Development and integration of machine learning and AI pattern recognition in malware detection: a quantitative and summative analysis of models.

Madeleine Rabitoy

Davenport University

Summer 2023

In fulfillment of the requirements for the Degree of Master of Science in Information Assurance and Cyber Security

Table of Contents

Abstract

Malware detection is a field that is constantly in flux. Increasingly organized and intelligent malware distributors and authors make the detection and prevention of malware infections more difficult now than it has ever been before, as does the proliferation of easy-to-use malware creation tools and more complex and dangerous samples. Modern warfare and international cybercrime has also increasingly involved cyberattacks, targeted distribution attacks and highly networked cyberespionage, resulting in extremely dangerous malware that was explicitly designed and released by nation state actors for the purposes of enacting large-scale damage to organizations or infrastructure. Because of this growing and evolving threat, more advanced tools for malware detection, prevention and recovery are increasingly necessary to defend computing networks from attack.

Machine learning has been posited in the past as a potential solution to many of the problems that the field currently faces with traditional methodologies. The ability for algorithms to learn from existing samples and then apply that knowledge to novel samples is one that holds great potential for this field, especially since many high-profile malware attacks in the last decade were caused by previously unknown samples that exploited novel vulnerabilities and were not detected by traditional systems until it was too late.

This work presents a summary of the current body of work in this area of study, tests a few sample models from previous works of literature on modern malware samples from the last decade, and establishes a base for futher research in the field of malware analysis with machine learning techniques.

Chapter 1: Introduction

Problem Statement

The purpose of this research project is to investigate whether the integration of machine learning techniques and pattern recognition algorithms into traditional malware detection frameworks and methodologies will result in a quantifiable improvement in detection performance. This will be a research project exploring the assertion that if machine learning and artificially intelligent pattern recognition techniques were to be more fully applied and integrated into the problem of completely novel ("zero-day") vulnerability and malware identification, specifically in the areas that currently rely on primarily behavior analysis and signature analysis alone, it will demonstrate a substantial improvement over existing identification methods and sample detection rate.

Thesis Statement

Traditional signature analysis techniques in the problem of malware detection has the disadvantage of only being able to detect malware that has a previously encountered signature, meaning that these detection techniques are limited in their ability to detect all the novel types of malware that are being released into the wild every day. Behavioral analysis is similarly limited by its knowledge of specific behaviors it has seen before, and thus can fail to detect malware that has a new attack vector or exploit type that has never previously been encountered or recorded. The integration of machine learning techniques into these two traditional spheres of malware detection has the potential to improve the accuracy of malware recognition beyond the current capabilities of signature analysis and behavioral analysis methodology, as it may allow for the discovery of new malware detection patterns and correlations that will result in the ability to detect malware even when its specific signature has never been encountered or recorded before.

Research Questions

Specific questions that will be addressed in the study are:

1. What is the existing body of research on this topic, and are there any gaps in knowledge that can be addressed?

2. Which specific machine learning and pattern recognition algorithms prove to be the most effective for this problem application?

3. What types of training data should be utilized, and how large of a data corpus is required for useful results to be obtained using this approach?

4. Can we sufficiently demonstrate with experimental methodology that this approach improves on existing methods of "zero-day" or novel malware identification, beyond what current signature analysis and behavior analysis techniques are capable of?

5. How feasible would it be to incorporate machine learning and AI-driven pattern recognition capability into existing antivirus and malware defense solutions, and what technical

challenges and user experience challenges would this present? Which companies and products have already done this, and how effective is it and where could it be improved?

Chapter 2: Literature Review

Overview

This literature review is intended to contribute the following:

- Present a summary of the current state of the field of machine learning-powered malware detection and prevention.
- Identify current trends in malware detection pattern recognition and new approaches to the detection of novel malware.
- Discuss the challenges, limitations and gaps in knowledge that are presented by the current state of machine learning integration into malware detection.
- Evaluate potential avenues for future research and study that have not yet been explored in the known literature on this subject.

Synthesis

Machine learning is a constantly growing, rapidly evolving subfield of artificial intelligence that involves the use of algorithms and statistical models to enable a system to improve its performance on a specific, narrow task over a period of time and through many cycles of trial and error. The training process for a machine learning system is typically done by feeding the system a large amount of training data and allowing it to define correlations, identify meaningful patterns and make useful predictions or mimic a specific desired behavior based on those patterns, after which these patterns and predictions are reinforced and guided by a human overseer to encourage or discourage specific interpretations and improve performance on edge cases. Machine learning has a wide range of potential and practical applications in the field of computer science, and has already been applied to a wide array of tasks such as image generation and recognition, sentiment analysis, conversational text generation, facial recognition, natural language processing, and predictive modeling.

There is a great deal of existing literature on the integration of machine learning in malware detection, most of it from the past ten years as the field of machine learning overall has rapidly developed and flourished. Most of this literature acknowledges the assertion that signature-based analysis is limited by its database of known signatures, and will not detect novel or unknown samples of malware. The literature also recognizes that these databases must be constantly updated in order to remain useful in a rapidly changing and evolving malware development landscape, and as such they require a great deal of manual maintenance and upkeep by cybersecurity experts who will be constantly obligated to survey malware landscapes and keep up with “industry” trends. Finally, the literature notes that signature-based analysis will not identify variants or zero-day exploits, as both of these will not be available in the signature database ahead of time for the system to be aware of their presence. Machine learning methods have the potential to revolutionize the field of malware detection with the ability to detect patterns rather than individual signatures, expanding the capability of these systems with regard

to detecting novel samples and lessening the workload required to maintain these systems once they reach a certain level of complexity. However, almost every paper studied in the literature review noted limitations, challenges and daunting knowledge gaps in the idea of fully integrating machine learning into the problem of malware detection to produce a more effective and robust detection system, as described further below.

Limitations and Challenges

Most of the literature identifies several key limitations of machine learning applications in malware detection. These limitations are largely well-known open problems in artificial intelligence as a whole, and their resolution requires further research and development.

