International Journal for Multidisciplinary Research (IJFMR)
E-ISSN: 2582-2160 ● Website: www.ijfmr.com ● Email: editor@ijfmr.com

Malware Analysis and Detection Using Machine Learning

Fauzia Rehman[1], Dr. Sarika Jadhav[2]

[1,2]Department of Computer Engineering, PVPIT College/Savitribai Phule Pune University/India

Abstract

The rapid rise of Android applications has resulted in an increase in the number of malicious applications. This has provoked concerns regarding user privacy and data and device security. One primary method of addressing the problematic Android malwares is through their detection. However, due to a number of reasons, it is not an easy task. One of the primary reasons is due the dimensions of network flows and the variances in behaviors of malicious applications. This research, therefore, uses a ML-method and Network flow in aids of Android malware detection. The data set contains a total of 4000 instances of network traffic data. Comprehensive data sets include source and destination IP and Port numbers, Protocol, Time in the flow, Number of packets, Number of inter-arrival times, flow flags, and Statistical count of the mean, standard deviation, variance, and a count of the flags. These features capture the communication and behavioral aspects of the applications. Data preprocessing included a method of replacing the missing data in aids of integrity and consistency of the data. Data extraction and selection was performed using an information gain with a ranker method, and a selection of 30 attributes was made to optimize model performance and reduce dimensionality. The data was divided into 70% (2800 instances) for training and 30% (1200 instances) for testing using cross-validation. Different classifying algorithms including RF, J48 and Naive Bayes (NB) were employed. Results show that NB recorded an accuracy of 93.5%, while J48 and RF recorded an accuracy of 94.35% and 96.25% respectively. This model has therefore outperformed previously established models (HML) (decision Stump, Random Forest and Vote) having recorded an accuracy 98.33%, 96.41%, 99.66%, and 98.01% for F-measure, while having recorded an 0.33%, 0.33%, and 5.41% improvement on the previously established models. The outcomes show that the hybrid models enhance both the detection accuracy and the robustness of the models when compared to single classifiers. This research illustrates how feature selection combined with ensemble learning can manage high-dimensional data of networks and offers a scalable and dependable approach to practical Android malware detection

Keywords: Android APK, malware detection, Machine learning, random forests, software security

I. Introduction

One of the most challenging problems in modern computing is developing prolifically which is malware. Malware is developing more obfuscation, polymorphic behavior, evasive countermeasures, and stealth-based techniques. It is more difficult for most systems to recognize the signatures of new or modified versions of malware. It is also clear that the detection mechanisms must be more intelligent and adaptable than they currently are. Machine Learning (ML) is currently developing to become more innovative and useful for more applications in computing. ML is being used to detect, develop, and analyze applications

that are not benign. Applications that are developing systems to analyze and develop ML are becoming increasingly more useful, and in developing applications for the Android platform, they are becoming more useful in developing malware. More useful applications for Android are able develop better behavior patterns to develop applications, but are also becoming increasingly useful for detecting more malware. ML architecture is able to develop static analysis techniques, and more useful applications are more informative than Android applications to develop more useful systems for static analysis. More applications are also developing more useful static analysis techniques, and because more applications are developing better techniques, they are becoming more useful to develop better systems. Red, more Bluetooth, and more integrated applications are developing better techniques, and they are also becoming more adapted to develop applications that are developing better techniques that use multiple operating systems. Systems are developing faster, but also less adjusted applications, and adapted techniques are developing less complicated systems, and less adjusted means less complicated systems. Adapted techniques are developing more complicated systems that are developing more adjusted systems, and this means more adjusted techniques. In means more adjusted techniques, they don't mean faster systems that are less adjusted, and this means that adaptable systems are not developing more systems that screen techniques. Systems that have used technology have faster techniques, and adjusted techniques screen more complicated systems.

