

Automated adversarial detection in mobile apps using API calls and permissions

Sanjaikanth E Vadakkethil Somanathan Pillai¹, Rohith Vallabhaneni², Srinivas A Vaddadi²,
Santosh Reddy Addula², Bhuvanesh Ananthan³

¹School of Electrical Engineering and Computer Science, University of North Dakota, Grand Forks, USA

²Department of Information Technology, University of the Cumberlands, Williamsburg, USA

³Department of Electrical and Electronics Engineering, PSN College of Engineering and Technology, Tirunelveli, India

Article Info

Article history:

Received Mar 19, 2024

Revised Sep 13, 2024

Accepted Oct 7, 2024

Keywords:

Android mobile apps
Application programming
interface calls
Deep learning
Permission
SD_ConvAE model
Third-party attacks

ABSTRACT

Android mobile phones' growing popularity has led to developers creating more malicious apps, which can be included in third-party arcades as protected applications. Detecting these malware applications is challenging due to time-consuming and high-cost techniques. This study proposes a robust deep learning (DL) model for detecting adversarial third-party apps using adaptive feature learning. The strategy involves preprocessing raw apk files, extracting permission behavioral features, and using the proposed spatial dropout-assisted convolutional autoencoder (SD_ConvAE) model to determine if the app is benign or malignant. The approach is simulated using a Python tool and assessed using various measures like accuracy, recall, weighted F-score (W-FS), false discovery rate (FDR), and kappa coefficient. The overall accuracies achieved by the developed techniques are about 99.6% and 99% for detecting benign and malignant apps, respectively.

This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.



Corresponding Author:

Sanjaikanth E Vadakkethil Somanathan Pillai

School of Electrical Engineering and Computer Science, University of North Dakota

Grand Forks, USA

Email: s.vadakkethil@und.edu

1. INTRODUCTION

In day-to-day life, smartphones are playing an essential part as they can provide similar advantages as desktop computers such as online shopping, online banking, website browsing, and social networking [1]. Moreover, additional features like location services, messaging facilities, updating data and worldwide access. However, this features and functionalities makes the smartphone more prone to malware attacks [2]. As a result, the third-part developers utilizes these advantages by fetching user confidential information, accessing chargers from the users with exclusion short messaging services (SMS) and calls. It is deliberated that more than 650K mobile applications (apps) are available on official android platform [3]. Moreover, the reports from Fortinet encompasses that more than 2K third-party apps are presented from 80 diverse families. The number of adversarial apps keeps on increasing because of open source and generously contemplated android customized manufactures [4].

The evolving of new malware families makes the technicians difficult in detecting third-part attacks using conventional mechanisms. Some common challenges are reported in [5] such as failure in visibility over mobile platforms, network anonymity, resource constrictions, and requires customized apps which uses remote capabilities. Several existing studies utilized permission to determine the third-party behaviors of an application [6]. However, it is complex to declare that the permissions being applied on the application code. There is some other traditional technique that uses only API call details to excerpt features using frequency components on a larger database [7]. But these techniques necessitates adaptive feature learning capability to

learn the malware features accurately [8]. Nowadays, artificial intelligence (AI) based deep learning (DL) techniques are becoming more popular among the researchers in identifying malware apps by utilizing complex API and permission level features over larger databases [9]. Hence, this research put forth a robust hybridized DL framework for detecting the adversarial third-party apps using complex malevolent features.

Motivation: due to rapid technological advancements, there have caused major enhancement in the networking capabilities and computing power over mobile phones. In today's scenario, smartphones, computers and tablets are considered as the powerful tool that combines larger wireless broadcasting networks. For performing multiple tasks, user uses mobile phones rather than the conventional computer systems. The improvements in user-computer interaction have paved the way to accessing mobile devices without any technical knowledge. As a result, numerous applications are present that are centralized to adversarial repositories. However, these apps are harmful that illegally gains user's confidential information by using API calls or permission level application codes. Detecting these kinds of third-party apps is highly challenging and needs effective techniques for learning the complex malevolent features. At present, DL techniques are providing fascinating performance in detecting adversarial attacks on mobile devices. These kinds of major concerns motivate to develop an innovative DL model for detecting the benign and malignant mobile apps effectively. The contributions are:

- To develop an innovative DL spatial dropout assisted convolutional autoencoder (SD_ConvAE) based technique to detect the third party mobile apps using malware behavioral features
- To preprocess the android apk files by performing binary vector conversion and to extract API calls and permission features present in the mobile applications.
- To present a robust SD_ConvAE model to detect whether the mobile apps are normal or malware.
- To validate the developed method with various conventional schemes by assessing several performance measures like accuracy, false discovery rate (FDR), recall, weighted F-measure (W-FM) and kappa coefficient for identifying both normal and malware apps.

