The Detection of Android Malwares Using the Features Selection Technique

Researcher Farah Rafid Salman and Assist. Prof. Dr. Amer Abdulmajeed AbdulRahman

Computer College, Baghdad University, Baghdad / Iraq
Ms202120668@iips.icci.edu.iq
amer.abdulrahman@sc.uobaghdad.edu.iq

الكشف عن البرامج الضارة لنظام Android باستخدام تقنية اختيار الميزات

الباحثة فرح رافد سلمان
و أ.م.د.عامر عبد المجيد عبد الرحمن

كلية الحاسوب, جامعة بغداد, بغداد \ العراق

Abstract

The use of Android smartphones has been rising quickly as Internet-based services gain more popularity and develop. The Android operating system's enormous popularity has drawn malware attacks on these devices. It is difficult and ineffective to detect malware versions with features that modify their behavior in order to avoid detection by machine learning (ML) techniques. An effective feature selection plays a crucial role in detecting malware characteristics and reduces the dimensionality of a large dataset by removing the unnecessary features that are not useful and keeping those relevant features that increase classification accuracy and detection rate. This helps to solve the problems associated with malware feature detection.in this paper a malware detection model was proposed it contains three major stages: the data preprocessing stage, the feature selection stage, and finally, the classification stage. It was tested and evaluated on CICandMal2017 dataset, the number of samples used was (500000) samples. More than one steps had been applied for preparing the dataset SMOTE applied to balance the multi class dataset. The Chi-square had been applied in the feature selection stage. In the classification stage, the Random Forest algorithm had been applied. The results showed the features had values differently in importance. The features selection technique had a positive effect on the performance, where the accuracy was 89.93 % for all features, and 93.30 % when using the Chi-square method.

Keywords: Android operating system, feature selection, Chi-square, Random Forest.

المستخلص

يتزايد استخدام الهواتف الذكية التي تعمل بنظام Android بسرعة مع اكتساب الخدمات المستندة إلى الإنترنت المزيد من الشعبية والتطور. حيث أدت الشعبية الهائلة التي يتمتع بها نظام التشغيل الاندرويد إلى ظهور هجمات البرامج الضارة على هذه الأجهزة. من الصعب وغير الفعال اكتشاف إصدارات البرامج الضارة التي تحتوي على ميزات تعمل على تعديل سلوكها لتجنب اكتشافها بواسطة تقنيات التعلم الآلي. يلعب التحديد الفعال للميزات دورًا حاسمًا في اكتشاف خصائص البرامج الضارة ويقلل من أبعاد مجموعة كبيرة من البيانات عن طريق إزالة الميزات غير الضرورية والحفاظ على الميزات ذات الصلة التي تزيد من دقة التصنيف ومعدل الاكتشاف. يساعد هذا في حل المشكلات المرتبطة باكتشاف ميزات البرامج الضارة. في هذا البحث، تم اقتراح نموذج للكشف عن البرامج الضارة والذي يحتوي على ثلاث مراحل رئيسية: مرحلة المعالجة المسبقة للبيانات، ومرحلة اختيار الميزات، وأخيرًا، مرحلة التصنيف. حيث تم اختباره وتقييم هذا النموذج على مجموعة بيانات(CICandMal2017)، وبلغ عدد العينات المستخدمة (500000) عينة. استخدمت تقنية SMOTE لموازنة مجموعة البيانات هذه. وتم تطبيق مربع كاي في مرحلة اختيار الميزة. وفي مرحلة التصنيف، تم تطبيق خوارزمية الغابة العشوائية. وأظهرت النتائج أن الميزات لها قيم مختلفة في الأهمية. وكان لتقنية اختيار السمات تأثير إيجابي على الأداء، حيث بلغت الدقة 89.93% لجميع السمات، و93.30% عند استخدام طريقة مربع كاي.

الكلمات المفتاحية: نظام تشغيل الاندرويد، اختيار الميزات، مربع كاي، الغابة العشوائية.

1 - Introduction

Today, Android is the most commonly used and favored mobile operating system in the world. The fact that third-party applications may be downloaded and installed for free and offer users a variety of advantages is one of the factors leading to their popularity. Unfortunately, this flexibility to install any application created by a third party also results in an infinite stream of malware software that continue to develop and intended to harm users in a variety of ways (Alhebsi, 2022).

in the last quarter of 2022 and according to https://www.statista.com/statistics/272698/ global-market-share-held-by-mobile-operating-systems-since-2009/, Android will continue to be the most widely used mobile operating system in the world.

Systems that detect intrusion in the network, on a PC, or on a mobile device, and protect against harm from malicious software are known as intrusion detection systems. Mobile applications may enable intrusion to a mobile device. Signature-based, behavior-based, and anomaly-based are the three different types of intrusion detection approaches. In an anomaly-based intrusion detection system, the system observes whenever a mobile application differs from the usual pattern of benign application behavior. The two types of behavior pattern analysis approaches used in anomaly intrusion detection are static analysis and dynamic analysis. Without running the application, the static features of Android applications, such as permission requests, method calls, and API call sequences, are extracted from the source code using the static analysis methodology. The features of Android applications extracted using the dynamic analysis technique by running the application's code (Malik and Khatter, 2020).

ML is the study of computer programs that can learn from their own prior experience to enhance performance on a task. The two types of ML algorithms are supervised and unsupervised learning. Determining a function from labelled data is referred to as supervised learning, and it is used to perform regression on continuous and real-valued answers or to classify data for responses with few known values. Unsupervised learning involves the exploration of the data to discover certain intrinsic structures in it, such as grouping data to find similarity group called clusters (Muttoo , *et al*., 2017).

