ISSN: 2583-9888
DOI: doi.org/10.51219/JAIMLD/Samuel-Akerele/581

URF PUBLISHERS
connect with research world

Journal of Artificial Intelligence, Machine Learning and Data Science

https://urfpublishers.com/journal/artificial-intelligence

Vol: 3 & Iss: 3

Research Article

Modern Deep Learning Approaches for Malware Detection and Classification

Samuel Akerele[1], Nabeela Temitayo Adebola[2], Oluwole Fagbohun[3], Goodness Obaika[4], Light Chukwubuikem Nwokocha[4] and Paul Stephen[5]

[1]Vuhosi Limited, Nigeria

[2]University of Salford United Kingdom

[3]Vuhosi Limited, United Kingdom

[4]Independent Researcher, United Kingdom

[5]University of Bedfordshire, United Kingdom

Citation: Akerele S, Adebola NT, Fagbohun O, et al., Modern Deep Learning Approaches for Malware Detection and Classification. *J Artif Intell Mach Learn & Data Sci 2025* 3(3), 2761-2768. DOI: doi.org/10.51219/JAIMLD/Samuel-Akerele/581

Received: 25 June, 2025; **Accepted:** 01 July, 2025; **Published:** 05 July, 2025

***Corresponding author:** Samuel Akerele, Vuhosi Limited, Nigeria

ABSTRACT

TMalware continues to evolve in complexity, employing evasion tactics that challenge traditional detection methods. In response, deep learning has emerged as a powerful approach to automate and enhance malware detection across static and dynamic analysis domains. This paper provides a comprehensive review of recent advances in deep learning-based detection systems, with particular emphasis on hybrid models that integrate static code features and runtime behavioral indicators. We examine key architectural approaches including convolutional neural networks for spatial pattern recognition, recurrent neural networks for sequential data analysis, graph neural networks for structural understanding and transformers for context-aware, multi-modal inference. Benchmarks such as EMBER, Drebin and Malicia are discussed as standard datasets supporting reproducibility and comparative evaluation. Case studies of prominent malware families such as WannaCry, TrickBot and Emotet illustrate the operational relevance of hybrid approaches. In addition, we explore emerging trends such as federated learning for privacy-preserving collaboration, multimodal architectures for enriched feature learning, lightweight models for edge-based detection and adversarial defences for model robustness. Persistent challenges include limited labelled data, the interpretability of model decisions and the need to address concept drift in evolving threats. This review highlights the growing maturity of deep learning techniques in cybersecurity and outlines future directions for building more resilient, efficient and explainable malware detection frameworks.

1. Introduction

As malware becomes more sophisticated, it presents mounting challenges and risks to people organizations and key technological infrastructure. Traditional signature-based antivirus methods have become increasingly ineffective against modern polymorphic or zero-day malware, driving a surge of interest in machine learning (ML) and deep learning (DL) for automated detection[1,2]. Malware analysis techniques can be broadly classified as static, where the code is examined without execution or dynamic, where the program's behavior is observed at runtime. Static analysis is fast and scalable, typically involving the extraction of features such as Portable Executable (PE) header fields, embedded strings or requested permissions without

running the file[3-5]. However, it performs poorly against encrypted, packed or heavily obfuscated code, as these techniques hinder feature extraction[6,7]. In contrast, dynamic analysis observes system behavior such as API calls, registry modifications and network activity within a sandboxed environment, allowing the detection of runtime tactics[8,9]. While effective in revealing malicious intent, dynamic analysis is slower and vulnerable to evasion by malware that detects virtualized environments or employs delayed execution strategies[10].

To address these limitations, hybrid analysis combines static and dynamic features, thereby leveraging the strengths of both techniques[11]. As noted by Hussain, et al., "the hybrid approach is concerned with solutions of both dynamic and static analysis of malware detection and classification." Similarly, Damodaran, et al.[6] demonstrate that hybrid methods often achieve higher accuracy than single-mode approaches by capturing a broader set of malware behaviors. This fusion of code-level and behavioral features allows detection systems to generalize better across different malware types, including novel or obfuscated variants[12]. Modern malware detection increasingly integrates deep learning, which automates feature extraction from raw input data and enables complex pattern recognition across large-scale corpora[13]. DL models such as convolutional neural networks (CNNs), recurrent neural networks (RNNs), graph neural networks (GNNs) and transformers have been effectively applied to both static inputs (e.g. raw binaries, opcode sequences or visual representations) and dynamic inputs (e.g. system call traces, performance counters)[14].