The forthcoming sections are: the works associated to android malware detection using DL models are interpreted in section 2. The developed methodology is described in section 3. The outcomes are described in section 4. The conclusion of the developed study is presented in section 5.

2. RELATED WORKS

Millar *et al.* [10] defined the zero-day-based malevolent detection in android phones using the DL technique. In this study, a permission neural network with API calls convolutional neural networks (CNN) based Multiview DL model was introduced to extract and select hand-crafted features for detecting the malware apps. For the experimentation process, four different datasets namely Malgenome, debrin, and AMD, and publicly available Google Play Store datasets consisting of more than 28K samples with malignant and benign apps were considered. In analyzing the simulation part, the f-measure was computed and compared with other techniques. However, this technique faces high data redundancy problems due to a low effective feature selection (FS) scheme.

Mahindru and Sangal [11] put forth FS with machine learning (ML)-based detection schemes for analyzing malevolent apps accurately. Initially, API calls-based features were extracted and selected from the android apk files. Then, the least square support vector machine (LSSVM) technique was introduced to identify whether the apps were third-party or not. For the experimentation process, more than 2 lakh android samples were considered. In analyzing the simulation part, the accuracy, cost, and F-score were investigated and compared with different kernels. However, this technique faces high time complexity and overfitting issues while processing with larger malware apps.

Imtiaz *et al.* [12] introduced a deep artificial neural network (DANN) for detecting and identifying the android malevolents efficiently. Initially, min-max normalization was performed to rescale the random values into a fixed range. Then, FE was performed which extracts static and dynamic features based on network layers. Finally, the DANN model detects whether the applications were normal or malignant. But, this method failed to overcome the black box issues while dealing with permission and API calls.

Yadav *et al.* [13] defined a two-stage DL model for recognizing android malware based on malevolent images. Here, the efficientNetB0 model was introduced to detect mobile third-party apps accurately. Moreover, ML classifiers like linear support vector machine (LSVM) and RF technique were utilized to efficiently classify the various malignant attacks like Adsware, Adware, Trojan, and Spyware. In the simulation part, accuracy was analyzed and distinguished from other studies. However, this method had a high error as it failed to consider the effective features for the detection process.

Kim *et al.* [14] established a practical-oriented DL model for determining malware issues in mobile phones. In this work, CNN based DL model was introduced to detect third-party apps on android platforms. Moreover, API call-based features were considered for learning the malignant apps effectively. For the

simulation process, Google Play Store apps consisting of 10,000 android samples were utilized. In the simulation part, accuracy was analyzed and distinguished from other studies. However, this method causes high gradient explosion problems and overfitting issues when processing with larger samples.

In an effort to poison the adaptive face recognition system, Biggio and Zhu [15]-[17] made an attempt. The attacker's picture may be validated by inserting fraudulent data during the model update, which shifted the central value of the recognition feature in the model. Biggio *et al.* [18] launched assaults against SVM, a technique used for supervised learning. The experimental results demonstrate that the model classifier's test error may be substantially amplified as the gradient rises. To fool the model, the injected sample data must adhere to certain rules, and the attacker must own the injection point label. To test poisoning attacks on neural network learning algorithms, Yang *et al.* [19] ran an experiment. The suggested technique may double the speed of attack sample creation when compared to the direct gradient approach.

While it's true that a poisoning attack might cause the model to malfunction, the perpetrator must exert some effort to figure out how to introduce harmful data. A more prevalent technique, adversarial sample assault, may quickly lead models to the incorrect conclusion. It was Szegedy *et al.* [20] who first suggested the idea of adversarial samples. The perturbed samples will confidently lead the model to provide an inaccurate answer by intentionally altering the dataset in a small way. The initially properly categorised sample may move to the other side of the decision region and be reclassified into a different category if adversarial samples raise the model's prediction error. Adversarial samples may exploit existing models [21]-[24].

From the detailed literature review, the following research gaps are identified. Effectiveness of spatialized dropout in adversarial detection. Optimizing spatialized dropout: while spatialized dropout is intended to improve model generalization and robustness, research is needed to optimize its parameters specifically for adversarial detection in mobile apps. Studies could explore the impact of different dropout strategies on the detection accuracy of various adversarial techniques. Comparison with other regularization techniques: there is a need for comparative studies to evaluate the effectiveness of spatialized dropout against other regularization techniques like batch normalization, standard dropout, or L2 regularization in the context of adversarial detection.

