SMARTPHONE MALWARE DETECTION USING PERMISSIONS AND MCNEMAR TEST

A DISSERTATION
SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENTS
FOR THE AWARD OF DEGREE
OF
MASTER OF SCIENCE(M.Sc.)
IN
MATHEMATICS

Submitted By:
GARIMA KUMARI
$(2k21/MSCMAT/17)$

Under the supervision of
DR. ANSHUL ARORA

DEPARTMENT OF APPLIED MATHEMATICS
DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
Bawana Road, Delhi-110042
MAY, 2023

DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College of Engineering)

Bawana Road, Delhi-110042

CANDIDATE'S DECLARATION

I, Garima Kumari, Roll No.s 2K21/MSCMAT/17 student of Master in Science (Mathematics), declaring that the project's dissertation titled SMARTPHONE MALWARE DETECTION USING PERMISSIONS AND MCNEMAR TEST is original and not copied from any source without proper citation and is presented by me to the Department of Applied Mathematics, Delhi Technological University, Delhi, in partial fulfilment of the requirement for the award of the degree of Master of Science in Mathematics, is original and not copied from any source without proper citation. This work has not previously formed the basis for the award of any Degree, Diploma, Associateship, Fellowship or other similar title or recognition.

Place: Delhi

Date: May 25, 2023

Garima Kumari

2K21/MSCMAT/17

DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College of Engineering)

Bawana Road, Delhi-110042

CERTIFICATE

I hereby attest that the project dissertation SMARTPHONE MALWARE DETECTION USING PERMISSIONS AND MCNEMAR TEST submitted by Garima Kumari, Roll No. $2K21/MSCMAT/17$ and of Department of Applied Mathematics, Delhi Technological University, Delhi in partial fulfillment of the requirement for the award of the degree of Masters of Science in Mathematics, is a record of the project work carried out by the students under my supervision. To the best of my knowledge, this work has not been submitted in part or full for any Degree or Diploma to this University or elsewhere.

Place: Delhi

Date: May 25, 2023

Dr. Anshul Arora

Supervisor

ACKNOWLEDGEMENT

My supervisor, **Dr. Anshul Arora** of the Department of Applied Mathematics at **Delhi Technological University**, has my sincere gratitude for his meticulous guidance, profound expertise, constructive criticism, attentive listening, and amiable demeanor have been invaluable throughout the process of composing this report. I am eternally grateful for his benevolent and supportive approach, as well as his perceptive counsel, which played a pivotal role in the successful culmination of my project. Furthermore, I would like to express my appreciation to all my classmates who have played a pivotal role in aiding me to complete this endeavor by offering assistance and facilitating the exchange of pertinent information.

Garima Kumari
2K21/MSCMAT/17

ABSTRACT

A recent report has shown that the availability of smartphones is increasing at an alarming rate and hence the number of mobile malware is exponentially increasing with the increase in popularity of smartphones. From young children to senior citizens everybody uses smartphones as a daily necessity. In 2022, 142.6 billion games and apps were downloaded, which works out to 1.6 million apps downloaded every hour.In 2022, iOS had 32.6 billion downloads, while Google Play had 110.1 billion, which is more than thrice the downloads of iOS. Looking at the level of threat from malware applications for Android users, it becomes essential to detect malware applications in a quick and effective way. One such way is to use permissions. To make an effective system for malware detection using permissions we need a large dataset and different permissions to analyze the pattern. We can use different machine learning techniques to find the pattern using dataset and permissions, but if we increase the number of permissions and dataset. With a large number of permissions for analysis, the time of computation increases drastically. The time of computation can be reduced if we reduce the number of datasets or the number of permissions. Reducing the number of features is preferred over decreasing the number of datasets. We can reduce the number of permissions if we only choose the permissions that are most distinguishing and ignore the permissions that don't play a huge role in distinguishing between malware and benign applications. If a permission only exists in malware applications and not in benign applications then such permissions are recognised as distinguishing, but as the malware applications are getting more incognito, the overlapping of the permissions used by both malware and benign application is increasing. Thus we require a method to rank the permissions based on how well that permission can be used to detect the nature of the application. In this thesis, we introduce a statistical technique named McNemar test to find the correlation of a set of permissions with malware and benign applications and rank the permissions. The correlation gives a numerical value for the overlapping of each permission in malware and benign applications. The greater the correlation value lesser will be its usefulness in distinguishing the nature of the application. Such ranking helps us eliminate irrelevant permissions. This ranking can be further used for detection using various machine-learning algorithms. As a result, we narrowed down the total set of permissions from 129 to 38 and got 97% detection accuracy with the Random Forest classifier.

Contents

Candidate's Declaration **ii**

Certificate **iii**

Acknowledgement **iv**

Abstract **v**

1 Introduction **1**

1.1 Role of smartphones in today's world 1

1.2 Motivation 3

1.3 Thesis Structure 4

2 Smartphone Malware Detection using Permissions and McNemar test **5**

2.1 What is Malware? 5

2.2 What do we understand by permissions in smartphones ? 6

2.3 What is the McNemar Test ? 7

2.3.1 Categorical Variables 7

2.3.2 Contingency Table 8

2.3.3 McNemar Test 9

3 Related work **11**

3.0.1 Literature Analysis 11

3.0.2 Merits and Demerits of Existing work 13

4 Methology **14**

4.0.1 Datasets 14

4.0.2 Permissions Extraction 15

4.0.3 Permission Ranking 15

4.0.4 Detection with Machine Learning techniques 18

5 Results **20**

6 Conclusion **24**
6.0.1 Thesis Summary 24
6.0.2 Contribution 24
6.0.3 Future Work 25

References **26**

Details of Candidate's Publication **30**

List of Tables

2.1 Contingency Table 8

4.1 McNemar Table 16
4.2 Observation of feature-I 17
4.3 McNemar Table for Feauture-I 17

5.1 McNemar statistic for Permissions 21
5.2 Ranked Permissions 22
5.3 Detection Accuracies 22
5.4 Detection Accuracies 23

List of Figures

1.1 Smartphone subscription data 1

4.1 Chi-Square value 18

Chapter 1: Introduction

For more than ten years now, smartphones have been a crucial part of our lives, and their acceptance only appears to grow with time. In 2023, it is expected that the use of smartphones will continue to rise, with more people relying on these devices for communication, entertainment, and productivity. By 2028, it is anticipated that there will be more than 7.8 billion smartphone mobile network subscriptions worldwide, up from almost 6.6 billion in 2022 according to Statista report 2023, as summarized in Figure 1 [1].

