UNIVERSITY OF NAIROBI

KIBUI PAULINE WAITHERERO
P53/34303/2019

CDT 599: RESEARCH PROJECT

A COMPARATIVE ANALYSIS OF A.I. ALGORITHMS FOR MALWARE DETECTION

SUPERVISOR: DR. ELISHA ABADE

RESEARCH PROJECT REPORT SUBMITTED IN PARTIAL FULFILLMENT FOR THE REQUIREMENTS OF THE AWARD OF THE DEGREE OF MASTER OF SCIENCE IN DISTRIBUTED COMPUTING TECHNOLOGY, SCHOOL OF COMPUTING AND INFORMATICS.

DECLARATION

Research Declaration

I hereby declare that this project is my original work. It was done under the guidance and supervision of Dr. Abade. It has not been previously submitted anywhere else for any other reason. Where I have used other people's work, this has been acknowledged accordingly by citation and referencing according to the requirements of the University of Nairobi.

SIGNED:.. DATE: 10 - 12-2021

Kibui Pauline Waitherero
P53/34303/2019

Declaration by Supervisor

This project report has been submitted to the Department of Computing and Informatics, University of Nairobi for examination. This has been done with my approval as the supervisor for the research.

SIGNED:.. DATE: 10/12/2021.

Dr Elisha Odira Abade
Department of Computing and Informatics

Dedication

This work is dedicated to my two families for their love, moral support and understanding during the entire period I have been undertaking these studies.

Acknowledgement

I am grateful first of all to God for having granted me the strength and help to bring to completion this project work, to my family for their understanding and for always being there for me the entire time. I am grateful too to my research supervisor Dr. Elisha Abade whose valuable guidance has been key in making this Project a success.

Thank you too to the entire team of the School of Computing, University of Nairobi for your selfless efforts in trying to help us grow professionally and complete our course.

Abstract

Malware - also known as malicious programs or code, are one of the biggest threats in computing today. They have become very easy to develop and thousands are produced every day. They mutate very easily making it very difficult to control. The most accessible mitigation is anti-malware tools. However, due to the reasons above the traditional signature based malware scanning tools have proved insufficient. For this reason, the antimalware industry is constantly rethinking ways of improving their detection methods. (Ye, Li, Adjeroh, & Iyengar, 2017).

This research was conducted to assess and compare the performance of machine learning algorithms in the detection of malware.

TABLE OF CONTENTS

Introduction

Background

Internet usage today is growing very fast, the world over. This has contributed to malware becoming one of the biggest threats. Further to this, developing malware has been overly simplified. This is because today there are many malicious tools and applications easily accessible on the internet, tools for automated detection. Purchasing malware has also been made very accessible. All these facilitate for any interested persons to easily become an attacker, even with very basic skill levels(Mouhammd Al-Kasassbeh, Mohammed, Alauthman, & Almomani, 2020).

According to definition given by Souppaya et al, malware, or malicious programs/code,are programs that are maliciously inserted into an existing system or program with the ill intention of causing some damage, by destroying data, executing destructive or intrusive programs, or by compromising the victim's data, applications, or operating system, in terms of their confidentiality, integrity, or availability. Malware is considered the most prevalent external threat to technology users. A lot of damage and service or business disruption in many organizations are as a result of this. To overcome this challenges, these organizations have to go through extensive recovery efforts. (Souppaya & Scarfone, 2013)

Although newer forms of malware don't always properly fit into these, the classical categories of malwares include: (a) viruses which replicate themselves by inserting copies of itself into already installed or existing programs or files in a system. They could either be compiled viruses (meaning that the operating System is the one that executes them. These are of two types: *file infector viruses* - these attach themselves to .exe files or programs, and the second type is called *boot sector viruses,* this kind infects the boot sector of a machine or of removable media) or interpreted viruses (this means that they are executed by an application eg macro viruses and scripting viruses); (b) worms which are self-contained programs that replicate themselves. They don't need any user to execute them, they execute on their own. They could either be mass mailing worm or worms that affect the network service . (c) Trojan-horses which are programs that are self contained and which do not replicate themselves. However, although they appear to be harmless, in reality they have a malicious motive that is usually hidden. (d) Malicious mobile code, which is a software usually created using Java, ActiveX, Javascript or VB Script. They don't need a user to explicitly execute them, however, they tend to have a malicious intention. Their mode of operation is that while connected to a network, they tend to be transmitted from another machine accessible on the network, to the user’s machine.They are then executed on the user’s machine, and (e)Blended Attacks which use several transmission methods at the same time, for example, combining the propagation methods of viruses and worms. (Souppaya & Scarfone, 2013)

To curb the increasingly pronounced cyber security problems, a variety of security technologies are used to improve security outcomes, among them, Intrusion Detection Systems, Intrusion Prevention Systems, Firewalls, Commercial antiviruses (AV), etc. (Jardine, 2020). As protection from these threats, the majority of legitimate users make use of anti-malware software products sourced from the different companies specializing with these products. Typically, until very recently, the main detection method employed in the available and heavily employed solutions, is the signature-based malware detection method. Using this method, it has been possible to recognize the different threats that have already been identified. A *malware signature* is a sequence of bytes, usually not long, each of the already identified malware tend to have a distinct signature from any other. A record of this signatures are maintained inorder to ensure that newly encountered files by the antimalware product, are correctly identified with an optimum accuracy level (Ye, et al., 2011). The existence of malware development toolkits allow even attackers without experience to easily develop and even modify malware samples. This is done to easily evade detection thus counteracting largely the war against malware.

Through the signature based detection method, plenty of malware already identified previously are detected and deleted or blocked. However, there are still plenty of malware files, for example the “zero-day” malware, that are generated or which have mutated. These ones have a tendency to escape detection by the scanning tools that employ traditional signature-based methods. These variants have been a key consideration in the antimalware industry, always looking for better ways of addressing them more effectively. This is because most approaches that have been in use, are mainly based on different versions of the signature-based methods (Ye, Li, Adjeroh, & Iyengar, 2017).

With 2,818 valid responses of a survey conducted by AV – Comparatives, which was voted the most trustworthy and reliable among the Antivirus testing labs, the most-requested for desktop security solutions for home users and business/enterprise products are antimalware softwares. Worldwide, the majority of the participants in this survey, who use Windows Operating System,use commercial antivirus products to protect themselves from malware (AV-Comparatives, 2020). Due to its ease of use even by people with little technological or security know-how, antimalware products have proven to be indispensable solutions in society today, where almost everyone needs to use technology.

Problem Statement

In order to prevent the occurence of malware incidences, it is clear that antimalware softwares are key. However, up to now, it has not been possible for these solutions to achieve total success in stopping all malware incidents, despite having a lot of work going into improving the effectiveness of the products in existence today. Due to the fact that the process of malware mutation has become very easy, there is a rapid growth of new malware file samples. This has made the fight against malware even more complicated (Ye, Li, Adjeroh, & Iyengar, 2017).

This research seeks to rate and measure the performance of existing antimalware algorithms, a key factor in aiding malware designers in making better decisions in improving existing antimalware products and in designing new ones.

Overall Objective

- To evaluate the performance of selected machine learning algorithms.

Particular Objectives

1. To investigate existing machine learning algorithms that can be used for malware detection.
2. Evaluate their strengths and weaknesses.
3. o measure the performance of these algorithms in terms of accuracy, precision, recall rate and speed of operation.

Justification

The breakthrough in internet technology and computer networking have made high speed shared internet possible (During the ongoing global pandemic, this has heightened exponentially, with practically everyone needing to work or school online.) The effect of this development is the daily increase in the number of computer systems that have become susceptible or have suffered to malware attacks (Chakrabortya & Dey, 2016). Security analyzers are constantly competing with malware scholars as innovation grows in the search for better detection methods. However, the detection methods they propose have not been sufficient while the evolving nature and complexity of malware keeps changing fast and getting harder to recognize. (Alireza & Rahil, 2018)

With so much work going into malware research and analysis in an effort to come up with a solution that is able to effectively detect and provide complete protection against the variety of malware in existence, we have not yet got to the ideal situation of a conclusive antimalware solution. This research assesses the performance of the variety of existing machine learning algorithms for malware detection. This will help in creating more robust and efficient algorithms that have the capacity to overcome the weaknesses of the existing ones. This will be a great contribution to future research as well as security solution developers in narrowing down their work in the right direction.

