TrojanDroid: Android malware detection for Trojan discovery using convolutional neural networks

Saeed Seraj, Michalis Pavlidis, and Nikolaos Polatidis

School of Architecture, Technology, and Engineering, University of Brighton, BN2 4GJ, Brighton, U.K
{S.Seraj@Brighton.ac.uk, M.Pavlidis@Brighton.ac.uk N.Polatidis@Brighton.ac.uk}

Abstract. Android platforms are widely used nowadays in different forms such as mobile phones and tablets, and this has made the Android platform an attractive target for hackers. While there are many solutions available for detecting malware on Android devices there aren't that many that are concentrated on specific malware types. To this extent, this paper delivers a new dataset for Trojan detection for Android apps based on the permissions of the apps, while the second contribution is a neural network architecture that can classify with very high accuracy if an Android app is a genuine app or a Trojan pretending to be a normal app. We have run extensive evaluation tests to validate the performance of the proposed method and we have compared it to other well-known classifiers using well-known evaluation metrics to show its effectiveness.

Keywords: Android, Malware detection, Trojan, Convolutional Neural Networks.

1 Introduction

Nowadays, in the world people can get all types of Android devices such as mobile phones and tablets and numerous applications (apps) can be easily downloaded from available websites in cyberspace. However, many apps are being produced daily, with some of which being infected and being malware instead of a genuine app. Many exploiters infect applications using malicious approaches for their profit to steal information from mobile devices. Malware can come in various forms, such as viruses, trojans, worms, botnets, and many others and among that malware, trojans are a type of malware that is often disguised as legitimate software; however, they will perform malicious activities on the operating system that most of the users will not even notice or understand [1, 6, 18].

Therefore, in this article, we study how to detect Android Trojans using the permissions of the applications. To do this we have collected and processed data and created a new dataset that is described in detail in section 3. The Trojan dataset is a classification dataset that contains only Trojan and genuine Android applications and to this extent, we have developed a Convolutional Neural Network (CNN) architecture that detects Trojans with very high accuracy. To achieve this, we first had a theory that a Trojan can be identified based on the requested permissions during app installations.

The contributions of the paper are as follows:

- We introduce a novel dataset for Android Trojan detection based on the permissions of the applications.
- We deliver a CNN neural network architecture for Trojan detection.

The rest of the paper is organized as follows: Section 2 is the related work, section 3 describes the dataset, section 4 explains the proposed method, section 5 delivers the experimental evaluation and section 6 contains the conclusions.

2 Related work

In the Android platform, there are several works available in the literature due to its popularity and numerous malware, that exist. We have identified recent relevant works and discussed them here but the related works that are about Trojan detection are non-existent in the literature. To the best of our knowledge, there is only one related work about Trojan detection that is based on dynamic analysis and not on permissions which are discussed later in [2].

We start with MCDM which is 'a multi-criteria decision-making based' mobile malware detection system that uses a risk-based fuzzy analytical hierarchy process (AHP) approach to evaluate the Android mobile applications. This research concentrated on static analysis by using permission-based features to assess the Android mobile malware detection system approach [1]. In another research dynamic analysis was used to detect their features. Therefore, a parameter such as a system call was investigated in this study. The purpose of this research is to detect android Trojan based on dynamic analysis [2]. Another research paper proposes a novel detection technique called PermPair that builds and compares the graphs for malware and normal samples by extracting the permission pairs from the manifest file inside the application [3]. Yet another research presents a platform named DroidCat which is a novel dynamic application classification model to complement those methods that are existing. DroidCat uses various sets of dynamic features based on method calls and inter-component communication (ICC) Intents without involving any permission, application resources, or system calls [4]. One other study proposes an innovative Android malware detection framework based on feature weighting with the joint optimization of weight-mapping and the parameters of the classifier named JOWMDroid [5].

