Cloud-Network-End Collaborative Security-Automated Android Malware Detection Using Optimal Ensemble Learning Approach

Dr R Jegadeesan[1], Dr Midhun chakkaravarthy[2], Dr. Mudassir Khan[3]

[1]Professor & Head- Computer Science and Engineering, Jyothishmathi Institute of Technology and Science, karimnagar, India, Corresponding Authors: [2]Professor, Lincoln University College Malaysia, [3]Assistant Professor, Department of Computer Science, College of Computer Science, Applied College Tanumah, King Khalid University, Saudi Arabia.

Email ID: ramjaganjagan@gmail.com[1], midhun@lincoln.edu.my[2], mkmiyob@kku.edu.sa[3]

Abstract: As processing power has increased, many people living in virtual worlds have found that their lives are considerably simpler. Software threats frequently target computers in Cloud-Network-End Collaborative Security. Malware is constantly changing because of better packaging and obfuscation tactics. Using these techniques makes detecting and identifying malware more difficult. Instead of relying solely on outdated procedures, it is essential to design new and inventive strategies of Cloud-Network-End Collaborative Security to contain the spread of infections. There are too many complex diseases for machine learning (ML) systems to distinguish between them all. It might be simpler to identify a person's bug type using deep learning (DL) technology. In this proposal, we will build an automated malware detection system using Cloud-Network-End Collaborative Security for Android phones using the AAMD-OELAC algorithm. It is one of the primary objectives of AAMDOELAC to detect and classify malware autonomously. In this regard, The technique uses three ML models (Security mechanism for end system security, Network Connection Security, Cloud Service Security) to identify Android phone issues. The least square support vector machine (LS-SVM), RRVFLN, and KELM are only a few instances of such. Along with the three DL models, the hunter-prey optimization (HPO) approach is used for optimization. It is believed that more experimental investigations are required to completely comprehend the method's operation.

Keywords: Cloud-Network-End Collaborative Security, machine learning; Deep learning; support vector machine; hunter-prey optimization.

Introduction

Despite their busy schedules, Cloud-Network-End Collaborative Security is a top priority for computer scientists and network specialists. As technology is integrated into people's daily lives more and more, there is a corresponding increase in awareness of possible dangers and harmful programs. Software that is compatible with Android is the most popular type of software that can be found online. At the moment, Android has more users than any other OS. Malicious software sometimes tries to avoid

detection by having fewer than fifty features. Finding, halting, or removing Android software is essential, as is devising measures to stop its spread. Because of these worries, the organization's employees are constantly scanning for malware and responding accordingly to stop its spread. That being said, researchers Android users can detect malware in one of three ways: following regular procedures, using alternative approaches, or combining the two. Although static analysis does not necessitate any software to run, it is nevertheless possible to identify non-functional apps. However, the virus authors hid their code from static defences and made the study more difficult. In order to identify malware while it is running, a technique called "dynamic analysis" might be employed. While dynamic analysis can detect malware in action, static analysis is more commonly employed to find malware in source code. Unfortunately, malware is a common source of unnecessary danger for both Android device users and manufacturers. In this research, we look at various approaches to malware detection. Malware in Android APKs can be detected using a set of characteristics generated by the machine learning model. By utilizing methods such as deep learning (DL) and machine learning (ML), potentially harmful APKs can be detected. Malware detection is not the same as software defect detection. To begin, code analysis provides traits. After that, we use machine learning to train the features to differentiate between vulnerable code portions.

This article discusses the best ensemble learning method for Cloud-Network-End Collaborative Security, AAMDOELAC, and techniques for automatically identifying malware on Android devices. Data is first organized using the AAMDOELAC method. The AAMD-OELAC system uses three distinct group learning techniques to detect issues on Android phones. The Least Square Support Vector Machine (LS-SVM), the Kernel Extreme Learning Machine (KELM), and the Regularized Random Vector Functional Link Neural Network (RVFLN) are the following. Last but not least, the hunter-prey optimization (HPO) method is used to optimize all three DL models. The AAMDOELAC method is proven to be superior in a thorough experiment. Here you can find a concise overview of the key proposals. Data preparation, ensemble learning, and HPO-based hyper parameter change are the three main components of an intelligent AAMD-OELAC approach to malware detection on Android devices. The AAMD-OELAC proposal has not yet produced any final papers.

Find Android malware using an ensemble learning-based classification approach that combines the LS-SVM, KELM, and RRVFLN models. The combination of the HPO algorithm with the ensemble learning method allows Android devices to detect malware faster. Using a wide range of optimization techniques and classifiers, the system quickly scans Android apps for potentially harmful patterns and actions.

