AIS ARTIFICIAL INTELLIGENCE STUDIES

Deep Learning Approaches on Image Representations of Android Malware: A Review

Oğuzhan Sezer*[a], İbrahim Alper Doğru [b], Kazım Kılıç[c]

ABSTRACT

This review emphasizes the transition from conventional static analysis to graph-driven, vision-based, and transformer-focused methodologies in deep learning-based and hybrid multimodal Android malware detection from 2023 to 2025. This review utilizes various pertinent studies from Elsevier, IEEE Xplore, and ScienceDirect to analyze ten representative methodologies selected for their methodological diversity and their contributions to visual, hybrid, and interpretable detection frameworks. End-to-end image models that turn DEX bytecode into grayscale or RGB matrices, graph-attention and multimodal fusion techniques that combine structural and semantic features, and deeper architectures like 3D-CNNs and Vision Transformers that can find multiscale contextual patterns are all examples of these kinds of methods. The works that were looked at also include a number of explainable AI methods that use SHAP or Grad-CAM, as well as lightweight learning frameworks that try to make models less complicated. There is a clear change from manually made static features to automatically learned image-based representations. For example, graph-based models can be up to 99.5% accurate, while CNN models like MADRF-CNN can be between 96% and 98% accurate. The remaining issues are adversarial robustness, computational cost, and dataset imbalance. Lightweight CNN-Transformer hybrid detectors, the development of balanced benchmarks with adversarial and obfuscated samples, and a more thorough integration of explainability within multimodal learning for real-time usability and robustness enhancement are the main focuses of emerging research directions. The idea that the next generation of scalable and robust Android malware defense systems has been propelled by the convergence of visual computing, graph reasoning, and explainable AI is supported by these trends taken together.

[a,*] Department of Computer Engineering, Graduate School of Natural and Applied Sciences, Gazi University, Ankara 06560, Türkiye
ORCID: 0009-0005-4973-670X

[b] Department of Computer Engineering, Faculty of Technology, Gazi University, Teknikokullar, Ankara 06560, Turkey
ORCID: 0000-0001-9324-7157

[c] Department of Computer Programming, Vocational School of Yozgat, Yozgat Bozok University, Yozgat 66100, Turkey
ORCID: 0000-0003-2168-1338

* Corresponding author.
e-mail: oguzhansezer@gazi.edu.tr

Keywords: Android malware, image-based detection, deep learning, mobile security.

Anahtar Kelimeler: Android kötücül yazılımları, görüntü tabanlı tespit, derin öğrenme, mobil güvenlik.

Submitted: 15.12.2025
Revised: 30.12.2025
Accepted: 31.12.2025

doi:10.30855/ais.2025.08.02.07

Android Zararlı Yazılımlarının Görsel Temsillerine Dayalı Derin Öğrenme Yaklaşımları: Bir İnceleme

ÖZ

Bu inceleme, 2023-2025 yılları arasında derin öğrenme tabanlı ve hibrit çok modlu Android kötü amaçlı yazılım tespitinde geleneksel statik analizden grafik tabanlı, görsel tabanlı ve transformatör odaklı yaklaşımlara geçişi vurgulamaktadır. Bu inceleme, metodolojik çeşitlilikleri ve görsel, hibrit ve yorumlanabilir tespit çerçevelerine katkıları nedeniyle seçilen on temsili yaklaşımı incelemek için Elsevier, IEEE Xplore ve ScienceDirect'ten çeşitli ilgili çalışmaları kullanmaktadır. DEX bayt kodunu gri tonlamalı veya RGB matrislere dönüştüren uçtan uca görüntü modelleri, yapısal ve anlamsal özellikleri birleştiren grafik dikkat ve çok modlu füzyon teknikleri ve çok ölçekli bağlamsal kalıpları yakalayabilen 3D-CNN'ler ve Vision Transformer'lar gibi daha derin mimariler temsili yaklaşımlardır. İncelenen çalışmalar ayrıca SHAP veya Grad-CAM kullanan bir dizi açıklanabilir yapay zeka tekniğinin yanı sıra model karmaşıklığını azaltmayı amaçlayan hafif öğrenme çerçevelerini de içermektedir. Tüm bunlar göz önüne alındığında, elle oluşturulan statik özelliklerden otomatik olarak öğrenilen görüntü merkezli temsillerine doğru belirgin bir geçiş vardır. Örneğin, grafik tabanlı modeller %99,5'e varan doğruluk oranlarına ulaşırken, MADRF-CNN gibi CNN modelleri %96 ile %98 arasında doğruluk elde etmektedir. Geriye kalan sorunlar ise düşman saldırılarına karşı dayanıklılık, hesaplama maliyeti ve veri kümesi dengesizliğidir. Hafif CNN-Transformer hibrit dedektörleri, düşman saldırıları ve gizlenmiş örneklerle dengeli kıyaslama testlerinin geliştirilmesi ve gerçek zamanlı kullanılabilirlik ve dayanıklılık artırımı için çok modlu öğrenme içinde açıklanabilirliğin daha kapsamlı entegrasyonu, ortaya çıkan araştırma yönlerinin ana odak noktalarıdır. Ölçeklenebilir ve sağlam Android kötü amaçlı yazılım savunma sistemlerinin yeni neslinin, görsel hesaplama, grafiksel akıl yürütme ve açıklanabilir yapay zekanın birleşimiyle desteklendiği fikri, bu eğilimlerin birlikte ele alınmasıyla desteklenmektedir.