Accuracy

The accuracy and reliability of successful malware detection is one of the largest challenges facing this new breed of malware detection algorithms. Studies in this area vary widely on the success rates of the algorithms and methodologies they present, ranging from near-perfect to spotty at best, and many of these initially impressive models greatly decrease in accuracy when presented with new datasets or samples that fall outside the narrow band of samples they were trained on. A signature-based analysis system will detect a known signature one hundred percent of the time, while its machine learning counterpart may fail to detect some percentage of samples simply because they do not match the samples it has in its training corpus or fall outside its known behavioral criteria. As such, a machine learning system with a high enough accuracy level to be reliably useful, consistent in its detections, and trustworthy to the everyday user is an ongoing area of study and experimentation. Even the most advanced machine learning models may still struggle to match the accuracy of traditional models.

Many of the studies examined in this literature review did achieve notably high accuracy results on their corpus of samples, which seems to suggest that steps have been taken towards more reliable systems already, albeit on a narrow and curated subset of all the malware samples that could possibly exist. J. Saxe and K. Berlin (2015)[16] achieved a 95% accuracy rate on their corpus of samples, and Zhu et. al (2017)[10] accomplished a rate of 95.05%. However, other models that were presented struggled noticeably to achieve accurate results with consistency, and often had a substantial error rate that hindered their potential accuracy. Huang and Stokes (2016)[37] reported an error rate of 2.94% in one of their malware family classification models, which does not initially seem to be significant but rapidly becomes substantial in a scenario involving scanning thousands of files.

Performance

One well-known and often-documented limitation of modern machine learning is performance. There are many factors that can affect the performance of a machine learning model, including the quantity of training data, the type of algorithm used, and the amount of computational resources available to the machine learning system. Machine learning is an extremely processor-

intensive and demanding process for most computers, and as such the applications of advanced machine learning algorithms may not be feasible for home and personal computers with limited computing power or tolerance for latency. The literature also notes that machine learning training databases are often very large and will take up a cumbersome amount of storage space on most machines, limiting their practicality on older machines or machines with less disk space, and the quick and efficient retrieval of this data for detection purposes may not be possible anymore as the database grows larger and larger and searches become more computationally difficult. As noted by several papers in the review, additional layers of complexity and protection will result in additional performance problems, and trying to add further rules and edge cases to account for more and more exotic strains of malware only complicates this further.[36]

Tayyab et. al (2022)[36] specifically pointed out the issue of performance in machine learning malware detection in their survey of machine learning trends, noting that additional layers of data engineering to cater to the increasing volume of data introduces further delays and that well-performing small systems must focus on narrow, useful subsets of their overall training data to be successful and performant for the average user.[36]

Corpus Construction

The requirement for a very large, correctly labeled and well-organized corpus of curated training data to prepare a machine learning algorithm is one of the most difficult requirements to fulfill in the field of modern malware detection. The manual labor and sheer time investment required to tag every sample in the training set accurately and correctly alone, in a format that each individual machine learning system can understand and ingest, is enough to prevent most corpuses from growing larger than a few thousand samples in the literature that was studied. Modern malware samples also do not readily fall into sortable categories or give up their attributes and behaviors without careful, individual study and manual decompilation and labeling. This is especially true of the more dangerous and complex varieties of modern malware, such as polymorphic malware that can modify its own code or change itself to become more obfuscated and difficult for researchers to dissect. This means that the problem of preparing a complex, extensive and deep corpus of dissected malware samples for a malware detection algorithm is one that has thus far proved highly challenging in the literature. The obstacle of obtaining a sufficiently extensive and useful variety of live samples is also one that presents problems due to the technical, ethical and legal barriers that hosting and distributing malware naturally presents.

One of the papers studied that attempts to refute the limitation of a curated, labeled corpus in an interesting way is Y. Ye et. al (2017)[40], which attempted to train a system on a corpus that was partially or completely unlabeled. Their goal in doing so was to determine if a detection system could be capable of unsupervised “bottom to top” learning without human input or an explicitly labeled corpus.[40]

Real-Time Detection and Responsiveness

Another limitation of this approach noted in the literature is that real-time detection is difficult to reconcile with current machine learning algorithm execution speeds. The real-time detection and prevention of malware is one of the most critical components of a modern antivirus, and many modern antivirus programs are able to block malware programs from executing in real time before they even cause damage and promptly alert the user of the malicious program's presence. This capability is incredibly important for the function of a useful antivirus, and classical machine learning algorithms often cannot produce these split-second results fast enough to prevent malware from doing damage once it is detected. By the time the artificial intelligence is aware of it, the malware may already have begun to do damage or execute its malicious payload, which is an undesirable outcome for the end user. Home versions of antivirus software are often expected to be responsive to the user and relatively fast to react to their actions, and machine learning is traditionally not a reactive, user-friendly or responsive design space. Various strategies have been attempted in the literature to reconcile this problem, such as splitting malware machine learning algorithms into speed categories and attempting faster detection approaches that don't rely on traditional file-based detection.

A few of the papers studied in the literature attempt to explore and refute this limitation, such as J. Saxe and K. Berlin (2015)[16], whose conclusions contend that a small, accurate and effective machine learning system can indeed run in real time on a real environment. Other papers, however, affirm the limitation, such as Huang and Stokes (2016)[37].

Obfuscation and Encryption

Another challenge presented for future study in many of the papers surveyed is that malware authors often use techniques like obfuscation, encryption and hiding malicious code inside of innocent code to evade detection algorithms. Several algorithms have been developeed in attempts to break or circumvent popular obfuscation techniques, or to detect when malicious obfuscation has been applied. Common obfuscation methods have also been identified for use in pattern recognition writing, such as register reassignment, junk code or "dead code" insertion, string hiding, instruction substitution, and code transposition. Furthermore, the literature notes that even if code itself can be obfuscated, libraries, dependencies and imports often cannot be obfuscated because they are required for the code to function at all. Thus, this detail can be incorporated into malicious pattern detection if suspicious or known malicious libraries are imported as part of the code's function.

Methodologies

Many methodologies for machine learning-powered malware detection have been developed, tested and refined in current literature that attempt to address these challenges and limitations. Some success has already been found in the area of applying deep learning techniques to this problem, especially with hybrid approaches that pooled a combination of malware sample features rather than just training on individual features alone.

