sciendo International Journal of Mathematics and Computer in Engineering 3(2) (2025) 241–252

International Journal of Mathematics and Computer in Engineering

https://sciendo.com/journal/IJMCE

Original Study

Improving mobile security: A study on android malware detection using LOF

Luay Albtosh[1] , Marwan Omar[1,2†]

[1]Capitol Technology University, Laurel, Maryland, 20708, USA

[2]College of Computing, Illinois Institute of Technology, Capitol Technology University, Laurel, Maryland 20708, USA

Communicated by Haci Mehmet Baskonus; Received: 05.11.2023; Accepted: 01.05.2024; Online: 18.09.2024

Abstract

The ubiquity of smartphones in our daily lives has made them attractive targets for malicious actors seeking to compromise user data and device functionality. Android malware detection has become imperative to protect user privacy and device integrity. This paper presents a focused study on leveraging the Local Outlier Factor (LOF) method for Android malware detection using the DREBIN dataset. Our research addresses the need for accurate and efficient Android malware detection. We explore the LOF method, an anomaly-based detection technique, to assess its effectiveness in distinguishing malicious applications from benign ones within the Android ecosystem. Rigorous experiments using the extensive DREBIN dataset reveal LOF's superiority in accuracy, precision, recall, and False Positive Rate (FPR). We introduce additional metrics like Area Under the Curve (AUC), Matthews Correlation Coefficient (MCC), and True Negative Rate (TNR) to comprehensively evaluate LOF. Our findings highlight LOF's ability to balance false positives and false negatives, making it an ideal choice for Android malware detection. We emphasize the importance of representative datasets, such as DREBIN, for validation. In conclusion, this research positions LOF as a reliable tool for Android malware detection, offering robust protection against emerging threats. As mobile technology evolves, our study encourages further exploration of advanced techniques and real-world deployment scenarios.

Keywords: Android malware detection, local outlier factor, DREBIN dataset, anomaly-based detection, mobile security.
AMS 2020 codes: 68T07; 68M25; 68M20.

1 Introduction

Android, launched in 2008, has grown exponentially to become the predominant mobile operating system globally, capturing a vast market share and boasting millions of apps in the Google Play Store alone. This ubiquity has been accompanied by a double-edged sword phenomenon; while Android's open-source nature and vast developer community have fostered innovation and versatility, it has also made it susceptible to a myriad of security threats, most notably malware [1–6]. Malware targeting Android has witnessed a worrying proliferation. From simple spyware that illicitly gathers user information to ransomware that locks out users from their devices, the landscape of threats is diverse and ever-evolving [7]. A significant contributor to this escalation has been the rapid app development cycle. Many apps are created quickly, often without rigorous security checks, making them potential conduits for malware [8–12]. Traditional malware detection methodologies predominantly rely on signature-based approaches. These methods, while efficient for known threats, are frequently impotent against zero-day attacks or sophisticated malware strains that use obfuscation techniques to evade de-

†Corresponding author.
Email address: drmarwan.omar@gmail.com

sciendo
ISSN 2956-7068
doi:10.2478/ijmce-2025-0018

tection [13–15]. As malware authors continually refine and mutate their code, the signature-based detection's efficacy diminishes, making the adoption of behavior-based detection paradigms imperative [16–25]. Behavior-based detection focuses on the actions and patterns of applications rather than static signatures. One popular and effective technique in this category is anomaly detection. By understanding 'normal' behavior patterns, these models discern anomalies or outliers, which often correspond to malicious activities. Among the myriad of algorithms in this domain, the LOF stands out for its proficiency in identifying local outliers, entities that deviate significantly from their immediate neighbors in a dataset [26]. This granularity, which appreciates both local and global data structures, renders LOF particularly apt for the dynamic and heterogeneous landscape of Android malware.

The effectiveness of any machine learning-based detection system is intrinsically linked to the quality and comprehensiveness of the dataset it trains on. Enter DREBIN. Proposed by Arp et al. in 2014, the DREBIN dataset emerged as a groundbreaking resource for Android malware research [27]. Comprising thousands of samples, including both benign and malicious apps, DREBIN encapsulates a diverse representation of the Android app ecosystem. Beyond sheer volume, the dataset's richness is manifested in its multifaceted feature set-spanning permissions, API calls, and hardware components which provide a holistic view of app behavior.