1. Introduction

Mobile technology pervades modern communication, business, and entertainment. Android dominates the smartphone ecosystem with 72.15% of the global market share and serves over 3 billion active users globally as of June 2024 [1]. These figures demonstrate how the quick expansion has attracted malevolent actors. According to a 2024 Kaspersky report, there were roughly 33.8 million mobile device attacks recorded worldwide in 2024, representing a 50% increase compared to 2023. Additionally, during the first quarter of 2024, Kaspersky identified approximately 390,000 distinct malware variants and blocked more than 10.1 million malware attacks [2].

Open source encourages innovation in Android, but it has also made the platform vulnerable to a number of security flaws, privacy violations, and financial scams. Obfuscation and polymorphic transformations are frequently employed by malware developers, rendering conventional signature-based and rule-driven defenses outdated. More than half of contemporary malware variants use one or more obfuscation techniques to avoid detection mechanisms, according to recent research [1].

In order to counter these dynamic threats, scientists have been ever more willing to utilize machine learning (ML) and deep learning (DL) methods. Initial models like Drebin relied on static features gleaned from APK files. Even though these treatments were extremely specific, they again faced attack by adversaries. More contemporary works have favored the use of images as Android program representations, converting bytecode or DEX files into grayscale or RGB images suitable for inspection directly by convolutional neural networks (CNNs). This model inscribes structural and spatial properties within the code, providing increased robustness towards obfuscation and redundant constructs [3].

For example, developed a 3D convolutional model (3DMalDroid) with 99.4% accuracy and 0.993 F1-score, surpassing earlier 2D CNN-based models [4]. Also, developed an end-to-end MADRF-CNN-based framework that had 96.9% detection accuracy, greatly outperforming handcrafted-feature models [3].

Recent research conducted from 2024 to 2025 indicates that Android malware detection is progressing towards graph-based, multimodal, and interpretable deep learning frameworks. Using structural representations like class-call graphs or function-call graphs to model Android apps in a graph-oriented way lets deep learning models find complicated interprocedural relationships that aren't easy to see in pictures alone [5]. At the same time, deep learning-based detectors have been used with explainable artificial intelligence (XAI) techniques more and more to make them more reliable and clear, especially in places where security is important [6]. Multimodal learning techniques that use deep learning architectures to combine static and dynamic features have also shown to be more resistant to polymorphic malware and obfuscation [2]. These trends show that modern Android malware detection research is moving away from single visual models and toward hybrid, structural, and interpretable architectures.

In addition to algorithmic progress, the community has been mindful of dataset integrity and reproducibility. AndroZoo, CICMalDroid2020, and MalDroid repositories contain millions of malicious and deceitful APK samples collected over the past decade or more. However, imbalanced class distributions as well as duplicate samples remain major roadblocks to fair benchmarks [4].

