IJITCE

ISSN 2347- 3657

# International Journal of Information Technology & Computer Engineering


Email : ijitce.editor@gmail.com or editor@ijitce.com

International Journal of Information Technology & Computer Engineering

ISSN 2347–3657

Volume 13, Issue 2, 2025

https://doi.org/10.62647/ijitce.2025.v13.i2.pp370-381

# AN ENHANCED ENSEMBLE LEARNING FRAMEWORK FOR AUTOMATED ANDROID MALWARE DETECTION IN CYBERSECURITY

[1] *V . Riyaz Ahammmed, MCA Student, Department of MCA*

[2] *Shaik Haseena, M.Tech,(Ph.D), Assistant Professor,Department of MCA*

[12] *Dr KV Subba Reddy Institute of Technology,Kurnool*

**ABSTRACT**

Human life has changed from real-world to virtual worlds due to recent advancements in computer technology. Malware is superfluous software that is often used to initiate cyberattacks. Advanced packaging and obfuscation techniques are still being used by malware strains to evolve. These methods complicate the categorisation and detection of malware. To successfully battle emerging malware strains, new methods that vary from traditional systems should be used. All complicated and novel malware strains cannot be detected by machine learning (ML) techniques. The deep learning (DL) approach may be a viable way to identify every kind of malware. In this research, the Optimal Ensemble Learning Approach for Cybersecurity (AAMD-OELAC) approach for Automated Android Malware Detection is presented. The automatic categorisation and detection of Android malware is the main goal of the AAMD-OELAC approach. The AAMD-OELAC approach preprocesses data at the preliminary stage in order to do this. Three machine learning models—the Regularised Random Vector Functional Link Neural Network (RRVFLN), the Kernel Extreme Learning Machine (KELM), and the Least Square Support Vector Machine (LS-SVM)—are used in the AAMD-OELAC technique's ensemble learning process for Android malware detection. Lastly, the three DL models' optimum parameter tuning is achieved by using the hunter-prey optimisation (HPO) technique, which also contributes to better malware detection outcomes. A thorough experimental investigation is carried out to demonstrate the superiority of the AAMD-OELAC approach. The simulation results demonstrated the AAMD-OELAC technique's superiority over other methods already in use.

## 1. INTRODUCTION

Network engineers and computer scientists are increasingly concerned about cyber security, thus finding satisfactory answers to a number of issues is necessary [1]. As a result, different malware programs and targets are well-identified and researched, as are the rapid advancements in technology and their intrinsic integration into all facets of lives [2]. The malware kind that attracted the most attention in the online community is Android malware. Android is a popular operating system that leads the market for operating systems [3].

Since few malware programs contain more than 50 characteristics, making

International Journal of Information Technology & Computer Engineering
ISSN 2347–3657
Volume 13, Issue 2, 2025
https://doi.org/10.62647/ijitce.2025.v13.i2.pp370-381

detection challenging, intrusive malware techniques are developed to evade detection [4]. Therefore, it is crucial to develop methods to address the ongoing proliferation of Android malware in order to effectively detect, deactivate, or eliminate it. All of these challenges motivate researchers in the field to carry out further studies in order to identify malware and effectively handle it [5]. Thus, three mechanisms—dynamic, static, and hybrid analytic methods—have been established by researchers to detect Android malware. Without requiring a real application deployment, static analysis retrieves the elements that help discover detrimental performance for applications [6]. However, this kind of analysis was hampered by code obfuscation techniques, which aid virus authors in avoiding static approaches. App malware may be identified during runtime via dynamic analysis [7]. The capacity to locate the malware element using source code is often provided by the static analysis function, while the ability to locate malware in a runtime context is provided by the dynamic analysis feature. Malware may expose Android users and developers to needless risks and hazards [8]. Malware detection techniques are covered in this paper. Android Application Packages (APKs) are used to derive a suitable collection of characteristics for malware detection using the machine learning model. Malicious APKs may be identified using machine learning (ML) and deep learning (DL) techniques [9]. Similar to malware detection, software code vulnerability detection involves two steps: feature development using code analysis and training machine learning on derived characteristics to identify susceptible code segments [10].

