ANDROID MALWARE DETECTION USING GENETIC ALGORITHM BASED OPTIMIZED FEATURE SELECTION AND MACHINE LEARNING

Pilli Madhavi, H.No: 21s41d5811,Mtech (Cse), Department Of Computer Science, Vaageswari College Of Engineering Thimmapur,Karimnagar, Telangana, India, Email-Id: Madhavipilli607@Gmail.Com.

Y.Susheela, Associate Professor, Department Of Computer Science, Vaageswari College Of Engineering Thimmapur,Karimnagar, Telangana, India,Email-Id: Chiduralasusheela@Gmail.Com.

ABSTRACT

The Android platform has the biggest worldwide market share because to its open source nature and support from Google. Due to the widespread dissemination of malicious programmes, the most widely used operating system in the world has attracted the attention of cybercriminals. This research suggests an efficient machine-learning method for Android malware detection that uses an evolutionary genetic algorithm to choose features based on discrimination. Machine learning classifiers are trained using specific features using genetic algorithms, and their ability to identify malware both before and after feature selection is compared. The testing findings confirm that the best optimised feature subset produced by the genetic algorithm aids in reducing the feature dimension to less than half of the original feature set. While operating on a much smaller feature dimension, machine learning-based classifiers retain a classification accuracy of over 94% after feature selection, which positively affects the computing cost of learning classifiers.

Index Terms:— Android malware analysis; feature selection; Genetic algorithm; machine learning; reverse-engineering.

I.INTRODUCTION

1.1 MOTIVATION:

Android is an open source free operating system and it has support from Google to publish android application on its Play Store. Anybody can developed an android app and publish on play store free of cost. This android feature attract cyber-criminals to developed and publish malware app on play store. If anybody install such malware app then it will steal information from phone and transfer to cyber-criminals or can give total phone control to criminal's hand. To protect users from such app in this paper author is using machine learning algorithm to detect malware from mobile app. To detect malware from app we need to extract all code from app using reverse engineering and then check whether app is doing any mischievous activity such as sending SMS or copying contact details without having proper permissions. If such activity given in code then we will detect that app as malicious app. In a single app there could be more than 100 permissions (examples of permissions are transact, API call signature, onServiceConnected, API call signature, bindService, API call signature, attachInterface, API call signature, ServiceConnection, API call signature, android.os.Binder, API call signature, SEND_SMS, Manifest Permission, Ljava.lang.Class.getCanonicalName, API call signature etc.) which we need to extract from code and then generate a features dataset, if app has proper permission then we will put value 1 in the features data and if not then we will value 0. Based on those features dataset app will be mark as malware or good ware.

1.2 PROBLEM DEFINITION:

Android Apps are freely available on Google Playstore, the official Android app store as well as third-party app stores for users to download. Due to its open source nature and popularity, malware writers are increasingly focusing on developing malicious applications for Android operating system. In spite of various attempts by Google Playstore to protect against malicious apps, they still find their way to mass market and cause harm to users by misusing personal information related to their phone book, mail accounts, GPS location information and others for misuse by third parties or else take control of the phones remotely. Therefore, there is need to perform malware analysis or reverse-engineering of such malicious applications which

pose serious threat to Android platforms. Broadly speaking, Android Malware analysis is of two types: Static Analysis and Dynamic Analysis. Static analysis basically involves analyzing the code structure without executing it while dynamic analysis is examination of the runtime behavior of Android Apps in constrained environment. Given in to the ever-increasing variants of Android Malware posing zero-day threats, an efficient mechanism for detection of Android malwares is required. In contrast to signature-based approach which requires regular update of signature database.

1.3 OBJECTIVE OF PROJECT:

This project proposed a novel and efficient algorithm for feature selection to improve overall detection accuracy.

Machine-learning based approach in combination with static and dynamic analysis can be used to detect new variants of Android Malware posing zero-day threats.