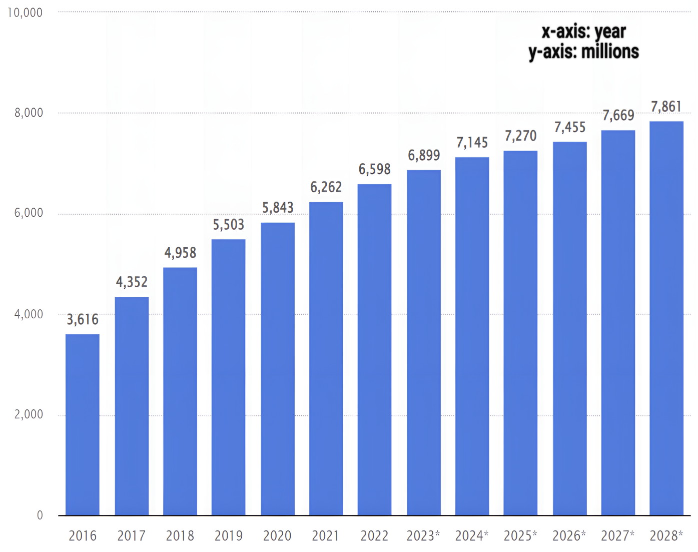


Figure 1.1: Smartphone subscription data

1.1 Role of smartphones in today's world

The adaptability of smartphones is one of the main factors contributing to their appeal. They can be used as a camera, music player, gaming console, portable computer, and more, in addition to being phone and messaging devices. In fact, for many people, their smartphone is the primary way they access the internet, check emails, and stay connected

[1]https://gs.statcounter.com/os-market-share/mobile/worldwide

with the world. Furthermore, smartphones are now more accessible and inexpensive than ever before. More consumers are now able to buy these gadgets due to the expansion of budget-friendly smartphone models and the availability of affordable data plans. This has led to increased smartphone adoption, particularly in emerging markets. Another factor that has contributed to the growing popularity of smartphones is the constant evolution of technology. Manufacturers constantly release new models with improved features, such as better cameras, faster processors, and longer battery life. This has led to a culture of upgrading and replacing phones frequently, as people seek to stay up-to-date with the latest technology. In conclusion, the popularity of smartphones in 2023 is expected to continue to grow, driven by their versatility, affordability, and constant technological advancements. These devices have become an essential part of modern life, and their importance is only set to increase in the years to come.

The portability of smartphones is one of the primary reasons for the shift towards them from personal computers (PCs). Smartphones are light and small enough to easily slip into a purse or pocket, making them ideal for individuals who are consistently on the go and need to remain connected with their work or personal life. The adaptability of smartphones is yet another factor driving the shift towards them. They are able to use applications, send emails, and access the internet. Smartphone users can access a number of well-known productivity applications, including Microsoft Office. This means that users won't have to sit at a desk to complete many of the same tasks on their smartphone as they would on a PC. Moreover, cell phones have become more powerful over the years, and many models now feature advanced processors and a lot of memory. As a result, more complex smartphone applications like video editing software and gaming applications can now be developed. Many people are now using their smartphones to do things that were only possible on a PC in the past.

The trend toward cloud computing is another factor driving the move toward smartphones. Users of cloud computing can access their applications and data from any location, regardless of their device. This implies that clients can begin an errand on their cell phone and finish it on their PC, or the other way around. Smartphones are a popular choice for users who value convenience and adaptability due to their adaptability.

When we compare the mobile operating systems in terms of their global usage strength, Android emerges as the unquestionable leader. According to the most recent stats from Statcounter, Android controls 70.93 percent of the global mobile operating system market, while iOS controls 28.37 percent. Their combined share is greater than 99 percent of the market. The remaining mobile operating systems, such as KaiOS and Samsung, together account for less than 1% of the market share. This further demonstrates that iOS and Android are the only mobile operating systems that can really be beaten.

1.2 Motivation

Android is the most popular smartphone operating system in the world, and as a result, it is also the biggest target of smartphone malware developers. Malware refers to malicious software designed to infiltrate or damage a computer system, and it can take many forms on a smartphone, such as viruses, spyware, or ransomware.

There are several reasons why Android is particularly susceptible to malware attacks:

1. Open-source nature: While Android's open-source nature allows for greater customization, it also makes the operating system more vulnerable to malware attacks. Hackers can easily access the source code of the operating system, identify vulnerabilities, and exploit them.

2. Fragmentation: Android is used by many different smartphone manufacturers, which means that there are many different versions of the operating system in use at any given time. This fragmentation can make it more difficult for developers to patch vulnerabilities and issue security updates, leaving older versions of the operating system vulnerable to attack.

3. Third-party app stores: Android allows users to install apps from third-party app stores, in addition to the official Google Play Store. While this provides users with more options, it also increases the risk of downloading malware-infected apps.

Android malware apps can pose a variety of threats to smartphones and the personal data stored on them.
Some of the most common threats posed by Android malware apps include Ransome, Ad fraud, Botnets,Data theft etc. To protect against these threats, Android users should be cautious when downloading apps and should only install apps from trusted sources. They should also keep their operating system and apps up to date with the latest security patches, and use antivirus software to scan for and remove any malware that may be present on their device. Additionally, users should practice good password hygiene and avoid clicking on suspicious links or downloading attachments from unknown sources. By taking these precautions, users can help protect themselves and their data from the many threats posed by Android malware apps. Finally, user behavior also plays a role in the susceptibility of Android to malware attacks. Many users are not careful when downloading apps or browsing the web, which can result in the unintentional downloading of malicious software. To mitigate the risk of malware attacks on Android devices, users should take several precautions. These include only downloading apps from trusted sources, keeping the operating system and apps up to date, using antivirus software, and being cautious when clicking on links or downloading attachments. By following these best practices, users can help protect their Android devices from malware attacks. As a

result, it is critical to conduct research into effective application store malware detection techniques. However, the most common issues with permission feature-based detection methods are as follows:

1. Different kinds of applications have different rules about whether or not to ask for the same permission. For instance, mentioning consent to peruse contacts is legitimate to conduct in social media chatting applications, yet it is typically viewed as malevolent conduct in photograph-taking applications. Generally speaking, including all kinds of Android applications in the same data set might lead to wrong conclusions.

2. Numerous malicious actions require multiple permission combinations to be called. For instance, the pernicious way of behaving of transferring contact data to a site requires calling READ_CONTACTS and Web consents. As a result, the accuracy of malware detection will suffer if information about permission invocation is analyzed without taking into account the effect of a particular permission combination.

3. Certain permission features are unable to effectively differentiate between legitimate and malicious applications. Taking permission features into account will result in a lot of invalid features, which will make detection less accurate and take up more time.

With due consideration of these limitations, our endeavor is to devise a methodology for the detection of malware applications by leveraging permissions.

1.3 Thesis Structure

The structure of this thesis report is systemized into 6 chapters.
Chapter 2 provides a concise overview of the fundamental prerequisites necessary for comprehending the terminologies discussed in the thesis.
Chapter 3 presents related work that has been done for malware perceived in Android OS using permissions.
Chapter 4 presents the proposed ranking methods of this research which include structure, extracting permissions, selecting permissions using ranking techniques, and detection process.
Chapter 5 shows the outcomes, the significant findings, and the performance of the classifiers.
Chapter 6 Concluded the thesis and discusses future work.
References.

Chapter 2: Smartphone Malware Detection using Permissions and McNemar test

Before proceeding further, it is imperative to establish a foundational understanding of the fundamental concepts essential for comprehending the ensuing work. These concepts include:

1. What is Malware ?
2. What do we understand by permissions in smartphones ?
3. What is the McNemar Test ?

2.1 What is Malware?

Malware, a contraction of ”malicious software,” refers to any software or code designed with malicious intent. In the realm of smartphones, malware represents a significant threat to user privacy, data security, and the overall integrity of mobile ecosystems. Android smartphones, being one of the most popular platforms globally, are particularly vulnerable to malware attacks due to their open nature and vast user base.
Types of Mobile Malware: Mobile malware encompasses various forms, each with distinct characteristics and attack vectors. These include:

- Viruses: Malicious code that replicates itself by attaching to legitimate applications and spreading through file sharing or malicious downloads.
- Trojans: Deceptive applications that appear harmless but carry malicious payloads. Trojans often masquerade as legitimate apps to trick users into installing them.
- Ransomware: Malware that encrypts user data, rendering it inaccessible, and demands a ransom for its release.

