Republic of Iraq
Ministry of Higher Education
and Scientific Research
University of Diyala
College of Science

Detection of Malware under Android Mobile Application

A Thesis
Submitted to the Department of Computer Science\ College of Science \ University of Diyala as a Partial Fulfillment of the Requirements for the Degree of Master in Computer Science

By

Saja Ibraheem Hani Ismail

Supervised By

Prof. Naji M. Sahib

2020 A.D. 1441 A.H.

بِسْمِ ٱللَّهِ ٱلرَّحْمَٰنِ ٱلرَّحِيمِ

يَا أَيُّهَا الَّذِينَ آمَنُوا إِذَا قِيلَ لَكُمْ تَفَسَّحُوا فِي الْمَجَالِسِ
فَافْسَحُوا يَفْسَحِ اللَّهُ لَكُمْ ۖ وَإِذَا قِيلَ انْشُزُوا فَانْشُزُوا يَرْفَعِ
اللَّهُ الَّذِينَ آمَنُوا مِنْكُمْ وَالَّذِينَ أُوتُوا الْعِلْمَ دَرَجَاتٍ وَاللَّهُ بِمَا
تَعْمَلُونَ خَبِيرٌ ﴿١١﴾

صَدَقَ اللهُ العَظِيم

سورة المجادلة آية ﴿١١﴾

Dedication

To those who taught me how to stand firmly on the ground

Dear father

To the source of love, altruism and generosity

My Mother

To the closest people to myself
My faithful husband

To my soul, my eyesight, and my heartbeat
My children

I present to you a summary of my scientific efforts

Saja Ibraheem

Acknowledgment

At the end of my thesis, I am pleased to thank and show my gratitude to those who have helped me to achieve my study whether through assisting to provide resources or with guidance and presenting me advice.

First, I would like to thank my honorable prof. Naji Muttar Sahib who supervised my thesis and offered me many scientific advices and guidelines; to him I give all of my appreciation.

In addition, I want to thank my respected professors in Department of Computer Science with whom I completed a very important stage of education within the academic journey through their scientific experiences and helping us to complete it.

Also, I would love to thank prof. Azhar Hasan Nussief who provided me with solid resources, so I give my gratitude to her.

In addition, I want to thank my family who supported me and stood by me during my study journey.

Saja Ibraheem

Supervisors' Certification

I certify that this thesis entitled **"Detection of Malware under Android Mobile Application"** was prepared by **"Saja Ibraheem Hani"** under my supervisions at the University of Diyala collage of Science Department of Computer Science, as a partial fulfillment of the requirements needed to award the degree of Master of Science in Computer Science.

(Supervisor)
Name: Prof. Naji M. Sahib

Signature: [illegible]
Date: / / 2020

Approved by University of Diyala\ College of Science\ Department of Computer Science.

Signature: [illegible]
Name: Asst. Prof. Dr. Taha Mohammed Hasan
Date:

Head of Computer Science Department

(Linguistic Certification)

I certify that this research entitled “**Detection of Malware under Android Mobile Application**” was prepared by “**Saja Ibraheem Hani**” and was reviewed linguistically. Its language was amended meet the style of English language.

Signature:
Name:
Date:

Scientific Amendment

I certify that the thesis entitled “**Detection of Malware under Android Mobile Application”** was prepared by **"Saja Ibraheem Hani"** has been evaluated scientifically; therefore, it is suitable for debate by examining committee.

Signature :
Name :
Date : **/ / 2020**

Abstract

Smartphones have become essential in our daily life. It also can do a lot of work and can browse the Internet, and download many applications for each device, through the available store. As a result, the number of malware applications downloaded also increases.

This malware carries out various activities behind the scenes; Such as confidentiality, breach of privacy, loss of confidentiality, system breakdown, theft of sensitive information, etc.

Many types of research and studies that proposed different techniques to detect malicious programs, but they contained weak points, which are illustrated by efficiency, speed, and lack of comprehensiveness.

In this thesis, a proposed system is developing implementing to detect malware in smartphones, and contains two parts:

In the first part, access control is initially started upon the system launch. The user authentication algorithm adopts the user's permission to detect a threat factor after applying a user's permission policy by improving the method with which the user's activities are extracted. While, **in the second part,** anomaly detection technology begins to extract the important features that play an effective role in detecting malicious code.

The proposed system has been tested by using a hybrid genetic algorithm, and the SVM data has been registered an accuracy of (0.9282).

The experimental results indicate that the proposed system has a high average accuracy rate compared to other existing methods where it (0.8848) average accuracy using PNN, while the average accuracy is (0.8835) and (0.8715) with SVM and K-NN respectively.

Table of Contents

List of Figures

List of Table

List of Abbreviations

Abbreviations	Meaning
API	Application Programming Interface
APK	Android Application Package
AV	Anti-Virus
GA	Genetic Algorithm
IOS	IPhone Operating System
JVM	Java Virtual Machine
K-NN	K-Nearest Neighbors
PNN	Probabilistic Neural Networks
RF	Random Forests
SVM	Support Vector Machine
PE	Portable Executable
DEX	Dalvik Executable
XML	Extensible markup Language
HTML	Hypertext Markup Language
SQL	Structured Query Language

Chapter One

Introduction

Chapter one

Introduction

1.1 Overview

In the past few years, it was clear that smartphone users have increased exponentially. Besides, the operating systems for smartphones are Symbian, iPhone Operating System (IOS), Android, and Blackberry. The smartphone is viewed as a portable Personal Computer system, PCs, as they have all the functionalities of a desktop PC integrated into them. Just as there are hackers/attackers releasing malware for PCs, there are attackers who are now targeting smartphones. The main reason for this is that mobile security is still in its initial stages and the lack of user awareness regarding how their devices can be undermined by using if they are not careful enough. Google's is open-source operating systems. Android is among the most popular smartphone operating systems. Android is a Linux-based operating system that also includes key applications and middleware. In order to fully benefit from and explore the functionality of Android, Google allows third-party developers to create applications and release them to the Android [1].

A recent work indicates the number of mobile applications is increased extremely which also increases malware application as shown in Figure (1.1) [2].

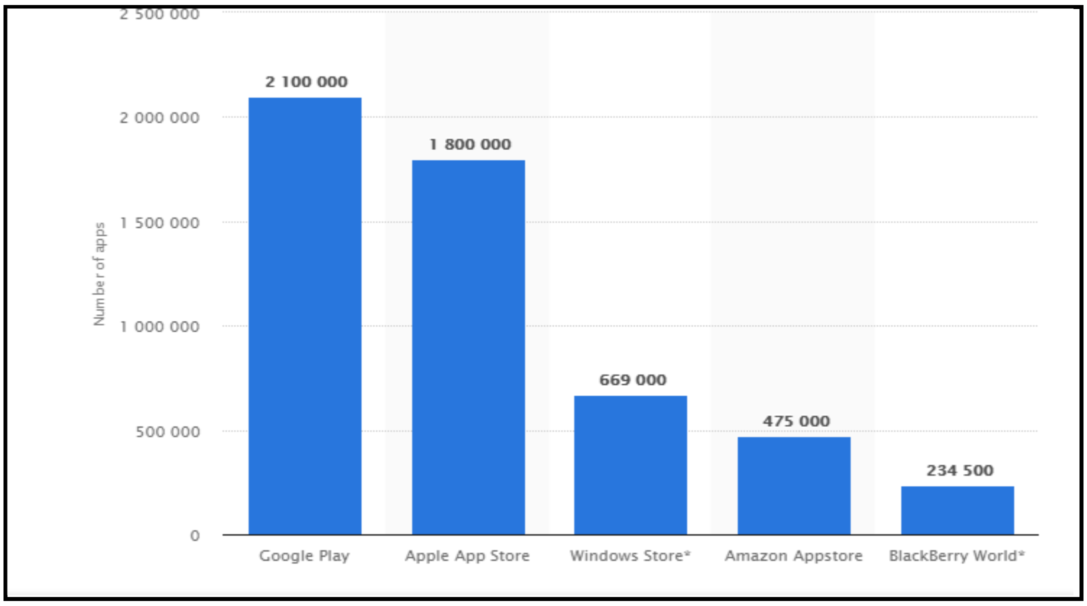


Figure (1.1): Rate of mobile Application downloaded [2]

Malware produced as malicious content that damages the stand-alone computer or networked computers damaged by its harmful effects. It can come in many forms; Spyware, Worm, Trojan or Virus, etc. But whatsoever the form is; its function is the same that is to harm the computers. This intrusive virus can be executable code or no- executable code. Many malware types have the ability to multiply so strict precautions are needed [3].

There are various types of malicious software with different structures, properties, and effects. They also vary in the intensity of the threats they pose [4].

There are different types of malware, the most common one is the virus (an infected code), which after execution multiplies itself and infects other files. It also adds malicious code to other files in order to attack more vigorously. Although viruses are hosted and controlled by a third party, worms create all the damage without being controlled by anyone. They can propagate themselves by infecting the other files. Another side we have that worms are simply standalone malicious software, which creates the same damage as viruses [5].

Another most common type of malware is adware. It is supposed to be used solely for advertisement and generating revenue. However, nowadays adware has been combined with spyware, it keeps track of user’s activities. Addition, Spyware attacks to steal user’s sensitive information. On the other hand, adware is simply the popping up of ads on websites or applications. Normally this comes up with free software. Intruders are making use of it and transform adware into spyware stealing user’s information and misusing it [6].

Spyware is one of the dangerous malwares, which keeps an evil eye on the user’s activities by recording those keystrokes and personal information. Furthermore, personal information can vary from login credentials to sensitive bank account details. Not only stealing users’ information, but also spyware can intrude into the user’ s computer in order to change some software’ s security or privacy settings or browser’ setting making the networks public. The other type of malware is the bot which is an automatic malicious code that intrudes exclusively network of computers. The bots are casually used for positive purposes, but now malicious harmful attacks have been imposed on them. Denial-of-service-attack is an attack on the host computer that transfers its virus to all the networked computers, which is an output of bots. Also, this is a malicious code, which works automatically. Spambots are one of its types, which spam the internet with malicious websites. It is difficult to get rid of such malware but not impossible [7]. Ransomware is also a type of malware, which takes over the hard drive of the user, and the user has to pay some ransom to regain access. It is a crucial kind of malware, which restricts user access to its own computer. Furthermore, it spreads through the downloaded file or any vulnerability in the network service. Trojan horse is a trick to the users; it presents itself as an authentic file which the user can download. Afterward, the attacker gets access to the infected computer remotely. Now the Trojan can add more malware to the infected computer, control the security configurations, monitor user key logs, steal sensitive information, etc. Such infected computers can be used in botnets [8].

The user authentication algorithm of user permission is dependent in order to detect a threat actor after applying the user permission to approach via the improvement of the user activities extraction method.
In addition, to adapt the unknown sessions and use the rule-based on integration with the attacks, the heuristic sequence will be used. A very important aspect is the identification of the classification algorithm of the most reliable detection accuracy. Therefore, the strongest classifiers are identified through the evaluation of the activeness and detection accuracy of all machine-learning algorithms. Modifying the default input values can enhance the efficiency and accuracy of a classifier. However, enabling an equivalent comparison between the classifiers dictated the implementation of the classifiers with their default input values. Many studies have been emerged to discover and treat malicious programs based on the artificial intelligence algorithm by many researches, such as K-Nearest Neighbors (k-NN), Naive Bayes, Random Forests (RF), Support Vector Machines (SVM), and Genetic algorithm [9].

The k-NN algorithm belongs to the family of methods known as instance-based methods. These methods are based on the principle that observations (instances) within a dataset are usually placed close to other observations that have similar attributes; this method selects the closest observations from the dataset in such a way to minimize the distance.

In machine learning, random forest (RF) is already widely used in bioinformatics the best available methods and superior to most methods in common use. As the name suggests, RF combines many classification trees to produce more accurate classifications. By-products of the RF calculations include measures of variable importance and measures of similarity of data points that may be used for clustering, multidimensional scaling, graphical representation, and missing value imputation.

The machine learning techniques are Support Vector Machine (SVM) which is used for binary classification. It is a very general technique that can be applied in a wide variety of situations. Also, it has special characteristics that are used to implement efficient parallel algorithms in terms of time and memory. One characteristic is that the solution to the classification problem is obtained by only a few samples called Support Vectors (SV) that determine the maximum margin separating hyperplane. Another characteristic of SVM is to perform the nonlinear mapping without knowing the mapping function using predefined functions called kernels for calculating the inner product of mapping functions [10].

Genetic algorithm (GA) is a class of stochastic global search techniques based on biological evolution principles; several have applied the genetic algorithm to geophysical optimization problems such as seismic attributes. The genetic algorithm represents parameters as an encoded binary string and works with the binary strings to minimize the cost, while the other works with the continuous parameters themselves to minimize cost [11].

1.2 Related work

Several researchers have shown their interest in the detection of malware in a smartphone, the following are some of the published works that are relevant to the current work:

- **Lu, et al., 2013 [12]:** compared Bayesian method alone and Bayesian method combined with Chi Square feature selection method results are compared to evaluate the performance of the two ML algorithms. The study concluded that Bayesian method with Chi Squared yielded an accuracy of 89% while Bayesian method alone yielded 80%.
- **Nuray Baltaci, et al., 2014 [13]:** The main purpose of the study is to investigate the contribution of other application market metadata to the detection of malicious applications in addition to requested permissions. Hence, the information of applications presented on the official market when a user wants to download them was used as the feature set for training supervised classification algorithms.
- **Kurniawan, et al., 2015 [14]**: used Logger, a default application which is inbuilt in Android was used to extract the sum of Internet traffic, percentage of battery used and battery temperature for every minute. This information collected as set of features and is fed into weka, an open source learning library for testing and training with Naive Bayes, J48 decision tree and Random Forest algorithms. The author concluded that Random Forest has high accuracy of 85.6% with these features and proposes other features that can be combined with existing system to improve the accuracy.
- **Weng, et al. [15]:** in their work, published in 2017, propounded a model that classifies using machine learning techniques such as SVM, the nearest neighbors, by extracting 11 deferent static features. This model can be used in the management of large application markets. A correct classification rate of 99% for the malicious set and 82% for the benign set was obtained working with 100,000 benign and 8,000 malicious application set.

- **Mohsen Kakavand, et al.,2018 [16]:** this work involves in static analysis of applications, which checks for the presence and frequency of keywords in Android application' manifest file and drives the static feature sets from a 400- application dataset to produce better malware detection results. The classification performance of the ML algorithms is measured in terms of accuracy and true positive rate and interpreted to determine which algorithm is more applicable for the Android malware detection. The experimental results for a dataset of real malware and benign applications indicate the average accuracy rate of 79.80% and 80.50% with average true positive rate of over 67% and 80% using SVM and KNN, respectively.

- **Matthew Leeds, et al., 2017 [17]:** Malware is a current threat facing Android users. As users have come to depend on these devices for communication and information, it is essential to make sure they are secure. Therefore, developing and testing new sophisticated malware detection techniques must be a priority. This paper compared two prominent features used to detect Android malware, permissions and system calls, and applied machine learning to both. The results showed that permissions data was better at detecting malware than system call data. An average classification accuracy rate of 80% was achieved when using permissions data to determine malicious activity on Android devices. Therefore, it is a reliable way to detect malware.
- **Michal Kedziora, et al., 2018[18]:** In this study, an overview of Android malware analysis was presented, and a unique set of features was chosen that was later used in the study of malware classification. Five classification algorithms (Random Forest, SVM, K-NN, Nave Bayes, and Logistic Regression) and three attribute

selection algorithms were examined in order to choose those that would provide the most effective malware detection.

- **H al-kaaf1, et al., 2019 [19]:** In this study, proposed feature selection methods to identify clean and malicious applications based on selecting a set combination of permission patterns using different classification algorithms such as sequential minimal optimization (SMO), decision Tree (J48), and Naive Bayes. The experimental results show that sequential minimal optimization (SMO) combining with the SymmetricalUncertAttributeEval method achieved the highest accuracy rate of 0.88, with the lowest false positive rate of 0.085 and the highest precision of 0.910. And the findings prove that feature selection methods enhanced the result of classification.

1.3 Problem Statement

Develop a new method for more classifications performance serve to protect Mobil Application malware.

Applications on smartphones must take Care when Downloading, Due to that, many malicious attacks target them. The majority of operating systems in the smartphone business are operating using the Android OS. However, around 97% of mobile malware targets Android phones. In the Therefore, these incidents motivated us to study mobile application security, especially in Android because the viruses pose risk to the applications as well as the operator.

1.4 Aim of Thesis

The current study aims to find whether or not Google Play, Google's app, and Android's official application market, metadata of Android applications assist in explaining the malicious behaviors when joined with user’s permissions analysis.

1.5 Contribution

Using a hybrid system to detect malware based on static and dynamic approaches. Hence, this contribution will provide protection for the Android mobile based-on access control and anomaly detection.

1.6 Thesis Outline

Beside this chapter, the remaining parts of this thesis include the following chapters:

Chapter Two: Theoretical background

The start with an overview of the Android system architecture and describe the implementation design of Android. Also, discuss an overview of the core components which are found in android applications and it's included in the concepts. That concept is used in this thesis, where the methods used in the malware app such as classifying android apps, SVM, and Genetic Algorithm.

Chapter Three: The propose Detection Malware System

In this chapter, the discussed the proposed system for the Authentication process and the check authentication method with Algorithms SVM as well as the Genetic algorithm.

Chapter Four: Implementation and Experimental Results

This chapter involves studies and results, which are obtained from the system running as well as the performance measures of the results of the test, and comparisons.

Chapter Five: **Conclusion and Future Work**

In this chapter, the present a list of conclusions from the results of the presented work and some suggestions for future works.