Feature engineering and selection. API call and permission feature relevance: research could focus on identifying which specific API calls and permission features are most indicative of adversarial behavior. This involves exploring FS techniques that can enhance the model's performance. Dynamic vs. static features: the effectiveness of static features (e.g., permissions declared in the manifest) versus dynamic features (e.g., runtime API calls) in detecting adversarial attacks needs further investigation. Research could explore how the model can balance or integrate these two types of features.

Robustness to evasive adversarial techniques. Detection of sophisticated adversarial attacks: new types of adversarial attacks continue to emerge, making it essential to test the robustness of the SD-CAE model against more sophisticated evasion techniques. Research could explore how to adapt the model to detect novel attacks that exploit weaknesses in both API calls and permission-based features. Generalization to unknown attacks: the ability of the model to generalize and detect unknown or zero-day adversarial attacks in mobile apps remains an open challenge. Developing methods to enhance the model's adaptability to unforeseen adversarial behaviors is crucial.

Scalability and performance in real-world scenarios. Real-time detection capabilities: the feasibility of deploying the SD-CAE model for real-time adversarial detection in mobile apps, especially on resource-constrained devices, is an area that requires further research. Studies could focus on optimizing the model to reduce computational overhead while maintaining high detection accuracy. Scalability across diverse app ecosystems: research could explore how well the model scales across different types of mobile apps (e.g., games, social media, financial apps) and different operating systems (e.g., Android, iOS), which may have varying API call patterns and permission structures.

2.1. Problem statement

From the deep analysis of the conventional techniques, several drawbacks have been noted while detecting adversarial apps accurately. The existing models are less capable of learning the complex malware apk files due to complex application codes and black box issues. Some past studies utilized user permissions and API call features to determine the third-party apps that result in outstanding detection performance. However, the techniques reduce its generalization ability while processing with larger databases. To tackle these issues, AI-based DL techniques are introduced that automatically learn the relevant complicated malware features and recognize the unwanted malevolent apps thereby preventing dimensionality issues. Thus, this article describes a robust approach to identify third-party apps using permission and API call features.

3. DEVELOPED METHOD

Nowadays, the popularity of the android mobile phones has been growing and this popularity makes the developers for creating their solicitations (apps) on this platform. Due to the increased number of apps, made an advantage on developing various malevolent and include them on a third party arcades as protected application. Figure 1 depicts the workflow of the developed framework. Detecting these kinds of malware application is a challenging task as it is time consuming and requires high cost techniques. Hence, this study put forth a robust DL models for detecting the adversarial third party apps with adaptive feature learning. The developed strategy comprises of three stages: preprocessing, feature extraction and classification. Initially, the raw apk files consist of exiled, accessible and registered subsets of applications are preprocessed, followed by API and permission behavioral features are extracted. Finally, the excerpted features are given into the proposed SD_ConvAE approach for identifying whether the app is benign or malignant.

3.1. Feature extraction and preprocessing

When a mobile app is installed on an android device, it is granted permissions to access system resources. API calls and permissions requested by the app are then extracted to analyze its behavior. These features reveal how the app interacts with the device, identifying potential security risks or malicious activity. Analyzing API calls and permissions helps detect adversarial behaviors in mobile apps.

3.1.1. API calls-based feature extraction

The pre-defined codes present in android libraries are considered as the API calls. By statistically investigating the samples, the most relevant API call features are involved for training process. To perform this, larger benign and malignant android samples are taken that learns the identity of malevolent applications. The API call features are extracted based on the nature of the resources requested and they are of two types broadcast APIs and telephony manager APIs. The functionalities of these features are depicted as follows. Table 1 depicts the extracted telephony manager API and broadcast APIs with their functionalities.