In this research, the Optimal Ensemble Learning Approach for Cyber Security (AAMD-OELAC) approach for Automated Android Malware Detection is presented. At the first step, data preparation is done using the AAMDOELAC approach. Three machine learning models—the Regularised Random Vector Functional Link Neural Network (RRVFLN), the Kernel Extreme Learning Machine (KELM), and the Least Square Support Vector Machine (LS-SVM)—are used in the AAMD-OELAC technique's ensemble learning process for Android malware detection. Lastly, the three DL models' optimum parameter tuning is achieved by using the hunter-prey optimisation (HPO) technique, which contributes to better malware detection outcomes. To demonstrate the superiority of the AAMD-OELAC method, a thorough experimental investigation is conducted. The following is a brief summary of the major contributions.

• For Android malware detection, an intelligent AAMD-OELAC method that combines ensemble learning, data preparation, and HPO-based hyperparameter tweaking is provided. The AAMD-OELAC approach has never been documented in the literature, as far as we are aware.

LS-SVM, KELM, and RRVFLN models are used in an ensemble learning-based classification process for Android malware detection. The HPO algorithm and ensemble learning process work together to

International Journal of Information Technology & Computer Engineering
ISSN 2347–3657
Volume 13, Issue 2, 2025
https://doi.org/10.62647/ijitce.2025.v13.i2.pp370-381

increase the detection accuracy of Android malware. Malicious patterns and behaviours in Android apps may be successfully identified by the model via the use of several classifiers and optimisation techniques.

## 2. LITERATURE SURVEY

**"Adversarial superiority in Android malware detection: Insights from evasion attacks and defences based on reinforcement learning,"**

With billions of users now, Android devices have become a lucrative target for malware developers. It is thus more of a need than a desire for the anti-malware community and malware authors to stay one step ahead in this zero-sum game of malware detection. In order to create adversarially better Android malware detection models, our study focusses on a proactive adversary-aware framework. First, we examine the adversarial robustness of 36 different malware detection models built with 18 classification algorithms and two static features (intent and permission). To take advantage of flaws in the aforementioned malware detection models, we created two reinforcement learning-based Targeted Type-II Evasion Attacks (TRPO-MalEAttack and PPO-MalEAttack). In order to trick the malware detection algorithms, the attackers try to cause as little disruption as possible to each malicious program. With an average fooling rate of 95.75% (with 2.02 mean perturbations), the TRPO-MalEAttack lowers the average accuracy of 36 malware detection models from 86.01% to 49.11%. In contrast, the PPO-MalEAttack reduces the average accuracy from 86.01% to 48.65% in the same 36 detection models by achieving a greater average fooling rate of 96.87% (with 2.08 mean perturbations). Additionally, we create a list of the TEN most susceptible Android permissions and intents that an attacker may exploit to create additional malicious apps. In order to combat the adversarial assaults on malware detection algorithms, we subsequently provide a defence approach (MalVPatch). Higher detection accuracy and a significant increase in the adversarial resilience of malware detection models are both attained by the MalVPatch defence. Lastly, we draw the conclusion that in order to achieve adversarial supremacy in Android malware detection, it is essential to examine the adversarial robustness of models prior to their practical implementation.

**"You are what I was told by the permissions!" detection of Android malware using hybrid strategies,**

The usage of Android smartphones in many facets of our lives has significantly increased in recent years. Nevertheless, customers have the option to acquire Android applications via unaffiliated channels, which gives malware a lot of options. Attackers get sensitive user private information by using unsolicited permissions. There is an urgent need for effective and flexible antiviral solutions, particularly in novel versions, since signature-based solutions are no longer practicable. We suggest a hybrid Android malware detection method that blends static and dynamic strategies as a solution. First, using a machine-learning-based approach, we use static analysis to

International Journal of Information Technology & Computer Engineering
ISSN 2347–3657
Volume 13, Issue 2, 2025
https://doi.org/10.62647/ijitce.2025.v13.i2.pp370-381

infer distinct permission consumption patterns between malicious and benign applications. We create a dynamic feature base by extracting the object reference associations from the RAM heap in order to further identify the suspicious applications. Next, we provide an enhanced DAMBA-based state-based algorithm. Our method surpasses the popular detector with 97.5% F1-measure, according on experimental findings on a real-world dataset of 21,708 apps. Furthermore, it has been shown that our system is resistant to obfuscation tactics and permission abuse behaviours.