- Spyware: Malicious software designed to covertly monitor and collect sensitive user information, such as passwords, browsing habits, or personal data.
- Adware: Malware that displays unwanted advertisements, often leading to intrusive and disruptive user experiences.
- Botnets: Networks of infected devices controlled by a remote attacker, typically used for activities like distributed denial-of-service (DDoS) attacks or spam distribution.

The question arises, **how can malicious software get into our systems?**
Malware can infiltrate Android smartphones through various vectors, including: Malicious Apps, Drive-by Downloads, Phishing Attacks, Malvertising etc.
The landscape of smartphone malware is continuously evolving as attackers employ advanced techniques and exploit emerging vulnerabilities. Malware authors adapt their strategies to bypass security measures, utilize encryption to obfuscate their activities, and employ polymorphic or metamorphic techniques to evade detection.

2.2 What do we understand by permissions in smartphones ?

Permissions in the context of smartphones play a crucial role in maintaining user privacy and security. When users install an app on their smartphones, it often requests permissions to access specific resources and functionalities of the device. These permissions act as a safeguard, ensuring that apps have limited access to sensitive data and device capabilities. The Android operating system, for instance, uses a permission model that requires users to grant or deny permissions during the app installation process or when the app attempts to access certain features for the first time. This system provides users with control over which permissions they want to grant to each app. Users can review the permissions requested by an app before installing it, and they have the flexibility to grant or revoke permissions at any time through the device settings.
By granting permissions to apps, users allow them to interact with different aspects of their device. Some common types of permissions include:

- Device Hardware: Permissions like camera, microphone, and sensors allow apps to access specific hardware functionalities. For example, a photo editing app needs camera permission to capture photos, while a fitness app might request access to the device's motion sensors.
- Personal Data: Permissions related to personal data, such as contacts, calendar, and call logs, grant apps access to user information. This enables features like syncing

contacts, scheduling events, or providing caller ID services. It's important to review the requested permissions to ensure that apps have a legitimate need for accessing such data.

- Location: Location permissions enable apps to determine the device's geographic location using GPS, Wi-Fi, or cellular network data. This functionality is utilized by various apps for services like navigation, weather updates, or location-based recommendations.

- Network and Connectivity: Permissions such as internet access or Bluetooth enable apps to connect to networks or other devices. These permissions are necessary for apps that require internet connectivity, data synchronization, or communication with other devices.

Permissions are designed to strike a balance between granting apps the necessary access to provide their intended functionality while protecting user privacy and security. It's important for users to exercise caution and review the permissions requested by apps, particularly for apps from unfamiliar or untrusted sources. Additionally, regularly reviewing and managing app permissions on your device can help maintain control over the data and capabilities accessible to each app. App stores and operating system providers continuously work to improve the security of their platforms, implementing measures to detect and prevent malicious apps that might misuse permissions. Keeping your device's operating system and apps up to date with the latest security patches can also help mitigate potential risks associated with app permissions.

Following the same cause, we worked to make detection more efficient and quicker.

2.3 What is the McNemar Test ?

Before jumping to McNemar Test, we'll go through some basics of statistics for better understanding of McNemar test.

2.3.1 Categorical Variables

Categorical variables are a type of variable in statistics that represent distinct categories or groups. These variables have values that fall into specific categories or levels, and they do not have a natural numerical order or magnitude. Categorical variables are often used to classify or group observations based on certain characteristics or attributes. They provide a way to organize data into meaningful categories and can be used for various purposes, such as comparing groups, identifying patterns, or analyzing associations between vari-

ables.
There are two main types of categorical variables:

1. Nominal Variables: Nominal variables are categorical variables that represent unordered categories or groups. The categories in a nominal variable have no inherent order or ranking. For example, a nominal variable could be the color of a car, with categories such as red, blue, green, etc. Each color category is distinct, and there is no inherent order among them.

2. Ordinal Variables: Ordinal variables are categorical variables that represent ordered categories or groups. The categories in an ordinal variable have a meaningful order or ranking. For example, an ordinal variable could be the satisfaction level of customers, with categories such as ”very satisfied,” ”satisfied,” ”neutral,” ”dissatisfied,” and ”very dissatisfied.” In this case, the categories have a clear order based on the level of satisfaction.

We'll be using Nominal variable in our work in further chapters.

2.3.2 Contingency Table

In the context of the McNemar test, the contingency table is a specific type of table used to analyze the association or change between two dependent categorical variables. The McNemar test is typically applied when comparing the proportions or frequencies of the two categories within a paired or matched data set.The contingency table used in the McNemar test is a 2x2 table that organizes the frequencies or proportions of observations falling into the four possible combinations of the two categories being compared.
Here's a breakdown of the components of the contingency table for the McNemar test:

Table 2.1: Contingency Table

	variable 2 :B	**variable 2 : NOT B**
variable 1 :A	cell(1,1)	Cell(1,2)
variable 1 :NOT A	cell(2,1)	cell(2,2)

1. Row and Column Categories:

   - Row 1, Column 1: Observations with both variables in category A (e.g., ”Success” or ”Positive Outcome” or ”Pre-test success”).
   - Row 1, Column 2: Observations with the first variable in category A and the second variable in category B (e.g., ”Success-Failure” or ”Positive Outcome-Negative Outcome”).

- Row 2, Column 1: Observations with the first variable in category B and the second variable in category A (e.g., ”Failure-Success” or ”Negative Outcome-Positive Outcome”).
- Row 2, Column 2: Observations with both variables in category B (e.g., ”Failure” or ”Negative Outcome” or ”Pre-test failure”).

2. Cell Values:

   - Cell (1,1): Frequency or proportion of observations falling into both category A for both variables.
   - Cell (1,2): Frequency or proportion of observations with category A for the first variable and category B for the second variable.
   - Cell (2,1): Frequency or proportion of observations with category B for the first variable and category A for the second variable.
   - Cell (2,2): Frequency or proportion of observations falling into both category B for both variables.

3. Marginal Totals:

   - Row Marginal Total: The sum of frequencies or proportions in each row, representing the total number of observations for each category of the first variable.
   - Column Marginal Total: The sum of frequencies or proportions in each column, representing the total number of observations for each category of the second variable.
   - Overall Total: The total number of observations across all categories.

2.3.3 McNemar Test

The McNemar test is a statistical procedure used to compare the proportions of two related groups or treatments when the outcome variable is dichotomous (having two possible outcomes). It is commonly employed in situations where the data is paired or matched between the groups. The test examines whether there is a significant difference in the proportions of a specific outcome between the two treatments being compared. It is often used when comparing a new treatment or intervention to a standard or existing treatment.
To conduct the McNemar test, you gather data on the outcome variable from both groups and create a 2x2 contingency table. This table represents the counts of the four possible combinations of outcomes for the two treatments.
By calculating a test statistic using the counts from the contingency table, you can assess the significance of the difference between the treatments. The McNemar test statistic

follows a chi-square distribution, and the resulting p-value indicates the likelihood of obtaining the observed data or more extreme data if there is no actual difference between the treatments. If the p-value is below a chosen significance level (e.g., 0.05), it suggests that there is a significant difference in the proportions of the outcome between the groups. Conversely, if the p-value exceeds the significance level, there is insufficient evidence to conclude a significant difference.