One of the most common methods for detecting these attacks is ML, which proposed in studies as a variety of methodologies. This is so that a classifier can be created using ML techniques using a (restricted) collection of training examples. Thus, while creating malware detectors, the use of examples eliminates the requirement to explicitly describe signatures. Defining signatures includes expertise and time-consuming human effort, and for some attack scenarios there are no stated rules (signatures), but examples can be easily obtained. (Senanayak, *et al*., 2021).

Pre-processing is a crucial step in converting raw data into a format that is easily understandable and usable. The majority of raw data shows incompleteness, inconsistency, and noise. Furthermore, a significant proportion of them exhibit numerous errors and problems in certain aspects of their behavior. Therefore, the technique of data pre-processing is implemented to address this issue by executing certain procedures that facilitate the transition of processed data to the subsequent stage within the framework (Zulkifli , *et al*., 2018).

The high dimension of features typically results in increased computation usage and even over-fitting when using ML to handle data. The

efficiency of Android malware detection can be enhanced through the use of feature selection, which involves the elimination of redundant and irrelevant features. According to the relationship between feature selection and classifiers, feature selection includes filter, wrapper, and embedded model, making it one of the most often used techniques. The filter is utilized widely and flexibly because it is totally independent of classifiers. Fisher, Correlation Coefficient, Info Gain, Chi-square Test, and Mutual Information are typical feature selection techniques (Xie , *et al.*, 2023).

The problem of data imbalance is one of the issues that classifier algorithms frequently face. The suggested methods for data-level enhancement include oversampling the minority class to balance the amount of data in different classes, Commonly employed is the Synthetic Minority Oversampling Technique (SMOTE) (Dehkordy and Rasoolzadegan , 2021).

The main contribution of this paper is

- Construct an android malware detection system using feature selection technique, and RF algorithm and use Synthetic Minority Oversampling (SOMTE) technique for balancing the dataset and comparing the results when using imbalance and balance dataset.

The remain of this paper was organized as follows: the related work was presented in section two. Section three showed the details of the proposed model. Section four illustrated the results and discussion of the proposed model, while the conclusion for the suggested method was presented in section five.

2 - Related Works

(Lekssays , *et al.*,2020) employed Convolutional Neural Networks (CNN) in a novel manner for the analysis of Android mobile applications. The research presents an approach for detecting malware through the utilization of malware visualization as a static analysis technique. Initially, the APK-format Android applications are transformed into gray scale images. ML model was developed to identify malicious and benign Android applications through pattern recognition, based on the fact that malware belonging to the same family has similar patterns. The present study employed a combination of self-generated datasets, consisting of scanned APK files obtained from freely available sources on the internet through public APIs, and a research dataset provided by the University of New Brunswick in Canada. The recommended approach yielded an accuracy rate of 84.9% in detecting mobile malware.

The model in this study, (Arslan, 2021) the proposed ensemble ML model aims to detect the type of malware, including ransomware, adware, scareware, or SMS malware. The model trained and evaluated using the CIC-AndMal-2017 dataset. The accuracy of malware type detection was 90.4% in the tests conducted on 486 malicious samples. The precision, recall, and F1-score metrics were observed to be 90.4%. Research has shown that the use of ensemble models can produce better results compared to conventional classification algorithms when detecting android malware types.

(Elayan, , Mustafa, 2021) The author has presented a new methodology for identifying malicious software within Android applications, applying a specific form of recurrent neural network referred to as a gated recurrent unit (GRU). The extraction of static features such as permissions and Application Programming Interface (API) calls from Android applications is a common

action. The approach is trained and evaluated using the CICAndMal2017 dataset. Based on the results of experiments, the deep learning algorithm outperforms alternative approaches with a precision rate of 98.2%.

In their study, (Zhixing , *et al.*,2021) The model receives the encrypted traffic produced by the malware as its input. by using clustering, the model eliminates unnecessary third-party traffic and preserves the integrity of the first-party traffic. The proposed approach involves the extraction of traffic features for the purpose of constructing a host-level traffic fingerprint, followed by the classification of malware using a stacking-based ensemble learning technique. The classification model was constructed utilizing the CICAndMal2017 dataset, which is accessible to the general public. The dataset effectively categorizes malware into different types. The SVM and Random Forest models are utilized in the controlled experiments. Results indicate that this model is significantly more occur in the classification of malware compared to SVM and Random Forest models. Specifically, the model achieved an accuracy rate of 96.7% under optimal conditions.

(Xue , *et al.*,2021) developed a model for detecting and categorizing Android malware using features such as encrypted traffic and the ensemble learning approach based on stacking. To obtain pure first-party traffic, the model first eliminated the third-party traffic from the traffic by clustering. 86 flow-level features were then obtained by feature engineering, and 73 statistical features that were most appropriate for classification were chosen through comparison tests. Following that, host-level traffic feature fingerprints were created by statistically combining the flow-level feature vectors. Finally, the ensemble learning method based on stacking was used to classify malware. The results of the experiment show that the classification

accuracy increased by up to 96.7% through removing third-party traffic and using appropriate traffic attributes. Although this method successfully detects Android malware, it is constrained by the samples in the dataset and is not able to effectively detect novel malware types.

(Niu, 2022). The present study applies a sensitivity coefficient-based approach for feature selection in order to identify traffic features. The detection and recognition of malicious application traffic is carried out using k-means, support vector machine (SVM), and multi-layer perceptron (MLP) methods. The experiments are conducted on the CIC-AndMal2017 dataset. The study found that the MLP approach demonstrated high efficiency in detecting and recognizing malicious application traffic with an accuracy of 99.87% when 40 features were selected. In contrast, the K-means algorithm exhibited poor performance with an accuracy of 86.75%.

3 - The Proposed Model

The proposed work contains three main phases: (preprocessing, features selection, and classification phases). Each stage contains more than one step for achieving the goal of that phase. The dataset is multiclass label which has five classes that represent the five types of android attacks are: (Benign, adware, SMS malware, Ransomware, and Scareware),(0,1,2,3,4) respectively , also the dataset is imbalance. So, the proposed model applied when the dataset is imbalance and after balancing it using SOMTE technique Figure (1) presents the structure of a proposed model.