**"Deep learning model-based metaheuristics for cybersecurity and Android malware classification and detection,"**

Since information systems have advanced over the past ten years, cybersecurity has grown to be a major worry for several institutions, groups, and organisations. One of the most popular tools and strategies for carrying out an assault on Android devices is malware programs, and it's becoming harder to find new methods to detect them. To defend the Android operating system from these kinds of assaults, there are many malware detection methods available. Based on the patterns found in the characteristics of Android applications, these malware detectors classify the target apps. The defence systems of Android are adversely affected by the growing amount of analytics data. Feature selection strategies are proven to be useful since a high number of undesired characteristics causes a performance bottleneck for the detection system. A deep learning-based Android malware detection model called Rock Hyrax Swarm Optimisation with Deep Learning (RHSODL-AMD) is presented in this paper. This method effectively distinguishes between malicious and legitimate apps by identifying the most important permissions and Application Programming Interfaces (API) calls. To enhance the classification outcomes, an RHSO-based feature subset selection (RHSO-FS) method is developed. Additionally, Android malware detection uses the Adamax optimiser with attention recurrent autoencoder (ARAE) model. With a maximum accuracy of 99.05%, the RHSODL-AMD technique's experimental validation on the Andro-AutoPsy dataset demonstrates its promising performance.

**"A deep learning and static analysis-based approach for automatic Android malware detection,"**

Nowadays, the majority of internet users worldwide are switching from desktops to smartphones, with Android accounting for the largest share of the smartphone market. Since more people are using smartphones in general and the Android system in particular, there is a greater need to safeguard Android since malware authors are using complex and disguised malicious apps to target Android. As a result, several research were conducted to provide a reliable technique for identifying and categorising dangerous software (malware) for Android. Using datasets that have become obsolete and include programs for older Android versions that are seldom used today, some of them were successful, while others were not; some had accuracy below 90%. This study presents a novel approach

International Journal of Information Technology & Computer Engineering
ISSN 2347–3657
Volume 13, Issue 2, 2025
https://doi.org/10.62647/ijitce.2025.v13.i2.pp370-381

that uses static analysis to collect the most valuable aspects of Android apps, along with two additional features that are suggested. These features are then fed into a deep learning model that we developed for a functional API. A fresh and categorised dataset of Android applications was used to test this approach. A total of 14079 malware and benign samples were used, with the malware samples being divided into four malware groups. This dataset was used in two significant experiments: one for malware identification, in which samples were divided into two groups: benign and malicious. The second experiment was designed for malware detection and classification, using all five classes in the dataset. Consequently, our model outperforms the related studies with an F1-score of 99.5% when just two classes are used. Additionally, the five classes yielded strong malware detection and classification performance, with an F1-score of 97%.

## 3. EXISTING SYSTEM

A novel malware detection technique linked to DL is developed by Shaukat et al. [11]. By combining the advantages of static and dynamic analysis, it produced better results than traditional techniques. A portable executable (PE) file is first shown as coloured pictures. Second, it used a refined DL technique to extract deep features from colour photos. Thirdly, it detects malware associated with SVM's deep features. A technique known as innovative multi-view Android malware detection, used threefold, was introduced by Geremias et al. [12] employing image-based DL. First, applications were evaluated based on the various feature sets in multi-view settings, increasing the amount of data shown for categorisation. Second, the data for the classifier job is retained by converting the resulting feature set into picture formats while maintaining the crucial components of data distribution. Thirdly, DL structure may be implemented since created pictures are all shown in a single shot inside a preset image channel.

