CHAPTER 1
INTRODUCTION

1.1 Background

In today's society, technology can be found almost everywhere, from the simplest computer in form of a calculator to advanced computational systems implemented in cars and houses. Technology plays a monumental role in even the most mundane activities. A prime example of the integration of technology in everyday life is the habitual usage of mobile phones. According to a recent count by Ericsson and presented by Statista, on a worldwide scale, there is a total of 7.1 billion mobile phone users, and this number will continue to rise with the predicted count of 7.49 billion in the year 2025 [1]. If compared to the current world population which is 7.9 Billion [2], the number of mobile phone users worldwide holds its weight, therefore proving the commonality of mobile devices in everyday society.

Mobile devices need to run on an operating system and the top contenders of mobile operating systems include Android and Apple, with Android holding the top spot at 70.97% in the first quarter of 2022 [3]. Consequently, with such a large market share, the importance of keeping the devices that run on said operating system secured becomes a quintessential concern for all, as smartphones stores valuable information and assets such as pictures, personal documents, and access to bank accounts [4]. Due to the Android operating system being open-source, the potential for harm is enabled as users can download and install applications liberally from the Google App Store, or third-party sources [5]. This property of the operating system allows individuals to freely develop malicious applications and publish them on seemingly trustworthy platforms which eventually deceives the average user [6].

Over the years, Android as an operating system has attempted to reduce the amount of malware, or simply to complicate the process of malware being installed in an Android system. These include the in-house malware detection system implemented on the Google Play Store called Bouncer, the detection system implemented through the Google App Store and the Android Permission System [7]. Bouncer was proven to be unsatisfactory due to its lack of sensitivity in detecting malware, while the detection system by the Google App Store lacks protection as it detects malicious intent through analysis of the application metadata, this included user comments and user ratings. This methodology was proven to be ineffective as

the malware would have already affected some users before the system flags it for malicious intent [8]. An approach that has been the topic of discussion within recent years is the usage and development of the Android Permission System [7]. This mechanism is highly effective and essential as it regulates the access an application has to hardware features, mobile network, and data storage [9, 10].

With the constant evolution of malware, counter measuring approaches need to be made. This may come in the development of new complex detection systems or a new approach to existing detection systems. A popular approach that has been gaining traction is the implementation of machine learning models in detecting malicious software [11]. Multiple research has been made using various machine learning models that process a variety of features within the Android system. When detecting malware, there are two main approaches, static and dynamic analysis. Static analysis examines an application's code and determines its intent without executing it [12]. While a dynamic analysis monitors an application in an executed state. When compared, static analysis has a bigger advantage in terms of detecting malware as it serves as a preemptive attempt to halt a possible attack, dynamic approaches also require more resources as it needs to run the application in question [13].

This paper will take a static approach in the detection of Android malicious applications based on the reasons mentioned previously. Static approaches use the features that can be extracted from the application code, this includes hardware components, requested and used permissions, app components, filtered intents, API calls, and network addresses [14]. In 2014, a project called *DREBIN*[14] introduced a system that implements a Support Vector Machine (SVM) to perform a broad static analysis and extracted the previously mentioned features, which then are used to determine whether an application is malicious or benign. This project however used data that were collected over an older version of Android that was common from 2010 to 2012. Another approach was proposed by Wu *et al.* with their *DroidMat*[15] system. *DroidMat* implements and compared the k-Nearest-Neighbour(kNN) and Naive Bayes algorithm in classifying the intent of an application using the application's API calls and permissions. Similarly, Idrees *et al.* proposed another methodology that compared Naive Bayes, KStar, and Prism algorithms in detecting Android-based applications based on Android intents and permissions [16]. When it comes to static feature analysis to detect malicious applications, permissions play an important role as it allows the user to actively select what is allowed to happen to their devices preemptively, therefore allowing the choice

to protect access and privacy before anything is properly installed on the device itself. The importance of permissions and static analysis through permissions was also seen through the number of explorations that was done on the topic over the past few years. The application of machine learning to statically analyze the intent of an application is also said to be effective, this is seen through high accuracy and detection rates proven by past researches [17, 18, 19].

