spyware is frequently utilized in the reconnaissance phase of more complex attacks. It's frequently linked to whaling, espionage, and data theft for blackmail (Davis, 2019). Table 1 shows the different notations used.

Table 1: Notations definition table

Notation	Description
ML	Machine Learning
DNN	Dynamic Neural Network
NSL	Network Security Laboratory
UNSW-NB15	Network Intrusion Detection System
WSN-DS	Wireless Security Network- Data Set
NIDS	Network Inventory and Design System
HIDS	Host Based Intrusion Detection

We present a malware analysis approach in this paper to find and classify malware more accurately. We implemented the feature extraction module to extract features from the analysis report for malware. Furthermore, most important characteristics are extracted from the collected features and used to construct training and testing datasets using multiple feature selection techniques. Finally, we run the dataset using various Machine Learning (ML) methods. We employed Cuckoo sandbox, an automated malware analysis system for detection and categorization, in this study to provide a Machine Learning based Intrusion analysis system to calculate exact and on point Intrusion identification. We integrated feature extraction and component selection from the file, as well as selecting the much higher quality, resulting in exceptional accuracy and cheaper computing costs.

The next sections of the paper describe the previous related research work as literature review and then the proposed methodology of the framework we used in our research. The next section presents the experimental result and the finding of our research. After this, we discussed the outcomes and concluded our research with our discussed findings.

2. Literature Review

This section gives a comprehensive summary of recent work on machine learning in cyber security. We additionally narrowed our search by considering algorithm details and efficiency, feature extractions and selection methods (if applicable), and the relevant dataset used to solve an issue (s). This document also includes a quick summary of all the strategies.

A Multi-Loss Siamese Neural Network with Batch Normalization Layer was suggested by (Zhu et al., 2020), which can identify more accurately with fewer samples. Their model, which is trained with only a few samples, uses the Siamese Neural Network to detect new malware types. Their model includes batch normalization and several loss functions to handle over fitting caused by tiny samples, which can result in a vanishing gradient problem due to binary cross-entropy loss, as well as embedding space to improve detection accuracy. In addition, they show how to transform raw binary data into malware grey scale

images, as well as how to generate positive and negative pairs for training the popular Siamese Neural Network. Their results suggest that their model outperforms similar methods currently in use.

They presented a method for converting a binary file, such as the intrusion dataset, into raw images that may be fed into a neutrality network model, such as a Siamese Network. Their suggested model has been adjusted to operate well with a small number of training datasets. This is accomplished by appropriately adjusting the network parameters for feature extraction and applying batching normalization to prevent overfitting induced by the usage of tiny datasets. In the Siamese Neural Network for binary classification, the repeated error function is important for improving the feature embedding space. Each positive pair belonging to the same class has a tiny distance in the characteristic embedding. For one classification in malware detection, their suggested model outperforms the standard methods, according to their experimental data.

In another study, a variety of publicly available benchmark malware datasets, a complete finding of experiments with Dynamic Neural Network and other conventional machine learning classifiers (Ravi Vinayakumar et al., 2019). With the KDDCup 99 dataset, the ideal network Structure for Dynamic Neural Network are determined using hyper parameter selection approaches. All DNN trials are done for 1,000 epochs with learning rates ranging from [0.01-0.5]. To conduct the benchmark, the Dynamic Neural Network framework that over powered on KDDCup 99 is performed to various data files such as NSL-KDD, UNSW-NB15, Kyoto, WSN-DS, and CICIDS 2017. By feeding IDS data through several hidden layers, their Dynamic Neural Network model learns the multi-dimensional representation of feature. It has been proven through thorough experimental testing that DNNs outperform traditional machine learning classifiers. Finally, they presented Scale-Hybrid-IDS-AlertNet (SHIA), hybrid Dynamic Neural Network framework that will be utilized in run time to successfully discover network traffic and events in order to notify incoming cyber-threats. To avoid detection by the IDS, an attacker pretends to be a regular user. Intrusive behavior patterns, on the other hand, differ in some ways. This is related to an attacker's specific goal, such as gaining illegal entry to hardware and communications resources. The trend of internet resource usage can be captured, but current approaches have a significant false positive rate. Intrusion patterns can be found in typical traffic with a low key across long spans. An effective deep learning strategy is provided by building a deep neural network (DNN) to identify cyberattacks proactively by merging both NIDS and also HIDS jointly. On various NIDS and HIDS datasets, the efficacy of various traditional machine learning techniques and DNNs in identifying whether internet traffic behavior is normal or anomalous related to an attempt which can be categorized into appropriate attack categories is investigated in their paper.

$$h_i(x) = f(w_i^T x + b_i)$$

The mathematical formulae for sigmoid and tangent are given below.

$$Sigmoid = \frac{1}{1+e^{-x}}$$

$$Tangent = \frac{e^{2x}-1}{e^{2x}+1}$$

Authors (Shaukat et al., 2020) proposed that their paper aims to provide a comprehensive overview of the challenges that ML techniques face in protecting cyberspace against attacks by presenting a literature on ML techniques for cyber security, including threat detection, spam filtering, and intrusion detection on computer networks and mobile networks over the last decade. It also includes brief definitions of each

machine learning method, as well as security datasets that are often utilized, fundamental machine learning tools, and assessment metrics for assessing a categorization model. Finally, it analyses the difficulties of employing machine learning techniques in cyber security. This presentation includes a wide bibliography as well as recent ML trends in cyber security. ML techniques are used between sides of the attack, i.e., the attacker and the cyber security side. On the cybercriminal side, ML approaches are being used by malicious hackers and fraudsters to uncover system flaws and advanced attack methods to get past the defense wall. On the defense side, machine learning models are helping to give more robust and intelligent strategies for improving efficiency and early intervention of attacks, reducing the impact and damage.

