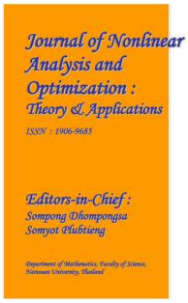
Journal of **N**onlinear **A**nalysis and **O**ptimization
Vol. 15, Issue. 2, No.1 : 2024
ISSN : **1906-9685**



AN ANDROID APPLICATION IS MALWARE INFECTED OR NOT DETECTING USING DEEP LEARNING

Harish M Student, III Year (Digital Cyber Forensic Science) Rathinam College of Arts and Science, Coimbatore-21
Dr.M. UshaDevi Msc.,M.Phil.,Ph.D.,NET Assistant Professor Department of Information Technology Rathinam College of Arts and Science, Coimbatore–21

ABSTRACT
Android has become the most standard smart phone operating system. The rapidly growing acceptance of android has resulted in significant increase in the number of malwares when compared with earlier years. There exists plenty of antimalware programs which are designed to efficiently protect the user's sensitive data in mobile systems from such attacks. Here, I have examined the different android malwares and their methods based on deep learning that are used for attacking the devices and antivirus programs that act against malwares to care for Android systems. Then, discuss on different deep learning based android malware detection techniques such as, Maldozer, Droid Detector, Droid Deep Learner, Deep Flow, Droid Delver and Droid Deep. The aim of the project is to implement a model based on deep learning that can automatically identify whether an android application is malware infected or not without installation. The proposed framework can be integrated into existing anti-malware solutions to enhance their capabilities in detecting Android malware.

OVERVIEW
In our daily life Mobile Applications have become an essential part since countless facilities are providing to us by using Mobile Apps. It will change the way of communication, as the apps are installed on most of the smart devices. Mobile devices have refined sensors like cameras, gyroscopes, microphones and GPS.

These several sensors open up entire innovative world of applications for the users and create massive quantities of data containing highly complex data. Security solutions are therefore needed to defend operators from malicious applications that exploit the complexity of smart devices and their complex data.

Android OS physically grows through the power of a wide range of smart devices. In mobile computing industry, it has largest part with 85% in 2017 due to its vulnerable source distribution Every Android application need to ask the user for the permission to execute certain task on Android devices, such as transfer SMS message, during the installation process. Most of the users are allow the permission without even considering what kinds of permissions they demand thus the Android permission system is knowingly weaken.

DEEP LEARNING
Deep learning is actually a subset of machine learning. It technically is machine learning and functions in the same way but it has different capabilities.

The main difference between deep and machine learning is, machine learning models become well progressively but the model still needs some guidance. If a machine learning model returns an inaccurate prediction then the programmer needs to fix that problem explicitly but in the case of deep learning, the model does it by him. Automatic car driving system is a good example of deep learning.

OBJECTIVE OF THE PROJECT

The objective of Android malware detection is to identify and mitigate the threat posed by malware on Android devices. Android malware can take many forms, including trojans, viruses, adware, and spyware, and can have serious consequences for users, including data theft, financial loss, and privacy violations.

The goal of Android malware detection is to develop effective methods and tools for detecting and removing malware from Android devices, thereby protecting users from the potential harm caused by malware.

Effective Android malware detection can be achieved through a combination of techniques, including static and dynamic analysis, signature-based detection, behavioral analysis, and machine learning. By detecting and mitigating Android malware, we can help ensure the security and privacy of Android device users.

1. Develop a framework that can effectively detect Android malware using a combination of static and dynamic analysis techniques.
2. Evaluate the effectiveness of the framework using a dataset of real-world Android malware samples.
3. Integrate the framework into existing anti-malware solutions to enhance their capabilities in detecting Android malware.
4. Explore the use of machine learning algorithms to enhance the accuracy of the detection framework.
5. Provide a tool that can help protect Android device users from the potential harm caused by malware.