On the basis of these developments, in the current work, we desire a comprehensive survey and synthesis of recent image-based Android malware detection paradigms from 2023 till 2025, encompassing architectural modes, consumption of dataset, policies of feature extraction, and explainability factors. Along with this, the work alludes to open issues such as the balance between the detection accuracy and computational efficiency. Indicates directions towards the development of stronger, comprehensible, and scalable Android security paradigms. To ensure a systematic and unbiased literature selection, this review followed a PRISMA-inspired screening process. An initial pool of studies was collected from major scientific databases including IEEE Xplore, ScienceDirect, and SpringerLink using predefined keywords related to Android malware detection and deep learning. After removing duplicates and applying inclusion criteria focused on image-based, graph-based, and hybrid approaches published between 2023 and 2025, a final set of representative studies was selected for detailed analysis.

2. Traditional and AI-Based Detection Paradigms

Android malware detection approaches began transitioning from the earlier traditional and non-traditional methods towards AI-based as well as image-based paradigms. Static analysis reviews the components of the application like the permissions, API calls, opcodes, as well as the manifest files, without running the application, providing quick execution but low resilience to code obfuscations along with polymorphism. Dynamic analysis, on the other hand, monitors system calls, memory use, and network traffic at runtime in a sandboxed environment, among other behaviors, which gives richer behavioral insight but suffers from high computational overhead and poor scalability. To overcome this shortcoming, AI-based methods were developed in which machine learning, and later deep learning techniques, automatically learned discriminative representations from raw data. Early machine learning models relied on handcrafted feature engineering, while recent deep neural architectures such as CNNs, LSTMs, and Transformers extract high-level semantic and structural patterns directly from code or visualized data, improving detection accuracy and generalization against evolving malware variants [3],[7],[8].

2.1. Android Malware Detection Workflow

Android malware detection is a general process comprising four key steps: data gathering, bytecode conversion, characteristic extraction, and categorization. Android package files (APKs) are gathered from open repositories like AndroZoo, Drebin, MalGenome, AMD, and Google Play in the initial stage. APKs are then unpacked in order to get DEX or the smali code that can be analyzed further [9]. Extracted bytecode is then converted into images or audio formats in order to facilitate learning-based feature learning via deep learning. Common transformations are opcode-based grayscale, API call encoding into two-dimensional matrices, hex dump transformation into RGB images, and creation of Mel-spectrogram or MFCC in order to get code structure that is like audio [10]. On the characteristic extraction stage, deep learning models like CNN, ResNet, and DenseNet are employed to extract discriminative semantic as well as spatial structures. Lastly, the extracted features are employed in the classification process, either separating malware as well as the normal software or assigning samples into particular malware sets.

2.2. Bytecode Transformation into Images

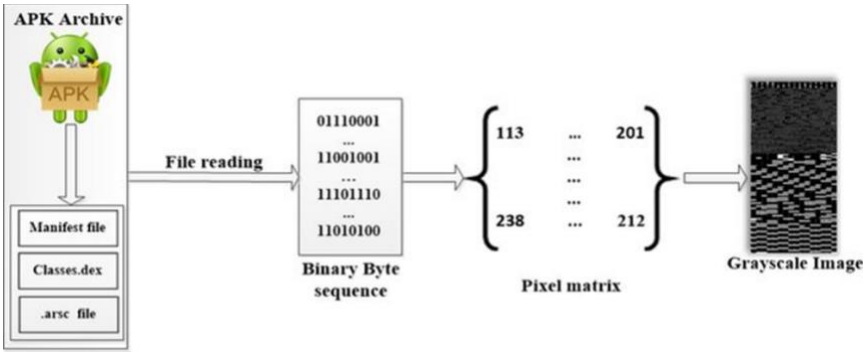

Figure 1. Conversion of Android APK files into grayscale images [9].

Figure 1 illustrates the process of transforming Android APK bytecode into grayscale image representations, which serve as input for deep learning–based malware detection models. In the conversion phase, the bytecode sequences or DEX that have been extracted are transformed into visual formats like grayscale or RGB images, allowing deep learning models to learn spatial relationships between opcode patterns [3]. Due to the byte values ranging from 0 through 255, each byte value is directly mapped into a pixel intensity, generating a two-dimensional matrix that illustrates the structural properties of the DEX file. In grayscale, each pixel is equivalent to one byte, whereas in RGB, sets of three bytes are packed into red, green, and blue channels, and as such, the feature space is further enhanced. Certain works have further converted byte-level information into Mel-frequency cepstral coefficients (MFCC) or spectrogram such that the audio-inspire presentations can be developed, further augmenting the generalization capability of the model [9].