Literature Review

Introduction

A review of existing literature related to the research being carried out, helps to find out what has been done before in the same area, and what topics or aspects remain to be addressed.

It is important to evaluate previous work with a critical outlook, looking for related themes that show relationships of work done by different authors. This is done by reviewing and analyzing the literature in order to synthesize it into a well reasoned and logical explanation which then is able to justify the research being done, placing it in context (Oates, 2006).

Existing Work

Construction of Malware

In order to propose a solution, it is important to understand the construction, functioning and operation of the targeted malware. There exists programs that generate malware programs easily. These Malware construction kits operate in such a way as to combine features such as encryption and anti-debugging with metamorphic engines, allowing any person using them to create metamorphic viruses. Some of these kits have the capacity to generate a virtually unlimited number of metamorphic variants. There exists construction kits for viruses, trojans, logical bombs and worms. As these kits are able to come up with plenty of variants very easily, they present a big problem to anti-virus software. Besides, obfuscating code is sometimes applied to produce code that can then not be understood. This is opposed to recommended software engineering practices, although sometimes software developers perform code obfuscation to make it difficult to reverse engineer an attack. Those who create viruses often use the same techniques of code obfuscation to generate metamorphic variants of a virus (Attaluri, McGhee, & Stamp, 2009)

There are two ways of detecting malware by the different security solutions in existence today- network-based detection and host-based detection (Chiueh, Tzi-cker, Symantec). Different techniques are then employed to actually detect the malware. The most popular techniques employed by antivirus technologies to detect malware include – (1) signature detection – which is still the dominant approach used in AV industry, because of its low false positive rate. It is strong but is not able to detect new malwares. Signature explosion, also creates performance overhead and bandwidth cost problem, (2) checksum which is able to detect new malwares but has a high rate of false positives, (3) heuristic analysis which is able to detect new malwares but it's very costly and it is also unproven, and (4) virtual machine execution which is able to detect encrypted viruses but is also costly. Todate, the most applied detection method is signature scanning. This is a fact known to those who create malware and for this reason, they tend to pay a lot of attention to ensuring as much as possible that their malware escapes signature detection. Often they also try to hide the malware signatures to avoid detection. Although artificial intelligence has been employed lately in the existing anti-malware, they are an extra layer above the signature detection.

In the very recent past, the sophistication, severity and number of malware attacks, as well as the cost of malware suffered by the world economy have been growing rapidly. Attacks with these kinds of software,which include ransomware, have horrible effects and cause a lot of material damage to individuals, private organisations, and governments' assets. It is therefore important that malware is detected before they destroy valuable assets in organisations. There are still large gaps however, in the research area of malware detection and prevention (Aslan & Samet, 2020).

Technical controls are helpful in addressing cyber threats vulnerabilities, but as yet, there is none that has managed to provide absolute protection. (Butavicius, et al., 2020).

According to some content available on the VirusTotal website, it is evident that upto now, there is yet no available solution against malware that offers 100% success rate in the detection of viruses, malware and malicious urls.

Signature Based Detection

Signature based detection method works by maintaining a database of signatures already identified as malware. For this to be 100% effective, it will be necessary that the signatures of all the malware being released at every moment, be identified and thus the database updated with this information. This is why in this regard, Chiueh, et al of Symantec recommend that since it is very likely that in terms of numbers malware now must be many more than the goodware. This could imply that for a signature based detection system, maintaining the signatures of goodware would be more effective than keeping a database of malware signatures which are being developed at a faster rate.(Chiueh, Tzi-cker, Symantec). If the practicality of this fact is verified, then this could be a key design idea for the malware detector being modelled, thinking of signature based detection, also because many of the malwares created today, have the capacity to mutate and change signature very quickly.

An empirical study done (Bishop, Bloomfield, Gashi, & Stankovic, 2012), involving 32 different antimalware products and working on one thousand and six hundred malicious files collected from a remotely accessible honeypot that had been deployed for one hundred and seventy eight days, revealed that although some Antivirus products detected most of the viruses, none of them was able to detect all the viruses in the study. The detection failures in this case were due to a lack of the particular signatures in the antivirus products, and also due to regressions in the ability to detect previously known viruses, perhaps because some signatures have already been deleted. This is still applicable today.

According to a study done on improving security in reference to Commercial AV (Bishop, Bloomfield, Gashi, & Stankovic, 2012), each malware sample was sent to a given AV product for several consecutive days after it was first uncovered in the SGNET honeypots. This was done for a maximum of 30 days. Signature-based antiviruses should eventually detect all the malware present, as long as these identified signatures are all integrated in the system. The speed at which these updates are availed is an important factor for signature based detection.

It is not unusual for malicious attackers to regularly evaluate their products on anti-malware software to establish whether or not their malware will be detected. This attacker-driven anti-malware testing is something defenders would ideally want to limit. Given that anti-malware products must be widely distributed to be commercially viable, it is not feasible to prevent attackers from running them.

A study was done to evaluate the possibility of reducing the effectiveness of attacker tests. For this purpose a game-theoretical model was developed. Important factors under consideration were coverage and detection timelines which were reduced or increased alternatively while effects were being observed. When the coverage was reduced, the response would be slower and the attackers would then find it difficult to know whether their malicious files would be detected. This would also reduce the level of protection provided by the antimalware for legit users. This demonstrated that it is not worthwhile for antimalware vendors to maximize coverage and detection time. This is an important insight for the design of anti-malware products and testing methodologies employed. (Moni, Salahudeen, & Somayaji, 2015)

Static Analysis and Dynamic Analysis

In malware analysis, static analysis is a method used to examine and inspect malware samples without having to execute them. This is done by inspecting the sample files and identifying external features that are obvious. possible identified features could include: header information, packer detection, strings, fingerprinting and import functions etc. However, this method is not able to detect malware files which are obfuscated and also those that are polymorphic. The work around for this is to carry out the analysis dynamically, by executing the suspicious files in a controlled or virtual environment. During the execution, it is important to monitor run-time activities and the general behavior of the system. Important aspects to be monitored include changes in the registry, activity captured on the network, changes in the file system etc.

Although the dynamic analysis of malicious files is more effective when compared to the static one, it consumes a lot of time and other resources. Another drawback of this method is that the malicious sample files which are executed in the virtual environment could behave abnormally as opposed to when executed in a physical system. This is so as not to be identified. This happens in order to deceive the malware analyst into believing that the file is clean. The developers of these malicious files, often are able to build their files to be able to differentiate a virtual machine environment from a physical one.

It is clear therefore that as research done indicates(Gandotra, Bansal, & Sofat, 2016), most of the existing anti-malware solutions like Intrusion Detection Systems, Intrusion Prevention Systems and Antivirus softwares are often not able to detect unknown malware. This is because their mode of operation is signature based. To counteract this problem, the researchers of this project propose using machine learning approaches which can be built counting on features extracted through static and dynamic analysis of malware.

Due to pros and cons of static and dynamic analysis methods, and to improve malware detection accuracy levels, features derived from both dynamic and static analysis of suspicious files should be combined. At the same time, an important consideration is the number of features considered, to avoid building a solution that lasts very long thus reducing efficacy. Using very many features for the machine learning building model can take more time and in this way, reduce the speed of detection. This is why it may help to establish which of the existing malware features have more weight in the detection of malware. Making a selection of these features facilitates for the classification model to be built in little time and with the best accuracy levels. Feature selection helps to identify the most influential features in a dataset in giving optimum results in terms of accuracy of the prediction made by the model as well as time taken to build it.. Once the most influential features are identified, then the other features can be discarded.

This research quoted above, (Gandotra, Bansal, & Sofat, 2016), proposes a way of detecting zero-day malware. This is done by extracting the influential features of a dataset. This is done from static and dynamic analysis as well as employing WEKA machine learning library. The experiment demonstrates that Selecting features facilitates faster machine learning solutions without affecting the accuracy levels. This aspect however is beyond the scope of this project.