Aslan and Samet (Aslan & Samet, 2020) presented that the Detection approaches based on signatures and heuristics are both quick and effective for detecting well-known threats also signature based approaches are not best to discover novel intrusion. For unknown and difficult intrusion, behavior-dependent, Neural learning-based, smart phone, and Internet - of - things technologies are also emerging to detect some portion of predictable and unpredictable threats. Model-checking-based and cloud-based systems perform well. However, no approach can detect all malware in the wild. This illustrates that finding an effective method for detecting malware is a difficult endeavor, and that new research and methodologies are desperately needed. The proposed study examines malware detection technologies in depth, as well as recent detection methods that use these approaches. The purpose of this methodology is to provide researchers with a broad understanding of malware detection approaches, as well as the benefits and drawbacks of each detection methodology and the methods employed in these approaches. Malware must be studied in order to comprehend its content and actions.

Roseline et al. (Roseline et al., 2019) suggested that in comparison to deep learning models, a hybrid stacked multilayered ensembling technique is more resilient and economical. With an accuracy of 98.91 percent, the suggested approach beats machine learning and deep learning models. The recommended technique works well for both little and large-scale data due to its versatility in automatically altering parameters (number of consecutive levels). In terms of resources and time, it is computationally efficient. When compared to deep neural networks, the approach requires far less hyper-parameters. Pattern analysis through visualization is used as an alternate approach of analyzing malware. Malware samples are transformed to grey images, which allow different classes to see distinct patterns. This method does not necessitate the execution of samples. They don't necessitate any special skills or monitoring software.

Farhan et al. (Ullah et al., 2019) proposed that, to detect source code plagiarism, a deep learning algorithm is deployed. The data was gathered as part of the Google Code Jam (GCJ) investigation into software piracy. Aside from that, through color image visualization, the deep convolutional neural network is used to detect dangerous infections in IoT networks. Malware samples were gathered from the Maling dataset for testing purposes. The experimental results show that the suggested solution's classification performance for measuring cybersecurity threats in IoT is superior to current methodologies. Traditional approaches may address code concealment issues, however texture feature mining with virus visualizations requires a significant computational cost. With substantial malware data analysis, these kinds of data mining algorithms do not perform effectively. Virus is presently constantly producing, updating, and manipulating itself, making identification more difficult. The suggested malware identification system aims to answer the questions below: How can malware be identified with minimal effort? How can malware traits be extracted with a lower computational cost? How to improve accuracy by processing large malware datasets.

Sethi et al. (Sethi, Kumar, Sethi, Bera, & Patra, 2019) presented that they created a ML based intrusion analysis system for improved and precise intrusion identification and categorization. They used framework for multi-dimensional identification, which runs intrusions over a separate space also creates a report according to behaviors. They also performed extraction of features with implementing module that retrieve features from file and selects the most special characteristics to ensure more improved accuracy while lowering power costs. They utilized a number of ML methods for precise identification and streamed categorization. By this they achieved good identification and also categorization precisions when compared to high end approaches.

Gulmez et al. (Gülmez & Sogukpinar, 2021) proposed that the prevention detection methods based on static analysis, most malware makers employ obfuscation and encryption techniques. This type of malware is known as packed malware, and it is widely believed that in order to identify it, it must either be unpacked or evaluated dynamically. To address these issues, a graph-based malware detection method is suggested. The proposed method works by getting the opcode graph of each executable file in the dataset and using it to extract data later. As a result, the proposed approach achieves a detection accuracy of up to 98 percent. Aside from the high accuracy rate, the suggested method allows for the detection of packed malware without the requirement for unpacking or dynamic analysis. Graphs are used to represent source code. The classification of the graphs, they believe, leads to the discovery of malware. Different methods have been implemented to achieve this goal. The initial step is to deconstruct all of the dataset's files. The opcodes of a file are exposed during deconstructing, and these opcodes create opcode series.

Zhang et al. (Zhang, Kodituwakku, Hines, & Coble, 2019) suggested that Traditional process monitoring systems look for cyber-threats that are not identifiable by supervising network, such as data manipulation and false input attacks by an inside user. Regression model is investigated in the suggested detection system to increase early assault detection. The results suggest that following method identify damaging cyber threats as soon as they have a significant impact. A viable method for protecting an ICS is the suggested multiple layered input driven intrusion identification system, which uses data as well as networks and system. The proposed cyber-attack detection system's architecture, based on the defense and security concept. The classic intrusion detection and prevention layer, which includes firewalls, data diodes, and gateways, is the initial protection layer and is already widely used in the industry. However, in some cases, the attackers may be able to get beyond this defense line. The second security layer is made up of information models for detecting cyber-attacks based on network traffic and system data, such as the M1 classification algorithm and M2 big data models.

Vinayakumar et al. (Vinayakumar, Alazab, Soman, Poornachandran, & Venkatraman, 2019) presented that the first part of their paper compares and contrasts traditional MLAs as well as deep learning framework for intrusion identification and also categorization utilizing available and restricted datasets. By employing distinct portions of the available and restricted datasets to learn and run the model in a separable manner using timelines, they eliminated all data biasness from the experimental results. They proposed a unique visual processing technique that uses best settings for MLAs as well as deep learning framework to produce a precise intrusion identification model. They suggested deep learning architectures outperform classical MLAs, according to a comprehensive comparison assessment of our model. Their innovation in integrating visualization and deep learning architectures for a hybrid method based on static, dynamic, and image processing performed in a big data platform is unique in the world in terms of providing robust intelligence zero-day threat detection.

The author of (Sehatbakhsh et al., 2019) suggested that they offer REMOTE, a new structure that is meant

to address logistical problems for tracking resource-constrained devices (e.g., embedded devices, IoTs, CPS, and so on), such as: Availability of code, assessment, and/or instruments equipment: source code, measurement, and/or instrumentation infrastructure may be unavailable. Flexibility in Software as well as Hardware: the equipment may be determined by different CPU architectures, and the device may use different operating systems or just not be there at all, Containers may limit immediate access to the network, such as placing a power or EM sensor very near to a microprocessor or indeed the main panel. When employing analogue signals for surveillance, the environment can modify the signal that is sent out and/or add interference to the signal that is received.