2.3. AI-Based Malware Detection Workflow

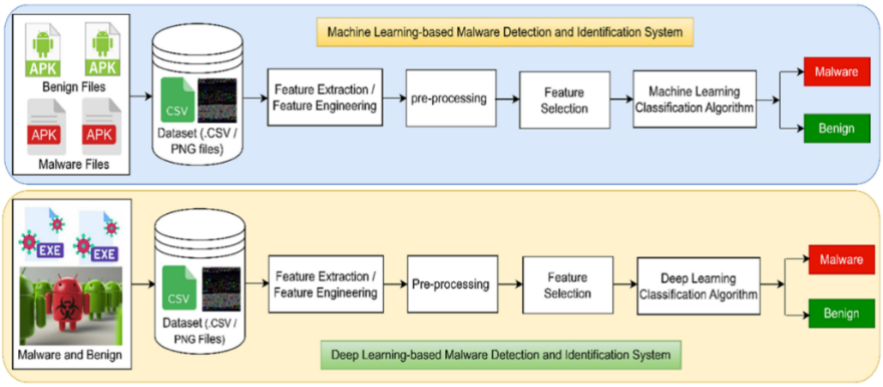

Figure 2. Comparison of machine learning-based and deep learning-based malware detection workflows [10].

Figure 2 provides a conceptual comparison between traditional machine learning pipelines and modern deep learning–based malware detection workflows, highlighting the shift toward automated feature learning. As illustrated in Figure 2, the artificial intelligence–based malware detection process integrates both machine learning (ML) and deep learning (DL) within a unified analytical pipeline consisting of four main stages: data preprocessing, feature extraction, feature selection, and classification. In preprocessing, the benign and malicious Android APKs are collected from public databases and transformed into structured formats such as CSV, bytecodes, or image representations, providing standardized input during model training. Feature extraction then deals with extracting significant features from the data. Whereas the ML methods rely on hand-engineered features such as permissions or API callings, the DL methods such as CNN or ViT learn high-level patterns directly from raw inputs in the absence of human intervention. In feature selection, redundancy is suppressed by applying methods such as Principal Component Analysis (PCA) or metaheuristic methods such as the Rüppell's Fox Optimizer (RFO) that enhance efficiency as well as generalization. Lastly, the classification phase separates the benign and malicious samples via algorithms such as SVM, Random Forest, or deep ones that offer increased robustness to obfuscations. In general, Figure 2 presents the shift from the traditional, rule-based approaches towards visionary, adaptive AI frameworks for detecting Android malwares [3],[4],[10].

3. Results and Discussion

A comparative analysis between traditional and image-based Android malware detection approaches presents distinct differences along multiple axes. Traditional static and dynamic methods rely significantly on hand-engineering features and domain knowledge like permissions, API calls, or opcode inspection so they become laboriously slow and less transferable. They further exhibit low accuracy and are weak towards obfuscations as well as polymorphic malwares. On the contrary, image-based methods autonomously learn discriminative features out of behavioral or code visual representations, retaining high detection accuracy and resilience. Although traditional dynamic methods can be costly computationally and slower in scale, image-based models that utilize CNN, ResNet, or DenseNet-based structures are faster and scalable towards large-scale datasets. In addition, their adaptive learning property facilitates them in recognizing new patterns of malwares more easily. In general, image-based approaches provide a fresh and viable outlook by converting codes into images and making use of computer vision principles, consequently decreasing computational overhead as well as maintenance cost over the long horizon. [1].

Table 1. Comparative Summary of Recent Deep Learning–Based Android Malware Detection Studies

References	Dataset	Summary of Models Used	Accuracy (%)
[1]	CICMalDroid2020, AndroZoo, new obfuscated dataset	12 fine-tuned CNNs (ResNet, VGG, DenseNet, Xception, Inception, Inception-ResNet-V2)	99.26
[3]	VirusShare (2,507) + Google Play (1,417)	MADRF-CNN	96.90
[4]	AndroZoo, CICMalDroid2017, CICMalDroid2020	B2IMG bytecode-image; 2D+ and 3D CNNs	99.40
[7]	CICMalDroid2017, CICMalDroid2020, AndroZoo	Dalvik bytecode -RGB & grayscale via B2IMG; DenseNet121, ResNet50, ResNeXt50	98.70