In [6] the study introduces a new scheme for Android malware detection and familial classification based on the Graph Convolutional Network (GCN). The general idea is to map Android applications and APIs into a large heterogeneous graph and convert the original problems into a node classification task. The study in [7] was a novel hybrid-featured Android dataset that provides timestamps for each data sample which covers all years of Android history from the years 2008 to 2020 and considers the distinct dynamic data sources. Researchers presented a new malware detection framework for Android applications that are evolutionary 'HAWK'. Their model can pinpoint rapidly the proximity between a new application and existing applications and

assemble their numerical embeddings under different semantics as described in [8]. MAPAS is a malware detection platform that achieved high accuracy and adaptable usage of computing resources. Moreover, MAPAS analyzed malicious apps behaviors based on API call graphs of them by using convolution neural networks (CNN) [9]. NSDroid is 'a time-efficient malware multi-classification approach based on neighborhood signatures in local function call graphs (FCGs). This method uses a scheme based on neighborhood signature to calculate the similarity of the different applications which is significantly faster than traditional approaches according to subgraph isomorphism [10]. A work that presents a web-based framework that helped to detect malware from Android devices is named 'MLDroid'. The proposed framework detects Android malware applications by performing its dynamic analysis measures can be found in [11].

In their work 'NATICUSdroid' a new Android malware detection system that investigates and classifies benign and malware using statistically selected native and custom Android application permissions as features for various machine learning classifiers [12]. One more work is an innovative android malware detection framework that uses a deep CNN neural network. In this system, Malware classification is performed based on static analysis of the raw opcode sequence from a disassembled program [13]. A machine learning-based malware detection platform is proposed to distinguish Android malware from benign applications. It is aimed to remove unnecessary features by using a linear regression-based feature selection approach at the feature selection stage of the proposed malware detection framework. [14]. Another research proposes a novel approach based on behavior for Android malware classification. In the proposed method, the Android malware dataset is decompiled to identify the suspicious API classes and generate an encoded list. In addition, this framework classifies unknown applications as benign or malicious applications based on the log-likelihood score generated [15]. In their paper researchers have delivered a completely novel and innovative dataset of malicious or benign Android anti-malware detection, including, and a customized multilayer perceptron neural network (MLP) that is being used to detect fake anti-malware that pretend to be genuine ones based on the permissions of the applications [16].

In their article researchers introduced a novel TAN (Tree Augmented naive Bayes)-based—a hybrid Android malware detection mechanism that involves the conditional dependencies which are required for the functionality of an application among relevant static and dynamic features [17]. The next work is a survey aimed to provide an overview of the way machine learning (ML) has been employed in the context of malware analyses. They also conducted survey papers based on their objectives, what kind of information about malware they used specifically, and what type of machine learning techniques they employed [18]. DAE is a hybrid model based on a deep autoencoder and a CNN. This mechanism is proposed to improve the Android malware detection accuracy. To achieve this, they reconstructed the high-dimensional features of Android applications and employed multiple CNN to detect Android malware [19]. In the next research article, a new detection approach is introduced based on deep learning techniques to detect Android malware from trusted applications. To achieve that, they treat one system call sequence as a sentence in the language and build a classifier according to the Long Short-Term Memory (LSTM) language model [20].

An EfficientNet-B4 CNN-based model is presented for Android malware detection by employing image-based malware representations of the Android DEX file. This model extracts relevant features from the Android malware images [21]. In the following paper, a new classifier fusion scheme based on a multilevel architecture is introduced that enables an effective combination of machine learning algorithms for improved accuracy which is called DroidFusion. The induced multilevel model can be utilized as an improved accuracy predictor for Android malware detection [22]. A Machine Learning-based method that utilizes more than 200 features extracted from both static analysis and dynamic analysis of Android applications for malware detection was proposed in [23]. A platform that is capable to detect android malware applications is introduced to support the organized Android Market. The proposed framework intended to develop a machine learning-based malware detection framework on Android to detect malware applications and to increase the security and privacy of smartphone users [24]. CoDroid is a hybrid Android malware detection approach based on the sequence which utilizes the sequences of static opcode and dynamic system call [25]. Finally, researchers have combined the high accuracy of the traditional graph-based method with the high scalability of the social network analysis-based approach for Android malware detection [26].

Although all these works are interesting there is only one work that is about Trojan detection that is based on dynamic analysis. Therefore, in this work, we developed a new dataset about Trojan detection using permissions to fill this gap and we have developed a CNN architecture to detect trojans with high accuracy.