To reduce dimensionality of feature-set, the CSV is fed to Genetic Algorithm to select the most optimized set of features. The optimized set of features obtained is used for training two machine learning classifiers: Support Vector Machine and Neural Network.

In the proposed methodology, static features are obtained from AndroidManifest.xml which contains all the important information needed by any Android platform about the Apps. Androguard tool has been used for disassembling of the APKs and getting the static features.

II.LITERATURE SURVEY

- D. Arp, M. Spreitzenbarth, M. Hübner, H. Gascon, and K. Rieck proposed a malicious applications pose a threat to the security of the Android platform. The growing amount and diversity of these applications render conventional defenses largely ineffective and thus Android smartphones often remain unprotected from novel malware. In this paper, we propose DREBIN, a lightweight method for detection of Android malware that enables identifying malicious applications directly

on the smartphone. As the limited resources impede monitoring applications at run-time, DREBIN performs a broad static analysis, gathering as many features of an application as possible. These features are embedded in a joint vector space, such that typical patterns indicative for malware can be automatically identified and used for explaining the decisions of our method. In an evaluation with 123,453 applications and 5,560 malware samples DREBIN outperforms several related approaches and detects 94% of the malware with few false alarms, where the explanations provided for each detection reveal relevant properties of the detected malware. On five popular smartphones, the method requires 10 seconds for an analysis on average, rendering it suitable for checking downloaded applications directly on the device.

- N. Milosevic, A. Dehghantanha, and K. K. R. Choo described about the widespread adoption of Android devices and their capability to access significant private and confidential information have resulted in these devices being targeted by malware developers. Existing Android malware analysis techniques can be broadly categorized into static and dynamic analysis. In this paper, we present two machine learning aided approaches for static analysis of Android malware. The first approach is based on permissions and the other is based on source code analysis utilizing a bag-of-words representation model. Our permission-based model is computationally inexpensive,

and is implemented as the feature of OWASP Seraphimdroid Android app that can be obtained from Google Play Store. Our evaluations of both approaches indicate an F-score of 95.1% and F-measure of 89% for the source code-based classification and permission-based classification models, respectively.

- J. Li, L. Sun, Q. Yan, Z. Li, W. Srisa-An, and H. Ye, explains about the alarming growth rate of malicious apps has become a serious issue that sets back the prosperous mobile ecosystem. A recent report indicates that a new malicious app for Android is introduced every 10 seconds. To combat this serious malware campaign, we need a scalable malware detection approach that can effectively and efficiently identify malware apps. Numerous malware detection tools have been developed, including system-level and network-level approaches. However, scaling the detection for a large bundle of apps remains a challenging task. In this paper, we introduce SIGPID, a malware detection system based on permission usage analysis to cope with the rapid increase in the number of Android malware. Instead of extracting and analyzing all Android permissions, we develop 3-levels of pruning by mining the permission data to identify the most significant permissions that can be effective in distinguishing between benign and malicious apps. SIGPID then utilizes machine-learning based classification methods to classify different families of malware and benign apps. Our

evaluation finds that only 22 permissions are significant. We then compare the performance of our approach, using only 22 permissions, against a baseline approach that analyzes all permissions. The results indicate that when Support Vector Machine (SVM) is used as the classifier, we can achieve over 90% of precision, recall, accuracy, and F-measure, which are about the same as those produced by the baseline approach while incurring the analysis times that are 4 to 32 times less than those of using all permissions. Compared against other state-of-the-art approaches, SIGPID is more effective by detecting 93.62% of malware in the data set, and 91.4% unknown/new malware samples.