McNemar Test is explained in more detail with examples in further chapters. The use of the McNemar Test for malware detection is also explained in further chapters.

Chapter 3: Related work

3.0.1 Literature Analysis

In this chapter, we review the existing works in the field of Android malware detection. Various works exist in the literature related to anomaly or intrusion detection for desktop systems such as [26]- [28]. However, we aim to build an Android malware detector, hence, we have discussed works related to Android malware detection. Some of the works such as [29]- [32] have used dynamic features for malware detection such as network traffic. However, we have built a static malware detector, hence, we focus on techniques that employ permissions for Android malware detection.

The authors in [1] proposed a model "RPNDroid" in order to develop a hybrid Android malware detector using ranked permissions and network traffic features. The authors used frequency to rank the permissions and added a threshold to eliminate redundant features from the dataset. They then merged the ranked permissions with the network traffic features to form a hybrid vector resulting in 95.96 % accuracy in detection. The authors in [2] discussed the detection using permissions and packages. The authors introduced the use of packages of Android applications and concluded that the information provided by the package is useful for detection. Li et al. [3] described static feature selection-based Android malware detection. The authors took principal component analysis approach for feature selection and found 96.05% accuracy using random forest classifier. The authors in [4] proposed a two-layered Android malware detection. The first layer used an improved random forest algorithm for analysis and the second layer of detection used sensitive permission rules matching to analyze the fuzzy sets generated by the first layer of detection. The authors in [5] developed a detection system based on multilayer perceptron for the detection of Android malware. Dataset from Drebin and Google Play Store has been selected.

The authors of [6] have concentrated on detecting malware based on permissions in Android using a deep neural network model. The result was found to be of more than 85% accuracy. The authors in [7] proposed a malware detection system using Bayesian probability on permissions to battle the malware issue. They used chi-square as an algorithm and Naïve Bayes as a classifier. The authors in [8] have introduced three levels of pruning by mining the permission data to get the most significant permissions in distinguishing between benign and malicious apps. The authors in [9] made a model using five different machine-learning methods combined with features picked from the retrieval of Android permissions in order to categorize applications as malicious or benign. They found that the tensor flow decision forest performed the best giving 90% accuracy. The authors of [10] used *androguard* tool for permission extraction, service, receiver, and intent, and merged them into a text report. The authors then applied the BiLSTM network to extract essential information from the text.

The authors of [11] used CICInvesAndMal2019 as a dataset and used Android permissions and intent as the feature set. Principal Component Analysis was used for the feature selection approach. The authors of [12] proposed an ensemble learning-based framework in order to detect malware using risky permissions as the features to train the classifier. For evaluation, they used a series of real-world Android app datasets of both malicious and benign apps. The experiments clearly showed that the proposed malware detection approach was effective with high accuracy. The authors of [13] adopted the LibSVM to classify the unknown apps and presented an SVM-based mechanism for detecting malware and normal apps.

The authors of [14] generated two types of feature vectors. One as common and the other as the combined feature vector. The authors attained 97.25% accuracy for the common whereas 96.56% accuracy for the combined features by logistic regression. Then to minimize the training and testing time they optimized the features to 131 by eliminating low variance features resulting in 95.87% accuracy. The authors of [15] used both permissions as well as hardware features for multimodal input scenarios and deep network shapes. They demonstrated that the blend of both sets of data could improve total performance, accomplishing an accuracy of 94.5%. The authors of [16] used a hybrid analysis model for the combination of permission from the static analysis method and API from the dynamic analysis method. The accuracy rate was found to be 88% whereas TP (true positive) rate was 89%. The authors of [17] used feature reduction for identifying the most influential permissions using gain ratio. They also used Multilayer Perceptron, Sequential Minimal Optimization (SMO), J48, Random Committee and Randomizable filtered classifiers for the evaluation of the selected features. The authors of [18] created an ensemble model that takes permission combinations in order to distinguish malicious and benign apps. They also revealed that the combination of classifiers in an ensemble model gave better accuracy than an individual classifier.

The authors in [19] introduced an innovative weighting method i.e.TF-IDFCF that works on the class frequency (CF) of the feature. They got a detection rate of 95.3% with a low false positive rate on testing with different classifiers. The authors in [20] proposed a permission weight approach to assign each of the permissions a different score. They then applied K-nearest Neighbor (KNN) and Naïve Bayes (NB) algorithms and used the proposed method to compare with their previous studies. The authors of [22] created three layers of pruning by mining the permission data to identify the most important permissions that are later used to differentiate the benign from malicious apps. They then used SigPID techniques to categorize families of malicious software and benign applications. The authors of [23] proposed a framework for Android malware detection based on different permission patterns, they also developed an ensemble classifier called Enclamald to determine if an application may be harmful. They have in the end claimed that the Enclamald classifier outperforms widely-used classifiers, according to tests on real-world applications. The authors in [33] analyzed permission pair graphs for Android malware detection whereas the authors in [34] combined intents with permissions for malware detection.

3.0.2 Merits and Demerits of Existing work

The existing work ranks the permissions based on various concepts that are effective but still complicated and time-consuming whereas we can use the simple concept of statistics and correlation and get an accuracy of 97% which is more than the accuracy achieved by most of the techniques used. The authors of [14] used the concept of eliminating permissions but still the last report uses 131 permission and accuracy has dropped to 95%.To the best of our knowledge, no other existing work has applied the Mcnemar test to rank the permissions in order to detect Android malware.

Chapter 4: Methology

Now, we explain the proposed methodology in detail, which is described in sections below.

4.0.1 Datasets

We have collected 2 datasets. One consists of data for normal Android apps and the other for malicious Android apps. For normal apps, we have used the Google play store whereas for malicious data we have used the AndroZoo website[1].
In March 2023, we gathered information on apps that were offered in the Google Play Store. 2,673,292 apps were available in the Google Play Store at the time of data collection. It is significant to note that this study does not examine consumer access to apps across all platforms; rather, it solely examines apps in the Google Play Store. The Google Play Store was chosen for analysis not because it is an accurate representation of all available apps for all device kinds, but rather because of the store's popularity and the data's relatively open access.
AndroZoo is a growing library of Android apps gathered from multiple sources, such as the official Google Play app market, as well as a growing collection of various app-related metadata, with the goal of helping research projects related to Android. The most important step is to extract permission data from an APK file. That includes downloading the APK file, extracting it using a tool such as APK Extractor or by renaming the APK file to have a .zip extension or extracting it using a file archiver like WinZip or 7-Zip, locating the AndroidManifest.xml file which is mostly located in the "META-INF" folder then searching for the "uses-permission" tag in the AndroidManifest.xml file. Now copy the permission names and descriptions from the AndroidManifest.xml file into a spreadsheet or database. We can repeat these steps for each app to extract permission and form a dataset. We have taken the dataset of 1,11,010 applications, out of which 55,505 are labeled malicious taken from the AndroZoo website and the rest 55505 are labeled normal that were extracted from the Google play store. Then both of the datasets were combined and as a whole, there were 129 permissions. We marked the existence of the permission in the concerned app as 1 and the absence was marked as 0.

[1]https://androzoo.uni.lu/

In the dataset formed we have 1,11,010 applications (55,505 malicious and 55,505 benign) and 129 features that are marked 1 or 0 based on the existence of that permission in that application.