[11]	VirusShare, Apkpure, Drebin, AMD, AndroZoo	Class-Set Call Graph (CSCG) + GAT + Topic Model + Multimodal Fusion Network;	99.54
[12]	CICAndMal2017 (14,534 RGB images, 60 features)	8 models: CNN-only, ViT-only, and CNN+ViT hybrids; best = 2×CNN + 2×ViT	99.61
[13]	CICAndMal2017	DNN, 1D-CNN, BiLSTM	99.02
[14]	Drebin, CICMalDroid Datasets, AndroZoo, AMD, Genome	Permission-based images, mini-ResNet	98.34
[15]	Drebin, AMD	Ensemble of VGG16, ResNet50	98.65
[16]	CICInvesAndMal2019	Transfer learning with DenseNet169, Xception, InceptionV3, ResNet50, VGG16	95.83

Table 1 provides a comparative overview of recent works between 2023 and 2025 that depict the progress in the development of image and hybrid Android malware detection models. Initially proposed frameworks like MADRF-CNN [3] reached approximately 96.9% accuracy when DEX and manifests data were transformed into image matrices, whereas later works like Graph Attention Network (GAT) based multimodal fusion networks [11] and Convolutional Neural Network–Vision Transformer (CNN–ViT) hybrids [12] reached over 99% accuracy, which underscores the advantage that structural and visual feature integration provide. Contributions by Yapici [4],[7] and Najibi & Bidgoly [13] proposed the introduction of 3D-CNN and BiLSTM structures, which enhanced contextual learning and diminished false alarms on obfuscated specimens. In turn, Kılıç, Toklu, & Doğru [14] and [15] investigated the permission-based and ensemble learning framework, whose high interpretability was combined with generalizability across datasets ranging from Drebin, AndroZoo, and CICMalDroid. By including the incorporation of transfer learning in the DenseNet [16] as well as Inception-based frameworks [1] further enhanced the robustness of the model versus polymorphic malwares, as verified the fact that hybrid and deep visual learning frameworks always surpass the traditional static or behavioral approaches in terms of accuracy and scalability. It should be noted that direct numerical comparison across the reported studies is inherently limited, as the reviewed works employ different datasets, class distributions, preprocessing strategies, and evaluation protocols. Therefore, the reported performance metrics should be interpreted as indicative rather than strictly comparable across methods. Overall, Table 1 shows that, in terms of detection accuracy across a variety of datasets, hybrid and graph-based deep learning models consistently outperform standalone image-based approaches.

A closer look at the published results shows that both the representation strategy and the dataset characteristics have a significant impact on detection performance. Particularly on extensive and varied datasets like AndroZoo and CICMalDroid, models utilizing graph-based representations or multimodal fusion consistently attain higher accuracy levels, indicating that structural context is crucial for identifying malicious behaviors. On the other hand, even though they are computationally efficient, standalone image-based CNN models perform worse when tested on obfuscated or polymorphic samples. Furthermore, by capturing contextual dependencies beyond static bytecode patterns, architectures that incorporate temporal or sequential modeling—like 3D-CNNs and BiLSTM-based frameworks—show enhanced robustness. These findings suggest that effective integration of structural, temporal, and semantic information within the detection pipeline is responsible for performance gains rather than model depth alone. [1], [2], [4], [13]

Graph-based Android malware detection approaches differ primarily in how program structure is represented and analyzed. Control Flow Graphs (CFGs) model the execution flow within individual methods, capturing low-level control structures such as branches and loops, but often lack interprocedural context. Call Graphs represent method invocation relationships across the application, providing a higher-level view of functional interactions, while Function Call Graphs (FCGs) further refine this representation by focusing on function-level dependencies and call frequencies. Compared to CFGs, call graph–based and FCG-based models are more effective at capturing global behavioral patterns and interprocedural relationships, which explains their superior performance in recent graph attention and multimodal Android malware detection frameworks [5], [11].

