

ANDROID MALWARE DETECTION USING ENSEMBLE LEARNING APPROACH

Suhaib Ahmed¹, Mohd Shoeb², Shaik Nayeem³, Mohammed Rahmat Ali⁴

^{1,2,3}B. E Student, Department of CSE, ISL College of Engineering, India.

⁴Assistant Professor, Department of CSE, ISL College of Engineering, Hyderabad, India.

ABSTRACT: Android platform due to open-source characteristics and Google backing has the largest global market share. Being the world's most popular operating system, it has drawn the attention of cyber criminals operating particularly through the wide distribution of malicious applications. This paper proposes an effectual machine-learning-based approach for Android Malware Detection making use of an evolutionary chi-square algorithm for discriminatory feature selection. Selected features from the chi-square algorithm are used to train machine learning classifiers and their capability in identification of Malware before and after feature selection is compared. The experimentation results validate that the chi-square algorithm gives the most optimized feature subset helping in the reduction of feature dimension to less than half of the original feature set. Classification accuracy of more than the previous percentage is maintained post-feature selection for the machine learning-based classifiers, while working on much reduced feature dimension, thereby, having a positive impact on the computational complexity of learning classifiers.

Index Terms — Cybersecurity, malware detection, ensemble learning, hunter prey optimization, machine learning.

1. INTRODUCTION

Network engineers and computer scientists are starting to focus more and more on cybersecurity, thus there are a number of issues that need to be resolved. As a result of the rapid advancement of technology and its innate incorporation into all facets of life, targets and malware applications are more recognized and researched. The malware kind that has attracted the most attention online is Android malware. Android is one popular operating system that now leads the operating system market.

Since few malware apps contain more than 50 attributes that make detection challenging, malware intrusive techniques arise to evade recognition. Therefore, developing strategies to address the ongoing proliferation of Android malware is crucial in order to locate, disable, or eradicate it effectively. Researchers in the field are motivated to carry out further study in order to identify malware and correctly handle it by all of these challenges.

2. OBJECTIVE OF THE PROJECT

A. Existing System:

Network engineers and computer scientists are starting to focus more and more on cybersecurity, thus there are a number of issues that need to be resolved. As a result of the rapid advancement of technology and its innate

incorporation into all facets of life, targets and malware applications are more recognized and researched. The malware kind that has attracted the most attention online is Android malware. Android is one popular operating system that now leads the operating system market.

Since few malware apps contain more than 50 attributes that make detection challenging, malware intrusive techniques arise to evade recognition. Therefore, developing strategies to address the ongoing proliferation of Android malware is crucial in order to locate, disable, or eradicate it effectively. Researchers in the field are motivated to carry out further study in order to identify malware and correctly handle it by all of these challenges.

Disadvantages

- No Detection of Malware using ML & DL

B. Proposed system:

Because of its ability to identify a feature subset chosen from the original feature vector that provides the greatest accuracy for classifiers on which they are trained, the chi-square technique has been employed in the proposed study. It has also been used in the past to find the best feature subset when combined with machine learning methods. The primary contribution of the work is the chi-square algorithm's reduction of the feature dimension to less than half of the original feature set, which enables machine learning classifiers to be trained with less complexity while retaining accuracy in malware classification. The RandomForestClassifier and Extra Trees Classifier ensemble learning techniques are trained utilizing the optimum feature set that is acquired by the application of the chi-square algorithm. Additionally, neural networks and deep learning techniques are being used in this system to increase detection accuracy. It is found that while operating on a much smaller feature dimension, a respectable classification accuracy of more than the prior % is maintained, which lowers the classifiers' training time complexity.

Advantages:

- Analyze the Malware API calls by using Machine Learning & Deep Learning algorithm
- Comparing the performances of each algorithm with accuracy scores.
- maintaining privacy, and security, and providing better accuracy.
- Using 5 ML algorithms and Deep Learning Algorithms.

C. Modules Description

Algorithms:

- **RandomForestClassifier**
- **Extra Trees Classifier**
- **Artificial neural network**
- **CNN**

Random Forest

We use random forest as a classifier in this experiment. Decision tree models are widely used in data mining because of their flexibility in handling various data attribute kinds and their algorithmic simplicity. Nevertheless, the single-tree model may be easily overfitted and susceptible to certain training data. Ensemble approaches are more accurate than single classifiers and can tackle these difficulties by combining a number of individual choices in some manner. One of the ensemble approaches, random forest, combines many tree predictors such that every tree in the forest has the same distribution and each tree relies on a random independent dataset. The strength of each individual tree as well as the association between several trees determine the random forest's capacity. The random forest performs better the stronger the single tree and the lower the correlation between various trees. The randomization of trees, which uses bootstrapped samples and a random selection of a subset of the data characteristics, accounts for their variance.

Below is the step by step Python implementation. ...

Step 2 : Import and print the dataset.

Step 3 : Select all rows and column 1 from dataset to x and all rows and column 2 as y.

Step 4 : Fit Random forest regressor to the dataset.

Step 5 : Predicting a new result.

Step 6 : Visualising the result.

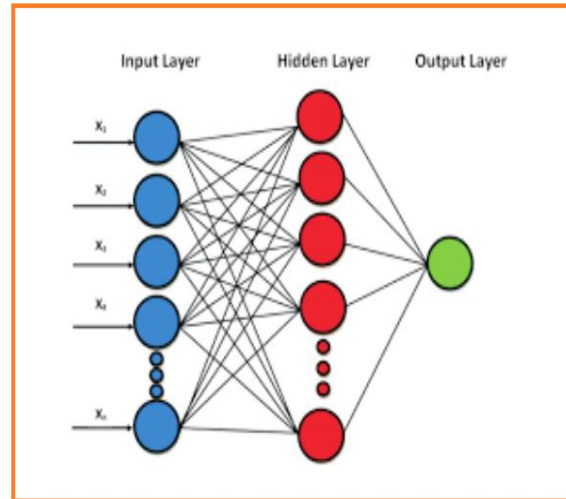
Extra Trees Classifier

Extremely Randomized Trees Classifier(Extra Trees Classifier) is a kind of ensemble learning method that produces a classification result by combining the output of many de-correlated decision trees gathered in a "forest." It is conceptually extremely similar to a Random Forest Classifier and only varies in how the forest's decision trees are built.

