International Journal of Innovative Technology and Exploring Engineering (IJITEE)
ISSN: 2278-3075 (Online), Volume-9 Issue-12, October 2020

Deep-Droid: Deep Learning for Android Malware Detection

Ahmed Hashem El Fiky

Abstract*: Android OS, which is the most prevalent operating system (OS), has enjoyed immense popularity for smart phones over the past few years. Seizing this opportunity, cybercrime will occur in the form of piracy and malware. Traditional detection does not suffice to combat newly created advanced malware. So, there is a need for smart malware detection systems to reduce malicious activities risk. Machine learning approaches have been showing promising results in classifying malware where most of the method are shallow learners like Random Forest (RF) in recent years. In this paper, we propose Deep-Droid as a deep learning framework, for detection Android malware. Hence, our Deep-Droid model is a deep learner that outperforms exiting cutting-edge machine learning approaches. All experiments performed on two datasets (Drebin-215 & Malgenome-215) to assess our Deep-Droid model. The results of experiments show the effectiveness and robustness of Deep-Droid. Our Deep-Droid model achieved accuracy over 98.5%.

***Keywords*: Android Malware; Malware Detection; Static Analysis; Deep-Droid; Deep-Learning.**

I. INTRODUCTION

In recent years, Mobiles and tablets have come to prominence. The number of dynamic cell phones worldwide was about 7 billion, and in the countries in which they were created, the ratio between individuals and cell phones was valued at 120.8% In the end of 2016. Due to its massive circulation as well as its capabilities, over the last two years' smart phones have become the basic focus of attackers. Android is a Google's open-source framework (OS), now has the biggest piece of the cake, or in other words, 80%. Due to its responsiveness and popularity, Android [1] is the basic goal of the attackers against smart phones (98.5%), in the range of the more than a million malicious apps.

Malware [2] is the primary carrier of security attacks against smart phones. Malware can be blind and obfuscate, they hide complex code that performs out of sight activities that undermine customer protection, device respect, or even customer credit. Many the usual forms of knock out malicious Android apps are the transfer of contacts, login certificates, instant messages, or malicious purchases in the customer to expensive premium departments. Hence, all these problems can be done on Android phones without the customer notification. The primary concern in the Android development stage is the malware website. The malware detection strategies in the Android stage are similar to the systems used at any stage. Malware detection is basically divided into static detection, by examining Android app without running; Dynamic detection, by analyzing runtime behavior, for example: memory, battery, and system usage of the tool; Or the examination of half and half, by joining Dynamic and Static detection. Machine learning methods have shown promising results in categorizing Android malware. The methods can provide a rewarding degree of accuracy and overcome the limitations of traditional methods for Android Malware. Machine learning methods such as Decision Tree (DT) Support Vector Machine (SVM), and Logistic Regression (LR) have been previously suggested for malware detection [3]. Currently, Neural networks are in wide use for many applications entitlement to flexibility in architecture design and extremely non-linear systems capacity. In this paper, we introduce a deep learning neural network framework for Android malware detection called Deep-Droid. Our contributions include:

- We are conducting a comprehensive assessment with rigorous beta setup to assess Deep-Droid's performance with two sets of Android application data available to the public in the real world (Malgenome Project [4] and Drebin [5]).
- Deep-Droid provides low false positive rate, high true positive rate and high accuracy based on deep learning neural network framework.

The rest of the paper is summarized as follows in Section 2 provides a summary of relevant past research. Section 3 describes the model proposed, and in Section 4, the dataset of benign and malware Android applications is presented and experimental results, and Section 6 concludes the paper and points to future work.

II. RELATED WORK

Traditional methods have been applied to categorize Android malware detection. Detection by Signature is the most widely used by anti-malware system. Recognizes the malware sample by looking at specific byte sequences (called signatures) in an object to investigate matching with known signatures from blacklisted malware. This method of detection is not effective against "zero-day attacks" as the system is configured based on known malware signatures [6].

Revised Manuscript Received on October 30, 2020.
* Correspondence Author
Ahmed Hashem El Fiky *, Systems and Computer Engineering Dept., Faculty of Engineering, Al-Azhar University, Cairo, Egypt. Email: 0x4186@gmail.com

Detection by Signature has been suggested for detection Android Malware that makes use of signature matching algorithms. A signature-based extension has been suggested that combines signature and anomaly approaches. The combined approach achieved 96% accuracy in categorizing malicious applications by using 3 different data sets [7].