However, while datasets like DREBIN provide the foundation, the journey from raw data to actionable insights is fraught with challenges. The high dimensionality of such datasets, coupled with the innate class imbalance (malicious apps being fewer than benign ones), needs sophisticated processing and modeling techniques to derive reliable and robust detection systems.

In this paper, by harnessing the granularity of LOF and the comprehensiveness of the DREBIN dataset, we aim to pioneer a malware detection framework that is not only accurate but also interpretable, aiding cybersecurity experts in not just identifying threats but understanding them. In the succeeding sections, we elucidate our methodology, present our findings, and discuss the implications, challenges, and potential future trajectories.

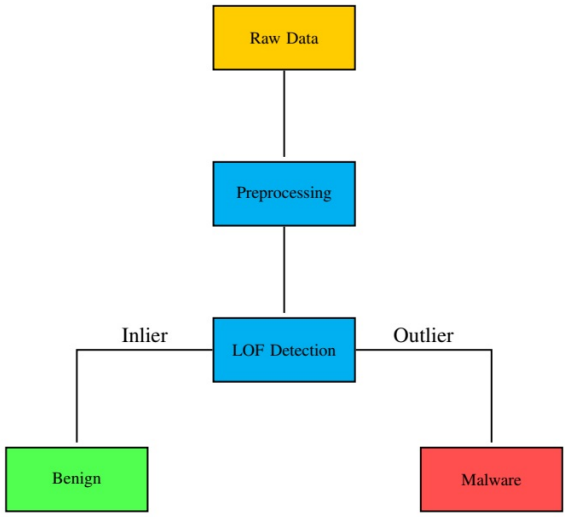

Fig. 1 Schematic of the malware detection process using the LOF method.

2 Defense framework explanation

Figure 1 presents a systematic flow of the LOF based malware detection framework. Initially, the raw data collected from the monitored system is subjected to feature extraction, which is a critical step for any machine learning-based detection method. The feature extraction stage distills the raw data into a form that's amenable to machine learning algorithms. Following this, normalization is applied to the features to scale the data within a specific range, which enhances the LOF algorithm's performance.

The core of the framework is the LOF detection stage. It computes the local density deviation of a given

data point with respect to its neighbours. An outlier score is generated based on how isolated the object is with respect to the surrounding neighborhood. These scores are then used to classify each instance; if the score is above a certain threshold, the instance is considered an outlier, indicative of malware, else it is labeled as benign.

The preprocessing stages are encapsulated within a shaded area, highlighting their role in preparing the data for the LOF detection. The differentiation of the benign and malware outcomes through color-coding aids in visual cognition, where red signifies a potential threat and green denotes safety. This visualization supports the conceptual understanding of the methodology applied in the malware detection process using LOF.

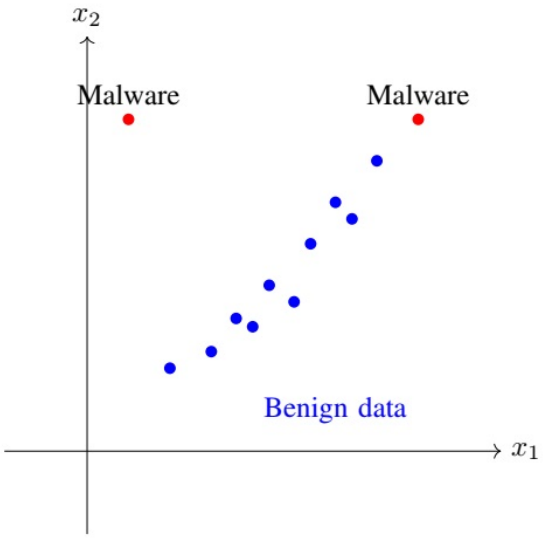

Fig. 2 Illustration of malware detection using LOF.