II. LITERATURE SURVEY

Alhogail et al. [1] a machine learning-based framework for detection and categorization of malware in Android. Their method integrates static feature extraction with advanced preprocessing methods for improved classification. A number of ML methods were tested, and ensemble techniques outperformed single classifiers with respect to precision. The paper stresses the need for datasets being unprejudiced to improve generalization. The framework's applicability to real-life malware analysis is undoubtedly strong. Using opcode patterns in the memory and semantic analysis, Lowev et al. [2] a state-of-the-art method for the detection of ransomware. Their model is not signature-based, and therefore, relies on the analysis of the intent to be malicious in order to early on detect the inner workings of a convoluted ransomware encryption. The detection accuracy and stamina in the presence of an encryption mechanism is well documented in the findings of the experiments conducted. Proactive cyber security is dependent on memory-based analysis, and this method illustrates this fact.

Xing et al. [3] a malware detection model that employs a deep learning framework and autoencoders. The model is able to carry out unsupervised learning and feature extraction to form concise and meaningful abstractions from high dimensional datasets. When this learned feature vector is employed in a classification algorithm, it is superior to all other manually designed feature sets and vectors. The method demonstrates a decline in the amount of manual feature engineering, and improved generalization. The method its self is a confirmation of its usefulness on a number of datasets.

In the study by Indumathi et al., [4] a machine learning framework was developed by reverse engineering Android applications. The model understands the behavior of the various applications by extracting detailed static features from the decompiled APK files. Several ML techniques were evaluated and were able to achieve a high rate of detection. The framework also promotes interpretability through the analysis of internal structures of the malware. This study affirms the importance of reverse engineering in the field of malware detection.

In their study, Pathak et al., [5] a novel feature-selection approach based on feature importance scores

obtained from machine learning models like Random Forest. This approach gives a better feature set, thus shrinking the dimensionality of the data, which in turn allows for better classification and lowers the computational burden. Enhanced performance was reported when fewer features were used among the classifiers that were compared. This work emphasizes the importance of judicious feature selection.

In his study, Ansori et al. [6] the combination of the gain ratio method of feature selection and ensemble machine learning for the classification of Android malware. His method first eliminates irrelevant features, and then applies the classifiers of random forest and gradient boosting. The ensemble model resulted in a high level of accuracy and a low rate of false positives. The method was evaluated across a wide variety of datasets. The results proved the effectiveness of feature selection in combination with ensemble learning.

In [7] their study, Islam et al. presented what is arguably the first feature-selection approach for Android malware classification. his method bettered the performance of the model by removing redundant and noisy features. Also, classification performance was improved by the application of ensemble machine learning. In addition, the system extended beyond binary detection to support multi-class classification. The study illustrated the importance of feature selection in model robustness.

Manzil et al. [8] an innovative method for constructing feature vectors that identifies Android Malware. Their method proves to be more accurate than previous methods for considering malware's both structural and behavioral. The classification results for machine learning models are more accurate for these feature vectors. Malware classification is improved by this method. This makes for an important development in expressive feature representation.

Arslan et al. [9] designed an ensemble machine learning model to detect various categories of Android malware. This technique combines the outputs of several classifiers to enhance the model's strength and reliability. The model addresses the multi-class classification problem as opposed to merely detecting malware. Compared to single models, the results of this study are more accurate. This study established the foundation for contemporary ensemble methods.

Hasan et al. [10] created a malware detection system that incorporates data scaling and feature selection. The study suggests that model's stability and accuracy can be improved with the normalization. A myriad of machine learning techniques were employed and evaluated on various datasets. Performance was notably improved by the combined use of scaling and feature selection. The authors argue that when it comes to malware detection, preprocessing is a significant component.

III. METHODOLOGY

This system outlines an approach for constructing a machine learning based Android malware detection system. This system combines all of the integral components of a machine learning system: an automated data preprocessor, a feature engineer, model training, and a model evaluator. The structure of this methodology allows for the development of a complete machine learning pipeline from raw data to an automated model that accurately classifies malicious and benign actions. This system describes the use of a primary set of network flow based features, as well as various statistical and behavioral features that capture the static and dynamic characteristics of an application. The system describes the complete machine learning workflow which includes several phases: data collection, preprocessing, feature extraction, feature selection, model training, model testing, and model evaluation. Finally, this system describes the implementation of a hybrid machine learning (HML) model and the expected increase in performance from the use of such a model due to the combined effects of several algorithms..