Related work

Shaukat Discover a cutting-edge deep learning method for identifying malicious software. Compared to other methods, comparative analysis is better since it combines the greatest parts of static and dynamic analysis. At the beginning of the copying process for a PE file, colorful images are shown. The next step was to apply a fresh deep learning framework specifically for color picture feature extraction. That's doable because it leverages SVM's core capabilities, which effectively eliminate malware. Using three separate deep learning models, Geremias developed an excellent multi- view malware detection app for Android. As a first step in our app review, we explored the available feature sets through the different view options. A bigger quantity of data can be partitioned into several sets

in this way. Secondly, it is still possible to access the data used by the classifiers, even after saving the final features to an image file. This way, the most important parts of the information will stand out. Thirdly, the DL structure renders all images from a specific channel in real time[10].

Kim et al. By developing a model, MAPAS was able to better allocate its resources to identify malware. In order to assess the extent to which malicious apps function, MAPAS employs CNN to analyze API call logs. This article discusses the MAPAS method as an alternative to CNN's classifier approach. This method traverses the API call tree of the virus in search of comparable attributes. To identify malicious software, Fallah and Bidgoly created algorithms based on long short-term memory (LSTM). With these methods, you can detect malware you didn't know existed and tell safe samples apart from dangerous ones. In order to prove that the relevant technology worked, the study's author ran a battery of tests. Three groups were formed from the test results: malware identification, fastest malware detection, and new malware subgroup discovery[12].

Sihag et al. One effective method of using Deep Learning to obfuscate malware on Android devices is De-LADY, which stands for dynamic features. The methodology was developed using emulators that allowed for real-time app testing and the identification of behavioral patterns. Wang et al. suggested a way to combine CNN and DAE. The author improved the accuracy of Android malware detection by fully utilizing CNNs and modifying high-dimensional app properties. For faster training, the author used CNN's pre-training method, DAE. It is possible to study trends that are changing at a rapid pace by combining DAE and CNN (DAE-CNN)[14].

Yadav et al. Over 26 popular CNN algorithms were tested for their ability to detect Android vulnerabilities that had already been trained. The research led to the creation of a system that can identify malicious apps on Android by utilizing EfficientNet-B4 CNN[16]. Malicious software that poses as an Android DEX file is distributed using this technique. In order to find useful features, the EfficientNet-B4 tool could search through pictures of malware. For the purpose of making malware classification easier, Masum and Shahriar [18] created the Droid-NNet DL framework. Thanks to its deep learning capabilities, Droid-NNet remains the most intelligent machine learning system available today. Idrees et al. developed Pin droid, a novel approach to detecting malicious Android applications by analysing their objectives and authorizations. By integrating method, intention, and permission lookups, Android is the only platform that can detect malware. The authors of prove that people have the right to build drift- resistant, long-term malware detection systems. An innovative approach to malware detection on Android phones is proposed by Taha and Barukab. It employs ensemble learning and evolutionary algorithms. In order to improve the accuracy of Android malware classification, the RF technique's parameters were tweaked using the GA. Sabanci et al. The plan was to use convolutional neural network (CNN) methods to classify pepper seed varieties. Two classification schemes will be considered now. Pepper seeds were used to train the initial CNN algorithms, ResNet50 and ResNet18. After acquiring secondary features that can be modified using the original CNN techniques, the newly acquired features are subsequently used in feature selection. The authors are actively developing cutting-edge methods for identifying malware on Android devices. This means that some of the most important features and security holes in the Android system were revealed while the investigation was underway.

PROPOSED SYSTEM

Security mechanism for end system security, Network Connection Security, Cloud Service Security method is the principal goal of this proposal. The data is first combined using the AAMDOELAC method. In order to find Android phone vulnerabilities, AAMD-OELAC employs three different kinds of machine learning. That is, (Cloud-Network-End Collaborative Security) RRVFLN, LS-SVM, and KELM are all part of it. Utilizing the hunter-prey optimization (HPO) technique further improves all three DL models. In the future, this will greatly facilitate the detection of viruses. The AAMD-OELAC method was found to be superior in an extensive test.

ADVANTAGES

- In order to identify malicious apps on Android devices, the Security mechanism for end system security, Network Connection Security, Cloud Service Security system employs strategies such as group learning, data preparation, and hyper parameter adjustments based on HPO. We regret to inform you that we are currently unaware of any proposal materials associated with AAMD-OELAC.
- One way to detect Android infections is by using a technique called "ensemble learning" that combines the LS-SVM, KELM, and RRVFLN models. The HPO algorithm and ensemble learning make it easier for Android to detect malware. The model employs a number of classifiers and optimization strategies to search for problematic patterns and behaviors within Android applications.