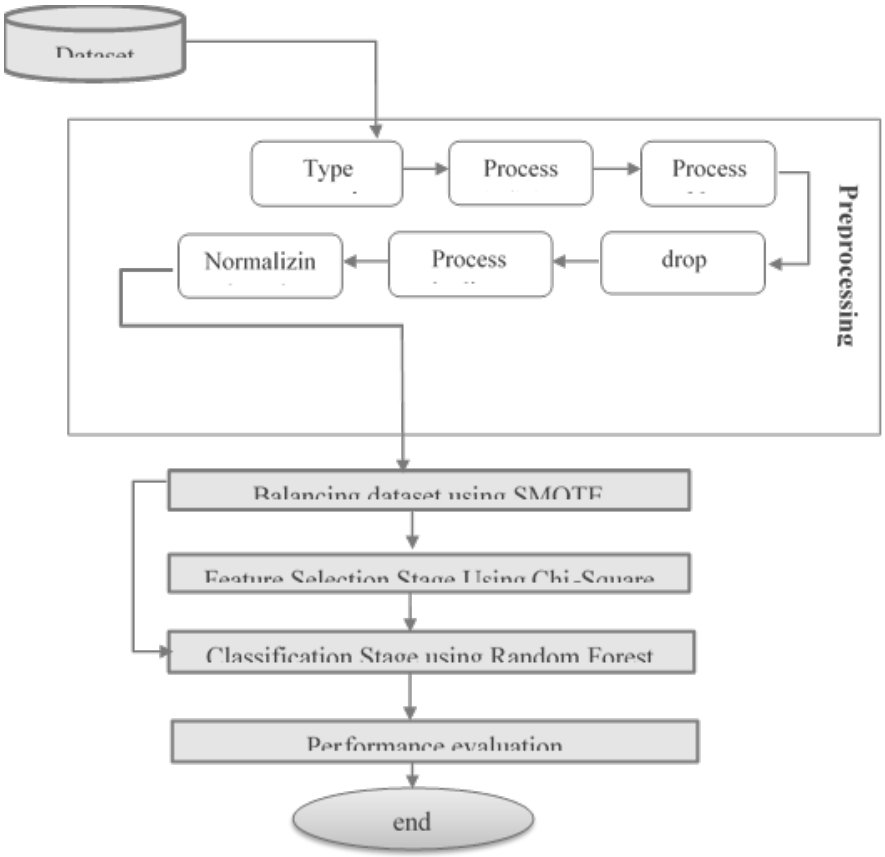

Figure (1) The proposed model

3 - 1 The Preprocessing Stage

This stage contains more than one step for preparing the dataset for extracting features and for classification as follow:

- Data collection:- the dataset used was CICandMal2017,available on: https://www.unb.ca/cic/datasets/andmal2017.html. this dataset specialized for android attacks. In the proposed model 500.000 sample selected from this dataset with 84 features. Table(1) presents the name of

this dataset features. The dataset is imbalance which contains multi class label are: (Benign, adware, SMSmalware, ransomware, and scareware) with different count are :(230360, 81050, 45314, 66679, 76596), while the percentage are (46.07%, 16.21%, 9.06%, 13.34%, 15.32%) respectively. Figure(2) shows the counts of the class labels in the dataset. While Figure (3) shows the percentage of the class labels in the dataset.

Table (1) Dataset feature names

Feature No.	Feature Name	Feature No.	Feature Name
0	FlowID	42	FwdPackets-s
1	SourceIP	43	BwdPackets-s
2	SourcePort	44	MinPacket Length
3	DestinationIP	45	MaxPacket Length
4	DestinationPort	46	PacketLength Mean
5	Protocol	47	PacketLength Std
6	Timestamp	48	PacketLength Variance
7	FlowDuration	49	FINFlag Count
8	TotalFwdPackets	50	SYNFlag Count
9	TotalBackwardPackets	51	RSTFlag Count
10	TotalLengthofFwdPackets	52	PSHFlag Count
11	TotalLengthofBwdPackets	53	ACKFlag Count
12	FwdPacketLengthMax	54	URGFlag Count
13	FwdPacketLengthMin	55	CWEFlag Count
14	FwdPacketLengthMean	56	ECEFlag Count
15	FwdPacketLengthStd	57	Down-Up Ratio
16	BwdPacketLengthMax	58	Average Packet Size
17	BwdPacketLengthMin	59	AvgFwd Segment Size
18	BwdPacketLengthMean	60	AvgBwd Segment Size
19	BwdPacketLengthStd	61	FwdHeader Length_1
20	FlowBytes_s	62	FwdAvg Bytes-Bulk
21	FlowPackets_s	63	FwdAvgPackets-Bulk
22	FlowIATMean	64	FwdAvgBulk Rate

Feature No.	Feature Name	Feature No.	Feature Name
23	FlowIATStd	65	BwdAvgBytes-Bulk
24	FlowIATMax	66	BwdAvgPackets-Bulk
25	FlowIATMin	67	BwdAvgBulk Rate
26	FwdIATTotal	68	SubflowFwd Packets
27	FwdIATMean	69	SubflowFwd Bytes
28	FwdIATStd	70	SubflowBwd Packets
29	FwdIATMax	71	SubflowBwd Bytes
30	FwdIATMin	72	Init-Win-bytes-forward
31	BwdIATTotal	73	Init-Win-bytes-backward
32	BwdIATMean	74	Act-data-pkt-fwd
33	BwdIATStd	75	Min-seg-size-forward
34	BwdIATMax	76	Active Mean
35	BwdIATMin	77	Active Std
36	FwdPSHFlags	78	Active Max
37	BwdPSHFlags	79	Active Min
38	FwdURGFlags	80	Idle Mean
39	BwdURGFlags	81	Idle Std
40	FwdHeaderLength	82	Idle Max
41	BwdHeaderLength	83	Idle Min