- Saracino, D. Sgandurra, G. Dini, and F. Martinelli explains about the android users are constantly threatened by an increasing number of malicious applications (apps), generically called malware. Malware constitutes a serious threat to user privacy, money, device and file integrity. In this paper we note that, by studying their actions, we can classify malware into a small number of behavioral classes, each of which performs a limited set of misbehaviors that characterize them. These misbehaviors can be defined by monitoring features belonging to different Android levels. In this paper we present MADAM, a novel host-based malware detection system for Android devices which simultaneously analyzes and correlates features at four levels: kernel, application, user and package, to detect and stop

malicious behaviors. MADAM has been designed to take into account those behaviors characteristics of almost every real malware which can be found in the wild. MADAM detects and effectively blocks more than 96% of malicious apps, which come from three large datasets with about 2,800 apps, by exploiting the cooperation of two parallel classifiers and a behavioral signature-based detector. Extensive experiments, which also includes the analysis of a testbed of 9,804 genuine apps, have been conducted to show the low false alarm rate, the negligible performance overhead and limited battery consumption.

- Martin, F. Fuentes-Hurtado, V. Naranjo, and D. Camacho proposed Deep Neural Networks (DNN) have become a powerful, widely used, and successful mechanism to solve problems of different nature and varied complexity. Their ability to build models adapted to complex non-linear problems, have made them a technique widely applied and studied. One of the fields where this technique is currently being applied is in the malware classification problem. The malware classification problem has an increasing complexity, due to the growing number of features needed to represent the behaviour of the application as exhaustively as possible. Although other classification methods, as those based on SVM, have been traditionally used, the DNN pose a promising tool in this field. However, the parameters and architecture setting of these DNNs present a serious restriction, due to the necessary

time to find the most appropriate configuration. This paper proposes a new genetic algorithm designed to evolve the parameters, and the architecture, of a DNN with the goal of maximising the malware classification accuracy, and minimizing the complexity of the model. This model is tested against a dataset of malware samples, which are represented using a set of static features, so the DNN has been trained to perform a static malware classification task. The experiments carried out using this dataset show that the genetic algorithm is able to select the parameters and the DNN architecture settings, achieving a 91% accuracy.

III. EXISTING SYSTEM

In existing System,MADAM[4] provides an multi-level analysis framework where behavior of Android Apps is captured upto four levels: from package, user, application to kernel level, achieving detection accuracy as high as 96% with one disadvantage being that it could run only on rooted devices, making it incapable for commercial use. SAMADroid proposed a three-way novel host server based methodology for improved performance as far as asset usage is concerned for malware detection on mobile devices. The current drift in malware detection has shifted towards deep learning applications where it requires least human intervention as propose.

DISADVANTAGES OF EXISTING SYSTEM:

- We got less accuracy. A novel and efficient algorithm for feature selection to improve overall detection accuracy.
- The performance is not well for Application.

IV PROPOSED SYSTEM:

In this paper author detecting malware from android app by extracting permissions list from android APK file and then building features vector, if any APK asking for any permissions such as

sending SMS, calling someone programmatically etc. All this permissions will be extracted from APK and then put value 1 if any APK request for any permission and if APK not request permission then value 0 will put. This feature vector will be input to NEURAL Networks and KMEANS algorithm to build Machine Learning model. This model will be applied on new test android test features to predict whether android APK contains malware or not.

Malicious programmers may request sensitive permissions from android and then once granted permission then steal sensitive information from device and send to malicious users. By using this application we will analyse each permission and then predict whether user requesting that permission is performing genuine or malicious activity.

ADVANTAGES OF PROPOSED SYSTEM:

- Genetic algorithm has been used because of its capabilities in finding a feature subset selected from original feature vector.
- It is observed that a decent classification accuracy of more than 94% is maintained.