Behaviour Based Detection

Apart from signature based detection methods, the other main category of approaches is behavior-based methods (ME, M, & N, 2015).Although signature based detection as explained earlier is easy to execute, very accessible, able to identify known malware pretty fast and is also able to find complete malware information, it doesn't easily detect the polymorphic malwares. It also tends to replicate information in the huge database of malware signatures. Behavior based detection on the other hand is able to detect obfuscated attacks, is able to detect data flow dependency, and is also able to detect polymorphic malwares, however the challenge comes in when there is not sufficient storage space, because it tends to take up a lot of storage space while analysing the behavior, because a lot of space is used up. This method also takes up a lot more time (Alireza & Rahil, 2018). Behavior-based detection approaches have the advantage of being able to detect all of the suspicious files and programs according to their calls' behavior. The accuracy levels of predicting malware detection are therefore higher.

Machine Learning Detection Methods

Regarding the machine learning strategies used in malware detection, protection of obscure malware is a key concern in recognizing malicious files or programs (Sun, Z, Q, W, & Y, 2016). The machine learning methodologies are classified into either supervised or unsupervised methods.

Binary features and assembly features are the two main applications of machine learning methods in malware detection. Together with the API calls features approach, they predict and detect malicious files.

As earlier mentioned, the machine learning methodologies apply either the static or dynamic malware analysis of suspicious files features, in order to look for malicious applications (T, Cislak , Ochoa, & Pretschner , 2017). It's more effective to employ dynamic analysis when the malware features under consideration are opcode sequence and API calls (Damodaran, Troia, Visaggio, Austin, & Stamp, 2017).

Data Mining Techniques

For the data mining methods, the process of detecting malware usually involves two consecutive steps. First the features are extracted then the samples are classified or clustered. (YANFANG et al., 2017). In the first step, a variety of features are extracted statically and/or dynamically to help understand the characteristics of the suspicious files. The features extracted could include behavior of the programs, binary strings, API calls, etc. After this extraction, the samples are automatically categorized into different classifications, by using machine learning classification or clustering. The extracted features are used as the basis for this classification or clustering. The resulting data-mining-based malware detectors will differ in each case depending on the data mining method employed and the representation of the features extracted.

This method has been recently employed by some of the vendors of commercial antiviruses. These are: Kingsoft's security products [Ye et al. 2009c, 2010], Comodo AntiVirus products [Ye et al. 2011; Chen et al. 2015; Hardy et al. 2016], Symantec AntiMalware products [Chau et al. 2011], and Microsoft Internet Explorer [Stokes et al. 2012]. All of these have deployed data mining methods in the process of detecting malware. These methods as explained above, are able to classify, even file samples that were not previously seen, and thus identify them as a particular kind of malware and even inferring their signatures.

According to a research done (Alireza & Rahil, 2018), among the data mining techniques that can be used in malware detection, *malicious code based on API* technique based on data using Decision Tree, SVM and Random Forest classification methods, analyzing the data dynamically achieved the highest accuracy in malware detection, followed by *Feature Extraction* method in cloud, using the same classification methods, on a much larger random data set, and using static analysis, with most of the data mining techniques giving over 90% accuracy. However, of importance to note is the fact that these data mining techniques are unable to detect malware in the cases of data hiding/steganography, metaheuristic detection and real time malware detection. This work further recommends SVM method as being the best method when accuracy in detection is a key consideration, and when malware signatures are being used as the key data mining feature.

Currently there is a lot of discussion and research going on regarding deep learning malware detection as being a probable way of waging the malware war effectively, although there is little documented or done in this line.
Meta-heuristic algorithms are considered to find the optimal signature detection for polymorphic malware.

In measuring effectiveness of malware detection techniques, recall is a function of the sample match cases that are correctly classified. Precision is a function of the samples classified as positive and actually they belong to that classification.

Research Gap

Although there has been research done in the line of malware detection, it applies very specific and related approaches to detect malware. By counting on and putting together feedback from prior research, this research work aims to scrutinize the performance of a much broader selection of algorithms, based on very specific metrics including precision, accuracy, recall rate, speed of operation and the false error rate. The algorithms considered in this research are those recommended for the kind of data used here - binary malware data (Sarker, I.H. (2021)).

Approach used to deal with the Research Gap

This research compares the performance of machine learning algorithms by first using Cuckoo sandbox to analyse suspicious files, then submitting the resultant report to WEKA for machine learning training of detection models. Specific metrics were used as a measure of comparison.

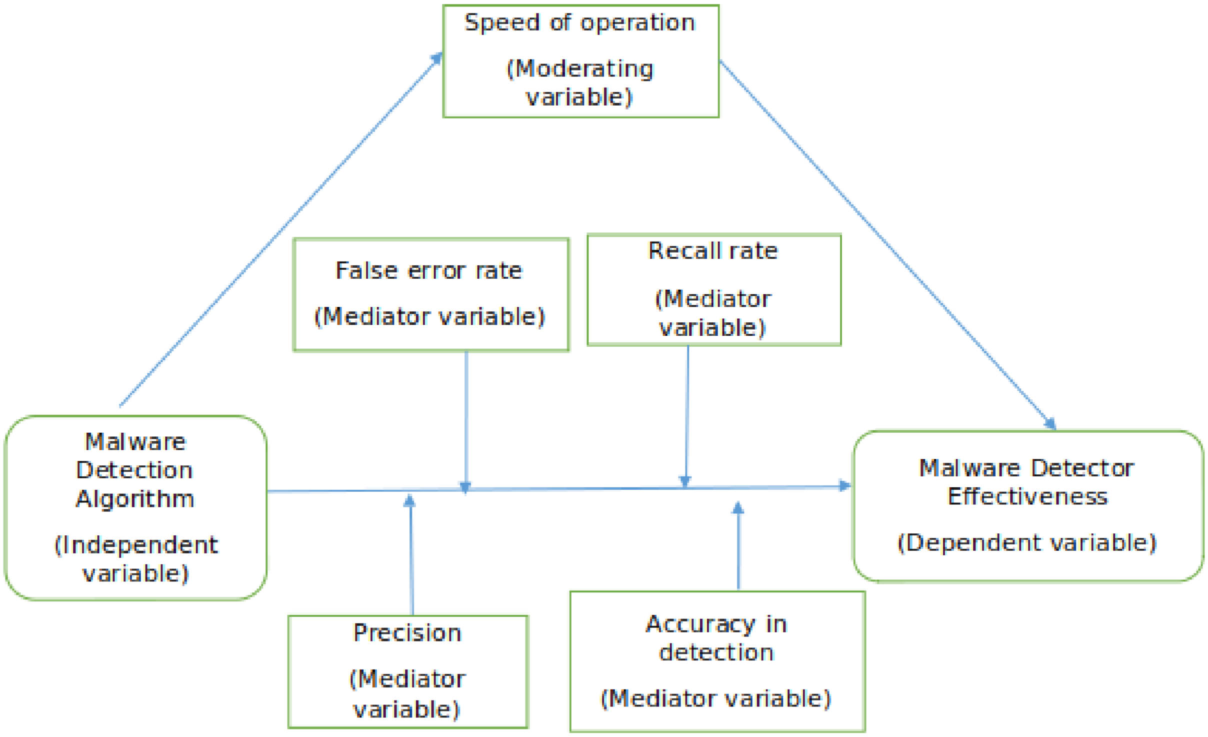
Research Questions

1 What are the malware detection accuracy levels of each of the algorithms available given the same malware data set, consisting of a variety of malware?
2 What is the recall rate of each of these algorithms?
3 What is the speed of operation of each of the algorithms?
4 What is the ideal best approach to deal with malware detection based on the above important factors?

Conceptual Framework

This section makes explicit the structure of the research topic under consideration and the process undertaken. It therefore makes clear such things as: the different factors that comprise the topic; the logic or thought process of the topic; the way of tackling the research question(s) (that is, the combination of strategies and methods adopted - often called the research methodology); the approach used to analyzing any generating data (for example, quantitative analysis, which uses mathematics and statistics, or qualitative analysis, which uses thematic approaches); the approach used to design and create any new IT product (for example, the systems development methodology followed, or the genre used); the approach used to evaluate the research (for example, whether the focus is on on technical accuracy, greater understanding, increased efficiency or aesthetic criteria). Much of the conceptual framework is derived from, and justified by, a study of the literature.