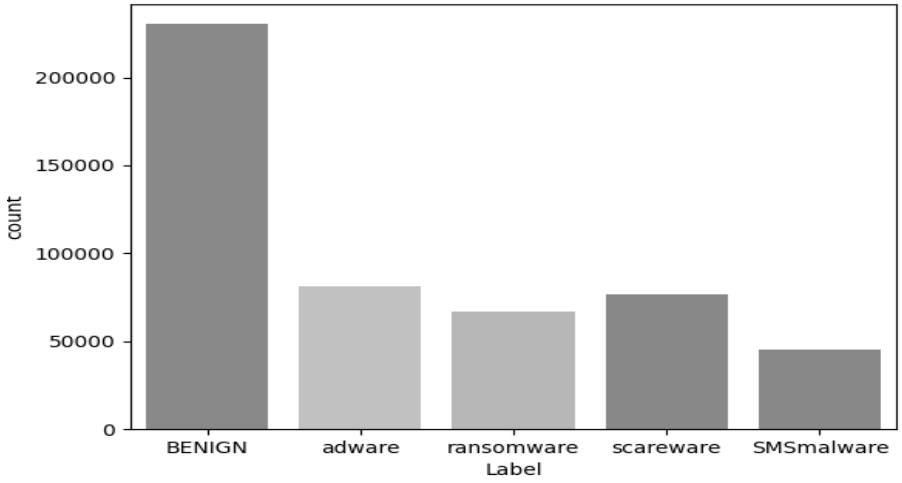

Figure (2) The count of each multi class label for the dataset

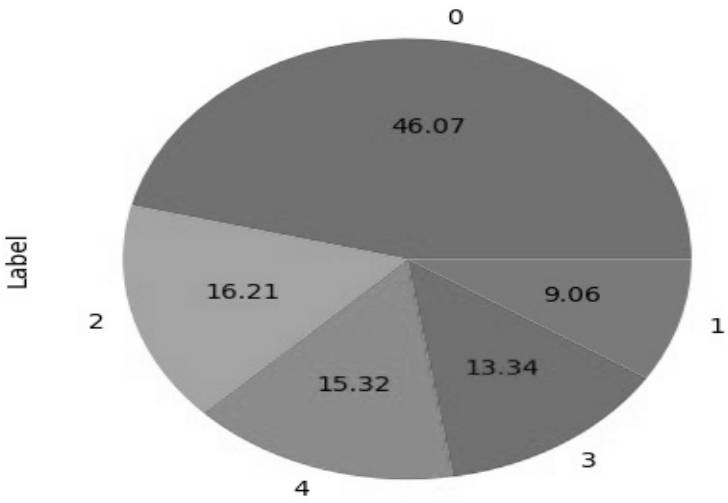

Figure (3) Each class percentage in the dataset datasett datase database database model Dataset

As shown in Figures (2) and (3), the dataset is unbalance. The engine class has the maximum percentage, while the less percentage was SMS malware class. So, the Synthetic Minority Oversampling Technique (SMOTE) technique has been applied for equaling the class label with same percentage. The following steps had been applied for processing the dataset: -

- Converting the features into numerical system.
- Dropping the following columns (FlowID, SourceIP, SourcePort, DestinationIP, Destination Port, Protocol, Timestamp) because these features have information about the networks, which the type of these features are string and don't affect on the model. So, the number of the features becomes 77 features.
- Replace the nan value with Standard deviation (STD) using the following equation:

$$\sigma = \sqrt{\frac{\sum (x_i - \mu)^2}{N}} \quad \text{..... (1)}$$

Where N is the size of the population, x_i is each value from the population, and μ is the population mean. While the number of nan value in all dataset is (172), these values replaced with STD value.

- Dropping the duplicated row. then, the number of instances after dropping is 486440.
- Normalizing the value of the dataset by computing the Min-max scaling.

After the preprocessing steps, the number of features is 77 features, with no duplicated rows, no NAN or infinity values. The features selection stage has been applied for extracting the important features and the suitable number of features which is effect on the accuracy of the model.

Also, the proposed model had been applied after balancing dataset using SMOTE technique. The number of class label was 1151795. Figure (4) shows the dataset after balancing using SMOTE technique. The proposed model applied for imbalanced and balanced dataset

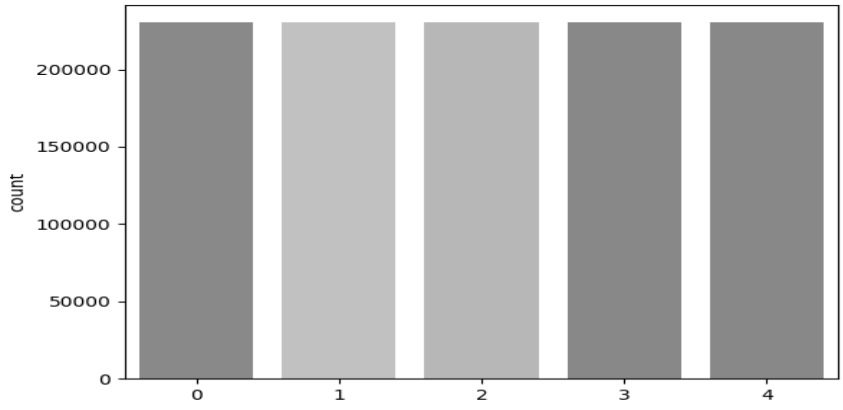

Figure (4) Class label count after balancing the dataset

3 - 2 The Features Selection Stage (FS)