As mentioned, machine learning is an effective approach in detecting malicious applications as seen in previous research done. However, the methodology and model implemented in each system are essential in deciding which would be the most accurate and effective [20]. Based on the comparisons of the systems proposed, the models that resulted in the highest accuracy when using static analysis included the algorithms; Naive Bayes, Decision Trees, and Logistic Regression [21, 22, 23]. As these models work well on their own, the question that is posed consequently is how can it be improved. One approach is to implement Ensemble Learning Systems using these proven models [24]. Based on research done by Wang *et al.* when implementing Naive Bayes and Decision Trees in an ensemble, the accuracy improvement percentage is always positive [25]. The research that was done using the 3 algorithms (Naive Bayes, Decision Trees, and Logistic Regression) [26] also yielded a 97 to 99 percent detection rate on a dataset with 6800 data and 65 features, which further the implication that the implementation of an ensemble learning system would prove to be more effective than the single model traditional approach.

Based on previous research and the reasons previously mentioned, this paper will further explore the effects of implementing an ensemble learning system using the three models; Naive Bayes, Decision Trees, and Logistic Regression, to detect malicious applications within the Android system, based on the permissions the applications require.

1.2 Problem Statement

The research that is going to be conducted will explore the following problem statements:

1. How to implement Decision Trees, Naive Bayes, and Logistic Regression in an Ensemble Learning System to detect malware in Android Mobile Devices?

2. What combination of models within the proposed Ensemble Learning System

has the best Accuracy, Precision, Recall, F-1 Score and ROC Area values?

1.3 Scope and Limitations

The scope and limitations of the study are as follows:

1. The dataset to be used will be from the third version of the *Android Permission Dataset* by Arvind Mahindru[27]. This dataset has the different permissions available for android applications as its feature and therefore the scope of judgement to determine if an application has malicious intent will be strictly limited to its permissions.
2. The ensemble that will be built will focus on a parallel learning system in which the models are independent of one another and will run in side-by-side to produce the final decision.
3. The deployment of the proposed system will not scan an application file, though an application will be made with the deployed model to simulate the ability of the model to identify whether an android mobile application is malicious or not.

1.4 Research Objectives

The objectives for this research includes:

1. Implement Decision Trees, Naive Bayes, and Logistic Regression in an Ensemble Learning System to detect malware in Android Mobile Devices
2. Determine the combination of models within the proposed Ensemble Learning System that has the best Accuracy, Precision, Recall, F-1 Score and ROC Area values.

1.5 Research Benefits

The benefits of this study includes:

1. Aid in the development of a static mobile malware detection system that utilizes permissions to determine whether an application is malicious or benign, doing so with a machine learning based system through an ensemble learning method.

2. Spread knowledge on the dangers of malicious mobile applications and provide a demo platform that classifies if an application is malicious or not based on the permissions it requires before being downloaded.

1.6 Report Structure

The report will be written with a structure as such:

- CHAPTER 1 INTRODUCTION
  Chapter one discusses the report and study as a whole, it gives an insight into what will be discussed and what was explored throughout the study. This chapter consists of the background of the issue at hand, problem statement, scope and limitations, research objectives, and research benefits.

- CHAPTER 2 LITERATURE REVIEW
  Chapter two dives deeper into the theories that help support the study. Here, the terms, theories and formulas are expounded upon. The theories that were explained here comprises of the concept of Android Permissions, Ensemble Learning, Naive Bayes, Logistic Regression, Decision Trees and the Evaluation Metrics that were used to validate the models.

- CHAPTER 3 RESEARCH METHODOLOGY
  Chapter three is a chapter dedicated to the process of the study. In this chapter, the methodology of the study is explained. This includes the steps that were taken to conduct the study as a whole.

- CHAPTER 4 RESULTS AND DISCUSSION
  Chapter four put into words the results of the study, here all the outcome and the product of the study is described and explained. The chapter shows and explains the data that was gathered and an analysis of how those results came to be.

- CHAPTER 5 CONCLUSIONS AND FUTURE WORK
  This last chapter serves as the closing to the report in which suggestions and conclusions are made. The suggestions for future work that stems from this report will be fully written in this chapter.