Hybrid Feature Methods

J. Saxe and K. Berlin (2015)[16] used a hybrid machine learning approach that collectively incorporated contextual byte features, import patterns, and a calibration-based scoring model. They were able to achieve a 95% accuracy rate with their deep learning system using this methodology, with a 0.1% false positive rate (FPR) based on over 400,000 malware samples and malicious binaries.[16] They concluded that their results were a promising indicator that it was now possible and feasible for everyday customers to run a small, robust and accurate machine learning model in real-world malware detection scenarios, with a false positive rate that approaches the false positive rate of most traditional antivirus programs. This style of program would also have the additional benefit of potentially being able to detect unknown or "zero-day" malware, which is a sentiment shared by many other papers in this review and a commonly touted benefit of machine learning incorporation.

Zhu et. al (2017)[10] achieved a similarly high detection score of 95.05%, outperforming many other traditional machine learning algorithms, using a unique "DeepFlow" deep learning technique and a hybrid mix of malware features. They focused specifically on Android application malware and malicious apps, since due to the open nature of the Google Play Store there is a comparatively high prevalence of malicious third-party applications being installed on Android devices when compared to the more regulated and tightly controlled Apple App Store and other app storefronts. The authors tested the system on both Android-native "benignware" (i.e. benign programs that may look like malware but aren't) and real malware in order to train their system to identify the difference between the two.[10]

Ye et. al (2018)[40] built on top of this research with "greedy" layer-based training in order to improve the performance of these novel machine learning techniques, using what they defined as a heterogeneous deep learning framework.[40] They focused on Windows application programming interface calls as one of their main features, as well as associative memory detection, and ran the system in an unsupervised learning mode followed by a supervised fine-tuning mode. Unlike previous work, they did not explicitly label the training data as either "malicious" or "benign," but rather used unlabeled file samples as part of the training process to see if the system could learn from the bottom up without necessarily needing to have fully labeled data.[40] This is an interesting rebuttal to the corpus construction limitation described earlier in this review, as the idea that a machine learning system could learn from almost entirely unlabeled training data removes much of the manual labor involved in tagging every sample in a very large data corpus.

Limitation Exploration

Other authors set out to explore the limitations of the machine learning detection approach.

Huang and Stokes (2016)[37] showcased the inefficiency of deep learning methods by using a raw-byte approach in their training data, indicating that there is a hard ceiling to the ability of

machine learning to have real-time performance on deeper levels of analysis and that sacrifices may have to be made.[37] Their system was trained on 4.5 million files and tested on a subset of 2 million files, which they claimed was the largest set of training data for a malware detection system to date. They achieved a binary classification error rate of 0.358%, which is more substantial than the 0.1% false positive rate achieved earlier by J. Saxe and K. Berlin (2015)[16]. They also made the bold claim that "no one has been able to demonstrate any gains for deep learning applied to malware classification," despite the previous papers in the field indicating that there was potential in the idea and demonstrating smalll proof-of-concept systems with promising results.

Cloud-Based Methods

Still other authors sought to apply the methods to cloud-based detection.

Martignoni et. al (2009)[19] was one of the first to use cloud-based malware detection methods by incorporating API traces and system calls into the detection features, resulting in the ability to detect execution traces in the cloud.[19] Their framework allowed the end user to create containers or small virtual machines where the malware samples would execute as if they were on a real machine, allowing users to analyze the samples on a fine-grained level with a low computational overhead cost.

Baseline Establishment and Anomaly Detection Methods

Yet another methodology described in the literature is the establishing of a baseline level for a machine, with malicious activity being defined as anything that falls outside of that baseline. One method that is explored is the real-time monitoring of system calls, using deep learning to determine which ones fall outside the usual use pattern of that machine. Another is the monitoring of file changes, so that a large amount of files changing outside the normal activity of the machine is detected as potentially malicious activity. Still another is that of comparing registry and system snapshots to detect pattern irregularities; the monitoring of network activities for network events that fall outside the normal user pattern; and real-time process monitoring and malicious process identification by unusual resource consumption or file location.

Procedural Signature Generation

There have also been proposed methods for procedurally generating malware signatures based on slight variations of an existing sample, and using known malicious strings and chunks rather than the entire malicious program as a whole in the machine learning training corpus.

Santos et al. (2009)[15] first proposed the idea of creating a corpus of malware *n*-grams – that is, the smallest possible amount of code that can be definitively classified as malicious – and applying them to the study of unknown programs in order to successfully detect novel malware that reincorporates existing malicious lines of code from other samples or the Internet.[15] This methodology had already been proven and demonstrated in other problem domains to be helpful

for machine learning systems. They used a common mathematical formula known as the nearest-neighbor algorithm to determine how similar an unknown, potentially novel malware n-gram was to known malicious malware n-grams, with high neighbor scores resulting in a higher detection score. The next logical step would then be to procedurally generate code samples that were neighbor-scored as close to other known samples, such that they were also malicious but were unknown to current signature databases. A large enough corpus of this type could potentially be able to recognize malicious n-grams inside of novel malware samples and flag the samples, given that there are only so many ways to code a given malicious function that are highly dissimilar.[15] The results of this experiment produced a database of 2,000 file signatures made up of malicious n-grams, with a detection rate of 69.66% for 2-grams and a maximum detection rate of 91.25% for 4-grams.[15]

Past Survey Papers

Many survey papers have been written about machine learning and statistical approaches to malware detection. The surveys that were located and synthesized in this literature review are noted below and described in detail. Several of them also proved to be intellectually valuable for my research project, as discussed further below.

Y. Meng, H. Zhuang, Z. Lin and Y. Jia (2021)[39]

This paper served as a comprehensive literature review of the field overall and summarized some of the same issues and challenges that I described above in the synthesis. It also discussed a few potential research directions that the field had yet to explore. It mentioned how the proliferation of polymorphic and metamorphic malware had resulted in significantly more challenges in malware detection in the past decade, as well as how research in this area had advanced to a stage where a more thorough review of the literature was needed to address these new challenges. The authors generally assert that the identification of malware through malicious features and behaviors, using machine learning as an asset to this process, will inevitably eclipse signature-based detection methods. The survey was presented to the 2021 International Conference on Computer Information Science and Artificial Intelligence in Kunming, China.