V. SYSTEM DESIGN

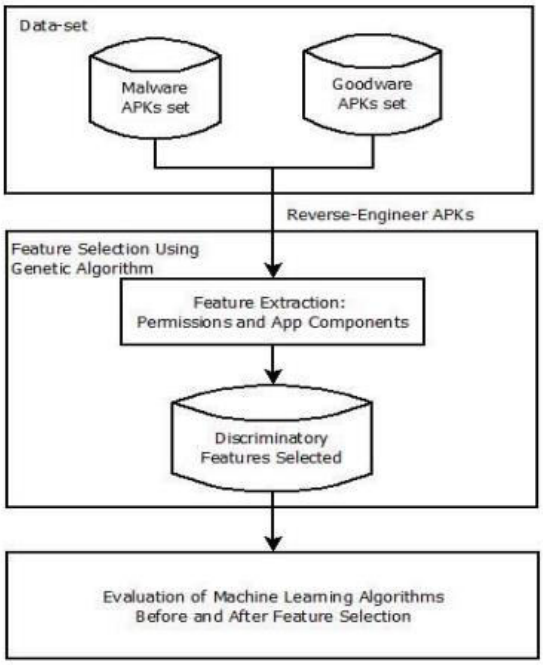


Fig1: Architecture of system.

VI.MODULE DESCRIPTION:

1. Data Set: This phase analyzes data and its parameters to check any redundancy in data values that may affect prediction results. If a dataset contains any irrelevant parameters, then those data values are removed. This phase also analyze data for the possible merging of data for improved model predictability.

2. Data Filtration Phase: This phase filters data to remove all empty/redundant values.

3. Train-Test Split Phase: This phase splits data into training and testing data subsets. For example, data are divided into two parts per a ratio of 70% training data and 30% test data.

4. Data-Scaling Phase: Before data are passed to the model, the data are scaled according to model requirements. In this way, this phase reshapes data to make them more suitable for the model.

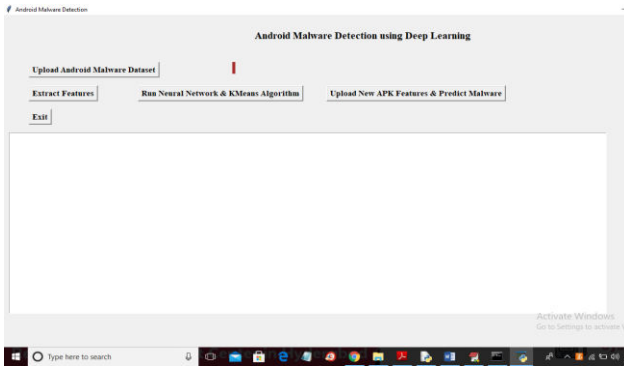
5. Model-Building Phase: The proposed approach is implemented in Python. For any machine learning models, data in depth to analyze all kinds of patterns formed in the dataset to make the model more precise. Then the data are passed to that model for training.

6. Model Learning and Evaluation Phase: The Neural Network and K-Means Algorithms is used build Model.

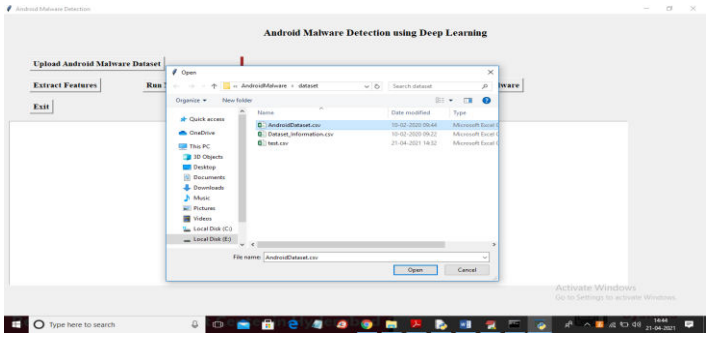
7. Prediction Phase: Prediction is made using the saved model. Input values are passed to the model to give predicted values as the output. Then that output is compared with testing data to calculate accuracy and losses.