This review paper synthesizes recent advances in hybrid malware detection using deep learning, while also summarizing progress in static-only and dynamic-only DL approaches for context. We explore the major DL architectures used, evaluate benchmark datasets such as EMBER, Drebin and Malicia and discuss case studies of prominent malware families (WannaCry, TrickBot, Emotet). Furthermore, we address critical challenges such as adversarial evasion, data scarcity, lack of explainability and concept drift and highlight emerging trends including federated learning, multimodal deep networks, transformer-based models and real-time edge deployment.

1.1. Static, dynamic and hybrid analysis

Static analysis inspects a program's code or structure without executing it. Features typically include Portable Executable (PE) header fields, imported libraries, embedded strings, n-grams of opcodes or visual representations of binaries[10]. Static analysis is fast, scalable and safe, as it avoids the risk of executing malicious code. However, it is vulnerable to obfuscation, encryption and packing techniques, which can conceal malicious payloads and hinder feature extraction. As Kang and Won explain, static analysis has the "advantage of taking less time to extract features," but "feature extraction is difficult if the malware is packed or obfuscated." Furthermore, static-only deep learning (DL) detectors trained on known patterns can be bypassed through minor code mutations and analysis may be computationally expensive for large binaries[15]. Examples of static DL approaches include converting binaries into grayscale images for classification with convolutional neural networks (CNNs)[16] or processing raw byte sequences with 1D CNNs such as MalConv[15].

Dynamic analysis involves executing a program within a sandboxed environment or emulator and observing its behavior. Typical features extracted include API or system call traces, file system activity, network traffic and CPU performance counters[6]. Dynamic analysis can expose malicious actions that are not evident statically, particularly for packed or encrypted malware[8,6]. For instance, ransomware may only reveal its encryption routines during execution. However, dynamic analysis is slower, resource-intensive and susceptible to evasion, many malware samples now include sandbox-detection techniques or delayed payload activation[9,17,7]. Dynamic techniques often employ recurrent neural networks (RNNs) to model sequential behavior like API-call patterns.

Hybrid analysis combines static and dynamic techniques to offset the weaknesses of each and achieve broader coverage. A hybrid system may fuse static opcode features with dynamic API-call statistics, producing a richer representation[11]. As Damodaran, et al. highlight[6], "both static and dynamic features are used to detect the malware more accurately than either approach alone." Similarly, Nazim, et al.[12] note that their proposed hybrid system "can overcome the disputes presented by dynamic analysis and provide a more extensive consideration of malware activities." Studies consistently show hybrid models outperform static or dynamic only models on detection benchmarks[12]. For example, combining PE metadata with sandbox logs can yield high-precision classifiers[18].

While static analysis is quick and suitable for early-stage detection, dynamic methods uncover complex malicious actions during execution. Hybrid models bring these two strengths together, offering broader protection. The optimal approach depends on system requirements: static for lightweight environments, hybrid for comprehensive threat monitoring. This paper concentrates on deep learning approaches, with a focus on hybrid analysis systems.

1.2. Deep learning architectures for malware detection

Deep learning enables end-to-end malware classification by automatically learning hierarchical features from raw data. Key DL model families used in malware detection include:

1.2.1. CNNs (Convolutional Neural Networks): CNNs are widely applied to static malware analysis, particularly when binaries are converted into images. One method maps binary bytes into 2D grayscale images and trains a CNN to detect visual patterns associated with malicious code[16]. Another method, MalConv, processes entire raw byte sequences using a 1D convolutional architecture, avoiding manual feature extraction[15]. CNNs are also used on dynamic features such as CPU performance counters or memory traces[19,20]. While CNNs excel at recognizing local spatial or sequential patterns, they typically require large datasets and may be susceptible to adversarial byte-level modifications.

1.2.2. RNNs (Recurrent Neural Networks): RNNs, including LSTMs and GRUs, are suited to sequential inputs such as API calls, system logs or opcode streams. These models can capture temporal dependencies in program execution. For instance, a bidirectional GRU with attention has been used to classify PowerShell scripts based on their semantic structure[21]. However, RNNs may suffer from vanishing gradients on long sequences and can be slower to train than feedforward models. They are often combined with CNN layers in hybrid models that capture both spatial and sequential dependencies.

1.2.3. GNNs (Graph Neural Networks): Graph neural networks have emerged for learning on structured representations like control-flow graphs (CFGs) or call graphs. GNNs learn embeddings over program nodes and their relationships, enabling nuanced modelling of malware's logical structure[22]. As Bilot, El Madhoun, Al Agha and Zouaoui[23] notes, GNNs "reach competitive results in learning robust embeddings from malware represented as expressive graph structures." These models excel at capturing code logic but are computationally demanding and can be vulnerable to adversarial graph manipulation (e.g. injecting dead code to confuse the model).