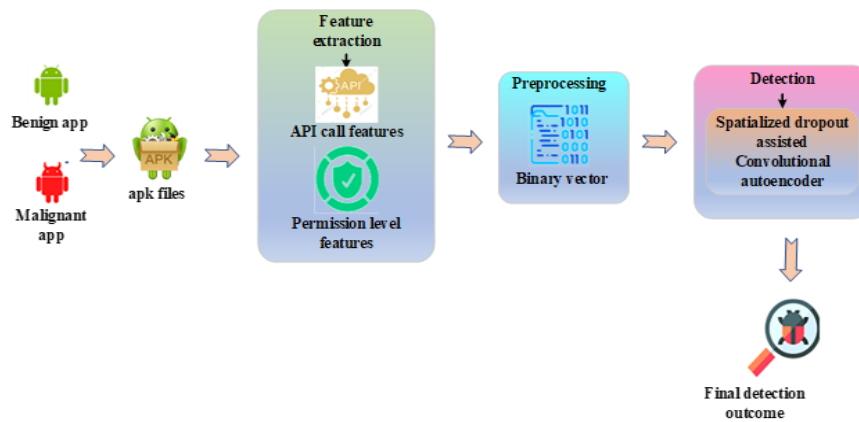


Figure 1. Workflow of the developed framework

Table 1. Telephony manager API and broadcast APIs with their functionalities

API call features	Functionalities
getDataActivity()	Yields a constant, representing the activity type
getLineNumber()	Yields the number of strings for each line
getCallState()	Yields call state
getDeviceId()	Yields IMEI number
getSIMOperator()	Yields the provider’s mobile network and country code of the SIM card
getSubscriberID()	Yields IMSI number of each device
getNetworkType()	Yields type of network for present data link
PhoneCallReceiver	Phone changes state when call back that fire offs
AutoSMSReceiver	Automatically receives incoming SMS
abortBroadcast	Automatically distracts the present broadcast

3.1.2. Permission-level feature extraction

The permissions are finalized by the android developer to gain the app access to some secured android APIs. However, some apps request access (permission) that is not required for the normal implementation process. The set of permissions that are extracted by the malignant applications to bargain user confidential information and connect to remote servers from the customers for commercial purposes. Some of the common permission requests by the malevolent apps are depicted in the table given below. Table 2 depicts the extracted Permission-level features with their functionalities. These features are then converted into binary form (0, 1) to avoid confusion during the training process.

Table 2. Permission-level features with their functionalities

Permission features	Functionalities
Access_Network_State	Asking for permission to access data about networks
Call_Phone	Asking for permission to access any number without going to the dialer
Send_SMS	Asking for permission to access and read messages
Read_Phone_State	Asking for permission to access the phone state
Internet	Asking for permission to access open network plugs
Change_Network_State	Asking for permission to access network connectivity
Get_Accounts	Asking for permission to access account details
Read_Contacts	Asking for permission to access user contacts

3.2. Adversarial app detection using SD_ConvAE technique

The extracted features are then fed into the developed SD_ConvAE approach for identifying whether the app is genuine or malignant. The proposed ConvAE is composed of two sections namely encoder and decoder. Here, a loss function is utilized to determine the losses at the time of training. The obtained channel features from the actual input signal are highly similar only when the losses are small or negligible. The encoding and decoding parameters of the AE are optimized using the loss function. For extracting the channel properties, 3D convolutional (3D-Conv) layers are used and the mathematical formulation for extracted features from encoder and decoder are depicted as (1).

$$y = (K * U + bias)\sigma \quad (1)$$

Here, K manipulates the Conv kernels, U represents the input, and σ manipulates the activation function. The encoder part of AE consists of dual Conv layers and single average pooling (AP) layers. Moreover, in the decoder part dual de-convolutional layers are present to decode the encoded features accurately. The Conv layers in the AE perform local computations and the pooling layers perform the down-sampling process. The outcome can be mathematically interpreted as (2).

$$l = \|\tilde{U} - U\|^2 \quad (2)$$

Here, l indicates the loss, \tilde{U} represents the reconstructed channel parameters and U manipulates the input signal. In addition to this, a normalization function and rectified linear unit (ReLU) activation function (AF) are included to speed up the extraction process. Hence, the AF is not deployed on the final de-conv layers. To overcome this issue, a spatialized dropout layer is introduced in the ConvAE model to learn the extracted features accurately.

The dropout function works dynamically under zero elements and computes scale transformation from the non-zero elements. The scalar conversion magnitude is highly associated with dropout layers. After performing zero operations, the zero parts are removed from the training process. The remaining parts are given into the DL the decoding part of the ConvAE model to determine the outcome. Each dropout is random in nature and hence, the model is forced to train the low-level features resulting in a changing of time for learning the features. Hence, the features considered for the training must not produce any overfitting issues due to dimensionality issues. The spatialized dropout is almost similar to conventional dropout however it detaches the complete feature sets instead of using one-dimensional (1D) features. The normalization fails when there is no strong correlation between the extracted features while processing with conventional dropouts. However, the SDs can enhance the strong independence among the features and it sets randomly to zeros for a particular dimension in Figure 2.