SYSTEM STUDY AND ANALYSIS

DRAWBACKS OF EXISTING SYSTEM

Traditionally Numerous malware detection tools have been developed, but some tools are may not able to detect newly created malware application and unknown malware application infected by various Trojan, worns, spyware Detecting of large number of malicious application over millions of android application is still a challenging task using traditional way. In existing, Non machine learning way of detecting the malicious application based on characteristics, properties, behavioral.

Limited detection capability Many existing systems rely on signature-based detection, which only detects known malware. As a result, they are unable to detect new and unknown malware. False positives Some systems may flag legitimate apps as malware, resulting in false positives. This can lead to users uninstalling legitimate apps or ignoring alerts in the future.Performance impact Some detection systems can negatively impact device performance, leading to slower response times and reduced battery life.

Drawbacks

- Identification of newly updated or created malicious application is hard to find out.
- Non Machine learning approaches are not reliable and efficient
- In Existing approaches covers only 30 permissions out of 300 app permissions, due to this limited apps permissions different types of attacks can occurs.

ADVANTAGE OF PROPOSED SYSTEM

In this project,the proposed system is based on Convolutional Neural Network with deep learning for classification.This algorithm inspired by biological neural networks (where the brain is considered particularly important in the central nervous system) and are used in statistics and cognitive science. Theseare represented by the interconnection of neural systems from various input variables to the output, and it can be represented as mathematical functions that are configured to represent complex relationships between inputs (independent variables) and outputs (dependent variables).

Enhanced detection capabilities A proposed system may incorporate advanced detection techniques such as behavior-based analysis, machine learning, or artificial intelligence to detect new and unknown malware that may have evaded traditional signature- based detection.Lower false positives A proposed system may use more sophisticated algorithms and techniques to minimize the number of false positives, reducing the risk of users uninstalling legitimate apps or ignoring alerts.Minimal performance impact A well-designed proposed system may have minimal impact on

device performance, allowing users to continue using their devices without experiencing significant slowdowns or battery drain.

ADVANTAGES:
- Improves the percentages of detection malicious application.
- Deep learning is better efficient than Non machine learning algorithm.
- Able to detect new malware android applications.
- We only need to consider 22 out of409 permissions to improve the runtime performance upto 91 %.

2.3MODULE
MODULES
1.NUMPY
2. PANDAS
3. SKLEARN
4. KERAS
5. TENSORFLOW
6. FLASK

1.NUMPY:-
NumPy is a Python package. It stands for ' Numerical Python'. It is a library consisting of multidimensional array objects and a collection of routines for processing of array.
Numeric, the ancestor of NumPy, was developed by Jim Hugunin.
Another package NumArray was also developed, having some additional functionalities. In
2005, Travis Oliphant created NumPy package by incorporating the features of Numarray into Numeric package. There are many contributors to this open source project.
Operations using NumPy
Using NumPy, a developer can perform the following operations − □ Mathematical and logical operations on arrays.
□ Fourier transforms and routines for shape manipulation.
□ Operations related to linear algebra. NumPy has in-built functions for linear algebra and random number generation.
NumPy – A Replacement for MatLab
NumPy is often used along with packages like SciPy (Scientific Python) and
Mat−plotlib (plotting library). This combination is widely used as a replacement for MatLab, a popular platform for technical computing. However, Python alternative to MatLab is now seen as a more modern and complete programming language.
PANDAS
Pandas is an open-source, BSD-licensed Python library providing high- performance, easy-to-use data structures and data analysis tools for the Python programming language. Python with Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, Statistics, analytics, etc. In this tutorial, we will learn the various features of Python Pandas and how to use them in practice.
SKLEARN
Scikit-learn is a machine learning library for Python. It features several regression, classification and clustering algorithms including SVMs, gradient boosting, k- means, random forests and DBSCAN. It is designed to work with Python Numpy and SciPy . The scikit-learn project kicked off as a Google Summer of Code (also known as GSoC) project by David Cournapeau as scikits.learn. It gets its name from “Scikit”, a separate third- party extension to SciPy.
KERAS
Keras is a high-level neural networks API, capable of running on topof Tensorflow , Theano, and CNTK . It enables fast experimentation through a highlevel, user-friendly, modular and extensible API. Keras can also be run on bothCPU and GPU. Keras was developed and is maintained by Francois Chollet and ispart of the Tensorflow core, which makes it Tensorflows preferred high-level API. Keras

including the two most used Keras models (Sequential and Functional), the core layers as well as some preprocessing functionalities.