Important Variables in this research include malware detection algorithm as the independent variable that is not changing. False error rate, accuracy in detection, precision, and recall rate are mediator variables, in the sense that they help to explain how or why the independent variable affects the dependent variable. Speed of operation is a moderating variable, in the sense that it is an interaction effect between the independent and the dependent variables.
The conceptual framework explaining the logic of this research is as shown below:



Research Methodology

Introduction

The research Methodology defines the process that a researcher will take to conduct its study. It is important that it is clear and complete, so that someone reading it would be able to replicate it. The details of the research methods, the research strategy, and the study environment, the size of study, participants or subjects should be included here. The variables under observation including expected outcomes, should be included as well (Edgar & Manz, 2017). This research used the applied experimentation research method.

General Methodological Approach

Applied experimentation methodology used in this research, helps to understand the performance or effectiveness of a solution(Edgar & Manz, 2017). In this case, the performance of a selection of recommended existing machine learning algorithms were assessed for malware detection .

Experimentation is one of the strongest methods we have to understand the behavior and response of a system under varied conditions (Edgar & Manz, 2017). All applied experimental methods leverage controlled tests that attempt to capture real-world behavior to determine how the applied system behaves. For this, this research used Cuckoo Sandbox to analyse suspicious files, the leading open source automated malware analysis system, which provides a realistic but isolated environment.
The generated reports were then submitted to WEKA, a tool containing machine learning algorithms or capable of being installed with these algorithms, which can then be used to train and test models.

With applied research, there is always an implied hypothesis of the solution being tested, which will solve some problem. In our case the implied hypothesis is the proposal that the effectiveness and efficiency (resource utilization) of malware detection algorithms is determined by the precision level, accuracy level, speed of running the algorithm, the false positive, false negative and recall rates.
The dependent variables for applied experimentation are always defined around performance or the effectiveness of the system under test in solving the problem, as is the case with these metrics. Finally, applied experimentation is about understanding the behavior of a system that exists, which is the case of this research.

Research Design

An experiment needs to follow a specific design that can easily be followed to replicate and reproduce the same effect. An experimental design lists the variables involved (both independent and dependent) - malware detection algorithms as the independent variables and malware detection effectiveness as the dependent variables. The algorithms to be used are the recommended ones from previous research from the various malware detection techniques.

The experimental setup used Cuckoo sandbox, configured on a virtual environment due to the malicious nature of the specimens being used, this will be done on a high specification box to avoid unnecessary delays, and carried out by the researcher.

The experimentation process consisted of iteratively resetting cuckoo sandbox to process suspicious files in the standardized virtual machines. The resultant reports were applied to different algorithms in WEKA iteratively, using the same settings, and while only iterating on the algorithms. These experiments were run using the same data set of malware. For each of these processes, the important metrics mentioned earlier were observed keenly and recorded, as well as keeping track of the variables.

This project is pure research.

Data Acquisition

During data collection, it is important to keep in mind the research questions and the approach to take to answer them. Although these questions, the ways of analyzing the data or coming to conclusions could change in the course of the research, keeping the questions in mind helps the researcher to gather data that is relevant to the line of enquiry under consideration (Edgar & Manz, 2017).

Scope –since this research focuses mainly on malware, the main source of data was from a large collection of malicious samples targeting Windows OS collected from Virusshare (https://virusshare.com) database which is available in the public domain after free registration. The other data source that helped ensure availability of a collection of all categories of malware will be “TheZoo”, a simple tool that is installed in linux to facilitate downloading live malware for educational purposes.

The objective of this data collection is to be able to facilitate and point towards answering the research questions.

Challenges encountered during data collection include: ensuring a secure environment as this is very risky data - malware. This was achieved by using a virtual environment and configuring a host only network for the virtual machines, thus isolating communication to that subnet alone.

Population

This research used Case Control type of observational study, whereby we shall have a case group of the malware acquired from the data sources – of malware infections. The second group of files, the control group will consist of files from the different versions of Windows. Due to the nature of the datasets under observation, this research uses a non-probabilistic sampling method, because the population distribution is unknown. The specific method of choice is quota sampling. The population under observation was subdivided into either malware or not, depending on the characteristics exhibited.
After categorization, these binary blobs are then mathematically analyzed.
Categorization is a method of mapping qualitative data onto a nominal scale for analysis. Sufficient data was required to generally answer the research questions. The study was tailored to the information collected, of 373 files analysed on Cuckoo Sandbox. It was desirable to collect enough artifacts of different types to paint a complete picture of the environment. For this reason, some windows clean files were also analysed.

Sample size

The factors that determine effective sample size include population size – the approximate or absolute number of members in a population, power which is the likelihood that a study will accurately detect a true effect (adequate power in an experiment will mean that the likelihood of achieving a false negative is sufficiently low), minimum expected difference or also known as the effect size, estimated measurement variability, desired statistical power, significance criterion, and whether a one- or two-tailed statistical analysis is planned.
This research counts on a sample size of 373 samples of a binary data set. Although this may not be sufficient for a production solution, for purposes of the specified experimentation for the specified metrics, the size of the dataset will be sufficient. Larger data sets, on the other hand, would require much higher system resources. However, for the purposes of this specific experimentation, a higher dataset would give the same results.

For the machine learning model, the top 9 recommended algorithms for binary data sets were tested (Sarker, I.H. (2021)

Evaluation and Validation

Validation testing method of applied research helps to evaluate solutions in controlled environments to see how they behave under varied but realistic conditions. Validation Testing is a rigorously controlled process to investigate the performance of an application of science to solve a problem. It is done in a fully controlled environment to ensure that only certain aspects of the problem and solution are studied together. As the project is using existing machine learning algorithms that have already been tested in research and recommended, this research will be affirming and at the same time building on the existing knowledge.

Applied experimentation focuses on how well an engineered system solves a problem. By standardizing one's testing processes and measurements, it becomes possible to evaluate the best solution under different conditions. This provides decision makers actionable and measurable information on what approaches to invest and deploy in real-world situations. Apart from the specific algorithm procedure, all the algorithms will be handled in the same way and using the same data set, to ensure a standardised and therefore objective feedback.

The training data was used for the machine learning algorithms to build the models. The test data was used to test the models so built. Validation of the models was done by using cross validation technique which is used for evaluating the results by generating independent datasets. In the experiments, 10-fold cross validation was used, as it is a well-accepted method to evaluate the predictions over unseen data for malicious and benign files. The algorithms were evaluated by using following performance measures which are computed using the fields of confusion matrix.

Reliability and Validity

This research used a standard tool for experimentation, Cuckoo sandbox, which is a well known and the leading open source automated malware analysis tool, widely used for research. The true feedback from this system is known to be reliable and valid. The dataset used was from well known malware sources as well as legit Windows systems files, making it reliable and valid. The data analysis procedures of using cuckoo and WEKA Libraries, have also been used in other research work. This is good ground to count on their reliability and validity.

Research Ethics

It is important that the objectives of researchers should be to make the world a better Place. Their research should lead to improvement in the safety and security for users of cyberspace. However, science can be used incorrectly such that it causes more harm than good. It is the duty of all scientists to evaluate the worth of the research, not just from a perspective of novelty and impact, weighing the usefulness of the results with the cost of collecting it. Every research field is guided by a set of ethics on what is and is not acceptable research. Morality is a set of guiding principles in determining what is right and wrong. Scientific ethics are a set of accepted guidelines on what constitutes justifiable research actions and those behaviors that are intolerable. Each field of study develops their own ethics (Edgar & Manz, 2017).

Important ethical issues in this research includes:
1 Vulnerability disclosure: - Once cyber security researchers discover vulnerabilities in cyber space, ethically following the personal rights perspectives, they should not disclose vulnerabilities publicly until there is a fix to prevent exploitation.
Following release of new malware this research seeks to provide the best approach of effectively detecting malware as soon as availed, thus being able to address this ethical gap by providing timely security fixes that could otherwise lead to misuse by malicious cyber actors.

2 In carrying out this research, the people involved had a great understanding of the existing cyber vulnerabilities that they seek to seal. It is possible to unethically or maliciously utilize these vulnerabilities to cause harm to other people or organizations for some other selfish gain. These people will be instructed against any such misuse of knowledge.
3 It is important that the malware and any other data collected for this research is used only for the intended purpose, while at the same time honoring the Local data Laws (GDPR) of the country of use.
4 Against plagiarism, this research acknowledges contributing research work in the Bibliography.