A malware detection system named MAPAS, which achieves greater accuracy and flexible use of computing resources, was simulated by Kim et al. [13]. MAPAS used CNN to analyse harmful applications' performance based on their API call graphs. The proposed MAPAS strategy, on the other hand, leverages CNN to identify typical characteristics of the malware's API call graph rather than a CNN-produced classifier method. Fallah and Bidgoly [14] created an LSTM-related malware detection method that can distinguish between malware and benign samples as well as find and detect novel and undiscovered malware kinds. The author of this paper has conducted several tests to demonstrate the capabilities of the method that is being given, such as identifying malware families, detecting new malware families, and determining the least amount of time required to locate malware.

Sihag et al. [15] used DYnamic features (De-LADY), a robust obfuscation technique, to introduce DL-based Android malware detection. Behavioural characteristics from dynamic analysis of a program run in the

International Journal of Information Technology & Computer Engineering
ISSN 2347–3657
Volume 13, Issue 2, 2025
https://doi.org/10.62647/ijitce.2025.v13.i2.pp370-381

simulated environment were employed. A hybrid approach relating to CNN and DAE is presented by Wang et al. [16]. First, the author recreated the high-dimensional features of applications and used several CNNs to identify Android malware in order to improve the accuracy of malware detection. Second, the author used DAE as a pre-training strategy for CNN in order to shorten the training time. The DAE and CNN approach (DAE-CNN) can swiftly analyse flexible patterns thanks to the consolidation.

A performance comparison of 26 pretrained CNN techniques currently in use for Android malware detection was given by Yadav et al. [17]. Based on the results, an EfficientNet-B4 CNN-based method was developed to detect Android malware using an image-based malware representation of the Android DEX file. EfficientNet-B4 collected pertinent properties from the malware pictures. Droid-NNet is a DL structure that Masum and Shahriar [18] developed for malware classification. However, Droid-NNet is a deep learner that outperforms current state-of-the-art machine learning techniques. In order to identify Android malicious apps, Idrees et al. [19] investigate PIndroid, a novel framework based on permissions and intents. As is well known, the main solution is PIndroid, which employs a collection of permissions and intents in addition to Ensemble techniques to accurately identify malware. The authors of [20] demonstrate that after discussing idea drift, permissions produce effective and long-lasting malware detection techniques. Taha and Barukab [21] provide a method for classifying Android malware that relies on GA and optimiser ensemble learning. To achieve the highest Android malware classification accuracy, the GA was used to optimise the RF technique's parameter values. Using CNN approaches, Sabanci et al. [22] aimed to classify pepper seeds from different cultivars. Two classification approaches are provided. First, pepper seeds are used to train the CNN techniques (ResNet50 and ResNet18). The features of pre-training CNN techniques are fused, and feature selection has been carried out to the fused features. The first is secondary and diversified. The authors of [23] look into new techniques used for Android malware detection. Consequently, an overview of the Android system revealed the fundamental mechanisms and issues with its security framework.

**Disadvantages**

• Data complexity: To identify Android malware, the majority of machine learning models now in use need to be able to correctly analyse large and intricate datasets.

• Data availability: In order to provide precise predictions, the majority of machine learning models need a lot of data. The accuracy of the model may degrade if data is not accessible in large enough amounts.

• Inaccurate labelling: The accuracy of the machine learning models that are now in use depends on how well the input dataset was used for training. Inaccurate labelling of the data prevents the model from producing reliable predictions.

