Cyber Threat Analysis on Android apps using Machine Learning Algorithms

[1]Dr. T Sunil Kumar, [2] Y. Teja, [3] P.V.V. Vamsi, [4] L. Sandeep, [5] K. Pavan Kumar

[1] Professor, [2] Student, [3] Student, [4] Student, [5] Student,
[1]Department of Computer Science and Engineering,
Sanketika Vidya Parishad Engineering College, Visakhapatnam, India.
[1]suneilkumart@gmail.Com, [2]tejamanikantayenugula@gmail.com, [3]vamsi984888@gmail.com,
[4]sandeeplolla2003@gmail.com, [5]pavankumarkodi123456@gmail.com

Abstract

Android apps have become a popular target for cyber threats due to their widespread adoption and open nature. The previous studies have used various methods like static, dynamic and hybrid analysis methods. These methods are used for detecting cyber threats that helps users to overcome data breaches. In this project, static analysis is performed on support vector machine algorithm to detect malware using malware detection schema. Genetic algorithm is the process of extracting features (only relevant data) from the data gathered from permissions, API – calls (Application Program Interface). This gathered data is then trained using Support Vector Machine (SVM) and Artificial Neural Network (ANN) algorithms. The result generated after testing is used to identify malware in the app. This result shows the malware detection in app's data. The proposed malware detection schema is able to identify malicious android applications efficiently.

Keywords: Android Malware; SVM and ANN algorithms; API-calls; Permissions.

Introduction

Mobile phones became an integral part of our lives. Android apps became the main target for cyber threats leading to the financial losses and other harm. A cyber-attack is any intentional effort to steal, expose, alter, disable, or destroy data, applications, or other assets through unauthorized access to a network, computer system or digital device. Threat actors start cyber-attacks for all sorts of reasons, from petty theft to acts of war. They use various tactics, like malware attacks, social engineering scams, and password theft, to gain unauthorized access to their target systems [1]. To install an app from the Google Play Store or a third-party source, you're essentially downloading and installing an APK file. Android then unpacks the APK, extracts the necessary files, and installs the app on your device. APK files are crucial for the Android app ecosystem, allowing users to install apps from various sources and giving developers more flexibility in distributing their applications [2]. To overcome the false positive and to generate accurate data in time is required. As new technologies are emerging in today's world along with this new kind of cyber are also going to take place. Machine learning techniques are powerful innovation used to predict the final outcome which has many algorithms [3-5].

Cyber breaches such as fraudulent emails and text are the major threat today, aiming to trick users into revealing sensitive information or downloading harmful software. Detecting breaches is crucial to eliminate cyber risks and developing security measures. The main aim of this paper is to detect the breaches, risks posed by the android apps that are being used daily. The techniques used here reduce the effects occurred due to the malware that present in applications. The ANN algorithm gives the accurate result. SVM and ANN processes the various application's data from permissions requested, API calls to get the output.

System Requirements

Hardware Requirements	Software Requirements
Processor: Intel Core i3 **Speed:** 2.2GHz **RAM:** 4GB **Hard Disk:** 500GB	**Operating System:** Windows 10 **Technology:** Python 3.9 **IDE**: Jupiter Notebook

Technologies used:

- Python flask framework.
- Genetic Algorithm for feature selection.

- Andro guard tool to extract APK data.
- Kera's library to build model.

