IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

Android Malware Detection Using Data Mining Techniques

Mr. Konjeti Surya Hanuman[1], Mrs. M. Sree Vani[2],

[1] PG scholar, Dept. of MCA, VEMU Institute of Technology, P. Kothakota, AP, India,
[2]Assistant Professor, Dept. of CSE, VEMU Institute of Technology, P. Kothakota, AP,India

1. Abstract This project proposes a machine learning model based on the co-existence of static features for Android malware detection. The proposed model assumes that Android malware requests an abnormal set of co-existed permissions and APIs in comparing to those requested by benign applications. To prove this assumption, the paper created a new dataset of co-existed permissions and API calls at different levels of combinations, which are the second level, the third level, the fourth level and the fifth level. The extracted datasets of co-existed features at different levels were applied on permissions only, APIs only, permissions and APIs, and APIs and APIs frequencies. To extract the most relevant co-existed features, the frequent pattern growth (FP-growth) algorithm, which is an association rule mining technique, was used. The new datasets were extracted using Android APK samples from the Drebin, Malgenome and MalDroid2020 datasets. To evaluate the proposed model, several conventional machine learning algorithms were used. The results show that the model can successfully classify Android malware with a high accuracy using machine learning algorithms and the co-existence of features. Moreover, the results show that the achieved classification accuracy depends on the classifier and the type of co-existed features. The maximum accuracy, which is 98.47%, was achieved using the Support Vector Machine and 99.27% was achieved using the K Nearest Neighbor algorithms the co-existence of permissions features at the second combination level.

1.1 OBJECTIVES

Gather a diverse dataset of Android applications, comprising both benign and malicious samples, from various sources. Extract relevant features from Android application files, including permissions, API calls, intent filters, and metadata Identify the most discriminative features using techniques such as information gain, chi-square test, or feature importance scores. Evaluate different data mining techniques such as decision trees, random forests, support vector machines, and neural networks for their effectiveness in malware detection. Train the selected models using labelled data, ensuring proper splitting into training, validation, and test sets to prevent overfitting. Develop a lightweight and efficient implementation of the detection model suitable for deployment on Android devices, considering resource constraints Create an intuitive user interface for the detection system, providing users with actionable insights and alerts about detected malware Implement mechanisms for continuous monitoring of new malware threats and updating the detection system accordingly to maintain effectiveness.

1.2 Architecture

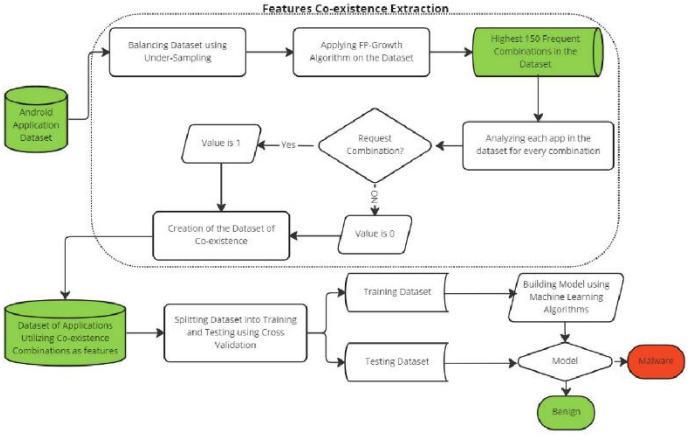


FIGURE 1. The Proposed co-existence-based detection model at every level.

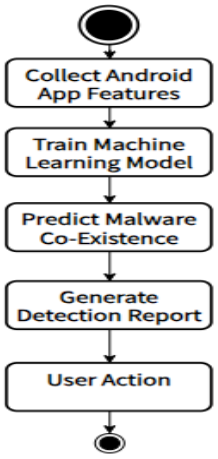
Fig -1: Architecture for Android Malware Detection Using Data Mining Techniques

1. **Android application Dataset**: Creating a dataset for Android applications, especially one suitable for malware detection research, requires careful curation

and consideration of various factors

2. **Balancing a dataset using under sampling:** effectively balance your dataset using under sampling and train models for Android malware detection with improved performance and generalization capabilities.
3. **Applying the FP-growth algorithm to dataset:** Apply the FP-growth algorithm to your dataset and extract useful patterns and associations that can contribute to Android malware detection efforts
4. **Highest 150 frequent combination in the dataset:** specific implementation of the FP-growth algorithm and the process for extracting the top frequent item sets may vary depending on the programming language or software library you are using.
5. **Analysing each app in the data set for every combination:** systematically analyze each app in the dataset for every combination of features and derive meaningful insights for Android malware detection.
6. **Request combination:** request combinations of permissions, API calls, or other characteristics extracted from Android applications.
7. **Creation of the data set of co-existence:** A dataset that captures the co-existence of features within Android applications, which can be valuable for various purposes, including malware detection, app categorization, and behavior analysis.
8. **Data set of applications utilizing co-existence combinations features:** A dataset that captures the utilization of co-existence combinations of features within Android applications. This dataset can be valuable for various purposes, including malware detection, app categorization, and behavior analysis.
9. **Splitting dataset into testing and training using cross validation:** Cross-validation is a resampling technique used for assessing how the results of a statistical analysis will generalize to an independent dataset.
10. **Training dataset:** To prepare the training dataset for cross-validation machine learning model using for classification.
11. **Testing dataset:** To prepare the testing dataset for cross-validation represents the machine learning model using for classification.
12. **Building model using machine learning algorithms:** Build a model for Android malware detection using machine learning algorithms training and testing feature matrices and label vectors, respectively. Adjust hyperparameters and other parameters of the classifier as needed.

