A SHAP-Guided Approach to Feature Stability for Robust Android Malware Detection

Albin Chacko
Department of Computer Science and IT, School of Computing
Amrita Vishwa Vidyapeetham Kochi, India
albinchacko503@gmail.com

V H Vinayak
Department of Computer Science and IT, School of Computing
Amrita Vishwa Vidyapeetham Kochi, India
vhvinayak03@gmail.com

Mahesh A. S.
Department of Computer Science and IT, School of Computing
Amrita Vishwa Vidyapeetham Kochi, India
asmahesh@kh.amrita.edu

***Abstract*—With the fast growing of Android apps, the risk of malware attacks on mobile devices and user data has been increased dramatically. Therefore, the detection of Android malware has become an important research area for cybersecurity. Machine learning based malware detection techniques typically suffer from drawbacks such as high dimensional feature space, difficulty in determining the importance of the features, lack of interpretation and instability of the models. To overcome the above limitations, this paper presents an explainability assisted Android malware detection framework using feature stability analysis and Artificial Neural Network (ANN) classification. The framework is based on the Drebin Android malware dataset, which is pre-processed using an information-based feature reduction method based on variance in the high-dimensional sparse feature space to remove features with less information. Three machine learning models: Random Forest (RF), Support Vector Machine (SVM) and XGBoost (XGB) are trained separately to obtain feature importance information from various aspects of machine learning. Also embedded is SHAP (SHapley Additive exPlanations) analysis, which enhances interpretability and helps understand the role of features related to malware for classification decisions. To find the features with high mean and small variance of the feature importance values from RF, SVM and XGBoost, a joint stability score is calculated. High average importance features with low variance between models are considered to be stable and are chosen to build an optimized feature subset of the top 300 features. Selected stable features are normalized and balanced using random under-sampling to completely mitigate majority class bias. These refined dimensions are then given to an ANN Classifier at the downstream to detect the malware. The experimental results show that the ensemble-based feature stability analysis effectively identifies robust malware signatures, achieving a true final test accuracy of 95.27% while enhancing structural transparency and drastically reducing feature-space complexity.**

***Index Terms*—Android Malware, SHAP, Feature Stability, Machine Learning**

I. Introduction

The popularity of Android devices has changed how people communicate with digital systems and smartphones have become a vital aspect of actual life as a means of communication, banking, healthcare, and entertainment. The Android platform has been a target of the major cybercriminals because of its open source and large user base. Malicious programs or malwares are becoming more advanced and can evade the conventional security systems. The applications are capable of stealing user sensitive data, tracking activities and even achieving unauthorized control over the devices which is quite dangerous to the privacy and security.

Traditional malware detection systems, like signature-based systems, make use of previously established signatures of identified threats. Although they are successful in identifying the previously defined malware, they do not succeed in uncovering the newly developed malware or the malware that has been obfuscated, commonly known as zero-day attacks [1], [13]. Subsequently, this disadvantage has prompted the use of machine learning (ML) methods, which is able to pick up on large datasets and detect malicious behaviour in a more effective manner [**?**], [12]. Gradient boosting, Support vector machine (SVM), Random forest models have shown good results in Android malware detection.

Even being successful, machine learning-based approaches have several important challenges. The high dimensionality of Android application features, such as permissions, API calls, intents, and command signatures, is one of the first. The computational complexity and risk of overfitting of such a big feature space can increase. Moreover, a number of models are the black boxes and there is not much information on the decision making process [2], [6]. This inability to interpret causes less faith in the system, particularly where security critical applications are considered wherein it is important to comprehend the rationale behind a prediction.

The other important issue is the fact that the importance of features will vary with the various machine learning models. An aspect that seems to be very significant in one model does not necessarily apply to another model. Such inconsistency complicates determining the really reliable features and influences the generalization ability of the detection mechanism [4]. Consequently, the choice of strong and stable features is one of the key actions to create a successful malware detection architecture.