Flow Diagram of Cyber Threat Analysis

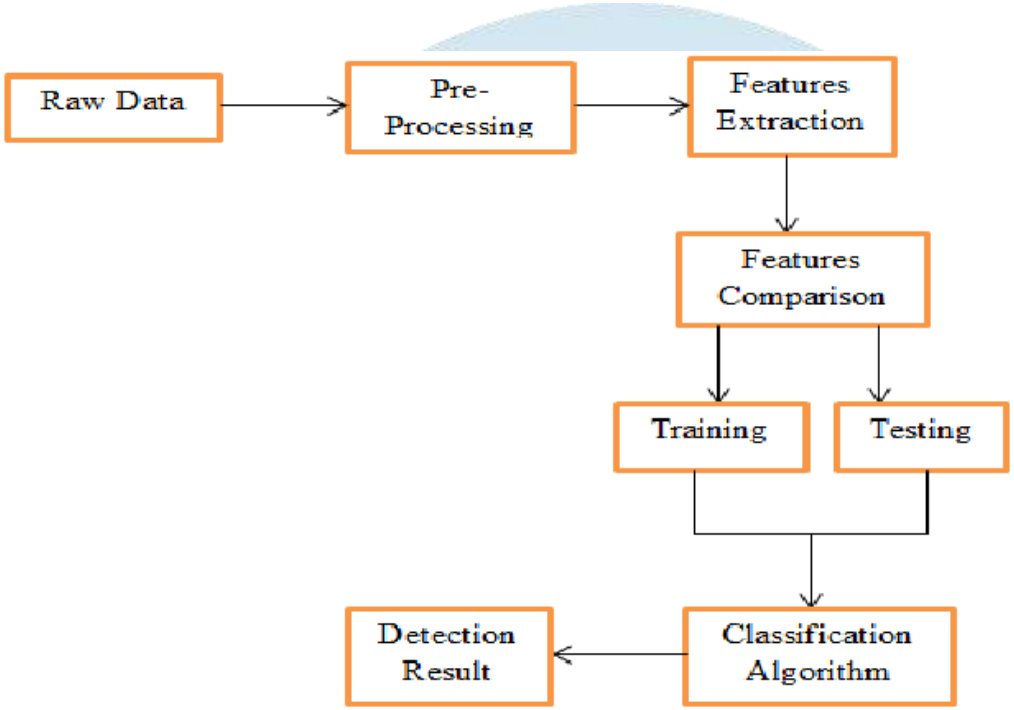

Figure 1 Flow Diagram of Cyber Threat Analysis

Here we will consider one of the datasets and then we will perform data analysis, this mainly deals with duplicate values, missing data and null values.And we deploy our model using ANN and SVM which are supervised machine learning algorithms.

Description

- Raw Data
- Data Pre-processing
- Feature Extraction
- Classification Algorithms
- Result

Raw Data:

It is the process of collecting data from sources to find the result we desire. Here the datasets required for malware is taken from cyber-crime. And the app data for checking malware is extracted from APK file. Further API calls, Permissions are extracted from the APK file. This data is further used to detect malware in the android app. These Permissions, API calls are very crucial for detecting malware.

Data Pre-Processing:

Data Pre-Processing is the process of cleaning data. It helps in increasing the quality of the data. For Pre-Processing, python pandas and NumPy are used. Pre-Processing also helps in removing the redundancy or duplicated data from the dataset. Removing the null values or finding the missing values can improve the performance.

Essential Libraries: Pandas, Matplotlib, NumPy, Seaborn and Matplotlib.

Feature Extraction:

Feature Extraction is the process of extracting only the relevant data from the whole data. In this project there are a total of 428 features. After Feature Extraction ,409 features are left. These features are further grouped forming a model which further goes through training and testing to get the result.

Performance Metrics

In a classification problem, the category or classes of data is identified based on training data. The model learns from the given dataset and then classifies the new data into classes or groups based on the training. It predicts class labels as the output, such as *Yes or No, 0 or 1, Spam or Not Spam,* etc. To evaluate the performance of a classification model, different metrics are used, and some of them are as follows: Accuracy, Confusion Matrix, Precision, Recall, F1score

I. Accuracy

The accuracy metric is one of the simplest Classification metrics to implement, and it can be determined as the number of correct predictions to the total number of predictions. It can be formulated as:

$$Accuracy = \frac{Number\ of\ correct\ Predictions}{Total\ number\ of\ predictions}$$

II. Confusion Matrix

A confusion matrix is a tabular representation of prediction outcomes of any binary classifier, which is used to describe the performance of the classification model on a set of test data when true values are known. The confusion matrix is simple to implement, but the terminologies used in this matrix might be confusing for beginners. A typical confusion matrix for a binary classifier looks like the below image (However, it can be extended to use for classifiers with more than two classes).

In general, the table is divided into four terminologies, which are as follows:

1. True Positive (TP): In this case, the prediction outcome is true, and it is true in reality, also.
2. True Negative (TN): in this case, the prediction outcome is false, and it is false in reality.
3. False Positive (FP): In this case, prediction outcomes are true, but they are false in actuality.
4. False Negative (FN): In this case, predictions are false, and they are true in actuality.