2.WORK FLOW OF THE PROPOSED SYSTEM



Collection android features: Collecting Android features typically involves extracting relevant information from Android application files (APKs). some common types of features that are often extracted for various purposes, including Android malware detection

2. Train mechainlearning model: Training and testing feature matrices and label vectors, respectively. Adjust hyperparameters and other parameters of the classifier as needed.

3. Predict malware co- existence: Predict the co-existence of malware feature combinations in an Android application using a trained machine learning model. Specific feature combinations and model requirements.

4. Generate detection report: Generating the detection report, ensure that it is well-organized, easy to understand, and provides sufficient detail to support its conclusions. Use visualizations, tables, and descriptive text to effectively communicate the analysis results and insights

5. User action: This scenario illustrates a typical user action of installing an Android application and highlights considerations for detecting potential malware during the installation process.

CONCLUSTION

The paper has proposed a novel approach for detecting Android malware using the FP-growth algorithm, which is an association rule mining technique that is used to extract the frequent patterns of features at different feature co-existence levels. Moreover, the paper has created three datasets of co-existed features, which are co-existed permissions features only, co-existed API features only and co-existed permissions and API features. Furthermore, the paper has created the features co-existence at four levels, which are level two, level three, level four and level five. To test the proposed approach, several machine learning algorithms were used, which are SVM and KNN. The experiments have shown that KNN and SVM algorithms achieved the best accuracy of about 99.27% using KNN and 98.47% using SVM at level 2 of co-existence using permission-API co-existence in the CIC_MALDROID2020 dataset. Furthermore, the

experiments have shown that all machine learning algorithms achieved the best results at the second level of features co-existence. Moreover, the experiments have shown that using the frequent API co-existence is better than using API features in Android malware detection. That is, extracting API frequencies from each APK, converting frequencies to existence based on an average of API call frequencies as a threshold, and then applying the co-existence technique achieved better accuracy than using API co-existence without considering API calls frequencies.

Reference:

[1] h. Menear. (2021). Idc predicts used smartphone market will grow 11.2% by 2024. Accessed: oct. 30, 2022. [online]. Available: https://mobile-magazine.com/mobile-operators/idc-predicts-usedsmartphone-market-will-grow-112-2024?page=1

[2] d. Curry. (2022). Android statistics. Accessed: oct. 30, 2022. [online]. Available: https://www.businessofapps.com/data/android-statistics/

[3] o. Abendan. (2011). Fake apps affect android os users. Accessed: oct. 30, 2022. [online]. Available: https://www.trendmicro.com/ vinfo/us/threat-encyclopedia/web-attack/72/fake-apps-affect-android-osusers

[4] c. D. Vijayanand and k. S. Arunlal, ‘‘impact of malware in modern society,’’ j. Sci. Res. Develop., vol. 2, pp. 593–600, jun. 2019.

[5] m. Iqbal. (2022). App download data. Accessed: oct. 30, 2022. [online]. Available: https://www.businessofapps.com/data/app-statistics/

[6] k. Allix, t. Bissyand, q. Jarome, j. Klein, r. State, and y. L. Traon, ‘‘empirical assessment of machine learning-based malware detectors for android,’’ empirical softw. Eng., vol. 21, pp. 183–211, jun. 2016.

[7] y. Zhou and x. Jiang, ‘‘dissecting android malware: characterization and evolution,’’ in proc. Ieee symp. Secur. Privacy, may 2012, pp. 95–109.

[8] j. Scott. (2017). Signature based malware detection is dead. Accessed: oct. 30, 2022. [online]. Available: https://icitech.org/wpcontent/uploads/2017/02/icit-analysis-signature-based-malwaredetection-is-dead.pdf

[9] q. M. Y. E. Odat. Accessed: dec. 27, 2022. [online]. Available: https://github.com/esraa-cell28/a-novel-machine-learning-approach-forandroid-malware-detection-based-on-the-co-existence

[10] s. R. Tiwari and r. U. Shukla, ‘‘an android malware detection technique based on optimized permissions and api,’’ in proc. Int. Conf. Inventive res. Comput. Appl. (icirca), jul. 2018, pp. 258–263.