Discussion

This part explains the project execution work.
This work is a pure research that assesses the performance of existing malware detection algorithms in a bid to gain and recommend useful insights in building or improving the efficiency and effectiveness of malware detectors.

From the research questions that are highlighted in the research design, this study pursued to evaluate the performance of malware detection algorithms, using the specified metrics. This was performed by using Cuckoo sandbox to process then applying selectede algorithms using WEKA machine learning tool

1. Cuckoo sandbox.

The Cuckoo sandbox is an open source automated malware analysis system. Once you configure it, through a tedious way, you can throw files at it and it will give a detailed report including about the files.

To set up the sandbox, you need to install all the required dependencies, as outlined in the website of this open source project - https://cuckoo.sh/docs/. This sandbox is an ongoing open source project. Therefore, it is important to use the specified versions of the dependencies of the specific version of cuckoo you are working with for it to function as desired. Once you have installed the dependencies, install the sandbox, and then follow the detailed guidelines of customizing each of the modules according to your requirements. Among the modules that have to be customised according to your preferences, are auxilliary modules, machinery modules, analysis packages, processing modules, and reporting modules.

As this project is dealing with malware, it was appropriate to use a virtual environment. This was achieved by setting up a a Linux host machine using Ubuntu 18.04.5 LTS. In this environment, was installed Oracle Virtualbox, 5.2.44, as recommended in the cuckoo version in use documentation quoted above, to ensure compatibility.

I. Virtual box

This virtual box was set up as the Cuckoo Laboratory. In it, two virtual machines were set up:

(a) The first one was set up using Ubuntu 18.04.5 LTS Operating System and assigned more RAM. Cuckoo sandbox and its dependencies were configured on this machine.

(b) The second was set up using Windows 7. This is the machine on which cuckoo runs the submitted files and analyzes them and their behavior. It was installed with a few other applications. Some documents and other files were also stored on this machine, so that it mimics as much as possible a normal machine in use. This is important because some malware are known to be able to recognize isolated sandbox environments and when this is the case they hide, and thus are not correctly identified by the sandbox. Mimicking an ordinary machine is very helpful in ensuring that the malware operate as they are designed to function.

II. Virtual Network for the Laboratory:

The first network adapter of each of the two machines was set up on the same sub net.

The two machines are attached to the *Host only Adapter* network option of the virtualbox settings . This ensures ease of communication between the attached machines, thus facilitating machine 1 to inject the submitted malware and analyse their activity on machine 2.

Adapter 2 of the cuckoo machine was enabled and configured as NAT. This enables Cuckoo to access the internet in its operation and be able to consult configured reference tools like Virus total. At the same time, as these machines are on a different subnet, they are prevented from communicating with and thus infecting the host machine and network.

To enable the communication in this set up, it was necessary to share the folder containing the *vboxmanage* file, so that it is accessible by the three machines and thus cuckoo was able to run appropriately.

2. Data Collection

The data used for this project included malware files and clean windows files. By installing **theZoo,** you receive live malware in your machine which you can then submit for analysis. TheZoo is a repository project of live malware, created for educational purposes only, to facilitate malware analysis. To install theZoo repository on your terminal, go to github and clone this link in your terminal and type the command
https://github.com/ytisf/theZoo.git, then cd theZoo then pip install –user -r requirements.text
Once installed, you will have a folder named theZoo with the malware files which you can then submit for analysis.

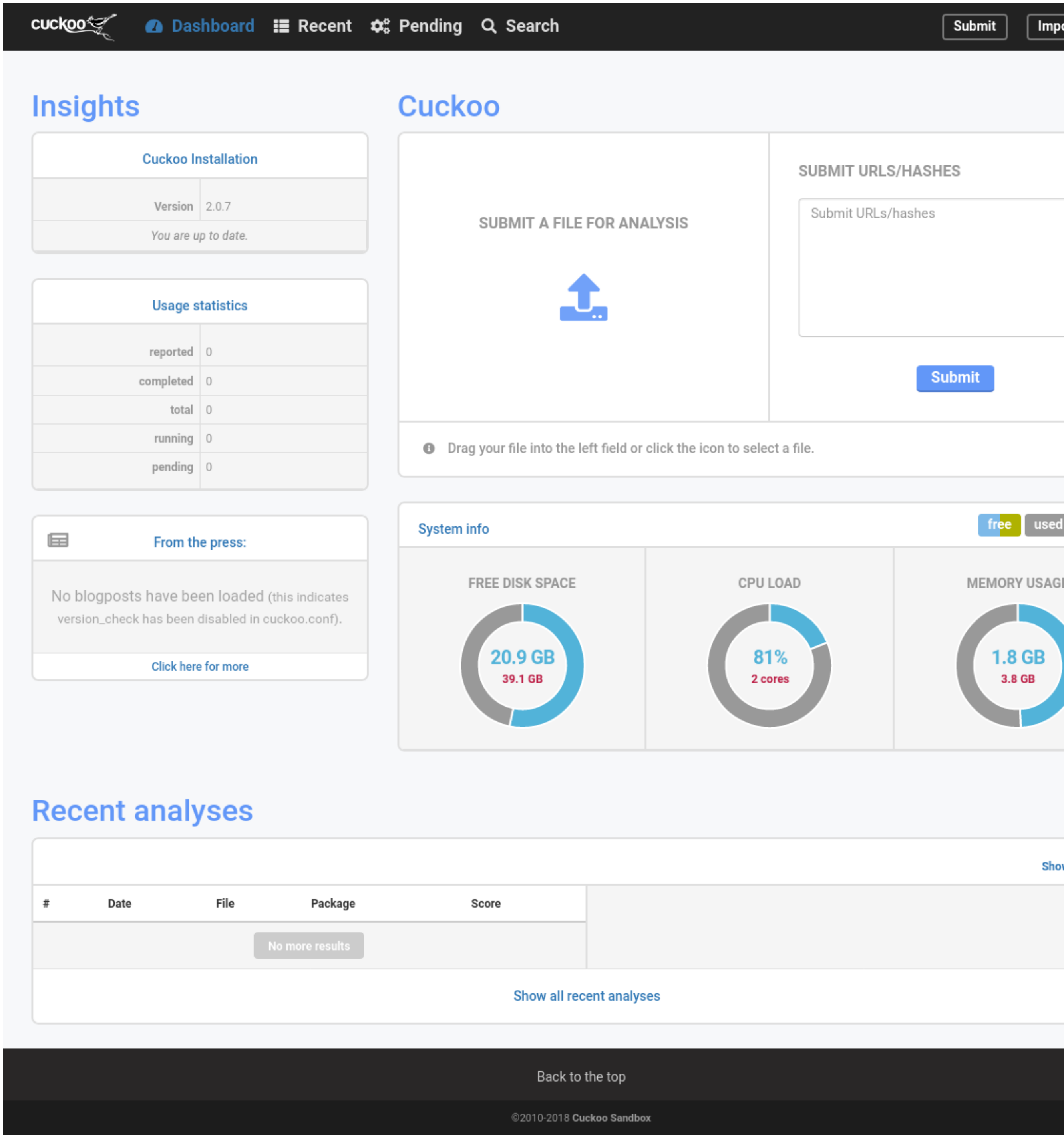
The other source of data was virusshare website. Inorder to obtain a dataset, an account was created on the virusshare.com website. Once the account was approved by the managers of the database, it was possible to login and select a dataset of choice. The available datasets are zipped files, majority of them very big files. For this project a small dataset was selected and downloaded. After downloading the zipped file was submitted to the cuckoo sandbox.
Together with these some clean windows files were also included as part of the

data.

3. Data Analysis

I. Cuckoo Sandbox Analysis

In order to be able to analyse data using cuckoo, it is important to: Start the machine running cuckoo, start the windows machine on which the files will be executed by cuckoo, start the cuckoo by running this command on the terminal: *cuckoo,* then *sudo cuckoo rooter* on the admin account of cuckoo machine. Open another terminal and start the web platform using the command: *cuckoo web.* When this starts, you can now run https:127.0.0.1:8000 the default cuckoo web address. Below is a screenshot of the just loaded cuckoo web homepage:



Cuckoo Installation	
Version	2.0.7
You are up to date.	

Usage statistics	
reported	0
completed	0
total	0
running	0
pending	0

From the press:

No blogposts have been loaded (this indicates version_check has been disabled in cuckoo.conf).