Xu et al. (Xu et al., 2020) proposed that When it comes to online attacks, earlier research has all assumed that virus will carry out attacks on the electric grid as soon as it infects a new host. In fact, the virus will not initiate attacks until enough hosts have been compromised in order to achieve the best attack effect. The virus has a lag phase during which it causes no damage but merely spreads quietly to infiltrate victims and evade detection (for example, in the Ukrainian incident, the virus had already been spreading surreptitiously in the telecommunications for more than six months before carrying out attacks). Previous research has rarely analyzed in depth the implantation duration and malware detection likelihood, both of which are important elements in determining the level of harm caused by malware attacks in CPPSs. As a result, we suggest a more practical paradigm in this brief that takes these overlooked elements into consideration.

Javaheri et al. (Javaheri, Lalbakhsh, & Hosseinzadeh, 2021) presented that the method to identifying unusual virus types, as well as present and prospective alterations produced by tectonic algorithms Several of the malware classes their studied are highly rare and cryptic. From the Adminus, VirusSign, and VirusShare computer viruses' datasets, they were able to collect and acquire some important data for these classes. They selected examples from different types of unusual malware at randomly to construct the first population, comprising stealthy spyware, kernel rootkit, injector, blocker, bootkit, evader, and file-less malware. To evade suspicion and exposure of their reaction, all samples from the early population malware were highly packed and secured by tough packers, as well as various unknown packers. The initial population is insufficiently large to allow for good and reliable categorization. This difficulty was solved by creating a fresh dataset, which increased the population's size and quality, allowing for more reliable training.

Sreekumari et al. (Sreekumari, 2020) suggested that By collecting feedback' device action without one's approval, types of malware can play a number of roles, including theft, encoding, erasing, and changing personal details such as bank account information, credit card information, login information, Security Numbers, and so on. Furthermore, malware can alter the system's needed functions by adding, changing, or removing programs. It takes advantage of a digital system's weaknesses, which are flaws or risks of harm. Malware's delivery reveals the malware's intent. Hackers write payloads, which are little pieces of code that operate on the machines they infect. Whereas many people are delighted with their modern technological presents and technology, they may be unaware of the frightening rate at which attackers are hacking their gadgets. Deep learning, for example, is one of the most essential ways for identifying malware because it has been shown to have a positive impact on natural language processing and image categorization.

Further, the comparison of closely related techniques is presented in Table 2.

Table 2: Comparative analysis of malware detection technique

Article	Methodology	Research Findings	Dataset Used	Malware Addressed	Accuracy
Jang-Jaccard et al. (Zhu et al., 2020)	Multi loss Siamese neural network	Batch Normalization and multi loss function	VirusTotal Dataset	All malware family	99.2%
Vinayakumar et al. (Ravi Vinayakumar et al., 2019)	Deep neural network	Scale-Hybrid-IDS-Alert Net	HIDS, NIDS KDDCup 99	All malware family	92.1%
Shaukat, K et al. (Shaukat et al., 2020)	Machine learning Techniques	Applications of ML Models	DARPA ID	All malware family	94.1%
Aslan, O et al. (Aslan & Samet, 2020)	Malware detection approaches	Signature and heuristic based approaches	NSL-KDD, Drebin, EMBER	All malware family	97.1%
Roseline et al. (Roseline et al., 2019)	ML Random Forest	Effective with contemporary deep learning model.	Kaggle's BIG 2015	All malware family	98.8%
Ullah, F et al. (Ullah et al., 2019)	Deep learning Convolutional neural network	Deep learning for identification of pirated and malware files	Google code jam, Malimg	All malware family	96%
Sethi, K et al. (Sethi et al., 2019)	Machine learning Algorithm	Decision tree out performed other ML Algorithms	VirusTotal, VirusShare	All malware family	99.1%
Gulmez et al. (Gülmez & Sogukpinar, 2021)	Graph based operational codes	Proposed model out performed other techniques	VX-Heaven	All malware family	97%
Zhang, F. et al. (Zhang et al., 2019)	Machine learning Algorithm	High detection accuracy and wide attack coverage.	MITM	All malware family	98.8%
Vinayakumar et al. (R Vinayakumar et al., 2019)	Deep learning techniques MLAs	Deep learning out performed classical MLAs	Malimg	All malware family	98.9%
Sehatbakhsh et al. (Sehatbakhsh et al., 2019)	REMOTE	REMOTE out performed state of the art external malware detection framework.	MiBench	All malware family	99.8%
Xu, S et al. (Xu et al., 2020)	Cyber Physical Power System (CPPS)	Defer the maximum payoff of the attacker.	CPPS	----	97.8%
Javaheri, D et al. (Javaheri et al., 2021)	Genetic Algorithm	Genetic algorithm out performed.	Microsoft, Malimg	All malware family	98%
Sreekumari et al. (Sreekumari, 2020)	Deep learning Algorithms	Deep learning is the profound solution for Malware detection.	Maling	----	96-99%

3. Proposed Methodology

In this portion, we present a full overview of our suggested framework for identifying and categorizing provided data.

We created a Python script to extract significant features, which was then used to choose the most important features, resulting in training as well as testing data pool. The data pool consists of a Mega data pool used for identification as well as a Mini data pool for categorization. For the identification and categorization of the given dataset, we employ multiple machine learning methods offered by the Scikit-learn module in Python. Scikit-Learn is a popular machine learning library in Python that provides a range of supervised and unsupervised learning algorithms. The library includes models based on machine learning approaches such as linear models, tree-based models, support vector machines, naive Bayes, clustering, and neural networks. Scikit-Learn also provides various tools for data preprocessing, model selection, and evaluation. The proposed methodology's operational flow is depicted in Figure 1. There are three steps to it as listed in Algorithm 1. We provide a full description of these three stages in this article.