4.0.2 Permissions Extraction

In this work, we have used permissions as a feature for malware detection. The crucial process of extracting and analyzing permissions from Android apps is known as Android Permission Extraction, and it is used to identify potential malware. For extracting permissions, static analysis and dynamic analysis are the two most popular techniques. In order to extract the manifest file, which provides details about the app's rights, static analysis entails decompiling the app's APK file using programs like Apktool, JADX, or Androguard. The permission declarations are then extracted from the manifest file using XML parsing libraries. By comparing these extracted permissions to a predefined list of known dangerous permissions or looking for anomalous or excessive permissions that an app seeks beyond the scope of its authorized functionality, these extracted permissions can be further examined. Dynamic analysis, on the other hand, is running the app on a device or an emulator and observing its behavior during runtime using apps like DroidBox, TaintDroid, or MobSF. The uses-permission¿ tag in the manifest file or runtime permission requests made using the Android Runtime Permissions system are both ways that the dynamic analysis tools record the permissions that the app seeks during runtime.

4.0.3 Permission Ranking

Most of the permissions in normal and malware datasets are similar. Hence, we aim to rank the permissions so that we can identify which permissions are distinguishing and which ones are irrelevant so that the irrelevant permissions features can be eliminated from our analysis. For ranking purposes, we have applied the Mcnemar test.

McNemar's test was first published as an article in a Psychometrika in 1947. Quinn McNemar, a professor in the Psychology and Statistics department at Stanford University created it. This non-parametric (distribution-free) test determines whether there has been a statistically significant shift in proportions on a dichotomous trait between two-time points in the same population. The dichotomous variable is applied using a 2x2 contingency table at times 1 and 2. In medical research, a count of the subjects is recorded (as + and – signs, or 0 and 1) in a table before and after being administered the medicine in order to ascertain whether or not a specific drug has an effect on a disease (e.g., yes vs. no). The Chi-Square test statistic is then used to use McNemar's test to statistically determine whether or not a medicine affects the condition.

Let's denote by $\mathbf{p_a}, \mathbf{p_b}, \mathbf{p_c}, \mathbf{p_d}$ the theoretical probability of their respective group with $p_a + p_b + p_c + p_d = 1$

Table 4.1: McNemar Table

	Test-1 positive	**Test-1 negative**	**Row Total**
Test-2 Positive	a	b	a+b
Test-2 Negative	c	d	c+d
Column sun	a+c	b+d	**N**

* **N** is the total number of elements for experiment

Now, the **null hypothesis** of McNemar's test is a claim about the marginal distributions. It states that the row marginal and column marginal are equal.

$$H_0: p_a + p_b = p_a + p_c$$

Whereas **Alternative hypothesis** is :

$$H_1: p_a + p_b \neq p_a + p_c$$

Or we can simply use H_0: $p_b = p_c$

$$H_1: p_b \neq p_c$$

McNemar Test statistic:

$$\chi^2 = \frac{(b-c)^2}{b+c} \tag{4.1}$$

where:
b: Number of observations where test-1 gives negative result and test-2 gives positive result. In the context to our experiment, b denotes to the number of observations where the concerned permission is absent in malware applications but present in benign applications.
c: Number of observations where test-1 gives positive result and test-2 gives negative result. In the context to our experiment, c denotes to the number of observations where the concerned permission is present in malware applications but absent in benign applications.
Under the null hypothesis and if the values of b and c are large enough, then χ^2 follows approximately the chi-squared distribution with 1 degree of freedom.

As per Fig 4.3 and Equation 4.1, we can conclude only b and c are required to calculate the test statistic i.e. values where Test 1 is positive and Test 2 is negative and where Test 1 is negative with positive Test 2 are important. In the case of permissions, a represents the number of applications where that feature is present in both benign and malware apps and b represents the number of apps where the feature is absent in Malware apps but present in Benign applications. Similarly, c is for apps where the feature is only present in Benign and d is for apps where the feature is absent in both. For example:

Table 4.2: Observation of feature-I

Malware apps	**Benign Apps**	**Malware-Benign**
0	1	-1
0	1	-1
1	1	0
0	1	-1
0	1	-1
0	1	-1
0	1	-1
1	1	0
1	1	0
0	1	-1
0	0	0
1	0	1
1	1	0
0	0	0
0	0	0

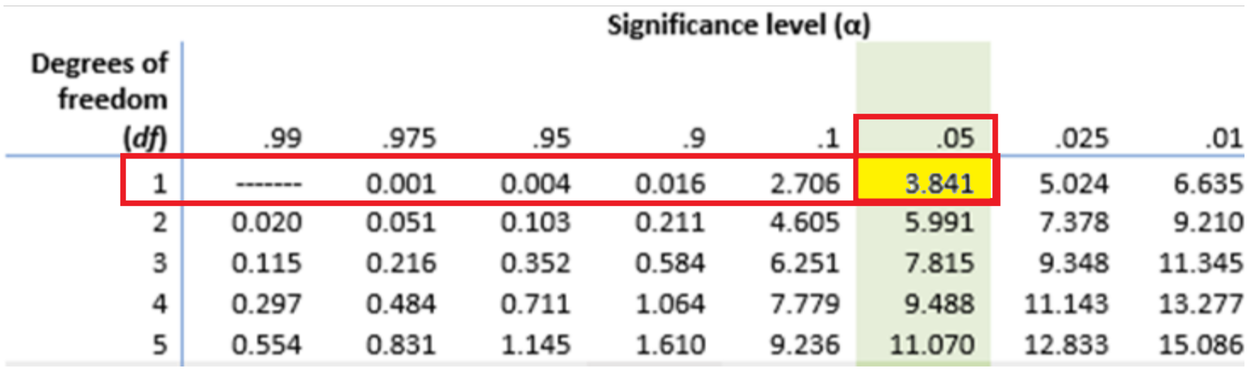
We'll count the number of 1's and (-1)'s of Table 4.2 as they represent the b and c of Fig 4.3.

Table 4.3: McNemar Table for Feauture-I

		Malware		
Benign		**Present**	**Absent**	**Total Applications**
	Present	4	7	11
	Absent	1	3	4
	Total Applications	5	10	**15**

$$\chi = \frac{(7-1)^2}{7+1} = 4.5$$

As the test statistic is less than the chi-square value with $\alpha = 0.05$ and df=1, i.e., 3.814 as shown in fig.4.1. Thus feature-1 is a good feature to make the prediction.
The higher the score of a feature in test statistics, the better rank it will get as the numerator will be bigger only when the difference between the values of b and c would be. Keeping this concept in mind we have calculated the McNemar score for all 129 permission features and ranked them in decreasing order.



Degrees of freedom (*df*)	Significance level (α)							
	.99	.975	.95	.9	.1	.05	.025	.01
1	-------	0.001	0.004	0.016	2.706	3.841	5.024	6.635
2	0.020	0.051	0.103	0.211	4.605	5.991	7.378	9.210
3	0.115	0.216	0.352	0.584	6.251	7.815	9.348	11.345
4	0.297	0.484	0.711	1.064	7.779	9.488	11.143	13.277
5	0.554	0.831	1.145	1.610	9.236	11.070	12.833	15.086

Figure 4.1: Chi-Square value

4.0.4 Detection with Machine Learning techniques

We use machine learning for detection. We add a column to the dataset as malicious_presence and assign 1 to all the malicious applications and 0 to the benign applications. Use the malicious_presence column as y_train and y_test and the rest of the features as x_train and x_test. We are using different machine learning algorithms to get to the best technique for comparing them. We have used the following Machine learning techniques:

- Decision Tree:
  A decision tree is used for both classification and regression problems. It works by recursively splitting the dataset into subsets based on the most important attributes until a stopping criterion is met. The result is a tree-like model that can be easily interpreted and used for predictions.
  The observation shows that decision tree's prediction are fairly accurate and comparatively faster than others

- Logistic Regression:
  Logistic regression is a statistical method used to analyze data with one or more independent variables and a binary dependent variable (where the outcome can be either "yes" or "no", or 1 or 0). It works by fitting a logistic function to the data and making predictions based on the probability of the outcome.