Click here for more

Drag your file into the left field or click the icon to select a file.

System info

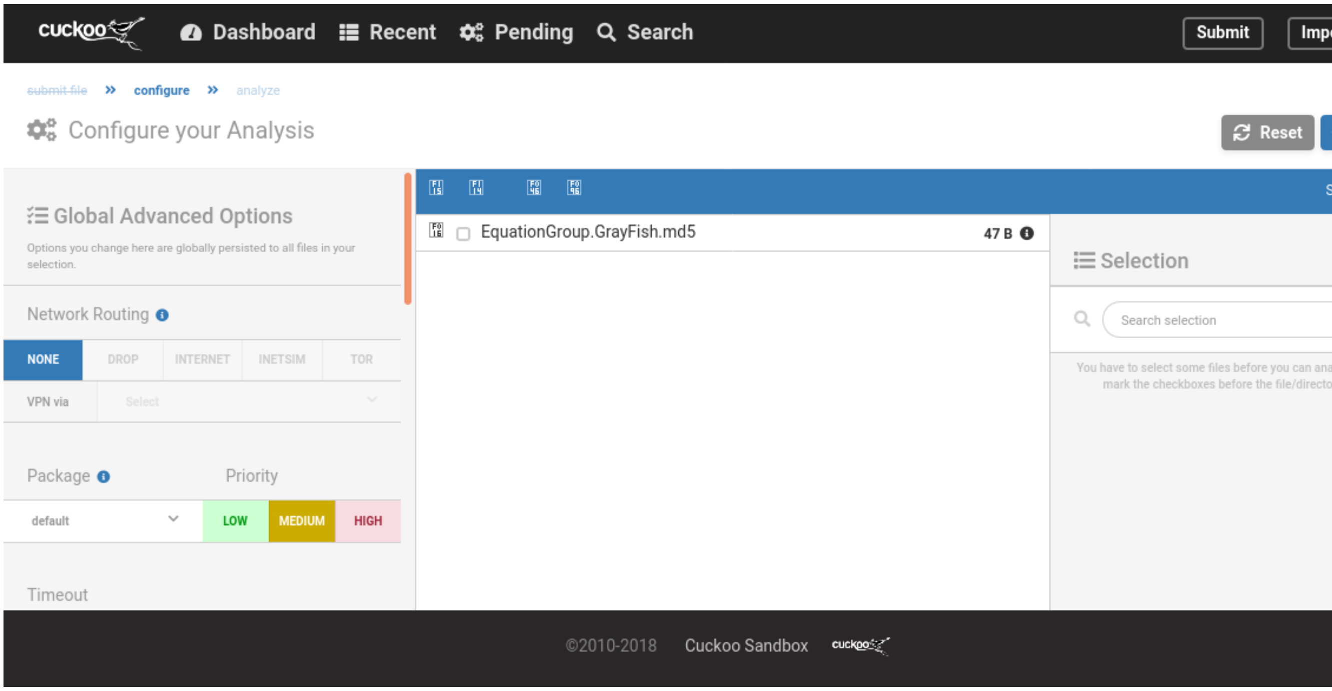
FREE DISK SPACE	CPU LOAD	MEMORY USAGE
20.9 GB 39.1 GB	81% 2 cores	1.8 GB 3.8 GB

#	Date	File	Package	Score

©2010-2018 **Cuckoo Sandbox**

Fig 1. Cuckoo sandbox web homepage

Dump/ drag and drop your data on the sandbox web platform for analysis. Once you dump your files for analysis, you will need to set your preferred configuration for the current analysis task by choosing your preferred options as shown below.



Configure your Analysis

Global Advanced Options

Options you change here are globally persisted to all files in your selection.

Network Routing: NONE | DROP | INTERNET | INETSIM | TOR

VPN via: Select

Package: default

Priority: LOW | MEDIUM | HIGH

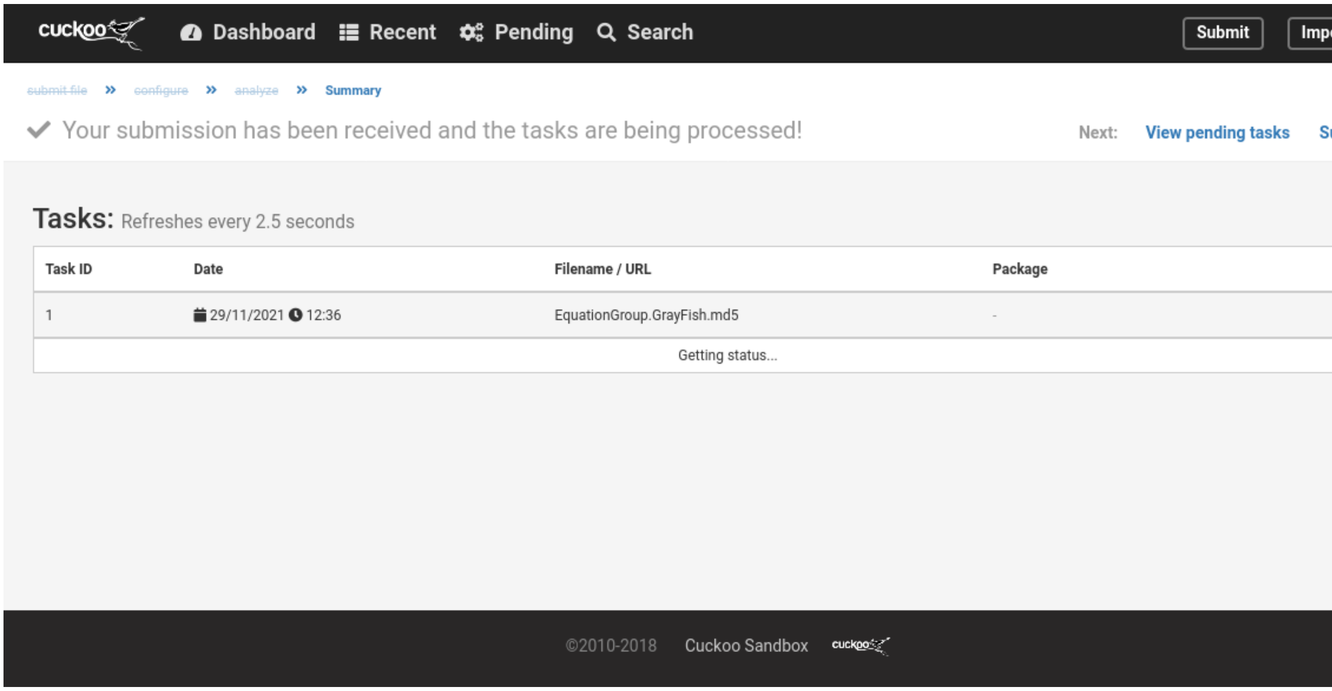
Timeout

□ EquationGroup.GrayFish.md5 47 B

You have to select some files before you can ana mark the checkboxes before the file/directo

Fig. 2. Cuckoo Analysis task configuration of preferred options.

Once done with the configurations, submit the job for analysis by clicking the submit button. If the job is successfully submitted, you receive a successful submission and you have to wait for the analysis to complete.


cuckoo
Dashboard
Recent
Pending
Search
Submit
Imp
~~submit file~~ » ~~configure~~ » ~~analyze~~ » Summary
Next: View pending tasks S
©2010-2018 Cuckoo Sandbox cuckoo

✓ Your submission has been received and the tasks are being processed!

Tasks: Refreshes every 2.5 seconds

Task ID	Date	Filename / URL	Package
1	29/11/2021 12:36	EquationGroup.GrayFish.md5	-

Getting status...

Fig 3. Screenshot of successful file submission on Cuckoo sandbox.

Depending on the size of your data and the machine specifications, the analysis job could take some time to complete the analysis.

II. WEKA Machine Learning tool

The resulting json report from cuckoo is then converted into arff format and passed to the WEKA machine learning knowledge Analysis tool to train a machine learning model and test it against the project's metrics.

WEKA is a machine learning tool, consisting of a variety of machine learning algorithms. Using the ARFF viewer module of WEKA, you can convert your report into a format readable by WEKA, i.e. into a .arff file. Open the arff file and you can now apply the desired algorithms to train and test the model.