1.3. Transformer models

Transformers and attention-based architectures are increasingly popular in malware detection. Originally designed for NLP, transformers have been applied to disassembled code, system logs and even visual binary representations. Text-based models like BERT or GPT can tokenize and model opcode sequences or shellcode[13], while Vision Transformers (ViT) operate on binary images partitioned into patches. Graph Transformers extend attention mechanisms to structural data. Alshomrani, et al.[13] propose a hybrid transformer framework that integrates static metadata and dynamic behavioral features via self-attention layers. Transformers can model long-range dependencies more effectively than RNNs but are computationally intensive and often require fine-tuning on large corpora.

1.4. Hybrid networks and ensembles

To harness the strengths of multiple architectures, many recent systems adopt hybrid or ensemble models. A typical setup uses CNNs to extract features from raw byte inputs and passes them to an RNN or transformer for sequential reasoning[24,25]. Ensemble models may also combine static and dynamic classifiers using voting or stacking. Song, et al.[21] report that such combinations improve overall robustness and reliability. These architectures are well-suited to hybrid malware detection, where both code structure and runtime behavior are critical. Each architecture can be deployed in static-only, dynamic-only or hybrid configurations, depending on data availability and application context. CNNs are ideal for raw binary data, RNNs excel at modelling behavior and transformers offer flexibility across modalities. GNNs remain a powerful but niche option for structural analysis. The next section explores real-world malware case studies and public datasets that support these modelling approaches.

2. Case Studies: Malware Families

Real-world malware families illustrate the need for robust DL-based detection systems. Three significant examples underscore the diversity and evolution of modern malware:

- **WannaCry (2017 Ransomware):** WannaCry was a fast-spreading ransomware worm that exploited the Eternal Blue vulnerability in Microsoft's SMB protocol. It infected over 300,000 machines across 150 countries, causing an estimated $8 billion in global damages[21]. Static analysis can detect known signatures or suspicious encryption routines, but dynamic analysis is needed to observe its real-time encryption behaviors. Deep learning-based hybrid systems are particularly suited to generalizing across variants, especially as WannaCry-like malware continues to evolve beyond its original exploit vector.
- **TrickBot:** Initially designed as a banking Trojan, TrickBot evolved into a modular, multi-stage malware framework capable of credential harvesting, lateral movement and ransomware delivery[26,27]. It is typically distributed via phishing emails with malicious attachments or embedded links. TrickBot is polymorphic, its components are frequently encrypted or obfuscated, complicating static detection. DL models must integrate both static artefacts (e.g. obfuscated PE headers) and dynamic indicators (e.g. registry edits, command-and-control beacons). TrickBot's architecture demands hybrid or multimodal detection systems capable of recognizing stage-specific patterns.
- **Emotet (2014-2021):** Emotet is a notorious banking Trojan turned malware delivery botnet. It spreads through malspam campaigns using malicious Word macros, then downloads modules for credential theft and further infections[28,29]. It is polymorphic and virtual-machine-aware, often halting execution in sandboxed environments. Emotet has caused individual organizations up to $1 million in damages[29]. Effective DL systems must address both its adversarial evasion capabilities (e.g. VM detection, API obfuscation) and structural modularity by fusing static and behavioral signals. These case studies demonstrate the complexity of contemporary malware. Models focused on a single modality, static or dynamic may miss essential clues. Hybrid DL approaches, by design, offer broader coverage and improved adaptability to sophisticated threats such as TrickBot and Emotet.

2.1. Datasets and benchmarks

Developing robust deep learning models for malware detection relies heavily on access to large-scale, diverse and well-labelled datasets. Below are some of the most widely used benchmarks in academic and applied research **(Table 1)**:

Table 1: Commonly Used Malware Detection Datasets.

Dataset	Domain	Samples	Description/Citation
EMBER (2018)	Windows PE files	~1.1 million (0.9M train, 0.2M test)	Provided by Endgame; includes static features for benign and malicious Windows binaries[30].
Drebin (2014)	Android APKS	5,560 malware apps	Android malware dataset covering 179 families; features include permissions, intents, API calls[31].
Malicia (2016)	Drive-by malware	11,688 binaries	Collected from 500 malicious domains over 11 months; useful for dynamic analysis[32].

Several foundational datasets have significantly shaped the landscape of modern malware detection research by enabling reproducible experimentation and comparative evaluation. EMBER[30] is widely recognized as the leading public dataset for Windows malware detection. It provides rich, static features extracted from Portable Executable (PE) files and is commonly used in both classical machine learning and deep learning studies. Drebin[31], on the other hand, targets Android malware

and includes a labelled set of malicious and benign applications with extracted features such as API calls and permission lists. This makes it a vital resource for mobile security research. Complementing these is the Malicia dataset[32], which comprises a curated collection of drive-by download malware samples, supporting research into browser-based and dynamic malware threats.