- Random Forest:
  Random forest is an ensemble learning method that combines multiple decision trees to improve the accuracy and reduce the overfitting of the individual trees. It works by building a large number of decision trees on random subsets of the data and combining their predictions to get a more robust result.

- K-Neighbours classifier:
  The k-nearest neighbors algorithm (k-NN) is a simple yet effective machine learning algorithm used for both classification and regression problems. It works by

finding the k closest data points in the training set to a given input data point and predicting the output variable based on the average of the k nearest neighbors' outputs.

- Gaussian NB:
  Gaussian Naive Bayes is a probabilistic machine learning algorithm used for classification. It works by using Bayes' theorem to calculate the probability of a given sample belonging to each class, assuming that the features are independent and follow a Gaussian (normal) distribution. It give the gradual decrease in the accuracy with each elimination, enabling us to see the affect of each feature on detection.

- Perceptron:
  The Perceptron algorithm is a type of supervised learning algorithm used for binary classification. It is a single-layer neural network with one or more inputs and a single output. The algorithm learns by adjusting the weights of the input features to minimize the error between the predicted output and the actual output.
  This technique gives accuracy which doesn't monotonically decrease or increase. The accuracy changes with elimination of each permission.

- SGD Classifier:
  Stochastic Gradient Descent (SGD) Classifier is a linear classifier used for large-scale machine learning problems. It works by updating the weights of the features in small batches (rather than the entire dataset), making it computationally efficient. The algorithm updates the weights based on the gradient of the loss function, which measures the error between the predicted output and the actual output.

We recorded the accuracy of each machine-learning technique for 129 features.
Now as the time of computation was more, we were required to decrease the number of features.
We start eliminating features one by one based on their ranks. The permission ranked the last will be eliminated first as it is least effective in distinguishing the nature of the application. We re-calculate and record the accuracy of each machine-learning technique. The time of computation is not changed much and the accuracy is unaltered. We continue eliminating the features in descending order according to their ranks, one at a time, and recorded the accuracy using the above Machine learning techniques.
For our dataset, with the deletion of features rank-wise, the computation time was decreased effectively but the accuracy was not compromised to a great extent. Thus, we could successfully delete some of the features with fewer distinguishing factors from our dataset. The remaining features are used to detect the test dataset.

Chapter 5: Results

In this chapter, we describe the results obtained from the proposed approach.
We calculated the McNemar test score in MS Excel using the 1 and -1 concept explained in 4.0.3 .
Now as the score are calculated, we will check which permissions can be used to distinguish using the null hypothesis mentioned in 4.0.3. All 129 permissions were found to be eligible to distinguish the nature of the application from malware to benign application. Fig. 5.1 summarizes the McNemar statistic for various top permissions extracted from both normal and malicious datasets. Fig. 5.2 summarizes the ranked permissions obtained from the Mcnemar test. The ranking is done in decreasing order of the Mcnemar test score. Thus, the top-ranked permission, i.e., MOUNT_UNMOUNT_FILESYSTEMS is the most distinguishing permission with the highest Mcnemar test score. Similarly, READ_PHONE_STATE is the second most distinguishing permission between malware and normal Android apps. Such a ranking helps us eliminate lower-ranked features in our detection model.

We applied all the machine learning algorithms discussed in 4.0.3. Detection with machine learning and recorded the accuracy. Now using the rank obtained and mentioned in Fig. 5.2 we eliminate one permission from the bottom. And observed that the accuracy was not altered. The result was recorded with the column head " 129" as 129^{th} ranked permission has been deleted. Moving on we eliminated the permission ranked 128 and applied the machine learning algorithms to record the results. The accuracy is still not altered. We recorded the result under the column head "128".As shown in Fig. 5.4 Continuing this process, we observe a drastic drop in accuracy in eliminating permissions ranked less than 38.
We still record the results of machine learning techniques on the elimination of permissions ranked less than 38 to observe the change in accuracy as shown in Fig. 5.4

The set of 129 permissions extracted from malware and normal apps from the Androzoo website and Google play store respectively gave an accuracy of 97.4% (highest) with the Random forest technique and 94.6% as an average of the scores of the Decision tree, Logistic Regression, Random forest, K-Neighbor classifier, Gaussian NB and SGD classifier. On deleting permissions from the bottom of the ranked list, we observed a minute variation in accuracy in all the machine-learning algorithms used. Till 91 per-

Table 5.1: McNemar statistic for Permissions

Permissions	**McNemar**
RECEIVE_SMS	8576.757551
RUN_INSTRUMENTATION	1865.253536
READ_EXTERNAL_STORAGE	20774.55954
RECORD_AUDIO	18530.40717
AUTHENTICATE_ACCOUNTS	1760.282051
ACCESS_DOWNLOAD_MANAGER	4436.691657
INSTALL_SHORTCUT	8630.160636
CHANGE_BADGE	477.2357445
VIBRATE	19206.62323
READ_PROFILE	972.5190698
SET_WALLPAPER_HINTS	14.30970556
READ_APP_BADGE	1295.892009
REORDER_TASKS	3097.990669
RECEIVE_WAP_PUSH	994.1038062
GET_PACKAGE_SIZE	1863.046314
SYSTEM_OVERLAY_WINDOW	2199.155287
BROADCAST_STICKY	9599.004979
ACCESS_FINE_LOCATION	20697.61252
WRITE_EXTERNAL_STORAGE	15343.24878
READ_USER_DICTIONARY	608.570297

Table 5.2: Ranked Permissions

Rank	**Permission**	**Score**
1	MOUNT_UNMOUNT_FILESYSTEMS	39594.740
2	READ_PHONE_STATE	38994.367
3	CHANGE_WIFI_STATE	38321.160
4	GET_TASKS	38246.832
5	SYSTEM_ALERT_WINDOW	32718.927
6	WRITE_SETTINGS	31189.217
7	READ_LOGS	27290.567
8	CHANGE_NETWORK_STATE	27072.370
9	ACCESS_WIFI_STATE	24819.881
10	ACCESS_COARSE_LOCATION	22933.530
11	READ_EXTERNAL_STORAGE	20774.559
12	ACCESS_FINE_LOCATION	20697.612
13	VIBRATE	19206.623
14	RECORD_AUDIO	18530.407
15	RECEIVE_USER_PRESENT	17559.991
16	CAMERA	16820.061
17	CALL_PHONE	16702.998
18	ACCESS_LOCATION_EXTRA_COMMAI	16687.617
19	RECEIVE	16468.852
20	WRITE_EXTERNAL_STORAGE	15343.248