3 Dataset

With regards to trojan detection in Android platforms, we introduce a new dataset based on Android app permissions. To this extent, we developed an Android Trojan dataset that contains 2593 entries. To do this we downloaded 1058 Android Trojan malware and 1535 general benign apps from various categories from Google Play. We analyzed all apk files using VirusTotal.com to extract all their features including internet access and other required app permissions. Moreover, we have used over 70 reputed anti-malware detection engines to classify the apk files. The android Trojan dataset consists of the following families: BankBot, Binv, Citmo, FakeBank, LegitimateBankApps, Sandroid, SmsSpy, Spitmo, Wroba, ZertSecurity and Zitmo. For the dataset to be in a usable form, we added all the information in a file.csv file format which can be easily opened and processed. There is a total number of 450 columns in the dataset that includes 449 specific permissions plus the label which is the last column. The first row in the dataset describes column titles, and the rest are features from 2593 android Trojans and benign applications apk files. All values are in binary format i.e., 0 or 1. When an app requires permission, then the value in the respective entry of the dataset is 1, and unnecessary permissions of an app are set to zero. An Android app that is recognized as malware by most antivirus companies based on VirusTotal report, is considered risky and the value in the label column is set as 1, being Trojan. However, the other Android genuine apps have zero value. The complete dataset is accessible at: https://www.kaggle.com/saeedseraj/trojandroidpermissionbased-android-trojan-dataset/

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	internet	access	access	get	change	write	read	system	c2d	camera	call	bluetooth	bluetooth
2	1	1	1	0	0	1	0	0	1	0	0	0	0
3	1	1	1	0	1	1	1	0	0	0	0	1	1
4	1	0	1	0	0	0	0	1	0	0	0	0	0
5	1	1	1	0	0	1	0	0	1	0	0	0	0
6	1	1	1	0	0	1	0	0	1	0	0	0	0
7	1	0	1	0	0	0	0	0	0	1	0	0	0
8	1	0	1	0	0	0	0	0	1	0	0	0	0

Figure 1. An illustration of a small part of the proposed dataset

4 Proposed method

A 1-dimensional CNN sequential architecture has been developed to classify trojans using the above dataset and the Python programming language with the Keras library. The architecture includes one 1D-CNN layer, followed by a 1D MaxPooling layer, followed by a Flatten layer, followed by 2 dense layers. The architecture is presented in detail in figure 2. The Specific settings are as follows:

- A learning rate of 0.01 has been used and the optimizer is Adam
- The number of epochs is 6
- The batch size is 16
- The activation functions used are the Relu for the 1D CNN layer and the first dense layer and the Sigmoid for the final dense layer
- Bias has been set to true in the 1D CNN layer

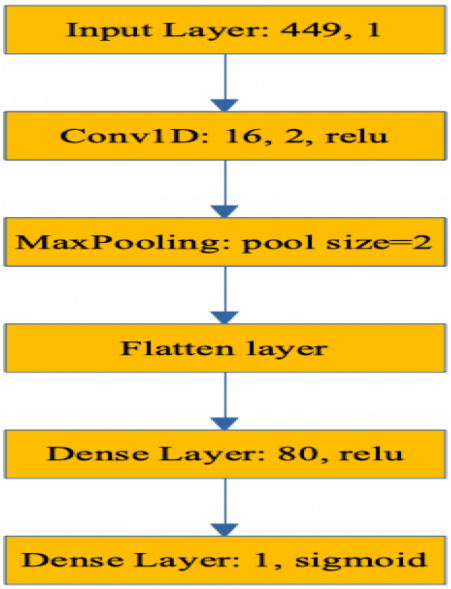

Figure 2. Proposed method architecture

5 Experimental evaluation

For the experimental evaluation, we have proposed the CNN architecture described in section 4 developed using the Python programming language and the Keras library. For all experiments, 5-fold cross-validation has been used.

5.1 Evaluation metrics

For the experimental evaluation, we have used the Python programming language and the Keras machine learning library. With regards to evaluation metrics, we have used the Accuracy, Precision, Recall, and F1 which are described in equations 1, 2, 3, and 4 respectively. TP stands for true positive, TN for true negative, FP for false positive, and FN for false negative. Accuracy, which is equation 1, shows the overall performance. Another significant metric is Precision which describes the portion of predicted Trojans and is calculated by equation 2. Equation 3 explains the Recall metric which is the portion of Trojan that is correctly classified. The F1-score is a number between 0 and 1 and is the harmonic mean of precision and recall which is computed according to equation 4. These are well-known metrics that have been used in recent studies for similar problems in Android malware detection [5, 6, 21]. Overall, our proposed method outperforms alternative classifiers in all metrics.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

$$Recall = \frac{TP}{TP + FN} \tag{3}$$

$$F1 = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{4}$$

5.2 Results

This section delivers the results of the experimental evaluation. Figure 3 presents the results of the proposed method architecture for both the train and test accuracy over 6 epochs. Figure 4 presents the loss results over 6 epochs.