SYSTEM ARCHITECTURE

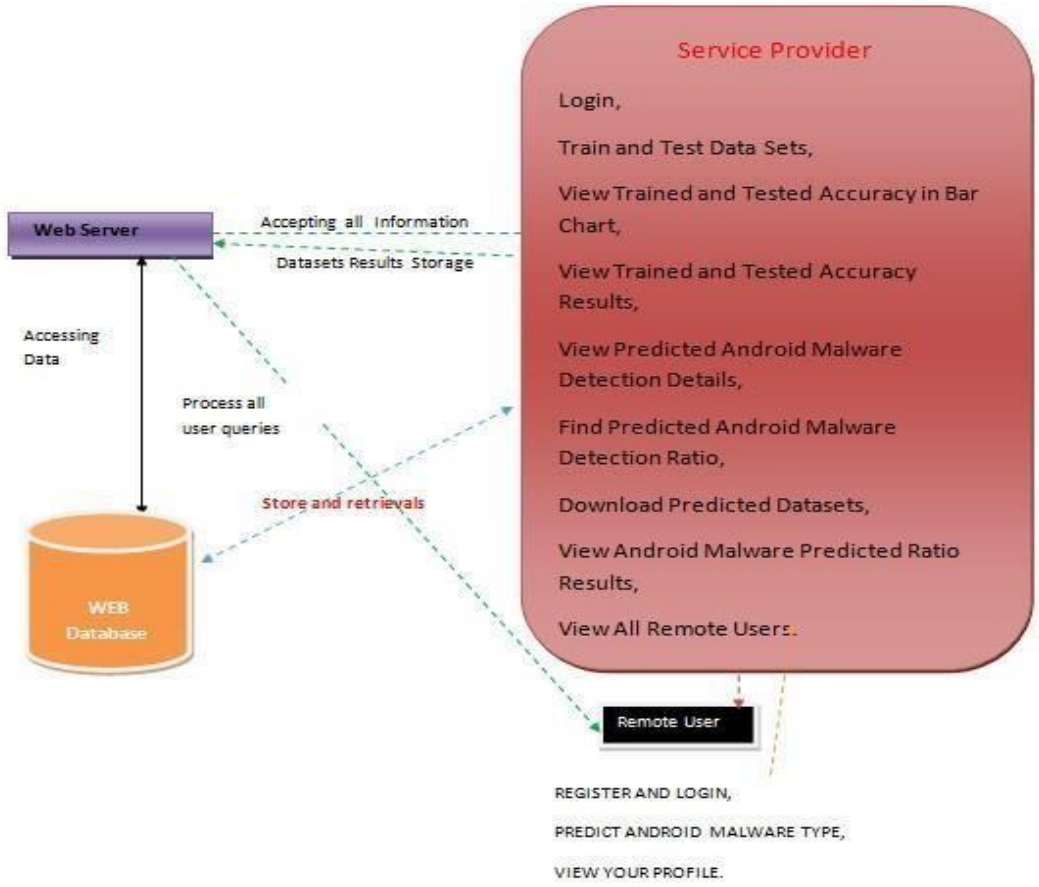

Figure 1.1 System Architecture

IMPLEMENTATION

MODULES

Service Provider

This module can only be accessed by the Service Provider who has the necessary keys. A bar chart displays the user's accuracy in both the training and testing processes. Moreover, they have the ability to peruse a directory of all remote users, acquire datasets that are predicted to be used, and learn more about the anticipated detection of Android malware.

Remote User

At least n people with ties to this region are active here. Prior to proceeding, it is necessary for every user to register. A directory is maintained with an individual's registration details. Users are required to log in with their credentials after successfully completing the registration process. After logging in,

users have the ability to evaluate their accounts, register, and verify their profiles. This empowers them to make informed judgments on potential Android infections.