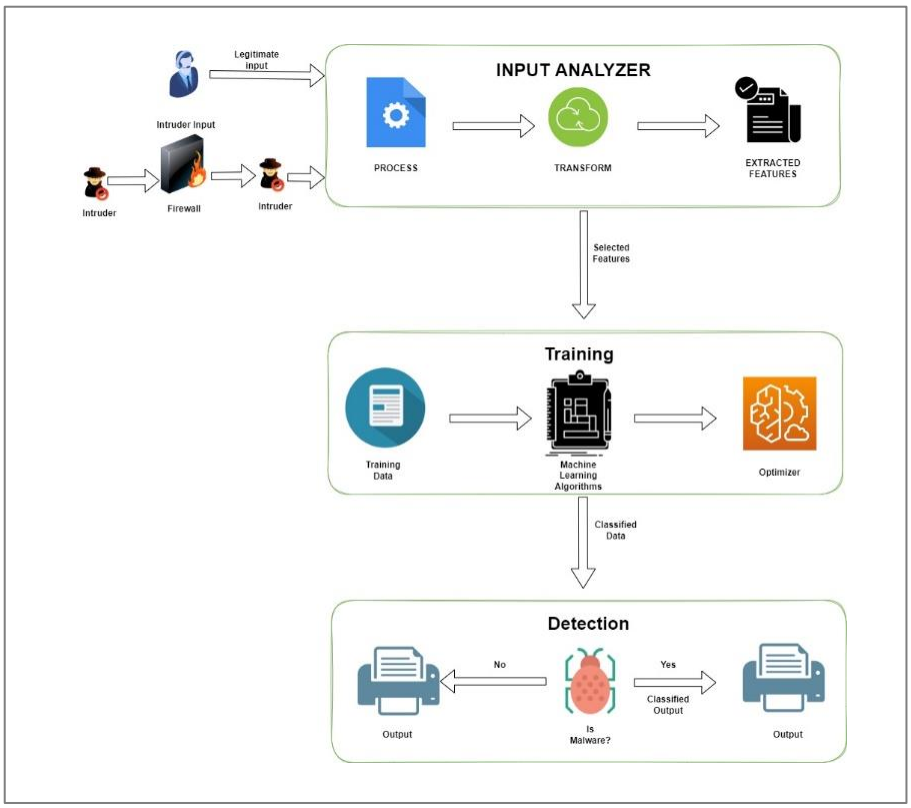

Figure 1: Structural design of implemented framework

3.1 Input Analyzer (Data Collection)

Due to the lack of a comprehensive dataset for malware analysis, we created our own. We'll use this dataset to conduct more malware analysis utilizing machine learning algorithms. We gathered malware and clean files containing windows PE header files from the VirusShare and VirusTotal websites.

3.2 Training (Pre-processing and Feature Extraction)

Because each source code has a different syntax and semantic structure, detecting pirated software among several types of source codes is a difficult operation. Detecting pirated software can be challenging

***Algorithm** 1 Proposed Methodology Steps*

Let ζD= dataset

Begin

Step 1: Get(ζD)

Step 2: Input Analyzer

Step 3: Training dataset

3.1. Dataset splitting for training, testing and validation

3.2. Feature Extraction EfficientNetB0 pre-trained model

3.3. Optimize (epochs, batch size learning weights)

Step 4: Evaluation Metrics (accuracy, precision, F1 score and recall)

End

because pirates use various techniques to hide their activities and disguise the origin of the software. Some common techniques include code obfuscation, modifications, encryption, distribution methods, and changing file names. These tactics make it difficult to identify pirated software among legitimate sources. However, software companies and law enforcement agencies are continuously developing new methods to detect and prevent software piracy. To detect source code similarity, we used software plagiarism approaches. The source codes are broken down into little parts for deep analysis using pre-processing techniques. We used Scikit Learn library to complete the Data Preprocessing. It covers stemming, root word extraction, and frequency extraction, as well as the elimination of words. It translates the codes into relevant information and filters out the noise. Unwanted features, such as special symbols, constants, and stop words, are removed from the data. The cleansed data is then transformed into useable tokens using the tokenization process. In the pre-processing step, stemming, root words, and frequency limitations are employed to extract more valuable characteristics. The contribution of each token is then zoomed using weighting techniques. Data cleansing and transformation are important steps in data analysis because they help to ensure that the data being used is accurate, complete, and relevant to the research question at hand. When working with data, it is common to encounter missing values, outliers, and other types of errors or inconsistencies. These issues can affect the results of statistical analysis and make it difficult to draw meaningful conclusions from the data.

Weighting techniques are often used to adjust the importance of different observations in the data based on various factors such as sample size, response rate, or demographic characteristics. However, if the data is not cleansed and transformed prior to applying weighting techniques, the resulting weights may be biased or inaccurate. For example, if there are missing values or outliers in the data, these may affect the weighting calculations and lead to incorrect results.

Additionally, data cleansing and transformation can help to ensure that the variables being used are in the appropriate format and have the desired level of granularity. For example, if a variable is recorded in a different unit of measurement than what is needed for analysis, it may need to be transformed before applying weighting techniques.

In summary, data cleansing and transformation are important steps in data analysis that can help to ensure

the accuracy and relevance of the data being used. By preparing the data properly before applying weighting techniques, researchers can increase the reliability of their results and draw more meaningful conclusions from the data.

In the weighting step, the TFIDF and Logarithm of Term Frequency (LogTF) are utilized. TF-IDF (Term Frequency-Inverse Document Frequency) is a statistical weighting scheme commonly used in information retrieval and text mining. It is used to represent the importance of each word or term in a document based on its frequency of occurrence across a collection of documents.