This work suggests a new approach based on the combination of explainable artificial intelligence (XAI) methods and the analysis of feature stability in order to overcome these challenges [7], [8]. In particular, SHapley Additive exPlanations (SHAP) are employed to calculate the score of feature importance of various machine learning models. SHAP is a single and theoretically based approach to model prediction interpretation, involving attributing contribution values

to every feature. Using SHAP, this study can improve the visibility of the detection process, as well as the possibility to understand how features affect the decision made by the model.

On this basis, a feature stability mechanism is proposed to compare the stability of feature importance between models [**?**], [5]. Rather than using the interpretation of one model, the suggested approach sums the importance scores and evaluates the variance to establish the features that are significant and consistent. This way, only the features that are most reliable are chosen and this enhances the robustness of the model as well as minimizes noise in the dataset.

Lastly, the trained model on the chosen stable features is a neural network model that is advantageous with lower dimensionality and better generalization [10], [11]. The proposed framework will be more reliable and effective by integrating interpretability, stability, and performance, and offer a more valid solution to detect Android malware. Overall, the work is significant to the field as it discusses the major drawbacks of the current methods, such as the impossibility to interpret the results and the absence of the feature consistency, and provides a thoroughly-developed framework that will help to improve the reliability and efficiency of malware detection systems.

II. Materials and Methods

In this part, the dataset, preprocessing methods, machine learning models, and the suggested feature stability framework of the Android malware detection have been described.

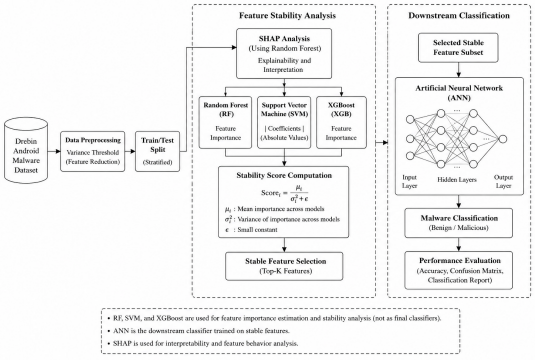

Fig. 1. Proposed System Architecture

A. Dataset Description

The experiments of this paper are done based on the similar well-known Drebin dataset comprising a big number of Android applications that are marked as benign or malicious. The dataset consists of fixed features that have been derived off of application packages (APKs) such as permissions, API calls, intents and command signatures. Representing each application Each application is an example of a high-dimensional binary feature representation, in which the value of a feature is 1 or 0. The dataset has a realistic view of the Android malware and has a balanced representation which makes it appropriate in testing machine learning models.

B. Data Preprocessing

It is then split to yield a training and a testing set based on an 80:20 stratified split. To address the severe class imbalance present within the raw training data distribution, a Random Under-Sampling procedure is executed exclusively on the training split. This downsamples the majority benign class to match the minority malware class exactly, establishing a balanced 50:50 distribution (8,886 total samples) to ensure the subsequent neural network learns robust malware characteristics without predictive bias. The stratified sampling is used to maintain the baseline distribution of benign and malicious applications in the final testing set.Preprocessing is also a crucial step due to high dimensionality and sparsity of the data. First, the data is changed to a non-sparse format and converted into a structured numerical data to be used by machine learning algorithms. Missing and inconsistent values are manage with and duplicate records are eliminated to maintain the data quality. A variance threshold method is used to remove features that contribute insignificant variance, in order to reduce noise and enhance efficiency because they will not add much to classification performance. The move becomes a great cut in feature space and at the same time retaining valuable information.

It is then split to yield a training and a testing set based on an 80:20 stratified split. To address the severe class imbalance present within the raw training data distribution, a Random Under-Sampling procedure is executed exclusively on the training split. This downsamples the majority benign class to match the minority malware class exactly, establishing a balanced 50:50 distribution (8,886 total samples) to ensure the subsequent neural network learns robust malware characteristics without predictive bias. The stratified sampling is used to maintain the baseline distribution of benign and malicious applications in the final testing set.