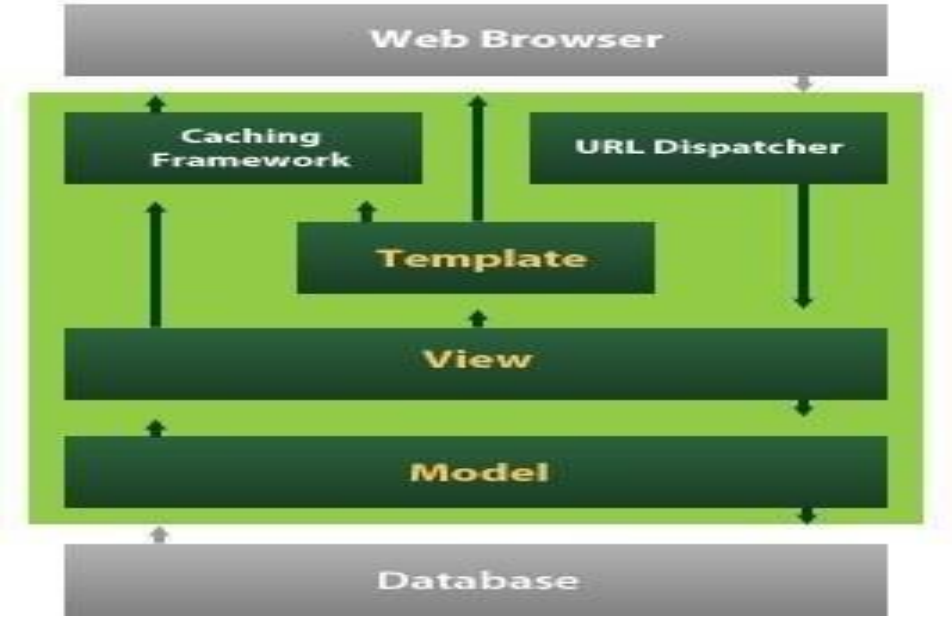

Figure 1.2 The administration interface of Django allows users to read, add, modify, and delete records. This interface is dynamically created and altered by the admin models using reflection.

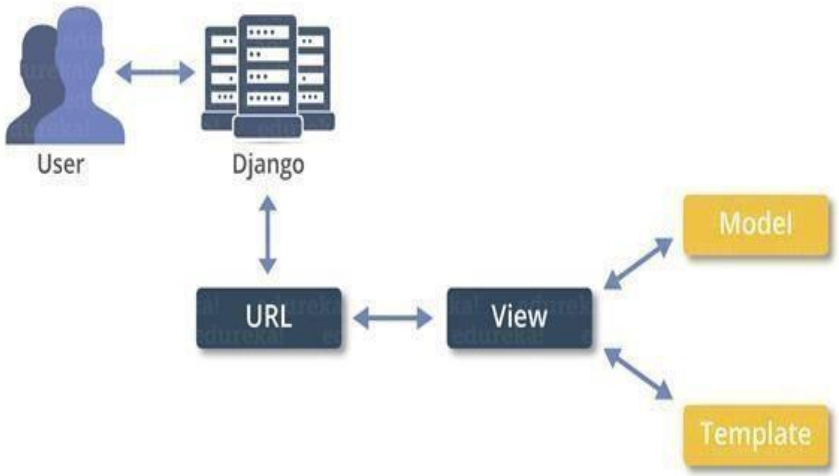

Figure 1.3 The administration interface of Django allows users to read, add, modify, and delete records. This interface is dynamically created and altered by the admin models using reflection.