## 4. PROPOSED SYSTEM

International Journal of Information Technology & Computer Engineering
ISSN 2347–3657
Volume 13, Issue 2, 2025
https://doi.org/10.62647/ijitce.2025.v13.i2.pp370-381

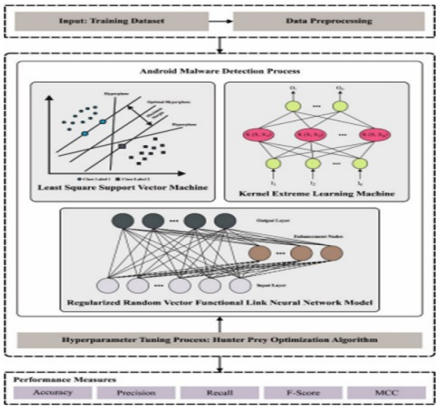
In this research, the Optimal Ensemble Learning Approach for Cybersecurity (AAMD-OELAC) approach for Automated Android Malware Detection is presented. At the first step, data preparation is done using the AAMDOELAC approach. Three machine learning models—the Regularised Random Vector Functional Link Neural Network (RRVFLN), the Kernel Extreme Learning Machine (KELM), and the Least Square Support Vector Machine (LS-SVM)—are used in the AAMD-OELAC technique's ensemble learning process for Android malware detection. Lastly, the three DL models' optimum parameter tuning is achieved by using the hunter-prey optimisation (HPO) technique, which contributes to better malware detection outcomes. To demonstrate the superiority of the AAMD-OELAC method, a thorough experimental investigation is conducted.

**Advantages**

• For Android malware detection, an intelligent AAMD-OELAC method that combines ensemble learning, data preparation, and HPO-based hyperparameter tweaking is provided. The AAMD-OELAC approach has never been documented in the literature, as far as we are aware.

• Use LS-SVM, KELM, and RRVFLN models in an ensemble learning-based classification approach to identify Android malware. The accuracy of Android malware detection is increased by combining the HPO algorithm with the ensemble learning process. Malicious patterns and behaviours in Android apps may be successfully identified by the model via the use of several classifiers and optimisation techniques.

## 5. SYSTEM ARCHITECTURE




## 6. IMPLEMENTATION

**Modules**

**Service Provider**

The Service Provider must use a working user name and password to log in to this module. Following a successful login, he may do several tasks including training and testing data sets, Discover the Predicted Android Malware Detection Ratio, Download Predicted Datasets, View Trained and Tested Accuracy in a Bar Chart, View Trained and Tested Accuracy Results, and View Predicted Android Malware Detection Details View All Remote Users and the Android Malware Predicted Ratio Results.

**View and Authorize Users**

The administrator may see a list of all registered users in this module. Here, the administrator may see the user's information, like name, email, and address, and they can also grant the user permissions.

**Remote User**

A total of n users are present in this module. Before beginning any actions, the user needs

International Journal of Information Technology & Computer Engineering
ISSN 2347–3657
Volume 13, Issue 2, 2025
https://doi.org/10.62647/ijitce.2025.v13.i2.pp370-381

register. Following registration, the user's information will be entered into the database. Following a successful registration, he must use his password and authorised user name to log in. Following a successful login, the user may do tasks including registering and logging in, predicting the kind of Android malware, and seeing their profile.

## 7. ALGORITHIMS

### Naïve Bayes

The supervised learning technique known as the "naive bayes approach" is predicated on the straightforward premise that the existence or lack of a certain class characteristic has no bearing on the existence or nonexistence of any other feature.

However, it seems sturdy and effective in spite of this. It performs similarly to other methods of guided learning. Numerous explanations have been put forward in the literature. We emphasise a representation bias-based explanation in this lesson. Along with logistic regression, linear discriminant analysis, and linear SVM (support vector machine), the naive bayes classifier is a linear classifier. The technique used to estimate the classifier's parameters (the learning bias) makes a difference.

Although the Naive Bayes classifier is commonly used in research, practitioners who want to get findings that are useful do not utilise it as often. On the one hand, the researchers discovered that it is very simple to build and apply, that estimating its parameters is simple, that learning occurs quickly even on extremely big datasets, and that, when compared to other methods, its accuracy is rather excellent. The end users, however, do not comprehend the value of such a strategy and do not get a model that is simple to read and implement.