The number of features affects the classification stage and hence on the accuracy of the model, because some features may reason in poor performance, more complexity, and overfitting for the model. Feature Selection is also known as variable elimination assists in decreasing computation requirement, understanding data, improving the predictor performance, and decreasing the effecting of the curse of dimensionality. Generally, the variables considered as noise, which causes bias in the predictor and reduces the classification performance if there is no correlation between the variables and class label. Three kinds of approaches for feature selection are the filter approach, wrapper approach, and embedded approach. Firstly, The filter approach relies on the general characteristics of data and evaluates features without involving any learning algorithm, is based on statistical measures such as Information gain, chi-square test, and Pearson's correlation Secondly, the wrapper method requires a learning algorithm and uses its accomplishment as an evaluation criterion to choose features. Recursive Feature Elimination (RFE) is one technique of the wrapper methods. Lastly, the embedded model, which selects the feature during the execution of a model algorithm. The common methods in the embedded model are Classification and Regression Tree (CART), and the C4.5 algorithm (Ali, 2022).

In this work, Chi- square filter method used to select most relevant features among (77) features resulted from preprocessing stage.

Chi-Square Method

The Chi-square often makes use of statistical techniques. It is dependent on two different types of hypotheses: first, the "Null" hypothesis

when the features are independent, and second, the "alternate" hypothesis when the features are dependent. By calculating the "degree of freedom" and the "expected value" using equations (2) and (3), one can determine the relationship between the attributes. Accept or reject the "Null" hypotheses as a result. Figure (2) depicts the Chi-square method's phases. A relationship between features or a relationship between a feature and the class label is indicated by a high Chi-square value.

$$X_c^2 = \sum \frac{(O_i - E_i)^2}{E_i} \quad \text{.....(2)}$$

Where c is the degree of freedom calculated using Eq. (3), O is the observed value, and E is the expected value.

$$Df = N - 1 \quad \text{.....(3)}$$

Where Df the degree of freedom, N is the size of a sample.

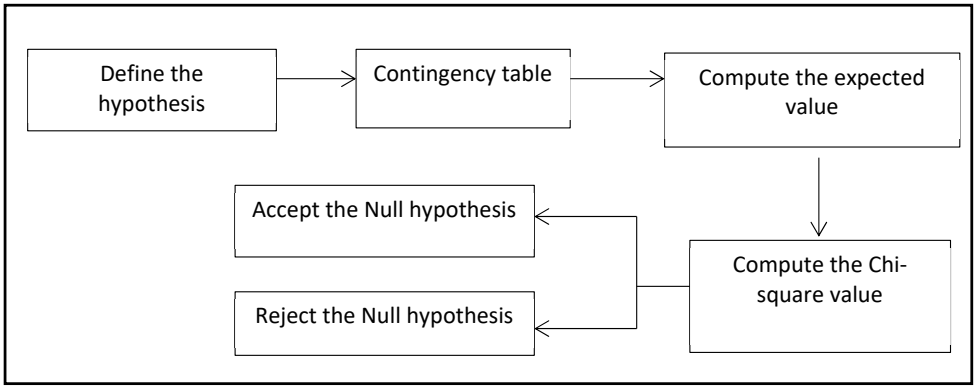

Figure (5) The steps of Chi-square method

3 - 3 The Classification Stage

In this stage, the features vector that was constructed in the previous stage and prepare for classification the android malware using RF algorithm.

3 - 3 - 1 Random Forest (RF) Algorithm

RF is a group of decision trees that differs slightly from one another. The theory states that different overfitting develops on various parts of the data when numerous decision trees are applied, each slightly different from the others. By averaging their results, the overfitting can be reduced. As a result, we may take advantage of decision trees' predictive abilities and the results of their over-fitting average (Biau , 2012)..

The Random Forest method gets its name from infusing randomization to the trees, which ensures that each one is different. The following is a description of the algorithm:

- Several decision trees are constructed using 70% of the total dataset, but the data used are randomly selected
- From among the predicted variables, a random variable is chosen. The algorithm then chooses the split that best matches these chosen variables and uses that split to divide the nodes.
- The prediction error or wrong classification rate is calculated using the remaining data.
- The algorithm selects the best result as the final result after comparing the classification outcomes and votes from the training trees (Chumachenko, 2017).

4 - Results and Discussion

Programming language Python (V.3.5) and framework Jet Brains Pycharm (V.2018.2) used to complete the proposed work. The intended effort had three primary stages, as was already described. The data preparation comes first. Second, the Chi-square approach is used to choose

features that have an impact on how well the malware attack is classified within the context of the Android ecosystem. Table (2) presents the value of each feature using the Chi-square method.

Table (2) The values for each feature by using Chi-square method

Feature No.	Feature Name	Feature Value	Feature No.	Feature Name	Feature Value
0	FlowDuration	332.9	40	PacketLengthMean	4.45
1	TotalFwdPackets	2.80	41	PacketLengthStd	120.8
2	TotalBackwardPackets	5.2	42	PacketLengthVariance	123.99
3	TotalLengthofFwdPackets	1.13	43	FINFlagCount	159.91
4	TotalLengthofBwdPackets	6.31	44	SYNFlagCount	394.59
5	FwdPacketLengthMax	216.7	45	RSTFlagCount	0.000
6	FwdPacketLengthMin	40.19	46	PSHFlagCount	281.61
7	FwdPacketLengthMean	58.96	47	ACKFlagCount	654.87
8	FwdPacketLengthStd	150.5	48	URGFlagCount	1826.43
9	BwdPacketLengthMax	181.8	49	CWEFlagCount	0.00
10	BwdPacketLengthMin	69.8	50	ECEFlagCount	0.00
11	BwdPacketLengthMean	116.7	51	Down_UpRatio	64.21
12	BwdPacketLengthStd	0.64	52	AveragePacketSize	83.8
13	FlowBytes_s	2.31	53	AvgFwdSegmentSize	59.016
14	FlowPackets_s	0.009	54	AvgBwdSegmentSize	116.77
15	FlowIATMean	432.1	55	FwdHeaderLength_1	6.29
16	FlowIATStd	116.8	56	FwdAvgBytes_Bulk	0.00
17	FlowIATMax	296.53	57	FwdAvgPackets_Bulk	0.00
18	FlowIATMin	495.25	58	FwdAvgBulkRate	0.00
19	FwdIATTotal	263.55	59	BwdAvgBytes_Bulk	0.00