III. Precision

The precision metric is used to overcome the limitation of Accuracy. The precision determines the proportion of positive prediction that was actually correct. It can be calculated as the True Positive or predictions that are actually true to the total positive predictions (True Positive and False Positive).

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

IV. Recall or Sensitivity

It is also similar to the Precision metric; however, it aims to calculate the proportion of actual positive that was identified incorrectly. It can be calculated as True Positive or predictions that are actually true to the total number of positives, either correctly predicted as positive or incorrectly predicted as negative (true Positive and false negative). The formula for calculating Recall is given below:

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

From the above definitions of Precision and Recall, we can say that recall determines the performance of a classifier with respect to a false negative, whereas precision gives information about the performance of a classifier with respect to a false positive .So, if we want to minimize the false negative, then, Recall should be as near to 100%, and if we want to minimize the false positive, then precision should be close to 100% as possible.

Classification Algorithms

Support Vector Machine

Support Vector Machine (SVM) is a popular supervised machine learning algorithm used for both classification and regression problems. Although it is capable of handling regression tasks, SVM excels in classification scenarios. The primary goal of SVM is to identify an optimal hyperplane in an N-dimensional space that separates data points into distinct classes by maximizing the margin between the closest points from different classes. SVM is one of the classification algorithms that helps in solving complex problems.

Artificial Neural Network

Neural networks are advanced machine learning models inspired by the functioning of the human brain. These models consist of interconnected neurons or nodes that analyse data, identify patterns, and perform tasks like decision-making and pattern recognition. Neural networks are instrumental in solving complex problems, detecting intricate patterns, and adapting to ever-changing environments. By learning from large datasets, the power technologies like self-driving cars, natural language processing, and automated systems, revolutionizing multiple fields. Across industries, neural networks enhance decision-making, improve efficiency,

and streamline operations. As the cornerstone of artificial intelligence, they continue to shape technological innovation and redefine the future.

Implementation:

- Initially take any APK that has to be checked for the malware.
- Collect the data from the APK like API calls, permissions etc using Andro-guard tool.
- After the data is collected, this data has to go through preprocessing.
- After preprocessing, feature selection has to be done to create a model of datasets.
- Genetic algorithm is used for feature selection.
- Genetic algorithm helps in selecting feature selection and trains the data for the accurate result.
- In the same way it helps in increasing performance i.e. if there are any errors, it reduces it.
- After the feature selection is done, a machine learning model is created using Kera's library.
- After the model is being created, it has to be connected to the user in web. For this python flask framework is integrated to it.
- After the integration is done, the APK file is uploaded into it. It gives the result that the malware is present in the app or not.

Result

In this paper, the pure APK is taken as input to verify the malware. We developed an HTML code for Web Interface. This code is structured in a way that builds a simple web interface for Android Malware Detection using a classification model. It provides an input form for selecting the algorithm and uploading an APK file, and displays the predicted result along with some metadata about the APK. Here Flask Framework is used for integrating the web with python code. This Flask application provides a simple interface for users to upload APK files, select a classification algorithm ANN and SVM, and get predictions about whether the APK is malicious or not.

