Journal of Data Acquisition and Processing
ISSN: 1004-9037
https://sjcjycl.cn/
DOI: 10.5281/zenodo.777416

FEATURE EXTRACTION AND RANKING THROUGH PCA FOR ANDROID MALWARE DETECTION- AN ANALYSIS

J pavithra[1] and S Selvakumara Samy[2]
Pavithra first author
Selvakumara samy second author
Corresponding author
Department of Computational Intelligence[1]
and Department of Computational Intelligence[2]
SRM Institute of Science and Technology
Kattankulathur, India
Vels Institute of science technology and advanced studies
Pallavaram chennai
E-mail- pj1089@srmist.edu.in, selvakus1@srmist.edu.in

Abstract
Malicious software has increased massively on smartphones, tablets, etc. and has become popular all over the world in recent times. Multiple static and dynamic malware detection methods were proposed efficiently to detect Android malware. Permission-based/feature-based malware detection through API feature selection is one of the most popular techniques for detection. However, ignoring specific features or securing unrelated features may cause incredulity for classification algorithms with respect to classification time and improvement of accuracy. In order to face this challenge, different feature reduction tools were suggested by researchers to improve the accuracy of detection without considering the time. In this work, feature extraction, and feature ranking, are performed through PCA analysis and classification of apk's attained better accuracy with less prediction time. This method involves the combination of malware and benign applications (19611 entries) and the list of ranked features. Among the 79 features in the dataset, the feature visualization of 11 features is shown for benign and malicious applications. Here, the random forest algorithm performs the classification with parameters like precision, f1-score, and recall. This model observed a 0.90 f1-score while accuracy is 96% and the weighted accuracy is 94%.
Keywords: Malicious software, android malware, API features, classification, Feature Ranking, Feature Extraction, PCA.

1 Introduction

With the rapid growth of Internet consumption, malware has become one of the malicious threats. Malware is code that performs malicious activities but pretends to software. The malware aims to break the computer system without the owner's knowledge. It may also be categorized as file viruses or as autonomous malware. Malware can also enter in the form of applications or software such as worms, adware, spyware, or Trojans [1].
Further, the skill needed to develop malware has been reduced because of the widespread wicked tools which are now available open on the Internet. There exist malware developers

who gain money by exposing the malware to someone who is not an expert in it [2]. The reason behind the spread of malware may be due to a lack of knowledge ranging from unknown to known. The users provide access to various types of private information through mobile applications and downloading unauthorized applications [3].

Many applications ofandroid in the repository were identified as malicious,and the distribution of malware is often influenced by the policies of the market suppliers. For example, the Apple App Store has recently implemented a policy that strict enrollment and digital certification is to be issued by Apple Company for any applications to be published and reduces the scope to upload their software. Its competent application store like Google Play Market allows developers to upload their software without prior verification. In the event of identified malicious activity, the applications are then withdrawn from the market. The Android operating system (new one) requires the malicious device to be removed remotely from the device by the users. The Apple App Store also uses a similar removal mechanism [4]. The malicious software detection policy requires that malware applications be detected early on the downloading stage. The extraction of features helps to set the threshold based on the average value. But in need of a high priority feature, PCA reduces the dimensionality to select the appropriate features. Once the feature extraction and selection process is done, then the ranking and classification report that the applications are malware or benign based on the features ranked on the database.

This paper is subdivided into six main sections. Section 1 describes the malware behaviors, effects of malware, app creation policies, the importance of feature analysis, and so on. In section 2, the survey describes the malware analyzing methods and the feature selection categories. The description of the dataset collection is discussed in section 3. Section 4 discusses the importance of feature selection, ranking a feature, through PCA. In section 5, the classification, testing, and the evaluation metrics at the learning phase. Finally, the conclusion of the paper is discussed in Section 6.

2 Problem Description

Previous research has discussed the problem of malware analysis and classification based on how malware is structured. The structure helps to analyze, implement the model to detect, and predict with better accuracy. In this paper, two types of surveys have been performed for different analyzing methods and feature selection categories.