In [8] the authors suggested detecting malware for Android apps. using the API call sequence and constructing a convolutional neural network model (CNN) for classification. They did better with CNN compared to other n-gram methods called LSTM (Repetitive Neural Network Model).

The authors proposed in [9] a fully connected deep learning model for Android malware detection. The proposed model is a Static Analysis. They used the name of the Android malware, and the release packages with an installed as a main features and they achieved high accuracy 94.65%.

In [10] the authors proposed Droid-NNet is a deep learning framework for categorizing Android malware. Whereas, their Droid-NNet method is a deep learner who outperforms current state-of-the-art machine learning methods. All experiments performed on two datasets (Drebin-215 & Malgenome-215) Android apps to assess Droid-NNet. They achieved high accuracy for Malgenome dataset of 99.3% and Drebin dataset of 98.8% compared to newer machine learning classifiers such as DT, SVM, and LR.

The authors focus in [11] on discovering malware that can install through permissions on Android, using a deep neural network model. Their proposed model detects malware-dependent permissions on Android APKs in real time with more than 85% accuracy for a dataset of 398 APK files.

III. PROPOSED METHODOLOGY

Sequential neural networks are currently widely used for many applications due to the flexibility in architecture design and the ability of non-linear systems. The essential architecture of a serial neural network includes input layers, one or more hidden layers, and output layers where each layer contains a certain number of neurons. To capture nonlinear data, a nonlinear method, called the activation function, can be applied to weighted aggregates of neurons. All neural network weights are set to random values in the initial phase of the training phase. The data is entered into the input layer in the network, then it transfers through the hidden layers, and finally the output is produced in the output layer. The network is constantly updating the weights by applying backpropagation-based on the outputs and the desired target of the neural network. Thus the network reduces the error between output and target in each iteration [12]. In this phase, the loss function is used to calculate the network error and reduce the error by the optimization function during backpropagation. Deep-Droid is a framework for a sequential neural network that we propose it in this paper. Figure 1 shows the Deep-Droid architecture, a chained neural network that contains 3 layers: an input layer, a hidden layer, and an output layer to classify Android applications as benign or malware. The input layer includes 215 neurons (number of application features), the hidden layer includes 25 neurons, and finally, the output layer contains only one neuron because the problem is binary classification. We applied binary entropy as a loss function and an optimizer to estimate the adaptive moments (ADAM) to calculate the error and update the network weights. Also, Figure 1 consists of the training phase and testing phase. In the training phase, both malicious and benign software samples are identified through a serial neural network model. To train the model, Melgenome and Drebin datasets were used which consist of 18,835 APK files with 215 features. In the testing phase, the neural network model is tested using data collected from Google Play Store and the Virusshare website that contains both benign and malicious APK files.

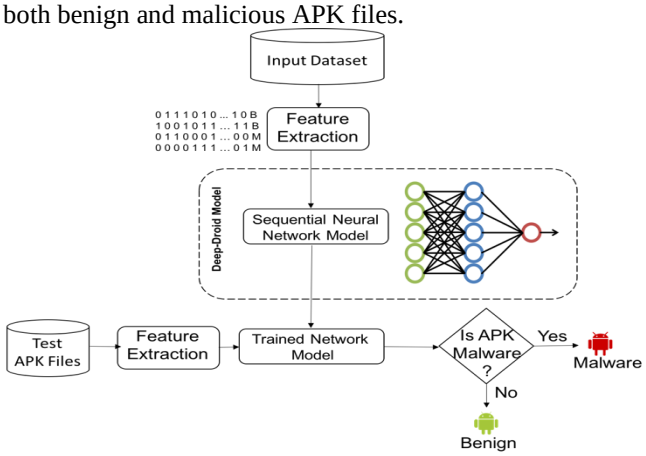

Fig. 1 Deep-Droid Architecture

IV. EXPERIMENTAL & RESULTS

A. Dataset Description

All experiments performed on two datasets (Drebin-215 & Malgenome-215) of Android apps to evaluate our proposed mode Deep-Droid. In Table I, we are shown the details of each dataset. Malgenome-215 dataset is collected from the supplementary section of [12]. Malgenome dataset has a total of 3,799 app samples, where 2,539 benign samples and 1,260 malware samples from the Android malware genome project [4]. Drebin-215 dataset is publicly available, it consists of 15,036 app samples, where 9,476 are benign samples and the remaining 5,560 are malware samples from the Drebin project [5]. Both datasets involve 215 features. Some statistical about features are shown in Table II where there are four categories
from features Manifest Permission, Intents, Commands, and API- Calls. Each category contains specific number of feature such Manifest Permission contains 113 features like WRITE_SETTINGS and SET_TIME, API-Calls contains 73 features like URLClassLoader and PathClassLoader.