Experiments and Results

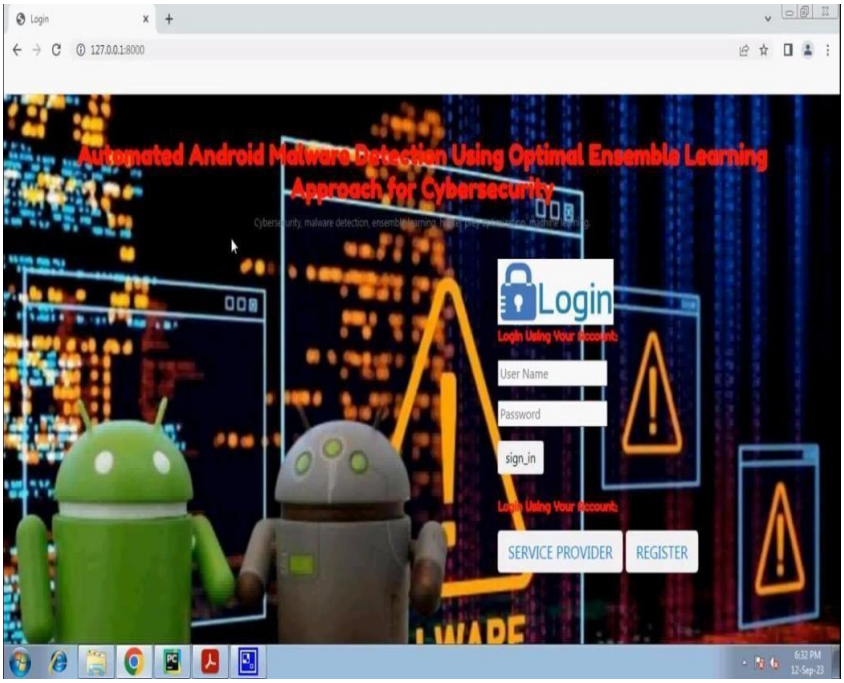

Figure 1.4 User Login Page

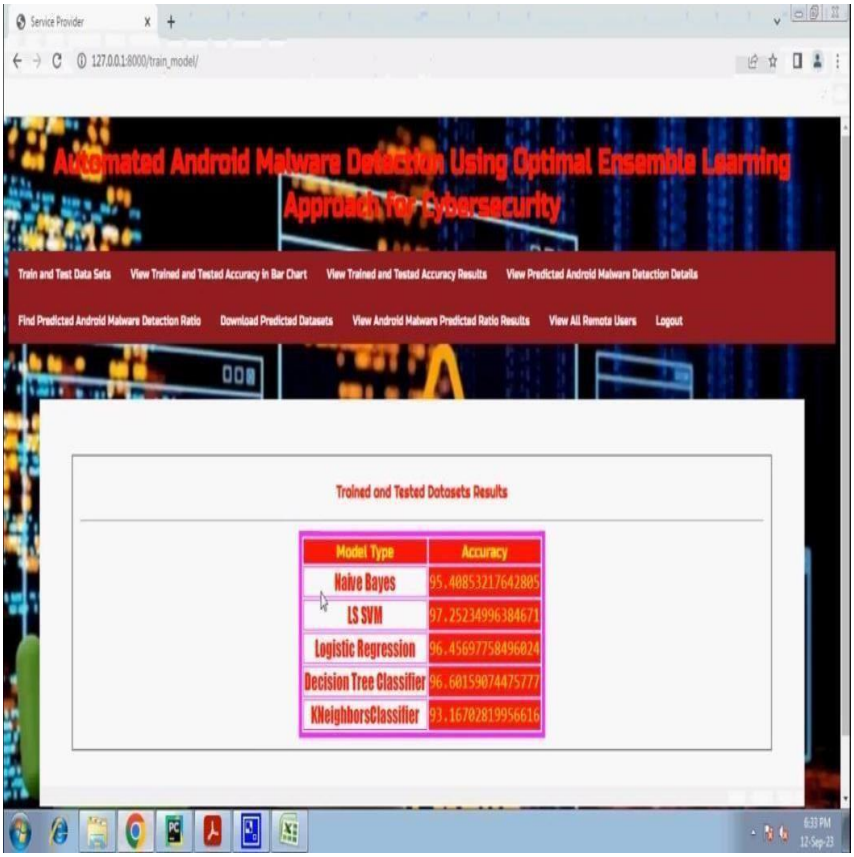

Figure: 1.5 Tested and Trained Datasets Details

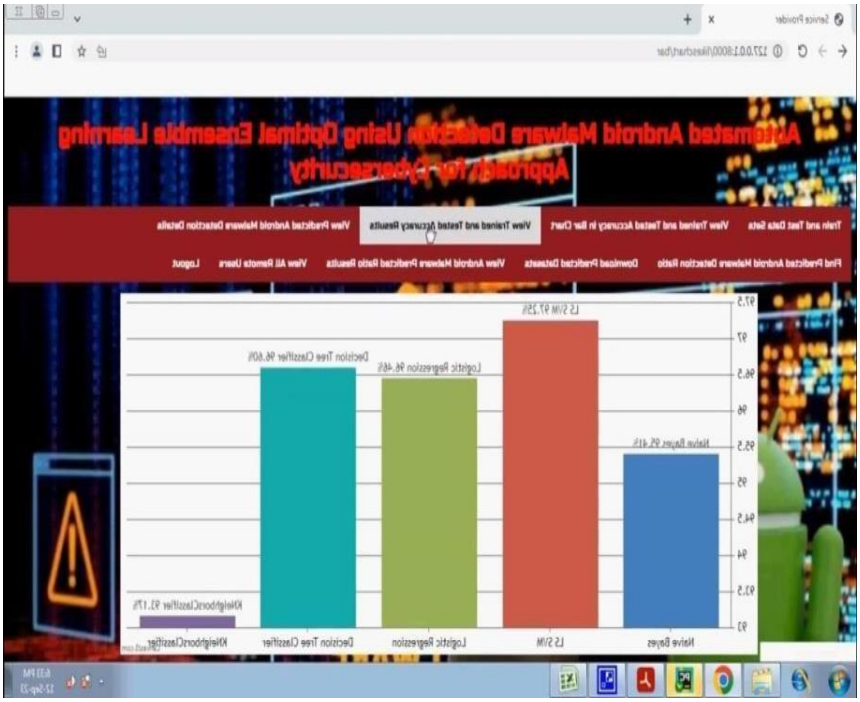

Figure: 1.6 Tested accuracy results in Barchart

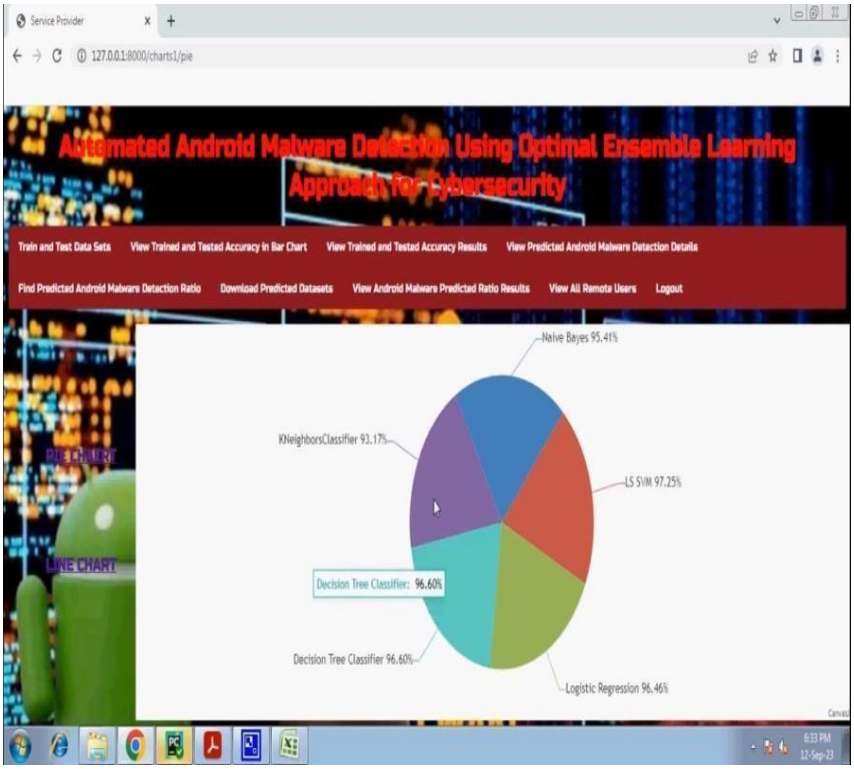

Figure: 1.7 Tested and Trained Accuracy Results in pie chart

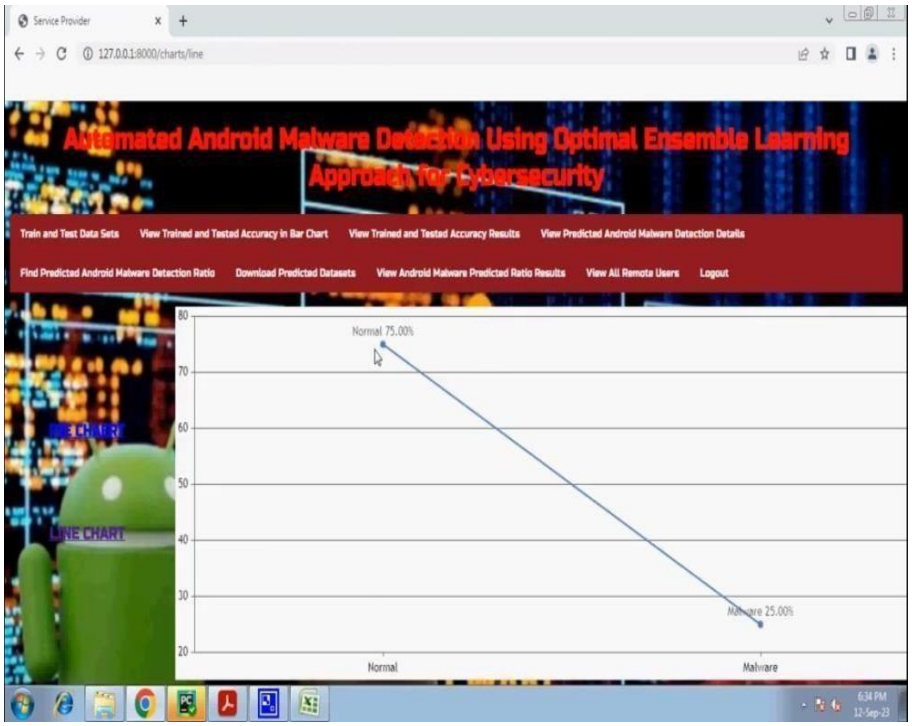

Figure: 1.8 Android Malware Prediction Ratio Details in line chart

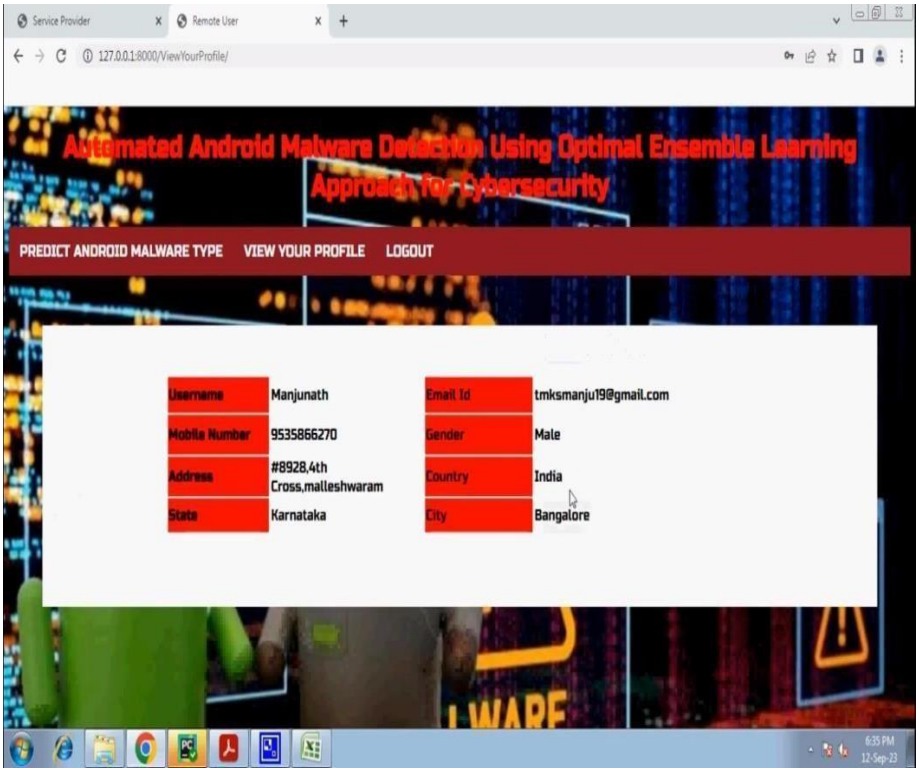

Figure: 1.9 User profile page

CONCLUSION

Our Proposal on automated malware detection for Android led to the development (Cloud-Network-End Collaborative Security) of the very effective AAMD-OELAC method. Finding and categorizing Android malware automatically is the aim of the AAMD-OELAC approach. This is