Tayyab, U.-e.-H.; Khan, F.B.; Durad, M.H.; Khan, A.; Lee, Y.S. (2022)[36]

This paper was a survey of recent deep learning trends in machine-based malware detection specifically, and focused on performance limitations, conventional and modern machine learning technique comparisons, and statistical analysis of the methods commonly used in the literature. They also discussed more thoroughly the latency problems and network limitations that machine learning introduces into malware detection systems, and suggest that large systems may need to be broken up into smaller modules and subsystems to address these challenges, with smaller subsets of the training data appropriate for each module's function. This is an interesting approach that I feel deserves more attention and discussion – perhaps my research could explore the idea of training a system specifically on ransomware, or specifically on remote access

Trojans, and so on. The survey was presented this year at the Pakistan Institute of Engineering and Applied Sciences in Nilore, Pakistan.

N. Pachhala, S. Jothilakshmi and B. P. Battula (2021).[25]

This paper was a study of malware classification techniques with machine learning specifically and had a narrow focus on how the field of cybersecurity classifies various types of malware, and how those types translate to machine learning labels. It discussed how many modern antiviruses assign malware into families, such as the WannaCry family of ransomware or the Cryxos family of Trojans, and how these labels can be both useful and too generalized to be helpful. It was an interesting discussion mainly because what initially seems like a very easy problem – tell what kind of malware you have – is actually far more complicated than it seems, especially because malware authors have an incentive to obfuscate that information or combine various types of malware together to evade detection and increase the sample's reach. The problem then becomes how to translate these labels into useful machine learning classifications and how to teach the system to assign novel samples into these boxes. The survey explored various works in this area and cited related work such as the Microsoft malware classification challenge, data mining advances, feature extraction and fusion, and executable grouping techniques to associate like-minded samples together.

S. Poudyal, Z. Akhtar, D. Dasgupta and K. D. Gupta (2019).[31]

This paper was a survey and examination of state-of-the-art machine learning malware detection approaches and what was on "the cutting edge" at the time the paper was written. It included discussion on deep eigenspace learning techniques, deep learning malicious code variant detection, sequence alignment, sequential pattern mining and heterogenous networks that use a variety of features to make their classification decisions. It cited many of the same papers that the other surveys cited, suggesting a general homogeneity of the literature in this area that has become more apparent the further I dive into this review.

Aslan, O.; Samet, R (2020).[7]

This paper was an extensive survey on the current state of malware detection approaches as a whole, not specifically focusing on machine learning but discussing it as part of the overall scope of the paper. It pointed out many of the same problems I mentioned in the synthesis – the known problems with signature-based and heuristic-based detection, the advent of behavior-based and cloud-based approaches for unknown and novel malware, deep learning as a potentially valuable new approach for zero-day and novel sample detection, and "in the wild" live detection of malware as it is produced and distributed by malware authors. The paper also discussed how there is a major gap in the current literature that must be filled by new studies, new methodologies and fresh ideas.

Chapter 3: Research Methods

Overview

For the experimental component of this survey project, I will be examining real-world sample data and testing real machine learning models on various curated research datasets that were uncovered during the literature review; these datasets and their origins, contents and sources are described in more detail further below. This experimental process will entail the examination, incorporation and testing of machine learning models that other authors have created in past surveys of this kind, in order to determine whether older machine learning models can still perform reasonably well on modern malware samples.

Experimental Design

Many of the previous experimental papers in this field resulted in the creation of testable machine learning models that performed well on specific subsets of malware data, such as Zhu et. al (2017)[10] and their "DeepFlow" machine learning model. The main methodology of this portion of the project will be gaining access to these models, wherever possible, and testing them on various types of data beyond the scope in which the original authors tested them, as well as adjusting them for modern and exotic types of malware to examine the impact on their performance and accuracy in order to challenge my hypothesis that machine learning improves the outcomes of these detection runs. These models will serve as real-world demonstrations of my experimental thesis that the field of malware detection will be served by the further study and incorporation of these and other machine learning systems, as well as addressing their limitations and challenges. The experimental portion of this survey project will thus encompass the areas of model acquisition, dataset formatting and ingestion, model testing, and analysis and statistical study of the results to determine whether they support or challenge my hypothesis.

As part of this process, I will be performing some degree of model comparison analysis, especially in situations where multiple models are capable of ingesting the same dataset due to a similarity in formatting or model design. Model comparison analysis is an experimental technique in machine learning and statistical modeling where several different models are trained and evaluated on the same data in order to determine which model performs the most optimally on that dataset. This project will evaluate these models according to accuracy, precision, recall, false positive rate (FPR), and F1 score, as many other experiments in the literature review did the same when testing various models against each other.

F1 Score

The F1 score of a machine learning model in mathematical terms is the harmonic mean of that model's precision and recall. *Precision* is defined as the number of correct positive classifications made by the model, divided by the total number of classification attempts made by the model during the test. *Recall* is the number of correct classifications made by the model divided by the total number of actual positives in the data. The F1 score is a shorthand that allows us to balance precision and recall such that the overall classification success of a model can be easily gleaned

from a single metric. A high F1 score indicates that a model has good precision and recall, while a lower F1 score typically indicates that a model is imperfect in one or both of these areas.

False Positive Rate (FPR)

A fundamental part of evaluating a malware detection model's performance is determining whether it does not incorrectly classify benevolent programs as malicious. This is usually referred to as the model's "false positive rate" or "false positive score" in the literature. Modern antivirus programs are usually described in the literature as having a 0.1% false positive rate or below, and several studies in the literature review achieved a similar or lower false positive rate, as described further in that section. This will therefore be the baseline false positive rate I will strive to achieve in my own experimental research.

Datasets

The study I am undertaking requires high-quality datasets and corpuses of malware data, which I sought out and organized as part of the literature review. The datasets that I found from the surveys in the literature review discussed above are listed below. As was pointed out several times in the literature, mainly by Aslan et. al (2020)[7], there are not many datasets that are curated and widely used for malware detection training in a research context, indicating a significant research gap in this area.[7] In addition, most of the existing datasets are not easily accessible for research and testing due to the risk of enabling malicious actors to gain access to large troves of malware, and in most cases the datasets accessed are not in the appropriate formats for data mining and ingestion. These samples must be formatted and organized in order to be properly processed and ingested by machine learning systems, which can be very time-consuming.