2.1 Malware analyzing methods

The main goal of malware analysis is to collect information from the malware sample that will aid in the response to a malware occurrence. To discover the characteristics of the malware's network, samples can be used to detect similar infections through access control. Each application follows a code structure. The code structure is characterized as static, dynamic, and hybrid. From [1][3][5][6][7][8][9], it is observed that the features were classified into four types, such as static, dynamic, hybrid, and application metadata. In all these categories, the discussion has been done with recent datasets and utilized evaluation metrics for analysis.

Dynamic analysis represents the reflection of the postinstallation behavior on mobile devices, which includes the behavior in the operating system and on the network [3].

The hybrid analysis is the combination of both static and dynamic approaches. The applications are examined from a variety of angles which leads to the extraction of most specific features. In this paper, hybrid analysis is used as the dataset used for training contains static applications and the testing dataset contains real-time applications which are dynamic [9].
The application metadata is made up of the current collection of information about Android applications, such as information from Google Play.

2.2 Analysis based on feature selection categories:

Predicting malware is the most important step in computer systems. The cryptic hackers are using them to attract into devices through the features mentioned in the vulnerable repositories. Hence, it is mandatory to predict the malware at an initial step to avoid loss in the future. When building a predictive model, feature selection is the process of reducing the number of input variables thereby lowering the computational costs of modeling. In some situations, it may also help to increase the model's performance. The feature selection involves forward selection, backward selection, and recursive feature selection. In this paper, a recursive method was taken for feature selection which mainly focuses on APK files, and permission features because it helps to detect the malware with better accuracy.
For android applications, permission-based features enforce a unique device identifier for each program running as a licensed operating system. A permission-based access control technique that prevents apps from accessing confidential data on mobile devices. As an outcome, before attaching to a system resource, programs would apply for corresponding permissions, and the system monitors the performance of programs. Because of the mechanism's limitations, regular malicious activities like sending short messages or invoking cameras require more private data. According to statistics, as a result, they tend to use more sensitive permissions than benign applications. For example, in [6] weimplemented a permission-based assessment of Google Play Apps through the removal of permissions from important AndroidManifest.xml files. The performance was compared in terms of true positive, false positive, accuracy, and retrieval rates concerning the use of the information gain feature selection method to reduce the feature set's size and apply the J48 Decision Tree, CART, and the Random Forest classification algorithms to the reduced data set. The analysis based on permission-based can be done by a two-step process. A set of feature selection methods to reduce the size of the dimension of the features based on permissions [7]. Second, the reduced dataset was used as input for the algorithm classification.
APK features for the android operating system use the complete format files from the android package which includes program source codes like.DEX files, .manifest files, resources, certificates, assets, and so on. The main advantage of the Android operating system is that it allows the user to manually download the APK after turning on the unknown source option, which helps to find trusted apps rather than the untrusted ones. These APK features were included in frameworks, algorithms, and so on to detect the malware. The Perceptron Training Subroutine algorithms framework suggested reducing the risk of false positives by separating benign or malicious code files using three datasets: training, testing, and a "scale-up" dataset [10]. The classification process performed through the data mining algorithm in the data set consists of 100,000 malware files and 16,000 benign apps and achieved 92% precision [11].

Seven key features were provided for the presentation of classifying benign or malware files. The features have been identified by visualizing that various parts of the features of the Portable Executable [PE ()] files can associate APK's as benign or malware, by comparing each other less than the class files [12]. Such features are implemented as machine learning algorithms to help classify malware, leading to the appropriate implementation of this classification in the antivirus framework to improve the detection rate.

Structural classification plays a major role in finding the malware. The structure is made of class, subclass, and groups. Every file follows structures, similar patterns, and a design format. Based on the structure, algorithms can be made with the help of structures. Based on the structures and algorithms, the malware structure can be accessed. The authors provided a malware structure that is fully based on the malware information. In addition to the decision tree, the data set consisted of 179 malware and here algorithms such as KNN, SVM were used for classification [13]. When the features were extracted with each cell graph, the authors concluded that the two malware showed similarities and that is done by applying the distance metering learning to the group and sorting it based on family, and keeping it at the boundary. Then, they finally put together and used an ensemble method to classify the malware and merged it into that family. However, there are many researchers who have selected network security features, such as pin certificates, access permission features, etc. They are used for the evaluation of intrusion detection based on the feature selection method [14, 15].