These three datasets represent different operational environments and threat models including Windows desktops android mobile platforms and web-based attacks, allowing researchers to evaluate detection techniques across a broad spectrum of malware behaviors. Beyond these, several additional datasets are also gaining prominence in the field. The MalConv dataset[15] offers raw PE binaries and is particularly useful for testing convolutional neural network-based detection approaches. VirusShare and VirusTotal serve as large-scale repositories with millions of diverse malware samples[33,34], often used for exploratory analyses, signature verification or dataset expansion. Meanwhile, datasets like CIC-MalNet, IoT-23 and CICIDS (2017/2022) address malware in networked and IoT environments, enabling studies in real-time intrusion detection and network-based threat assessment.

Collectively, these datasets underpin much of the contemporary progress in malware detection research. EMBER remains the de facto benchmark for static Windows malware classification, Drebin is central to Android-related studies and Malicia continues to be a key resource for investigating dynamic, drive-by malware. The availability of these datasets supports standardized evaluation and encourages methodological transparency across the research community.

2.2. Evaluation metrics and benchmarks

Deep learning-based malware classifiers are typically assessed using well-established classification metrics, including accuracy (overall proportion of correct predictions), precision (proportion of true positives among predicted positives), recall (proportion of actual positives correctly identified), F1-score (harmonic mean of precision and recall) and ROC-AUC (area under the receiver operating characteristic curve). For example, Nazim, et al. report a recall of 86.5%[12], an F1-score of 85.0% and a precision of 79.9% for their multimodal classification model. These metrics are often calculated using 10-fold cross-validation or stratified train-test splits from benchmark datasets such as EMBER[30], Drebin[31] and Malicia[32].

In cybersecurity contexts, recall is especially critical, as false negatives (missed malware) pose significant risks. However, low precision can result in excessive false positives, burdening security analysts with unnecessary alerts. Confusion matrices, along with false positive and false negative rates, provide additional insights into model performance and operational viability. Benchmarking against standard datasets facilitates fair comparisons between different architectures, but caution is needed: overfitting on outdated, synthetic or imbalanced data can lead to misleadingly high scores. As Song, et al. note[21], many models report accuracy above 90%, yet fail to generalize in real-world settings. For deployment, it is essential to implement continual learning and live model validation to maintain detection effectiveness over time.

2.3. Model comparison and evaluation

A comparative summary of deep learning models used in malware detection is outlined in Table 2, detailing their input modalities, advantages, drawbacks and notable sources. Convolutional Neural Networks (CNNs) perform well with inputs like raw binaries, opcode sequences or malware visualizations by capturing spatial features, though they typically demand extensive training data and lack transparency in decision-making[15,21]. Recurrent Neural Networks (RNNs), such as LSTMs and GRUs, are ideal for modelling sequential data like API call traces, though they face challenges with training efficiency and gradient stability[21]. Graph Neural Networks (GNNs) capture structural relationships within control or function call graphs, providing deep semantic insights at the cost of complex graph construction and vulnerability to structural manipulation[22,23]. Transformer-based models offer strong performance on tokenized code, text or image patches, benefiting from self-attention mechanisms but requiring significant compute resources and pretraining[13]. Hybrid or ensemble models integrate multiple modalities (e.g., static and dynamic features) to improve detection accuracy and resilience, though they tend to be more complex and computationally intensive[35,36].

The table serves as a quick reference for selecting suitable architectures based on the type of malware data, target use case and resource constraints **(Table 2)**.

Table 2: Comparison of deep learning model families for malware detection.

Model Type	Input Features	Advantages	Limitations	**Key References**
CNN	Raw bytes, opcodes, binary images	Learns spatial patterns; excels on image-like input	Needs large data; sensitive to adversarial noise; lacks interpretability	[15,21]
RNN (LSTM/GRU)	Sequences (API calls, opcode streams)	Captures temporal dependencies; ideal for dynamic sequences	Prone to vanishing gradients; slow training	[21]
GNN	Control Flow Graphs, Function Call Graphs	Captures graph structures; relational semantics	Graph construction cost; sensitive to structure-tampering	[22]
Transformer	Tokenized byte/code/text/image patches	Long-range dependencies; multi-modal flexibility	Large model size; requires pretraining	[13]
Hybrid/Ensemble	Multi-modal (static + dynamic features)	Combines complementary features; higher accuracy	More complex; higher compute; interpretation difficulty	[21]

2.4. Deep learning on static features

Static features are widely used with CNNs and vision-based approaches. For instance, converting a binary to a grayscale image enables CNNs or Vision Transformers (ViTs) to learn visual malware patterns[16,12]. MalConv, a CNN on raw PE bytes, learns end-to-end representations from binaries[15]. Some studies report CNN-based models achieving up to 98% accuracy on known datasets like Malicia[37-39]. In more experimental efforts, researchers have explored generative models (e.g. GANs or

autoencoders) for malware synthesis or detection[11]. However, these are often less interpretable and computationally expensive than discriminative models and rarely used in real-world systems.