As a consequence, we display the learning process's outcomes in a fresh way. Both the deployment and comprehension of the classifier are simplified. We discuss several theoretical facets of the naive bayes classifier in the first section of this lesson. Next, we use Tanagra to apply the method on a dataset. We contrast the outcomes (the model's parameters) with those from other linear techniques including logistic regression, linear discriminant analysis, and linear support vector machines. We see that the outcomes are quite reliable. This helps to explain why the strategy performs well when compared to others. We employ a variety of tools (Weka 3.6.0, R 2.9.2, Knime 2.1.1, Orange 2.0b, and RapidMiner 4.6.0) on the same dataset in the second section. Above all, we make an effort to comprehend the outcomes.

### Logistic regression Classifiers

The relationship between a collection of independent (explanatory) factors and a categorical dependent variable is examined using logistic regression analysis. When the dependent variable simply has two values, like 0 and 1 or Yes and No, the term logistic regression is used. When the dependent variable contains three or more distinct

International Journal of Information Technology & Computer Engineering
ISSN 2347–3657
Volume 13, Issue 2, 2025
https://doi.org/10.62647/ijitce.2025.v13.i2.pp370-381

values, such as married, single, divorced, or widowed, the technique is sometimes referred to as multinomial logistic regression. While the dependent variable's data type differs from multiple regression's, the procedure's practical application is comparable.

When it comes to categorical-response variable analysis, logistic regression and discriminant analysis are competitors. Compared to discriminant analysis, many statisticians believe that logistic regression is more flexible and appropriate for modelling the majority of scenarios. This is due to the fact that, unlike discriminant analysis, logistic regression does not presume that the independent variables are regularly distributed.

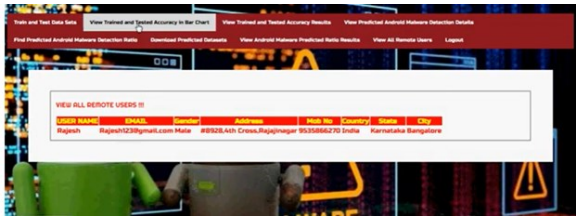
Both binary and multinomial logistic regression are calculated by this software for both category and numerical independent variables. Along with the regression equation, it provides information on likelihood, deviance, odds ratios, confidence limits, and quality of fit. It does a thorough residual analysis that includes diagnostic residual plots and reports. In order to find the optimal regression model with the fewest independent variables, it might conduct an independent variable subset selection search. It offers ROC curves and confidence intervals on expected values to assist in identifying the optimal classification cutoff point. By automatically identifying rows that are not utilised throughout the study, it enables you to confirm your findings.

**Decision tree classifiers**

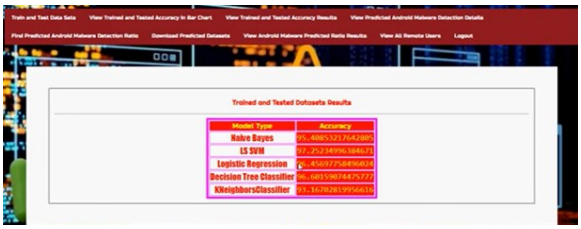
Decision tree classifiers are effectively used in a wide range of fields. Their capacity to extract descriptive decision-making information from the provided data is their most crucial characteristic. Training sets may be used to create decision trees. The following is the process for this kind of creation based on the set of objects (S), each of which belongs to one of the classes C1, C2, …, Ck:

Step 1: The decision tree for S has a leaf labelled with the class if every item in S is a member of the same class, such as Ci. Step 2. If not, let T be a test with the potential results O1, O2,..., On. The test divides S into subsets S1, S2,… Sn, where each item in Si has result Oi for T, because each object in S has a single outcome for T. T serves as the decision tree's root, and we construct a subsidiary decision tree for each outcome Oi by recursively applying the same process to the set Si.