The selected features used the neural network to detect P2P Bot [16]. After facing many difficulties, information gain techniques were applied instead of selecting a feature that uses evolving clustering method for detecting malware [17]. A method was proposed that considers several types of static features and chooses suitable features that affect malware detection [8]. They performed classification based on the symmetric pattern. The collection of android Urls were merged as a single feature, named remove URL_score. In addition, the feature space and memory space have been reduced,and 91% accuracy on SVM for the Drebin dataset.

Most commonly 29580 binary executables" for the objectives of training the classifiers were included in the techniques used in this paper [9]. To produce high-performance results, they have implemented several Data Mining algorithms such as KNN, SMO, and NB. The benefits of one data mining algorithm can be studied and assessed correctly to detect zero-day malware. The result showed 96 percent high precision. From the literature survey, it is observed that the feature ranking based on the importance and the visualization of whole behavior will help to predict with better accuracy. With these observations, the proposed method provides the importance of feature ranking. In the following sections, the dataset collection, feature ranking and selection, and its importance, new feature identification are described.

3 Dataset Description

The data set was collected from websites such as virus total, a repository such as system 32 malware, and DLLs (Dynamic Link Libraries) [18]. In total, there is a collection of 19612 entries in the malware dataset to train the model. The test dataset contains 18 entries, which is a combination of android applications like Spotify, WinRAR, Skype, and so on. Among 79 features, 11 features have been extracted in the dataset and visualized for both the feature

behavior of malware and benign application. The importance of the feature is visualized to have a better understanding.

4 Methodology

The proposed method comprises three phases: Feature Extraction, Feature Engineering and Selection using PCA, Identification of new feature, classification and testing as shown in Figure 1.

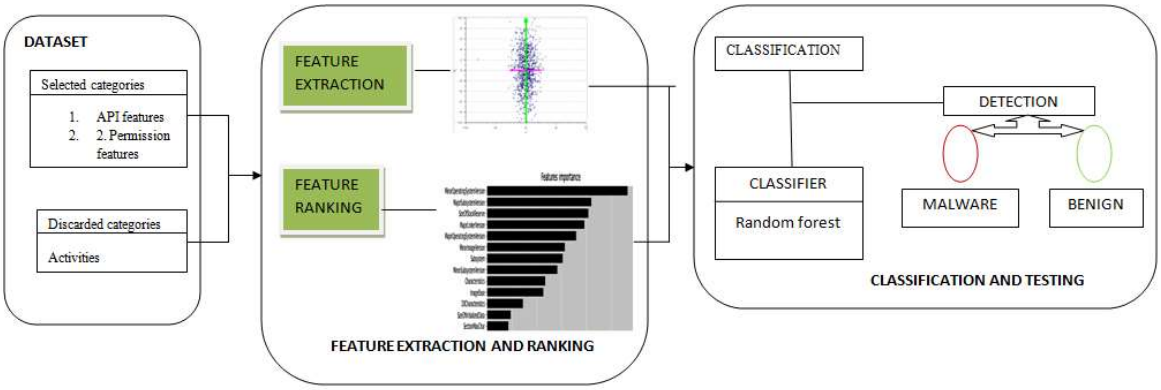

Figure.1: Architecture Of Proposed Work

The data set comprises two selected categories and one discarded category of characteristics. Once loaded, the features were extracted based on the PCA analysis for the selected categories and ranked in the feature ranking process to find the importance of the features. The addition of new features was calculated in the PCA and stored into the feed as an update. The ranked features were stored as data feeds for the classification process. It compares the apk features with the ranked features to declare that the application is malware or benign. The suitable classifier was chosen with the help of the better detection results obtained. Based on the results, the random forest provided better accuracy for detection and prediction [19]. In the learning phase, the predictive model performs the classification and testing process with the help of a random forest classifier for better results.