Feature No.	Feature Name	Feature Value	Feature No.	Feature Name	Feature Value
20	FwdIATMean	389.04	60	BwdAvgPackets_Bulk	0.00
21	FwdIATStd	119.17	61	BwdAvgBulkRate	0.00
23	FwdIATMax	234.31	62	SubflowFwdPackets	3.18
24	FwdIATMin	414.07	63	SubflowFwdBytes	1.14
25	BwdIATTotal	385.13	64	SubflowBwdPackets	5.24
26	BwdIATMean	39.33	65	SubflowBwdBytes	6.31
27	BwdIATStd	102.14	66	Init_Win_bytes_forward	291.39
28	BwdIATMax	219.34	67	Init_Win_bytes_backward	67.62
29	BwdIATMin	20.48	68	act_data_pkt_fwd	1.57
30	FwdPSHFlags	1.64	69	min_seg_size_forward	5.01
31	BwdPSHFlags	9.92	70	ActiveMean	16.47
32	FwdURGFlags	9.92	71	ActiveStd	17.8
33	BwdURGFlags	9.92	72	ActiveMax	27.81
34	FwdHeaderLength	6.25	73	ActiveMin	8.73
35	BwdHeaderLength	6.99	74	IdleMean	236.35
36	FwdPackets_s	4.61	75	IdleStd	48.37
37	BwdPackets_s	1.85	76	IdleMax	253.50
38	MinPacketLength	1.90	77	IdleMin	227.65
39	MaxPacketLength	2.14			

As shown in Table (2), the values are various between features, these values effect on the result of the model. So, the highest value means has maximum effect on the result of the model, and the lowest value means has minimum effect on the results of the model. Figure (6) shows the histogram of values for features using Chi-square method.

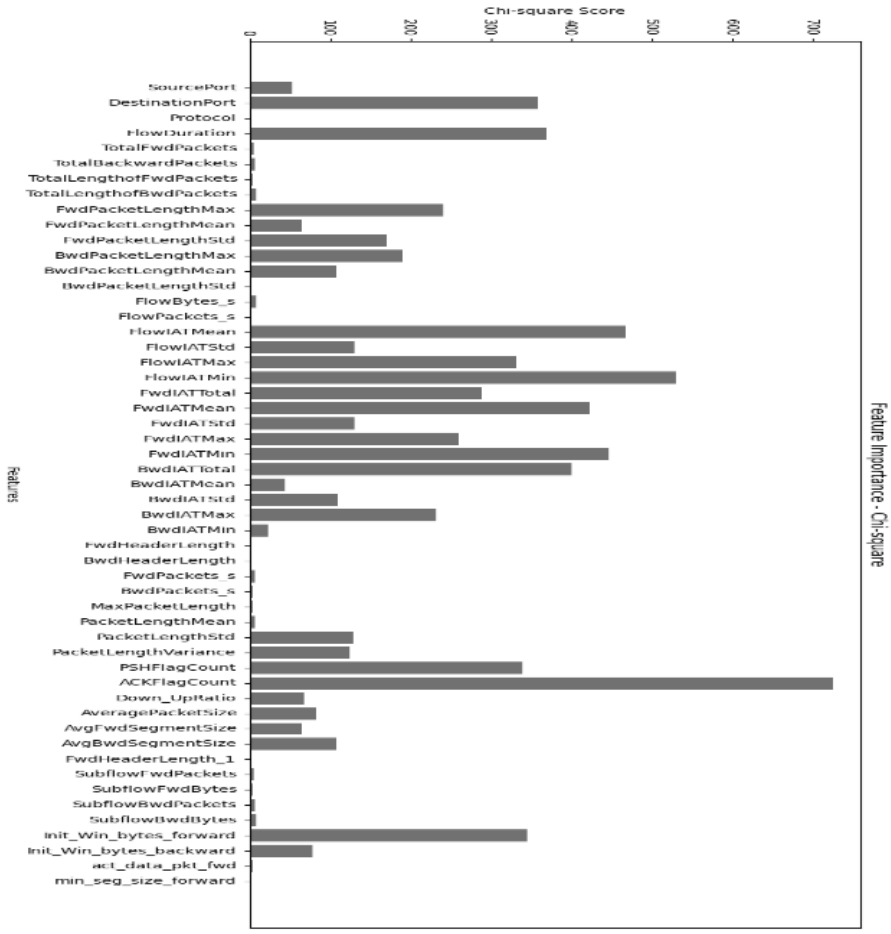

Figure (6) Sorted features according to Chi-square method

As shown in Figure (6), the feature named ("ACKFlagCount") has the maximum value compared with the other features, while there are more than one features have zero value.

Finally, the classification stage carried out using RF algorithm for classifying the features vector. The RF classifier had been applied in two stages: Firstly, when imbalanced dataset for a total number of features and the best (20, 40, 60)

features that were selected using Chi-square method. Secondly, when balanced dataset for a total number of features and the best (20, 40, 60) features that were selected using Chi-square method. The dataset was split into 70% of the dataset for training and 30% of the dataset for testing, for classifying the malware under android environment. The following equations used for computing the accurate measurements of the proposed work (Kumar, 2014).

$$Accuracy = \frac{Number\ of\ correct\ predictions}{Total\ number\ of\ predictions} = \frac{TP + TN}{TP + FN + TN + FP} \quad \text{.....(4)}$$

$$precision = \frac{TP}{TP + FP} \quad \text{.....(5)}$$

$$recall = \frac{TP}{TP + FN} \quad \text{.....(6)}$$

$$F1 = 2 * \frac{precision * recall}{precision * recall} \quad \text{.....(7)}$$

Where TP= true positive, FN= false negative, FP= false positive,

TN= true negative.