It is seen that the results indicate a distinct orientation on the part of Android malware detection towards AI-based, image-based, and hybrid deep learning methods. As tabulated in Table 1, almost all the studies resulted in accuracy levels greater than 95%, breaching the superiority of convolutional and transformer-based structures in identifying structural and semantic patterns from bytecode

images. Those like uitObfAMC and CNN–ViT Synergy showed accuracy levels greater than 99%, highlighting the rationale behind feature fusion as well as hybrid design of the model. The incorporation of Explainable AI (XAI) approaches into frameworks like FABLDroid improved interpretability, allowing cybersecurity experts to understand the rationale behind the predictions. The results indicate that the integration of visual, graph-based, and semantic learning resulted in robust and scalable malware detection [1],[8],[12].

A lot of progress has been achieved, but there are still some flaws with the existing image-based approaches. For example, the sets that are available are still out of date and unbalanced, and they often don't have malware samples that are obfuscated or dynamically constructed. Furthermore, real-time execution on portable devices is not possible due to the computational complexity of transformer or ensemble CNN models. Although XAI-based techniques improved the models' interpretability, there is still little evidence of their incorporation into real-world security tools. To ensure reliable and reliable detection on dynamic Android malware, the focus should soon shift to lightweight, interpretable, and continuously learning structures supported by balanced and multimodal sets [13]. Additionally, while explainable AI methods like SHAP and Grad-CAM help make models more transparent, they might also create a false sense of trust by giving explanations after the fact that don't always accurately show how deep learning models make decisions. Also, the extra work that XAI methods require can make them hard to use in mobile security settings where resources are limited or real-time, which shows how important it is to have lightweight and inherently interpretable detection frameworks. In this context, lightweight detection models are architectures that are meant to lower the cost of computing and memory use. Examples include MobileNet-based CNNs, shallow ResNet variants, and hybrid frameworks that use model pruning and parameter sharing. Balanced benchmarks are evaluation datasets that have about the same number of benign and malicious samples, a variety of malware families, and representative obfuscation techniques. This makes performance assessment more fair and realistic. To make Android malware detection systems that work in the real world and can be used right away, it's important to clearly define lightweight architectures and standard benchmark design principles.

3.1. Limitations of the Study

This page presents a complete description of recent image-based Android malware detection methods, although it is crucial to highlight that it has several faults. First, it's challenging to directly and objectively evaluate how well different strategies work because the research that were looked at employed diverse datasets, preprocessing methods, and evaluation methodologies. You should take the reported accuracy and F1-score findings with a grain of salt because the dataset, the class imbalance, and the settings of the experiment all have an effect on them. Second, several of the studies we looked at evaluated their models when they weren't online. They don't really think about the challenges that come up when they are used in real time, such needing additional processing power, latency, and battery life on mobile devices. Even though explainable AI approaches like Grad-CAM and SHAP are being employed in malware detection systems, people are still worried about how well they operate in dangerous conditions and how much they cost to deploy in real-world security applications. The absence of well-defined and up-to-date benchmark datasets featuring obfuscated and dynamically changing malware samples presents a significant challenge to current research and comparative evaluations in this field.

4. Conclusion

This paper conducted a comprehensive review of recent advances in image-based Android malware detection from the year 2023 until 2025, with a special focus on the shift from traditional static and dynamic analysis to deep learning, transformer-based, and hybrid visual approaches. The comparative results demonstrated that CNN, ViT, and ensemble architectures consistently achieved high accuracy often exceeding 95–99% by effectively capturing spatial and semantic patterns in bytecode representations. Moreover, a few studies, such as FABLDroid and Explainable Deep Learning, introduced interpretability features, signaling a gradual movement toward more transparent and trustworthy AI systems. In general, these findings confirm that both visual and hybrid deep learning frameworks represent a very promising direction in robust Android malware detection. Future research needs to be directed at the development of lightweight, interpretable, and adaptive models, trained on balanced and up-to-date multimodal datasets to guarantee scalability, resilience, and real-

world applicability in evolving cyber-threat environments.

Conflict of Interest Statement

The author reports no conflicts of interest in this work.