VII. RESULT:

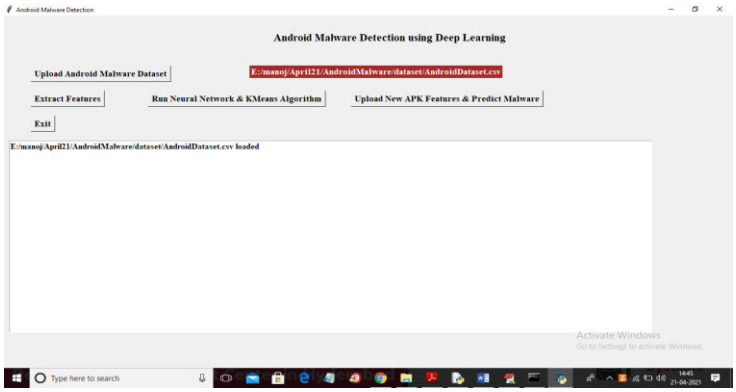
To run project double click on 'run.bat' file to get below screen



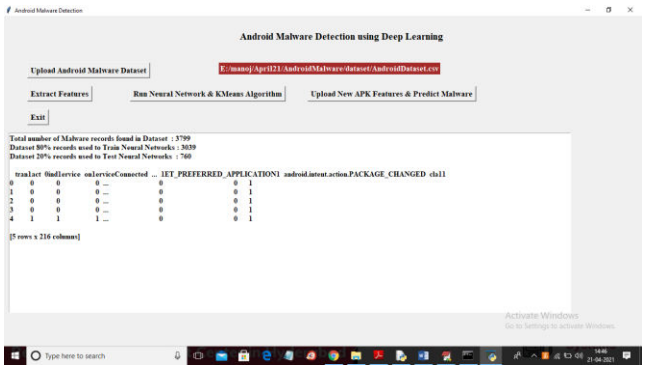
In above screen click on 'Upload Android Malware Dataset' button to upload dataset



In above screen selecting and uploading 'AndroidDataset.csv' file and then click on 'Open' button to load dataset and to get below screen

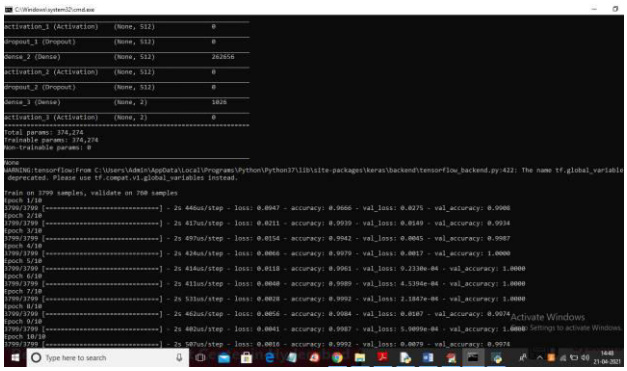


In above screen dataset loaded and now click on 'Extract Features' button to read all features from dataset and to get below screen

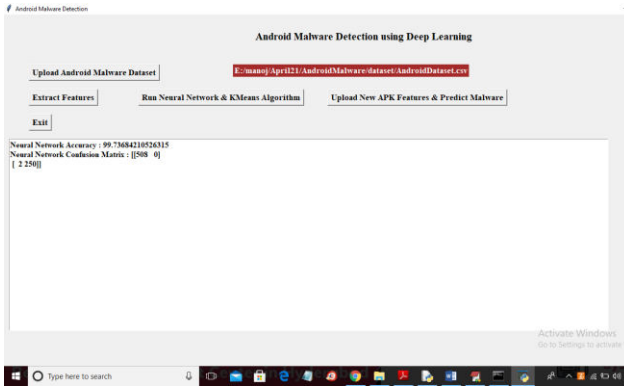


In above screen first line displaying dataset contains 3799 records and application splitting dataset into 80 and 20% where 3039 records used to train neural networks