- **NSL-KDD dataset (2009),** consisting of approximately 125,000 records and 41 features.[7]
- **Drebin dataset (2014),** consisting of 5,560 malware samples across 20 families and 123,453 benign samples.[7]
- **Microsoft malware classification challenge dataset (2015**), published by Microsoft and consisting of 20,000 malware samples. [7]
- **ClaMP (classification of Malware with PE headers) dataset (2016),** consisting of 5,184 samples with 55 features. [7]
- **Contagio dataset,** consisting of PDF files that are both benign and malicious.
- **AAGM dataset (2017),** consisting of 400 Android malware samples and 1,500 benign samples from 12 families. [7]
- **EMBER dataset. (2018),** consisting of 1 million records of both malware samples and benign samples. [7]

Other datasets may be used as part of this research that were not initially uncovered during the literature review. These datasets will be examined, verified and notated appropriately as they are encountered and used throughout the research process. These include:

- **VirusShare database,** consisting of several thousand modern, curated malware samples that have been verified as malicious samples by VirusTotal.
- **TheZoo database,** consisting of several thousand modern and well-known samples curated on GitHub.

Chapter 4: Results

Dataset Acquisition

The first step in my experimental process was to attempt to acquire access to the datasets described above and decide which ones would be the most directly useful for my research purposes, by examining each database collated from the literature review and determining the merits and drawbacks of incorporating each one into my research. A major hurdle that I encountered was the fact that several of these databases are over a decade old, meaning that all of their malware samples are only functional on older machines and operating systems that are no longer supported or considered secure. Many databases that I examined were also no longer actively supported or were locked behind verification, limited access, dead links, web archives, or other barriers to access. Furthermore, I encountered other databases during this stage of my research that I had not previously discovered as part of the literature review.

The VirusShare Database

In the process of this stage of my research investigation, I uncovered a malware database that I found extremely useful and had not discovered in my preliminary research. The VirusShare database of malware and hashes proved to be a highly valuable collection of actively traded, verifiably malicious, modern malware samples collated and curated by VirusTotal analyses, and it has been used in a great deal of recent publications and research, including Abbasi et. al. (2020)[41]. The database of hash sets is publicly available, while individual samples required verification to download for research purposes. I acquired access to these large hash set files through a refined version provided by MantaRay Forensics, and curated and prepared these files such that they would be easy to ingest by my machine learning models, which proved extremely valuable for testing.

The Zoo

Another malware source that was uncovered after the literature review was The Zoo, a live malware repository hosted on Github that allows the study of live malware samples. This valuable trove of malware information included live samples of infamous and well-known ransomware such as WannaCry, Jigsaw, CryptoLocker, Zeus, and TeslaCrypt, which I tested by successfully infecting a virtual machine environment running Windows 8 and another virtual machine running Windows 10.

Model Acquisition

After acquiring access to the necessary malware and file hash datasets and formatting them for model ingestion, which included devising a few custom Python scripts for feature extraction, I next sought to obtain access to some of the machine learning-based malware classification models that were uncovered as part of the literature review and further investigations. The models I was able to obtain access to for testing and research use are described below.

- *Group 1:* The **LightGBM, EmberNN, Random Forest Model** and **Linear SVM Classifier** models featured in the paper “Explanation-Guided Backdoor Poisoning

Attacks Against Malware Classifiers" by Severi, Meyer, Coull & Oprea[42]. The goal of the authors was to demonstrate the weaknesses of a variety of machine learning models against a specific type of backdoor poisoning attack where a machine learning dataset is corrupted by malicious attackers, but it featured a variety of malware classifier models as part of their experiment. The models are conveniently hosted on Github (github.com/ClonedOne/MalwareBackdoors).

- *Group 2:* The **Hardened DNN Model** featured in the paper "Adversarial Deep Ensemble: Evasion Attacks and Defenses for Malware Detection" by D. Li & Q. Li[43]. The code is available on Github (github.com/deqangss/adv-dnn-ens-malware).
- *Group 3:* The **Convolution Neural Network (CNN) Model** (Kyadige and Rudd et al., https://arxiv.org/abs/1905.06987)**, "Ember" Gradient Boosted Decision Tree (GBDT) Model** (Anderson and Roth, https://arxiv.org/abs/1804.04637), and **"MalConv" Byte-Level CNN** (Raff et al., https://arxiv.org/abs/1710.09435), featured in the paper "Quo Vadis: Hybrid Machine Learning Meta-Model Based on Contextual and Behavioral Malware Representations" by Trizna, Dmitrijs (2022)[44]. The model code is provided on Github (github.com/dtrizna/quo.vadis).
- *Group 4:* The **DNN Model** featured in the paper "Adversarial Deep Learning for Robust Detection of Binary Encoded Malware" by A. Al-Dujaili et al. (2018)[45]. The model code is provided on Github (github.com/ALFA-group/robust-adv-malware-detection).

Of these, Group 1's model set proved to be the most useful for testing, the most programmatically diverse in terms of model variety, and the most readily able to adapt for other, more customized datasets and specific sample files beyond the ones on which it had been tested in the paper. In addition, Group 2's model relied on a deprecated version of TensorFlow that is no longer readily available or compatible with modern packages, and so was not able to be used for testing on the machines and environments I had readily available. Thus, the Group 1 model set was the one that I primarily used for the final testing runs.

Model Testing

At this stage I was finally prepared to test the chosen subset of machine learning models on the datasets that I had acquired and curated. This testing was performed in a normal Windows 10 environment as well as within a Windows 10 virtual machine environment created with VirtualBox, and primarily utilized Python 3.6-3.8, Visual C++ 15, TensorFlow 1.0 (for backwards compatibility functions) and 2.0, SKLearn, and a variety of Python libraries and packages, including Numpy, Joblib and PEFile.

Model Performance Results

The model performance results that were acquired from the testing runs performed above are quantified and summarized below.