The initial training sample serves as the foundation for every Decision Tree in the Extra Trees Forest. The decision trees are then given a random sample of k features from the feature-set at each test node, and they are required to choose the optimal feature to divide the data according to a set of mathematical criteria (usually the Gini Index). Several de-correlated decision trees are produced as a result of this random feature selection process.

During the construction of the forest, for each feature, the normalized total reduction in the mathematical criteria used in the feature of split decision (Gini Index if the Gini Index is used in the construction of the forest) is computed in order to perform feature selection using the above forest structure. The Gini Importance of the feature is the name given to this value. The process of feature selection involves sorting each feature based on its Gini Importance in decreasing order, and the user chooses the top k features based on personal preference.

Neural networks



Output layer. Therefore, we can define neural network as information flows from inputs through hidden layers towards the output. For a 3-layers neural network, the learned function would be: $f(x) = f_3(f_2(f_1(x)))$ where:

$f_1(x)$: Function learned on first hidden layer

$f_2(x)$: Function learned on second hidden layer

$f_3(x)$: Function learned on output layer

Lets first introduce some notations that will be used throughout the post:

- W^l : Weights matrix for the l^{th} layer
- b^l : Bias vector for the l^{th} layer
- Z^l : Linear (affine) transformations of given inputs for the l^{th} layer
- g^l : Activation function applied on the l^{th} layer
- A^l : Post-activation output for the l^{th} layer
- dW^l : Derivative of the cost function w.r.t W^l ($\frac{\partial J}{\partial W^l}$)
- db^l : Derivative of the cost function w.r.t b^l ($\frac{\partial J}{\partial b^l}$)
- dZ^l : Derivative of the cost function w.r.t Z^l ($\frac{\partial J}{\partial Z^l}$)
- dA^l : Derivative of the cost function w.r.t A^l ($\frac{\partial J}{\partial A^l}$)
- n^l : Number of units (nodes) of the l^{th} layer
- m : Number of examples
- L : Number of layers in the network (not including the input layer)

Next, we'll write down the dimensions of a multi-layer neural network in the general form to help us in matrix multiplication because one of the major challenges in implementing a neural network is getting the dimensions right.

- W^l, dW^l : Number of units (nodes) in l^{th} layer x Number of units (nodes) in $l - 1$ layer
- b^l, db^l : Number of units (nodes) in l^{th} layer x 1
- Z^l, dZ^l : Number of units (nodes) in l^{th} layer x number of examples
- A^l, dA^l : Number of units (nodes) in l^{th} layer x number of examples

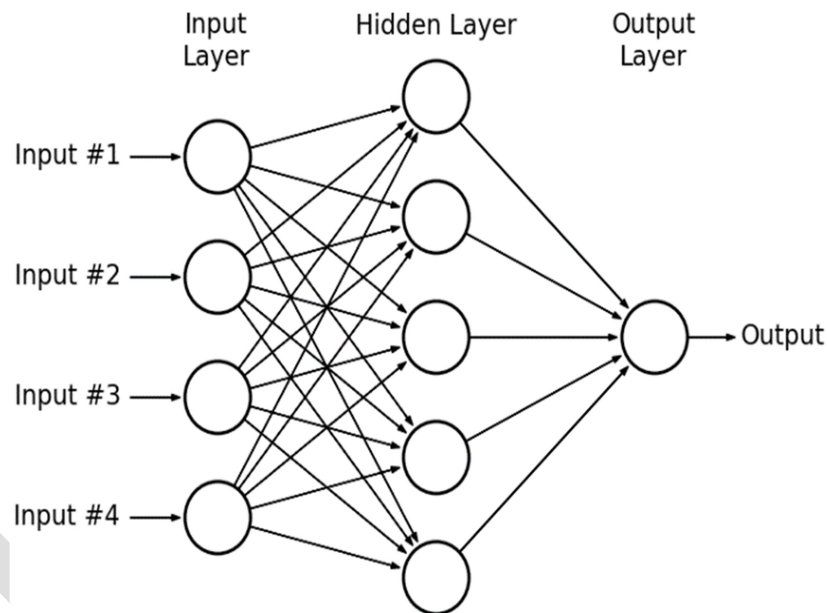
Multi-Layer Perceptrons

Artificial neural networks are sometimes termed multi-layer perceptrons after the most useful variety. Larger neural networks began with perceptrons, single neuron models.

It studies how basic biological brain models may tackle computing problems like machine learning predictive modeling. Instead of creating accurate brain models, we want to construct strong methods and data structures for modeling complex situations.

Neural networks are powerful because they learn how to match your training data to the output variable you wish to predict. Neural networks map. They can learn any mapping function and are a universal approximation method.

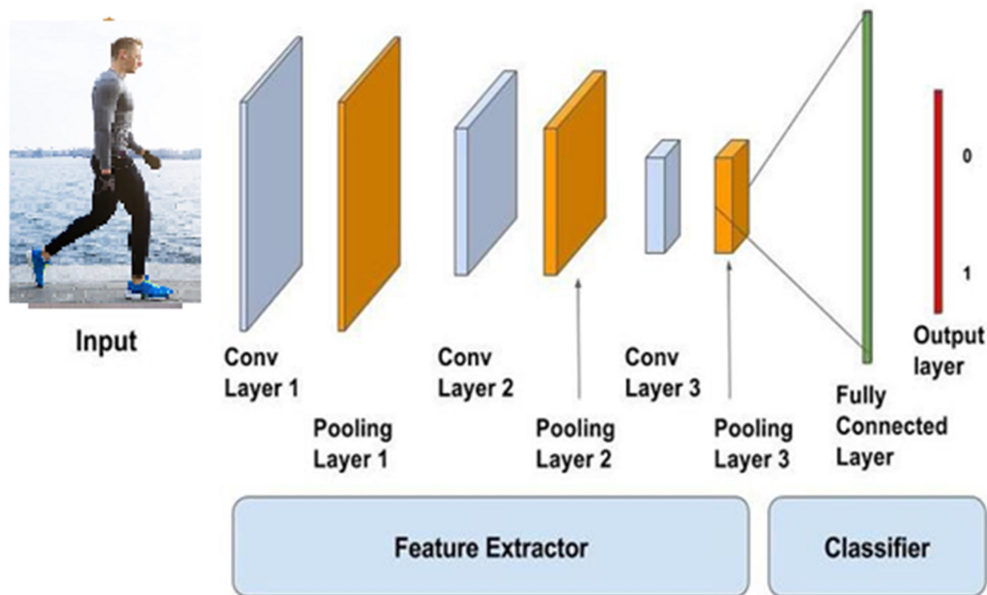
Their hierarchical or multi-layered structure makes neural networks predictive. A data structure may learn to represent characteristics at various sizes or resolutions and combine them into higher-order features. Examples include lines, groups of lines, and forms.