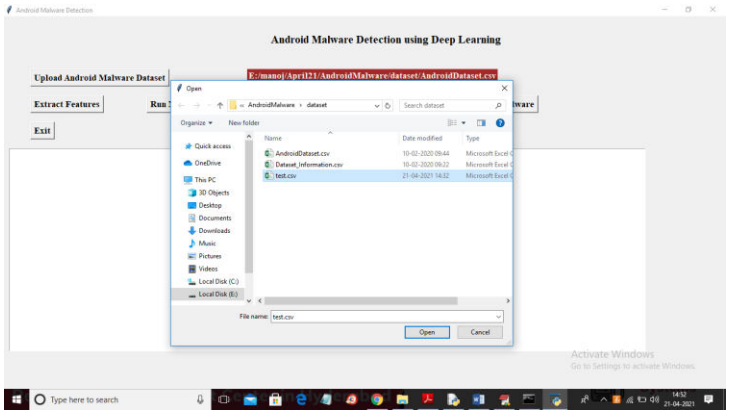
and 20% 760 records used to test Neural Network accuracy. To select similar records we are applying KMEANS algorithm and this similar records will be input to Neural Network and then calculate accuracy and confusion matrix



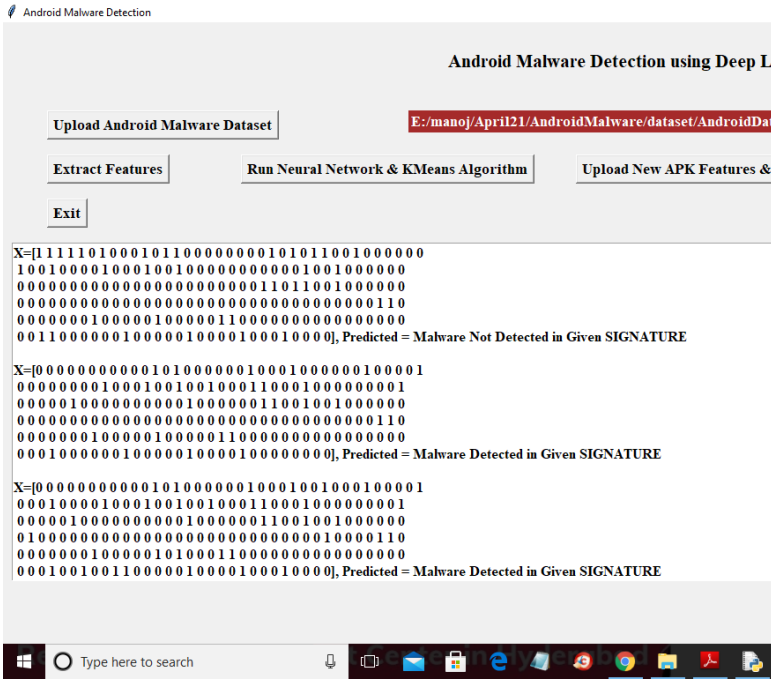
In above screen to train neural networks we took 10 epoch and at each increasing epoch accuracy of trained model is getting increase which means model is generating with accurately predicting with test data after building model will get below screen



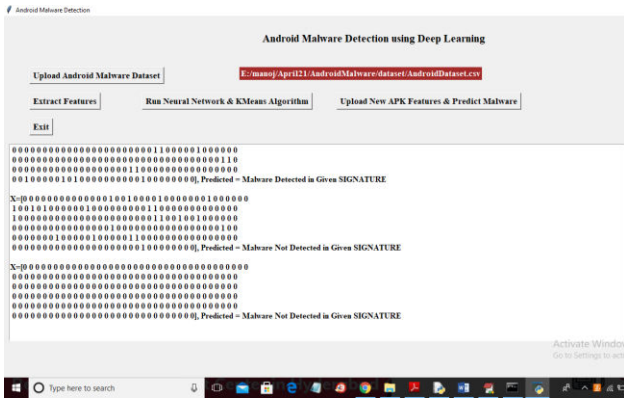
In above screen we got neural network accuracy as 99% and in above confusion matrix we got 508 classes predicting in class label 0 and 250 classes predicting in class label 1 and only 2 records are incorrectly predicted due to which 0.23% accuracy reduced. Now click on 'Upload New APK Features & Predict Malware' button to upload new android test features and then will get prediction result



In above screen selecting and uploading 'test.csv' file and then click on 'Open' button to load test dataset and then will get below prediction result