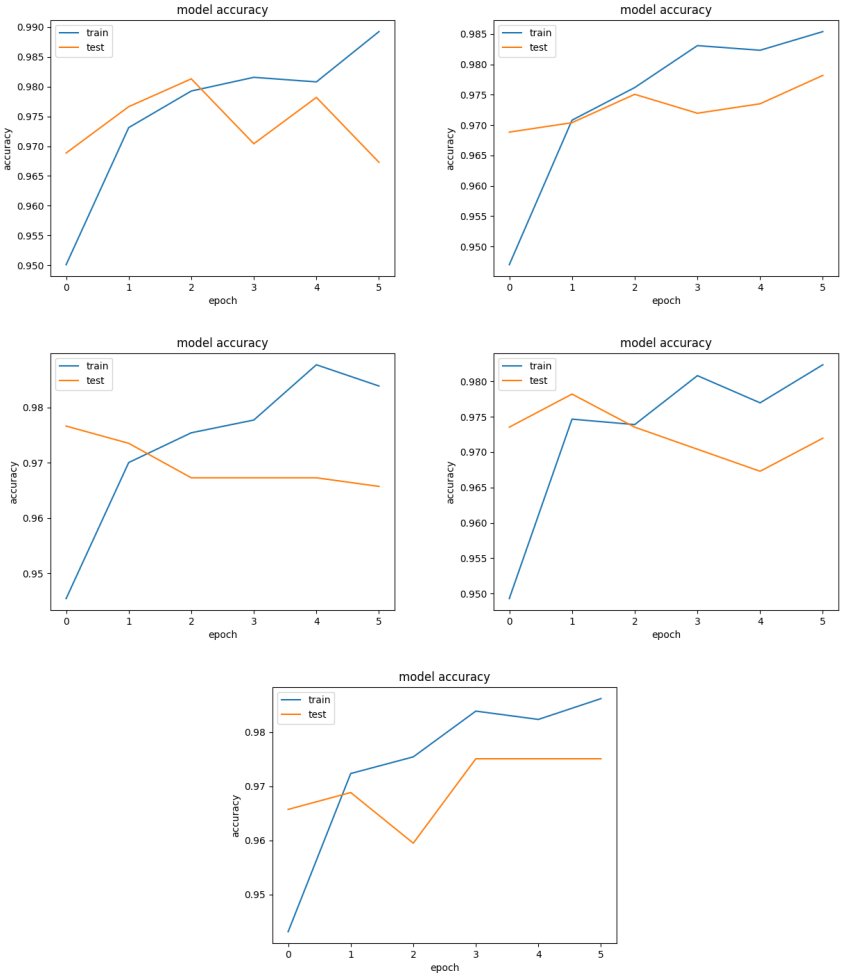

Figure 3. Accuracy for each of the 5 folds

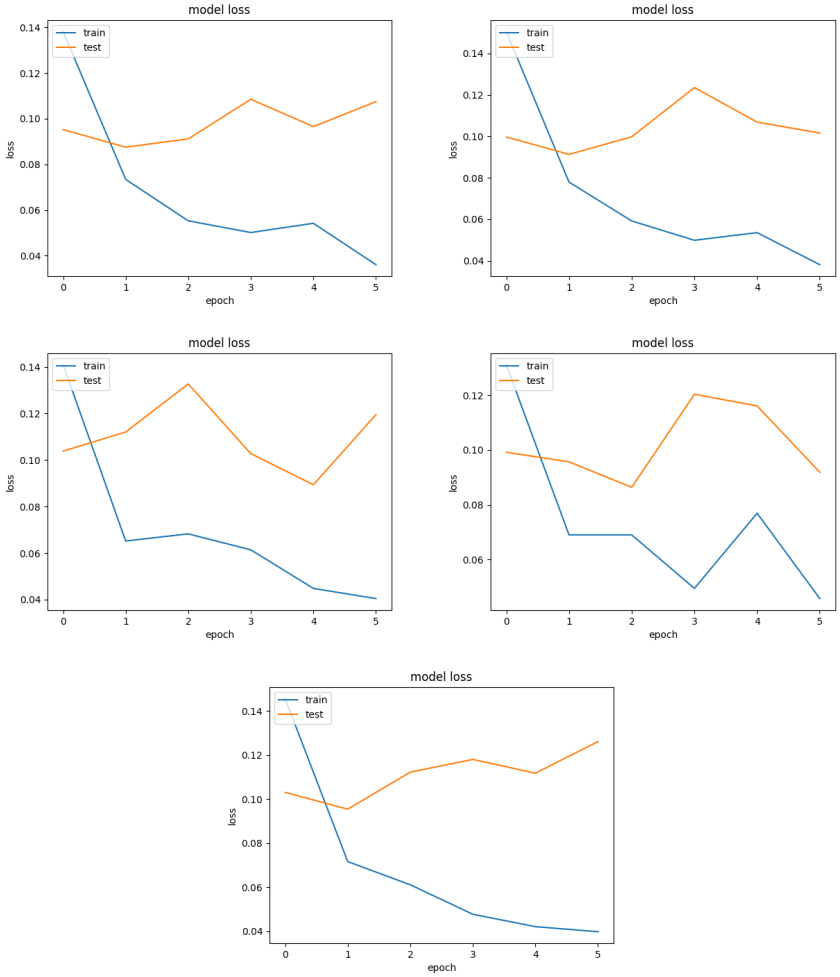

Figure 4. Loss for each of the 5 folds

5.3 Comparisons with other classifiers

The algorithms used in the comparisons are the following with the default settings used from the sci-kit learn library: Decision Tree, Random Forest, Multi-Layer Perceptron. The results are presented in Table 1 which provides a comparison between the proposed method and the other well-known classifiers using accuracy, precision, recall, and F1. 5-fold cross-validation has been used throughout.

Algorithm	Accuracy	Precision	Recall	F1
Decision Tree	96.1%	95.8%	95.7%	95.9%
Random Forest	97.9%	97.7%	97.3%	97.5%

Multi-Layer Perceptron	97.8%	97.8%	96.7%	97.7%
TrojanDroid	**98.06%**	**99%**	**97.71%**	**98%**

Table 1. Comparison results

6 Conclusions

In this paper, we have concentrated on Trojan detection on Android platforms. We have collected a new dataset which we have made available, and we delivered a novel neural network architecture that can detect trojans with very high accuracy. The results indicate that by using the permissions of Trojan and genuine Android apps, trojans can be detected in a straightforward way which can be useful to the research community and beyond.

In the future, we plan to extend our proposed method to include the specific trojan family that a trojan belongs to and adjust it accordingly to detect the family with high accuracy as well. Moreover, we plan to investigate how to use permissions to detect other types of malware such as botnets.