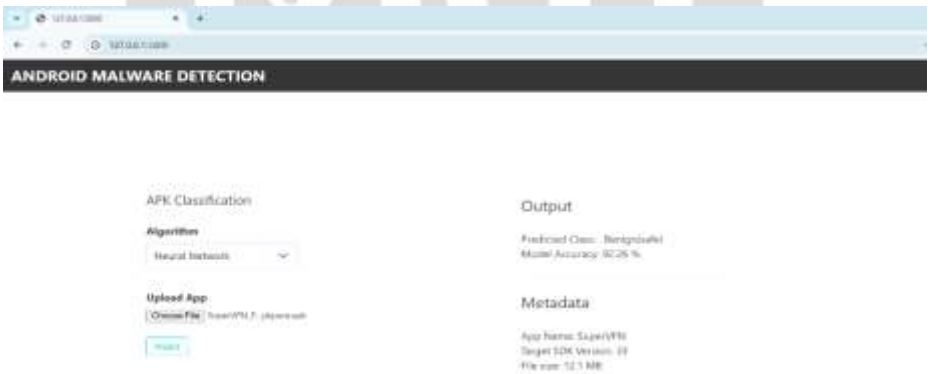

Figure 2 Web interface for ANN

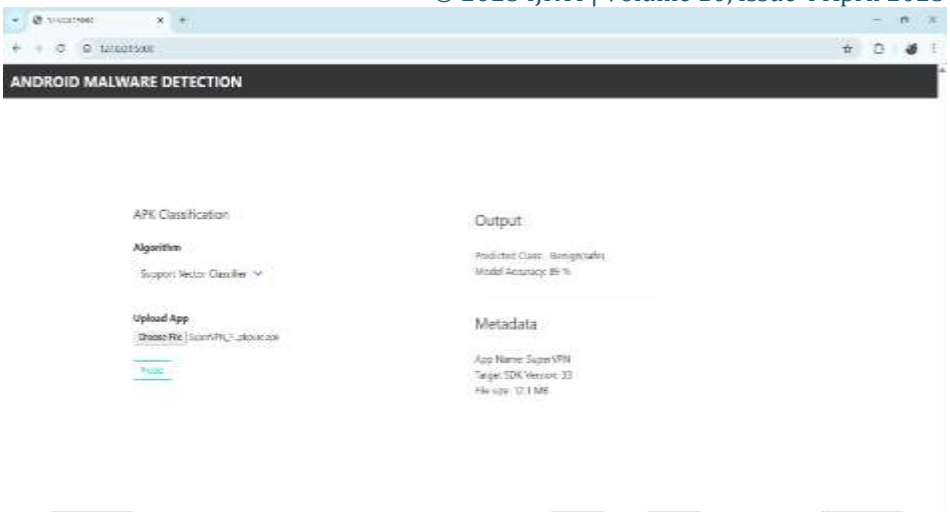

Figure 3 Web interface for SVM

Android malware detection using machine learning has shown promising results in identifying malicious apps. By analyzing various features such as API calls, permission usage, and code complexity, machine learning algorithms can effectively classify apps as malware or benign. Here ANN algorithm gives the best accuracy than the SVM algorithm. This ANN algorithm gives the promising results than the SVM algorithm.

Confusion matrix results:

Confusion matrix helps in getting the accuracy of the classifier. Here confusion matrix for two classification algorithms is generated i.e. Support vector machine and artificial neural networks. Among those ANN shows the accurate result than the SVM. And we can see in the below table that values of FN and FP of ANN confusion matrix is less than the SVM confusion matrix. So, we can say that ANN gives the accurate result than the SVM.

N=1680	**Predicted : No**	**Predicted : Yes**
Predicted : No	**623**	**54**
Predicted : Yes	**80**	**923**

Figure 4: Confusion matrix for ANN

N=1680	**Predicted : No**	**Predicted : Yes**
Predicted : No	**603**	**74**
Predicted : Yes	**110**	**893**

Figure 5: Confusion matrix for SVM

Performance Evaluation

Android malware detection was assessed with two ML models. ANN exhibited the highest accuracy of 92%, with perfect precision and recall for detecting malicious apps. SVM also performed well, achieving an accuracy of 89% with consistent precision and recall scores.

Model	**Accuracy**	**Precision**	**Recall**	**F1score**
ANN	0.92	0.92	0.98	0.94
SVM	0.89	0.89	0.84	0.84

By evaluating the values, it can be seen that ANN Machine learning based approach have achieved high accuracy rates in detecting android malware than the SVM. These models can analyze large datasets quickly, enabling efficient detection of malware.

Conclusion:

Android malware detection using machine learning is a rapidly evolving field with the potential to significantly improve mobile security. By leveraging machine learning models, researchers and developers can develop more accurate and effective detection systems that protect users from the growing threat of mobile malware.

In conclusion, cybersecurity threats are a constant and evolving danger in the digital age, necessitating a multifaceted approach to defense. This includes understanding the nature of these threats, empowering individuals with knowledge and awareness, and implementing robust strategies.