5 Feature extraction

In this phase, the data set with 79 features was loaded, of which 11 features were extracted for all the entries in the data set (19612). The extracted features comprised API and permission features. With the help of Principal Component Analysis (PCA), the features have been extracted and listed. Here, the memory usage was 11.3+ MB for extraction. The classes distributed as benign and malware are shown in Figure 2.

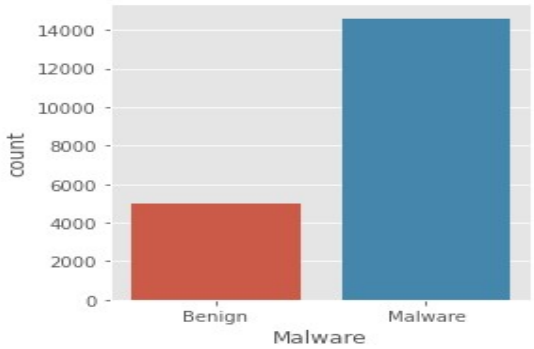

Figure.2: Number of malicious appsandmalware separated using Pca Analysis

Figure 2 shows the number of malware and benign applications separated using PC analysis. The y- axis represents the overall count (19612) and the X-axis shows the classification of benign and malware. This visualization clearly explains that 80% of the entries are malware and 20% are benign applications.

5.1 Feature extraction using PCA:

Unnecessary and excessive features can lead to a slow convergence of an ML classifier, a lower performance, and even complete failure. PCA is a linear projection that converts high-dimensional features into lower-dimensional ones. The variance of the anticipated data is maximized during the projection phase.

In data science, principal component analysis (PCA) is a popular feature extraction method. PCA works by finding the eigenvectors of a covariance matrix with the maximum eigenvalues and then projecting the data onto a new subspace with the same or fewer dimensions [20]. In reality, PCA transforms an n number of feature matrices into a new dataset with (ideally) less than n number of features. It decreases the set of features by creating new, small variables that capture a significant percentage of the data included in the unique features. The purpose of this analysis, however, is to demonstrate PCA in action rather than to explain the concept of PCA, which has been done quite well elsewhere.

Denote a set of M features A = {A1, A2,...,AN }, where Ai = [Ai(1), Ai(2),...,Ai(d)]$^T \in R^d$ and d is the number of features. PCA linearly transforms each Aisample into a new one $B_i \in R^d$.

$$B_i = U^T A_i \text{---------- (1)}$$

where U is a d × d orthogonal matrix. The i-th column u_i of U is the i-th eigenvector of the sample covariance matrix.

5.2 New Feature Identification:

In the future, a dynamic application appears for the installation and plenty of new features exist. It is difficult to understand and to add the new feature to the feed for referencing of the feature ranks.

$$C = \frac{1}{d}\sum_{i=1}^{d} A_i A_i^T \quad \text{.....(2)}$$

In summary, PCA first calculates the covariance matrix of A and denotes it as S. Thencalculate the eigenvalues and eigenvectors of S, and denote them as λ1, λ2,...,λd and u1, u2,...,ud. Finally, we can calculate the new feature by using equation(2). In this result, feature extraction and ranking of features are described in the next section.

6 Feature Engineering and Selection

Feature selection is the process of minimizing the number of input variables in a predictive model. The most comprehensive way to reduce modeling costs is by reducing the number of input variables. It may also help to improve the model's performance in some cases. Feature-ranking techniques that give a ranking to input features based on how effective they are at predicting a target variable are known as feature importance.

Statistical correlation scores, coefficients measured as part of linear models, decision trees, and permutation importance scores are some of the most common forms and sources of feature importance scores. In this paper, a random forest regression model is used for the feature importance selection.