Chapter Two

Theoretical Background

Chapter Two

Theoretical Background

2.1 Introduction

This chapter, starts with an overview of the Android system architecture and describe the implementation design of Android and the core components that are found in Android applications and Android malware detection by using a classification algorithm.

2.2 Android and Application Definition

Android is a new mobile platform that was built to be a quite open source. Advanced levels of software and hardware can be used by Android applications along with local data, all exposed by the platform to provide creativity and quality to users. The mechanism of the Android platform should be secure to protect users' data, information, application, and network. The security provided by this open-source platform requires delicate and powerful security architecture. Android is built with multi-layered security. Thus, it is a flexible platform. Also, it provides users' platform with protection. Android includes an operating system, middleware, and core application as a complete [20].

The Android system is composed of five important layers:

1. Applications refer to the software stack of native as well as user-based applications.
2. Android runtime allows the application to run on mobile devices by converting the Android code into DEX format or byte code. The conversion of DEX code into device-related code is done before compilation, and this kind of technique is referred to as ahead of

time.

3. Application framework manages and runs the applications using the services such as activity manager, content providers, telephony manager, package manager, location manager, etc.
4. Android libraries are a set of Java-based development application programming interfaces (APIs) that can help in performing general purpose tasks, as well as location-based and string handling.
5. Android kernel is based on the Linux 2.6 kernel and is used to provide abstraction between device hardware and other software layers. The efforts for making each of the component secure have been made. However, still there are issues due to open-source development, and every vendor and company following their own standards has led to serious security issues. The Android application contains four types of components:

a. Activities: each activity represents a single screen with a user interface.
b. Services: a service operates in the background to execute long-running operations. Services could be initiated by other components like activity or broadcast receiver.
c. Content providers: to share data between different applications.
d. Broadcast receivers: to listen for specific system-wide broadcast announcements and react to them [21].

Android applications are written in Java programming language and distributed as .apk files. An Android application package (APK) file is a ZIP compressed file that includes the following files:

- AndroidManifest.xml file: it describes the application's capabilities and informs the OS about the other components of the application. It identifies the needed hardware and software features such as the camera, in addition to, the minimum API level required by the application. The permissions requested by the app and the permissions required to access the application’s interfaces/data are defined in its manifest file.

- Dalvik executable or classes.dex file: the Java classes and methods defined in the application code are grouped into one single file (classes.dex).
- .xml files: which are used to define the user interface of the application.
- Resources: the external resources that is associated with the application (e.g., images).

Android applications run in a virtual environment to improve security. However, they can be downloaded from any source whether trusted or not. After an application is initiated, it grants its own virtual environment, so the code will be isolated from other apps. Although the applications are isolated, still they can interact with the system and other applications through APIs. Meanwhile, Android assigns Linux user ID for each application [22].

2.3 Application Programming Interface Calls

Application Programming Interface Calls can be written API. These calls provide various interfaces and methods that permit communication between the components of software. APIs enables access to data and key features within Android devices. Generally, a framework of API includes a set of API packages that comprises of certain methods and classes. The details of API components calls may uncover the behavior of an app and allows us to report its capabilities. However, security attackers hide their calls through cryptography, reflection or dynamic code techniques. Therefore, the analysis of an application becomes more difficult [23].

2.4 Android Features Extraction

Android applications are written in Java and executed within a custom Java Virtual Machine (JVM). Each application package is contained in a jar file with the extension of apk. Android applications consist of many components of various types. Each component has an entry point through which the system or a user can enter the application. In addition, there are four fundamental building blocks of an Android app: Activities, Services, Broadcast Receivers and Content Providers. All components must be declared in the application manifest file to be used. Communication between these components is achieved by using intents and intent filters. Intents are messages on objects that can be used to request actions from other application components. However, intent filters are expressions declared in the application manifest file that specify the intent type that a component will receive. Since application components interact via the intent method, it is critical to analyze both the components themselves, as well as their communication intents, for security concerns. Before classification on any model can be done, raw data must be processed. The feature engineering section of malware detection system consists of three parts: Unpacking and Decompiling, Feature Extraction, and Encoding [24] – all shown below in Fig (2.1).

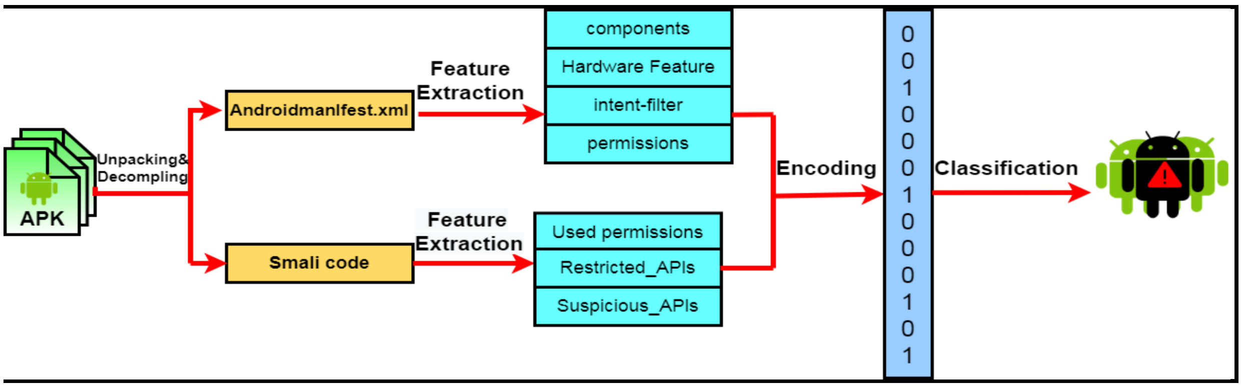


Figure (2.1): System Architecture of Malware Detection Model [24]

a) Unpacking and Decompiling:

Each apk file is actually a specialized zipped file that consists of the application source code, resources, assets, and manifest file. The source code is encoded as dex files (i.e., Dalvik Executable Files) that can be interpreted by the Dalvik VM. The manifest file consists of a number of declarations and specifications. Finally, other resources may contain images, HTML files, etc. Since the dex files are compiled, binary executable code, and therefore not meant to be read or interpreted, features cannot be readily extracted from them directly. Therefore, they must be decompiled into other formats that can be read and interpreted, such as Smali code or even Java code. Smali code is an intermediate form that is decompiled from the dex files; it is essentially the assembly code format of an application. Only after the apk files have been decompiled can continue onto our next step [25].

b) Feature extraction:

Is one of the most important aspects involved in the training of a machine learning model. The upper bound of a given model's performance directly depends on the nature of the features used. After performing a study of the Android system and comparing it to previous work experience in its field, chose to extract seven kinds of features from both the source code and manifest file, of which, the following four types are extracted from a given app's manifest file:

1) **App components:** The components declared in the manifest file define the different interfaces that exist between app and end-user as well as the app and the larger Android OS as a whole. The name of these components is collected to help identify variants of well-known malware, for example the DroidKungFu family

shares the name of several particular services.

2) **Hardware features:** If an application wants to request access to the hardware components of the device, such as its camera, GPS or sensors, then those features must be declared in the manifest file. Requesting certain hardware components or pairs of components may have security implications. For example, requesting usage of the GPS and network modules may be a sign of location leakage.
3) **Permissions Android:** uses a permission mechanism to protect the privacy of users. An app must request permission to access sensitive data (e.g. SMS), system features (e.g. camera) and restricted APIs. Malware usually tends to request a specific set of permissions. In this respect, this is similar to how handle hardware features.
4) **Intent filter:** declared within the declaration of components in the manifest file are important tools for inter-component and inter-application communication. Intent filters define a special entry point for a component as well as the application. Intent filters can be used for eavesdropping specific intents. Malware is sensitive to a special set of system events. Thus, intent filters can serve as vital features [26].

2.5 Android Permissions and Security Model

Android is a privilege-separated system, which employs mandatory access control of all processes. Each Android application has a unique identity in the system, which comprises user ID and group ID. Android system use a default denial approach to control application executions, which means all operations are denied unless a specific permission is explicitly granted. A collection of permissions are defined and aligned to cover a variety of system resources. Each application must declare permissions in a manifest file in order to use corresponding APIs. The declaration of permissions is statistically done, so cannot be done dynamically at runtime. Android system relies on users to decide whether permissions need to be granted to an application, which is done by prompting the user for consent at the time the application is installed. In Android 5.1 (API level 22) or lower version, once an application is installed and all required permissions are granted, there is no further checks with the user while an application is running. Any feature related to granted permission can be used by the application as desired, and any other features beyond the granted permissions will fail without prompting the user. Similar to Linux systems, only the kernel and a small subset of the core applications (usually the pre-installed applications) run with root permission, which is the highest privilege and capable to modify the operating system, kernel. Application with root permission also has full access to all application and all application data. In order to help users and developers understand the importance of permissions, Android has four levels of permission based on their intended use [27].

1. **Normal permissions:** That may have a minor impact on usage, but present minimal risk to the user's privacy or security, and thus will be granted automatically without prompting the user. For example, let the device vibrate upon receiving a message.

2. **Dangerous permissions:** That control access to the sensitive data or resources that involve the user's private information, or could potentially affect the user's stored data or the operation of other apps. For example, SEND_SMS allows an application to send SMS messages.

3. Signature permissions: That has highest privilege, and can be granted only if the application is signed with the device manufacturer’s certificate.

4. Signature or System permissions: Are only granted to apps that are signed or installed in a special system folder. Usually the per-installed apps have this kind of permission. Third-party apps can only use Normal and Dangerous permissions. When they request Signature or Signature or System permissions, they will be ignored. Staring from Android 6.0 (API level 23), the application needs to request permissions from the user at run-time, and the user can revoke granted permission at any time. The application needs to have then permissions every time when it executes. This change may raise the user's awareness and caution at runtime [28].

In general, the security policy for the phones is delegated to their users. The lists of permissions will be shown to the users where they can accept or reject. It is essential and challenging to make sure that these apps appropriately deal with great value sensitive data [29]

2.6 Malware Detection Methods

The term Malware is derived from Malicious Software. Initially, the malware was designed to show man's technical skills, but nowadays it became a profit- driven industry. Over the past few years, the Android system gained a significant popularity, and its current share of the mobile market is more than 75%. This popularity is attributed to its open-source platform and the availability of a big number of feature-rich applications for it. Ideally, these applications should be safe and benign, but due to the adversaries' malicious intent, they can be designed to perform rejected activities in the devices, e.g., sending short message service to a premium number, spying on the user, stealing the personal information, etc. [31].
The techniques of detection display in (2.7.2).

2.6.1 Types of malware and their effects on the system

The best-known types of malware, viruses and worms are known for the manner in which they spread, rather than any specific types of behavior. A computer virus is software that embeds itself in some other executable software (including the operating system itself) on the target system. The user' many not know about it's as well as when it is run, without consent. The virus is spread to other executable. On the other hand, a worm is stand-alone malware software that actively transmits itself over a network to infect other computers. These definitions lead to the observation that a virus requires the user to run an infected software or operating system for the virus to spread, whereas a worm spreads itself, as follows:

a) **Viruses**: - A computer virus is a product generally covered up inside another apparently harmless program. It can deliver duplicates of itself, and insert them into different other programs, or files that normally carries out a hurtful activity (for example, devastating information). Thus, this cases a PE infection, a procedure, and typically used to spread malware. Then it embeds additional information or executable code into PE documents. For more information.

b) **Worms: -** Worms are suitably named for their capacity to creep through systems. Also, they duplicate themselves; however, do not embed themselves in other programs as viruses act general. Furthermore, they move along a network connection looking for vulnerable machines to infect. For example, in 1988, the "Morris Worm" became so widespread that it managed to slow the entire internet [32].

c) **Trojans**: - A Trojan horse is a harmful program that misrepresents itself to masquerade as a regular, benign program or utility in order to persuade a victim to install it. A Trojan horse usually carries a hidden destructive function that is activated when the application is started. The term is derived from the Ancient Greek story of the Trojan horse used to invade the city of Troy by stealth. Trojan horses are generally spread by some form of social engineering, for example, where a user is duped into executing an e-mail attachment disguised to be unsuspicious, (e.g., a routine form to be filled in), or by drive-by download. Although their payload can be anything, many modern forms act as a backdoor, contacting a controller which can then have unauthorized access to the affected computer. While Trojan horses and backdoors are not easily detectable by themselves, computers may appear to run slower due to heavy processor or network usage. Unlike computer viruses and worms, Trojan horses generally do not attempt to inject themselves into other files or otherwise propagate themselves.

d) **Spyware**: - Spyware’s main function is to monitor what you are doing on your computer, on or off the internet, and send that information to a third party without your knowledge. In some cases, this data harvesting is used solely for marketing purposes. In other cases, the intent is more sinister. A theft might occur when an imposter, posing as a client, directs a CPA to send a payment to an illegitimate recipient.

e) **Screen-locking ransomware**: Lock-screens or screen lockers are a type of “cyber police” ransomware that blocks screens on Windows or Android devices with a false accusation in harvesting illegal content, trying to scare the victims into paying up a fee. Jisut and SLocker impact Android devices more than other lock-screens, with Jisut making up nearly 60 percent of all Android ransomware detections [33].

2.6.2 Malware Detection Techniques

With the increasing threats of malware, strict measures have to be taken in order to get rid of maliciousness certain. There are two types of Analyses:

a) **Dynamic analysis:** It is a technique of finding malicious code by executing the program in a sandboxed an environment. In such environment, malware is also detected without bringing the effects in real environment.

Dynamic analysis is also proposed for purposes of tracking data flow in mobile applications, especially in Android malware detection. It focuses on the App's runtime behavior, such as accessing private information without notifying the user or try to send text messages in the background.

Many benign Apps also transmit sensitive data for performing their required functions, especially multi-functional or communication and social Apps like WeChat, Facebook, and Instagram. For example, in Facebook App, one user takes a photo, and then accesses its GPS information, after that he accesses his contact list and send it to one of his friends. In this scenario, the App accesses quite a lot of sensitive information and transmit them through SMS message or the Internet, which should alert the dynamic analysis mechanism and regard as malicious behavior. Thus, taint analyses are not enough accurate and sufficient to automatically.

Differentiate malicious Apps from benign Apps, and the combination of taint analysis with high-level malware signatures is needed to identify malicious code efficiently [34].

b) **Static analysis**: It is an approach of detecting the malicious code without actually executing the computer program. There are tools too for performing static analysis but it includes human-involved techniques as well such as code review, dry run, etc. Nonetheless, with the presence of obfuscation methods, static analysis alone is not enough for detecting malware. So, dynamic analysis is also required. It is essential to integrate these techniques to get better results.

Static analysis can include various techniques:

1. **File Format Inspection:** File metadata can provide useful information. For example, Windows PE (portable executable) files can provide much information on compile time, imported and exported functions, etc.
2. **String Extraction:** This refers to the examination of the software output (e.g. status or error messages) and inferring information about the malware operation.
3. **Fingerprinting:** This includes cryptographic hash computation, finding the environmental artifacts, such as hardcoded username, filename, registry strings.
4. **AV scanning:** If the inspected file is a well-known malware, most likely all anti-virus scanners will be able to detect it. Although it might seem irrelevant, this way of detection is often used by AV vendors or sandboxes to “confirm” their results.
5. **Disassembly:** This refers to reversing the machine code to assembly language and inferring the software logic and intentions. The most common and reliable methods of static analysis [35].

2.6.3 Classification Detection Techniques

Now, having the background on malware analysis, can classify the detection methods.

A. **The signature-based analysis**

Is a static method that relies on predefined signatures. Signature-based analysis keeps track of the behavior of malicious code. It models the behavior of Malware and uses such models afterward in the detection of Malware on the computer software [36].

B. **The anomaly-based analysis**

Is an effective malware analysis that detects zero-day attacks as well. It is comprised of two stages:

- **Training phase**

The detector learns the normal behavior of the program or the code. In the monitoring phase, the program is analyzed for any sort of unusual or abnormal behavior. This is in one way hectic and may generate false Malcode alarms but this is highly useful as it helps in detecting zero-day attacks (the attacks which are previously unknown to the malware detectors).

- **The monitoring phase**

False malware alarms shall be raised. Therefore, it is essential to train or learn all the effective or valid features of the program in the training phase so that no false alarms could rise. In static anomaly-based detection, there is a sound advantage that the program would not necessarily have to be run on the host system. The errors or malware can be detected directly without executing the code. File print (n-gram) Analysis is one of the effective malware scanners, which have core principle of static anomaly-based analysis.

It detects the malware by observing the characteristics of the file types. In file print, (n-gram) analysis models or group of models are developed which have already learned the types of files structurally [37].

C. **The specification-based analysis** is an advanced form of anomaly-based analysis, which removes the threat of generating false alarms of malware detection.

The specification-based analysis is best suited for determining malware in the executable. The executable is first transformed into an intermediate representation of the binary format, which can be assembly code. The assembly language code is then parsed into a parsing tree, which draws an effective control graph [38].