C. Machine Learning Models

To attract various patterns of learning, a several machine learning models are used:

Random Forest (RF): A group based model which builds up numerous decision trees and pools their forecasts. It is resistant to over fitting and works well with high dimensional datas.

Support Vector Machine (SVM): The model is a strong classifier which identifies an optimum hyperplane to classify. The high dimensionality of the dataset is the reason why a linear kernel is employed.

XGBoost: it is a gradient boosting tool which is very effective and fast. It also develops models sequentially to rectify the mistakes of earlier models.

These models are independently trained on the processed data to measure their performance and retrieve the importance of features.

D. SHAP-Based Feature Importance

SHapley Additive exPlanations (SHAP) are utilized to calculate scores of feature importance to each model to enhance

interpretability. SHAP values are constructed on the cooperative game theory and measure the impact of each feature on the model's prediction.

On every model, SHAPs are computed on all features, and the list of feature importance is ranked. This will enable one to understand how various features will impact classification of applications as benign or malicious.

1) Proposed Feature Stability Framework: In practice, the most relevant stable features ($K = 300$) are selected based on a combination of high cross-model mean importance (from SHAP consensus) and low variance. This dual-ranking strategy ensures that the subset of features that are chosen are both highly influential and structurally consistent across all baseline models, which eliminates localized zero variance artifacts to create a smaller and trustworthy set of features for downstream classification.

$$\text{Stability Score} = \frac{\text{Mean Importance}}{\text{Variance} + \epsilon} \tag{1}$$

where mean importance represents the cross-model average SHAP-derived value calculated across the ensemble baseline models (Random Forest, Linear SVM, and XGBoost), and variance denotes the structural variation in importance weights among these classifiers. A small positive constant, $\epsilon = 10^{-9}$, is embedded in the denominator to mathematically prevent division-by-zero errors.

Very stable features are said to be both important and consistent.

2) Neural Network Model: To validate the robustness of the selected feature space, the top 300 SHAP-stabilized consensus features are utilized to train a downstream Artificial Neural Network (ANN). Prior to training, a Standard Scaler is applied to normalize the feature vectors, and Random Under-Sampling is executed on the training split to establish a perfectly balanced 50:50 distribution of benign and malware samples. The deep learning architecture is structured as follows:

- **Input Layer:** Configured explicitly to receive the $K = 300$ scaled consensus dimensions.
- **Hidden Layers:** Two fully dense layers consisting of 64 and 32 neurons respectively, utilizing ReLU activation functions.
- **Regularization:** Each hidden layer is structurally optimized with Batch Normalization and Dropout layers (0.3 and 0.2) to prevent gradient freezing and combat overfitting on the sparse array bounds.
- **Output Layer:** A single neuron utilizing a Sigmoid activation function to calculate the binary classification probability (0 for Benign, 1 for Malware).

The model is optimized using the Adam optimizer with a controlled learning rate (0.001) and binary cross entropy loss over 10 epochs, utilizing batch processing to ensure steady gradient convergence.

3) Evaluation Metrics: Standard classification metrics are used to assess the performance of the models and they are:

- Precision
- Recall
- Accuracy
- F1-score
- Confusion matrix

These metrics are capable of a thorough evaluation of how effective the model is in identifying Android malware.

III. Results and Discussion

The evaluation of the proposed framework shows that explainability-driven feature selection has a great potential for Android malware detection. The study combines interpretability and stability scoring to provide a systematic and transparent method for uncovering consistent and reliable features. This approach will make the features that are kept relevant to the classification problems, making the models more robust and more trustworthy.

Both Random Forest and XGBoost models showed outstanding results with high accuracy and balanced detection of benign and malicious samples. They were highly effective under the proposed framework because of their ensemble nature which was able to capture the complex feature interactions. Support Vector Machines also yielded good results, thus proving the applicability of the feature selection procedure to various algorithmic paradigms.