Convolution Neural Networks or covnets are neural networks that share their parameters

Types of layers:

1. Input Layer
2. Convolution Layer
3. Activation Function Layer
4. Pool Layer
5. Fully-Connected Layer



Input Layer: This layer holds the raw input of image with width 32, height 32 and depth 3.

Convolution Layer: This layer computes the output volume by computing dot product between all filters and image patch. Suppose we use total 12 filters for this layer we'll get output volume of dimension 32 x 32 x 12.

Activation Function Layer: This layer will apply element wise activation function to the output of convolution layer. Some common activation functions are RELU: $\max(0, x)$, Sigmoid: $1/(1+e^{-x})$, Tanh, Leaky RELU, etc.

LITERATURE SURVEY

4. Literature survey

4.1 Survey on “Android Malware Classification Using Machine Learning and Bio-Inspired Optimization Algorithms”:

Authors: Jack Pye, Biju Issac, Husnain Rafiq, Nauman Aslam

Description:

Number and complexity of Android viruses have risen considerably in recent years. A prototype framework that utilizes static analysis to categorize Android malware using two feature sets: AndroidManifest.xml permissions and Classes.dex classes. Random Forest, SGD, SVM, and Neural networks were trained using the retrieved features. Bio-inspired optimization methods including Particle Swarm Optimization, Artificial Bee Colony Optimization (ABC), Firefly optimization, and Genetic algorithm were used to optimize each machine learning algorithm. The prototype framework was tested using CICAndMal2019, KuafuDet, and Andro-Dump datasets. Better feature selection might boost accuracy.

4.2 Survey on “MADAM: Effective and Efficient Behavior-based Android Malware Detection and Prevention”:

Authors: A. Saracino, D. Sgandurra, G. Dini, and F. Martinelli

Description:

Android users are continuously threatened by more malware-laden apps. Malware endangers privacy, money, devices, and files. This work shows that by observing their activities, we may categorize malware into a few behavioral classes, each with a restricted collection of misbehaviors. Monitoring features from various Android layers reveals these misbehaviors. We introduce MADAM, a revolutionary host-based malware detection system for Android devices that analyzes and correlates kernel, application, user, and package characteristics to identify and halt harmful activity. MADAM was meant to account for the behaviors of most genuine malware. MADAM uses two simultaneous classifiers and a behavioral signature-based detector to identify and ban over 96% of malicious applications from three big datasets including 2,800 apps. The low false alarm rate, small performance overhead, and low energy usage have been shown by extensive trials, including the examination of 9,804 legitimate applications.

4.3 Survey on “Fest: A Feature Extraction and Selection Tool for Android Malware Detection”:

Authors: K. Zhao, D. Zhang, X. Su, and W. Li

Description:

Android's numerous applications make it a popular mobile OS. However, inefficient and faulty detection technologies leave most third-party Android spyware undetected, compromising users' privacy. Previous Android malware detection efforts manually chose attributes to create accurate classification models and seldom used feature selection techniques to discover common aspects. Festival is a feature-based machine learning malware detection approach provided in this study. First, we use AppExtractor to extract permissions and APIs by criteria. We then introduce FrequenSel, a feature selection algorithm. Because often used characteristics in malware but seldom used in benign applications are more crucial to detect malware from benign apps, FrequenSel picks features by frequency difference.

4.4 Survey on “Significant Permission Identification for Machine-Learning-Based Android Malware Detection”:

Authors: J. Li, L. Sun, Q. Yan, Z. Li, W. Srisa-An, and H. Ye

Description:

Malicious applications are growing rapidly, threatening the mobile ecosystem. A new Android malware app is released every 10 seconds, according to a research. We need a scalable malware detection method that can quickly identify malicious programs to fight this significant cyber campaign. Many system- and network-level malware detection techniques exist. Scaling app detection for a big bundle is difficult. In this work, we develop SIGPID, a permission use analysis-based malware detection system to combat the increasing rise of Android malware. We use 3-level trimming to discover the most important Android permissions that may differentiate benign from malicious applications instead of collecting and evaluating all permissions. SIGPID then classifies malicious and benign programs using machine learning. We found just 22 permissions meaningful. We next evaluate our 22-permission strategy against a baseline technique that evaluates all permissions. The results show

that using a Support Vector Machine (SVM) as the classifier yields over 90% precision, recall, accuracy, and F-measure, which are similar to the baseline approach, but with analysis times 4 to 32 times lower than using all permissions.

4.5 Survey on “A review on feature selection in mobile malware detection”:

Authors: A. Feizollah, N. B. Anuar, R. Salleh, and A. W. A. Wahab

Description:

Information exchange has changed due to the increased usage of mobile devices over PCs. Personal computer sales are declining while mobile device shipments are rising. Users are also interested in mobile devices' rising power and mobility. Users love such gadgets, and attackers love them too. Mobile malware is growing quickly, taking users' data, sending premium SMS, and calling unknown premium lines. Several research have found ways to prevent such assaults. We must choose a few characteristics among hundreds to create an effective detection system. We examined 100 2010–2014 research papers on feature selection in mobile malware detection. We divide features into static, dynamic, hybrid, and application metadata. We also examine contemporary research datasets and assessment methods.

TECHNOLOGY DESCRIPTION AND IMPLEMENTATION

History of Python

Python was developed by Guido van Rossum in the late eighties and early nineties at the National Research Institute for Mathematics and Computer Science in the Netherlands. Python is derived from many other languages, including ABC, Modula-3, C, C++, Algol-68, SmallTalk, and Unix shell and other scripting languages.

Python is copyrighted. Like Perl, Python source code is now available under the GNU General Public License (GPL).

Python is now maintained by a core development team at the institute, although Guido van Rossum still holds a vital role in directing its progress.