The term frequency (TF) component of TF-IDF measures the frequency of a word or term within a single document. The inverse document frequency (IDF) component measures how rare or common a word is across the entire collection of documents. The idea behind IDF is that words that occur frequently in many documents, such as "the" or "and", are less important in distinguishing between documents and should be given less weight, while words that occur rarely or only in a few documents are more important in distinguishing between documents and should be given more weight. LogTF, on the other hand, is a modification of the TF component of TF-IDF. It takes the logarithm of the raw frequency of a term in a document, which helps to reduce the impact of very high term frequencies. This is useful because when a term occurs very frequently in a document, it can skew the weight of that term and make it seem more important than it really is.

Stemming and root word extraction are techniques used in natural language processing to transform words into their base or root form, which can help to reduce vocabulary size and improve text analysis. Stemming involves removing suffixes to get to the base form of a word, while root word extraction identifies the base form without necessarily removing all suffixes. Both techniques can improve text analysis, but their effectiveness may vary depending on the language, domain, and task at hand.

3.3 Detection and Classification

The Scikit-Learn library was utilized to identify and categorize malware in our Implemented methodology. Marco dataset and Micro dataset are two datasets created utilizing feature purification. These datasets are then separated into learning and implementing sub datasets in a 70/30 parts for model learning and implementation, respectively. Used a python library of Scikit, a Model was created using machine learning approaches, the training data pools were called into a coded program of Python language. The models created in Scikit library are: (1) -Binary Classification: - A set of data is used to identify whether a provided sample is malicious or not; and (2)- Multi-Class Classifier: - Its extracted features or set of data is utilized to categorize a provided sample data into various virus kinds.

4. Experiment and Results

In this section, with detailed experimental findings, we'll show you how well our malware analysis process performed. For our investigation, we used 1200 data samples, 678 harmful samples and 5+ harmless samples were found. For both binary and multi-class classifiers, the Scikit-Learn module gave results with precise identification of each class and a matrix. We separated them into two groups for malware analysis: training and testing. Seventy percent of malware samples are in the training set, while thirty percent are in the testing set. Using the models created from the machine learning techniques, we observed results for identification and categorization. The following subsections explain the outcomes of the experimental examination.

4.1 Result Analysis

To analyze the effectiveness of our proposed malware analysis system, Reliability, Precise, Retention, F-measure, and Region underneath Fitted Model were the five performance measures we looked at. Real Positive, Untrue Positive, Real Negative, and Fake Negative are the performance measurements.

4.2 Evaluation Metrics

Before explaining the evaluation metrics, we would give a brief introduction to the confusion matrix. A confusion matrix is generally a 2x2 matrix layout that is beneficial for visualizing the performance of an algorithm. Actual performance measures that must or should be met are written vertically while the predicted ones by the algorithm are written horizontally (Ravi Vinayakumar et al., 2019).

True Positive Rate (TPR) is the total positive instances being identified as positive.

$$\text{TPR}=\frac{TP}{TP+FN}$$

True Negative Rate (TNR) is the number of negative instances being identified as negative.

$$\text{TNR} = \frac{TN}{TN+FP}$$

False Positive Rate (FPR) is the number of negative instances being classified or predicted as positive.

$$\text{FPR} = \frac{FP}{FP+TN}$$

False Negative Rate (FNR) is the number of positive instances being classified or predicted as negative.

$$\text{FNR} = \frac{FN}{FN+TP}$$

Accuracy: is the ratio between the number of correct predictions and a total number of predictions.

$$\text{Accuracy} =\frac{TP+TN}{TP+FP+FN+TN}$$

Precision: is the ratio between TPs combined to a number of TPs and FPs. It is the percentage of correctly identified positives out of all results which were said to be positive either correctly or not.

$$\text{Precision} = \frac{TP}{TP+FP}$$

Recall: is defined as the ratio between TPs combined to a number of TPs and FNs. It is the percentage of correctly identified positives out of all actual positives, either correctly or not.

$$\text{Recall} = \frac{TP}{TP+FN}$$

F1-score: It takes both false negatives and false positives into consideration, and it is the harmonic mean of recall and precision. It performs well on datasets that are imbalanced.

$$\text{F1-Score} = 2 * \frac{(\text{precision}*\text{recall})}{(\text{precison}+\text{recall})}$$

5. Malware Identification Outcomes

Results of intrusion identification using several classifiers in the trial. Table 3 and Figure 2 shows that with 108 selected features, Random Forest has a high identification rate of 99.37 percent and an accuracy of 99.37 percent. Among other classifiers, this one has the highest accuracy. Similarly, we used Random Forest to attain high Precision, Recall, and F-Measure values. In comparison to the other classifiers employed in the experiment, these measures are the best. The comparison metrics are shown in Figure 3.

Table 3: Malware identification results

Algo	Features	Accuracy	Precision	Recall	F-measure	AUC
KNN	108	94.68%	0.95	0.95	0.91	0.85
Random Forest	108	99.37%	0.99	0.99	0.99	0.98
SVM	1000	89.37%	0.92	0.90	0.90	0.93
Decision Tree	92	95.93%	0.96	0.96	0.96	0.81