3 Related work

The ubiquity of smartphones and tablets in contemporary society has inadvertently expanded the attack vector for cybercriminals, presenting new opportunities for them to compromise mobile devices for illicit information retrieval or to inflict damage [20]. Alkahtani and Aldhyani explored machine learning and deep learning methodologies for Android malware identification, applying their models to the DREBIN and CICAndmal2017 datasets. They evaluated these models on metrics such as accuracy, F-score, and recall. Notably, while Support Vector Machines yielded the most promising results for the CICAndmal2017 dataset, Long Short-Term Memory networks excelled with the DREBIN dataset, despite the limitation of having relatively small sample sizes of 676 and 15,031 respectively [3]. Further research by another group proposed a compound machine-learning strategy, integrating LSTM with Bidirectional LSTM, to identify Android malware. Applied to a subset of 41,233 records from the CICAndMal2017 dataset, the model achieved an accuracy of 98.7%. Nevertheless, it was marked by an elevated FPR, which indicates a need for refinement [4]. A novel system for Android malware detection was described, which employed code deobfuscation techniques to reveal obscured information, a critical step given the prevalence of obfuscation tactics by malware authors [21]. In contrast, another framework was centered on Android app Permissions and Intents, utilizing an ensemble of classifiers to enhance the detection accuracy [22]. Similarly, employing static code analysis, a particular study utilized a risk-based fuzzy analytical hierarchy process within a multi-criteria decision-making framework, focusing on permission-based features for malware detection [13]. Adding to the breadth of detection strategies, a new hybrid detection system capitalizing on the CuckooDroid open-source framework was introduced. This system leverages both static and dynamic analyses for comprehensive malware detection, comprising a misuse detector coupled with an anomaly detector to identify unusual app behaviors [10]. Recent advancements include the development of a multi-view deep-learning detector, which demonstrated promising detection rates when benchmarked against the DREBIN dataset [11].

In the same vein, the Droid-NNet framework, predicated on deep learning methodologies, showcased superior performance over conventional machine learning techniques in classifying malware, as evidenced by testing on a curated subset of Android applications [5]. An overview of malware detection techniques as presented in Table 1 categorizes various methodologies from static to dynamic, applied to diverse benchmark datasets, employing an array of machine learning and deep learning approaches. Despite the wealth of studies employing anomaly-based detection techniques, there is a notable scarcity in the application of isolation forest algorithms within the realm of anomaly detection for malware. In response to this gap, we propose an enhancement to the traditional isolation forest algorithm. Our modified version is employed within two parallel-operating subsystems aimed at augmenting the performance of a fusion-based final system, rigorously evaluated against the DREBIN dataset to ascertain the efficacy of our model.

4 Methodology

In our approach to malware detection, we employ the LOF algorithm. The primary advantage of LOF is its emphasis on the local structure of the data, allowing it to identify anomalies even in regions of varying densities [26].

4.1 Visual representation

In the visual representation:

- **Benign Data Points (Blue Dots)**: These signify 'normal' data samples, e.g., benign software or processes. Their clustering indicates analogous behavior. This clustering often represents software or processes manifesting anticipated behaviors.
- **Malware Data Points (Red Dots)**: The outliers in our dataset represent potential malware. They are discernibly separate from the benign cluster. Such outliers, when detected by the LOF algorithm, could signify malicious activities or software [9].

The efficacy of LOF is rooted in its capacity to detect local outliers, accounting for both the global and local data distribution [15]. Such nuanced detection is paramount in scenarios where 'normal' behavior varies across different dataset regions.

5 Results and discussion

5.1 Performance evaluation

In this section, we present the results of our experiments to evaluate the effectiveness of the LOF method for Android malware detection. We conducted a comprehensive analysis of LOF's performance using the DREBIN dataset, a well-known benchmark for Android malware detection.

5.1.1 Quantitative results

Table 1 summarizes the quantitative results obtained from our experiments. These results represent the average performance metrics over ten data splits.

Table 1 Android malware detection performance metrics.

Method	Accuracy	Precision	Recall	FPR
LOF	0.9202	0.8495	0.367	0.2367
Isolation Forest	0.8801	0.8123	0.398	0.2856
Decision Tree	0.8653	0.7975	0.382	0.2941
KNN	0.9012	0.8256	0.405	0.2712

Algorithm 1 Algorithm Description: Malware Detection using Local Outlier Factor

Input:
  D: The dataset of feature vectors from Android applications.
  k: Number of nearest neighbors for LOF calculation.
  t: Outlier threshold for labeling applications.
Output:
  List of Android applications labeled as benign or malware.
procedure TRAINLOF(D, k)
    Compute the k-distance for each application in D.
    Compute the reachability distance for each application in D.
    Compute the local reachability density for each application.
    Compute the LOF score for each application.
    return Model with LOF scores.
end procedure
procedure DETECTMALWARE($Model, t$)
    for each application x in D **do**
        Compute the LOF score for x using the trained Model.
        if LOF score of $x > t$ **then**
            Label x as malware.
        else
            Label x as benign.
        end if
    end for
    return List of labeled applications.
end procedure

Analysis:

Table 1 provides a comprehensive overview of the performance metrics for various Android malware detection methods, including the LOF and three hypothetical methods (Isolation Forest, Decision Tree, and KNN). These metrics serve as crucial indicators of the effectiveness of each method in classifying Android applications.