In this phase, the ranking process is based on the importance of assessing the weightage on the application to check whether it is malware or benign. The current methods for detecting Android malware can be defined based on the space size reduction technique of a particular feature. There are different ways for reducing the feature space size as follows: basically, small datasets result in small feature space without the need to reduce further. In a larger dataset, the size of the feature space has to be reduced with the help of machine learning algorithms among which random forest is used in the current research due to better accuracy [19]. To fine-tune the results, it is needed to focus on a small number of feature categories, such as Android permissions features as mentioned in figure I.

There are several selection techniques for feature ranking and feature selection such as univariate statistical tests (chi-square), community-based detection, feature evaluation based on correlation, detailed information, gain ratio method, and recursive feature elimination. Among these, community-based detection can be adopted for feature selection [8]. The intensity of the relationship between an individual characteristic and the response variable is measured using these feature classification and feature selection methods. As a result, each function is assigned a single score that reflects this relationship. The top features are recorded and the feature space is rated based on ranking.

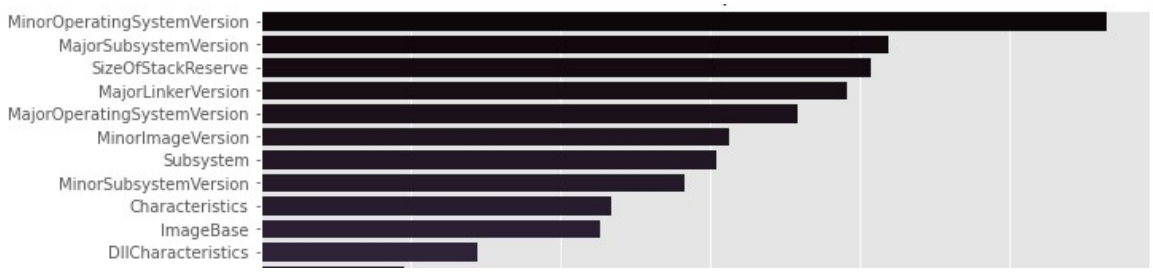

Figure.3.1 (A) High-ranking Features

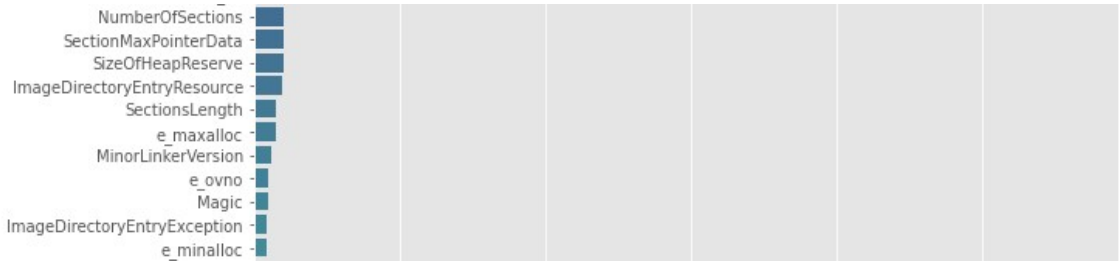

Figure.3.1(B)Low-Ranked Features

Figure 3 Importance of Importance

Among the list of 79 features, the 11 features which are highly ranked were taken for visualization as shown in figure 3. The features were ranked and visualized as descending to the top. In Figure 3.1(a), the high-ranked features that suit best for detecting the malware are shown. In contrast figure 3.1(b) shows the low ranked features that suit the least match for the detection of malware. For further reduction of the space size of the feature, a random forest model is implemented which discards the activity categories such as name, machine, timestamp, etc.

7 Learning Phase

Classification and Testing

In the proposed work, a random forest classifier file is used to classify the data. The pythonclassifier file (malwarecl.py) is dropped into the model after the precision results. The features can be secured by the classifier. After the classifier and features have been dropped, the next step is to decide if the file is benign or malicious. The parameters used are f1-score, precision, re-call for malware and benign applications, and the support files are mentioned. The main task is to determine whether a file is benign or malicious from some unknown execution file. It is important to delete the irrelevant features of the given execution file to test the model on an invisible file.