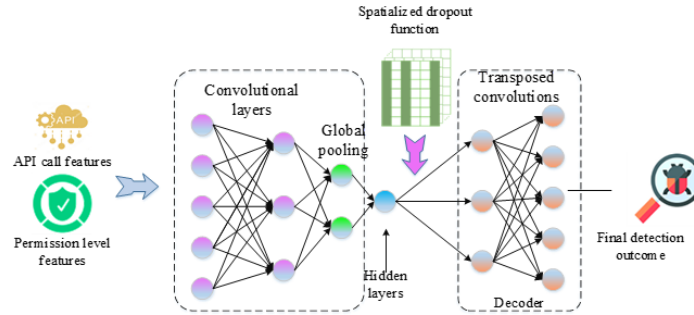


Figure 2. Architecture of SA_ConvAE model

4. RESULTS AND DISCUSSION

The proposed method is processed and experimented using the Python simulation tool. Various existing schemes like k-nearest neighbor (KNN), DL-android (DL-droid), recurrent neural network (RNN), LSTM, and Bi-directional LSTM (Bi-LSTM) are also examined and distinguished from the developed method by computing different measures accuracy (Acc), weighted F-score (W-FS), recall (Rec), FDR and kappa coefficient. For the experimentation process, a total of 1600 mobile applications are collected that are available apk files. Out of 1.6K applications, 800 are adversarial third-party apps and remaining are normal apps installed from authorized and unauthorized android platforms. These malevolent samples are collected from android app genome project [25] that consists of 49 malevolent families.

4.1. Assessment metrics

The metrics shown assess classification model performance. Accuracy measures the overall correctness of predictions, while W-FS balances precision and recall, weighted by class importance. Recall (sensitivity) emphasizes the identification of true positives, useful when minimizing missed positives. FDR helps assess the reliability of positive predictions by focusing on false positives. Kappa coefficient accounts for chance agreement, indicating model reliability beyond random predictions.

$$Acc(\%) = \frac{w+x}{w+x+y+z} \times 100\% \tag{3}$$

$$WeightedFscore(\%) = \frac{(X_m N_m)(X_b N_b)}{N_m + N_b} \times 100\% \tag{4}$$

$$rec(\%) = \frac{x}{x+y} \times 100\% \tag{5}$$

$$FDR = \frac{y}{y+w} \times 100\% \tag{6}$$

$$Kappa_{coefficient} = \frac{2*(x*w - y*z)}{(x+z)*(z+w)*(x+y)*(y+w)} \tag{7}$$

Here, w , x , y , z indicates the true negative (TN), true positive (TP), false negative (FN), and false positive (FP) respectively.

4.2. Simulation analysis of the developed framework over conventional schemes

In this section, the performance achieved by the suggested method is analyzed via tabulation under various conventional techniques. The analysis is made for detection of both normal and malware apps using different assessment measures. Various existing studies like KNN, DL-droid, RNN, LSTM, and Bi-LSTM are compared with the developed method to determine its efficiency. The detailed analysis of the obtained performance is conquered below:

Figure 3 states the accuracy-loss curves of the proposed SD_ConvAE by varying the epoch values. Tables 3 and 4 illustrates the performance analysis of various techniques for detecting benign and malignant mobile apps. From the tabulation, it is clear that the developed method achieved better performance compared to conventional techniques. The developed method accurately learns malevolent features of mobile apps and provides outstanding performance than other techniques. However, the conventional techniques showed less performance due to poor generalization capability while processing with complex features. Moreover, the error is high when malware features compared to benign apk files.

Table 3. Performance analysis of various techniques for detecting benign mobile apps

Methods	Accuracy (%)	Weighted F-measure (%)	FDR	Recall (%)	Kappa (%)
KNN	89.5	89	10.5	88	90
DL-Droid	97.7	97.2	3.3	96.5	97.3
RNN	98.2	98	2.8	97	97.9
LSTM	98.7	98.3	2.3	97.9	98.3
Bi-LSTM	99	98.9	1	98.2	98.8
Proposed SA_ConvAE	99.6	99.7	0.4	99	99.54

Table 4. Performance analysis of various techniques for detecting malignant mobile apps

Methods	Accuracy (%)	Weighted F-measure (%)	FDR	Recall (%)	Kappa (%)
KNN	87.9	87.11	12.1	87.66	87.6
DL-Droid	96.85	96.22	3.2	96.82	96.5
RNN	97.2	97	2.8	97.15	97.21
LSTM	97.8	97.34	2.2	97.79	97.80
Bi-LSTM	98.01	97.98	1.99	97.90	97.98
Proposed SA_ConvAE	99	98.97	1	98.99	98.9