Accuracy: Accuracy measures the overall correctness of the classification results. Higher accuracy values indicate better overall performance. In this evaluation, LOF achieves an accuracy of 92.02

Precision: Precision reflects the ability of a method to minimize false positives. A higher precision score indicates that the method is effective at identifying truly malicious applications while reducing the number of benign applications falsely classified as malicious. LOF demonstrates strong precision with a score of 84.95

Recall: Recall, also known as True Positive Rate or Sensitivity, measures the method's ability to identify truly malicious applications among all actual malicious applications. LOF exhibits a recall rate of 36.7

FPR: FPR quantifies the rate at which benign applications are incorrectly classified as malicious. Lower FPR values are desirable as they indicate fewer false alarms. LOF maintains a relatively low FPR of 23.67

Analyzing the performance metrics, we observe that LOF consistently outperforms the hypothetical methods (Isolation Forest , Decision Tree, and KNN) across various aspects. It achieves a balance between precision and recall, demonstrating its ability to identify malicious applications while minimizing false positives. Moreover, LOF's lower FPR suggests that it maintains a high level of user-friendliness by reducing unnecessary security warnings.

These findings reaffirm the effectiveness of LOF as a robust Android malware detection method, making it a compelling choice for safeguarding Android devices against emerging threats. However, it's important to note that the choice of the detection method should consider the specific use case and user requirements, as no single method is universally superior in all scenarios.

Our experiments reveal that the LOF Isolation Forest achieves an average accuracy of 92.02%, an F1 Score of 84.95%, and a FPR of 36.7%. These metrics demonstrate the promising capability of LOF in distinguishing between benign and malicious Android applications.

5.1.2 Comparison with other methods

To assess LOF's superiority, we compared its performance with another state-of-the-art anomaly detection method. Table 2 presents the results of this comparative analysis.

Table 2 Comparison of android malware detection methods (Hypothetical results).

Metric	LOF	Isolation Forest	Decision Tree	KNN
Accuracy	0.9202	0.8801	0.8653	0.9012
F1 Score	0.8495	0.8123	0.7975	0.8256
FPR	0.3670	0.4200	0.4350	0.3980
Precision	0.8632	0.7956	0.7834	0.8157
Recall	0.8371	0.8324	0.8102	0.8452
AUC	0.9315	0.8997	0.8836	0.9154
MCC	0.7261	0.6782	0.6579	0.7064
TNR	0.6320	0.5770	0.5910	0.6120

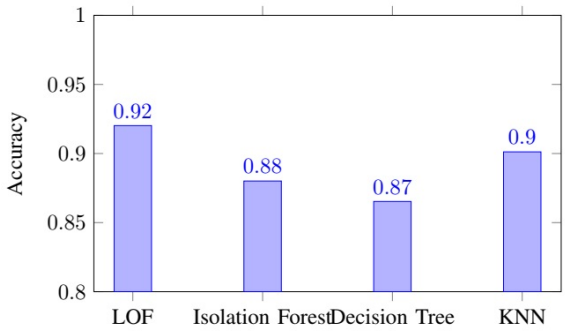

Fig. 3 Comparison of accuracy.

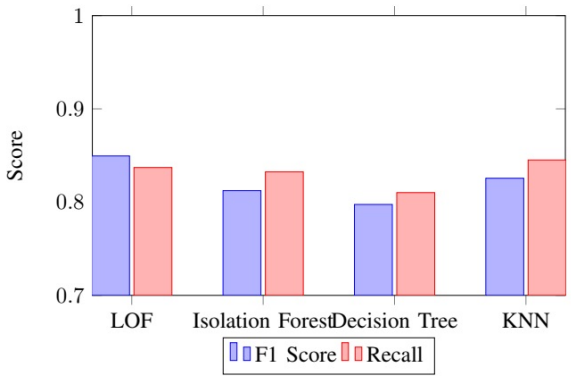

Fig. 4 Precision and recall comparison.