Table (4) presents the classification accuracy for imbalance dataset using RF algorithm for four cases (when classify all features, the best 20 features, the best 40 features, the best 60 features) which were selected depending on Chi-square selection method as mentioned above.

Table (4) The accuracy of the proposed work for imbalanced data

Classifier	All features	Chi-square (20_features)	Chi-square (40_features)	Chi-square (60_features)
RF	89.93	92.16	93.05	93.30

As shown in Table (4), the results showed there are improvements in accuracy when using features selection technique. And the best accuracy when the number of features is 40 features and 60 features. This is means, the number

of features effect on the accuracy of the model. Table (5) presents the accuracy of classification for balance dataset using RF algorithm for four cases (when classify all features, the best 20 features, the best 40 features, the best 60 features) which were selected depending on Chi-square selection method as mentioned above.

Table(5) The accuracy of the proposed work for balanced dataset

Classifier	All features	Chi-square (20_features)	Chi-square (40_features)	Chi-square (60_features)
RF	92.06	91.36	91.61	92.22

As shown in Table (5), The results showed that the accuracy was approximate when using features selection technique. And the best accuracy when the number of features is 60 features. By comparing between Tables (4) and (5), the accuracy of the model for balanced dataset and imbalanced dataset was approximate but in the balanced dataset has no bias in the results as all. The classification report presents in Tables (6) and (7) including (precision, recall, F1-score, and support) of performance measures for imbalanced and balanced dataset respectively.

Table (6) The classification report for imbalanced dataset

Method	Precision	recall	F1-score	support	Class label
All Features	0.83	1.00	0.91	66992	Class label (0)
	0.99	0.79	0.88	13469	Class label (1)
	0.98	0.84	0.90	23768	Class label (2)
	0.97	0.81	0.88	19235	Class label (3)
	0.98	0.82	0.89	22468	Class label (4)
			0.90	145932	Accuracy
20_Features	0.87	1.00	0.93	66992	Class label (0)
	0.99	0.83	0.90	13469	Class label (1)
	0.98	0.88	0.93	23768	Class label (2)
	0.97	0.85	0.91	19235	Class label (3)
	0.98	0.86	0.92	22468	Class label (4)
			0.92	145932	Accuracy

Method	Precision	recall	F1-score	support	Class label
40_Features	0.89	0.99	0.94	66992	Class label (0)
	0.98	0.86	0.91	13469	Class label (1)
	0.98	0.89	0.92	23768	Class label (2)
	0.97	0.87	0.92	19235	Class label (3)
	0.98	0.88	0.93	22468	Class label (4)
			0.93	145932	Accuracy
60_Features	0.89	0.99	0.94	66992	Class label (0)
	0.99	0.85	0.92	13469	Class label (1)
	0.98	0.90	0.94	23768	Class label (2)
	0.97	0.88	0.92	19235	Class label (3)
	0.98	0.88	0.93	22468	Class label (4)
			0.93	145932	Accuracy

Table (7) The classification report for balanced dataset

Method	Precision	recall	F1-score	support	Class label
All Features	0.92	0.92	0.92	66903	Class label (0)
	0.94	0.92	0.93	66883	Class label (1)
	0.93	0.93	0.93	66775	Class label (2)
	0.89	0.95	0.92	67103	Class label (3)
	0.95	0.91	0.93	67166	Class label (4)
			0.93	334830	Accuracy
20_Features	0.91	0.91	0.91	66903	Class label (0)
	0.92	0.91	0.91	66883	Class label (1)
	0.94	0.91	0.92	66775	Class label (2)
	0.88	0.94	0.91	67103	Class label (3)
	0.93	0.90	0.91	67166	Class label (4)
			0.91	334830	Accuracy
40_Features	0.92	0.91	0.92	66903	Class label (0)
	0.90	0.92	0.91	66883	Class label (1)
	0.95	0.91	0.93	66775	Class label (2)
	0.89	0.94	0.91	67103	Class label (3)
	0.93	0.90	0.92	67166	Class label (4)
			0.92	334830	Accuracy

Method	Precision	recall	F1-score	support	Class label
60_Features	0.94	0.92	0.93	66903	Class label (0)
	0.90	0.93	0.92	66883	Class label (1)
	0.95	0.92	0.93	66775	Class label (2)
	0.89	0.94	0.91	67103	Class label (3)
	0.94	0.91	0.92	67166	Class label (4)
			0.92	334830	Accuracy

As shown in Tables (6) and (7), the performance measures (Precision, recall, F1-score,and support) had been presented for showing the values of these measures for the all class labels which shows more accurate results especially when the dataset is imbalanced.

Also from Figures (2) and (3), it is clear that the number of the class labels were five. So, the proposed model was multi classification while the percentages of the class label were not equal, this cause to bias accuracy for the class label which has the high percentage in the dataset. So, the SMOTE technique had been applied for balancing the dataset. The total number of features in original dataset were 83 features, this number of features are very high. So, the dataset was processed by removing first seven columns, removing the duplicated instances, processing the null values by replacing it with STD value, and normalizing the values. The result of the preprocessing stage, the number of features became 77 features but also the number of features were high which is effect on the classification of the model. So, the Chi-square technique had been applied in the feature selection stage. As shown in Table (4), the accuracy of the model using RF algorithm for all features was 89.93%, while the accuracy of the model for selected number of features was more than the accuracy for all features. The best accuracy when the number of features was (40, 60) features, this means the number

of features effect on the accuracy. So, the feature selection or extraction had positive effect on the accuracy of the model in the imbalanced dataset and balanced dataset. As shown in Table (5), increasing the accuracy when using balanced dataset. Also, the best accuracy when number of features was 20 features. However, because the difference between the numbers of class labels, the classification report presented in Tables (6) and (7) these two Tables show the values of the performance measures, which explain the bias of the accuracy for the imbalanced and balanced dataset. Finally Table (8) shows the comparison between the previous works and the proposed work.