2.5. Deep learning on dynamic features

Deep learning approaches to dynamic malware analysis have demonstrated significant potential by leveraging models such as recurrent neural networks (RNNs) and transformers to capture execution-time behaviors, including API call sequences and performance counters. Alsumaidaee, Yahya and Yaseen introduced a hybrid 1D-CNN-LSTM architecture designed to process CPU and memory performance data during runtime[40]. Their model, which achieved 100% detection accuracy against sophisticated malware, illustrated the effectiveness of extracting spatial-temporal patterns from low-level hardware behavior for real-time anomaly detection in endpoint security systems. This direction is further supported by Damodaran, et al.[6], who demonstrated that RNNs and transformers are well-suited to identifying evasion-resistant sequential patterns within sandbox execution logs, reinforcing their utility in dynamic environments.

To improve robustness and generalizability, hybrid ensembles combining deep learning with classical machine learning techniques are increasingly adopted. Adamu, Awan and Younas implemented a CNN-XG Boost framework in which convolutional networks extracted representational features from raw byte sequences and XG Boost classified the outputs with 99.3% accuracy, while maintaining resilience against adversarial perturbations[41]. Similarly, Ahmed reported that integrating classical ensemble models such as Random Forests with deep learning components significantly reduced false positive rates by as much as 47 percent when detecting polymorphic malware variants[42].

These methods have proven particularly effective against advanced threats such as fileless or polymorphic malware, owing to their ability to encode time-series dependencies and support multi-modal fusion. As emphasized by both Alsumaidaee, et al.[40] and Or-Meir, et al.[9], the integration of deep sequence models with behavioral data enhances the reliability of detection systems operating in dynamic and adversarial threat landscapes.

2.6. Multimodal and hybrid networks

Multimodal DL systems integrate static and dynamic features through parallel branches or late fusion. Nazim, et al.[12] introduced a CNN+MLP hybrid that jointly learns from malware images and handcrafted numerical features. Their system achieved 95.36% accuracy, outperforming single-modal variants. The authors conclude that "late fusion of numeric and visual data makes the model more robust" to diverse malware types. Other hybrid systems include hierarchical CNN-BiLSTM models that combine temporal and spatial learning. While these models often deliver higher accuracy[43], they are more demanding to train and require paired datasets (static + dynamic) for each malware instance.

2.7. Core challenges

2.7.1. Adversarial evasion: DL models are vulnerable to adversarial manipulation. Attackers can inject benign bytes, reorder instructions or insert no-op API calls to alter classification outcomes[44]. Raff, et al. showed that simple byte padding could bypass MalConv[15]. Defenses like adversarial training and feature suppression have been proposed, but attackers often evolve faster than defenses[23]. GNNs are also at risk: inserting dummy nodes or altering graph structure can cause GNNs to misclassify malware[45,23]. The ongoing arms race demands robust, certifiable DL defenses.

2.7.2. Data scarcity and diversity: Malware datasets often suffer from issues related to size, diversity and annotation quality. As noted by Song, et al.[21], many deep learning studies depend on datasets that are small and imbalanced, which undermines model robustness and generalizability. Compounding this challenge is the fast-changing nature of malware, with millions of new, previously unseen variants appearing each month. Hybrid detection approaches further complicate data demands by requiring both static and dynamic information, which can be costly to obtain. Strategies such as data augmentation using GANs, transfer learning and federated learning offer promising solutions, particularly the latter, which supports decentralized training without sharing raw data[46].

2.7.3. Explainability (Interpretability): Security analysts need to understand model predictions. Unfortunately, many DL models act as black boxes. Song, et al.[21] call for better explainability, such as identifying which API call or binary segment triggered an alert. Existing efforts like attention maps or saliency scores are promising but insufficient. Models that can explain "why" will improve analyst trust; aid debugging and even support adversarial detection by exposing suspicious patterns.

2.7.4. Concept drift: Malware tactics and payloads evolve continuously. A model trained on yesterday's threats may underperform on today's. Li, Iyengar, Kundu and Bertino[47] propose domain-adversarial GNNs to learn invariant features across time, enhancing robustness against drift. Concept drift manifests as changes in binary structure, API usage or obfuscation patterns. Adaptive retraining, drift detection tools and semi-supervised learning frameworks are critical in mitigating performance degradation over time.