References

1. Arif, J. M., Ab Razak, M. F., Mat, S. R. T., Awang, S., Ismail, N. S. N., & Firdaus, A. (2021). Android mobile malware detection using fuzzy AHP. Journal of Information Security and Applications, 61, 102929.
2. Aminuddin, N. I., & Abdullah, Z. (2019). Android trojan detection based on dynamic analysis. Advances in Computing and Intelligent System, 1(1).
3. Arora, A., Peddoju, S. K., & Conti, M. (2019). Permpair: Android malware detection using permission pairs. IEEE Transactions on Information Forensics and Security, 15, 1968-1982.
4. Cai, H., Meng, N., Ryder, B., & Yao, D. (2018). Droidcat: Effective android malware detection and categorization via app-level profiling. IEEE Transactions on Information Forensics and Security, 14(6), 1455-1470.
5. Cai, L., Li, Y., & Xiong, Z. (2021). JOWMDroid: Android malware detection based on feature weighting with joint optimization of weight-mapping and classifier parameters. Computers & Security, 100, 102086.
6. Gao, H., Cheng, S., & Zhang, W. (2021). GDroid: Android malware detection and classification with graph convolutional network. Computers & Security, 106, 102264.
7. Guerra-Manzanares, A., Bahsi, H., & Nõmm, S. (2021). KronoDroid: Time-based hybrid-featured dataset for effective android malware detection and characterization. Computers & Security, 110, 102399.
8. Hei, Y., Yang, R., Peng, H., Wang, L., Xu, X., Liu, J., ... & Sun, L. (2021). Hawk: Rapid android malware detection through heterogeneous graph attention networks. IEEE Transactions on Neural Networks and Learning Systems.