For the prediction and detection results are higher on random forest, It is confirmed that random forest achieved 94% accuracy and provided better test results [19]. Based on the results, the decision tree also has achieved the next better result, 93% accuracy on detection. With the help of the accuracy results, the decision tree classifier was taken for the comparison of test results.

Classifier	Applications	Re-call	F1-score	Precision	Support
Decision tree	Malware	0.92	0.90	0.91	1004
	Benign	0.90	0.91	0.91	2920
Random Forest	Malware	0.94	0.94	0.93	1004
	Benign	1.00	0.94	0.93	2920

Table.1 classification report for applications

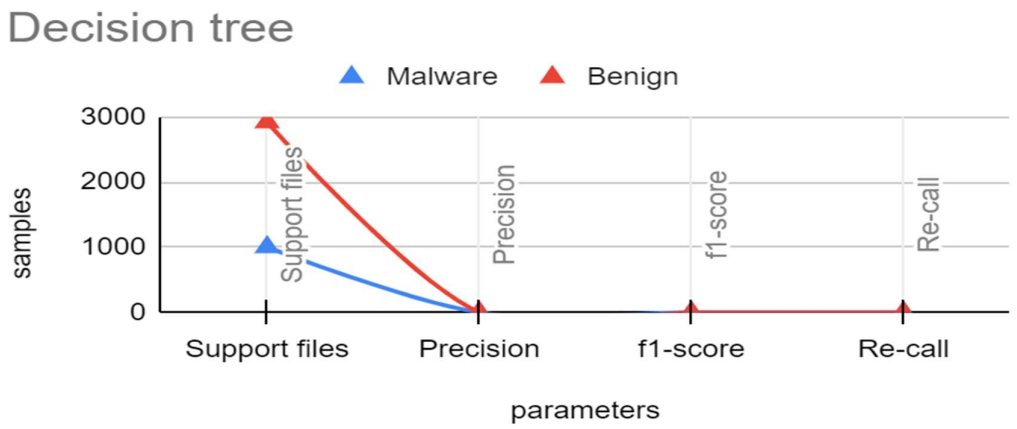

Figure.4 (a) Classification based on Applications (Decision Tree)

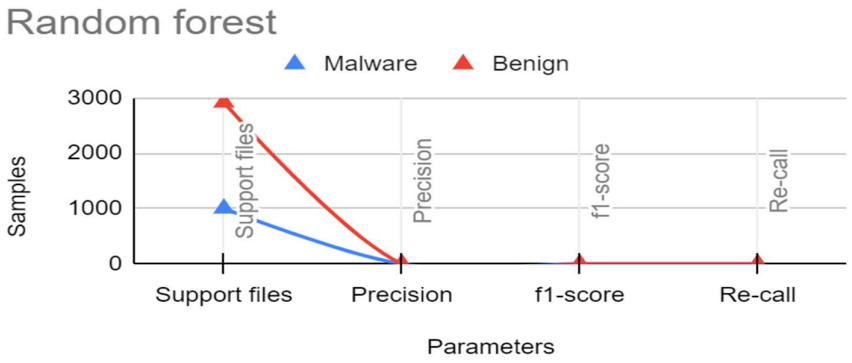

Figure.4 (b) Classification based on Applications (Random Forest)

The classification report for both the malware and benign applications is described in Table 1 and the performance analysis is plotted in figure 4. The x-axis refers to the samples and parameters mentioned in the y-axis. The parameter comparison focuses on higher values for the identification of classifier. Here, the analytical results shown the accuracy for prediction of malware is higher on random forest as shown in figure Table 1.The separate graphical analysis for classifiers shown in figure 4(a) and 4(b). The graphic analysis for the classification based on applications for the decision tree shows 0.90 on malware, 0.91 on Benign as f1 score and random forest achieved the f1-score as 0.94 for both the malware and benign applications,

which clearly explains that the random forest has a better value on prediction mentioned in Table 2. The benign applications have higher values (0.93 for recall, 0.94 for F1-score, and 0.91 for precision) for the 2920 support file.