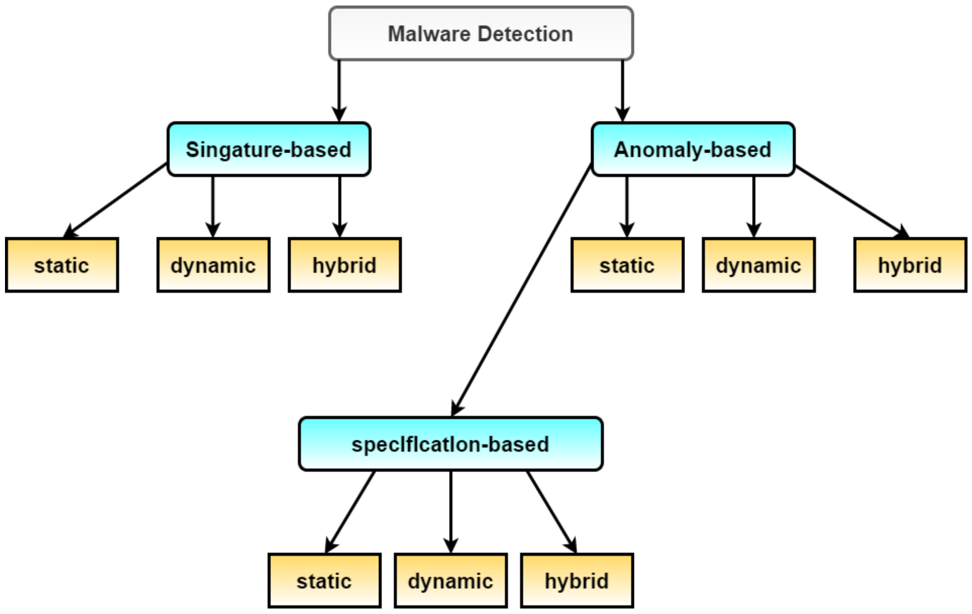


Figure 2.2: Malware Detection Techniques [38].

Categorization Based on Strengths and Weaknesses of Existing Malware Detection Techniques. Figure (2.2) depicts the relationship between the various types of malware detection techniques. Table (2.1) summarized the advantages and disadvantages of each technique.

Table 2.1 Advantage and Disadvantage [38]

Technique	Advantage	Disadvantage
Signature-based	- Recognized known attack precisely. - Minimum system resources are needed to detect attack mode. - Concentrates on normal behavior.	- Un recognized unknown attack method. - Impotent to act against invisible signatures.

Anomaly-based	- Can detect new attack modes. - Concentrates on normal behavior to defeat unrecognized intrusions.	- Necessary to renew information regarding the user's behavior and statistics in normal usage. - Difficult to predict the most suitable solution to detect the oncoming attacks. - CPU time, memory and storage space required additionally. - False positive alarm increasing.
Specification-based	- First time (Unknown) attacks detected. - Incidence of false positive alarm minimal.	-Anomaly detection is more effective in detecting unknown attacks .Better at network probing and denial-of -service intrusions. - Takes too much time to propose specifications with fine details. - Increase false negative due to attacks may be missed. - Possibility of missing increasing false negative alarms due to intrusions.

2.7 Classification Algorithms

This section, the supervised classification algorithms and feature selection methods are chosen and explained briefly as well as their advantages and disadvantages are given. They are selected for working with motivated by their advantages mentioned here, so the former studies applied them. Also, from a machine learning perspective, malware detection can be seen as a problem of classification unknown malware types should be clusterized into several clusters, based on certain properties, identified by the algorithm. On the other hand, having trained a model on the wide dataset of malicious and benign files can reduce this problem to classification. For known malware families, this problem can be narrowed down to classification only – having a limited set of classes, to one of which malware sample belongs, it is easier to identify the proper class, and the result would be more accurate. Some of the algorithm will be explained in the following sections.

2.7.1 K-Nearest Neighbor (k-NN) algorithm

K-NN is an instance-based learning algorithm and instance-based learning algorithms do not explicitly apply a target function of training data. The classification that uses the concept of distance of classifying data objects. The k-NN classifier is one of the simplest and most widely used in such classification algorithms.

The most used method for continuous variables is generally the Euclidean Distance, which is defined by the formulae below:

$$EuclidianDistance\colon \sqrt{\sum_{i=1}^{n}(pi-qi)^2} \qquad (2.1)$$

Where: $p\ and\ q\ are\ the\ points\ in\ n-space$

The main concept for k-NN depends on calculating the distances between the tested, and the training data samples in order to identify its nearest neighbors. The tested sample is then simply assigned to the class of its nearest neighbor [39].

2.7.2 Random forest algorithm

Random forests or random decision forests are a combination learning technique produced by growing numerous classification trees that contribute collectively to the latest judgment based on the general majority [40].

Random forests are an ensemble learning method for classification, regression and other tasks that operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees [41]:

$$H(x) = \underset{Y}{\operatorname{argmax}} \sum_{i=1}^{k} I(h_i(x) = Y) \qquad (2.2)$$

Where $H(x)$ is combination of classification model, h_i is a single decision tree model, Y is the output variable, I(.) is the indicator function. For a given input variable, each tree has the right to vote to select the best classification result.

2.7.3Support Vector Machine (SVM) algorithm

The main idea of SVM training is to find the optimal hyper–plane that separates two classes of training patterns, $\{(xi, yi)\}_{i=1}^{n}$, yi ∈ {+1, −1}, where n is the number of training patterns.

- The optimal hyper–plane will be:

$$f(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + b = 0 \qquad (2.3)$$

Where w^T is weight vector, x is input value, b is bias.

The margin region can be defined as the region between f(x) = 1 and f(x) = −1. Because SVM employs the structural risk minimization principle, the minimum margin of each class of patterns, 1/||w||, is to be maximized in SVM training, and all training patterns should be located on the margin boundary or outside the margin region. However, there can be some training patterns located inside the margin region or even in regions representing other classes of patterns [42].

- Maximize -margin problem can be defined as:

$$\text{Minimize } \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n} \xi_i \qquad (2.4)$$

Where (C > 0) is the trade-off cost between the empirical error margin. By introducing Lagrangian multipliers, the primal optimızatıon problem can be converted into a dual form:

$$\text{Maximize} \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j \tag{2.5}$$

Where $0 \leq \alpha i \leq C$, i = 1… n

$$\sum_{i=1}^{n} \alpha_i y_i = 0. \tag{2.6}$$

The solution of SVM is

$$\mathbf{w} = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i, \quad \text{thus} \quad f(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i^T \mathbf{x} + b. \tag{2.7}$$

2.7.4 Genetic Algorithms (GAs)

The process of GA includes the initial population, selection, crossover, and mutation. At the same time, to maintain population diversity and avoid premature convergence and speeding up the convergence, a niche strategy and an elitist strategy are incorporated into the traditional genetic algorithm. The specific process is as follows [43]:

(a) Initial Population

The initial population is generated randomly. As considered, the length of an encoding chromosome depends on the number of stages. Each gene of this chromosome is defined by choosing randomly one of the nodes in the corresponding stage [44].

(b) Selection

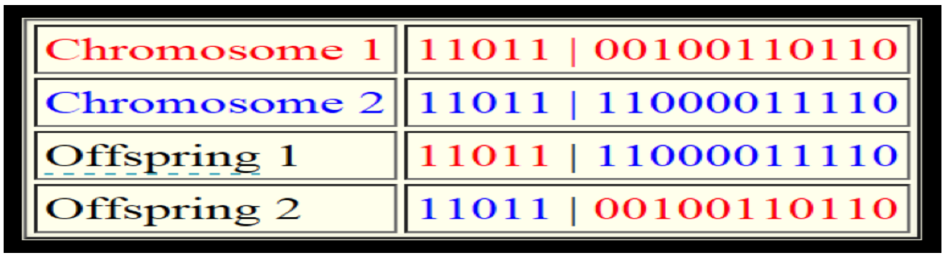
There are many different techniques that a genetic algorithm can use to select the individuals to be copied over into the next generation.

- **Elitist selection:** The fittest members of each generation are guaranteed to be selected.

- **Fitness-proportionate selection:** More fit individuals are more likely, but not certain, to be selected.
- **Roulette-wheel selection:** A form of fitness-proportionate selection in which the chance of an individual's being selected is proportional to the amount by which its fitness is greater or less than its competitors.
- **fitness Scaling selection:** As the average fitness of the population increases, the strength of the selective pressure also increases and the fitness function becomes more discriminating. The method can be helpful in making the best selection later on when all individuals have relatively high fitness and only small differences in fitness distinguish one from another.
- **Tournament selection:** Subgroups of individuals are chosen from the larger population, and members of each subgroup compete against each other. Only one individual from each a subgroup is chosen to reproduce [45].

(c) Crossovers:

Mechanism used in GA to develop the best solution is to crossover the genes from the parent chromosomes .Crossover is an operator for creating new offspring from two parents having the properties of both the parents. At present, numerous crossover operators are application dependent. Chromosomes selected for next generation will be exposed to crossover and mutation operation. It is evident that the modified partially matched crossover having a random two-point mutation performed best due to its capability to reach closer to the optimal solution [46]. Figure (2.3) shows the example of crossover.



Chromosome 1	11011 | 00100110110
Chromosome 2	11011 | 11000011110
Offspring 1	11011 | 11000011110
Offspring 2	11011 | 00100110110

Figure (2.3): Examples of crossover [46].

(d) Mutation

The operation of mutation is altering individual alleles locations of random chromosomes at variable locations and probability e.g. from 0 to 1 or 1 to 0 (Figure 2.4 and 2.5). The mutation slightly is a random operation that might enhance or damage the chromosome, which will either diminish or thrive through the next selection. The goal is finding the optimum; hence, one single unfortunate solution will temporarily cause damage to the population. In addition, generating an acceptable solution will be very helpful [47].

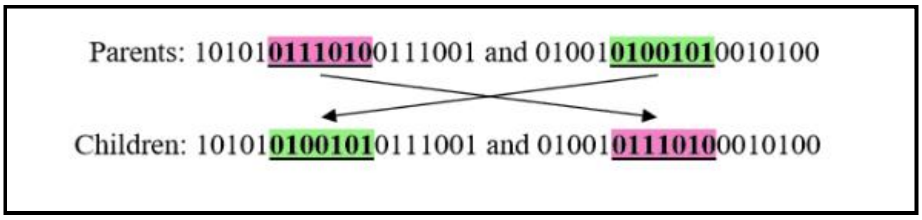

Parents: 10101**0111010**0111001 and 01001**0100101**0010100
Children: 10101**0100101**0111001 and 01001**0111010**0010100

Figure (2.4): Two-point crossover [47]

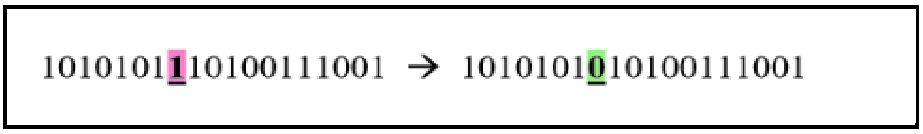


Figure (2.5): Mutation [47]

2.7.5 Probabilistic neural networks (PNNs)

The probabilistic neural network (PNN) is a promising machine learning technique that can be used to forecast financial markets with higher accuracy. The main limitation of PNN is the proposition of Gaussian distribution as the division of input variables which is violated with respect to financial data.

PNN provides global optimal solutions that lead to more accurate results. In addition to that PNN is a feed forward neural network with four layers: the input layer, pattern layer, summation layer and output layer, which are explained in Figure (2.6) [48].

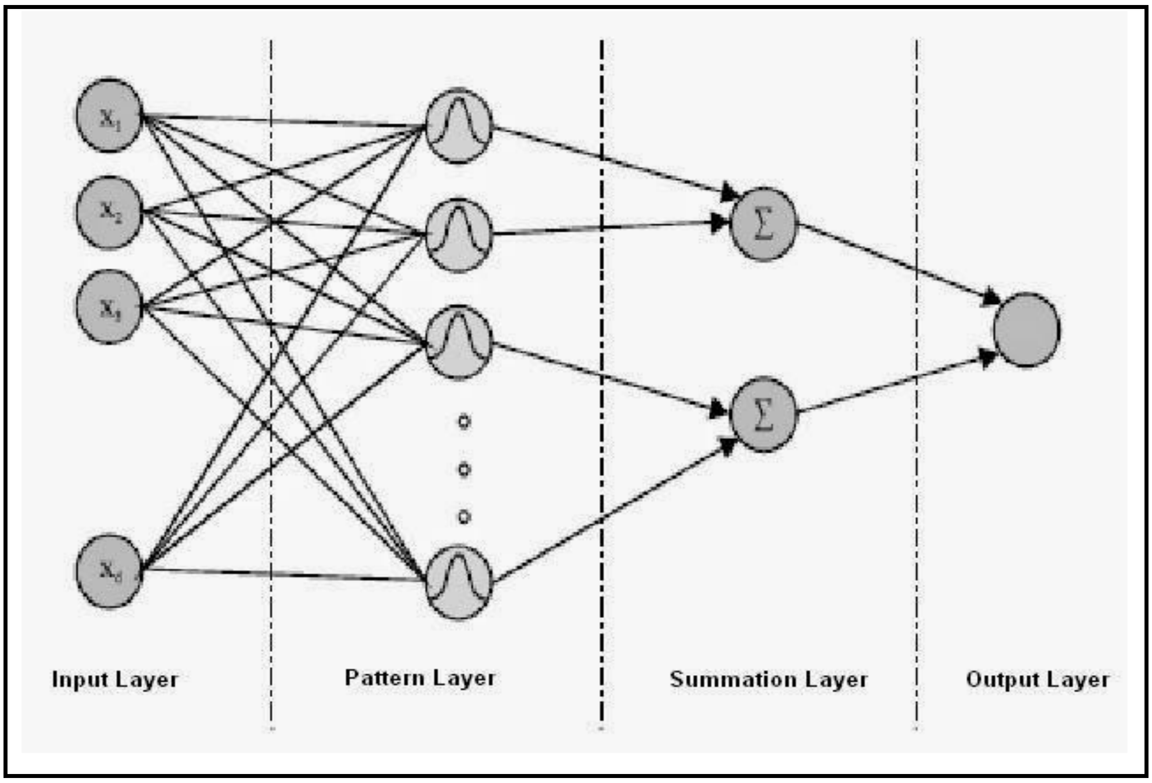


Figure (2.6): PNN general architecture [48]

The following sets up the PNN algorithm:

Step 1: Read in the file of exemplar vectors and class numbers

Step 2: Sort these into the K sets where each set contains one class of vectors Step 3: For each k define a Gaussian function centered on each exemplar vector in set k define the summed Gaussian output function.

2.8 Parameters used to Evaluate Classification

In machine learning studies, the performance of proposed models are evaluated by utilizing some well-known metrics. After running classification algorithms, a matrix named as confusion matrix is obtained including the actual and predicted number of instances with respect to class values. The representation of a sample confusion matrix is provided below [49]:

Table (2.2): Sample Confusion Matrix

		Predicted class	
		Positive	**Negative**
Actual class	**Positive**	True Positive	True Negative
	Negative	False Positive	False Negative

In this thesis, malicious applications are accepted as positive instances and benign applications as negative ones. Hence, the cells of the confusion matrix for this thesis can be interpreted as follow:

- ➢ True Positive (TP): Number of malicious applications that correctly classified.
- ➢ False Positive (FP): Number of benign applications that are incorrectly classified as a malicious, false alarm
- ➢ True Negative (TN): Number of benign applications that correctly classified, correct rejection.
- ➢ False Negative (FN): Number of malicious applications that are incorrectly classified as benign.

These values provided by the confusion matrix are used to calculate performance measures for the assessment of classification algorithms. The following measures derived from confusion matrix (except kappa statistic) is adopted in this study:

- ➢ Accuracy (ACC): Rate of correctly predicted applications to the total number of applications

ACC= TP+ TN / (TP+ TN+ FP+ FN)

- True Positive Rate (TPR): Rate of correctly predicted malicious applications to the a total number of malicious applications, a.k.a. recall rate

  TPR= TP/P = TP / (TP+FN)

- False Positive Rate (FPR): Rate of incorrectly predicted benign applications as malicious, to the total number of benign applications

  FPR= FP/N = FP / (FP+TN)

- True Negative Rate (TNR): Rate of correctly predicted benign applications to the total number of benign applications

  TNR= TN/N = TN / (TN+FP)

- False Negative Rate (FNR): Rate of incorrectly predicted malicious applications as benign, to the total number of malicious applications

  FNR= FN/P = FN / (TP+FN)

- Precision: Rate of correctly predicted malicious applications to the total number of predictions as malicious, a.k.a. positive predictive value.

  Precision= TP/ (TP+ FP)

Chapter Three

The proposed Detection Malware System

Chapter Three

The proposed Detection Malware System

3.1 Introduction

In this chapter, that presents an overview of intrusion detection methodology in Android mobile application. In specific, this chapter illustrates architecture of the malware detection based-on access control and two hybrid methodologies Support Vector Machine (SVM) and Genetic Algorithm (GA).

3.2 Proposed System Architecture

The proposed system aims to detect malware based on a hybrid system of anomaly and access control, which are the proposed technique of detection. The access control begins at first at the launch of the system. Since the control access determined the file as in abnormal, then beginning of the anomaly detection technique for further detection and certain results. Figure (3.1) shows a general description of the proposed system to detect the intrusions over Android mobile applications.