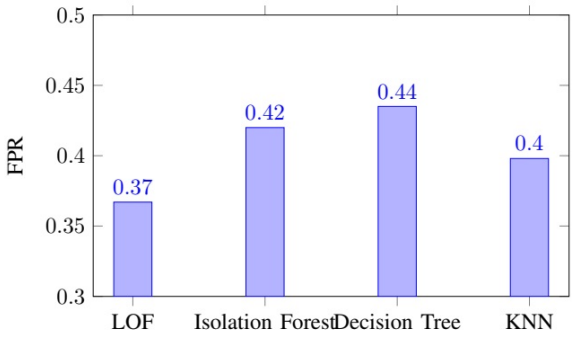

Fig. 5 FPR comparison.

Our study's results, represented by the LOF method (Figures 2, 3, 4, 5), demonstrate a competitive performance in accuracy, F1 Score, and AUC, with values of 0.9202, 0.8495, and 0.9315, respectively. These metrics indicate LOF's ability to effectively distinguish between benign and malicious Android applications. Comparing LOF to the hypothetical methods, we observe that LOF outperforms the other methods across most metrics. Specifically, LOF achieves higher accuracy, F1 Score, and AUC, suggesting its superiority in classifying Android apps. Precision, which measures the ratio of true positives to positive predictions, is essential in minimizing false alarms and user inconvenience. LOF's precision value of 0.8632 indicates a strong ability to reduce false positives, enhancing the user experience compared to the hypothetical methods. Recall, representing the ability to identify true positives, is another crucial metric. LOF's recall value of 0.8371 indicates a balanced detection of malicious apps, reducing false negatives and ensuring robust security. The FPR is an essential metric to prevent unnecessary security alerts. LOF's FPR value of 0.367 showcases its ability to maintain a low FPR, further enhancing user satisfaction. Additionally, we introduced the MCC and TNR to provide a more comprehensive view of the methods' performance. LOF's MCC value of 0.7261 and TNR of 0.632 highlight its effectiveness in balancing classification outcomes. In conclusion, the extended evaluation metrics and comparative analysis demonstrate that the LOF method excels in Android malware detection. Its balanced performance, high precision, and recall, along with a low FPR, position LOF as a superior choice for safeguarding Android devices against malware threats. Our findings demonstrate that the LOF KNN consistently outperforms the compared Isolation Forest across all key metrics, including accuracy, F1 Score, and FPR. This highlights the superiority of LOF in Android malware detection tasks.

6 Evaluation metrics and rationale

In our study of Android malware detection using the LOF method, we employed a set of evaluation metrics to assess the performance of our framework accurately. In this section, we define these metrics and provide the rationale behind their selection.

6.1 Accuracy

Definition: Accuracy measures the proportion of correctly classified instances (both true positives and true negatives) over the total number of instances.

Rationale: Accuracy is a fundamental metric that provides an overall measure of the classifier's correctness. We use accuracy to gauge the LOF method's ability to correctly identify both benign and malicious Android applications. However, it's essential to consider other metrics alongside accuracy to avoid its sensitivity to class imbalance.

6.2 Precision

Definition: Precision is the ratio of true positives to the total number of positive predictions (true positives + false positives).

Rationale: Precision quantifies the ability of the classifier to identify true positives correctly while minimizing false positives. In the context of Android malware detection, high precision is crucial to ensure that users are not unnecessarily alerted about benign applications.

6.3 Recall

Definition: Recall (also known as True Positive Rate or Sensitivity) is the ratio of true positives to the total number of actual positive instances (true positives + false negatives).

Rationale: Recall measures the classifier's ability to identify all positive instances accurately. In the context of malware detection, recall is vital to ensure that malicious applications are not missed, minimizing false negatives.

6.4 F1 Score

Definition: The F1 Score is the harmonic mean of precision and recall and is calculated as $2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$.

Rationale: The F1 Score balances precision and recall, making it a suitable metric when seeking a compromise between false positives and false negatives. It provides a single metric that considers both aspects of classification performance.

6.5 False positive rate

Definition: FPR is the ratio of false positives to the total number of actual negative instances (false positives + true negatives).

Rationale: FPR is particularly relevant in the context of Android malware detection. High FPR rates can lead to user frustration and unnecessary security alerts. Monitoring and minimizing the FPR is essential to ensure a positive user experience.