## 8. SCREENSHOTS

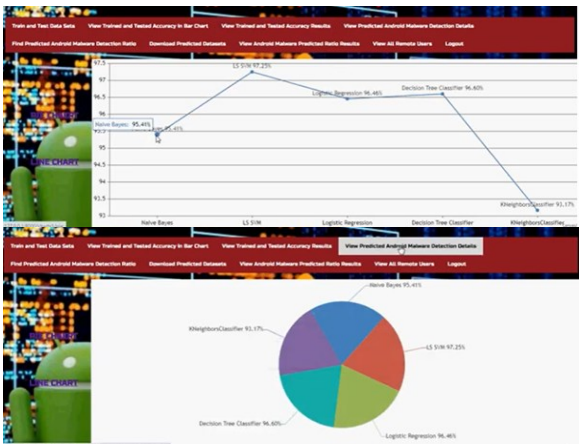



International Journal of Information Technology & Computer Engineering
ISSN 2347–3657
Volume 13, Issue 2, 2025
https://doi.org/10.62647/ijitce.2025.v13.i2.pp370-381



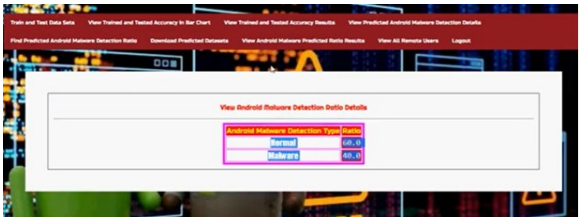




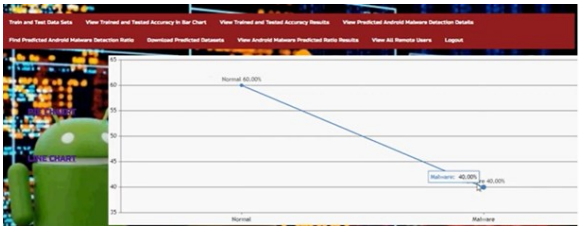


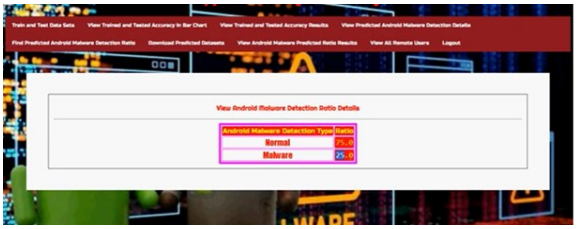









## 9. CONCLUSION AND FUTURE ENHANCEMENT

We have developed the AAMD-OELAC approach in this work to identify Android malware accurately and automatically. The goal of the AAMD-OELAC method was to automatically identify and categorise Android malware. The AAMD-OELAC

International Journal of Information Technology & Computer Engineering
ISSN 2347–3657
Volume 13, Issue 2, 2025
https://doi.org/10.62647/ijitce.2025.v13.i2.pp370-381

approach uses ensemble classification, data preprocessing, and HPO-based parameter adjustment to accomplish this. The AAMD-OELAC approach uses three machine learning models—LS-SVM, KELM, and RRVFLN—as part of an ensemble learning process for Android malware detection. Lastly, the HPO technique is used to optimise the three DL models' parameter tuning, which leads to better malware detection outcomes. A comprehensive experimental investigation is carried out to demonstrate the superiority of the AAMD-OELAC approach. The simulation results demonstrated the AAMDOELAC technique's superiority over other methods currently in use.

In order to improve the detection of complex malware, future research might concentrate on creating more sophisticated methods for capturing and analysing fine-grained behaviours. Future research might also examine privacy-preserving strategies like federated learning or safe multi-party computing, which allow for cooperative malware detection without jeopardising user privacy.

**REFERENCES**

[1] H. Rathore, A. Nandanwar, S. K. Sahay, and M. Sewak, ‘‘Adversarial superiority in Android malware detection: Lessons from reinforcement learning based evasion attacks and defenses,’’ *Forensic Sci. Int., Digit. Invest.*, vol. 44, Mar. 2023, Art. no. 301511.

[2] H. Wang, W. Zhang, and H. He, ‘‘You are that the permissions told me! Android malware detection based on hybrid tactics,’’ *J. Inf. Secur. Appl.*, vol. 66, May 2022, Art. no. 103159.