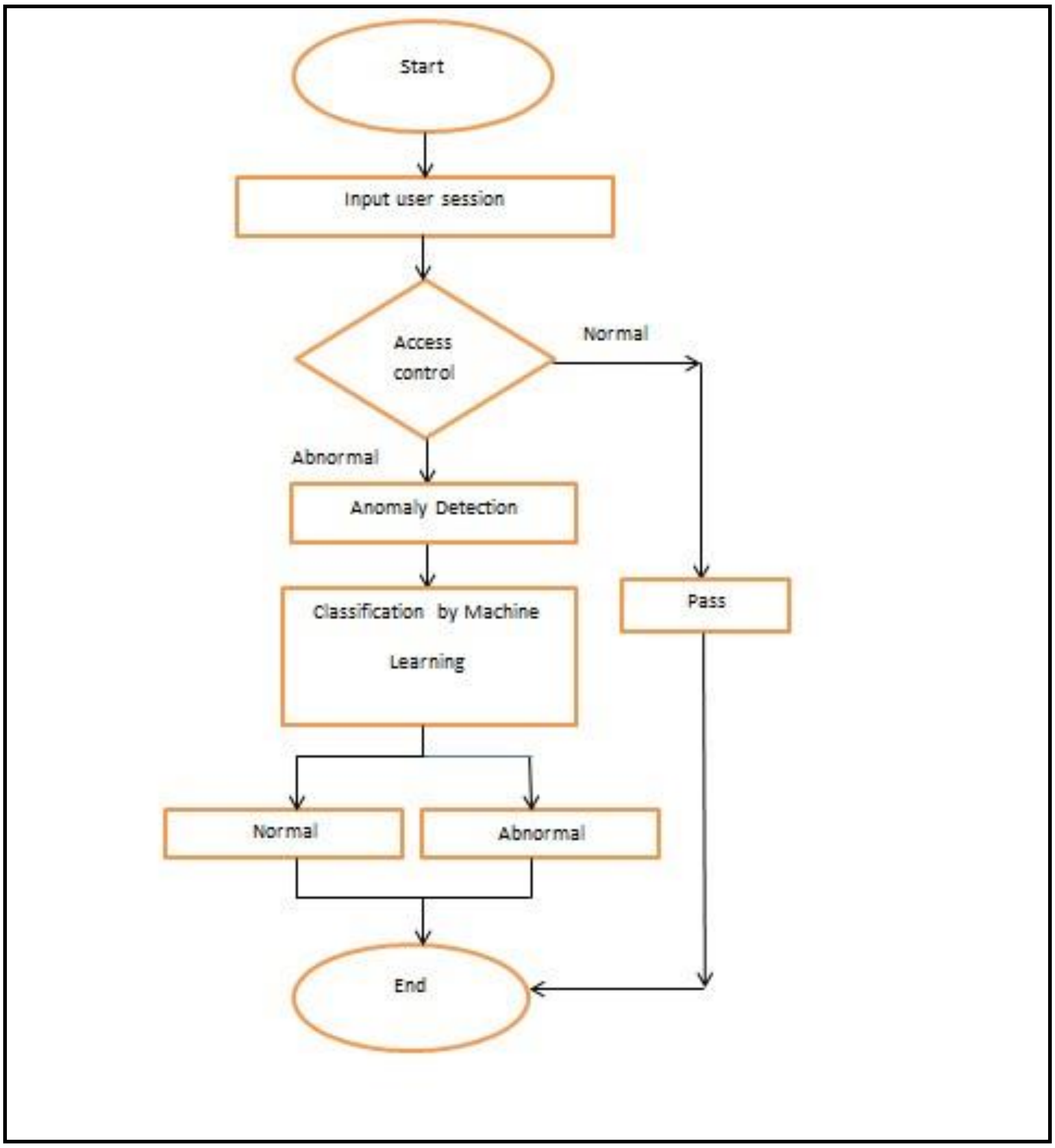


Figure 3.1 General flowchart of the proposed system

3.3 Access control detection

The following hypotheses are to be investigated to detect malware code:

Hypothesis 1: Mobile application that possesses malicious behaviors produces heavy traffic and decrease the performance of the enterprise web application dramatically.

Hypothesis 2: Robust policy-based detection approach which is installed on mobile can maintain enterprise data flow and security of the enterprise a mobile application deploy it.

Proposed System of the access control for malware detection that illustrated in figure (3.2)

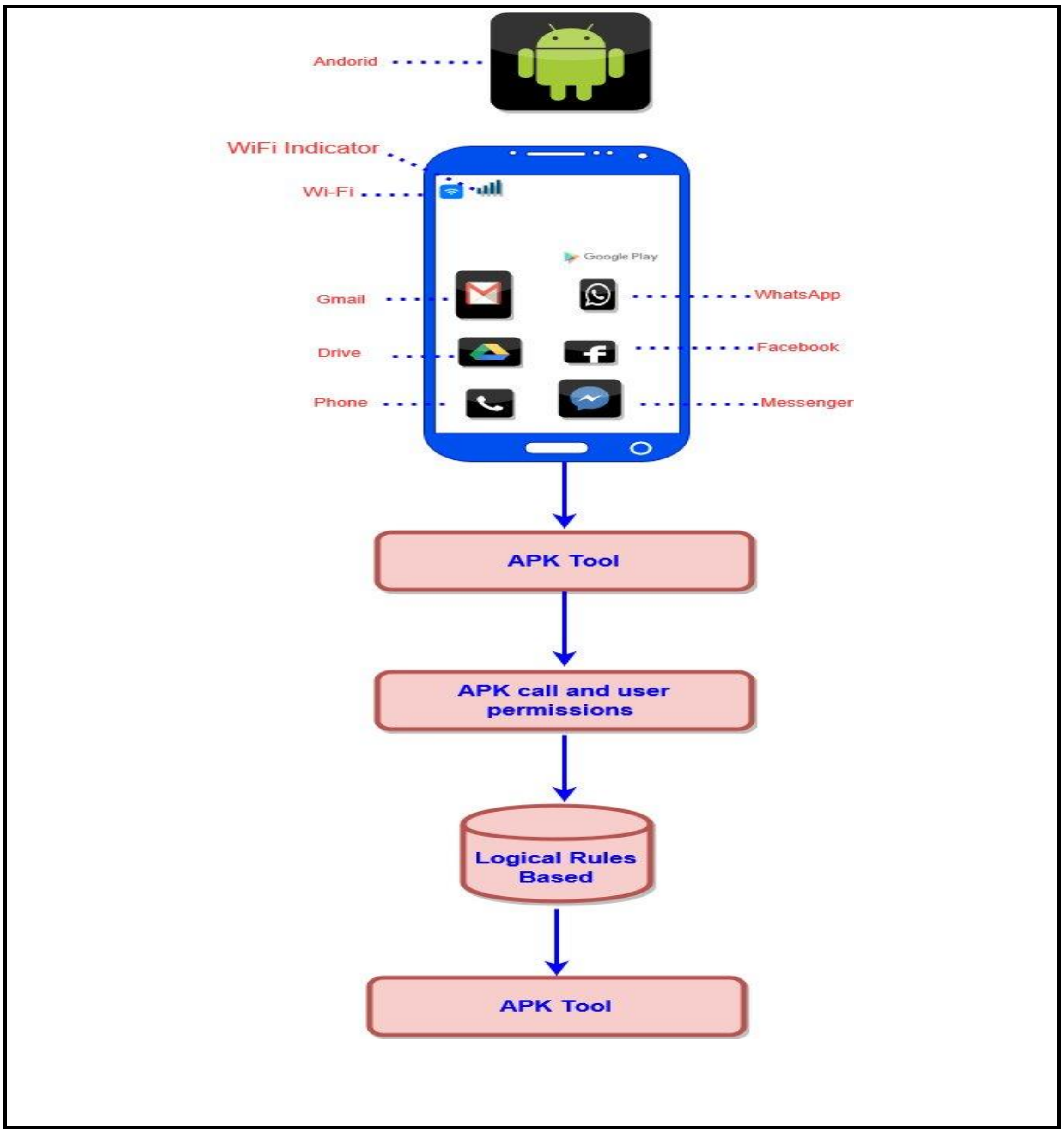


Figure (3.2): Access control detection

3.3.1 User permission approach:

Basically, the malware has a behavior like a real user once to get application permission. The key difference between them is that human operator to download mobile apps. Malware with virtual is given to be an automated attack code. It will contact a provider, and download an app.

The user authentication algorithm is the first step that is proposed to detect malware; the user permission technique detection begins at first at the launch of the system. This proposed system model is investigating the theory of determining user access with a privileged context, and has no detectable security signature; the abnormal behavior is proposed to be detected by the authentication tool fired within the system at any moment that the user tries to download the application, as shown in Figure (3.3).

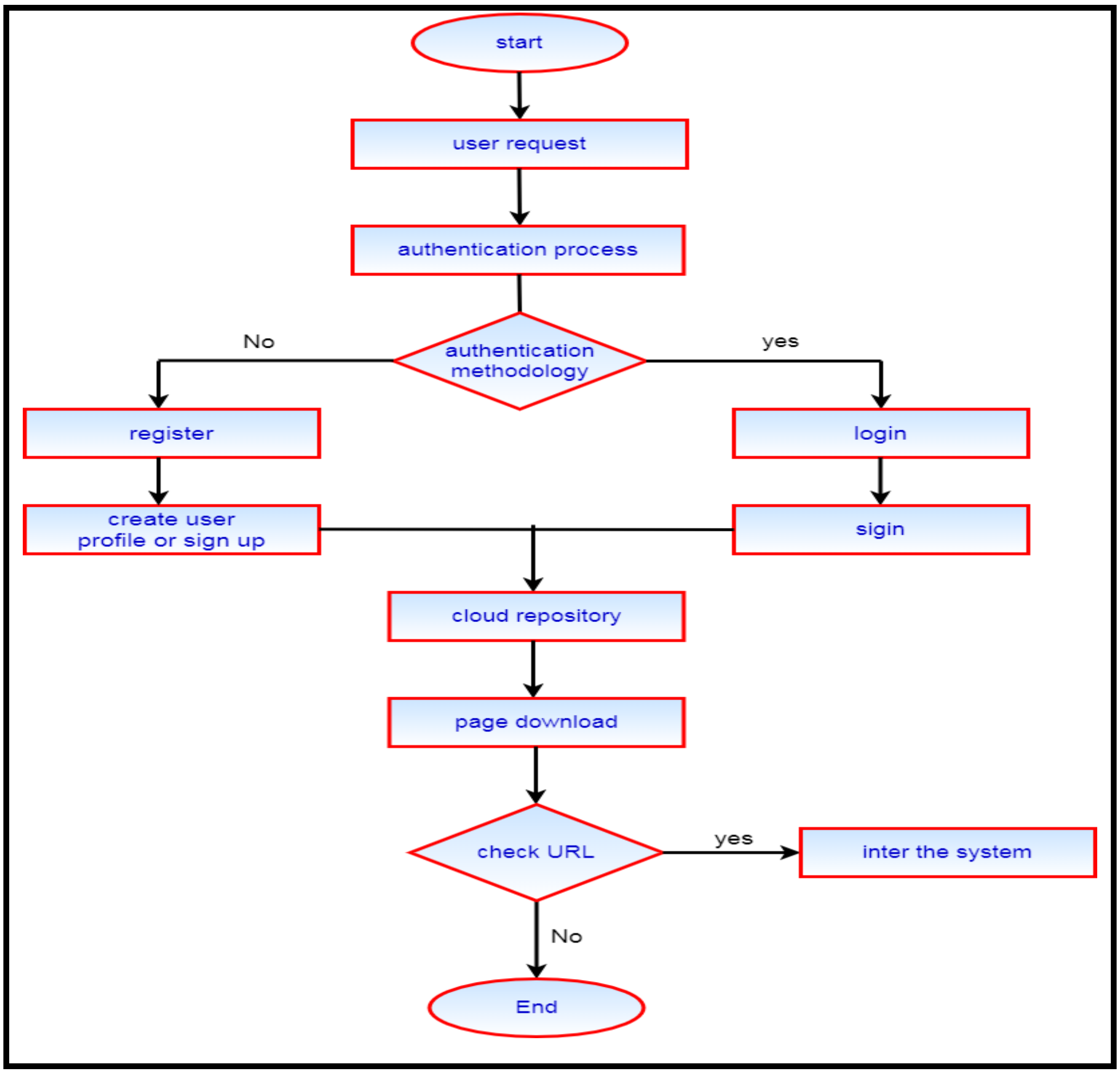


Figure (3.3): Authentication flowchart

a. **User request:** The user login process. The user can login based on the user name and password.
b. **Authentication process**: It is the process of logging by using an account with protection for data in the Database. (In other words, the application will be accessed only through logging in process, and that is its default mode).
c. **Check authentication methodology**: If the user has an account, he will log in into the system. On the other hand, if he does not have an account, he must perform the registration process by creating an account.
d. **Cloud repository**: A single place for a store, manages, and track code.
e. **Page download:** The downloading Page extension for download Monitor enables the downloads to be performed from another page.
f. **Check URL**: check the URL page download to explore the malicious page that activates the malware code. If the URL trusted, it enters the system, and if it is not trusted, it exits the system.

3.3.2 Logical Rules-based system

The logical rules-based defense is a manifestation of a modern technological quality breakthrough in the field of counteracting malware threat. It replaces the analysis of the file on the disk with an analysis of the behavior and activity of programs in the system.

Anti-malware does not have enough confidence about actions to be taken in most frequent situations when encountering malicious and suspected code inside a file. Many options are thrown like delete, skip, or add to trusted, but there is no way to make sure what action to be conducted. Anti-virus will resolve this confusion by getting the user involved in the decision.

The logical rules based defense component protects user from both known and new threats, whose information has not been added to the databases yet. All user and application activities are controlled and centered by rules defense with the help of authentication (models of suspicious application activities). If, as a result of activity detected, the sequence of the application's actions arouses any suspicion, the component blocks the activity of this application.

Now, let's define the traffic of the mobile application to be the data flow produced by mobile service invocation over the internet.

$$\text{Let } LR = \begin{cases} \text{Allow} & \text{If Manifest elements reference} \\ \text{Deny} & \text{Else} \end{cases}$$

Where LR = logical rules-based action toward the invocation of certain mobile service which has its corresponding description in passed Manifest elements reference.

Manifest elements reference Access rules are another approach to resolving the security issue. Through specifying authorization access rules and representing private data in cloud computing environments, security is addressed using this approach. Logical rules-based allows users to access secure data, and device files with restricted authorization. It also controls who or which application in the mobile computing system can be used.

logical rules leverage with the system based on the following: -

a. Check the characters in the string:

All know that can use structured statements or correct substitution/formatting rules to avoid SQL injection in our application. Nonetheless, looking at MySQL's list of literal Characters, found that it contains the following characters:

Now, while the percent and characters need to be escaped in order to prevent the insertion of unauthorized wildcards into the LIKE statements, and while the '(single quotation), \ (backslash) and '(double quotation) all need to be escaped in order to prevent the insertion of arbitrary SQL-may have some of these other undetected characters leading directly to an SQL injection vulnerability that would not otherwise have occurred.

b. This rule related to how many logins attempts that a user tries to do it:

From the login can identify how many logins the user did. Also, by using the intent filter can protect user from a threat actor invading the smartphone app.

c. Check the activity of components:

The activity of component in android suite API can be used to identify if the user makes different logins, using different user names and passwords through which abnormal behavior. Therefore, we can identify the activity as malware behavior.

d. Check the category name

Adds is the name of the group to the goal list. See Intents and Purpose Filters are used for information on the purpose filters and the role of the category specification within the filter. The name of that group. Standard categories are specified as CATEGORY name constants in the Aim class.

e. Check the screen configuration

When you take the same direction that Nick mentioned, but then go to Preferences > Display > Display Resolution, it will give the user choices for screen size. As for user pictures, take a picture, put it on screen, and check the resolution. Whether the resolution changed this can identify the threat actor exists, because somehow the threat actor can change the resolution to ease the hacking by labeling the app.

3.4 Anomaly detection

The current study aims to find whether or not Google Play, Google's app, and Android's official application market, metadata of Android applications assist in explaining the malicious behaviors when joined with user's permissions analysis. Therefore, it was required to collect the application's metadata on Google Play. First, a kind of web automation and testing tool and a browser-based macro recorder, use of Google Play Crawler was to acquire the permissions requested by applications and to download them as apk file. Permissions collected for the top unpaid applications, there are 851 different permissions as a whole in each application category, and some of them consist of developer-defined permissions. When users visit the Android application's page on Google to download, this crawler gives them incomplete information. Therefore, a Java application was applied for collecting other metadata of applications. The Applications were downloaded from (https://github.com/cskamil/Android-Application-Dataset-for-Malware-Detection). The consumed time for data collection is 12 hours approximately. Form each application category, the dataset includes top free ones, and totally they make 17244 applications on the date of collection. The following information is included in the mentioned Google metadata of applications in this study:

a. **Application category**: It refers to the definition of each category of an application given by the official market. The two main categories are applications and games.
b. **Developer's name**: It holds the name of the person or the company that developed the application. It can also belong to institutions such as metropolitan municipalities that provide transportation service through applications.
c. **Developer type:** Displays if a given application builds by the top developer's official market or it is editors' choice application. Editors' choice applications correspond to the applications as some of the best applications chosen by Google Team. While Google promotes top developers and this is utilized by developers for advertisement and financial gain, by users for finding more trustworthy applications. This attribute has three values for the dataset used in this study: Top developer, Editors' Choice Top Developer, Editors' Choice.
d. **Rating stars:** It a (5) starts counter that controlled by the users who download the application. There is no lower limit to show star ratings for an application on its page. If no user has rated the application, the starts rating will show zero on the application page. The rating star page is valid for all additions, in other words, this page will always appear with the application.
e. **Average rating of application:** It depends on ratting stars of an application. Also, it represents the average of ratting stars. It's maximum and minimum values are 100 and 0 respectively, since the application may not be rated at all. The current study involved applications with 0 rating stars and 0 average rating.
f. **Publish date of an application:** It displays the date of the current version of a given application. The developers are able to release new updates of their applications and the versions are the number of them. These

versions, then, are uploaded on the markets. The publish date of the most recent version of the application is displayed on the bottom right area of the application page under the label “Updated”.

g. **Application size**: It is a number made of values string from 11 to 994,304 bytes for the dataset used in the current study. Yet, this value for some application “varies with device”, therefore, and to make these application fall into one category.
h. **Minimum required Android OS version**: At the developing phase, the developers ask about the minimum required level of Android API, which is an inter number used for the compassion of Android platform on a mobile device with the Android application. If the API of the user’s mobile system is lower than the one stated on the application page as the minimum required API, then a given application will not run on that device.
i. **Application content rating**: While developers upload their applications on Google Play, they are required to rate the content of them. Dataset used in the current study utilized the categories “high maturity, low maturity, medium maturity and everyone” and the level of harm caused by the content is selected due to Google guideline (such as for: sexual content, drug, tobacco, gambling, alcohol, violence and hate).
j. **Number of application downloads in form of range**: This number shows the total users’ installations of a given application as an interval (e.g. 10,000-50,000).