Classifier		**Re-call**	**F1-score**	**Precision**	**Support**
Decision Tree	Accuracy		0.91		3930
	Mac avg	0.89	0.89	0.90	3930
	Weighted avg	0.93	0.91	0.90	3930
Random Forest	Accuracy		0.94		3930
	Mac avg	0.94	0.94	0.94	3930
	Weighted avg	0.94	0.93	0.94	3930

Table.2Accuracy And Average

Here, Table 2 describes the accuracy, macro, and weighted average of the parameters. The experimental results for both the classifiers achieved good results. The detection accuracy was better in the random forest when compared to the decision tree and it is mentioned in Table 2. The random forest classifier confirms that 94 % accuracy was obtained on f1-score for 3930 support files.

8 Evaluation Metrics

The confusion matrix consists of interpretation and prediction. The model is evaluated by measuring four parameters.

Accuracy= $\frac{(TP+TN)}{(TP+FP+FN+T\ )}$ …….. (3)

where, TP – True positive
TN- True Negative
FP- False Positive
FN- False Negative

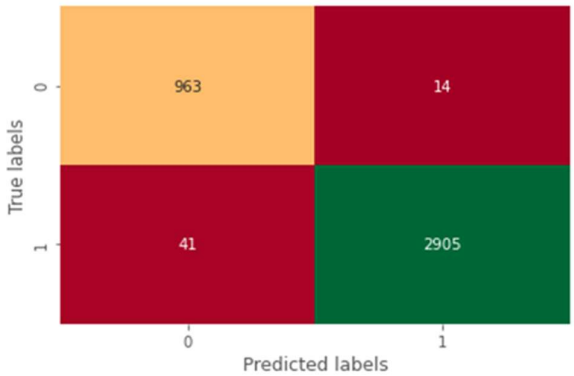

Figure.5 confusion matrix

The above figure 5, explains that, x-axis belongs to true labels and the y-axis belongs to predicted labels.

True Positive: Here, out of 3930, 963 entries were predicted as positive, and it is true. It denotes that the 963 samples were correctly classified as benign applications.
True Negative: For 3930 samples, here 2,505 entries were predicted as negative and that is true. It denotes that 2905 samples were not benign applications.
False Positive (Type 1 Error): for 3930 samples were,There were only 14 samples predicted as benign, but it is not benign.
False Negative (Type 2 Error): There were 41 samples that were predicted as malware among 3930 samples, and they aredenoted benign applications.

9 Conclusions:

The evolution of the Internet of Things (IoT) and smart devices has brought a new range of challenges to device vendors, software developers, and cyber security professionals. Unobserved Android malware is being identified and new malware is evolving. The features were extracted through PCA, and it is ranked. To counter malware in Android devices based on feature ranking, in this paper, a novel deep learning algorithm is used for algorithm detection. The comparative study was made along with the most competent algorithm like a decision tree to prove that the random forest model has better accuracy. The experimental results show that the Random forest model achieved 94% accuracy on the f1-score without any preprocessing on the data. The result is good even though the data is unbalanced. So, I found that we do not need to use any technique to rebalance it. The ranking of features and its visualization assign the weight-age to the features. It plays a major role in the comparison of features for future malware prediction.

10 Future work:

Malware has a strong tradition of posing a serious threat to computer security. The capability of classic detection approaches based on static and dynamic analysis has been hampered due to the rapid development of antidetection technologies. However, feature extraction is complex because of virus differences, which makes typical neural network applications ineffective. To solve this, the extraction and classification process was performed by extended versions of PCA and XGBoost. It is expected that the results will provide better accuracy in future.

11 References:

1. Almseidin, M., Alzubi, M., Kovacs, S. and Alkasassbeh, M., 2017, September. Evaluation of machine learning algorithms for intrusion detection system. In *2017 IEEE 15th International Symposium on Intelligent Systems and Informatics (SISY)* (pp. 000277-000282). IEEE.