accomplished by employing the AAMD-OELAC approach. Ensemble classification, data preparation, and parameter optimization based on HPO are all part of this approach. By integrating LS-SVM, KELM, and RRVFLN into a single approach known as "ensemble learning," the AAMD-OELAC application is able to detect malfunctions in Android devices. By optimizing the parameters of the three DL models, the HPO technique ultimately enhances malware detection. There is substantial scientific support for the AAMD- OELAC approach. The AAMDOELAC approach was the only one that showed any improvement over the other methods examined in the experiment. Finding and studying tiny behaviors more effectively might make it easier to detect and remove more complicated viruses in the future. Additional research on federated learning and secure multi-party computing is necessary to identify malware in a way that safeguards privacy.

FUTURE SCOPE

In addition, future work could explore privacy-preserving approaches such as secure multi-party computation or federated learning, which enable collaborative malware detection without compromising 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. https://doi.org/10.1016/j.fsidi.2023.301511
2. H. Wang, W. Zhang, and H. He, ‘‘You are what the permissions told me! Android malware detection based on hybrid tactics,’’ J. Inf. Secur. Appl., vol. 66, May 2022, Art. no. 103159. https://doi.org/10.1016/j.jisa.2022.103159
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,’’ IEEEAccess, vol. 10, pp.117334–117352, 2022. **https://doi.org/10.3390/app13042172**
4. 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. **https://doi.org/10.3390/app13095403**