LightGBM

This model produced the following results on the databases on which it was tested:

EMBER Subset (450,000 Samples, 144,487 Positive / 305,513 Negative)

	Precision	Recall	F1
0.0	0.83483	0.99833	0.90929
1.0	0.99746	0.76818	0.86793
Accuracy			**0.89245**
Macro Average	0.91615	0.88326	0.88861
Weighted Average	0.90965	0.89245	0.89026

Contagio PDF Dataset (6,000 Samples)

	Precision	Recall	F1
0.0	0.99750	0.99900	0.99825
1.0	0.99900	0.99750	0.99825
Accuracy			**0.99825**
Macro Average	0.99825	0.99825	0.99825
Weighted Average	0.99825	0.99825	0.99825

Specific Samples (When Trained On EMBER)

	Sha256 Hash	Identified?
Jigsaw Ransomware	3ae96f73d805e1d3995253db4d910300d8442ea603737a1428b613061e7f61e7	Yes

EmberNN

This model produced the following results on the databases on which it was tested:

EMBER Subset (200,000 Samples)

	Precision	Recall	F1
0.0	0.53995	1.00000	0.70126
Accuracy			**0.53995**
Macro Average	0.26998	0.50000	0.35063
Weighted Average	0.29155	0.53995	0.37864

Contagio PDF Dataset (6,000 Samples)

	Precision	Recall	F1
0.0	0.99795	0.97451	0.98609
1.0	0.97510	0.99800	0.98642
Accuracy			**0.98626**
Macro Average	0.98653	0.98626	0.98625
Weighted Average	0.98653	0.98626	0.98625

Random Forest Model

This model produced the following results on the databases on which it was tested:

EMBER Subset (200,000 Samples)

	Precision	Recall	F1
0.0	0.81073	0.99759	0.89451
1.0	0.99613	0.72666	0.84032
Accuracy			**0.87295**
Macro Average	0.90343	0.86213	0.86741
Weighted Average	0.89602	0.87295	0.86958

Contagio PDF Dataset (6,000 Samples)

	Precision	Recall	F1
0.0	0.99800	0.99750	0.99775
1.0	0.99750	0.99800	0.99775
Accuracy			**0.99775**
Macro Average	0.99775	0.99775	0.99775
Weighted Average	0.99775	0.99775	0.99775

Linear SVM Classifier

This model produced the following results on the databases on which it was tested:

EMBER Subset (200,000 Samples)

	Precision	Recall	F1
0.0	0.54657	0.99500	0.70556
1.0	0.84164	0.03119	0.06016
Accuracy			**0.55160**
Macro Average	0.69411	0.51310	0.38286
Weighted Average	0.68232	0.55160	0.40864

Contagio PDF Dataset (6,000 Samples)

	Precision	Recall	F1
0.0	0.95267	0.78461	0.86051
1.0	0.81691	0.96102	0.88312
Accuracy			**0.87281**
Macro Average	0.88479	0.87281	0.87182
Weighted Average	0.88479	0.87281	0.87182

Chapter 5: Conclusions and Recommendations

The current state of machine learning research in the field of malware detection still has a long way to go before it is fully commercially viable and can be relied on to perform consistently, both for personal security and for large organizations. The models tested in this work, while very strong on certain datasets and particular subsets of known malware samples, still did not compare to commercial antivirus programs in several respects, especially when attempting to achieve the 99% positive sample detection rate that is claimed by many commercial antivirus programs. It is also worth noting that while a detection rate in the 80-90% range seems very positive at first glance, it will still, on average, miss one or two out of every ten malicious files – which paints that seemingly high accuracy in a very different light when you think about how much malware is out there.

On the flip side, however, it is clear that with strong training data, especially the subsets of several hundred thousand samples that were used in the EMBER dataset testing done here and in the works of other authors, many models can get close enough to that threshold to be very promising. The high number of collated features in the EMBER dataset in particular provides a staggering amount of valuable correlations when it comes to training a modern machine learning model to recognize the typical properties of malicious and benign samples. As such, while there is still much room to grow, the future of this field still looks to be heading in a positive direction, and many authors are doing tremendous work to get us closer every day. Even some of the older machine learning models utilized in this work still performed very well on modern samples, indicating that the idea of applying artificial intelligence in this area is as strong on the conceptual level now as it was several years ago. It is the hope of the author that this field continues to grow and flourish in the future.

Works Cited

[1] A. Azmoodeh, A. Dehghantanha, M. Conti and K.-K.-R. Choo (2018). "Detecting crypto-ransomware in IoT networks based on energy consumption footprint", J. Ambient Intell. Hum. Comput., vol. 9, no. 4, pp. 1141-1152, Aug. 2018. Retrieved April 2023 from https://www.researchgate.net/publication/319252402_Detecting_crypto-ransomware_in_IoT_networks_based_on_energy_consumption_footprint

[2] A. bin Asad, R. Mansur, S. Zawad, N. Evan and M. I. Hossain (2020). "Analysis of Malware Prediction Based on Infection Rate Using Machine Learning Techniques," 2020 IEEE Region 10 Symposium (TENSYMP), 2020, pp. 706-709, doi: 10.1109/TENSYMP50017.2020.9230624. Retrieved November 2022 from https://ieeexplore.ieee.org/document/9230624

[3] A. Karnik, S. Goswami and R. Guha (2007). "Detecting obfuscated viruses using cosine similarity analysis", Proc. 1st Asia Int. Conf. Modeling Simulation (AMS), Mar. 2007. Retrieved April 2023 from https://ieeexplore.ieee.org/abstract/document/4148653

[4] A. Narayanan, M. Chandramohan, L. Chen and Y. Liu (2017). "Context–aware adaptive and scalable Android malware detection through online learning", IEEE Trans. Emerg. Topics Comput., vol. 1, no. 3, pp. 157-175, Jun. 2017. Retrieved April 2023 from https://ieeexplore.ieee.org/document/7935482

[5] A. Shabtai, U. Kanonov, Y. Elovici, C. Glezer and Y. Weiss (2012). "Andromaly: A behavioral malware detection framework for Android devices", J. Intell. Inf. Syst., vol. 38, no. 1, pp. 161-190, Feb. 2012. Retrieved April 2023 from https://link.springer.com/article/10.1007/s10844-010-0148-x

[6] A. Walker and S. Sengupta (2019). "Insights into Malware Detection via Behavioral Frequency Analysis Using Machine Learning," MILCOM 2019 - 2019 IEEE Military Communications Conference (MILCOM), 2019, pp. 1-6, doi: 10.1109/MILCOM47813.2019.9021034. Retrieved November 2022 from https://ieeexplore.ieee.org/document/9021034

[7] Aslan, O.; Samet, R (2020). “A Comprehensive Review on Malware Detection Approaches.” IEEE Access 2020, 8, 6249–6271. Retrieved December 2022 from https://ieeexplore.ieee.org/document/8949524

[8] B. Tahtaci and B. Canbay (2020). "Android Malware Detection Using Machine Learning," 2020 Innovations in Intelligent Systems and Applications Conference (ASYU), 2020, pp. 1-6, doi: 10.1109/ASYU50717.2020.9259834. Retrieved November 2022 from https://ieeexplore.ieee.org/document/9259834