9. Kim, J., Ban, Y., Ko, E., Cho, H., & Yi, J. H. (2022). MAPAS: a practical deep learning-based android malware detection system. International Journal of Information Security, 1-14.
10. Liu, P., Wang, W., Luo, X., Wang, H., & Liu, C. (2021). NSDroid: efficient multi-classification of android malware using neighborhood signature in local function call graphs. International Journal of Information Security, 20(1), 59-71.
11. Mahindru, A., & Sangal, A. L. (2021). MLDroid—framework for Android malware detection using machine learning techniques. Neural Computing and Applications, 33(10), 5183-5240.
12. Mathur, A., Podila, L. M., Kulkarni, K., Niyaz, Q., & Javaid, A. Y. (2021). NATICUSdroid: A malware detection framework for Android using native and custom permissions. Journal of Information Security and Applications, 58, 102696.
13. McLaughlin, N., Martinez del Rincon, J., Kang, B., Yerima, S., Miller, P., Sezer, S., ... & Joon Ahn, G. (2017, March). Deep android malware detection. In Proceedings of the seventh ACM on conference on data and application security and privacy (pp. 301-308).
14. Şahin, D. Ö., Kural, O. E., Akleylek, S., & Kılıç, E. (2021). A novel permission-based Android malware detection system using feature selection based on linear regression. Neural Computing and Applications, 1-16.
15. Sasidharan, S. K., & Thomas, C. (2021). ProDroid—An Android malware detection framework based on profile hidden Markov model. Pervasive and Mobile Computing, 72, 101336.
16. Seraj, S., Khodambashi, S., Pavlidis, M., & Polatidis, N. (2022). HamDroid: permission-based harmful android anti-malware detection using neural networks. Neural Computing and Applications, 1-10.
17. Surendran, R., Thomas, T., & Emmanuel, S. (2020). A TAN based hybrid model for android malware detection. Journal of Information Security and Applications, 54, 102483.
18. Ucci, D., Aniello, L., & Baldoni, R. (2019). Survey of machine learning techniques for malware analysis. Computers & Security, 81, 123-147.
19. Wang, W., Zhao, M., & Wang, J. (2019). Effective android malware detection with a hybrid model based on deep autoencoder and convolutional neural network. Journal of Ambient Intelligence and Humanized Computing, 10(8), 3035-3043.
20. Xiao, X., Zhang, S., Mercaldo, F., Hu, G., & Sangaiah, A. K. (2019). Android malware detection based on system call sequences and LSTM. Multimedia Tools and Applications, 78(4), 3979-3999.
21. Yadav, P., Menon, N., Ravi, V., Vishvanathan, S., & Pham, T. D. (2022). EfficientNet Convolutional Neural Networks-based Android Malware Detection. Computers & Security, 102622. Chicago
22. Yerima, S. Y., & Sezer, S. (2018). Droidfusion: A novel multilevel classifier fusion approach for android malware detection. IEEE transactions on cybernetics, 49(2), 453-466. Chicago

23. Yuan, Z., Lu, Y., Wang, Z., & Xue, Y. (2014, August). Droid-sec: deep learning in android malware detection. In Proceedings of the 2014 ACM conference on SIGCOMM (pp. 371-372).
24. Zarni Aung, W. Z. (2013). Permission-based android malware detection. International Journal of Scientific & Technology Research, 2(3), 228-234.
25. Zhang, N., Xue, J., Ma, Y., Zhang, R., Liang, T., & Tan, Y. A. (2021). Hybrid sequence-based Android malware detection using natural language processing. International Journal of Intelligent Systems, 36(10), 5770-5784. Chicago
26. Zou, D., Wu, Y., Yang, S., Chauhan, A., Yang, W., Zhong, J., ... & Jin, H. (2021). IntDroid: Android malware detection based on API intimacy analysis. ACM Transactions on Software Engineering and Methodology (TOSEM), 30(3), 1-32.