The algorithms that appear disabled, are not yet installed and therefore, it is necessary to install the module from the package manager of WEKA, in order to apply it to the data-set. In order to use the latest version of an existing algorithm, it is necessary to upgrade it from the same module. If you need to use an algorithm that does not appear in the list of the relevant WEKA algorithms, you could also download it into WEKA using the Package Installer module in WEKA.

III. Algorithms Analyzed

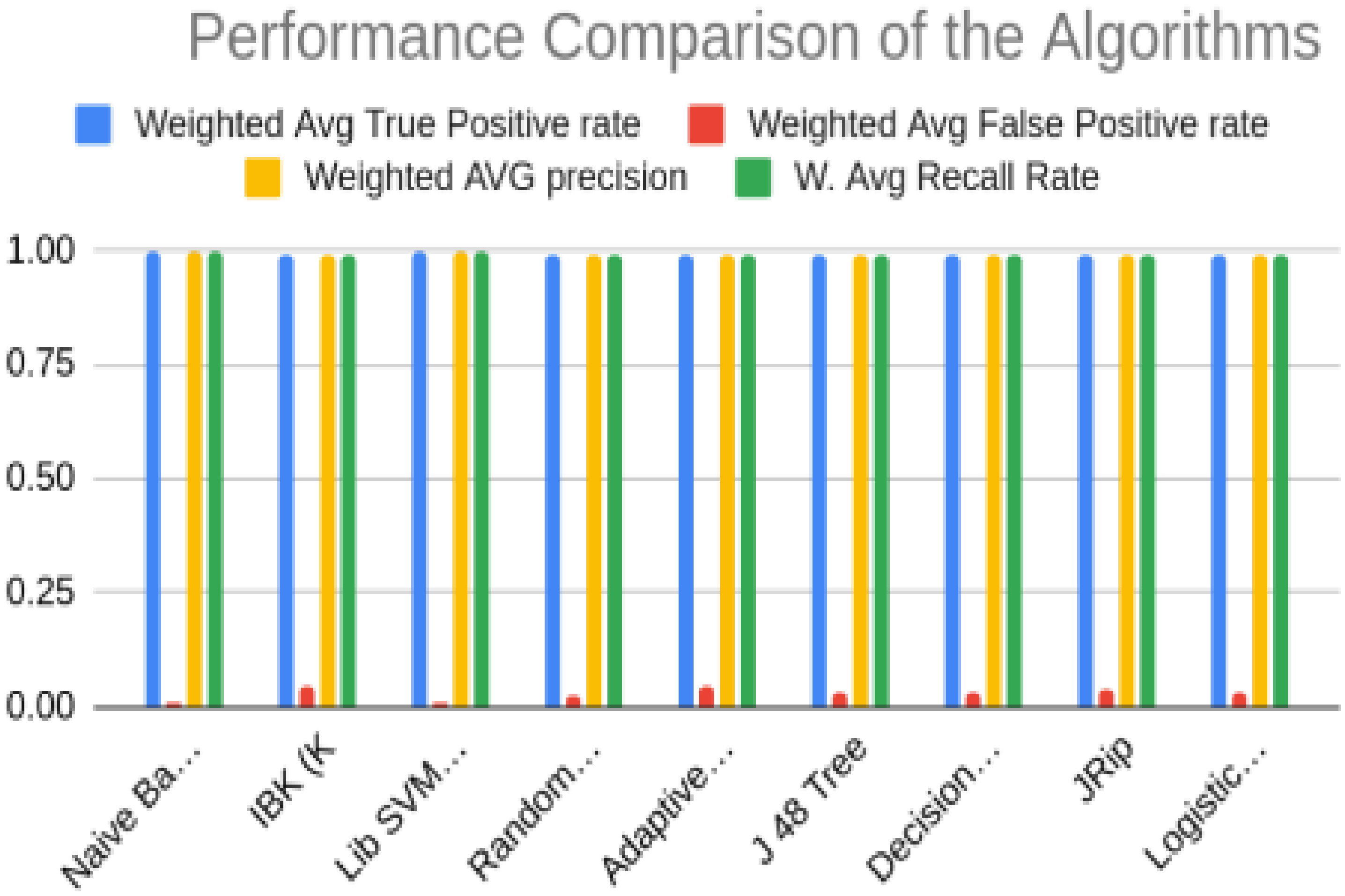
In a machine learning task, the nature of the dataset is a basic consideration that should inform the choice of algorithm made. The nature of the dataset to be applied on WEKA is binary classification. This refers to the classification tasks having two class labels, such as true and false or Yes and No (Sarker, 2021).This is the case of the dataset in use in this project, as it is either malware or non-malware. For such a dataset, the recommended classification algorithms were used for this research. They include: Naive Bayes, K-Nearest Neighbor, LibSVM, Random Forest, Adaboost, J 48 Tree, Decision Tree, J Rip and Logistic Regression.

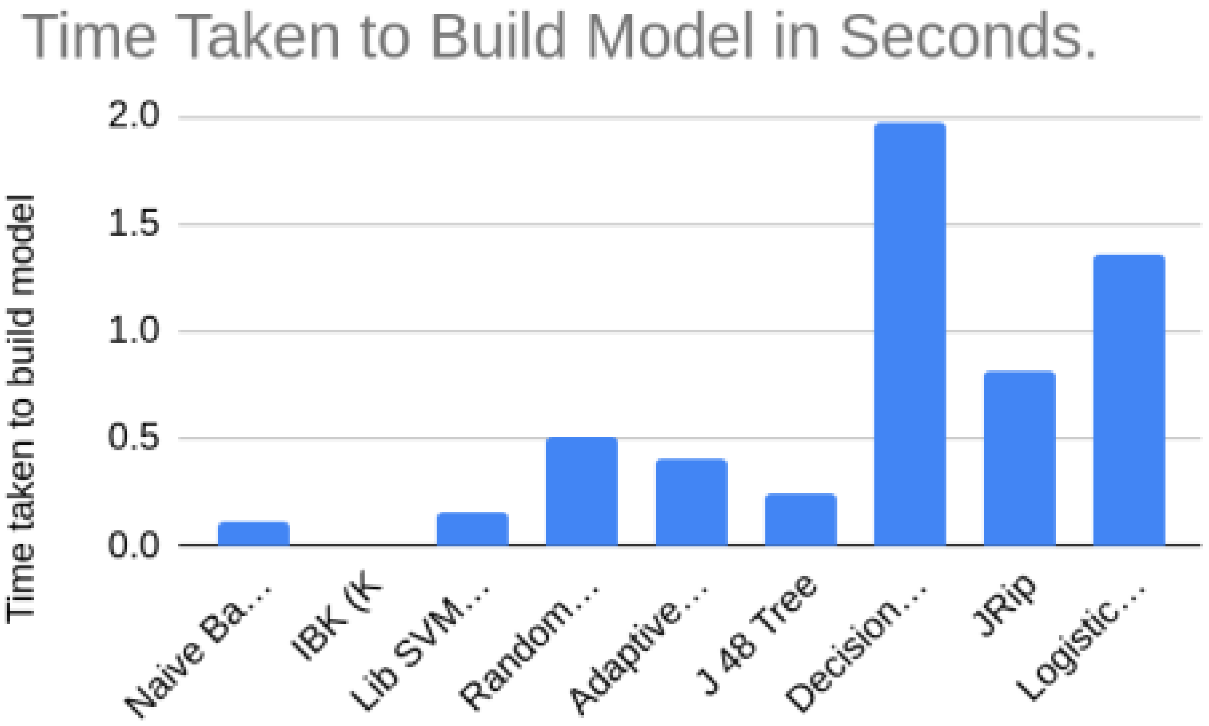
Results and Discussion

All the algorithms have been applied using a 10 fold *cross-validation* test option as a method of splitting the data between training set and testing set. This is a binary classification based on whether the instance is malicious or not.