Input as CSV File

Reading data from CSV (comma separated values) is a fundamental necessity in Data Science. Often, we get data from various sources which can get exported to CSV format so that they can be used by other systems. The Pandas library provides features using which we can read the CSV file in full as well as in parts for only a selected group of columns and rows.

The CSV file is a text file in which the values in the columns are separated by a comma. Let's consider the following data present in the file named input.csv. You can create this file using windows notepad by copying and pasting this data. Save the file as input.csv using the save As All files (*.*) option in notepad.

```
import pandas as pd
data= pd.read_csv('path/input.csv')
print(data)
```

Operations using NumPy

NumPy is a Python package which stands for 'Numerical Python'. It is a library consisting of multidimensional array objects and a collection of routines for processing of array.

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.

Key Features of Pandas

- Fast and efficient DataFrame object with default and customized indexing.
- Tools for loading data into in-memory data objects from different file formats.
- Data alignment and integrated handling of missing data.
- Reshaping and pivoting of date sets.
- Label-based slicing, indexing and subsetting of large data sets.
- Columns from a data structure can be deleted or inserted.
- Group by data for aggregation and transformations.
- High performance merging and joining of data.
- Time Series functionality.

Python Flask Tutorial



Flask Tutorial provides the basic and advanced concepts of the Python Flask framework. Our Flask tutorial is designed for beginners and professionals.

Flask is a web framework that provides libraries to build lightweight web applications in python. It is developed by **Armin Ronacher** who leads an international group of python enthusiasts (POCCO).

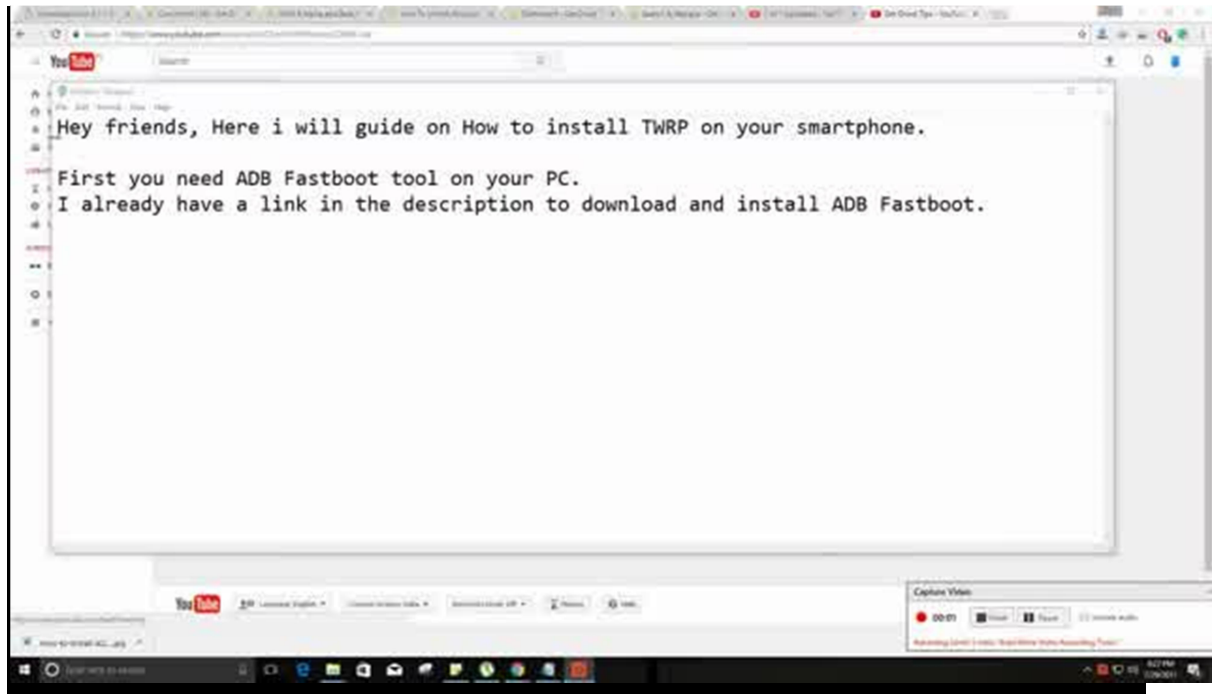
What is Flask?

Flask is a web framework that provides libraries to build lightweight web applications in python. It is developed by **Armin Ronacher** who leads an international group of python enthusiasts (POCCO). It is based on WSGI toolkit and jinja2 template engine. Flask is considered as a micro framework.

What is WSGI?

It is an acronym for web server gateway interface which is a standard for python web application development.

It is considered as the specification for the universal interface between the web server and web application.



What is Jinja2?

Jinja2 is a web template engine which combines a template with a certain data source to render the dynamic web pages.

Flask Environment Setup

To install flask on the system, we need to have python 2.7 or higher installed on our system. However, we suggest using python 3 for the development in the flask.

Install virtual environment (virtualenv)

virtualenv is considered as the virtual python environment builder which is used to create the multiple python virtual environment side by side. It can be installed by using the following command.

1. `$ pip install virtualenv`

Once it is installed, we can create the new virtual environment into a folder as given below.

1. `$ mkdir new`
2. `$ cd new`
3. `$ virtualenv venv`

To activate the corresponding environment, use the following command on the Linux operating system.

1. `$ venv/bin/activate`

On windows, use the following command.

1. `$ venv\scripts\activate`

We can now install the flask by using the following command.

1. `$ pip install flask`

However, we can install the flask using the above command without creating the virtual environment.

Pycharm Tutorial

PyCharm is the most popular IDE for Python, and includes great features such as excellent code completion and inspection with advanced debugger and support for web programming and various frameworks. PyCharm is

created by Czech company, Jet brains which focusses on creating integrated development environment for various web development languages like JavaScript and PHP.

Audience

This tutorial has been prepared for Python developers who focus on using IDE with complete package of running, debugging and creating projects in various python frameworks. Also, interested learners with a basic knowledge of any IDE can take up this tutorial.

Prerequisites

Before proceeding with this tutorial, you need a basic knowledge of any integrated development environment of Python like Sublime Text or most popular IDE like NetBeans. If you are a beginner, we suggest you to go through tutorials related to these topics first before proceeding further on this tutorial.

PyCharm is the most popular IDE used for Python scripting language. This chapter will give you an introduction to PyCharm and explains its features.