Table (8) The comparison between the previous studies and proposed work

Reference	year	Use dimension reduction	Classification techniques	The Results
Lekssays , *et al.*	2020	?	CNN	84.9% AC
Arslan	2021	?	ensemble ML model	AC of 90.4%. PRE, recall and PRE values were also %90.4.
Elayan, Mustafa	2021	?	GRU	AC 98.2%.
Zhixing , *et al.*,	2021	?	ensemble learning method based on Stacking.	AC 96.7%.
Xue , *et al*	2021	?	ensemble learning approach based on stacking	AC 96.7%
Niu	2022	?	K-means, SVM and MLP	AC K-means 86.75%, MLP 99.87%
The proposed model	2023	?	SMOTE +Chi2 feature selection +RF classifier	RF without SMOTE for (all,20, 40,60) feature respectively (89.93, 92.16, 93.05, 93.30) RF + SMOTE for (all,20, 40,60) feature respectively (92.06, 91.36, 91.61, 92.22)

5 - Conclusion

In this work, the prediction of cyber security attacks using feature selection techniques and ML techniques achieved. The dataset contained multi class label with five type of cyber and security attack were (Benign, adware, SMSmalware, Ransomware, and Scareware) with different percentage. So, the dataset was imbalance. In this work, more than one steps applied for processing the dataset such as (processing NaN value, duplicated rows, normalizing values). The Chi-square feature selection technique has been applied for determining the important attributes that affect the evaluation of the prediction for cyber security attacks. In addition, RF technique applied for classifying the features vector in hence predicting the attacks depending on these features. The usage of feature selection had positive role on the accuracy of the classification. Also, the number of features had important role on the accuracy, when the number of features were forty and sixty features got good accuracy compared with other number of features and when using all features. Also, the balancing of data had good results in the classification.

References

- Alhebsi, M. S. (2022). Android Malware Detection Using Machine Learning Techniques.
- https://www.statista.com/statistics/272698/
- Malik, S., & Khatter, K. (2020). Malicious Application Detection and Classification System for Android Mobiles. In Cognitive Analytics: Concepts, Methodologies, Tools, and Applications (pp. 122-142). IGI Global.
- Muttoo, Sunil Kumar, and Shikha Badhani.(2017), "Android Malware Detection: State of the Art." International Journal of Information Technology 9, 111-117.
- Senanayake, J., Kalutarage, H., & Al-Kadri, M. O. (2021). Android Mobile Malware Detection Using Machine Learning: A Systematic Review. Electronics, 10(13), 1606.
- Zulkifli, A., Hamid, I. R. A., Shah, W. M., & Abdullah, Z. (2018). Android Malware Detection Based on Network Traffic Using Decision Tree Algorithm. In Recent Advances on Soft Computing and Data Mining: Proceedings of the Third International Conference on Soft Computing and Data Mining (SCDM 2018), Johor, Malaysia, February 06-07, (pp. 485-494). Springer International Publishing.
- Xie, N., Qin, Z., & Di, X. (2023). GA-StackingMD: Android Malware Detection Method Based on Genetic Algorithm Optimized Stacking. Applied Sciences, 13(4), 2629.
- Dehkordy, D. T., & Rasoolzadegan, A. (2021). A New Machine Learning-based Method for Android Malware Detection on Imbalanced Dataset. Multimedia Tools and Applications, 80, 24533-24554.
- Lashkari, A. H., Kadir, A. F. A., Taheri, L., & Ghorbani, A. A. (2018). Toward Developing A Systematic Approach to Generate Benchmark Android Malware Datasets and Classification. In International Carnahan Conference on Security Technology (ICCST) (pp. 1-7). IEEE.
- Lekssays, A., Falah, B., & Abufardeh, S. (2020), A Novel Approach for Android Malware Detection and Classification using Convolutional Neural Networks. In ICSOFT (pp. 606-614).
- Arslan, R. S. (2021), Identify Type of Android Malware with Machine Learning Based Ensemble Model. In 5th International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT) (pp. 628-632). IEEE.
- Elayan, O. N., & Mustafa, A. M. (2021), Android Malware Detection Using Deep Learning. Procedia Computer Science, 184, 847-852.
- Xue, Z., Niu, W., Ren, X., Li, J., Zhang, X., & Chen, R. (2021), A Stacking-based Classification Approach to Android Malware Using Host-level Encrypted Traffic. In Journal of Physics: Conference Series (Vol. 2024, No. 1, pp. 012049). IOP Publishing.

- Niu, G. (2022). Malicious Application Traffic Detection and Identification for Mobile Android Devices. Informatica, 46(8).
- : https://www.unb.ca/cic/datasets/andmal2017.html
- Grace, M., & Sughasiny, M. (2022). Malware Detection for Android Application Using Aquila Optimizer and Hybrid LSTM-SVM classifier. EAI Endorsed Transactions on Scalable Information Systems, 10(1).
- Biau, G. (2012). Analysis of A Random Forests Model. The Journal of Machine Learning Research, 13(1), 1063-1095.
- Chumachenko, K. (2017). Machine Learning Methods for Malware Detection and Classification.
- Kumar, G. (2014). Evaluation Metrics for Intrusion Detection Systems-A Study. International Journal of Computer Science and Mobile Applications, II, 11.