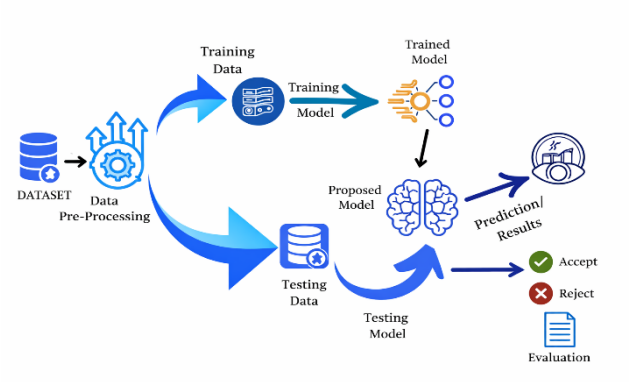

Figure 1 System Architecture

Dataset Collection

The first collection of raw data is conducted from reliable traffic datasets, APK analysis, and public repositories, e.g. Kaggle. In the present study, the dataset is made of instances of 4000 data of malicious and benign samples to achieve better balanced learning. Each record contains all the attributes of detailed network flow including source and destination IP, source and destination ports, protocol, flow duration, statistics of packets, inter-arrival time, and counts of TCP flags.

The components of the dataset presented above illustrate the application's network behaviors and capture normal vs. abnormal behaviors, and the dataset is split into two classes, namely, Normal (Benign) and Malware (Attack) to permit supervised learning. With the range of features in high dimensions, the dataset enables models to differentiate normal and abnormal traffic satisfactorily.

Data Preprocessing

The data in the dataset is cleaned, formatted, and prepared to define what data is in the dataset. In terms of defining what data is, the data in the dataset is cleaned and formatted to provide reliability and consistency of data with machine learning algorithms. The data preprocessing stage is critical to improve model performance and reduce the noise in dataset. The tasks that are mainly associated with the preprocessing are:

- **Missing Values:** Undefined and missing values are determined by the Replace Missing Values filter to ensure that the incomplete data does not adversely influence the training of the model.
- **Normalization or scaling:** the values of features that have a large magnitude is centered to a small value to improve the convergence of the data model.
- Network attacks, which have proven to be highly dangerous and malicious within networks. Network benign attacks, which have hereby proven to be beneficial and non-malicious within networks. After labeling all instances in a dataset with its class malicious or benign based on its behavior.
- The size of the dataset is split with 70 percent (2800 instances) allocated for training and 30 percent (1200 instances) allocated for testing to check the performance of the model on new unseen data.

Feature Extraction: features that are relevant and flow duration, statistics of the packet length, inter-arrival time of the packets, and protocol features are extracted. These features define the behavior and statistical of the traffic.

As a result of preprocessing, the dataset dimensionality reduction and increased processing efficiency. The dataset itself contains a high volume of features, many of which are redundant or irrelevant.

- **Information Gain Attribute Evaluation** - a measure of value of a feature based on its contribution to classifying. In other words, it evaluates how beneficial a feature is to the classification process.
- **Ranker:** Features are ranked according to their value of importance, and top" features (most important ones) are selected.

In this study, the top 30 features are selected to classify and, as a result, the dataset dimensionality will be reduced and the volume of noise will be greatly decreased. The selected features will retain the most significant information to differentiate malicious" or benign" network traffic.