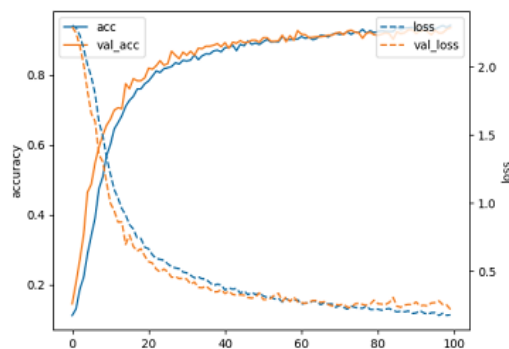


Figure 3. Accuracy-loss curves of the proposed SD_ConvAE

The proposed model have the following practical impacts:

4.2.1. Enhanced mobile security

Improved threat detection: by analyzing API calls and permission features, the SD-CAE model can detect adversarial behavior in mobile apps with high accuracy. This improves the overall security of mobile devices, protecting users from malicious activities like unauthorized data access, malware, and privacy breaches. **Real-time defense:** the model can be integrated into mobile security solutions to provide real-time detection of adversarial attacks, allowing for immediate response and mitigation, thereby reducing the risk of damage.

4.2.2. User privacy protection

Detection of privacy invasions: the model's ability to monitor and analyze permission requests ensures that mobile apps do not overreach in accessing sensitive user data. This can prevent apps from collecting more information than necessary, thereby protecting user privacy. **Compliance with privacy regulations:** the implementation of this model can help app developers and platform providers comply with privacy regulations such as general data protection regulation (GDPR) and california consumer privacy act (CCPA) by ensuring that apps adhere to proper data handling practices.

4.2.3. Reduction in false positives

Accurate threat classification: the use of spatialized dropout in the SD-CAE model helps to reduce overfitting, leading to more accurate detection and classification of adversarial threats. This reduces the number of false positives, minimizing unnecessary alerts and improving user experience. **Efficient resource utilization:** by accurately distinguishing between benign and malicious behaviors, the model ensures that security resources are only deployed when truly necessary, optimizing both computational and human resources.

4.2.4. Improved user trust and app adoption

Increased confidence in app security: as mobile apps become more secure with integrated adversarial detection mechanisms, users are more likely to trust these apps. This increased trust can lead to higher adoption rates and user engagement. Positive brand reputation: for app developers and platform providers, the implementation of such advanced security features can enhance their brand reputation as security-conscious and user-focused, attracting more users and retaining existing ones.

5. CONCLUSION

The suggested framework introduced and investigated a robust SD_ConvAE technique for recognizing harmful third-party apps using adaptive feature learning. The developed SD_ConvAE technique emphasized a novel dropout mechanism that prevents separate FS schemes for reducing dimensionality issues. Moreover, the developed framework conquered the apk files that consist of API calls and permission-level information and provides outstanding detection performance. Furthermore, binary vector conversion is performed to avoid confusion over the feature extraction. The proposed method is simulated via a Python tool and various assessment measures like accuracy, W-FS, FDR, recall, and kappa are investigated and compared with conventional schemes. The developed method achieves overall accuracies of 99.6% and 99% for detecting both malignant and benign apps respectively. However, due to the unavailability of the proper dataset, fewer apk samples are considered for the experimentation process. In the future, the developed framework will be extended by deliberating larger apk samples for detecting the malware apps, and its performance will be analyzed. Future research could explore combining the SD-CAE model with other adversarial defense mechanisms such as adversarial training, defensive distillation, or gradient masking to enhance its robustness against sophisticated attacks. Developing multi-stage frameworks where the SD-CAE model is one component of a broader detection system could be an area of interest. This could involve layering different detection models to capture a wider range of adversarial behaviors. Research could focus on developing adaptive learning techniques that allow the model to continuously learn from new adversarial attacks and evolve its detection strategies over time. This could include online learning or reinforcement learning approaches tailored to mobile security.