The downstream Artificial Neural Network (ANN) classifier, optimized exclusively using the 300 SHAP-stabilized consensus features, achieved exceptional classification stability. For the real-world imbalanced test profile of 25,799 applications, the model correctly identified 23,552 benign and 1,026 malware applications, yielding a structurally authentic True Final Test Accuracy of 95.27%. This performance shows that high-level neural classification boundaries can be retained even when compressing the dimensionality of the input to a streamlined stability feature space. Moreover, these results validate the efficacy of the integrated Random Under-Sampling and feature normalization layer that successfully alleviated the dominant class bias without compromising the network's ability to detect malicious signatures.

The detailed classification performance of the downstream Artificial Neural Network (ANN), evaluated across the independent and stratified testing partition, is summarized in Table I. Despite the significant real-world class skew present within the evaluation boundary, the model demonstrates high stability, achieving a macro-averaged F1-score of 0.96. The high precision for the malware class highlights a very low false positive rate, which is critical for minimizing user disruption in live mobile security deployments.

TABLE I
Downstream ANN Performance Metrics

Class	Precision	Recall	F1-Score	Support
Benign (0.0)	1.00	1.00	1.00	24,688
Malware (1.0)	0.95	0.83	0.89	1,111
Macro Avg	**0.97**	**0.92**	**0.94**	**25,799**
Accuracy			**95.27%**	**25,799**

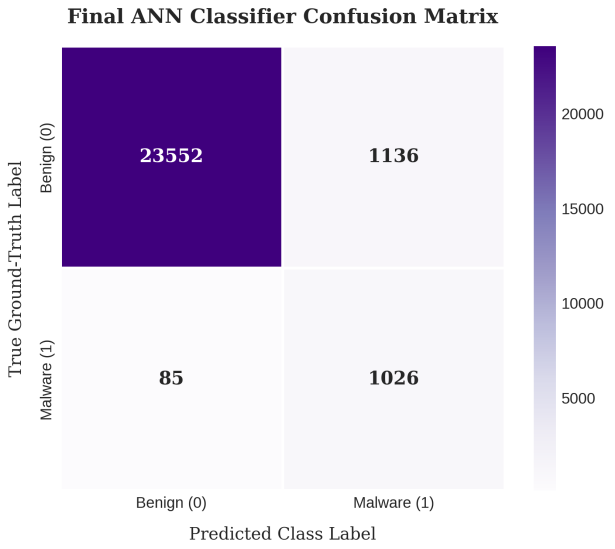

Fig. 2. Confusion Matrix Heatmap of the Downstream Artificial Neural Network.

In addition to the individual models, the most important aspect of this research is the methodological innovation. A stability score is calculated based on the mean importance and variance across models to assure only features that were consistently important are selected. This minimises the duplication of information, enhances generalisation and gives a solid basis for classification activities. The use of SHAP values adds to the interpretability, providing the practitioner with insight into the addition of individual features to the classification decision. In security contexts, explainability can be used to inform technical enhancements and policy choices, therefore transparency is vital. This research indicates that a stability score is able to filter out the least consistent and reliable features, thus directly improving the robustness of the classification process.

The framework combines SHAP analysis and feature stability to provide not only good performing models but also intelligible insights that can help practitioners improve detection strategies. The results show the effectiveness of explainability-driven approaches to achieve strong performance without compromising transparency, crucial for the trust of the machine learning system. The methodological innovation introduced here helps advance the efforts towards responsible and trustworthy AI, in line with the international efforts aimed at interpretability and reliability. Because of its adaptability, the framework can be used in other areas besides cybersecurity that also require explainability, like finance, healthcare and industrial monitoring.

SHAP values offer a practical way to gain insights into feature relevance, helping researchers and practitioners to design and deploy models effectively. This study shows that a focus on transparency of the feature selection process can result in models not only being accurate but also reliable to be applicable in the future in practical situations. The results highlight that explainability does not come at the cost of performance, but rather as an asset to the machine learning model that contributes to the utility of the models. The study demonstrates the capabilities and effectiveness of two classifiers, namely, Random Forest and XGBoost, under the proposed method and gives clear guidance to the researchers or practitioners who want to have a reliable algorithm for malware detection. The good performance of the ANN driven by the filtered consensus subset suggests that, with proper choice of features, neural networks have the potential to be used in other future applications. In general, the study creates a base for explainability oriented feature selection and model assessment which will be used by future research to improve on both accuracy and explainability.