3. Recent Trends and Emerging Directions

To address longstanding challenges in malware detection including adversarial evasion, data scarcity and explainability, a number of recent trends have emerged in DL-based security research.

3.1. Federated learning

Federated learning (FL) has gained traction as a privacy-preserving paradigm that enables collaborative model training across distributed datasets. In this framework, local models are trained on device or on site and only model updates (e.g., gradients or weights) are shared with a central aggregator. Çıplak, et al.[46] introduced FEDetect, a federated malware classifier that achieved detection accuracy of up to 99.9% while preserving the confidentiality of each participating organization's data. As the authors explain, FL "eliminates the requirement for centralized data collecting while preserving privacy." FL also has the potential to improve model generalizability by learning from non-IID (non-independent and identically distributed) data sources across enterprises[46]. It has also been applied in federated anomaly detection for IoT malware.

3.2. Multimodal learning

Given the diversity of malware representations, multimodal

learning is a natural extension of hybrid analysis. These models integrate different input types such as binary images, opcode sequences, API call logs or network metadata either via parallel model branches or attention-based fusion. Nazim, et al.[12] developed a CNN+MLP ensemble to process both malware images and numerical vectors, achieving superior performance compared to single-modal baselines. Their findings confirm that "late fusion of numeric and visual data makes the model more robust." Other works combine text (e.g., disassembled code), visual patterns and system behaviors using cross-modal transformers. Multimodal architectures are expanding into audio and side-channel signal domains as well, enabling richer behavioral profiling.

3.3. Transformer-based detection

Transformers, first established in NLP, are increasingly used for malware detection due to their powerful self-attention mechanism and sequence modelling capacity. Text-based transformers (e.g. BERT, GPT-2) have been fine-tuned on opcode or small code for Android malware[13], while Vision Transformers (ViTs) are used on malware byte images[48,49]. As Alshomrani, et al.[13] state, transformers are "among the most potent for text-based malware detection," though their computational overhead remains a concern. Researchers are actively working on more efficient transformer variants such as TinyBERT and DistilBERT to balance performance with resource constraints.

3.4. Real-time and edge-based inference

Malware detection on edge devices such as routers, IoT hubs or endpoint clients requires lightweight, efficient models. Recent research explores the deployment of compact LLMs (e.g., DistilBERT) on low-power hardware to provide near real-time threat detection[50]. However, these models must address strict limitations on compute, storage and energy. Techniques such as model pruning, quantization and knowledge distillation are widely used to compress DL architectures without significant loss in performance. Public datasets such as Edge-IIoTset, TON-IoT and CIC-IDS are commonly used to benchmark edge-aware malware detection models.

3.5. Adversarial and robustness research

Adversarial machine learning remains central to DL security research. Gibert, et al. proposed a randomized smoothing technique for malware detection by dividing binaries into chunks and aggregating their predictions[51], significantly enhancing model robustness against byte-level perturbations. Other work explores interpretability-guided attacks, which exploit saliency maps to find weak points in classifiers[44]. These developments point toward more certifiable defenses, with many systems now integrating adversarial training and explainability layers into deployment pipelines.

3.6. Multitask and meta-learning

Emerging research also explores multitask learning (e.g., malware detection, family classification, behavior prediction) and meta-learning, where models are trained to quickly adapt to novel malware variants. These paradigms reduce data requirements and training cycles, particularly useful in scenarios with fast-moving threat landscapes. Each of these innovations in federated learning, multimodal architectures, transformers, real-time inference and adversarial robustness contributes to solving critical limitations of current DL approaches and represents an exciting evolution of the field.

4. Conclusion

Modern malware detection is undergoing a paradigm shift driven by deep learning and hybrid analysis techniques. A diverse array of architectures including CNNs, RNNs, GNNs, Transformers and their combinations have demonstrated impressive performance in static, dynamic and hybrid detection pipelines. Benchmark datasets such as EMBER[30], Drebin[31] and Malicia[32] underpin much of this research. Notable case studies such as WannaCry, TrickBot and Emotet underscore the evolving sophistication of malware, necessitating robust and adaptable detection systems. Yet, the challenges are formidable. Adversarial evasion remains a high-priority threat[52,45]. Data scarcity, especially for zero-days, continues to hinder generalization[21]. Explainability is underdeveloped, limiting trust and adoption in operational contexts[21]. Concept drift in malware behavior demands models that adapt continuously to shifting attack patterns[47,53]. Promisingly, the research community is responding. Federated learning offers privacy-preserving training across organizations[46]. Multimodal architectures capture richer feature relationships[12]. Transformer models, particularly those adapted for code or vision tasks, enable context-aware detection[13]. Edge-aware inference expands deep learning coverage into constrained environments[54]. These innovations point towards a future of intelligent, autonomous and explainable malware detection frameworks. Hybrid deep learning detectors that integrate both static and dynamic perspectives have emerged as a highly effective strategy. By balancing scalability, detection accuracy and operational resilience, these models are well-suited to address the complex and evolving nature of modern malware threats.