	Naive Bayes	**IBK (K Nearest Neighbor)**	**Lib SVM (Support Vector Machine)**	**Random Forest**	**Adaptive Boosting (Adaboost/Meta Learning)**	**J 48 Tree**	**Decision Tree -Rule Based classification**	**JRip**	**Logistic Regression-LMT tree**
Instances	**373**	**373**	**373**	**373**	**373**	**373**	**373**	**373**	**373**
Attributes	**532**	**532**	**532**	**532**	**532**	**532**	**532**	**532**	**532**
Correctly Classified Instances	**371(99.46%)**	**368 (98.66%)**	**371 (99.46%)**	**370(99.20%)**	**368(98.66%)**	**369 (98.93%)**	**369(98.93%)**	**368(98.66%)**	**369 (98.93%)**
Incorrectly Classified	**2(0.54%)**	**5(1.34%)**	**2 (0.54%)**	**3(0.80%)**	**5(1.34%)**	**4(1.07%)**	**4 (1.07%)**	**5(1.34%)**	**4 (1.07%)**
Weighted Avg True Positive rate	**0.995**	**0.987**	**0.995**	**0.992**	**0.987**	**0.989**	**0.989**	**0.987**	**0.989**
Weighted Avg False Positive rate	**0.012**	**0.045**	**0.012**	**0.023**	**0.045**	**0.034**	**0.034**	**0.035**	**0.034**
Weighted AVG precision	**0.995**	**0.987**	**0.995**	**0.992**	**0.987**	**0.989**	**0.989**	**0.987**	**0.989**
W. Avg Recall Rate	**0.995**	**0.987**	**0.995**	**0.992**	**0.987**	**0.989**	**0.989**	**0.987**	**0.989**
Time taken to build model	**0.12 sec.**	**0 sec**	**0.16s**	**0.51s**	**0.41s**	**0.24 s**	**1.97 s**	**0.82 s**	**1.36 s**

Table 1. Tabulated Experiment Results.


Performance Comparison of the Algorithms
Weighted Avg True Positive rate
Weighted Avg False Positive rate
Weighted AVG precision
W. Avg Recall Rate
1.00
0.75
0.50
0.25
0.00
Naive Ba…
IBK (K
Lib SVM…
Random…
Adaptive…
J 48 Tree
Decision…
JRip
Logistic…



As demonstrated in the results above, choice of algorithm implemented in a machine learning solution is an important decision affecting the performance of the implemented solution. Although all the algorithms above are recommended for creating models of binary data, they have varied performance, some performing better than others, as regards the indicated metrics. When time of operation or resource utilization is an important factor of consideration for a binary dataset, K-Nearest Neighbor is the most efficient algorithm, followed by Naive Bayes and then SVM. This is clear in the results above.

When the accuracy of a model is very key for a binary dataset, then Naive Bayes and SVM will be the recommended algorithms followed by Random Forest and then K-Nearest Neighbor. The recall rate and precision were the same for every algorithm experimented. This happens when you use cross validation on the same dataset with the same amount of data (Saito T, Rehmsmeier M (2015)).

When one or both of these metrics are key for a model, then the preferred algorithms are Naive Bayes and SVM, followed by Random Forest.

Conclusion and Recommendation

Machine learning is today an essential and efficient way of improving performance of malware detection software. In the design of antimalware solutions, traditional detection methods need clear definitions of malware, to be able to detect existing malware. However, as thousands of malware are released daily, using already existing datasets to train detection models, will go a long way in detecting even zero day malware.

The algorithms of choice in this case should be informed as discussed above, by the predetermined priorities in terms of - accuracy, speed of operation, recall rate or precision. This will be a great contribution to future research as well as to security solution developers in narrowing down their work in the right direction.

The research work was very enlightening in malware analysis and also machine learning methods and algorithms

Future work

This research considers the performance of selected algorithms. Among the features used in the machine learning process, some will very likely have greater weight in determining whether an instance is malicious or not. In order to improve the performance of the machine learning models developed further, future work could be done to establish instance features that are more important and therefore could be given more weight in the development of the machine learning models.

Bibliography

1. Athira, B., Babu, N., & Krishnan, M. S. (2017). An Inspective Mode to Find the Optimum Antivirus App for Defending the Latest Banking Tordow. *International Journal of Pure and Applied Mathematics*, 79-88.
2. Attaluri, S., McGhee, S., & Stamp, M. (2009). Profile Hidden Markov Models and Metamorphic Virus Detection. *Computer Virology*.
3. AV-Comparatives. (2020). *IT Security Survey 2020.* Innsbruck, Austria: AV-Comparatives.
4. Bishop, P., Bloomfield, R., Gashi, I., & Stankovic, V. (2012). Diverse Protection Systems for Improving Security: a Study with AntiVirus Engines. *City Research Online:*. 5. Bitsch, F., Guiochet, J., & Kaâniche, M. (. (2013). Does Malware Detection Improve with Diverse AntiVirus Products? An Empirical Study. *SAFECOMP*, 94-105.
6. *Breadcrumb, Data Mining – Information Gain*. (n.d.). Retrieved from http://gerardnico.com/wiki/data_mining/information_gain
7. Butavicius, M., Parsons, K., Lillie, M., McCormac, A., Pattinson, M., & Calic, D. (2020). When believing in technology leads to poor cyber security: Development of a trust in technical controls scale. *Computers & Security*.
8. Chiueh, Tzi-cker, Symantec. (n.d.). *A Look at Current Malware Problems and Their Solutions, Symantec.* Retrieved from DEPARTMENT OF COMPUTER SCIENCE, SJSU: http://www.cs.sjsu.edu/~stamp/IACBP/IACBP08/Tzi-cker%20Chiueh/2008.ppt
9. Edgar, T. W., & Manz, D. O. (2017). *Research Methods for Cyber Security.* Cambridge, MA 02139, United States: Elsevier Inc.
10. Gandotra, E., Bansal, D., & Sofat, S. (2016). Zero-Day Malware Detection. *Sixth International Symposium on Embedded Computing and System Design (ISED).* IEEE. 11. Jardine, E. (2020). The Case against Commercial Antivirus Software: Risk Homeostasis and Information Problems in Cybersecurity. *Risk Analysis*.
12. Kritzinger, E., & Solms, S. v. (2012). A Framework for Cyber Security in Africa. *Journal of Information Assurance & Cybersecurity*.
13. L'evesque, F. L., Davis, C. R., Fernandez, J. M., Chiasson, S., & Somayaji, A. (2012). Methodology

for a Field Study of Anti-maware Software. Montr´eal : ´Ecole Polytechnique de Montr´eal.

14. *Modified edition of cuckoo*. (n.d.). Retrieved from https://github.com/brad-accuvant/cuckoo

15. Mohammed I. Al-Sale, H. M. (2018). On Studying the Antivirus Behavior on Kernel Activities. *ICIEB '18: Proceedings of the 2018 International Conference on Internet and e-Business*, (pp. 158–161).

16. Moni, T., Salahudeen, S., & Somayaji, A. (Performers). (2015, January). *The Malware Author Testing Challenge.*

17. Oates, B. J. (2006). *Researching Information Systems and Computing.* London: SAGE Publications Ltd.

18. Satzinger, J., Jackson, R., & Burd, S. (2002). *Systems analysis and design in a changing world.* Course Technology.

19. Schultz, M., Eskin, E., Zadok, F., & Stolfo, S. (2001). Data mining methods for detection of new malicious executables. *IEEE Symposium on Security and Privacy* (pp. 38-49). Oakland: IEEE.

20.

21. Souppaya, M., & Scarfone, K. (2013). *Guide to Malware Incident Prevention and Handling for Desktops and Laptops.* Gaithersburg, MD 20899-8930: NIST. 22. Sukwong, O., Kim, H. S., & Hoe, J. C. (2011). Commercial Antivirus Software Effectiveness: An Empirical Study. *IEEE Computer Society*, 63-70.

23. Mouhammd Al-Kasassbeh, Mohammed, S., Alauthman, M., & Almomani, A. (2020). *Handbook of Computer Networks and Cyber Security: Feature Selection Using Machine Learning to Classify Malware.* Switzerland AG : © Springer Nature .

24. Yanfang, Y., Tao, L., Donald, A., & Iyengar, S. S. (2017, June). A Survey on Malware Detection using Data Mining Techniques. *ACM Computing Surveys*, *50*(3). 25. Sarker, I.H. (2021). Machine Learning: Algorithms, Real-World Applications and Research Directions. SN COMPUT. SCI. 2, 160.

26. Saito T, Rehmsmeier M (2015) The Precision-Recall Plot Is More Informative than the ROC Plot When Evaluating Binary Classifiers on Imbalanced Datasets. PLoS ONE 10(3): e0118432.