Table I Dataset's details

Datasets	#samples	#malware	#benign	#features
Malgenome-215	3,799	1,260	2,539	215
Drebin-215	15,036	5,560	9,476	215
Malgenome+Drebin	**18,835**	**6,820**	**12,015**	**215**

Table II Samples of Features

Feature Name	Feature Category	#Features
WRITE_SETTINGS SET_TIME ADD_VOICEMAIL …	Manifest Permission	113

intent.action.BOOT_COMPELETED intent.action.TIME_SET intent.action.ACTION_SHUTDOWN …	Intents	23
remount chown …	Commands	6
URLClassLoader PathClassLoader …	API Calls	73

B. Model Evaluation Metrics

Both sets of data that we used in this paper are unbalanced. The ratio between benign and malicious software samples in the Malgenome-215 dataset ranges from approximately 66% to 33% and in the Drebin-215 dataset approximately from 63% to 37%. Therefore, we cannot consider the scale of accuracy to assess the performance of the models. Consequently, Table III shown the performance metrics that used in evaluation [12].

Table III Performance Metrics

Performance metric	Formula
TPR (True Positive Rate) / Recall	TP/(TP+FN)
FPR (False Positive Rate)	FP/(FP+TN)
Precision	TP/(TP+FP)
F-Measure	(2.Precision.Recall)/(Precision+Recall)

Where:

- TP: represents the number of Android applications as benign.
- TN: represents the number of Android applications as malware.
- FP: represents the number of benign Android applications as malware.
- FN: represents the number of malware Android applications as benign.

C. Experimental Design

The performance of our proposed model assessed by comparing it with the performance of the RF, SVM, and KNN methods. To check the consistency of the proposed model, we tested each of the models with 10-fold validation. The proposed model is a sequential neural network. We used the “ReLu” activation function in the hidden layer and the “Sigmoid” function in the output layer. 'Adam' and 'binary entropy' were used to improve the loss function respectively. Experiments were performed on a MacBook Pro at a speed of 2.8 GHz Intel Core i7 with 16.0 GB RAM. We implemented our experiment with the Keras framework in Python 3.7.

D. Experimental Results

The results of our proposed model Deep-Droid compared with the other 3 classifiers. An F-measure was used to evaluate the performance of the models. Cross validation was used 10-fold for each trial. The same settings were applied to all of the Drebin-215 and Malgenome-215 dataset to maintain consistency.

1) Performance evaluation on Drebin-215 dataset:

Our proposed model Deep-Droid trained for 100 epochs with an early stop. All classifiers were trained on 90% of the dataset and Deep-Droid tested on the remaining 10% of the dataset. Table IV shows the experimental results for the different classifiers in this dataset.

Table IV Drebin-215 dataset classifiers results

Classifiers	F-measure	TPR	FPR
RF	0.985	0.976	0.003
SVM	0.959	0.939	0.012
KNN	0.983	0.985	0.011
Deep-Droid	0.987	0.978	0.011

2) Performance evaluation on Malgenome-215 dataset:

The testing and training segmentation ratio for all classifiers was 10% and 90%. Experimental results of implementing our proposed model Deep-Droid and the other 3 algorithms on the Malgenome-215 dataset are shown in Table V.

Table V Malgenome-215 dataset classifiers results

Classifiers	F-measure	TPR	FPR
RF	0.990	0.981	0.001
SVM	0.966	0.942	0.004
KNN	0.989	0.991	0.007
Deep-Droid	0.992	0.983	0.007

3) Performance evaluation on combination of two datasets (Drebin-215 and Malgenome-215):

The testing and training segmentation ratio for all classifiers was 10% and 90%. Experimental results of implementing our proposed model Deep-Droid and the other 3 algorithms on a combination of two datasets (Drebin-215 and Malgenome-215 dataset) are shown in Table VI.

Table VI Drebin-125 and Malgenome-215 dataset classifiers results

Classifiers	F-measure	TPR	FPR
RF	0.985	0.976	0.003
SVM	0.955	0.934	0.013
KNN	0.984	0.985	0.010
Deep-Droid	**0.989**	**0.982**	**0.011**

The results obtained are compared with the results obtained from other previous research studies in several different ways based on static analysis to detect Android malware. In particular, we consider [9] and [11] to be the best published references to our knowledge. Table VII shows a comparison between the results of the current work and the published results of the references mentioned above.