After market-related information of applications are collected (including permissions), the second step is acquiring the target values of applications (malicious or benign) to have the ability of training a train a supervised classification algorithm. For the purpose, several antivirus engines along with Virus Total (free online service identifying malicious content) and web scanners are required. Malicious content can be found using Virus Total, the content includes (domains, IP addresses or webpages by querying URLs) and in different formats (including apk files) uploaded by the user to the VirusTotal or queried by using the hash values of files. To scan files by using hash values requires less time than uploading apk files to the web site one by one, so by using the "shortcut" (VirusTotal) provided by VirusTotal and sending http requests, the most recent reports of applications were obtained with hash values.

3.4 Machine learning technique for classifying dataset:

In this thesis, is accomplished before the application of all classification and feature selection algorithms and their combinations with a number of selected features. This thesis, aimed to find an estimated performance of the detection model, which is proposed, in the current study. The use of a genetic algorithm and SVM classifier came for a faster results generation and comparison. The first step included collecting the entire dataset of applications from the official market; it was used for applying the Genetic algorithm and SVM learning algorithm. Data of relation to the application market was gathered through writing a Java application for all application classes defined by Google Play and for top free applications under each of them. A Google Play crawler was used to collect the permissions requested by the application at the time of installation, it also is used to download applications again.

Downloading the applications is carried out to figure their hash values to query them via an online free AV engine (Virus Total). The collected applications, then labeled due to the number of AV engines detecting them as malicious. An application is defined as malicious if 2 or more AV engines recognize it as malicious, otherwise, it is benign for the initial phase. The investigation of the construction of official market metadata was performed using various datasets; one of which includes market metadata and users' permissions, while the other contains only the users' permissions. The same number of instances is included by these two datasets, consisting of 4512 malicious and 12719 benign applications. Show in the figure general flowchart of the anomaly detection (3.4).

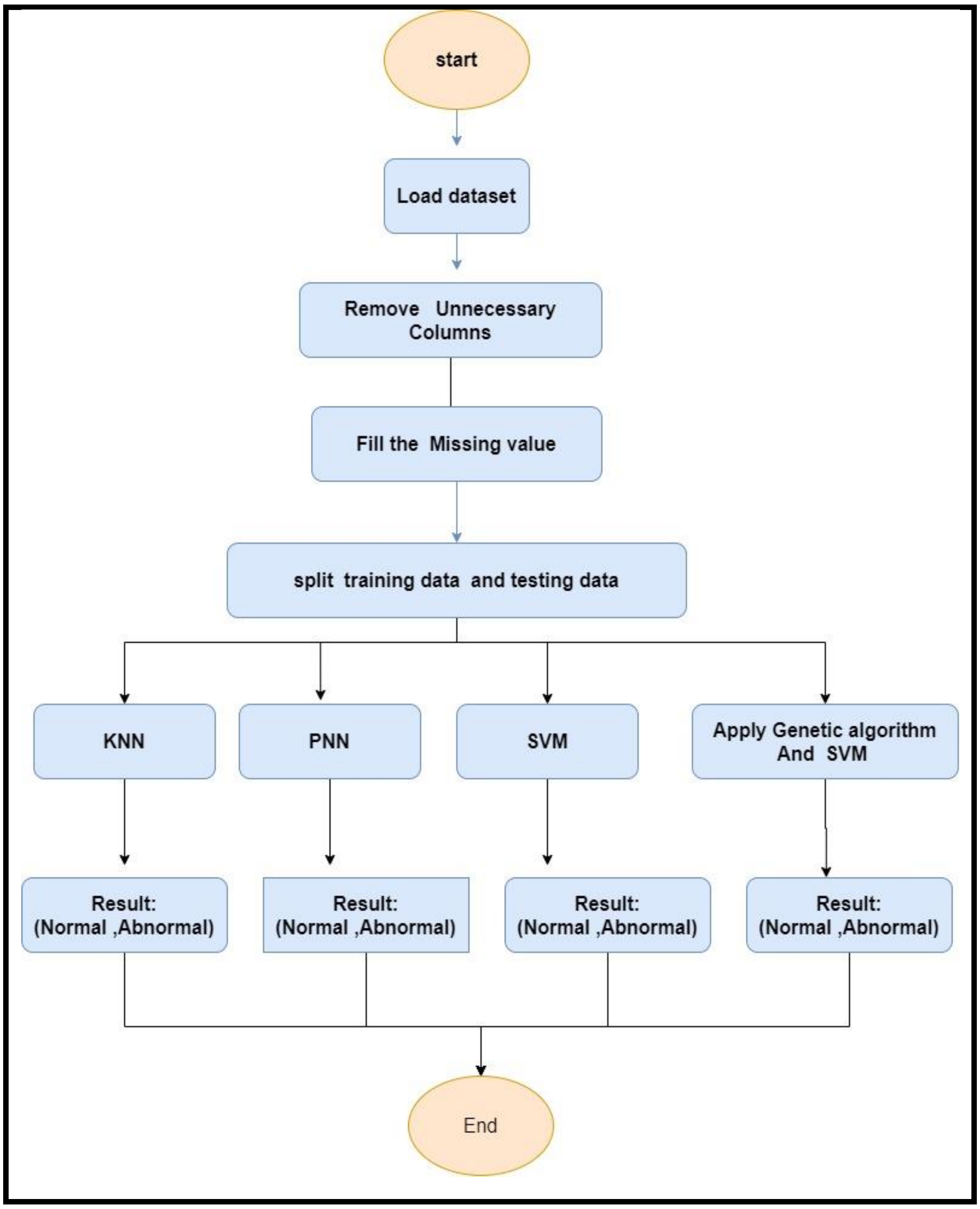
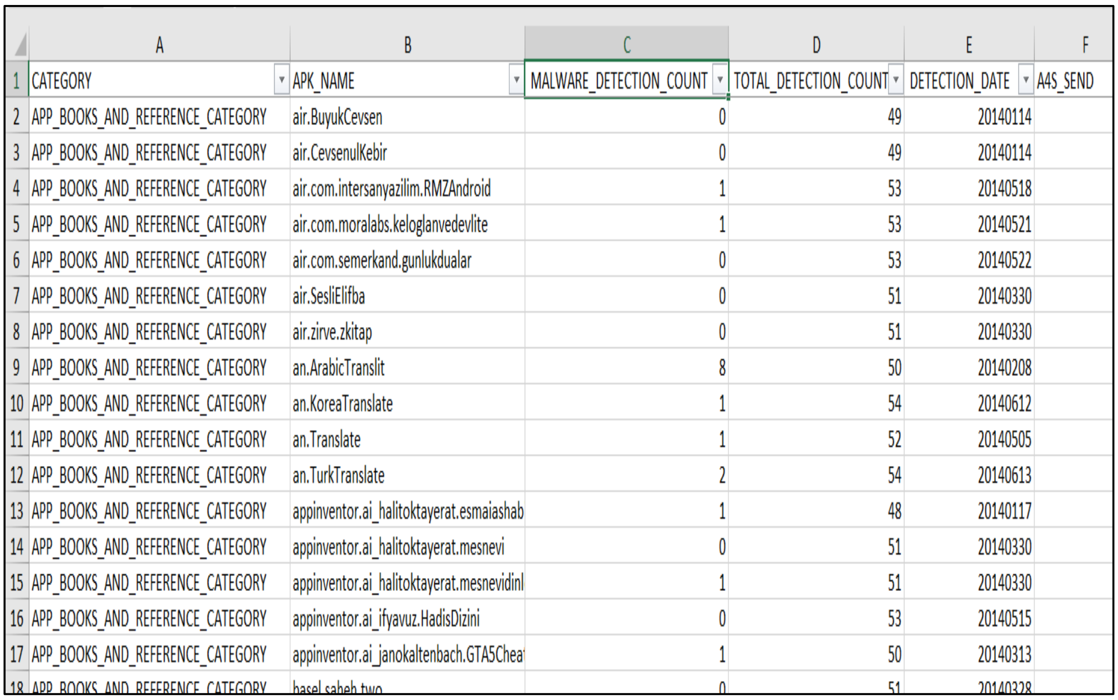

start
Load dataset
Remove Unnecessary Columns
Fill the Missing value
split training data and testing data
KNN
PNN
SVM
Apply Genetic algorithm And SVM
Result: (Normal ,Abnormal)
Result: (Normal ,Abnormal)
Result: (Normal ,Abnormal)
Result: (Normal ,Abnormal)
End

Figure (3.4): General Block Diagram (anomaly detector)

3.5.1 Cleaning dataset (remove unnecessary columns):

This step is the very basic steps required to work through a large data set, where data cleaning is the detection and correction (or removal) processes of the inaccurate or corrupt records from the



	A	B	C	D	E	F
1	CATEGORY	APK_NAME	MALWARE_DETECTION_COUNT	TOTAL_DETECTION_COUNT	DETECTION_DATE	A4S_SEND
2	APP_BOOKS_AND_REFERENCE_CATEGORY	air.BuyukCevsen	0	49	20140114	
3	APP_BOOKS_AND_REFERENCE_CATEGORY	air.CevsenulKebir	0	49	20140114	
4	APP_BOOKS_AND_REFERENCE_CATEGORY	air.com.intersanyazilim.RMZAndroid	1	53	20140518	
5	APP_BOOKS_AND_REFERENCE_CATEGORY	air.com.moralabs.keloglanvedevlite	1	53	20140521	
6	APP_BOOKS_AND_REFERENCE_CATEGORY	air.com.semerkand.gunlukdualar	0	53	20140522	
7	APP_BOOKS_AND_REFERENCE_CATEGORY	air.SesliElifba	0	51	20140330	
8	APP_BOOKS_AND_REFERENCE_CATEGORY	air.zirve.zkitap	0	51	20140330	
9	APP_BOOKS_AND_REFERENCE_CATEGORY	an.ArabicTranslit	8	50	20140208	
10	APP_BOOKS_AND_REFERENCE_CATEGORY	an.KoreaTranslate	1	54	20140612	
11	APP_BOOKS_AND_REFERENCE_CATEGORY	an.Translate	1	52	20140505	
12	APP_BOOKS_AND_REFERENCE_CATEGORY	an.TurkTranslate	2	54	20140613	
13	APP_BOOKS_AND_REFERENCE_CATEGORY	appinventor.ai_halitoktayerat.esmaiashab	1	48	20140117	
14	APP_BOOKS_AND_REFERENCE_CATEGORY	appinventor.ai_halitoktayerat.mesnevi	0	51	20140330	
15	APP_BOOKS_AND_REFERENCE_CATEGORY	appinventor.ai_halitoktayerat.mesnevidinl	1	51	20140330	
16	APP_BOOKS_AND_REFERENCE_CATEGORY	appinventor.ai_ifyavuz.HadisDizini	0	53	20140515	
17	APP_BOOKS_AND_REFERENCE_CATEGORY	appinventor.ai_janokaltenbach.GTA5Cheat	1	50	20140313	
18	APP_BOOKS_AND_REFERENCE_CATEGORY	basel.saheb.two	0	51	20140328	

record set or database. It is also the recognition of irrelevant, incomplete, inaccurate or incorrect parts of the data then delete, modify, or replace this dirty or coarse data, as shown in the figure (3.5).

Figure (3.5): Data sets

In this work, all the same fields are deleted in terms of data, as similar data adversely affects the learning model, and the mechanism of work is to take the first values in the column and subtract them from all cells in the field, then take the sum of cells if the value is not equal to zero, so this means that all the fields are similar as shown in algorithm (3.1).

Algorithm (3.1): Clean the Dataset
Input : Load Dataset
Output: Clean dataset
▪ **Step 1**:Read dataset ▪ **Step 2**: split of each column ▪ **Step 3**: Take the first cell and subtract on all cell in column ▪ **Step 4**: sum the result of step three , if the result zero delete the current column ▪ **Step 5**: read the next column and go to step three until complete all column

3.5.2 Filling in the missing value in the data set:

After cleaning the dataset the process of filling in the missing in the dataset where the null cell is searched and compensated with the intermediate value from the column, the Fill-in or impute the missing values. The remaining data is used to anticipate the missing values. The replacement of the average value with the missing value of a predictor is a simple method. There is the possibility of using regression on other predictors. The effects of the diagnostics and inference on the filled-in dataset remain unclear. Some additional uncertainty is caused by the imputation which needs to be allowed for, as shown in algorithm (3.2).

Algorithm (3.2): Fill The Missing Value
Input : Load Dataset
Output: Complete Dataset
▪ **Step 1**: Read dataset

- **Step 2**: split of each column
- **Step 3**: calculate the average value of column,
- **Step 4**: find the null cell and filling with result from step 3
- **Step 5**: read the next column and go to step three until complete all column

3.5.3 Classification with SVM

The learning model is built to categorize the data after preparing the input dataset. Also, the missing values have been cleaned and filled data.

An SVM model is created to categorize the dataset. Before building the learning model, the training data and test data are separated. Thus, the training data include 70% of the total data, and the model is built on this basis. This means that data entered into the SVM model include 600 raw in 167 columns as shown in Algorithm (3).

Algorithm (3) : SVM Classifier

Input : attribute of dataset

Output: classification dataset

- **Step1:** Training(420) raw and testing (180) raw , knowing that the data is divided into four categories
- **Step2:** SVM applied as describe bellow

  Where Input

  D: the Android malware detection dataset

  L: The number of sample

  m : the fraction of pattern in a sample

- **Step3** : output

  S: classy to normal and abnormal

- **Step4** : concord Dj for j=1 ...,m by 70% sampling from the dataset ;

Train SVM with $D_{j,j}$ by apply equation (2.3) and (2.5) where α=0.2 and start with random initial weight

- **Step5** : Testing SVM : Evaluate all D by $f_{j,j}$ as shown in equation (2.3)

  t= target value

  y= output value

  calculate the mean square error

  $MSE= (t-y)^2$

- **Step6:** Return the class of dataset

3.5.4 Classification with Genetic Algorithm and SVM

The learning model is built to categorize the data after preparing the input dataset where the missing values have been filled and cleaned data. In this work, the genetic algorithm was used to reduce the data entered into the SVM model, in order to obtain the highest accuracy and less time in determining the data class. The conditions that must be met are in the chromosome .The stays should not be repeated. The fitness function depends on the SVM function as shown in algorithm (3.4).

Algorithm (3.4) : Hybrid Genetic Algorithm And SVM Classifier

Input : Dataset

Output classification dataset

- **Step1:** After cleaning and missing value The Android malware detection data consists of(600 raw and 167 column)
- **Step2:** Random population generation, where 1000 individuals (chromosome) are generated and each chromosome contains 20 cells, as a cell contains the column heading in the used dataset, with no similar numbers being repeated in one chromosome

- **Step3**: Fitness function, the calculation of the efficiency of each chromosome using SVM (each chromosome contains 20 columns of data is taken ignore the rest and the accuracy of the data entered into the SVM is calculated, knowing that the separation of training data and test data is 70% randomly)
- **Step4**: Selection operation, Two chromosomes are randomly selected as parents.
- **Step5**: Crossover operation, In this step the two chromosome exchange is separated and the second part of the first chromosome is replaced by the second part of the second chromosome using (single point) and the formation of sons
- **Step6:** Mutation operation: from the previous step, two or more numbers may be repeated. In this step, the number repeated is replaced with a new number.
- **Step7:** Fitness The fitness calculation for children is produced using SVM
- **Step8:** Update the population
- **Step9:** Repeat previous operations (from the selection step, reach the desired goal or 100 iterations).

3.5.5 Classification with PNN

The learning model is built to categorize the data after preparing the input dataset where the missing values have been filled and cleaned data.

The PNN model is created to categorize the dataset, before building the learning model the training data and test data are separated so that the training data include 70% of the total data and the model is built on this basis This means that data entered into the PNN model include 600 raw in 167 columns as shown in pseudo-code as show bellow.

- Training (420) and testing (180), knowing that the data is divided into four categories
- PNN applied as describe bellow

```
Procedure:   classify_ PNN ( T, X, D, PT)
real array class Probabilities
real exponent;
integer array result
  for each    X (k)  in T    do
    for each   X ( r)  in X    do
      exponent= 0;
        for    i=1    to      D
          exponent  += (x(i k)    - X (ir) ) 2 / σ(ij) ;
        end for
      class Probabilities j+= exp( exponent ) ;
   end for
 for each j in class Probabilities do
   Class Probabilities (j)/ =((2π ) ^(D/2) ). PT(j)$\prod_{i=1}^{D}$ σ(ij);
End for

  results(i)= Find Array Max (class Probabilities)
end for
  return  (results)
```

- Return the class of dataset.

3.5.6 Classification with KNN

The learning model is built to categorize the data after preparing the input dataset where the missing values have been filled and cleaned data.

The K-NN model is created to categorize the dataset, before building the learning model the training data and test data are separated so that the training data include 70% of the total data and the model is built on this basis this means that data entered into the KNN model include 600 raw in 167 columns as shown in Algorithm(3.6).