Table 5.3: Detection Accuracies

Deleted Ranks	NULL	129	128	127	126	125
Decision Tree	0.967 ± 0.002	0.967 ± 0.001	0.967 ± 0.002	0.966 ± 0.002	0.967 ± 0.002	0.967 ± 0.002
Logistic Regression	0.956 ± 0.002	0.956 ± 0.002	0.956 ± 0.002	0.956 ± 0.002	0.956 ± 0.001	0.957 ± 0.001
Random Forest	0.974 ± 0.001	0.974 ± 0.002	0.974 ± 0.002	0.974 ± 0.002	0.975 ± 0.002	0.974 ± 0.002
K-Neighbours classifier	0.964 ± 0.002	0.966 ± 0.0020	0.9650 ± 0.02	0.966 ± 0.002	0.966 ± 0.002	0.965 ± 0.002
Gaussian NB	0.876 ± 0.004	0.877 ± 0.002	0.877 ± 0.002	0.877 ± 0.004	0.877 ± 0.004	0.878 ± 0.003
Perceptron	0.933 ± 0.008	0.927 ± 0.010	0.934 ± 0.006	0.936 ± 0.006	0.922 ± 0.029	0.927 ± 0.016
SGD classifier	0.953 ± 0.002	0.951 ± 0.003	0.953 ± 0.003	0.954 ± 0.004	0.953 ± 0.003	0.953 ± 0.003

Table 5.4: Detection Accuracies

Deleted Ranks	38	37	36	35	34
Decision Tree	0.963 ± 0.002	0.963 ± 0.002	0.962 ± 0.002	0.962 ± 0.002	0.962 ± 0.002
Logistic Regression	0.950 ± 0.002	0.950 ± 0.001	0.950 ± 0.001	0.949 ± 0.001	0.949 ± 0.002
Random Forest	0.970 ± 0.003	0.969 ± 0.001	0.968 ± 0.002	0.969 ± 0.002	0.967 ± 0.001
K-Neighbours classifier	0.963 ± 0.003	0.963 ± 0.002	0.956 ± 0.017	0.963 ± 0.002	0.962 ± 0.001
Gaussian NB	0.900 ± 0.002	0.901 ± 0.004	0.904 ± 0.003	0.903 ± 0.003	0.905 ± 0.003
Perceptron	0.892 ± 0.058	0.914 ± 0.033	0.921 ± 0.017	0.915 ± 0.017	0.914 ± 0.021
SGD classifier	0.944 ± 0.003	0.946 ± 0.003	0.944 ± 0.003	0.946 ± 0.002	0.944 ± 0.004

missions were not deleted the accuracy was not compromised to two decimal places. On reaching 4th ranked permission i.e. after deleting 125 permissions, accuracy changes to one decimal position and falls to 85% for Random forest. Thus, we conclude that we get the highest accuracy of 97% with the proposed detection model applied on ranked permissions with the Mcnemar test.

Chapter 6: Conclusion

6.0.1 Thesis Summary

In this Thesis, we introduced a statistical technique named McNemar test to find the correlation of a set of permissions with malware and benign applications and used the result to predict if the given application is malware using various machine learning algorithms. By using the correlation statistic we ranked the permissions based on the ability to differentiate between malware and benign applications. After ranking the permissions using McNemar's statistical score, we started deleting them in rank's descending order. With each deletion from the set of permissions, we applied the machine-learning algorithms to test the accuracy and found that accuracy was not compromised till the deletion of 38th ranked permission to two decimal positions. Using that data we can conclude that out of 129 permissions, only 38 permissions were essential to give the highest 97% detection accuracy with the Random forest technique. In our future work, we will look to integrate other manifest file components as well like intents, hardware features, services, activities, etc for our analysis. We have used the concept of correlation under statistics for permissions analysis. While examining relationships between quantitative or categorical variables, correlation is used. In other words, it's a measurement of the degree of a relationship. Correlation analysis is the investigation of the relationships between different variables. It also evaluates the strength and direction of the linear links between two sets of variables. Correlations are helpful because they allow you to predict future behavior by revealing the relationships that various factors have.

6.0.2 Contribution

We have used this concept in order to find the correlation between different features in malware applications and benign applications using McNemar Test [24]. McNemar's test is one of the best-known tests to compare two correlated binomial proportions. The salient feature of McNemar's test is that we compute the variance of the contrast estimator under the restriction that the null hypothesis is true. We explain the details of the test in 4.0.3 of the thesis. We rank the permissions obtained from malware and normal dataset using the Mcnemar test. Further, we apply a novel detection algorithm on the ranked permissions with machine learning techniques to effectively detect Android malware.

6.0.3 Future Work

In the future, it is envisioned that the integration of the McNemar test with various statistical methodologies and machine learning concepts will be undertaken to obtain results that are applicable to larger datasets. This approach aims to leverage the strengths of both the McNemar test and advanced analytical techniques in order to address the challenges posed by high-volume data. By combining these methodologies, it is anticipated that more accurate and robust insights can be derived, thereby enhancing the efficacy and applicability of the analysis. Such an integration holds promise for expanding the scope and scalability of the research.

Bibliography

[1] RPNDroid: M. Upadhayay, A. Sharma, G. Garg and A. Arora, ”RPNDroid: Android Malware Detection using Ranked Permissions and Network Traffic,” 2021 Fifth World Conference on Smart Trends in Systems Security and Sustainability (WorldS4), London, United Kingdom, pp. 19-24,2021.

[2] Xiangyu-Ju, ”Android malware detection through permission and package,” 2014 International Conference on Wavelet Analysis and Pattern Recognition, Lanzhou, China, pp. 61-65,2014.

[3] A. Sangal and H. K. Verma, ”A Static Feature Selection-based Android Malware Detection Using Machine Learning Techniques,” 2020 International Conference on Smart Electronics and Communication (ICOSEC), Trichy, India, pp. 48-51,2020.

[4] T. Lu and S. Hou, ”A Two-Layered Malware Detection Model Based on Permission for Android,” 2018 IEEE International Conference on Computer and Communication Engineering Technology (CCET), Beijing, China, pp. 239-243,2018.

[5] A. Utku, I. A. DoGru and M. A. Akcayol, ”Permission based android malware detection with multilayer perceptron,” 2018 26th Signal Processing and Communications Applications Conference (SIU), Izmir, Turkey, pp. 1-4,2018.

[6] S. P., K. P. B., A. K. K. and A. T., ”Detection of Permission Driven Malware in Android Using Deep Learning Techniques,” 2019 3rd International conference on Electronics, Communication and Aerospace Technology (ICECA), Coimbatore, India, pp. 941-945,2019.

[7] S. R. Tuan Mat, M. F. A. Razak, M. N. M. Kahar, J. M. Arif and A. Zabidi, ”Applying Bayesian probability for Android malware detection using permission features,” 2021 International Conference on Software Engineering & Computer Systems and 4th International Conference on Computational Science and Information Management $ICSECS - ICOCSIM$, Pekan, Malaysia, pp. 574-579, 2021.

[8] J. Li, L. Sun, Q. Yan, Z. Li, W. Srisa-an and H. Ye, ”Significant Permission Identification for Machine-Learning-Based Android Malware Detection,” in IEEE Transactions on Industrial Informatics, vol. 14, no. 7, pp. 3216-3225, July 2018.

[9] R. Rahman, M. R. Islam, A. Ahmed, M. K. Hasan and H. Mahmud, ”A Study of Permission-based Malware Detection Using Machine Learning,” 2022 15th International Conference on Security of Information and Networks (SIN), Sousse, Tunisia, pp. 01-06,2022.