[3] A. Albakri, F. Alhayan, N. Alturki, S. Ahamed, and S. Shamsudheen, ‘‘Metaheuristics with deep learning model for cybersecurity and Android malware detection and classification,’’ *Appl. Sci.*, vol. 13, no. 4, p. 2172, Feb. 2023.

[4] M. Ibrahim, B. Issa, and M. B. Jasser, ‘‘A method for automatic Android malware detection based on static analysis and deep learning,’’ *IEEE Access*, vol. 10, pp. 117334–117352, 2022.

[5] L. Hammood, İ. A. Doğru, and K. Kılıç, ‘‘Machine learning-based adaptive genetic algorithm for Android malware detection in auto-driving vehicles,’’ *Appl. Sci.*, vol. 13, no. 9, p. 5403, Apr. 2023.

[6] P. Bhat and K. Dutta, ‘‘A multi-tiered feature selection model for Android malware detection based on feature discrimination and information gain,’’ *J. King Saud Univ.-Comput. Inf. Sci.*, vol. 34, no. 10, pp. 9464–9477, Nov. 2022.

[7] D.Wang, T. Chen, Z. Zhang, and N. Zhang, ‘‘A survey of Android malware detection based on deep learning,’’ in *Proc. Int. Conf. Mach. Learn. Cyber Secur.* Cham, Switzerland: Springer, 2023, pp. 228–242.

[8] Y. Zhao, L. Li, H. Wang, H. Cai, T. F. Bissyandé, J. Klein, and J. Grundy, ‘‘On the impact of sample duplication in machine-learning-based Android

International Journal of Information Technology & Computer Engineering
ISSN 2347–3657
Volume 13, Issue 2, 2025
https://doi.org/10.62647/ijitce.2025.v13.i2.pp370-381

malware detection,'' *ACM Trans. Softw. Eng. Methodol.*, vol. 30, no. 3, pp. 1–38, Jul. 2021.

[9] E. C. Bayazit, O. K. Sahingoz, and B. Dogan, ''Deep learning based malware detection for Android systems: A comparative analysis,'' *Tehnički vjesnik*, vol. 30, no. 3, pp. 787–796, 2023.

[10] H.-J. Zhu, W. Gu, L.-M. Wang, Z.-C. Xu, and V. S. Sheng, ''Android malware detection based on multi-head squeeze-and-excitation residual network,'' *Expert Syst. Appl.*, vol. 212, Feb. 2023, Art. no. 118705.

[11] K. Shaukat, S. Luo, and V. Varadharajan, ''A novel deep learning-based approach for malware detection,'' *Eng. Appl. Artif. Intell.*, vol. 122, Jun. 2023, Art. no. 106030.

[12] J. Geremias, E. K. Viegas, A. O. Santin, A. Britto, and P. Horchulhack, ''Towards multi-view Android malware detection through image-based deep learning,'' in *Proc. Int.Wireless Commun. Mobile Comput. (IWCMC)*, May 2022, pp. 572–577. 72516 VOLUME 11, 2023 IEEE Transaction on MachineLearning,Volume:11,Issue Date:11.July.2023

[13] J. Kim, Y. Ban, E. Ko, H. Cho, and J. H. Yi, ''MAPAS: A practical deep learning-based Android malware detection system,'' *Int. J. Inf. Secur.*, vol. 21, no. 4, pp. 725–738, Aug. 2022.

[14] S. Fallah and A. J. Bidgoly, ''Android malware detection using network traffic based on sequential deep learning models,'' *Softw., Pract. Exper.*, vol. 52, no. 9, pp. 1987–2004, Sep. 2022.

[15] V. Sihag, M. Vardhan, P. Singh, G. Choudhary, and S. Son, ''De-LADY: Deep learning-based Android malware detection using dynamic features,'' *J. Internet Serv. Inf. Secur.*, vol. 11, no. 2, p. 34, 2021.