[9] Christodorescu, M.; Jha, S (2003). “Static analysis of executables to detect malicious patterns.” In Proceedings of the 12th USENIX Security Symposium (USENIX Security

03), Washington, DC, USA, 4–8 August 2003. Retrieved April 2023 from https://research.cs.wisc.edu/wisa/papers/security03/cj03.pdf

[10] D. Zhu, H. Jin, Y. Yang, D. Wu and W. Chen (2017). "DeepFlow: Deep learning-based malware detection by mining Android application for abnormal usage of sensitive data", Proc. IEEE Symp. Comput. Commun. (ISCC), Jul. 2017. Retrieved December 2022 from https://ieeexplore.ieee.org/abstract/document/8024568

[11] Dilhara, Shashie. (2021). "Classification of Malware using Machine learning and Deep learning Techniques." International Journal of Computer Applications. 183. 12-17. 10.5120/ijca2021921708. Retrieved November 2022 from https://www.researchgate.net/publication/355350071_Classification_of_Malware_using_Machine_learning_and_Deep_learning_Techniques

[12] H. Sun, X. Wang, J. Su and P. Chen (2015). "RScam: Cloud-based anti-malware via reversible sketch", Proc. Int. Conf. Secur. Privacy Commun. Syst., 2015.

[13] H. -Y. Chuang and S. -D. Wang (2015). "Machine Learning Based Hybrid Behavior Models for Android Malware Analysis," 2015 IEEE International Conference on Software Quality, Reliability and Security, 2015, pp. 201-206, doi: 10.1109/QRS.2015.37. Retrieved November 2022 from https://ieeexplore.ieee.org/document/7272933

[14] I. Firdausi, C. lim, A. Erwin and A. S. Nugroho (2010). "Analysis of Machine learning Techniques Used in Behavior-Based Malware Detection," 2010 Second International Conference on Advances in Computing, Control, and Telecommunication Technologies, 2010, pp. 201-203, doi: 10.1109/ACT.2010.33. Retrieved November 2022 from https://ieeexplore.ieee.org/document/5675808

[15] I. Santos, Y. K. Penya, J. Devesa and P. G. Bringas (2009). "N-grams-based file signatures for malware detection", Proc. 11th Int. Conf. Enterprise Inf., vol. 9, pp. 317-320, 2009.

[16] J. Saxe and K. Berlin (2015). "Deep neural network based malware detection using two dimensional binary program features", Proc. 10th Int. Conf. Malicious Unwanted Softw. (MALWARE), Oct. 2015. Retrieved December 2022 from https://arxiv.org/abs/1508.03096

[17] K. P. Subedi, D. R. Budhathoki and D. Dasgupta (2018). "Forensic Analysis of Ransomware Families Using Static and Dynamic Analysis," 2018 IEEE Security and Privacy Workshops (SPW), 2018, pp. 180-185, doi: 10.1109/SPW.2018.00033. Retrieved November 2022 from https://ieeexplore.ieee.org/document/8424649

[18] L. Binxiang, Z. Gang and S. Ruoying (2019). "A Deep Reinforcement Learning Malware Detection Method Based on PE Feature Distribution," 2019 6th International Conference on Information Science and Control Engineering (ICISCE), 2019, pp. 23-27, doi:

10.1109/ICISCE48695.2019.00014. Retrieved November 2022 from https://ieeexplore.ieee.org/document/9107644

[19] L. Martignoni, R. Paleari and D. Bruschi (2009). "A framework for behavior-based malware analysis in the cloud", Proc. Int. Conf. Inf. Syst. Secur., 2009.

[20] L. Xiao, Y. Li, X. Huang and X. Du (2017). "Cloud–based malware detection game for mobile devices with offloading", IEEE Trans. Mobile Comput., vol. 16, no. 10, pp. 2742-2750, Oct. 2017.

[21] M. G. Schultz, E. Eskin, F. Zadok and S. J. Stolfo (2001). "Data mining methods for detection of new malicious executables", Proc. IEEE Symp. Secur. Privacy, May 2001.

[22] M. Yeo et al. (2018). "Flow-based malware detection using convolutional neural network," 2018 International Conference on Information Networking (ICOIN), 2018, pp. 910-913, doi: 10.1109/ICOIN.2018.8343255. Retrieved November 2022 from https://ieeexplore.ieee.org/document/8343255

[23] N. B. Akhuseyinoglu and K. Akhuseyinoglu (2016). "AntiWare: An automated Android malware detection tool based on machine learning approach and official market metadata," 2016 IEEE 7th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON), 2016, pp. 1-7, doi: 10.1109/UEMCON.2016.7777867. Retrieved November 2022 from https://ieeexplore.ieee.org/document/7777867

[24] N. Nissim, A. Cohen and Y. Elovici (2017). "ALDOCX: Detection of Unknown Malicious Microsoft Office Documents Using Designated Active Learning Methods Based on New Structural Feature Extraction Methodology," in IEEE Transactions on Information Forensics and Security, vol. 12, no. 3, pp. 631-646, March 2017, doi: 10.1109/TIFS.2016.2631905. Retrieved November 2022 from https://ieeexplore.ieee.org/document/7762928

[25] N. Pachhala, S. Jothilakshmi and B. P. Battula (2021). "A Comprehensive Survey on Identification of Malware Types and Malware Classification Using Machine Learning Techniques," 2021 2nd International Conference on Smart Electronics and Communication (ICOSEC), 2021, pp. 1207-1214, doi: 10.1109/ICOSEC51865.2021.9591763. Retrieved November 2022 from https://ieeexplore.ieee.org/document/9591763

[26] P. R. K. Varma, K. P. Raj and K. V. S. Raju (2017). "Android mobile security by detecting and classification of malware based on permissions using machine learning algorithms," 2017 International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC), 2017, pp. 294-299, doi: 10.1109/I-SMAC.2017.8058358. Retrieved November 2022 from https://ieeexplore.ieee.org/document/8058358

[27] S. Naz and D. K. Singh (2019). "Review of Machine Learning Methods for Windows Malware Detection," 2019 10th International Conference on Computing, Communication and Networking Technologies (ICCCNT), 2019, pp. 1-6, doi: 10.1109/ICCCNT45670.2019.8944796. Retrieved November 2022 from https://ieeexplore.ieee.org/document/8944796