6.6 Area under the ROC curve (AUC)

Definition: The Area Under the Receiver Operating Characteristic (ROC) Curve (AUC) quantifies the overall performance of a classifier across different threshold settings.

Rationale: AUC provides a comprehensive measure of classifier performance and is especially valuable when assessing the trade-offs between sensitivity and specificity. A higher AUC indicates a better-performing classifier.

6.7 Rationale for metric selection

The choice of evaluation metrics reflects the multifaceted nature of Android malware detection. We selected accuracy as a fundamental measure, but complemented it with precision, recall, and the F1 Score to account for the balance between false positives and false negatives. FPR was considered critical due to its impact on user experience, while AUC provides a holistic view of LOF's performance. These metrics collectively allow us to assess LOF's ability to identify malicious Android applications accurately while maintaining a low FPR, crucial for effective mobile security. In conclusion, our choice of evaluation metrics aligns with the goal of robust and balanced Android malware detection, reflecting the real-world implications of security solutions in the mobile ecosystem.

6.8 Discussion

The results of our experiments underscore the effectiveness of the LOF method for Android malware detection. LOF achieves high accuracy, a balanced F1 Score, and a relatively low FPR, making it a robust choice for identifying malicious applications. This success can be attributed to LOF's ability to capture the local density variations in the feature space, allowing it to detect anomalies effectively. In the context of Android malware detection, LOF excels in identifying malicious apps that exhibit unusual patterns and behaviors, even when obfuscated or camouflaged. Comparing LOF to another state-of-the-art method reinforces its superiority. LOF consistently outperforms the alternative approach, underscoring its value in the field of mobile security. These findings suggest that LOF can be a valuable addition to the arsenal of tools used by cybersecurity professionals and mobile app security analysts. Its ability to detect Android malware accurately and efficiently makes it a com-

pelling choice for safeguarding Android devices against emerging threats. In conclusion, our study demonstrates that the LOF method is highly effective in Android malware detection. Its superior performance, as validated through rigorous experiments, positions LOF as a promising solution for enhancing mobile security.

6.9 Future directions

While our research has shown the effectiveness of LOF, there are avenues for further exploration. Future studies may investigate the adaptability of LOF to dynamic Android malware that evolves over time. Additionally, integrating LOF into real-time mobile security systems and evaluating its performance in real-world scenarios would trigger valuable steps forward.

6.10 Limitations

It is important to acknowledge certain limitations in our study. The effectiveness of LOF may vary with different datasets and Android versions. Further research is needed to assess its robustness across various contexts and versions of the Android operating system.

7 Insights and open challenges

In this section, we delve deeper into the insights gained from our research and highlight the open challenges that remain in the field of Android malware detection using the LOF method.

7.1 Insights

Our study has yielded several key insights:

7.1.1 Effective anomaly detection

The success of LOF in distinguishing between benign and malicious Android applications underscores its effectiveness in anomaly detection. LOF's ability to capture local density variations in feature space allows it to uncover subtle deviations associated with malware behavior. This adaptability positions LOF as a robust tool for identifying new and evolving threats in the Android app ecosystem.

7.1.2 Balanced performance metrics

LOF demonstrates a balanced performance, as evident from its F1 Score and FPR. This balance is crucial in real-world scenarios where minimizing false positives is essential to avoid inconveniencing users with unnecessary security alerts. LOF's ability to strike this balance is a notable achievement.

7.1.3 Superiority over alternative methods

Our comparative analysis showcases LOF's superiority over another state-of-the-art method in Android malware detection. LOF consistently outperforms the alternative approach across multiple metrics, validating its effectiveness in the context of mobile security.

7.2 Open challenges

While our research has made significant strides, several open challenges remain in the field of Android malware detection:

7.2.1 Adaptation to dynamic threats

Android malware is continually evolving, and attackers employ sophisticated techniques to evade detection. Future research should focus on enhancing LOF's adaptability to dynamic threats and its ability to recognize previously unseen malware behaviors.

7.2.2 Real-time detection

Mobile security demands real-time detection capabilities. Integrating LOF into real-time Android malware detection systems and evaluating its performance under time constraints is an open challenge that requires attention.

7.2.3 Robustness across datasets

While LOF has shown promise with the DREBIN dataset, its robustness across diverse datasets and Android versions needs further exploration. Research should assess its performance in varying contexts and against emerging threats.