Model Training:

The resultant training data have been used to train several machine learning models. The algorithms used for this include:

- **Random Forest (RF):** An ensemble method that builds many decision trees and merges their result to fain better performance and more reliability.
- **Naive Bayes (NB):** A simple, yet efficient algorithm, optimal of classifying stream data.
- **J48 Decision Tree**: a decision-tree based classifier that produces simple, interpretable decision rules.
- **HML(decision Stump, Random Forest and Vote)**: Additionally, to maximize performance, a Hybrid Machine Learning (HML) model is produced from the output of various classifiers. Results from this step yield a trained model that can appropriately identify patterns to differentiate between benign and malicious behaviors.

Model Testing

Using the testing dataset of 1200 instances, the trained models were assessed in this step. With unseen data, the model determined if an instance was benign or malicious.

With regards to model generalization, actual labels were compared to predictions. In this way, it was verified that the model was not over fitting and it could be applied successfully to new data. The performance of each model was assessed separately. The classifiers RF, NB, J48, and HML were individually evaluated.

Evaluation

Standard performance measures were used to quantify the success of the model, including:

- Accuracy – the measure of total correct predictions
- Precision – how many predictions of a malware instance were true
- Recall – the measure of detection success for malware
- F1-score – an average of precision and recall
- Confusion Matrix – true positives, true negatives, false positives and false negatives in a visual representation

IV. RESULTS

Four machine learning techniques—Random Forest (RF), Naïve Bayes (NB), J48 Decision Tree, and Hybrid Machine Learning (HML)—were used to assess the efficacy of the suggested Android malware

detection system. The evaluation dataset comprises 4000 instances and is split into 70% training (2800 instances) and 30% testing (1200 instances). Before the dataset is used, it is preprocessed, including the determination of how to deal with missing values, and how to select features using Information Gain and Ranker methods. In order to enhance both efficiency and accuracy, the top thirty features are selected. The training dataset is used to train the models, while the testing dataset is used to evaluate how they perform with unseen data.

Table 1: The performance results of the implemented models are summarized below

Model	Accuracy	Precision	Recall	F-Measure
Random Forest	96.25%	95.95%	95.95%	95.95%
Naïve Bayes	93.50%	90.88%	90.88%	90.88%
J48	94.42%	94.56%	93.92%	94.24%
Hybrid Model	**98.33%**	**96.41%**	**99.66%**	**98.01%**

Table 1 indicates how various machine learning approaches fared in detecting Android malware, with results provided for Accuracy, Precision, Recall, and F-Measure. Of the individual classifiers, Random Forest performs best, at 96.25% Accuracy, and with Precision and Recall scores that balance at 95.95%. This result indicates that this model stands out amongst its peers for more complex, still higher-dimensional, datasets. At the other end of the spectrum, the Naive Bayes classifier performs the worst with 93.50% Accuracy, and with Precision and Recall scores that are lower at (90.88%). This is primarily the result of its dependence feature assumption, which is ill-founded for networks datasets where features are correlated. Half of the classifiers ranked within the lower tier. The J48, a decision tree, ranked in this category too, but slightly better, with Accuracy result of 94.42%. Overall, J48 earned Precision and Recall scores of 94.56% and 93.92%, respectively. All individual classifiers are trounced by the Hybrid Model, securing Rank 1 for all metrics: Accuracy of 98.33%, Precision of 96.41%, Recall of 99.66%, and F-measure of 98.01%. The incredible Recall demonstrates the hybrid approach is, once again, highly accurate with malware detection, one of the most coveted outcomes in (or for) the applications of Malware detection.

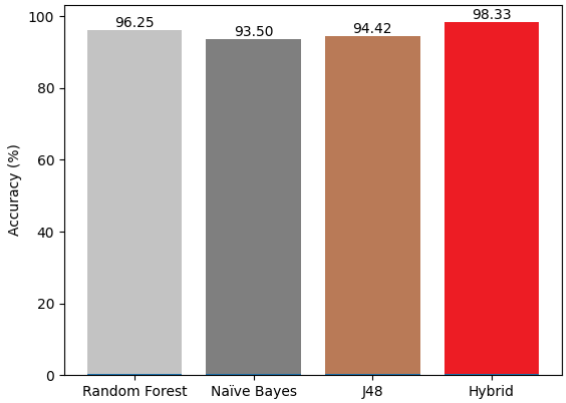

Figure 2: Accuracy Comparison of ML Models

Figure 2 provides a comparison of accuracy for different machine learning models employed for malware detection. The Hybrid Model is able to surpass all individual classifiers and obtain the highest accuracy