Algorithm (3.6) : KNN Classifier
Input : Dataset

Output classification dataset

```
▪ Step1: Training (420)raw and testing (180)raw , knowing that the data
  is divided into four categories
▪ Step2: for each y belongs to D do
      calculate the distance D(y,x) between y and x
  end for
• Step3: Select the subset N from the dataset D
  The N contains K training samples which are the k
  Nearest neighbors of the test sample x
  Step 4: Calculate the category of x:
  $c_x = \text{argmax} \Sigma I (c=class(y))$
  end
▪ Step5: Return the class of dataset
```

Chapter Four

Implementation and Experimental Results

Chapter Four Implementation and Experimental Results

4.1 Introduction

This chapter includes the results of the suggested system, where the user system is a hybrid made of static analyses like anomaly detection.
The machine learning methodology is very important for Android applications. There are four different algorithms to be implemented to determine malware through which experimental results for the suggested system are obtained for malware accurate determination.
The method to detect malware in Android is using an Analysis anomaly and depend-on the four algorithms extracted from APK files to construct a new hybrid classifier (genetic algorithm and SVM) to classify Android applications to benign and malignant.

4.2 Result presentation

However, the availability of software in retail on Android, most programmers prefers the android studio program because many of the features are the most important: It seems that you can type the code incorrectly, since you can type it. The space of the emulator or Android Emulator provides good speed and performance, when it is developed and equipped using operations resources such as screen zooming and minimizing, multi-touch, etc., The program Written in language

JavaScript uses the drag and drop system, It can also be shared through various kinds of different versions of Google Cloud for Developers..

The user must have a unique user ID to register and be eligible to access the web service. In the next step, the system will automatically check user cases if he/she allows each one to do according to the access permission.

This investigating is the theory of determining user access with a privileged Context, and has no detectable security signature; the abnormal behaviour is proposed to be detected by the authentication tool fired within the system at any moment of time that the user tries to download the application.

Basically, what are after is to build a detection control tool that can determine the threat actor behaviour, our system starts once the user sends a request to download App, then the authentication method. The Smartphone is personalizing users through the user logs in and register. Figure (4.1) shows the user login process. In this case, the user can log in based on the user name and password.

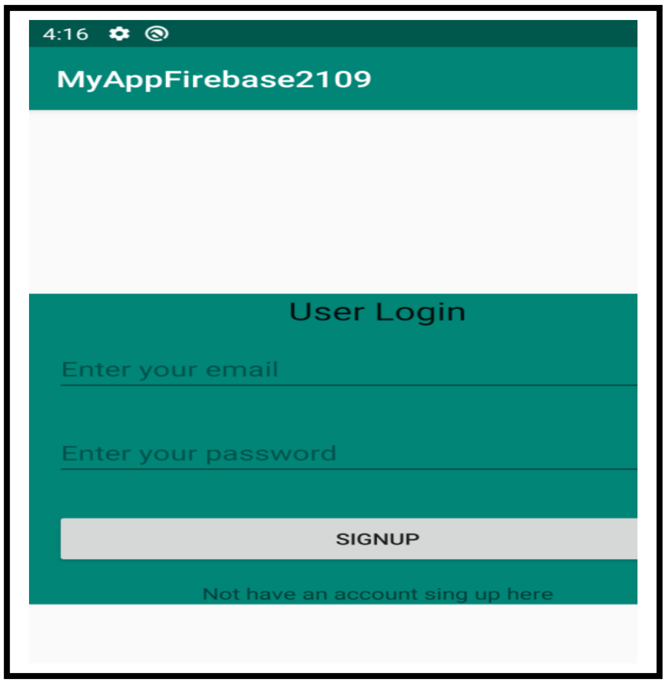


Figure (4.1): User login

If the user doesn't have an account cans he /she does sign up.

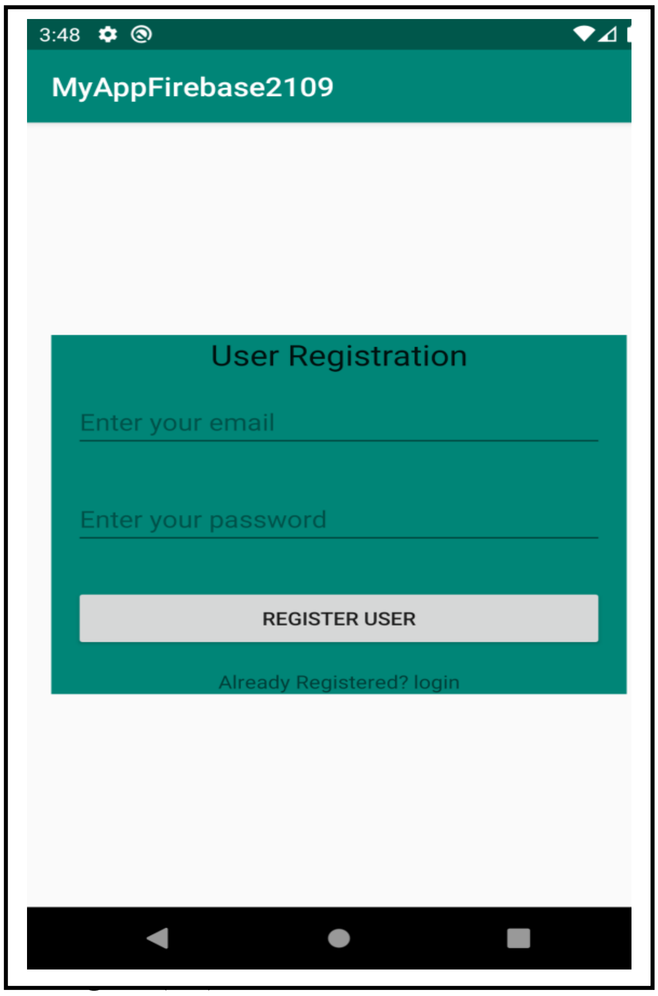


Figure (4.2): User Registration

Add robust password creation process security; put 6 characters in password creation condition, as figure (4.3)

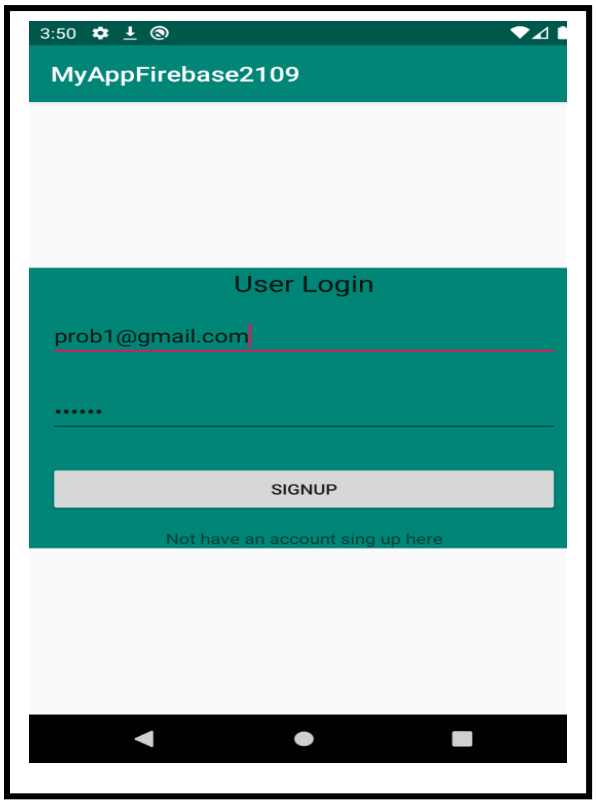
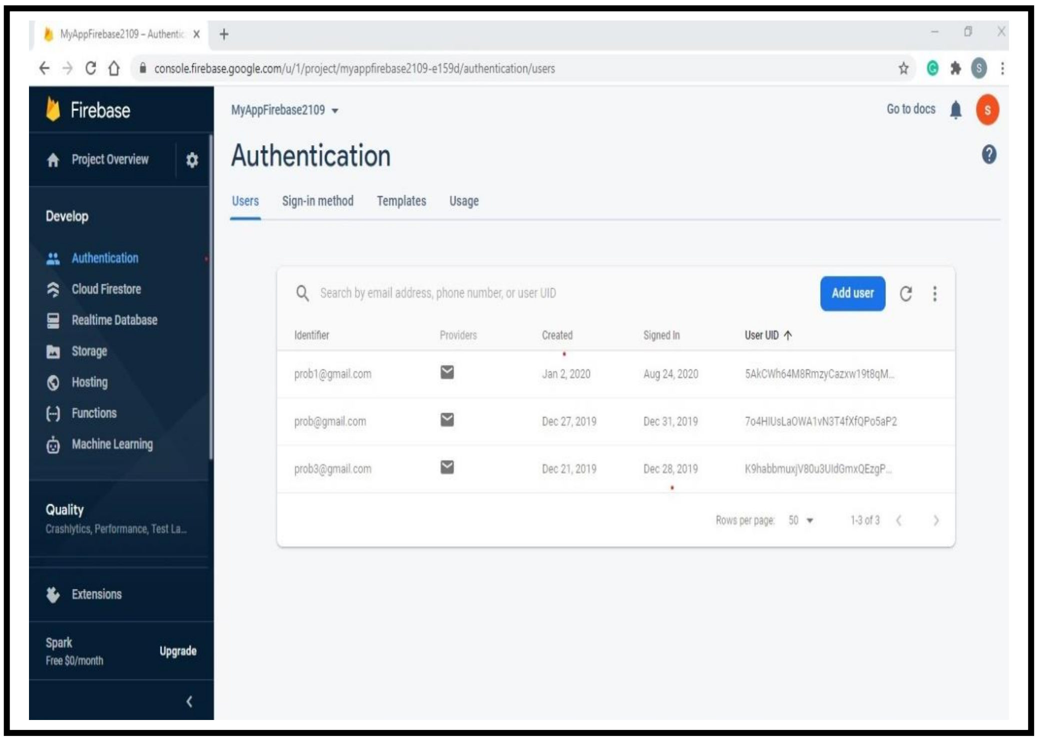


Figure (4.3): Login process

In **figure (4.4)** explore all users that match with registration and login process.



Identifier	Providers	Created	Signed In	User UID ↑
prob1@gmail.com	✉	Jan 2, 2020	Aug 24, 2020	5AkCWh64M8RmzyCazxw19t8qM...
prob@gmail.com	✉	Dec 27, 2019	Dec 31, 2019	7o4HIUsLaOWA1vN3T4fXfQPo5aP2
prob3@gmail.com	✉	Dec 21, 2019	Dec 28, 2019	K9habbmuxjV80u3UIdGmxQEzgP...

Figure (4.4): Shows all users that match with registration and login process.

A database cloud knowledge base could repository is used to take advantage of the platform's capacity and enchased our solution. The goal is to significantly improve the implementation of data access layers by reducing the commitment to the actual amount needed. The second protection methodology is used, as seen in Figure (4.5).

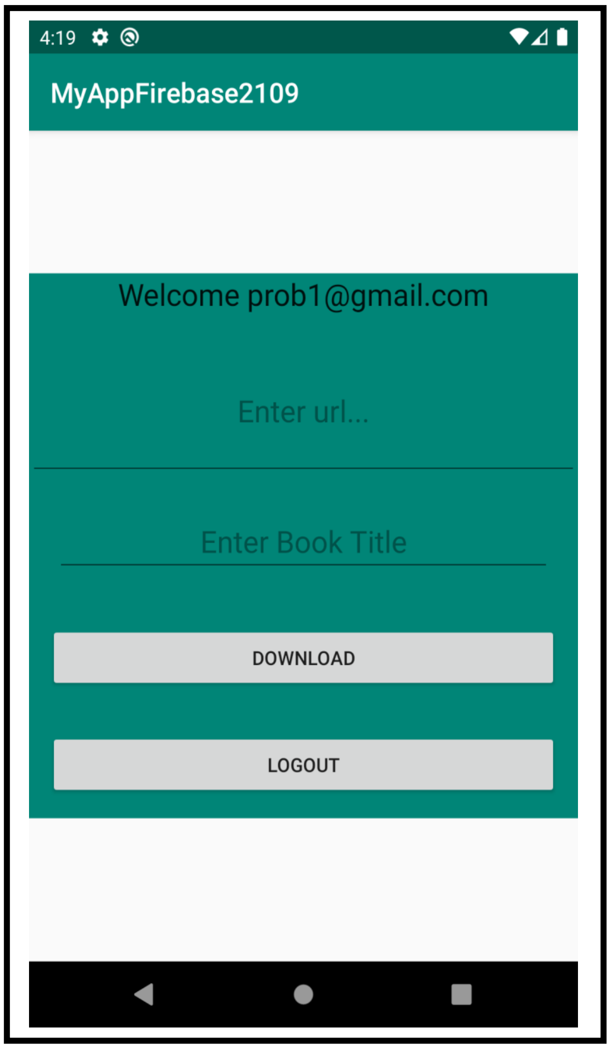
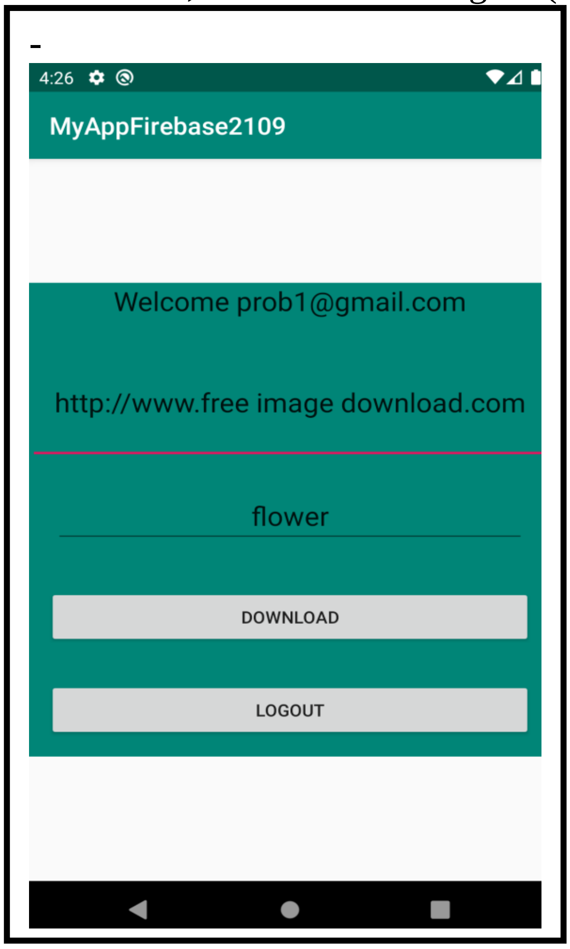


Figure (4.5): URL page download

Enter URL and enter book title, as shown in the figure (4.6)



4:26

MyAppFirebase2109

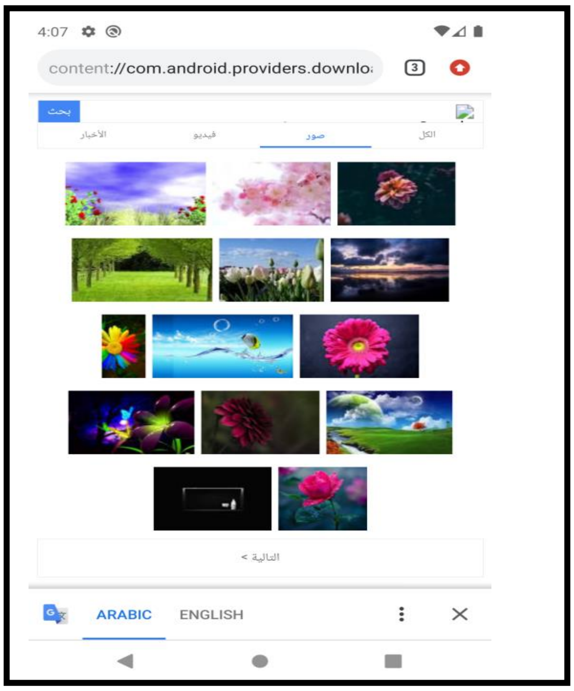
Welcome prob1@gmail.com

http://www.free image download.com

flower

Figure (4.6): Enter URL

If the URL is trusted, download the page, as the figure (4.7)



4:07

content://com.android.providers.downlo

Figure (4.7): Download the page

4.3 Results about Datasets and Configuration for Algorithms

Datasets were built in this study to show if any contribution is made by other official market metadata of Android applications to the malware detection model that only make use of requested permissions from users as a feature set. The first dataset comprises only requested permissions and the second one has additional information about applications presented by Google Play Store. Additionally, one more feature was used as an input for classification algorithms taken from Virus Total website, which is the number of passing days after detecting data of a malicious application. The presented datasets include malicious applications that have been identified as malware by at least 2 or more AV engines on Virus Total. If any of the AV engines on the date of data collection have not identified an application then it is benign. The other ambiguous applications were removed from the dataset.

Cross-validation was not preferred because a total of 40 algorithms were run together with combinations of classification and feature selection algorithms and this many algorithms would require more time to complete if cross-validation was used. Partition was done in a manner that the datasets included 70% training and 30% testing samples. First, the classification algorithms, PNN, SVM, hybrid Genetic algorithm and SVM, k-nearest neighbor, were run solely, and then feature selection algorithms were run in combination with classification algorithms.