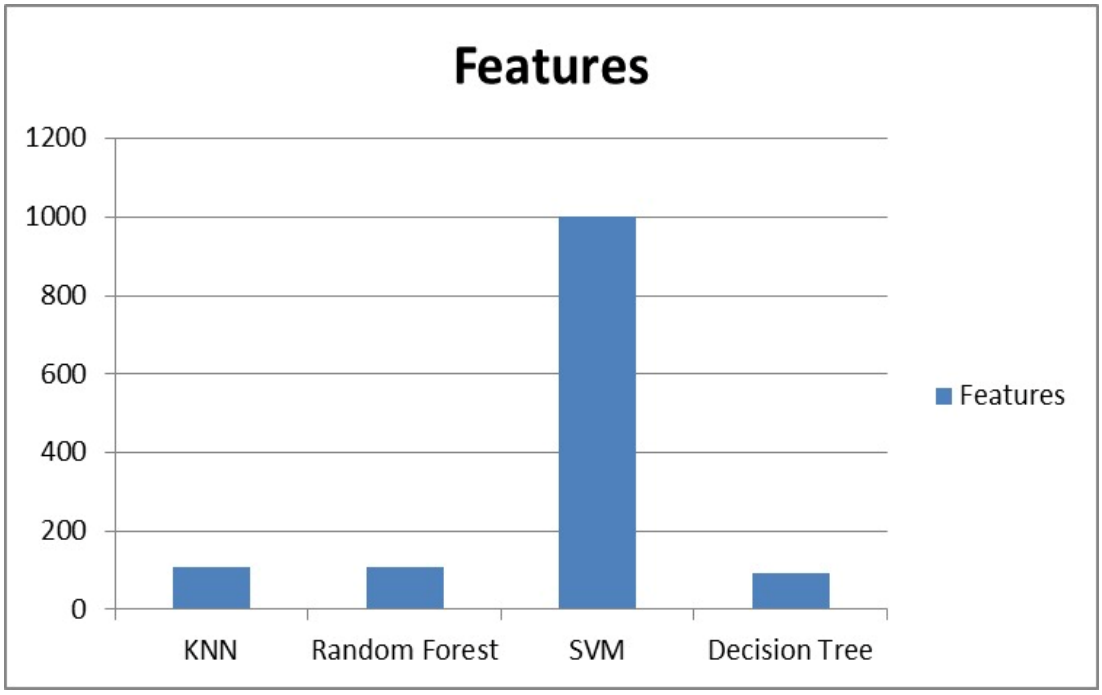

Figure 2: Graphical representation of features

Here next, different algorithms used for classification are briefly discussed.

5.1 K-nearest Neighbors

The function is only approximated locally in k-NN classification, and all computation is postponed until

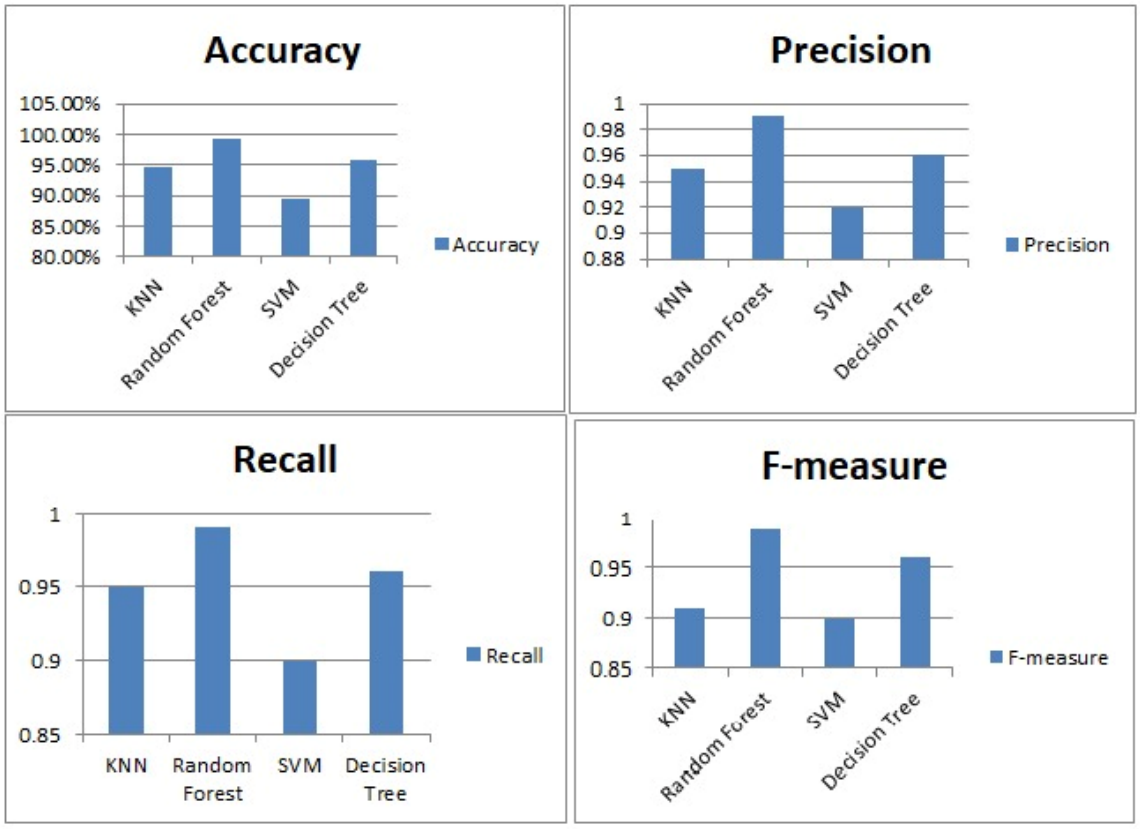

Figure 3: Graphical representation of experimental results

the function is evaluated. Because this method relies on distance for classification, normalizing the training data can greatly increase its performance if the features represent various physical units or come in wildly different scales. An effective strategy for both classification and regression is to assign weights to the contributions of the neighbors, so that the closer neighbors contribute more to the average than the farther neighbors. A popular weighting technique, for example, is to give each neighbor a weight of 1/d, where d is the distance between them.

In k-NN classification, the contributions of neighbors can be weighted to improve accuracy. One common method is the inverse distance weighting (IDW) method, which assigns weights to neighbors based on their distance to the point being classified. Another method is the Gaussian kernel weighting method, which uses a Gaussian function to weight the contributions of neighbors based on their distance. Both methods can be useful in cases where some neighbors are more informative than others due to their proximity to the point being classified.

5.2 Random Forest Algorithm

Random forests, also known as random choice forests, are an ensemble learning method for classification, regression, and other tasks that works by building a large number of decision trees during training. For classification tasks, the random forest's output is the class chosen by the majority of trees. The mean or average prediction of the individual trees is returned for regression tasks. Random decision forests address the problem of decision trees overfitting their training set. Random forests outperform decision

trees in most cases, but they are less accurate than gradient enhanced trees. Data features, on the other hand, can have an impact on their performance.