2. Raman, K., 2012. Selecting features to classify malware. *InfoSec Southwest*, *2012*, pp.1-5.
3. Gaurav, A., Gupta, B.B. and Panigrahi, P.K., 2022. A comprehensive survey on machine learning approaches for malware detection in IoT-based enterprise information system. *Enterprise Information Systems*, pp.1-25.
4. Altaher, A., ALmomani, A., Anbar, M. and Ramadass, S., 2012. Malware detection based on evolving clustering method for classification. *Scientific Research and Essays*, *7*(22), pp.2031-2036.
5. Feizollah, A., Anuar, N.B., Salleh, R. and Wahab, A.W.A., 2015. A review on feature selection in mobile malware detection. *Digital investigation*, *13*, pp.22-37.
6. Zarni Aung, W.Z., 2013. Permission-based android malware detection. *International Journal of Scientific & Technology Research*, *2*(3), pp.228-234.
7. Yan, G., Brown, N. and Kong, D., 2013, July. Exploring discriminatory features for automated malware classification. In *International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment* (pp. 41-61). Springer, Berlin, Heidelberg.
8. Yan, G., Brown, N. and Kong, D., 2013, July. Exploring discriminatory features for automated malware classification. In *International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment* (pp. 41-61). Springer, Berlin, Heidelberg.
9. Ranggadara, I., Sari, Y.S., Dwiasnati, S. and Prihandi, I., 2020, March. A Review of Implementation and Obstacles in Predictive Machine Learning Model at Educational Institutions. In *Journal of Physics: Conference Series* (Vol. 1477, No. 3, p. 032019). IOP Publishing.
10. Alauthaman, M., Aslam, N., Zhang, L., Alasem, R. and Hossain, M.A., 2018. A P2P Botnet detection scheme based on decision tree and adaptive multilayer neural networks. *Neural Computing and Applications*, *29*(11), pp.991-1004.
11. Salah, A., Shalabi, E. and Khedr, W., 2020. A lightweight android malware classifier using novel feature selection methods. *Symmetry*, *12*(5), p.858.
12. Kumar, A., Abhishek, K., Shah, K., Patel, D., Jain, Y., Chheda, H. and Nerurkar, P., 2020, November. Malware detection using machine learning. In *Iberoamerican Knowledge Graphs and Semantic Web Conference* (pp. 61-71). Springer, Cham.
13. Nachenberg, C., 2011. A window into mobile device security: Examining the security approaches employed in Apple's iOS and Google's Android. *Symantec Security Response*.
14. Teufl, P., Kraxberger, S., Orthacker, C., Lackner, G., Gissing, M., Marsalek, A., Leibetseder, J. and Prevenhueber, O., 2011, May. Android market analysis with activation patterns. In *International Conference on Security and Privacy in Mobile Information and Communication Systems* (pp. 1-12). Springer, Berlin, Heidelberg.
15. Alazab, M., Venkatraman, S., Watters, P. and Alazab, M., 2010. Zero-day malware detection based on supervised learning algorithms of API call signatures..
16. Pehlivan, U., Baltaci, N., Acartürk, C. and Baykal, N., 2014, December. The analysis of feature selection methods and classification algorithms in permission based Android malware detection. In *2014 IEEE symposium on computational intelligence in cyber security (CICS)* (pp. 1-8). IEEE.

17. Kotenko, I., Izrailov, K. and Buinevich, M., 2022. Static Analysis of Information Systems for IoT Cyber Security: A Survey of Machine Learning Approaches. *Sensors*, *22*(4), p.1335.

18. https://www.idera.com/glossary/dynamic-link-libraries, 2021.

19. Pavithra, J. and Josephin, J.F., 2020, December. Analyzing Various Machine Learning Algorithms for the Classification of Malwares. In *IOP Conference Series: Materials Science and Engineering* (Vol. 993, No. 1, p. 012099). IOP Publishing.

20. Al-Kasassbeh, M., Mohammed, S., Alauthman, M. and Almomani, A., 2020. Feature selection using a machine learning to classify a malware. In *Handbook of Computer Networks and Cyber Security* (pp. 889-904). Springer, Cham.