[10] M. Chen, Q. Zhou, K. Wang and Z. Zeng, ”An Android Malware Detection Method Using Deep Learning based on Multi-features,” 2022 IEEE International Conference on Artificial Intelligence and Computer Applications (ICAICA), Dalian, China, pp. 187-190,2022.

[11] A. Sangal and H. K. Verma, ”A Static Feature Selection-based Android Malware Detection Using Machine Learning Techniques,” 2020 International Conference on Smart Electronics and Communication (ICOSEC), Trichy, India, pp. 48-51, 2020.

[12] S. Guan and W. Li, ”EnsembleDroid: A Malware Detection Approach for Android System based on Ensemble Learning,” 2022 IEEE MIT Undergraduate Research Technology Conference (URTC), Cambridge, MA, USA, pp. 1-5, 2022.

[13] Y. -F. Lu, C. -F. Kuo, H. -Y. Chen, C. -W. Chen and S. -C. Chou, ”A SVM-Based Malware Detection Mechanism for Android Devices,” 2018 International Conference on System Science and Engineering (ICSSE), New Taipei, Taiwan, pp. 1-6,2018.

[14] S. R. Tiwari and R. U. Shukla, ”An Android Malware Detection Technique Based on Optimized Permissions and API,” 2018 International Conference on Inventive Research in Computing Applications (ICIRCA), Coimbatore, India, pp. 258-263,2018.

[15] J. McGiff, W. G. Hatcher, J. Nguyen, W. Yu, E. Blasch and C. Lu, ”Towards Multimodal Learning for Android Malware Detection,” 2019 International Conference on Computing, Networking and Communications (ICNC), Honolulu, HI, USA, pp. 432-436, 2019.

[16] W. -C. Kuo, T. -P. Liu and C. -C. Wang, ”Study on Android Hybrid Malware Detection Based on Machine Learning,” 2019 IEEE 4th International Conference on Computer and Communication Systems (ICCCS), Singapore, pp. 31-35,2019.

[17] S. J. K., S. Chakravarty and R. K. Varma P., ”Feature Selection and Evaluation of Permission-based Android Malware Detection,” 2020 4th International Conference on Trends in Electronics and Informatics (ICOEI)(48184), Tirunelveli, India, pp. 795-799, 2020.

[18] E. Amer, ”Permission-Based Approach for Android Malware Analysis Through Ensemble-Based Voting Model,” 2021 International Mobile, Intelligent, and Ubiquitous Computing Conference (MIUCC), Cairo, Egypt, pp. 135-139,2021.

[19] A. A. Sahal, S. Alam and I. Soğukpinar, ”Mining and Detection of Android Malware Based on Permissions,” 2018 3rd International Conference on Computer Science and Engineering (UBMK), Sarajevo, Bosnia and Herzegovina, pp. 264-268,2018.

[20] D. Ö. Şahın, O. E. Kural, S. Akleylek and E. Kiliç, ”New results on permission based static analysis for Android malware,” 2018 6th International Symposium on Digital Forensic and Security (ISDFS), Antalya, Turkey,pp. 1-4,2018. Özkan Şahın

[21] J. Li, L. Sun, Q. Yan, Z. Li, W. Srisa-an and H. Ye, ”Significant Permission Identification for Machine-Learning-Based Android Malware Detection,” in IEEE Transactions on Industrial Informatics, vol. 14, no. 7, pp. 3216-3225, July 2018.

[22] D. Ö. Şahın, S. Akleylek and E. Kiliç, ”LinRegDroid: Detection of Android Malware Using Multiple Linear Regression Models-Based Classifiers,” in IEEE Access, vol. 10, pp. 14246-14259, 2022.

[23] P. Xiong, X. Wang, W. Niu, T. Zhu and G. Li, ”Android malware detection with contrasting permission patterns,” in China Communications, vol. 11, no. 8, pp. 1-14, Aug. 2014.

[24] Pembury Smith, M.Q.R., Ruxton, G.D. Effective use of the McNemar test. Behav Ecol Sociobiol 74, 133 (2020).

[25] Westfall PH, Troendle JF, Pennello G. Multiple McNemar tests. Biometrics. 2010 Dec;66(4):1185-91.

[26] N. Sushmakar, N. Oberoi, S. Gupta and A. Arora, ”An Unsupervised Based Enhanced Anomaly Detection Model Using Features Importance,” 2022 2nd International Conference on Intelligent Technologies (CONIT), Hubli, India, pp. 1-7, 2022.

[27] Raman, S. K. Jha and A. Arora, ”An Enhanced Intrusion Detection System Using Combinational Feature Ranking and Machine Learning Algorithms,” 2022 2nd International Conference on Intelligent Technologies (CONIT), Hubli, India, pp. 1-8, 2022.

[28] Y. Sharma, S. Sharma and A. Arora, ”Feature Ranking using Statistical Techniques for Computer Networks Intrusion Detection,” 2022 7th International Conference on Communication and Electronics Systems (ICCES), Coimbatore, India, pp. 761-765, 2022.

[29] A. Arora, S. Garg and S. K. Peddoju, ”Malware Detection Using Network Traffic Analysis in Android Based Mobile Devices,” 2014 Eighth International Conference on Next Generation Mobile Apps, Services and Technologies, Oxford, UK, pp. 66-71, 2014.

[30] A. Arora and Sateesh K. Peddoju, ”Minimizing Network Traffic Features for Android Mobile Malware Detection”, In Proceedings of the 18th International Conference on Distributed Computing and Networking (ICDCN ’17). Association for Computing Machinery, New York, NY, USA, Article 32, 1–10, 2017.

[31] A. Arora and S. K. Peddoju, ”NTPDroid: A Hybrid Android Malware Detector Using Network Traffic and System Permissions,” 2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/ 12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE), New York, NY, USA, pp. 808-813, 2018.

[32] A. Arora, S. K. Peddoju, V. Chouhan, and A. Chaudhary, ”Hybrid Android Malware Detection by Combining Supervised and Unsupervised Learning”, In Proceedings of the 24th Annual International Conference on Mobile Computing and Networking (MobiCom ’18). Association for Computing Machinery, New York, NY, USA, 798–800, 2018.

[33] A. Arora, S. K. Peddoju and M. Conti, ”PermPair: Android Malware Detection Using Permission Pairs,” in IEEE Transactions on Information Forensics and Security, vol. 15, pp. 1968-1982, 2020.

[34] K. Khariwal, J. Singh and A. Arora, ”IPDroid: Android Malware Detection using Intents and Permissions,” 2020 Fourth World Conference on Smart Trends in Systems, Security and Sustainability (WorldS4), London, UK, pp. 197-202, 2020.

[35] G. Garg, A. Sharma, and A. Arora, “SFDroid: Android Malware Detection Using Ranked Static Features”, in International Journal of Recent Technology and Engineering, vol. 10, no.1, pp. 142-152, 2021.

[36] S. Gupta, S. Sethi, S. Chaudhary, and A. Arora. ”Blockchain Based Detection of Android Malware using Ranked Permissions”, in International Journal of Engineering and Advanced Technology (IJEAT), vol-10, Issue-5, pp. 68-75, 2021.

Details of Candidate's Publication

The Paper titled **"SMARTPHONE MALWARE DETECTION USING PERMISSIONS AND MCNEMAR TEST"** was accepted in Scopus Indexed conference titled "2023 International Conference on Sustainable Computing and Smart Systems (ICSCSS)", which will be held on 14^{th} June 2023.