References

1. K. Liu, S. Xu, G. Xu, M. Zhang, D. Sun, H. Liu, "A review of android malware detection approaches based on machine learning", IEEE Access 8 (2020) 124579–124607
2. Z. Wang, Q. Liu, Y. Chi, Review of android malware detection based on deep learning, IEEE Access 8 (2020) 181102–181126.
3. Q. Wu, X. Zhu, B. Liu, A survey of android malware static detection technology based on machine learning, Mobile Information Systems 2021.
4. K. Bakour, H. M. Unver, Vis droid: Android malware classification based on local and global image features, bag of ¨ visual words and machine learning techniques, Neural Computing and Applications 33 (8) (2021) 3133–3153.
5. H. M. Unver, K. Bakour, Android malware detection based on image-based features and machine learning techniques, ¨ SN Applied Sciences 2 (7) (2020) 1–15.
6. H. Chen, R. Du, Z. Liu, H. Xu, Android malware classification using XG Boost based on images patterns, in: 2018 IEEE 4th Information Technology and Mechatronics Engineering Conference (ITOEC), IEEE, 2018, pp. 1358–1362.
7. F. M. Darus, N. A. Ahmad, A. F. M. Ariffin, Android malware classification using XG boost on data image pattern, in: 2019 IEEE International Conference on Internet of Things and Intelligence System (IoTaIS), IEEE, 2019, pp. 118–122.
8. Y. Dai, H. Li, Y. Qian, X. Lu, A malware classification method based on memory dump grayscale image, Digital Investigation 27 (2018) 30–37.
9. S. Gu, S. Cheng, W. Zhang, From image to code: Executable adversarial examples of android applications, in: Proceedings of the 2020 6th International Conference on Computing and Artificial Intelligence, 2020, pp. 261–268.
10. R. Damasevi ˇ cius, A. Ven ˇ ckauskas, J. Toldinas, ˇ S. Grigali ˇ unas, Ensemble-based classification using neural networks and ¯ machine learning models for windows pe malware detection, Electronics 10 (4) (2021) 485.
11. X. Xiao, An image-inspired and CNN-based android malware detection approach, in: 2019 34th IEEE/ACM International Conference on Automated Software Engineering (ASE), IEEE, 2019, pp. 1259–1261.
12. Y. Ding, X. Zhang, J. Hu, W. Xu, Android malware detection method based on bytecode image, Journal of Ambient Intelligence and Humanized Computing (2020) 1–10.
13. W. Zhang, N. Luktarhan, C. Ding, B. Lu, Android malware detection using tcn with bytecode image, Symmetry 13 (7) (2021) 1107.
14. Y.-S. Yen, H.-M. Sun Y.-S. Yen, H.-M. Sun, An android mutation malware detection based on deep learning using visualization of importance from codes, Microelectronics Reliability 93 (2019) 109–114.
15. H. Zhang, S. Luo, Y. Zhang, and L. Pan, "An Efficient Android Malware Detection System Based on Method-Level Behavioral Semantic Analysis," IEEE Access, vol. 7, pp. 69246–69256, Jan. 2019, https://doi.org/10.1109/ACCESS.2019.2919796.
16. W. Zhang, H. Wang, H. He, and P. Liu, "DAMBA: Detecting Android Malware by ORGB Analysis," IEEE Transactions on Reliability, vol. 69, no. 1, pp. 55–69, Mar. 2020
17. Lekssays, B. Falah, S. Abufardeh, A novel approach for android malware detection and classification using convolutional neural networks., in: ICSOFT, 2020, pp. 606–614.
18. A. S. Shatnawi, A. Jaradat, T. B. Yaseen, E. Taqieddin, M. Al-Ayyoub, and D. Mustafa, "An Android Malware Detection Leveraging Machine Learning," Wireless Communications and Mobile Computing, vol. 2022, May 2022, Art. no. e1830201
19. C. S. Yadav et al., "Malware Analysis in IoT & Android Systems with Defensive Mechanism," Electronics, vol. 11, no. 15, Jan. 2022, Art. no. 2354
20. K. Aldriwish, "A Deep Learning Approach for Malware and Software Piracy Threat Detection," Engineering, Technology & Applied Science Research, vol. 11, no. 6, pp. 7757–7762, Dec. 2021