References

Journal;

[1] P. N. Duy and N. T. Cam, "uitObfAMC: Obfuscated Android malware classification using deep learning on multi-feature information approach," Inf. Sci. (Ny), vol. 720, no. 122528, p. 122528, 2025. doi: 10.1016/j.ins.2025.122528

[2] S. Zhang, H. Su, H. Liu, and W. Yang, "MPDroid: A multimodal pre-training Android malware detection method with static and dynamic features," Comput. Secur., vol. 150, no. 104262, p. 104262, 2025. doi: 10.1016/j.cose.2024.104262

[3] H. Zhu, H. Wei, L. Wang, Z. Xu, and V. S. Sheng, "An effective end-to-end android malware detection method," Expert Syst. Appl., vol. 218, no. 119593, p. 119593, 2023. doi: 10.1016/j.eswa.2023.119593

[4] M. M. Yapici, "3DMalDroid: A novel 3D image based approach for android malware detection and classification," Comput. Electr. Eng., vol. 127, no. 110542, p. 110542, 2025. doi: 10.1016/j.compeleceng.2025.110542

[5] M. He, J. Ge, Z. Chen, J. Ling, and W. Kong, "MCGDroid: An android malware classification method based on multi-feature class-call graph characterization," Comput. Secur., vol. 160, no. 104713, p. 104713, 2026. doi: 10.1016/j.cose.2025.104713.

[6] A. Mohanraj and K. Sivasankari, "Hybrid temporal convolutional networks with LSTSVM model for android malware detection using explainable AI," J. Electr. Eng. Technol., vol. 21, no. 1, pp. 929–939, 2026. doi: 10.1007/s42835-025-02413-0

[7] M. M. Yapici, "A novel image based approach for mobile android malware detection and classification," Knowl. Based Syst., vol. 323, no. 113855, p. 113855, 2025. doi: 10.1016/j.knosys.2025.113855

[8] K. Kılıç, İ. Atacak, and İ. A. Doğru, "FABLDroid: Malware detection based on hybrid analysis with factor analysis and broad learning methods for android applications," Eng. Sci. Technol. Int. J., vol. 62, no. 101945, p. 101945, 2025. doi: 10.1016/j.jestch.2024.101945

[9] K. Bakour and H. M. Ünver, "DeepVisDroid: android malware detection by hybridizing image-based features with deep learning techniques," Neural Comput. Appl., vol. 33, no. 18, pp. 11499–11516, 2021. doi: 10.1007/s00521-021-05816-y

[10] S. K. Smmarwar, G. P. Gupta, and S. Kumar, "Android malware detection and identification frameworks by leveraging the machine and deep learning techniques: A comprehensive review," Telematics and Informatics Reports, vol. 14, no. 100130, p. 100130, 2024. doi: 10.1016/j.teler.2024.100130

[11] S. Chen, B. Lang, H. Liu, Y. Chen, and Y. Song, "Android malware detection method based on graph attention networks and deep fusion of multimodal features," Expert Syst. Appl., vol. 237, no. 121617, p. 121617, 2024. doi: 10.1016/j.eswa.2023.121617

[12] M. S. Wasif, M. P. Miah, M. S. Hossain, M. J. F. Alenazi, and M. Atiquzzaman, "CNN-ViT synergy: An efficient Android malware detection approach through deep learning," Comput. Electr. Eng., vol. 123, no. 110039, p. 110039, 2025. doi: 10.1016/j.compeleceng.2024.110039

[13] M. Najibi and A. J. Bidgoly, "Towards a robust android malware detection model using explainable deep learning," J. Inf. Secur. Appl., vol. 93, no. 104191, p. 104191, 2025. doi: 10.1016/j.jisa.2025.104191

[14] K. Kılıç, İ. A. Doğru, and S. Toklu, "PermQRDroid: Android malware detection with novel attention layered mini-ResNet architecture over effective permission information image," PeerJ Comput. Sci., vol. 10, p. e2362, 2024. doi: 10.7717/peerj-cs.2362

[15] S. Nethala, P. Chopra, K. Kamaluddin, S. Alam, S. Alharbi, and M. Alsaffar, "A deep learning-based ensemble framework for robust android malware detection," IEEE Access, vol. 13, pp. 46673–46696, 2025. doi: 10.1109/access.2025.3551152

[16] A. Ksibi, M. Zakariah, L. Almuqren, and A. S. Alluhaidan, "Efficient android malware identification with limited training data utilizing multiple convolution neural network techniques," Eng. Appl. Artif. Intell., vol. 127, no. 107390, p. 107390, 2024. doi: 10.1016/j.engappai.2023.107390