5. References

1. Ayeni OA. Machine Learning Techniques for Automated Malware Detection. International Journal of Cyber Security and Digital Forensics, 2023.
2. Rajkumar T, Sapra P. Deep Learning for Malware Classification and Detection. In Scalable Neural Network Models for High Dimensional Data Analysis in Cyber Defense Applications, 2025.
3. Yousuf MI, Anwer I, Riasat A, et al. Windows malware detection based on static analysis with multiple features. PeerJ Computer Science, 2023;9: 1319.
4. Yuk CK, Seo CJ. Static analysis and machine learning-based malware detection system using PE header feature values. International Journal of Innovative Research and Scientific Studies, 2022;5: 368-374.
5. Puranik PA. Static malware detection using deep neural networks on portable executables (Master's thesis, University of Nevada, Las Vegas). UNLV Theses, Dissertations, Professional Papers and Capstones, 2019.
6. Damodaran A, Di Troia F, Visaggio C, et al. A comparison of static, dynamic and hybrid analysis for malware detection, 2017.
7. AliAhmad A, Eleyan D, Eleyan A, et al. Malware detection issues, future trends and challenges: A survey. In 2023 International Symposium on Networks, Computers and Communications (ISNCC), 2023: 1-6.
8. Zhang Z, Qi P, Wang W. Dynamic malware analysis with feature engineering and feature learning. In Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence. AAAI Press, 2020: 2756-2763.