PyCharm offers some of the best features to its users and developers in the following aspects –

- Code completion and inspection
- Advanced debugging
- Support for web programming and frameworks such as Django and Flask

Features of PyCharm

Besides, a developer will find PyCharm comfortable to work with because of the features mentioned below –

Code Completion

PyCharm enables smoother code completion whether it is for built in or for an external package.

SQLAlchemy as Debugger

You can set a breakpoint, pause in the debugger and can see the SQL representation of the user expression for SQL Language code.

Git Visualization in Editor

When coding in Python, queries are normal for a developer. You can check the last commit easily in PyCharm as it has the blue sections that can define the difference between the last commit and the current one.

Code Coverage in Editor

You can run .py files outside PyCharm Editor as well marking it as code coverage details elsewhere in the project tree, in the summary section etc.

Package Management

All the installed packages are displayed with proper visual representation. This includes list of installed packages and the ability to search and add new packages.

Local History

Local History is always keeping track of the changes in a way that complements like Git.

Local history in PyCharm gives complete details of what is needed to rollback and what is to be added.

IMPLEMENTATION

The basic levels of Testing:

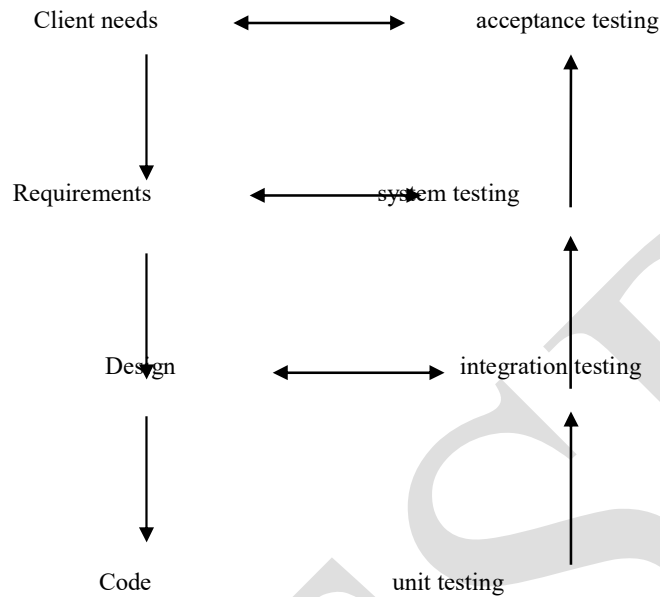


Figure: Levels of Testing

Code testing:

This examines the logic of the program. For example, the logic for updating various sample data and with the sample files and directories were tested and verified.

Specification Testing:

Executing this specification starting what the program should do and how it should performed under various conditions. Test cases for various situation and combination of conditions in all the modules are tested.

Unit testing:

We unit-test each module and integrate it with the system. Unit testing verifies the module's smallest software design unit. Also called module testing. System module testing is done individually. This testing occurs while programming. Tests show that each module produces the required result. Field validation is also done. Validation checks variation in user input to determine data veracity. System errors are simple to see. Each Module may be checked using two strategies:

1. Black Box Testing
2. White Box Testing

BLACK BOX TESTING

What is Black Box Testing?

Black box testing is a software testing techniques in which **functionality of the software under test (SUT) is tested without looking at the internal code structure**, implementation details and knowledge of internal paths of the software. This type of testing is based entirely on the software requirements and specifications.

In Black Box Testing we just focus on inputs and output of the software system without bothering about internal knowledge of the software program.

Test Case ID #1		Test Case Description - Validations in Registration Form		
S#	Prerequisites	S#	Test Data Requirement	
1	User should be Registered	1	Data should be valid	
Test Condition				
Entering data in registration form				
Step #	Step Details	Expected Results	Actual Results	Pass/Fail/Not Executed/Suspended
1	User gives First and Last Name	Pop showing email verification message	Enter valid email/password	Fail
2	Submitting the form without entering any details	Pop showing email verification message	Enter email /password	Fail
3	User enters invalid format of email id	Pop showing email verification message	Enter valid email id	Fail
4	User enters a phone number with < 10 digits	Pop showing email verification message	Enter valid phone number	Fail
5	Entering valid username and password	Pop showing email verification message	Pop showing email verification message	Pass

Table 1: Registration test case



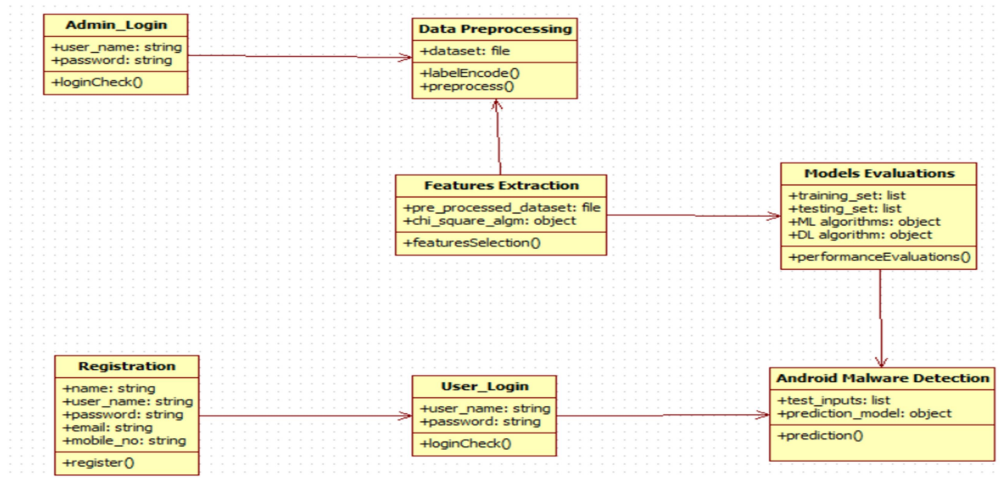
WHITE BOX TESTING

White Box Testing is the testing of a software solution's internal coding and infrastructure. It focuses primarily on strengthening security, the flow of inputs and outputs through the application, and improving design and usability. White box testing is also known as **clear, open, structural, and glass box testing**.