In above screen in square brackets we can see all android permission test features and then after square bracket we can see whether MALWARE detected or not in that record. You can scroll down above text area to view all results



VIII. CONCLUSION

Designing a framework that can accurately identify malware is crucial, since the amount of threats to Android platforms is growing daily. Malicious apps or malware are the major means of propagating these threats. Machine learning-based techniques are being employed in situations when signature-based methods are unable to identify novel malware variants presenting zero-day risks. The suggested technique aims to get the most optimised feature subset via the application of evolutionary genetic algorithms, which can then be used to train machine learning algorithms in the most effective manner.

IX. FUTURE ENHANCEMENT

Experiments show that when using Support Vector Machine and Neural Network classifiers on smaller dimension feature-sets, a respectable classification accuracy of over 94% is maintained, thereby lowering the classifiers' training complexity. Larger datasets may be utilised in future research to get better outcomes and to examine how employing Genetic Algorithm in combination with other machine learning algorithms affects those other algorithms.

X. REFERENCES

[1] D. Arp, M. Spreitzenbarth, M. Hübner, H. Gascon, and K. Rieck, “Drebin: Effective and Explainable Detection of Android Malware in Your Pocket,” in Proceedings 2014 Network and Distributed System Security Symposium, 2014.

[2] N. Milosevic, A. Dehghantanha, and K. K. R. Choo, “Machine learning aided Android malware classification,” Comput. Electr. Eng., vol. 61, pp. 266–274, 2017.

[3] J. Li, L. Sun, Q. Yan, Z. Li, W. Srisa-An, and H. Ye, “Significant Permission Identification for Machine-Learning-Based

Android Malware Detection," IEEE Trans. Ind. Informatics, vol. 14, no. 7, pp. 3216–3225, 2018.

[4] A. Saracino, D. Sgandurra, G. Dini, and F. Martinelli, "MADAM: Effective and Efficient Behavior-based Android Malware Detection and Prevention," IEEE Trans. Dependable Secur. Comput., vol. 15, no. 1, pp. 83–97, 2018.

[5] S. Arshad, M. A. Shah, A. Wahid, A. Mehmood, H. Song, and H. Yu, "SAMADroid: A Novel 3-Level Hybrid Malware Detection Model for Android Operating System," IEEE Access, vol. 6, pp. 4321–4339, 2018.

[6] T. Kim, B. Kang, M. Rho, S. Sezer, and E. G. Im, "A Multimodal Deep Learning Method for Android Malware Detection using Various Features," vol. 6013, no. c, 2018.

[7] A. Martin, F. Fuentes-Hurtado, V. Naranjo, and D. Camacho, "Evolving Deep Neural Networks architectures for Android malware classification," 2017 IEEE Congr. Evol. Comput. CEC 2017 - Proc., pp. 1659–1666, 2017.

[8] X. Su, D. Zhang, W. Li, and K. Zhao, "A Deep Learning Approach to Android Malware Feature Learning and Detection," 2016 IEEE Trust., pp. 244–251, 2016.

[9] K. Zhao, D. Zhang, X. Su, and W. Li, "Fest : A Feature Extraction and Selection Tool for Android Malware Detection," 2015 IEEE Symp. Comput. Commun., pp. 714–720, 4893.

[10] A. Feizollah, N. B. Anuar, R. Salleh, and A. W. A. Wahab, "A review on feature selection in mobile malware detection," Digit. Investig., vol. 13, pp. 22–37, 2015.

[11] A. Firdaus, N. B. Anuar, A. Karim, M. Faizal, and A. Razak, "Discovering optimal features using static analysis and a genetic search based method for Android malware detection *," vol. 19, no. 6, pp. 712– 736, 2018.

[12] A. V. Phan, M. Le Nguyen, and L. T. Bui, "Feature weighting and SVM parameters optimization based on genetic algorithms for classification problems," Appl. Intell., vol. 46, no. 2, pp. 455–469, 2017.