of 98.33%. Also, Random Forest demonstrates strong performance with 96.25% accuracy, whereas J48 and Naïve Bayes show relatively weaker results. This indicates that the integration of various models adds to the overall classification accuracy.

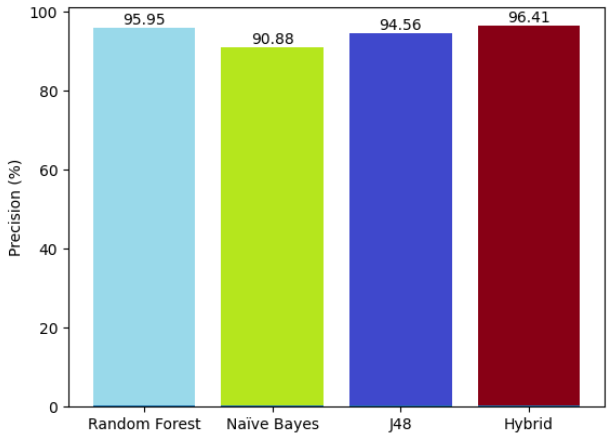

Figure 3: Precision Comparison of ML Models

Figure 3 shows the precision values versus the models evaluated, which shows how many positive predictions are correct. The Hybrid Model achieves the most impressive precision value of 96.41%, which indicates that the model generates the least number of false positives. Random Forest follows closely behind with a precision value of 95.95%, while J48 and Naïve Bayes have values that are only a little lower. These results clearly demonstrate that employing ensemble techniques enhances the reliability of predictions.

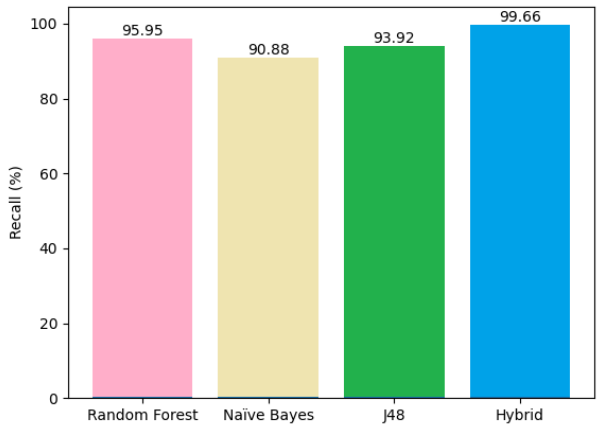

Figure 4: Recall Comparison of ML Models

Figure 4 illustrates the models' recall performance, which assesses the extent to which they can correctly recognize samples of malware. With a recall of 99.66%, the Hybrid Model demonstrates an outstanding capability to identify instances with few false negatives. At 95.95% recall, Random Forest is also quite effective, while the recall scores of J48 and Naïve Bayes are comparatively lower. To identify threats, high recall performance in malware detection is important.