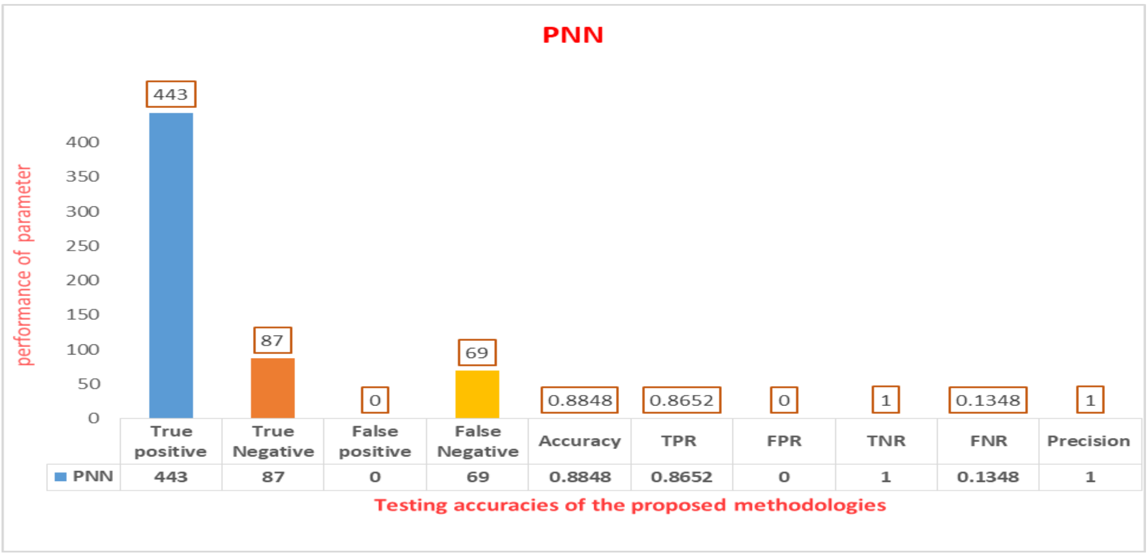
4.3.1 The PNN Detection Malware in Android

Training Data Simulation **70%** of the database was used for training. **420** samples of the training data belong to benign class and **180** samples belong to malignant class. The classification results of the training set by PNN, were given in the Table (4.1)

Table (4.1): Classification of training PNN

	Normal	**Abnormal**
True	443	87
False	0	69

PNN classifier has nearly perfect accuracy value of 88.86% when original features without select the best feature or reduce the amount of feature the performance parameter of PNN is (True positive=443, True Negative=87, False positive=0, False Negative=69, Accuracy=0.8848, TPR=0.8652, FPR=0, TNR=1, FNR=0.1348, Precision = 1), as shown in Figure (4.10).



	True positive	True Negative	False positive	False Negative	Accuracy	TPR	FPR	TNR	FNR	Precision
🟦 PNN	443	87	0	69	0.8848	0.8652	0	1	0.1348	1

Figure (4.8): Testing Accuracy using PNN

4.3.2 The K-NN Detection Malware in Android

In the proposed work apply KNN to detection to detect malware in Android; k-nearest neighbor (KNN) search lets you find the *k* closest points in INPUT to a query point or set of point's label. The *K*NN search technique and KNN-based algorithms are widely used as benchmark learning rules. The relative simplicity of the *K*-NN search technique makes it easy to compare the results from other classification techniques to K-NN results.

Training Data Simulation 70% of the database was used for training. 420 samples of the training data contained by benign class while 180 samples contained by malignant class. The classification results of the training set by K-NN, were given in Table (4.2)

Table (4.2): Classification of test KNN

	Normal	Abnormal
True	414	108
False	35	42

K-NN classifier has nearly perfect accuracy value of 88% when original features without select the best feature or reduce the amount of feature the performance parameter of K-NN is (True : positive=414, True: Negative=108, False: positive=35, False: Negative=42, Accuracy=0.8715, TPR=0.9079, FPR=0.2448, TNR=0.7552, FNR=0.0921, Precision = 0.922), as shown in Figure (4.11).

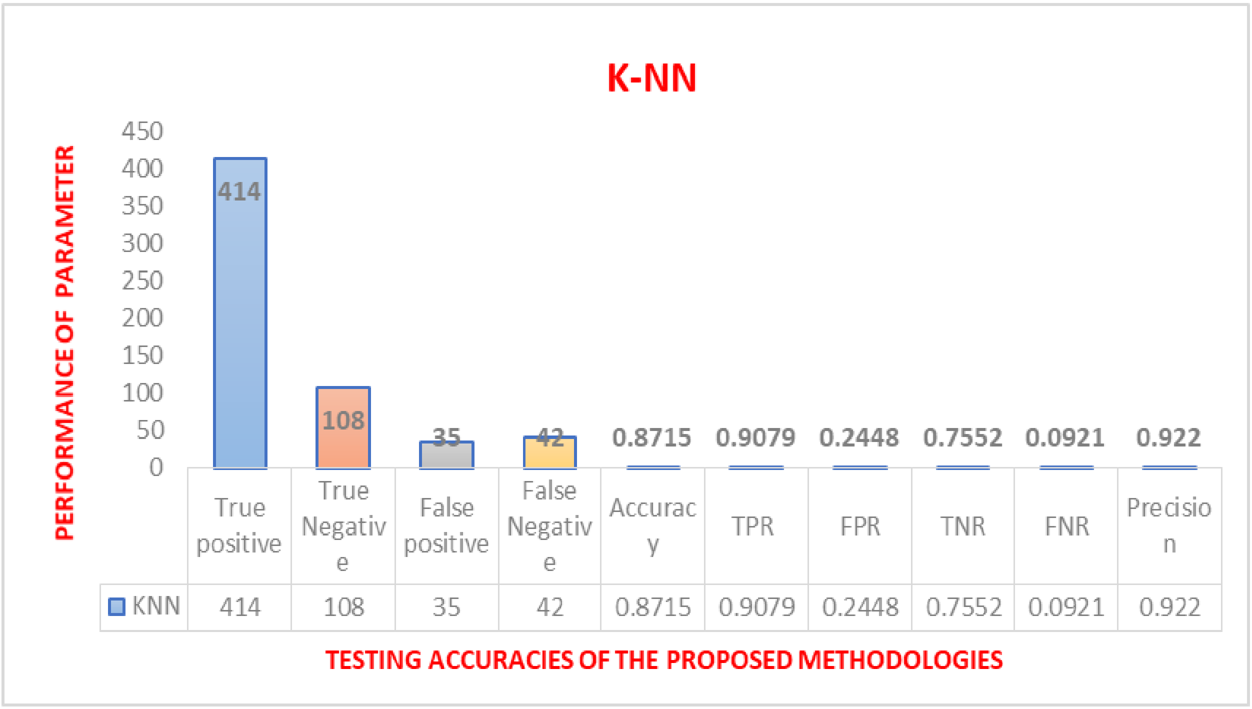


Figure (4.9): Testing accuracy using K-NN

The performance evaluations of each type distances and classification rules are chosen in function of: classification accuracy rate and time classification for each value of the nearest neighbors' parameter.

To validate the results, a number of tests have been carried out. The nearest rule was used for the classification rules to classify a new element. The high rate of classification accuracy was 88% recorded by the algorithm that utilizes Euclidean distance with a value of k = 1. Figure 2 illustrates that when K increases, the rate of classification accuracy decreases then takes a stable state at approximately 50, with almost 88% classification accuracy rate. However, the optimum result is generated with Euclidean distance (88%), this corroborates with what was presented in the literature. The smallest rate of classification achieved in attrition 50 with 73%. In contrast, Euclidean is time-consuming classification, these results are illustrated in Figure (4.12)

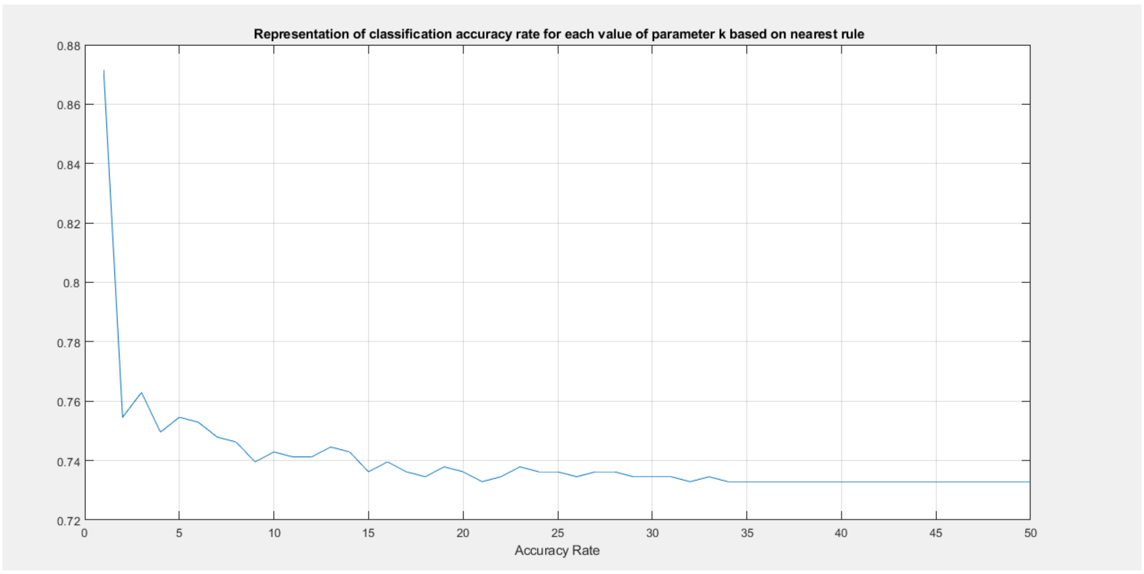


Figure (4.10): Representation of classification accuracy rate for each value of parameter k based on random rule.

4.3.3 The SVM Detection Malware in Android

In the proposed work, apply SVM to detection Malware in Android, Trained Classification SVM classifiers store prior probabilities, parameter values, support vectors, algorithmic implementation information and training data. These classifiers are used for tasks like fitting a score-to-posterior-probability transformation function.

Training Data Simulation 70% of the database was used for training. The benign class is contained 420 samples, while the malignant class included 180 samples of the training data. Table (4.3) is presented the classification results of the training set by SVM.

Table (4.3): Classification of test SVM

	Normal	Abnormal
True	414	108
False	35	42

Performances of the SVM classifiers obtained with linear functions with and without feature selection in terms of classification the evolution of SVM (True positive= 414, True Negative= 108, False positive= 35, False Negative=42, Accuracy = 0.8715, TPR= 0.9079, FPR= 0.2448, TNR= 0.7552, FNR = 0.0921, Precision =0.9220). As Shown in Figure (4.13).

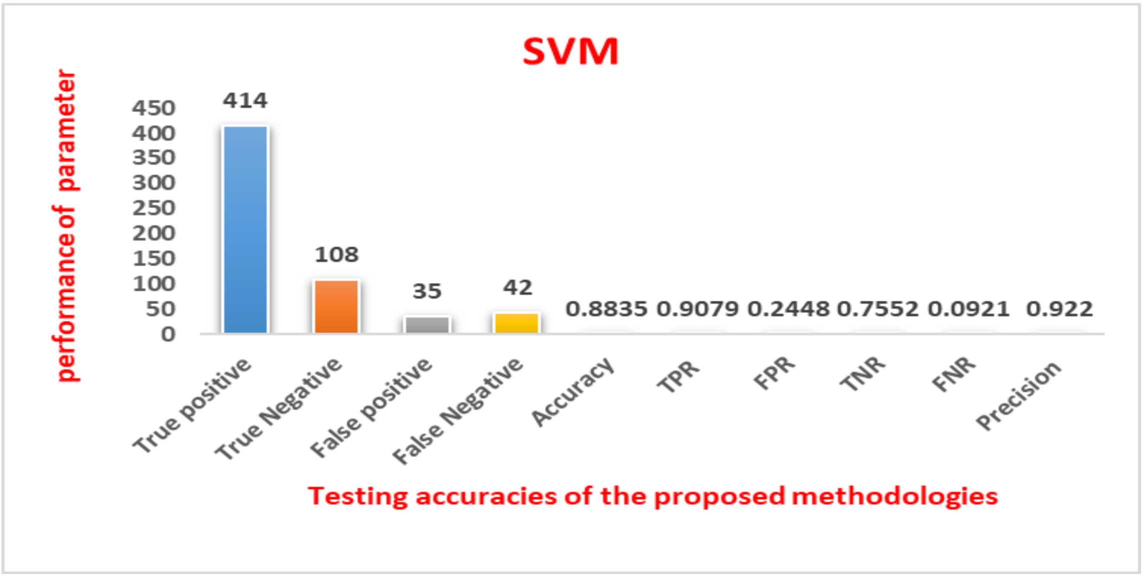


Figure (4.11): Testing accuracy using SVM

4.3.4 The hybrid Genetic algorithm and SVM Detection Malware in Android

As it is shown, a highly better performance is presented at performing feature selection with the SVM classifiers (i.e., classification accuracy, ROC, and F-measure). Were given in the Table (4.4)

Table (4.4): Classification of test hybrid Genetic algorithm and SVM

Classification of test hybrid Genetic algorithm and SVM		
	Normal	**Abnormal**
True	444	112
False	0	43

After achieving feature selection with use of the genetic algorithm, the numbers of features selected from the dataset with and without feature selection in terms of classification the evolution of hybrid Genetic algorithm and SVM (True positive=444, True Negative=112, False positive= 0, False Negative= 43., Accuracy= 0.9282, TPR= 0.9117, FPR= 0, TNR=1, FNR=0.0883, Precision=1). Show the figure (4.14).

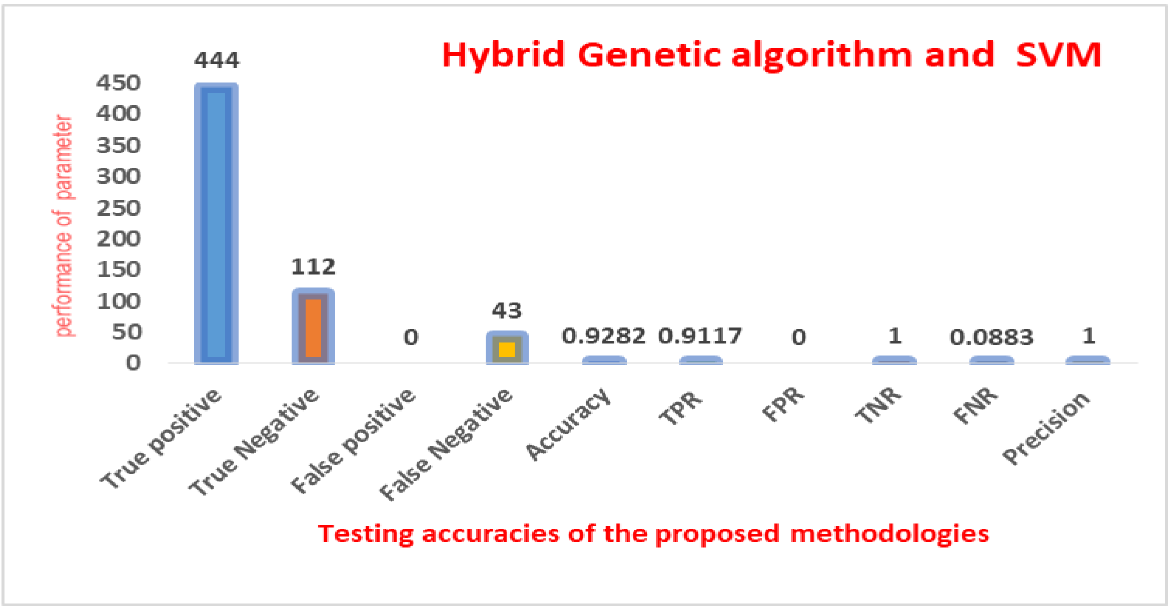


Figure (4.12): Testing accuracy using hybrid Genetic algorithm and SVM

4.4 Evaluation of Official Market Metadata

This study aims mainly to provide a suitable answer to the question: does whether Google Play market metadata plays a crucial role while detecting mobile malware and contributes to the detection model with only Android permissions? For answering this, two baseline datasets with corresponding classification algorithms are compared. Any feature selection algorithm does not accompany the classification algorithms, thus, when results are examined for them, they show that an improvement is gained in half of the prediction accuracy of models due to adding the official market metadata to the model. Addition of market metadata of accuracy (88.65%) for PNN, (87.15%) for SVM, (88.35%) for K-NN. When Google Play metadata is added on the permissions the accuracy of the model increased from 92% for hybrid Genetic algorithm and SVM. It is noted that the accuracy degrades more for SVM by reducing 87.37%. Due to the paired t-test results in the light of these values, the statically significant increase is that of prediction accuracy for the PNN classifier at 0.05 confidence value. A more informative aspect of the predictive accuracy is provided by investigating the classification algorithms with feature selection algorithms. Yet it produces the statistically more significant accuracies with the addition of market metadata when feature selection algorithms are applied before classification. Were given in the table (4.5)

Table (4.5): The Result of Performance Evaluation of machine learning technique

Statistic	PNN	KNN	SVM	Hybrid Genetic algorithm and SVM
True positive	443	414	414	444
True Negative	87	108	108	112
False positive	0	35	35	0
False Negative	69	42	42	43
Accuracy	0.8848	0.8715	0.8835	0.9282
TPR	0.8652	0.9079	0.9079	0.9117
FPR	0	0.2448	0.2448	0
TNR	1	0.7552	0.7552	1
FNR	0.1348	0.0921	0.0921	0.0883
Precision	1	0.9220	0.9220	1