5.3 Support Vector Machine

Support-vector machines (SVMs, also known as support-vector networks) are supervised learning models that examine data for classification and regression analysis using related learning techniques. Vladimir Vapnik and colleagues developed it at AT&T Bell Laboratories (Boser et al., 1992, Guyon et al., 1993, Cortes and Vapnik, 1995, Vapnik et al., 1997). SVMs, which are based on statistical learning frameworks or VC theory established by Vapnik (1982, 1995) and Chervonenkis, are one of the most reliable prediction approaches (1974).

5.4 Decision Tree

One of the predictive modelling methodologies used in statistics, data mining, and machine learning is decision tree learning, also known as induction of decision trees. It goes from observations about an item (represented in the branches) to inferences about the item's goal value using a decision tree (as a predictive model) (represented in the leaves). Classification trees are tree models in which the goal variable can take a discrete set of values; in these tree structures, leaves indicate classifiers and branching represent feature combinations that lead to those class labels. Logistic regression are decision trees in which the target variable can take continuous values (usually real numbers). Because of their comprehensibility and simplicity, decision trees are one of the most popular machine learning methods.

6. Malware Classification Results

6.1 Malware Categorization Results

The performance of classification utilizing the various classifiers employed in our experiment demonstrate that Random Forest has an extremely accurate percentage of 99.11 % in categorization. KNN, SVM, and DT, on the other hand, yield classification rates of 86.72 percent, 86.72 percent, and 88.23 percent, respectively. Random Forest also scored well in Precision, Recall, and F-Measure.

6.2 Malware Family Classification Results

Table 4 shows the malware family classification results for each class. have employed seven types of

Table 4: Intrusion family categorization results

Class	Samples	Accuracy	Precision	Recall	F-measure
Virus	81	90%	1.00	0.90	0.95
Adware	231	100%	1.00	1.00	1.00
Trojan	255	100%	0.94	1.00	0.97
Spyware	48	77.77%	0.58	0.78	0.67
Worm	27	66.66%	1.00	0.67	0.80
Backdoor	24	62.5%	1.00	0.62	0.77

malwares (viruses, adware, trojans, etc.) Backdoors, hoaxes, spyware, and worms are all examples of malicious software. Each class has its own set of rules. Figure 4 shows the number of samples we collected. For Adware and Trojans, the Random Forest classifier model worked best. Thus, Adware and Trojans viruses achieved the highest accuracy. Because of the enormous number of samples, malware types have been created. In comparison to other classes, for those classes. By achieving a high recall and precision rate for certain classes. This indicates that the projected results are very accurate. The results of evaluation metrics are shown in Figure 5.