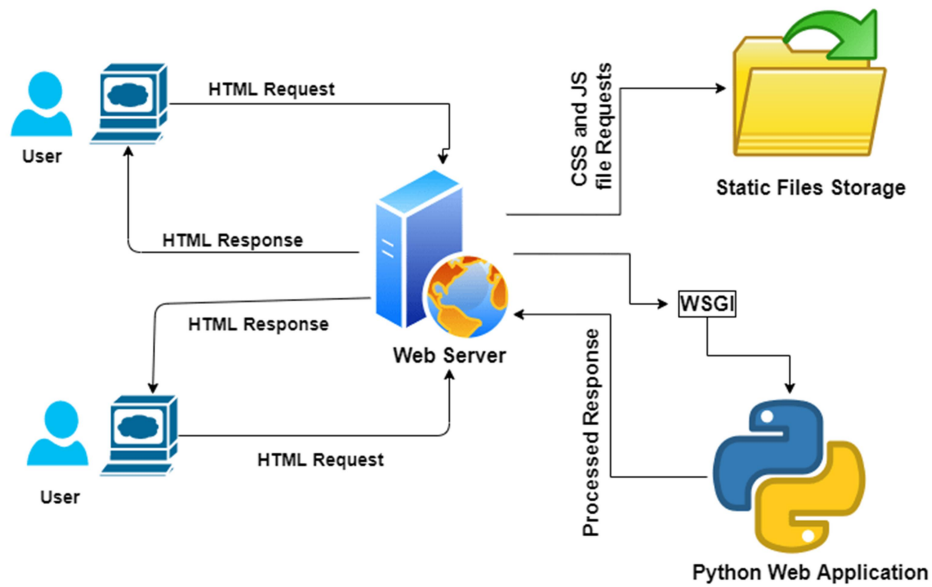
Test Case ID #2		Test Case Description - Validations in Login Form		
S#	Prerequisites	S#	Test Data Requirement	
1	User should have an email id	1	Data should be valid	
Test Condition				
Entering data in login form				
Step #	Step Details	Expected Results	Actual Results	Pass/Fail/Not Executed/Suspended
1	User gives aemail or password of <6 characters	User logged in	Enter valid email/password	Fail
2	Submitting the form without entering any details	User logged in	Enter email /password	Fail
3	User enters wrong Email and (or) password	User logged in	Enter correct email /password	Fail

Table 2 : Login test case

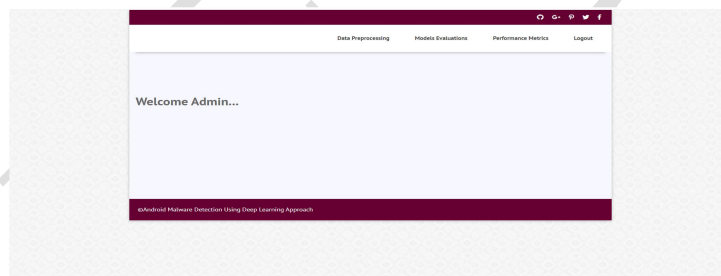
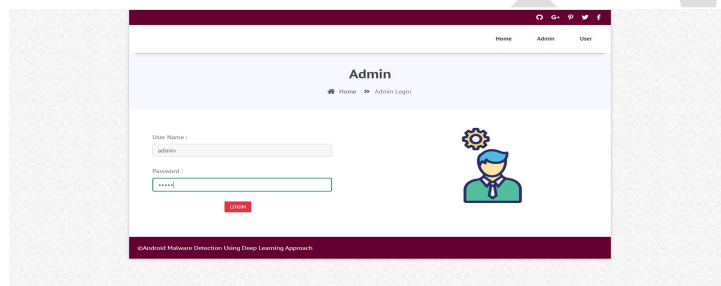
Class Diagram:



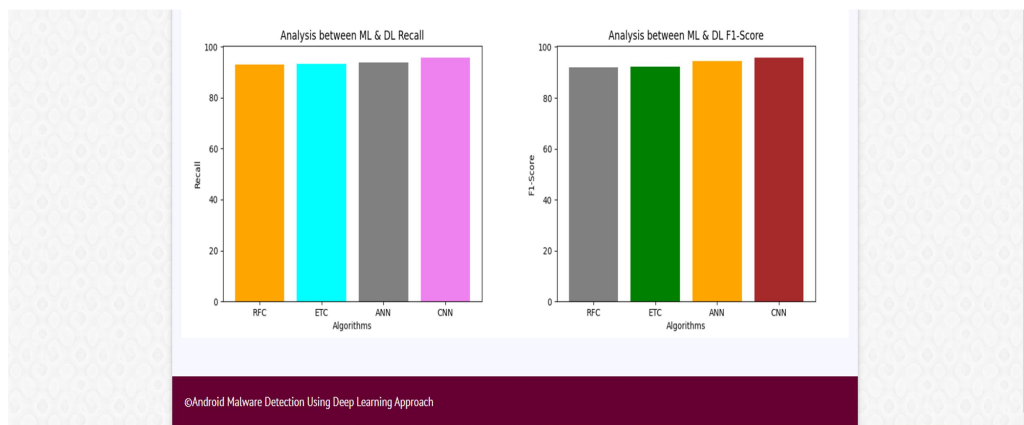
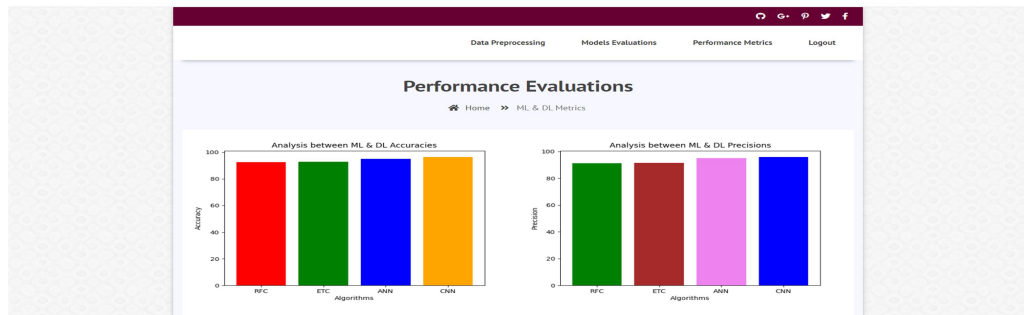
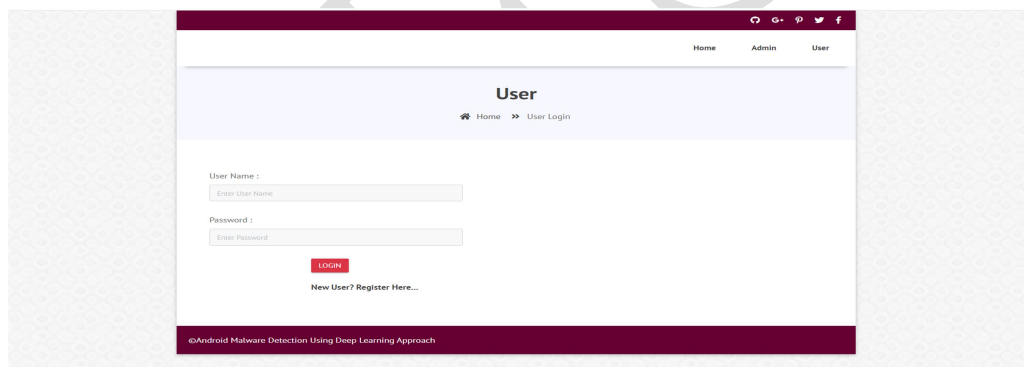
Deployment Diagram:



OUTPUT SCREENS



ML & DL Evaluations				
AdminHome Models Evaluations				
Techniques	Accuracy	Precision	Recall	F1 Score
RFC	92.6357894736842	91.31057178116002	92.86448427400217	91.9940887174542
ETC	92.89473684210526	91.5625328152893	93.24811598755716	92.2828711329738
ANN	95.08771929824562	95.18729408539221	93.8779412618359	94.47368421052632
ONN	96.84210526315789	97.03660798947831	95.97432503767854	96.472521670745952

User

Home >> User Login

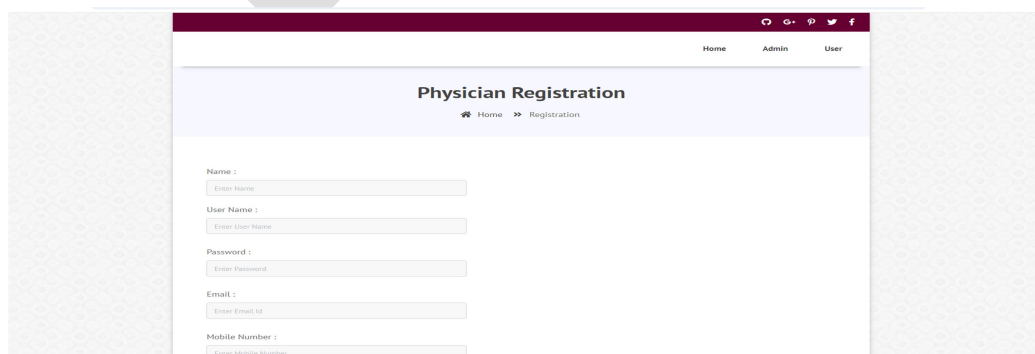
User Name :

Password :

[LOGIN](#)

[New User? Register Here...](#)

©Android Malware Detection Using Deep Learning Approach



Physician Registration

Home >> Registration

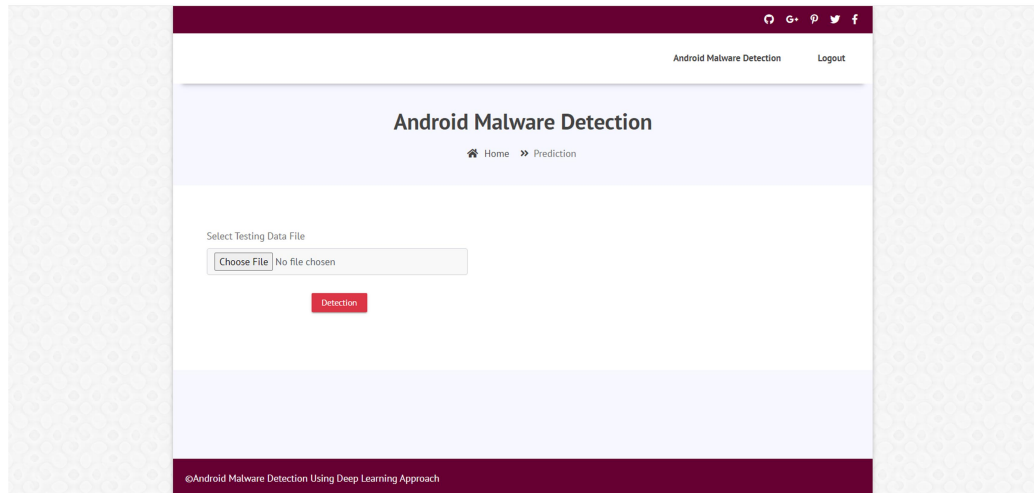
Name :

User Name :

Password :

Email :

Mobile Number :



CONCLUSION

We have effectively resolved the current issue and developed our The primary contribution of the work is the chi-square algorithm's reduction of the feature dimension to less than half of the original feature set, which enables machine learning classifiers to be trained with less complexity while retaining accuracy in malware classification. The RandomForestClassifier and Extra Trees Classifier ensemble learning techniques are trained utilizing the optimum feature set that is acquired by the application of the chi-square algorithm. Additionally, neural networks and deep learning techniques are being used in this system to increase detection accuracy. It is found that while operating on a much smaller feature dimension, a respectable classification accuracy of more than the prior % is maintained, which lowers the classifiers' training time complexity.