9. Or-Meir O, Nissim N, Elovici Y, et al. Dynamic malware analysis in the modern era-A state of the art survey. ACM Computing Surveys, 2019;52(5).
10. Kang J, Won Y. A study on variant malware detection techniques using static and dynamic features. Journal of Information Processing Systems, 2020;16: 1396-1412.
11. Hussain M, Ab Razak MF. Deep learning-based hybrid analysis of malware detection and classification. Journal of Cyber Security and Mobility, 2023;12: 1-32.
12. Nazim S, Alam MM, Rizvi S, et al. Multimodal malware classification using ensemble deep learning. Scientific Reports, 2025;15: 10567.
13. Alshomrani M, Albeshri A, Alturki B, et al. Survey of Transformer-Based Malicious Software Detection Systems. Electronics, 2024;13: 4677.
14. Maniriho P, Mahmood AN, Chowdhury MJM. A survey of recent advances in deep learning models for detecting malware in desktop and mobile platforms. La Trobe University, 2024.
15. Raff E, Barker J, Sylvester J, et al. Malware detection by eating a whole EXE. AAAI Workshop on AI for Cybersecurity, 2018.
16. Nataraj L, Karthikeyan S, Jacob G, et al. Malware images: Visualization and automatic classification. VizSec, 2011;11.
17. Vemparala S. Malware detection using dynamic analysis (Master's project, San Jose State University). SJSU ScholarWorks, 2015.
18. Alhaidari F, Abu Shaib N, Alsafi M, et al. ZeVigilante: Detecting zero-day malware using machine learning and sandboxing analysis techniques. Computational Intelligence and Neuroscience, 2022: 15.
19. Kim H, Kim M. Malware detection and classification system based on CNN-BiLSTM. Electronics, 2024;13: 2539.
20. Singh AK, Taterh S, Dadheech P. Dynamic feature-based detection of malware in non-executable files using a 1D convolutional neural network. Panamerican Mathematical Journal, 2025;35: 678.
21. Song J, Zhang D, Wang J, et al. Application of deep learning in malware detection: A review. Journal of Big Data, 2025;12.
22. Malhotra V, Potika K, Stamp M. A comparison of graph neural networks for malware classification, 2023.
23. Bilot T, El Madhoun N, Al Agha K, et al. A survey on malware detection with graph representation learning. Université Paris-Saclay, CNRS, LISITE Laboratory, ISEP and Sorbonne Université, 2024.
24. Li H, Li Z, Zhang S, et al. Malicious DNS detection by combining improved transformer and CNN. Scientific Reports, 2024;14: 30248.
25. Ullah F, Alsirhani A, Alshahrani MM, et al. Explainable malware detection system using transformers-based transfer learning and multi-model visual representation. Sensors, 2022;22: 6766.
26. CISA & FBI. Joint Cybersecurity Advisory - AA21-076A - TrickBot Malware, 2021.
27. U.S. Department of Health and Human Services (HHS). TrickBot, Ryuk and the HPH Sector, 2020.
28. Reddy DAR, Mohan CB. Emotet: A sophisticated and persistent malware for stealing information, its attack and prevention strategies. International Research Journal of Engineering and Technology (IRJET), 2023;10: 133-138.
29. Kuraku S, Kalla D. Emotet malware - A banking credentials stealer. IOSR Journal of Computer Engineering (IOSR-JCE), 2020;22: 31-40.
30. Anderson HS, Roth P. Ember: an open dataset for training static pe malware machine learning models, 2018.
31. Arp D, Spreitzenbarth M, Hubner M, et al. Drebin: Effective and explainable detection of Android malware in your pocket. In NDSS 2014 - Network and Distributed System Security Symposium. Internet Society, 2014.
32. Nappa A, Rafique MZ, Caballero J. Driving in the Cloud: Analysis of Drive-by Download Operations. DIMVA, 2013.
33. VirusShare. VirusShare.com malware sample repository, 2025.
34. VirusTotal. VirusTotal online malware analysis service, 2025.
35. Demirkıran F, Çayır A, Ünal U, et al. An ensemble of pre-trained transformer models for imbalanced multiclass malware classification. Computers & Security, 2022;121: 102846.
36. Kunwar P, Aryal K, Gupta M, et al. SoK: Leveraging transformers for malware analysis. IEEE Transactions on Dependable and Secure Computing, 2025.
37. Vasan D, Alazab M, Wassan S, et al. Image-based malware classification using ensemble of CNN architectures (IMCEC). Computers & Security, 2022;34: 1968-1983.
38. Falana OJ, Sodiya AS, Onashoga SA, et al. Mal-Detect: An intelligent visualization approach for malware detection. Journal of King Saud University - Computer and Information Sciences, 2022.
39. Prima B, Bouhorma M. Using transfer learning for malware classification. International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, XLIV-4/W3, 2020: 343-348.
40. Alsumaidaee YAM, Yahya MM, Yaseen AH. Optimizing Malware Detection and Classification in Real-Time Using Hybrid Deep Learning Approaches. International Journal of Safety and Security Engineering, 15(1): 141-150.
41. Adamu U, Awan I, Younas M. Malware classification using deep learning and ensemble framework. University of Bradford & Oxford Brookes University, 2024.
42. Ahmed SMAA. Applying ensemble machine learning techniques to malware detection. Journal of Information Systems Engineering and Management, 2025;10.
43. Zhang L, Liu T, Shen K, et al. A novel approach to malicious code detection using CNN-BiLSTM and feature fusion, 2024.
44. Li L. Comprehensive survey on adversarial examples in cybersecurity, 2024.
45. Peng H, Yu Z, Zhao D, et al. Evading control flow graph-based GNN malware detectors via active opcode insertion method with maliciousness preserving, 2025.
46. Çıplak Z, Yıldız K, Altınkaya S. FEDetect: Federated learning-based malware detection and classification. Applied Journal of Computer Security and Electronic Forensics, 2025;3: 1-14.
47. Li AS, Iyengar A, Kundu A, et al. Revisiting concept drift in Windows malware detection: Adaptation to real drifted malware with minimal samples. In Proceedings of the Network and Distributed System Security (NDSS) Symposium 2025: 240830.
48. Bavishi S, Modi S. Accelerating malware classification: A vision transformer solution, 2024.
49. Rahali A, Akhloufi MA. MalBERTv2: Code aware BERT-based model for malware identification. Big Data and Cognitive Computing, 2023;7: 60.
50. Rondanini C, Carminati B, Ferrari E, et al. Malware detection at the edge with lightweight LLMs: A performance evaluation, 2025.
51. Mukherjee K, Wiedemeier J, Wang T, et al. Evading provenance-based ML detectors with adversarial system actions. In Proceedings of the 32nd USENIX Security Symposium, 2023: 3535-3552.

52. Gibert D, Zizzo G, Le Q, et al. Adversarial Robustness of Deep Learning-based Malware Detectors via (De)Randomized Smoothing, 2024.

53. He Y, Lei J, Qin Z, et al. Going proactive and explanatory against malware concept drift. Zhejiang University, 2024.

54. Zhang G, Guo W, Tan Z, et al. AMP4EC: Adaptive Model Partitioning Framework for Efficient Deep Learning Inference in Edge Computing Environments, 2025.