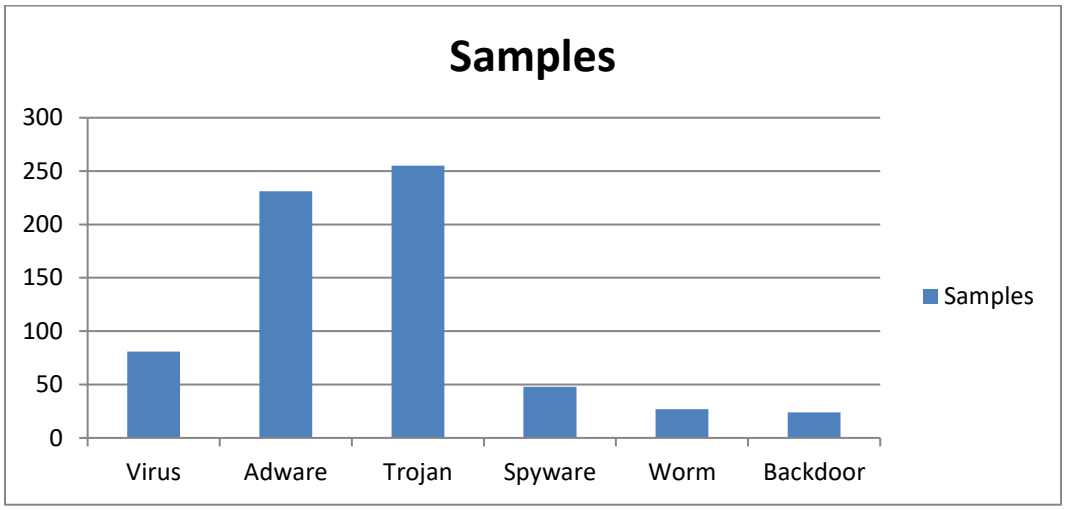

Figure 4: Graphical representation of samples

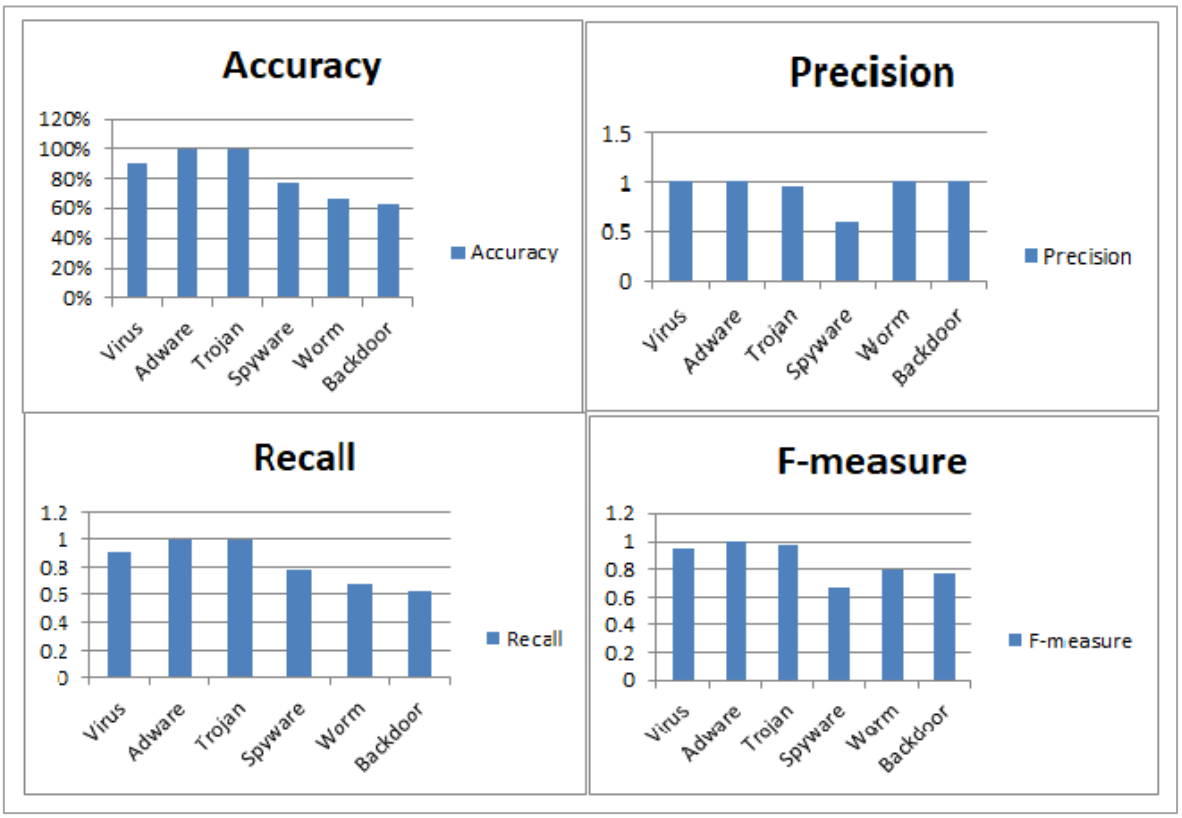

Figure 5: Graphical representation of results

7. Conclusion

We introduce a novel malware analysis methodology in this paper that can efficiently detect and categories malware. In order to extract the most important characteristics, our suggested approach uses two independent feature selection algorithms, this reduces learning time and enhances the reliability of detection and categorization. Gathered results reveal to with information like Random Forest has good detection and classification accuracy. Furthermore, we have classified intrusion based on its class and verified the identification of each intrusion class.

References

Aslan, Ö. A., & Samet, R. (2020). A comprehensive review on malware detection approaches. *IEEE access, 8*, 6249-6271.

Davis, R. S. a. G. (2019). McAfee Mobile Threat Report Q1. from https://www.mcafee.com/enterprise/en-us/assets/reports/rp-mobile-threat-report-2019.pdf

Gülmez, S., & Sogukpinar, I. (2021). *Graph-based malware detection using opcode sequences.* Paper presented at the 2021 9th International Symposium on Digital Forensics and Security (ISDFS).

Javaheri, D., Lalbakhsh, P., & Hosseinzadeh, M. (2021). A novel method for detecting future generations of targeted and metamorphic malware based on genetic algorithm. *IEEE access, 9*, 69951-69970.

Roseline, S. A., Sasisri, A., Geetha, S., & Balasubramanian, C. (2019). *Towards efficient malware detection and classification using multilayered random forest ensemble technique.* Paper presented at the 2019 International Carnahan Conference on Security Technology (ICCST).

Sehatbakhsh, N., Nazari, A., Alam, M., Werner, F., Zhu, Y., Zajic, A., & Prvulovic, M. (2019). REMOTE: Robust external malware detection framework by using electromagnetic signals. *IEEE Transactions on Computers, 69*(3), 312-326.

Sethi, K., Kumar, R., Sethi, L., Bera, P., & Patra, P. K. (2019). *A novel machine learning based malware detection and classification framework.* Paper presented at the 2019 International Conference on Cyber Security and Protection of Digital Services (Cyber Security).

Shaukat, K., Luo, S., Varadharajan, V., Hameed, I. A., & Xu, M. (2020). A survey on machine learning techniques for cyber security in the last decade. *IEEE access, 8*, 222310-222354.

Sreekumari, P. (2020). *Malware detection techniques based on deep learning.* Paper presented at the 2020 IEEE 6th Intl Conference on Big Data Security on Cloud (BigDataSecurity), IEEE Intl Conference on High Performance and Smart Computing,(HPSC) and IEEE Intl Conference on Intelligent Data and Security (IDS).

Ullah, F., Naeem, H., Jabbar, S., Khalid, S., Latif, M. A., Al-Turjman, F., & Mostarda, L. (2019). Cyber security threats detection in internet of things using deep learning approach. *IEEE access, 7*, 124379-124389.

Vinayakumar, R., Alazab, M., Soman, K., Poornachandran, P., Al-Nemrat, A., & Venkatraman, S. (2019). Deep learning approach for intelligent intrusion detection system. *IEEE access, 7*, 41525-41550.

Vinayakumar, R., Alazab, M., Soman, K., Poornachandran, P., & Venkatraman, S. (2019). Robust intelligent malware detection using deep learning. *IEEE access, 7*, 46717-46738.

Xu, S., Xia, Y., & Shen, H.-L. (2020). Analysis of malware-induced cyber attacks in cyber-physical power systems. *IEEE Transactions on Circuits and Systems II: Express Briefs, 67*(12), 3482-3486.

Zhang, F., Kodituwakku, H. A. D. E., Hines, J. W., & Coble, J. (2019). Multilayer data-driven cyber-attack detection system for industrial control systems based on network, system, and process data. *IEEE Transactions on Industrial Informatics, 15*(7), 4362-4369.

Zhu, J., Jang-Jaccard, J., & Watters, P. A. (2020). Multi-loss Siamese neural network with batch normalization layer

for malware detection. *IEEE access, 8*, 171542-171550.