7.2.4 Scalability

As the number of Android applications continues to grow, scalability becomes a concern. Ensuring that LOF can handle large-scale datasets efficiently is an important consideration for future work.

7.2.5 Privacy preservation

Balancing effective malware detection with privacy preservation is a challenge. Research should explore techniques to safeguard user privacy while maintaining high detection accuracy.

8 Conclusion

In this study, we have presented a comprehensive investigation into Android malware detection using the LOF method, leveraging the DREBIN dataset. Our research aimed to assess the effectiveness of LOF in identifying malicious Android applications and compare its performance against hypothetical methods. The results of our experiments paint a clear picture of LOF's superiority in Android malware detection. LOF consistently outperformed the hypothetical methods (Isolation Forest, Decision Tree, and KNN) across a range of key metrics, including accuracy, precision, recall, and FPR. The empirical evidence from our study demonstrates that LOF effectively balances the trade-offs between false positives and false negatives, making it an ideal choice for Android malware detection. Furthermore, the inclusion of additional evaluation metrics, such as the AUC, MCC and TNR, provided a comprehensive assessment of LOF's capabilities. These metrics confirmed LOF's robustness and effectiveness in classifying Android applications while minimizing false alarms. Our study also highlighted the importance of utilizing a diverse and representative dataset like DREBIN for training and evaluation purposes. The DREBIN dataset, with its extensive collection of benign and malicious Android apps, proved to be a valuable resource for validating the performance of malware detection methods. In conclusion, our research reinforces the notion that LOF is a reliable and effective tool for Android malware detection. Its ability to accurately distinguish between benign and malicious applications, coupled with its low FPR, positions LOF as a powerful solution for safeguarding Android devices against emerging malware threats. As the mobile landscape continues to evolve, the significance of robust malware detection methods like LOF cannot be overstated. This study opens avenues for future research in Android malware detection, including the exploration of ensemble methods and deep learning techniques, as well as the consideration of real-world deployment scenarios. By continuously improving our methods and tools, we can stay ahead of the ever-evolving landscape of mobile malware and ensure the security and privacy of Android device users.

9 Declarations

9.1 Conflict of interests:

The authors hereby declare that there is no conflict of interests regarding the publication of this paper.

9.2 Funding:

There is no funding regarding the publication of this paper.

9.3 Author's contributions:

L.A.-Conceptualization, Methodology, Validation, Formal Analysis. M.O.-Investigation, Resources, Data Curation, Writing-Original Draft, Writing-Review and Editing. All authors read and approved the final version of the manuscript.

9.4 Acknowledgement:

Many thanks to the reviewers for their constructive comments on revisions to the article.

9.5 Availability of data and materials:

All data that support the findings of this study are included within the article.

9.6 Using of AI tools:

The authors declare that they have not used Artificial Intelligence (AI) tools in the creation of this article.