[28] S. Poudyal, K. P. Subedi and D. Dasgupta (2018). "A Framework for Analyzing Ransomware using Machine Learning," 2018 IEEE Symposium Series on Computational Intelligence (SSCI), 2018, pp. 1692-1699, doi: 10.1109/SSCI.2018.8628743. Retrieved November 2022 from https://ieeexplore.ieee.org/document/8628743

[29] S. Poudyal and D. Dasgupta (2020). "AI-Powered Ransomware Detection Framework," 2020 IEEE Symposium Series on Computational Intelligence (SSCI), 2020, pp. 1154-1161, doi: 10.1109/SSCI47803.2020.9308387. Retrieved November 2022 from https://ieeexplore.ieee.org/document/9308387

[30] S. Poudyal and D. Dasgupta (2021). "Analysis of Crypto-Ransomware Using ML-Based Multi-Level Profiling," in IEEE Access, vol. 9, pp. 122532-122547, 2021, doi: 10.1109/ACCESS.2021.3109260. Retrieved November 2022 from https://ieeexplore.ieee.org/document/9526633

[31] S. Poudyal, Z. Akhtar, D. Dasgupta and K. D. Gupta (2019). "Malware Analytics: Review of Data Mining, Machine Learning and Big Data Perspectives," 2019 IEEE Symposium Series on Computational Intelligence (SSCI), 2019, pp. 649-656, doi: 10.1109/SSCI44817.2019.9002996. Retrieved November 2022 from https://ieeexplore.ieee.org/document/9002996

[32] S. Vanjire and M. Lakshmi (2021). "Behavior-Based Malware Detection System Approach For Mobile Security Using Machine Learning," 2021 International Conference on Artificial Intelligence and Machine Vision (AIMV), 2021, pp. 1-4, doi: 10.1109/AIMV53313.2021.9671009. Retrieved November 2022 from https://ieeexplore.ieee.org/document/9671009

[33] T. Isohara, K. Takemori and A. Kubota (2011). "Kernel-based behavior analysis for Android malware detection", Proc. 7th Int. Conf. Comput. Intell. Secur., Dec. 2011.

[34] T. Kim, B. Kang, M. Rho, S. Sezer and E. G. Im (2019). "A Multimodal Deep Learning Method for Android Malware Detection Using Various Features," in IEEE Transactions on Information Forensics and Security, vol. 14, no. 3, pp. 773-788, March 2019, doi: 10.1109/TIFS.2018.2866319. Retrieved November 2022 from https://ieeexplore.ieee.org/document/8443370

[35] T. T. Trang Nguyen, D. Tho Nguyen and D. L. Vu (2021). "A Hypercuboid-Based Machine Learning Algorithm for Malware Classification," 2021 RIVF International Conference on

Computing and Communication Technologies (RIVF), 2021, pp. 1-6, doi: 10.1109/RIVF51545.2021.9642093. Retrieved November 2022 from https://ieeexplore.ieee.org/document/9642093

[36] Tayyab, U.-e.-H.; Khan, F.B.; Durad, M.H.; Khan, A.; Lee, Y.S. (2022) “A Survey of the Recent Trends in Deep Learning Based Malware Detection.” J. Cybersecur. Priv. 2022, 2, 800–829. https://doi.org/10.3390/jcp2040041. Retrieved December 2022 from https://www.mdpi.com/2624-800X/2/4/41

[37] W. Huang and J. W. Stokes (2016). "MtNet: A multi-task neural network for dynamic malware classification", Proc. Int. Conf. Detection Intrusions Malware Vulnerability Assessment, 2016. Retrieved December 2022 from https://link.springer.com/chapter/10.1007/978-3-319-40667-1_20

[38] Westyarian, Y. Rosmansyah and B. Dabarsyah (2015). "Malware detection on Android smartphones using API class and machine learning," 2015 International Conference on Electrical Engineering and Informatics (ICEEI), 2015, pp. 294-297, doi: 10.1109/ICEEI.2015.7352513. Retrieved November 2022 from https://ieeexplore.ieee.org/document/7352513

[39] Y. Meng, H. Zhuang, Z. Lin and Y. Jia (2021). "A Survey on Machine Learning-based Detection and Classification Technology of Malware," 2021 International Conference on Computer Information Science and Artificial Intelligence (CISAI), 2021, pp. 783-792, doi: 10.1109/CISAI54367.2021.00158. Retrieved November 2022 from https://ieeexplore.ieee.org/document/9718826

[40] Y. Ye, L. Chen, S. Hou, W. Hardy and X. Li (2018). "DeepAM: A heterogeneous deep learning framework for intelligent malware detection", Knowl. Inf. Syst., vol. 54, no. 2, pp. 265-285, Feb. 2018. Retrieved April 2023 from https://link.springer.com/article/10.1007/s10115-017-1058-9

[41] Abbasi, Muhammad Shabbir, Al-Sahaf, Harith and Welch, Ian (2020). “Particle Swarm Optimization: A Wrapper-Based Feature Selection Method for Ransomware Detection and Classification.” International Conference on the Applications of Evolutionary Computation (Part of EvoStar), 181-196. Springer.

[42] Giorgio Severi, Jim Meyer, Scott Coull, and Alina Oprea (2020). “Explanation-Guided Backdoor Poisoning Attacks Against Malware Classifiers.” USENIX Security 2021.

[43] D. Li and Q. Li (2020). "Adversarial Deep Ensemble: Evasion Attacks and Defenses for Malware Detection," in IEEE Transactions on Information Forensics and Security, vol. 15, pp. 3886-3900, 2020, doi: 10.1109/TIFS.2020.3003571. Retrieved June 2023 from https://ieeexplore.ieee.org/document/9121297

[44] Trizna, Dmitrijs (2022). “Quo Vadis: Hybrid Machine Learning Meta-Model Based on Contextual and Behavioral Malware Representations.” Doi: 10.1145/3560830.3563726. Retrieved June 2023 from https://dl.acm.org/doi/10.1145/3560830.3563726

[45] A. Al-Dujaili et al (2018). “Adversarial Deep Learning for Robust Detection of Binary Encoded Malware.” Retrieved June 2023 from https://arxiv.org/pdf/1801.02950.pdf

Appendix A