References:

1. H. Rathore, A. Nandanwar, S. K. Sahay, and M. Sewak, "Adversarial superiority in Android malware detection: Lessons from reinforcement learning based evasion attacks and defenses," *Forensic Sci. Int., Digit. Invest.*, vol. 44, Mar. 2023, Art. no. 301511.
2. H. Wang, W. Zhang, and H. He, "You are what the permissions told me! Android malware detection based on hybrid tactics," *J. Inf. Secur. Appl.*, vol. 66, May 2022, Art. no. 103159.
3. A. Albakri, F. Alhayan, N. Alturki, S. Ahamed, and S. Shamsudheen, "Metaheuristics with deep learning model for cybersecurity and Android malware detection and classification," *Appl. Sci.*, vol. 13, no. 4, p. 2172, Feb. 2023.
4. M. Ibrahim, B. Issa, and M. B. Jasser, "A method for automatic Android malware detection based on static analysis and deep learning," *IEEE Access*, vol. 10, pp. 117334–117352, 2022
5. Ijteba Sultana, Dr. Mohd Abdul Bari ,Dr. Sanjay," *Routing Performance Analysis of Infrastructure-less Wireless Networks with Intermediate Bottleneck Nodes*", *International Journal of Intelligent*

Systems and Applications in Engineering, ISSN no: 2147-6799 IJISAE, Vol 12 issue 3, 2024, Nov 2023

6. Md. Zainabuddin, "*Wearable sensor-based edge computing framework for cardiac arrhythmia detection and acute stroke prediction*", Journal of Sensor, Volume 2023.
7. Md. Zainabuddin, "*Security Enhancement in Data Propagation for Wireless Network*", Journal of Sensor, ISSN: 2237-0722 Vol. 11 No. 4 (2021).
8. Dr MD Zainabuddin, "*CLUSTER BASED MOBILITY MANAGEMENT ALGORITHMS FOR WIRELESS MESH NETWORKS*", Journal of Research Administration, ISSN:1539-1590 | E-ISSN:2573-7104, Vol. 5 No. 2, (2023)
9. Vaishnavi Lakadaram, "Content Management of Website Using Full Stack Technologies", Industrial Engineering Journal, ISSN: 0970-2555 Volume 15 Issue 11 October 2022
10. Dr. Mohammed Abdul Bari, Arul Raj Natraj Rajgopal, Dr.P. Swetha, "*Analysing AWS DevOps CI/CD Serverless Pipeline Lambda Function's Throughput in Relation to Other Solution*", International Journal of Intelligent Systems and Applications in Engineering, IJISAE, ISSN:2147-6799, Nov 2023, 12(4s), 519–526
11. Ijteba Sultana, Mohd Abdul Bari and Sanjay, "*Impact of Intermediate per Nodes on the QoS Provision in Wireless Infrastructure less Networks*", Journal of Physics: Conference Series, Conf. Ser. 1998 012029, CONSILIO Aug 2021
12. M.A.Bari, Sunjay Kalkal, Shahanawaj Ahamad, "*A Comparative Study and Performance Analysis of Routing Algorithms*", in 3rd International Conference ICCIDM, Springer - 978- 981-10-3874-7_3 Dec (2016)
13. Mohammed Rahmat Ali, "*BIOMETRIC: AN e-AUTHENTICATION SYSTEM TRENDS AND FUTURE APPLICATION*", International Journal of Scientific Research in Engineering (IJSRE), Volume 1, Issue 7, July 2017
14. Mohammed Rahmat Ali, "*BYOD.... A systematic approach for analyzing and visualizing the type of data and information breaches with cyber security*", NEUROQUANTOLOGY, Volume 20, Issue 15, November 2022
15. Mohammed Rahmat Ali, "*Computer Forensics -An Introduction of New Face to the Digital World*", International Journal on Recent and Innovation Trends in Computing and Communication, ISSN: 2321-8169-453 – 456, Volume: 5 Issue: 7
16. Mohammed Rahmat Ali, "*Digital Forensics and Artificial Intelligence ...A Study*", International Journal of Innovative Science and Research Technology, ISSN:2456-2165, Volume: 5 Issue:12.
17. Mohammed Rahmat Ali, "*Usage of Technology in Small and Medium Scale Business*", International Journal of Advanced Research in Science & Technology (IJARST), ISSN:2581-9429, Volume: 7 Issue:1, July 2020.
18. Mohammed Rahmat Ali, "*Internet of Things (IOT) Basics - An Introduction to the New Digital World*", International Journal on Recent and Innovation Trends in Computing and Communication, ISSN: 2321-8169-32-36, Volume: 5 Issue: 10

19. Mohammed Rahmat Ali, Internet of things (IOT) and information retrieval: an introduction, International Journal of Engineering and Innovative Technology (IJEIT), ISSN: 2277-3754, Volume: 7 Issue: 4, October 2017.
20. Mohammed Rahmat Ali, How Internet of Things (IOT) Will Affect the Future - A Study, International Journal on Future Revolution in Computer Science & Communication Engineering, ISSN: 2454-424874 – 77, Volume: 3 Issue: 10, October 2017.
21. Mohammed Rahmat Ali, ECO Friendly Advancements in computer Science Engineering and Technology, International Journal on Scientific Research in Engineering(IJSRE), Volume: 1 Issue: 1, January 2017
22. Ijteba Sultana, Dr. Mohd Abdul Bari ,Dr. Sanjay, “*Routing Quality of Service for Multipath Manets, International Journal of Intelligent Systems and Applications in Engineering*”, JISAE, ISSN:2147-6799, 2024, 12(5s), 08–16;
23. Mr. Pathan Ahmed Khan, Dr. M.A Bari,: Impact Of Emergence With Robotics At Educational Institution And Emerging Challenges”, International Journal of Multidisciplinary Engineering in Current Research(IJMEC), ISSN: 2456-4265, Volume 6, Issue 12, December 2021,Page 43-46
24. Shahanawaj Ahamad, Mohammed Abdul Bari, Big Data Processing Model for Smart City Design: A Systematic Review “, VOL 2021: ISSUE 08 IS SN : 0011-9342 ;Design Engineering (Toronto) Elsevier SCI Oct : 021
25. Syed Shehriyar Ali, Mohammed Sarfaraz Shaikh, Syed Safi Uddin, Dr. Mohammed Abdul Bari, “Saas Product Comparison and Reviews Using Nlp”, Journal of Engineering Science (JES), ISSN NO:0377-9254, Vol 13, Issue 05, MAY/2022
26. Mohammed Abdul Bari, Shahanawaj Ahamad, Mohammed Rahmat Ali,” Smartphone Security and Protection Practices”, International Journal of Engineering and Applied Computer Science (IJEACS) ; ISBN: 9798799755577 Volume: 03, Issue: 01, December 2021 (International Journal,U K) Pages 1-6
27. M.A.Bari& Shahanawaj Ahamad, “Managing Knowledge in Development of Agile Software”, in International Journal of Advanced Computer Science & Applications (IJACSA), ISSN: 2156-5570, Vol: 2, No: 4, pp: 72-76, New York, U.S.A., April 2011