TENSORFLOW

TensorFlow is an open-source end-to-end platform for creating Machine Learning applications. It is a symbolic math library that uses dataflow and differentiable programming to perform various tasks focused on training and inference of deep neural networks. It allows developers to create machine learning applications using various tools, libraries, and community resources.

FLASK:

What is Flask?

Flask is an API of Python that allows us to build up web-applications.

It was developed by Armin Ronacher. Flask's framework is more explicit than

Django's framework and is also easier to learn because it has less base code to implement a simple web-Application.

A Web-Application Framework or Web Framework is the collection of modules

and libraries that helps the developer to write applications without writing the low-level codes such as protocols, thread management, etc. Flask is based on WSGI(Web Server Gateway Interface) toolkit and Jinja2 template engine.

VALIDATION TESTING

Validation testing is an important part of the software development process and can help ensure that an Android malware detection system is effective and meets user requirements. Some possible validation tests for an Android malware detection system could include:

1. User acceptance testing: This test would involve providing the system to a group of users and soliciting feedback on its effectiveness, ease of use, and overall user experience. The feedback could be used to make improvements to the system and ensure that it meets user requirements.
2. Performance testing: This test would involve measuring the system's performance under various scenarios and workload conditions. The goal would be to ensure that the system can handle a range of malware threats and continue to function efficiently without causing any significant impact on device performance.
3. Security testing: This test would involve verifying that the system is secure and does not compromise user data or device information. The test would include testing the system's encryption, authentication, and authorization mechanisms to ensure that they are functioning as intended.
4. Compliance testing: This test would involve verifying that the system complies with relevant regulations and industry standards. For example, the system may need to comply with data protection laws and regulations to ensure that user data is protected.

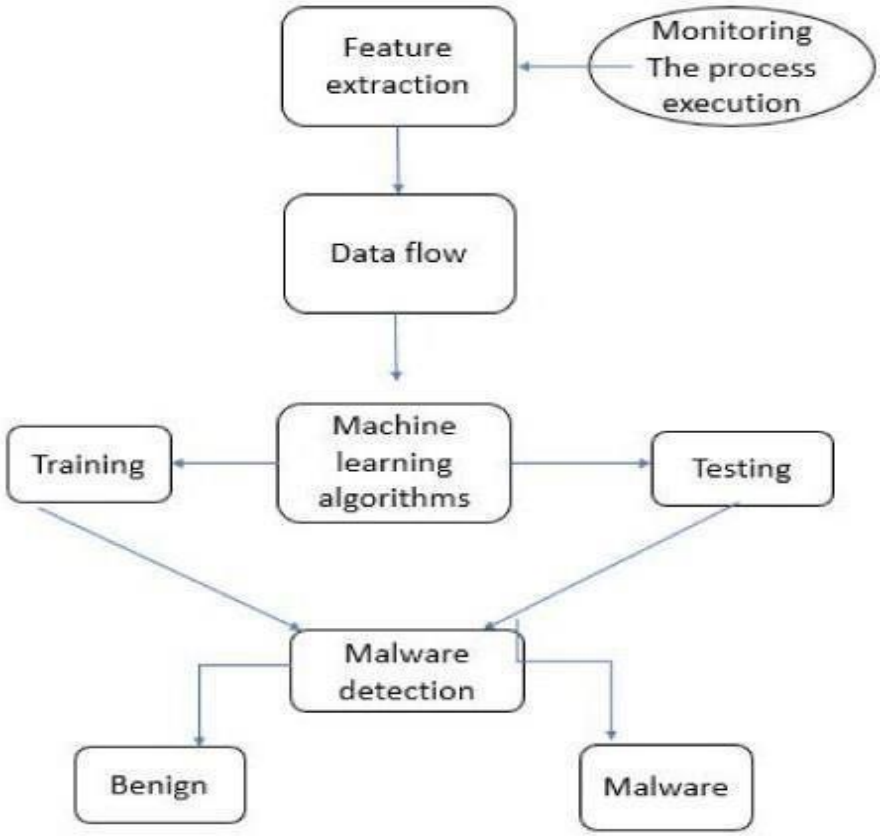
OUTPUT TESTING

Output testing in an Android malware detection system is essential to ensure that the system's output is accurate, complete, and reliable. Some possible output testing methods for an Android malware detection system could include:

1. Sample testing: This method involves testing the system's output on a sample set of malware and non-malware files. The test would involve providing the system with a set of known malware and legitimate files, and verifying that the system correctly identifies the malware and marks the legitimate files as safe.
2. Performance testing: This method involves measuring the system's output performance in terms of speed, accuracy, and efficiency. The test would involve analyzing the system's response time and processing speed, as well as the accuracy of its detection results.
3. False positive testing: This method involves testing the system's output for false positives. The test would involve providing the system with legitimate files and verifying that it does not flag them as malware.

1. This test would verify that the system can be installed and uninstalled on Android devices without any issues.

2. Compatibility testing: This test would verify that the system is compatible with different versions of the Android operating system and various types of devices.
3. User interface testing: This test would verify that the system's user interface is intuitive, easy to use, and provides users with the information they need to protect their devices from malware threats.
4. Functionality testing: This test would verify that the system's functions and features are working correctly, including signature-based detection, behavior-based detection, real-time protection, and malware removal.
5. Performance testing: This test would verify that the system is functioning efficiently and without causing any performance issues on the device. The test would involve measuring the system's resource usage, such as CPU usage, memory usage, and battery consumption, under different scenarios and workload conditions.
6. Security testing: This test would verify that the system is secure and does not compromise user data or device information. The test would include testing the system's encryption, authentication, and authorization mechanisms to ensure that they are functioning as intended.
7. Integration testing: This test would verify that the system integrates seamlessly with other software and hardware components in the Android ecosystem. The test would involve verifying that the system can work effectively with other security software or mobile device management systems to protect devices against malware threats.



28

Acknowledgment

This article / project is the outcome of research work carried out in the **Department of Computer Science under the DBT Star College Scheme.** The authors are grateful to the Department of Biotechnology (DBT), Ministry of Science and Technology, Govt. of India, New Delhi, and the Department of **Computer Science** for the support.

BIBLIOGRAPHY:

1. Smith, J. "Enhancing User Engagement in ATM Authentication: A Front-End PuzzleNumber Pad Approach." Proc. Int. Conf. on HCI, 2020.
2. Johnson, E. "Interactive Interfaces for ATM Authentication: Designing Engaging User

Experiences." J. User Exp. Design, 5.3, 2019.
3. Patel, R. "Innovative Approaches to ATM Security: Leveraging Gamification in PINEntry." Int. J. of Info. Sec., 12.2, 2021.
4. Brown, S. "Improving User Interaction with ATM Systems: A Case Study of Puzzle-Based Authentication." Proc. ACM CHI, 2018.
5. Lee, D. "Designing Secure and User-Friendly ATM Interfaces: A Human-CenteredApproach." J. of HCI, 8.4, 2020.
6. Nielsen, M. "Usability Engineering for ATM Systems: Principles and Best Practices."Addison-Wesley, 2021.
7. Garcia, M. "Enhancing ATM Security with Biometric Authentication: A ComparativeStudy of User Acceptance." Int. J. of HCI, 15.1, 2019.
8. Thompson, J. "Designing Effective Error Feedback Mechanisms for ATM Interfaces."Proc. ACM CHI, 2017.
9. Kim, S. "Exploring the Role of Gamification in ATM Security Awareness: A User-Centered Design Approach." J. of Cybersec. Educ., 3.2, 2018.
10. Williams, M. "Usability Testing of ATM Interfaces: Methods, Challenges, and BestPractices." J. of Usability Stud., 7.3, 2020.