REFERENCES

- [1] I. Almomani, A. Alkhayer, and W. El-Shafai, "An automated vision-based deep learning model for efficient detection of android malware attacks," *IEEE Access*, vol. 10, pp. 2700–2720, 2022, doi: 10.1109/ACCESS.2022.3140341.
- [2] J. Senanayake, H. Kalutarage, and M. O. Al-Kadri, "Android mobile malware detection using machine learning: a systematic review," *Electronics (Switzerland)*, vol. 10, no. 13, p. 1606, Jul. 2021, doi: 10.3390/electronics10131606.
- [3] N. Zhang, Y. an Tan, C. Yang, and Y. Li, "Deep learning feature exploration for android malware detection," *Applied Soft Computing*, vol. 102, p. 107069, Apr. 2021, doi: 10.1016/j.asoc.2020.107069.
- [4] M. E. Z. N. Kamar, A. Esmailzadeh, Y. Kim, and K. Taghva, "A survey on mobile malware detection methods using machine learning," in *2022 IEEE 12th Annual Computing and Communication Workshop and Conference, CCWC 2022*, IEEE, Jan. 2022, pp. 215–221. doi: 10.1109/CCWC54503.2022.9720753.
- [5] G. Iadarola, F. Martinelli, F. Mercaldo, and A. Santone, "Towards an interpretable deep learning model for mobile malware detection and family identification," *Computers and Security*, vol. 105, p. 102198, Jun. 2021, doi: 10.1016/j.cose.2021.102198.
- [6] T. Sharma, H. A. Dyer, and M. Bashir, "Enabling user-centered privacy controls for mobile applications: COVID-19 perspective," *ACM Transactions on Internet Technology*, vol. 21, no. 1, pp. 1–24, Feb. 2021, doi: 10.1145/3434777.
- [7] A. Mylonas, M. Theoharidou, and D. Gritzalis, "Assessing privacy risks in android: a user-centric approach," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 8418 LNCS, 2014, pp. 21–37. doi: 10.1007/978-3-319-07076-6_2.
- [8] G. L. Scoccia, I. Malavolta, M. Autili, A. Di Salle, and P. Inverardi, "Enhancing trustability of android applications via user-centric flexible permissions," *IEEE Transactions on Software Engineering*, vol. 47, no. 10, pp. 2032–2051, Oct. 2021, doi: 10.1109/TSE.2019.2941936.
- [9] N. Vinayaga-Sureshkanth, R. Wijewickrama, A. Maiti, and M. Jadhwal, "An investigative study on the privacy implications of mobile e-scooter rental apps," in *WiSec 2022 - Proceedings of the 15th ACM Conference on Security and Privacy in Wireless and Mobile Networks*, New York, NY, USA: ACM, May 2022, pp. 125–139. doi: 10.1145/3507657.3528551.
- [10] S. Millar, N. McLaughlin, J. Martinez del Rincon, and P. Miller, "Multi-view deep learning for zero-day android malware detection," *Journal of Information Security and Applications*, vol. 58, p. 102718, May 2021, doi: 10.1016/j.jisa.2020.102718.
- [11] A. Mahindru and A. L. Sangal, "FSDroid:- a feature selection technique to detect malware from android using machine learning techniques: FSDroid," *Multimedia Tools and Applications*, vol. 80, no. 9, pp. 13271–13323, Apr. 2021, doi: 10.1007/s11042-020-10367-w.
- [12] S. I. Imtiaz, S. ur Rehman, A. R. Javed, Z. Jalil, X. Liu, and W. S. Alnumay, "DeepAMD: detection and identification of android malware using high-efficient deep artificial neural network," *Future Generation Computer Systems*, vol. 115, pp. 844–856, Feb. 2021, doi: 10.1016/j.future.2020.10.008.
- [13] P. Yadav, N. Menon, V. Ravi, S. Vishvanathan, and T. D. Pham, "A two-stage deep learning framework for image-based android malware detection and variant classification," *Computational Intelligence*, vol. 38, no. 5, pp. 1748–1771, Oct. 2022, doi: 10.1111/coin.12532.