Table VII Comparison between proposed model and other works

Reference	Dataset		F-Measure
	Benign	Malware	
[9]	NA	NA	0.946
[11]	398		0.87
This work	**12,015**	**6,820**	**0.989**

V. CONCLUSION

Android Malware is increasingly a serious security threat to users of smart phones. It is an essential to develop a smart Android malware detection solution to reduce the risk of malicious activities. In this research, we suggested a framework based on sequential neural network called Deep-Droid.

All experiments performed on two datasets (Drebin-215 & Malgenome-215) to assess our model Deep-Droid. the performance of our model Deep-Droid evaluated by comparing it with that of the RF, SVM, and KNN approaches. The results of experimental show that our model Deep-Droid achieved the highest F-measure for each of the datasets and outperformed the other methods.

REFERENCES

1. R. Vinayakumar, K. P. Soman and P. Poornachandran, "Deep android malware detection and classification," 2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI), Udupi, 2017, pp. 1677-1683.
2. P. D. Ali and T. G. Kumar, "Malware capturing and detection in dionaea honeypot," 2017 Innovations in Power and Advanced Computing Technologies (i-PACT), Vellore, 2017, pp. 1-5. doi: 10.1109/IPACT.2017.8245158
3. Gavrilut, D., Cimpoesu, M., Anton, D. and Ciortuz, L. (2009) Malware Detection Using Machine Learning, Proceedings of the International Multiconference on Computer Science and Information Technology , 735-741.
4. Y. Zhou and X. Jiang, ”Dissecting android malware: Characterization and evolution” In proc. 2012 IEEE Symposium on Security and Privacy (SP), San Fransisco, CA, USA, 20-23 May, 2012 , pp. 95-109.
5. D. Arp, M. Spreitzenbarth, M. Hubner, H. Gascon, and K. Rieck, ”Drebin: Efficient and Explainable Detection of Android Malware in Your Pocket” In proc. 20th Annual Network & Distributed System Security Symposium (NDSS), San Diego, CA, USA, 23-26 Feb. 2014.
6. Mujumdar, A., Masiwal, G., & Meshram, D. B. (2013). Analysis of signature-based and behavior-based anti-malware approaches. International Journal of Advanced Research in Computer Engineering and Technology (IJARCET), 2(6).
7. Saracino, A., Sgandurra, D., Dini, G., & Martinelli, F. (2016). Madam: Effective and efficient behavior-based Android malware detection and prevention. IEEE Transactions on Dependable and Secure Computing, 15(1), 83-97.
8. Robin Nix, JianZhang: Classification of Android apps and malware using deep neural networks. In:2017 International Joint conference on Neural Networks(IJCNN), Jul 03,pp 216--4407 IEEE(2017)
9. S. HR, "Static Analysis of Android Malware Detection using Deep Learning," 2019 International Conference on Intelligent Computing and Control Systems (ICCS), Madurai, India, 2019, pp. 841-845, doi: 10.1109/ICCS45141.2019.9065765.
10. M. Masum and H. Shahriar, "Droid-NNet: Deep Learning Neural Network for Android Malware Detection," 2019 IEEE International Conference on Big Data (Big Data), Los Angeles, CA, USA, 2019, pp. 5789-5793, doi: 10.1109/BigData47090.2019.9006053.
11. S. P., K. P. B., A. K. K. and A. T., "Detection of Permission Driven Malware in Android Using Deep Learning Techniques," 2019 3rd International conference on Electronics, Communication and Aerospace Technology (ICECA), Coimbatore, India, 2019, pp. 941-945, doi: 10.1109/ICECA.2019.8821811.
12. Yerima, Suleiman Y., and Sakir Sezer. "Droidfusion: A novel multilevel classifier fusion approach for Android malware detection." IEEE transactions on cybernetics 49.2 (2018): 453-466.

AUTHORS PROFILE

Ahmed Hashem El Fiky received the BSc degree in Computer Engineering Department Faculty of Engineering Helwan University May 2012 (Grade: Excellent and the first with honors) and Post-Grad Diploma Degree in Computer Engineering May 2015 (Grade: Very Good). He is worked as a Teaching Assistant of Faculty of Engineering Helwan University Computer Engineering Department for 5 years ago (2013-2018). He is the author of several articles paper. Currently, He is an Information Security Team Leader at Tanmeyah Micro-Enterprise Services Company, Cairo, Egypt. and Also, He is a MSc student in Systems and Computer Engineering department of Faculty of Engineering Al-Azhar University. His main research interests are focused on Information Security, Network Security, Cryptography, Reverse Engineering, Malware Analysis, Digital Forensic and Space Science.