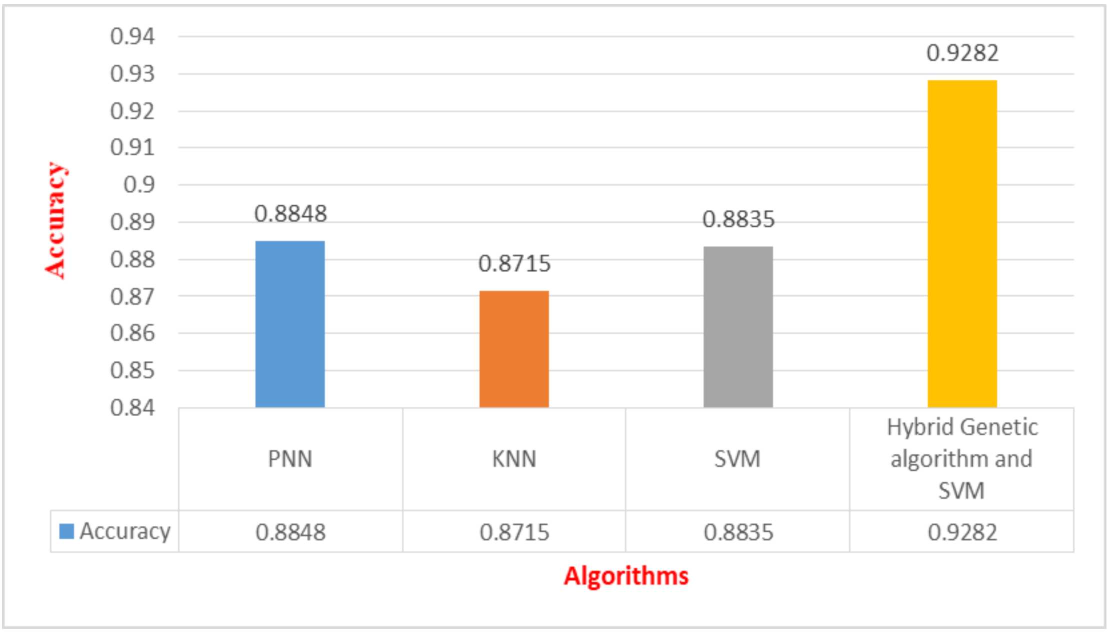


Figure (4.13): The compare of Algorithms

The proposed system result has been comparing that result with three previous related works and show that our system more accurate than the comparison works, as shown in table (4.6)

Table (4.6) Comparison with some related work

Authors	**Pref. no.**	**Algorithm for classification**	**accuracy**
Nuray Baltaci	[14]	Naïve bayes, k-NN , J48 and random forest	0.8647
Michal Kedziora	[17]	RF, SVM, K-NN, Nave Bayes, and LR	0.80
H al-kaaf1	[20]	Sequential minimal optimization (SMO), J48 and Naive Bayes	0.88
The proposed system	-	Hybrid Genetic algorithm and SVM	0.93

Chapter Five
Conclusion and Future Work

Chapter Five
Conclusion and Future Work

5.1Conclusions

In this chapter, the proposed system is summarized; the following conclusions were taken from the collection of test results. Some of those conclusions are listed in the following:

- ❖ The proposed system is a robust methodology to detect and classify malware of mobile applications with accurate and better results based on access control and anomaly detector that provide protection mechanism in the Android platform.
- ❖ This work is conducted to obtain better results using two different proposed systems. The first proposed system use access control and the second proposed system use a hybrid algorithm (GA algorithm and SVM).
- ❖ The results of the proposed system which is based on a hybrid algorithm using the GA algorithm and SVM obtained accuracy in the testing phase for the proposed system is (93%).

5.2 Future Work

The proposed system of Aroid mobile for malware detection and classification is a flexible system and there are many suggestions that can be performed for future work as follows:

1- Develop meta-heuristic model to evaluate the performance of traditional and modern detection in terms of their capability to detect attacks in a higher level of network abstraction.

2- Using various machine learning techniques two or more for the proposed system to obtain a better accuracy rate.

3- Developing a new different method for smartphone security by using another algorithm such as: (Ant Lion optimization algorithm).

References

References

[1] Chit La Pyae Myo Hein, Khin Mar Myo, “Permission-based Feature Selection for Android Malware Detection and Analysis “, International Journal of Computer Applications (0975 – 8887) Volume 181 – No. 19, September 2018.

[2] https://www.statista.com/statistics/276623/number-of-apps-available-in-leading-app Number of available applications in the Google Play Store from December 2009 to June 2019.

[3] Moser, A., Kruegel, C., and Kirda, E. “Limits of static analysis for malware detection”, in Twenty-Third Annual Computer Security Applications Conference, 2007.

[4] Devesa, J., Brezo, F., Nieves, J., and Bringas,” A Static- Dynamic Approach for Machine Learning- Based Malware Detection”, Advances in Intelligent Systems and Computing book series (AISC, volume 189), 2013.

[5] Lin, C.T., Wang, N.J., Xiao, H., and Eckert, C. “Feature Selection and Extraction for malware classification”, Inf. Sci. Eng., 2015.

[6] Zhou, Y. and Jiang, X., “Dissecting android malware: Characterization and evolution” , IEEE symposium on security and privacy, pp. 95–109, 2012.

[7] Hayati, P., Potdar, V., Talevski, A., and Chai, K. , “Characterisation of web spambots using self organising maps”,Computer Systems Science and Engineering, 2011.

[8] Mohurle, S. and Patil, M. (2017) A brief study of wannacry threat: Ransomware at- tack 2017. International Journal of Advanced Research in Computer Science, 8 (5).

[9] Kemalis, K. and Tzouramanis, T. (2008) SQL-IDS: a specification-based approach for SQL-injection detection, in Proceedings of the 2008 ACM symposium on Applied computing, pp. 2153–2158.

[10] Kateryna Chumachenko, “Machine Learning Methods for Malware Detection and Classification” , 2017 thesis J.C.S. Monteiro, "Robust Iris Recognition under Unconstrained Settings", M.Sc thesis, De Universidate Do Porto, Faculdate De Engenharia, Integrado em Bioengenharia, 2012

[11] Deeba Kannan, Kuntal Bajpayee, Samriddho Roy,”Solving Timetable Scheduling Problems Using Genetic Algorithm”, International Journal of Recent Technology and Engineering (IJRTE) ISSN: 2277-3878, Volume-7, Issue-5C, February 2019.

[12] Yu Lu, Pan Zulie, Liu Jingju, and Shen Yi. Android malware detection technology based on improved bayesian classification. In Instrumentation, Measurement, Computer, Communication and Control (IMCCC), 2013 Third International Conference on, pages 1338–1341, Sept 2013. doi: 10.1109/IMCCC.2013.297.

[13] Nuray Baltaci, “A Comparison of classification Algorithm for mobile malware detection: market metadata as input Source”, 2014.

[14] H. Kurniawan, Y. Rosmansyah, and B. Dabarsyah. Android anomaly detection system using machine learning classification. In Electrical Engineering and Informatics (ICEEI), 2015 International Conference on, pages 288–293, Aug 2015. doi: 10.1109/ICEEI.2015.7352512.

[15] Wei Wang, Meichen Zhao, Zhenzhen Gao, “Constructing Features for Detecting Android Malicious Applications: Issues, Taxonomy and Directions” , Received April 22, 2019, accepted May 15, 2019, date of publication May 22, 2019

[16] Mohsen Kakavand, Mohammed Dabnagh, Ali Dehghantanha," Application of Machine Learning Algorithm for Android Malware Detection", Proceedings of the 2018 International Conference on Computational Intelligence and Intelligent Systems, November 2018 Pages 32–36

[17] Matthew Leeds, Miclain Keffeler, Travis Atkison, "A Comparison of Features for Android Malware Detection", ACM SE '17, April 13-15, 2017, Kennesaw, GA, USA

[18] Michal Kedziora, Paulina Gawin, Michal Szczepanik and Ireneusz Jozwiak, "Android Malware Detection Using Machine Learning and Reverse Engineering", Computer Science & Information Technology (CS & IT) Conference · December 2018.

[19] H al-kaaf1, A Ali, S Shamsuddin and S Hassan , "Feature selection for malicious android applications using SymmetricalUncert AttributeEval method", Sustainable and Integrated Engineering International Conference 2019 (SIE 2019).

[20] Muttik, "A New Android Malware Detection Approach Using Bayesian Classification", IEEE 27th International Conference on Advanced Information Networking and Applications (AINA), p. 121 – 128, 2013.

[21] Shivam1, Ranjana sharma2," A Survey of Android Technology", International Conference on Advanced Computing (ICAC-2016).

[22] Kirthika.B, Prabhu.S and Visalakshi.S ,"Android Operating System :A Review", International Journal of Trend in Research and Development, Volume 2(5), ISSN 2394-9333,2015.

[23] Amah Alsoghyer and Iman Almomani , “Ransomware Detection System for Android Applications”, 5 August 2019.

[24] Chenglin Li, Keith Mills, Di Nju , “Android Malware Detection based on Factorization Machine” , arXiv:1805.11843v2 [cs.CR] 13 Aug 2019.

[25] N. Peiravian and X. Zhu. Machine Learning for Android Malware Detection Using Permission and API Calls. In Proceedings of the 2013 IEEE 25th International Conference on Tools with Artificial Intelligence (ICTAI 2013), pages 300–305, USA, November 2013.

[26] Mengyu Qiao, Andrew H. Sung, “Merging Permission and API Features for Android Malware Detection”, 2016 5th IIAI International Congress on Advanced Applied Informatics.

[27] Rubata RIASAT, “Machine Learning Approach for Malware Detection by Using APKs”, 2nd International Conference on Computer, Network Security and Communication Engineering (CNSCE 2017)ISBN: 978-1-60595-439-4.

[28] Z. Aung and W. Zaw. “Permission-based android malware detection”, International Journal of Scientific and Technology Research, 2(3):228–234, 2013.

[29] Bharati Wukkadada, Ramith Nambiar, Amala Nair, Mobile Operating System: Analysis and Comparison of Android and iOS, International Journal of Computing and Technology, Volume 2, Issue 7, July 2015 ISSN : 2348 – 6090.

[30] Hemant Rathore, Sanjay K. Sahay, Palash Chaturvedi and Mohit Sewak, “Android Malicious Application Classification Using Clustering” , arXiv:1904.10142v1 [cs.CR] 21 Apr 2019

[31] Aru Okereke Eze and Chiaghana Chukwunonso E.,” Malware Analysis and Mitigation in Information Preservation”, IOSR Journal of Computer Engineering (IOSR-JCE) e-ISSN: 2278-0661,

p-ISSN: 2278-8727, Volume 20, Issue 4, Ver. I (Jul - Aug 2018), PP 53-62.

[32] Youchao Dong, “Android Malware Prediction by Permission Analysis and Data Mining”, A thesis submitted in partial fulfillment of the requirements for the degree of Master of Science (Computer and Information Science) in The University of Michigan-Dearborn 2017

[33] Swapna Vemparala Fabio Di Troia, “Malware Detection Using Dynamic Birthmarks”, pp. 30–36, 2017.

[34] Kateryna Chumachenko, “Machine Learning Methods for Malware Detection and Classification”, thesis 2017.

[35] Zheng, M., Sun, M., and Lui, J.C. ,”Droid analytics: a signature based analytic system to collect, extract, analyze and associate android malware”, 12th IEEE International Conference on Trust, Security and Privacy in Computing and Communications, pp. 163–171, 2013.

[36] Sung, A.H., Xu, J., Chavez, P., and Mukkamala, S. , “Static analyzer of vicious executables (save)” , in 20th Annual Computer Security Applications Conference, pp. 326–334, 2013.

[37] Ruitenbeek, E.V., Courtney, T., Sanders, W.H., and Stevens, F. “Quantifying the effectiveness of mobile phone virus response mechanisms”, in 37th An- nual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN’07),pp. 790–800, 2007.

[38] Mustafa A.Ali, Wesam S.Baya, “Review on Malware and Malware Detection Using Data Mining Techniques”, November 2017 DOI: 10.29196/jub.v25i5.104.

[39] Hanif Mohaddes Deylami, Ravie Chandren Muniyandi, Iman Tabatabaei Ardekani, Abdolhossein Sarrafzadeh, “Taxonomy of

Malware Detection Techniques: A Systematic Literature Review “, Fourteenth Annual Conference 14, December, 2016 New Zealand.

[40] Najat Ali, Daniel Neagu1, Paul Trundle1, “Evaluation of k-nearest neighbour classifier performance for heterogeneous data sets”, 2019.

[41] Lakshmanaprabu, S. K., K. Shankar, M. Ilayaraja, Abdul Wahid Nasir, V. Vijayakumar, and Naveen Chilamkurti. "Random forest for big data classification in the internet of things using optimal features." International Journal of Machine Learning and Cybernetics, 2019.

[42] R. Jiang, W. Tang, X. Wu and W. Fu, “A random forest approach to the etection of epistatic interactions in case-control studies” BMC Bioinformatics, vol. 10 (Suppl 1): S65, Jan. 2009.

[43] Liu, Yanli, Yourong Wang, and Jian Zhang. "New machine learning algorithm: random forest." In International Conference on Information Computing and Applications, pp. 246-252. Springer, Berlin, Heidelberg, 2012.

[44] Jianfang Cao , Min Wang,Yanfei Li,Qi Zhang , “Improved support vector machine classification algorithm based on adaptive feature weight updating in the Hadoop cluster environment”, April 10, 2019.

[45] Shirin Tavara, “High-Performance Computing For Support Vector Machines”, 2018

[46] Xiuqiao Sun, JianWang ,WeitiaoWu and Wenjia Liu , “Genetic lgorithm for Optimizing Routing Design and Fleet Allocation of Freeway Service overlapping patrol”,2018.

[47] S. Behzadi ,M. Kolbadinejad, “Introducing a Novel Method to Solve Shortest Path Problem Based on Structure of Network using Genetic Algorithm”, The International Archives of the

Photogrammetry, Remote Sensing and Spatial Information Sciences, Volume XLII-4/W18, 2019.

[48] Ahmad B. Hassanat , V. B. Prasath, Mohammed A. Abbadi ,Salam A. Abu-Qdari and H. Faris, “An Improved Genetic Algorithm with a New Initialization Mechanism Based on Regression Techniques”, Received: 8 May 2018; ccepted: 4 July 2018; Published: 7 July 2018.

[49] Markdy Y. Orong, Ariel M. Sison, Ruji P. Medina, “A New Crossover Mechanism for Genetic Algorithm with Rank-based Selection Method”, 2018 5th International Conference on Business and Industrial Research (ICBIR), Bangkok, Thailand

[50] Ahmad Hassanat , Khalid Almohammadi , Esra’a Alkafaween , Eman Abunawas ,Awni Hammouri and V. B. Surya Prasath , “Choosing Mutation and Crossover Ratios for Genetic Algorithms—A Review with a New Dynamic Approach” , Received: 6 November 2019; Accepted: 2 December 2019; Published: 10 December 2019.

[51] Vasana Ch., Chandima T., and Musa M.,” An Improved Probabilistic Neural Network Model for Directional Prediction of a Stock Market Index”, Published: 6 December 2019

[52] Nuray Baltaci, “A Comparison of classification Algorithm for mobile malware detection: market metadata as input Source”, 2014.

الخلاصـــــــــــة

أصبحت الهواتف الذكية ضرورية في حياتنا اليومية. إذ يمكن بوساطتها القيام بالعديد من الاعمال، كما يمكن القيام بالتصفح على الانترنيت وتحميل العديد من التطبيقات لكل جهاز، من خلال المتاجر المتاحة. ونتيجة لذلك، يزداد ايضا عدد تطبيقات البرامج الضارة التي يتم تحميلها. إذ تقوم هذا البرامج الضارة بالأنشطة مختلفة خلف الكواليس؛ مثل خروقات السرية والخصوصية، خسارة السرية، تعطل النظام التشغيل، سرقة معلومات حساسة...ألـخ .

هناك الكثير من البحوث والدراسات التي استخدمت تقنيات مختلفة لكشف عن البرامـــــج الضارة ولكنها لاتخلو من نقاط الضعف التي تمثلت بالكفاءة والسرعة وكذلك عدم الشموليـــــة. في هذه الرسالة ، تم تصميم وتنفيذ نظام مقترح للكشف عن البرامج الضارة في الهواتف الذكية ويحتوي على جزئين:

في الجزء الأول ، يبدأ التحكم في الوصول مبدئيًا عند بدء تشغيل النظام. تتبنى خوارزمية مصادقة المستخدم إذن المستخدم لاكتشاف عامل التهديد بعد تطبيق سياسة أذونات المستخدم عن طريق تحسين الطريقة التي يتم بها استخراج أنشطة المستخدم. منذ تحديد التحكم في الوصول الملف على أنه غير طبيعي.

في الجزء الثاني ، تبدأ تقنية اكتشاف العيوب في استخراج الميزات المهمة التي تلعب دورًا فعالًا في اكتشاف الرموز الضارة وتطبيق خوارزميات التعلم الآلي.

النظام المقترح قد تم تجربته عن طريق استخدام خوارزمية جينية هجينة, ومجموعة بيانات الSVM وتم تسجيل بدقة (0.9282).اظهرت النتائج انه لدى النظام المقترح متوسط دقة عالية مقارنةً بالطرق الأخرى الموجودة التي تعطي متوسط دقة (0.8848) باستخدام PNN, ومع SVM فيكون متوسط الدقة (0.8835), و يكون متوسط الدقة (0.8715) مع استخدام K-NN.

وزارة التعليم العالي والبحث العلمي
جامعة ديالى
كلية العلوم

الكشف عن البرامج الضارة في تطبيق الاندرويد الجوال

رسالة
مقدمة الى قسم علوم الحاسوب / كلية العلوم / جامعة ديالى وهي جزء
من متطلبات نيل درجة الماجستير في علوم الحاسوب

من قبل
سجى ابراهيم هاني اسماعيل

بأشراف
أ. ناجي مطر سحيب

١٤٤١هـ ٢٠٢٠م