- [14] J. Kim, Y. Ban, E. Ko, H. Cho, and J. H. Yi, "MAPAS: a practical deep learning-based android malware detection system," *International Journal of Information Security*, vol. 21, no. 4, pp. 725–738, Aug. 2022, doi: 10.1007/s10207-022-00579-6.
- [15] B. Biggio, L. Didaci, G. Fumera, and F. Roli, "Poisoning attacks to compromise face templates," in *Proceedings - 2013 International Conference on Biometrics, ICB 2013*, IEEE, Jun. 2013, pp. 1–7. doi: 10.1109/ICB.2013.6613006.
- [16] B. Biggio, G. Fumera, and F. Roli, "Pattern recognition systems under attack: design issues and research challenges," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 28, no. 7, p. 1460002, Nov. 2014, doi: 10.1142/S0218001414600027.
- [17] X. Zhu, "Super-class discriminant analysis: a novel solution for heteroscedasticity," *Pattern Recognition Letters*, vol. 34, no. 5, pp. 545–551, 2013, doi: 10.1016/j.patrec.2012.11.006.
- [18] B. Biggio, B. Nelson, and P. Laskov, "Poisoning attacks against support vector machines," *Proceedings of the 29th International Conference on Machine Learning, ICML 2012*, vol. 2, pp. 1807–1814, 2012.
- [19] C. Yang, Q. Wu, H. Li, and Y. Chen, "Generative poisoning attack method against neural networks," 2017, [Online]. Available: <http://arxiv.org/abs/1703.01340>
- [20] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," arXiv, 2013, arXiv:1312.6199.
- [21] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *2017 IEEE Symposium on Security and Privacy (SP)*, IEEE, May 2017, pp. 39–57, doi: 10.1109/SP.2017.49.
- [22] S. M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, and P. Frossard, "Universal adversarial perturbations," arXiv, 2017, arXiv:1610.08401.
- [23] D. Warde-Farley and I. Goodfellow, "Adversarial perturbations of deep neural networks," in *Perturbations, Optimization, and Statistics*, The MIT Press, 2016, pp. 311–342, doi: 10.7551/mitpress/10761.003.0012.
- [24] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami, "The limitations of deep learning in adversarial settings," in *2016 IEEE European Symposium on Security and Privacy (EuroS&P)*, IEEE, Mar. 2016, pp. 372–387, doi: 10.1109/EuroSP.2016.36.
- [25] Y. Zhou and X. Jiang, "Dissecting android malware: characterization and evolution," in *Proceedings - IEEE Symposium on Security and Privacy*, IEEE, May 2012, pp. 95–109, doi: 10.1109/SP.2012.16.

BIOGRAPHIES OF AUTHORS







Sanjaikanth E Vadakkethil Somanathan Pillai     (senior member, IEEE) holds an M.S. in Software Engineering from The University of Texas at Austin, Texas, USA, and a B.E. from the University of Calicut, Kerala, India. Currently pursuing a Ph.D. in Computer Science at the University of North Dakota, Grand Forks, North Dakota, USA, his research spans diverse areas such as mobile networks, network security, privacy, location-based services, and misinformation detection. He is a proud member of Sigma Xi, the scientific research honor society, underlining his commitment to advancing scientific knowledge and research excellence. He can be contacted at email: s.evadakkethil@und.edu.






Dr. Rohith Vallabhaneni     is a dedicated worker with a strong work ethic in leading teams to solve organizational issues. He is capable of learning all aspects of information within a company and using the technical knowledge and business background to effectively analyze security measures to determine their effectiveness in order to strengthen the overall security posture. He has great work ethic and outstanding team leadership skills and seek to accomplish organizational goals, while growing in knowledge and experience. He can be contacted at email: rohit.vallabhaneni.2222@gmail.com.






Srinivas A Vaddadi     is a dynamic and forward-thinking professional in the field of Cloud and DevSecOps. With a solid educational foundation in computer science, Srinivas embarked on a journey of continuous learning and professional growth. Their relentless pursuit of knowledge and commitment to staying at the forefront of industry advancements has earned them recognition as a thought leader in the cloud and DevSecOps space. He can be contacted at email: Vsad93@gmail.com.



Santosh Reddy Addula    a senior member of IEEE, is a research scholar at the University of the Cumberlands. His educational qualifications include a Ph.D. and a Master of Science in Information Technology. With extensive experience in the IT industry, he has demonstrated expertise across multiple domains. Santosh is an innovator who has made significant contributions to academic research through his articles as an author and co-author. Additionally, he serves as a reviewer for esteemed journals, demonstrating his commitment to advancing knowledge and upholding high standards in scholarly publications within his field. He can be contacted at email: santoshaddulait@gmail.com.



Dr. Bhuvanesh Ananthan    received the B.E. degree in Electrical and Electronics Engineering from Anna University in 2012, M.Tech. in Power System Engineering from Kalasalingam University in 2014 and Ph.D. degree from Faculty of Electrical Engineering of Anna University in 2019. He has published more than 65 papers in reputed international journals, 25 papers in international conferences and 10 books. He is a life time member of International Society for Research and Development, International Association of Engineers. He can be contacted at email: bhuvanesh.ananthan@gmail.com.