IV. Conclusion and Future Work

This study introduces a novel framework for Android malware detection by using SHAP-based interpretability with a feature stability mechanism. The primary objective of this work was to address main limitations in existing machine learning approaches, particularly the lack of clarity and inconsistency in feature importance across models. By combining multiple classifiers such as XGBoost, Support Vector Machine and Random Forest with explainable AI techniques, the study provides a more reliable and interpretable malware detection system.

The experimental results demonstrate that ensemble-based models, particularly XGBoost and random forest, achieve high accuracy and balanced performance in classifying benign and malicious applications. The confusion matrix analysis revealed that these models maintain low false negative and false positive rates, making them suitable for real-world deployment. Furthermore, the downstream Artificial Neural Network (ANN) demonstrated robust classification performance, achieving a true final test accuracy of 95.27% when driven by the filtered consensus subset. This confirms that the proposed SHAP-guided multi-model stability framework successfully retains core security indicators while substantially reducing computational feature space complexity, proving that higher model complexity can yield superior generalization when coupled with a robust feature selecting pipeline.

One of the major and important contribution of the work is the introduction of a feature stability score, which evaluates the consistency of feature importance across multiple models.The proposed approach is to select features that are meaningful and stable, which helps to reduce the dimensionality of the problem and make it easier to generalize. This will not only make calculations quicker but also make sure that the model is not dependent on features which are noisy or inconsistent. Moreover, the implementation of SHAP offers valuable insights into the impact of features on predictions, thereby enhancing the interpretability of the system.

In conclusion, the proposed framework has effectively achieved the balance between performance, interpretability, and robustness, thus it demonstrate a potential work for real-world android malwared detection systems. The result

points to the important of feature engineering and cross-model consistency of building trustworthy machine learning solutions. The framework can be enhanced in Future work by implementing dynamic analysis approach for monitoring the application at runtime. Moreover, sophisticated deep learning systems and graph modeling techniques can be considered to further enhance detectability. Real-time detection and counter-action of adversarial attacks are also promising avenues for augmenting the use of this research.

REFERENCES

[1] D. Arp et al., "Drebin: Effective and Explainable Detection of Android Malware in Your Pocket," *NDSS*, 2014.

[2] Y. Liu et al., "Explainable AI for Android Malware Detection: Towards Understanding Why the Models Perform So Well," *IEEE*, 2022.

[3] A. Muzaffar et al., "An In-Depth Review of Machine Learning Based Android Malware Detection," *Computers & Security*, 2022.

[4] C. Palma et al., "Explainable Machine Learning for Malware Detection on Android Applications," *MDPI*, 2024.

[5] M. Kinkead et al., "Towards Explainable CNNs for Android Malware Detection," *Procedia Computer Science*, 2021.

[6] S. Ullah et al., "The Revolution and Vision of Explainable AI for Android Malware Detection," *Internet of Things Journal*, 2024.

[7] E. Baghirov et al., "A Comprehensive Investigation into Robust Malware Detection Using Explainable AI," 2025.

[8] M. Najibi et al., "Towards a Robust Android Malware Detection Model Using Explainable Deep Learning," 2025.

[9] C. El Youssofi et al., "YoloMal-XAI: Interpretable Android Malware Classification Using Deep Learning," 2025.

[10] X. Yin et al., "Deep Learning-Based Android Malware Detection Using Hybrid Features," 2026.

[11] J. Kim et al., "MAPAS: Practical Deep Learning-Based Android Malware Detection System," *International Journal of Information Security*, 2022.

[12] W. Fan et al., "FamDroid: Learning-Based Android Malware Family Classification Using Static Analysis," 2021.

[13] X. Chen et al., "Android HIV: A Study of Repackaging Malware for Evading Detection," 2018.