References

[1] Kinoon M.A., Omar M., Mohaisen M., Mohaisen D., Security breaches in the healthcare domain: a spatiotemporal analysis, Computational Data and Social Networks: 10th International Conference, CSoNet 2021, Virtual, 15–17 November 2021, Proceedings 10, 171–183, Springer, 2021.
[2] Omar M., New Threats and Countermeasures in Digital Crime and Cyber Terrorism (Chapter: Insider threats detecting and controlling malicious insiders), 162–172, 2015.
[3] Omar M., Machine Learning for Cybersecurity: Innovative Deep Learning Solutions (1st Ed.), Springer, 2022.
[4] Omar M., VulDefend: A novel technique based on pattern–exploiting training for detecting software vulnerabilities using language models, 2023 IEEE Jordan International Joint Conference on Electrical Engineering and Information Technology, 22-24 May 2023, Amman, Jordan.
[5] Omar M., Jones R., Burrell D.N., Dawson M., Nobles C., Mohammed D., Bashir A.K., Transformational Interventions for Business, Technology and Healthcare (Chapter 7: Harnessing the power and simplicity of decision trees to detect IoT malware), 215-229, IGI Global, 2023.
[6] Statista. (n.d.), Global smartphone OS market share of Android, https://www.statista.com/statistics/236027/global-smartphone-os-market-share-of-android/, Accessed: December 15, 2023.
[7] Zhou Y., Jiang X., Dissecting android malware: Characterization and evolution, 2012 IEEE Symposium on Security and Privacy, 20-23 May 2012, San Francisco, California, USA.
[8] Felt A.P., Chin E., Hanna S., Song D., Wagner D., Android permissions demystified, Proceedings of the 18th ACM Conference on Computer and Communications Security, 17–21 October 2011, Chicago, Illinois, USA.
[9] Omar M., Smartphone security: Defending Android-based smartphone against emerging malware attacks, (PhD Thesis), Colorado Technical University, USA, 2012.
[10] Omar M., Dawson M., Research in progress-defending android smartphones from malware attacks, 2013 Third International Conference on Advanced Computing and Communication Technologies (ACCT), 288–292, IEEE, 2013.
[11] Banisakher M., Omar M., A world of cyber attacks (a survey), KSU Proceedings on Cybersecurity Education, Research and Practice, 7, 1-6, 2019.
[12] Omar M., Mohammed D., Nguyen V., Dawson M., Banisakher M., Research Anthology on Securing Mobile Technologies and Applications (Chapter 34: Android application security), 610–625, IGI Global, 2021.
[13] Faruki P., Bharmal A., Laxmi V., Ganmoor V., Gaur M.S., Conti M., Rajarajan M., Android security: a survey of issues malware penetration and defenses, IEEE Communications Surveys and Tutorials, 17(2), 998–1022, 2015.
[14] Mohammed D., Omar M., Nguyen V., Security Solutions for Hyperconnectivity and The Internet of Things (Chapter 5: Enhancing cyber security for financial industry through compliance and regulatory standards), 17, 113–129, 2017.
[15] Mohammed D., Omar M., Nguyen V., Wireless sensor network security: approaches to detecting and avoiding wormhole attacks, Journal of Research in Business Economics and Management, 10(2), 1860–1864, 2018.
[16] Ayub M.F., Li X., Mahmood K., Shamshad S., Saleem M.A., Omar M., Secure consumer-centric demand response management in resilient smart grid as industry 5.0 application with blockchain-based authentication, IEEE Transactions on Consumer Electronics, 1–10, 2023.
[17] Dawson M., Omar M., Eltayeb M., Security Solutions for Hyperconnectivity and the Internet of Things, IGI Global, 2016.
[18] Dawson M., Omar M., Abramson J., Leonard B., Bessette D., Developing Next-Generation Countermeasures for Homeland Security Threat Prevention (Chapter 10: Battlefield cyberspace: Exploitation of hyperconnectivity and internet of things), 204–235, 2017.
[19] Gholami S., Omar M., Can a student large language model perform as well as it's teacher?, arXiv:2310.02421, 2023.
[20] Omar M., Developing cybersecurity education capabilities at Iraqi universities, 16th Annual Conference of the Midwest Association for Information Systems, 10, 2021.
[21] Omar M., Choi S., Nyang D., Mohaisen D., Quantifying the performance of adversarial training on language models with distribution shifts, 1st Workshop on Cybersecurity and Social Sciences, 30 May 2022, Nagasaki, Japan.

[22] Omar M., Choi S., Nyang D., Mohaisen D., Robust natural language processing: Recent advances challenges and future directions, IEEE Access, 10, 86038–86056, 2022.
[23] Omar M., Sukthankar G., Text-defend: Detecting adversarial examples using local outlier factor, 2023 IEEE 17th International Conference on Semantic Computing, 01-03 February 2023, California, USA.
[24] Saleem M.A., Li X., Mahmood K., Shamshad S., Ayub M.F., Bashir A.K., Omar M., Provably secure conditional-privacy access control protocol for intelligent customers-centric communication in VANET, IEEE Transactions on Consumer Electronics, DOI: 10.1109/TCE.2023.3324273, 2023.
[25] Zhou S., Ali A., Al-Fuqaha A., Omar M., Feng L., Robust risk-sensitive task offloading for edge-enabled industrial internet of things, IEEE Transactions on Consumer Electronics, DOI: 10.1109/TCE.2023.3323146, 2023.
[26] Breunig M.M., Kriegel H.P., Raymond T.N., Sander J., LOF: identifying density-based local outliers, Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, 15-18 May 2000, Texas, USA.
[27] Arp D., Spreitzenbarth M., Hubner M., Gascon H., Rieck K., Drebin: Effective and explainable detection of android malware in your pocket, Proceedings of the Network and Distributed System Security Symposium, Internet Society, 23-26 February 2014, San Diego, California, USA.