5. 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. https://doi.org/10.1016/j.jksuci.2021.11.004
6. 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. CyberSecur. Cham, Switzerland: Springer, 2023, pp. 228–242. DOI: 10.1109/ACCESS.2020.3028370
7. 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 malware detection,’’ ACM Trans. Softw. Eng. Methodol., vol. 30, no. 3, pp. 1–38, Jul. 2021. https://doi.org/10.1145/3446905
8. E. C. Bayazit, O. K. Sahingoz, and B. Dogan, ‘‘Deep learning-based malware detection for Android systems: A comparative analysis,’’ Tehničkivjesnik, vol. 30, no. 3, pp. 787–796, 2023. https://doi.org/10.17559/TV-20220907113227
9. 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. https://doi.org/10.1016/j.eswa.2022.118705
10. 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. https://doi.org/10.1016/j.engappai.2023.106030
11. 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.WirelessCommun. Mobile Comput. (IWCMC), May 2022, pp.572–577. 72516 VOLUME 11, 2023 IEEE Transaction on Machine Learning, Volume:11, Issue Date:11. July.2023 DOI: 10.1109/IWCMC55113.2022.9824985
12. J. Kim, Y. Ban, E. Ko, H. Cho, and J. H. Yi, ‘‘MAPAS: A practical deep learningbased Android malware detection system,’’ Int. J. Inf. Secur., vol. 21, no. 4, pp.725–738, Aug. 2022. https://link.springer.com/article/10.1007/s10207-022-00579-6
13. 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. **https://doi.org/10.1002/spe.3112**
14. 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. DOI: 10.22667/JISIS.2021.05.31.034
15. W. Wang, M. Zhao, and J. Wang, ‘‘Effective Android malware detection with a hybrid model based on deep autoencoder and convolutional neural network,’’ J. Ambient Intell. Humanized Comput., vol. 10, no. 8, pp. 3035–3043, Aug. 2019. DOI: 10.1007/s12652-018-0803-6
16. P. Yadav, N. Menon, V. Ravi, S. Vishvanathan, and T. D. Pham, ‘‘Efficient- Net convolutional neural networks-based Android malware detection,’’ Comput.Secur., vol. 115, Apr. 2022, Art. no. 102622. DOI: 10.1016/j.cose.2022.102622

17. M. Masum and H. Shahriar, ‘‘Droid-NNet: Deep learning neural network for Android malware detection,’’ in Proc. IEEE Int. Conf. Big Data (Big Data), Dec.2019, pp. 5789–5793. DOI: 10.1109/BigData47090.2019.9006053

18. F. Idrees, M. Rajarajan, M. Conti, T. M. Chen, and Y. Rahulamathavan, ‘‘PIndroid: A novel Android malware detection system using ensemble learning methods,’’ Comput. Secur., vol. 68, pp. 36–46, Jul. 2017. DOI: 10.1016/j.cose.2017.03.011
19. A. Guerra-Manzanares, H. Bahsi, and M. Luckner, ‘‘Leveraging the first line of defense: A study on the evolution and usage of Android security permissions for enhanced Android malware detection,’’ J. Comput. Virol.Hacking Techn., vol. 19, no. 1, pp. 65–96, Aug. 2022. DOI:10.1007/s11416-022-00432-3
20. A. Taha and O. Barukab, ‘‘Android malware classification using optimized ensemble learning based on genetic algorithms,’’ Sustainability, vol. 14, no. 21, p. 14406, Nov. 2022. https://doi.org/10.3390/su142114406
21. K. Sabanci, M. F. Aslan, E. Ropelewska, and M. F. Unlersen, ‘‘A convolutional neural network-based comparative study for pepper seed classification: Analysis of selected deep features with support vector machine,’’ J. Food Process Eng., vol. 45, no. 6, Jun. 2022, Art. no. e13955. DOI: 10.1111/jfpe.13955
22. A. Batouche and H. Jahankhani, ‘‘A comprehensive approach to Android malware detection using machine learning,’’ in Information Security Technologies for Controlling Pandemics. USA: Springer, 2021, pp. 171–212. DOI: 10.1007/978-3-030-72120-6_7
23. O. N. Elayan and A. M. Mustafa, ‘‘Android malware detection using deep learning,’’ Proc. Computer. Sci., vol. 184, pp. 847–852, Jan. 2021. https://doi.org/10.1016/j.procs.2021.03.106
24. S. S. Sammen, M. Ehteram, Z. Sheikh Khozani, and L. M. Sidek, ‘‘Binary coati optimization algorithm- multi- kernel least square support vector machine- extreme learning machine model (BCOAMKLSSVM- ELM): A new hybrid machine learning model for predicting reservoir water level,’’ Water, vol. 15, no. 8, p. 1593, Apr. 2023. https://doi.org/10.3390/w15081593