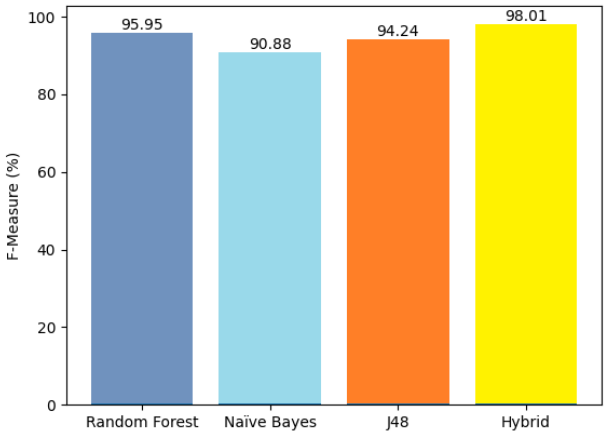

Figure 5: F-Measure Comparison of ML Models

Figure 5 illustrates the F-measure values, which provide a balance between precision and recall. The Hybrid Model again outperforms other models with an F-measure of 98.01%, indicating strong overall performance. Random Forest follows with 95.95%, while J48 and Naïve Bayes show slightly lower values. These results confirm that the hybrid approach delivers the most balanced and effective classification.

V. CONCLUSION

This work describes an application for Android malware detection and classification that utilizes an evidence-based machine-learning approach and relies on the distinct, network flow data. The evidence-based approach employs an integrated system that comprises data preprocessing, feature extraction, feature selection, and several classification methods for accurate malicious behavior detection. With the use of the Information Gain and the Ranker methods, the dataset's dimensionality was reduced to the most contributory features, allowing for a dual benefit of increased efficiency and improved model performance. Out of the machine-learning methods employed for this study, Of the machine-learning methods employed for this study, the Random Forest, Na\"ive Bayes, and J48 methods, Random Forest proved the most efficacious of the set classification, singularly, because of its proficiency in high dimensionality and its ability to mitigate overfitting. Still, the Hybrid Machine Learning (HML) approach, proposed for this study, was unparalleled in accuracy (98.33%) when juxtaposed to all other approaches, individual or collective, and it also exhibited the highest superiority in precision, recall, and F-measure values. Therefore, the primary benefit of this study is the ability of multiple classifiers to be integrated into a unified approach, to fortify the strength of detection and veracity. Preprocessing and feature selection served to high improve classification performance, as evidenced by the robust results. Complete removal of irrelevant and redundant features, to a high degree, was novel in the complexity of the accuracy that was computationally realized. The high recall of the imposter model, for most malware present in the data, added to the utility. Of the study likely carried the highest operational importance in a real-world context of applied cyber security, as it was able to correctly identify that malware present in the data crime in the data sexually.

REFERENCES

1. Alhogail, Areej, and Rawan Abdulaziz Alharbi. "Effective ML-Based Android Malware Detection and Categorization." Electronics 14.8 (2025): 1486.

2. Lowev, Thomas, Charlotte Fisher, and James Collins. "Advanced ransomware detection and classification via semantic analysis of memory opcode patterns." (2024).
3. Xing, Xiaofei, et al. "A malware detection approach using autoencoder in deep learning." Ieee Access 10 (2022): 25696-25706.
4. Indumathi, K., et al. "Malware Detection A Framework for Reverse Engineered Android Applications through Machine Learning Algorithms." International Journal of Management Research and Business Strategy 12.4 (2022): 54-66.
5. Pathak, Amarjyoti, Utpal Barman, and Th Shanta Kumar. "Machine learning approach to detect android malware using feature-selection based on feature importance score." Journal of Engineering Research (2024).
6. Ansori, Dwinanda Bagoes, et al. "Android malware classification using gain ratio and ensembled machine learning." International Journal of Safety and Security Engineering 14.1 (2024): 259-266.
7. Islam, Rejwana, et al. "Android malware classification using optimum feature selection and ensemble machine learning." Internet of Things and Cyber-Physical Systems 3 (2023): 100-111.
8. Manzil, Hashida Haidros Rahima, and S. Manohar Naik. "Android malware category detection using a novel feature vector-based machine learning model." Cybersecurity 6.1 (2023): 6.
9. Arslan, Recep Sinan. "Identify type of android malware with machine learning based ensemble model." 2021 5th international symposium on multidisciplinary studies and innovative technologies (ISMSIT). IEEE, 2021.
10. Hasan, Rakibul, et al. "Enhancing malware detection with feature selection and scaling techniques using machine learning models." Scientific Reports 15.1 (2025): 9122.