IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940



International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.151106

Malware Image Classification: A Literature Survey

Abhijith G[1], Jijohns Johnson[2], Vishnu S[3], Vishakh V Prakash[4], Anpu Alexander[5], Girija V R[6]

Department of Computer Science and Engineering, College of Engineering Kottarakkara, Kollam, Kerala, India[1-6]

Abstract: Malware classification is an important cybersecurity task because identifying malware families can help analysts understand threats and support appropriate response. This literature survey reviews sixteen studies addressing malware classification and detection through image-based learning, one-dimensional signal representations, feature selection, memory-dump analysis, multimodal fusion, hybrid deep learning, and Android-focused methods. Particular attention is given to binary-to-image conversion and convolutional neural networks (CNNs), which learn visual patterns from executable bytes without requiring conventional disassembly. The reviewed studies report promising results, including accuracy above 90% in several image-based settings and 99.05% accuracy for an epoch-selection ensemble on the Malimg dataset. However, reported results are not directly comparable because studies use different datasets, class definitions, representations, and evaluation protocols. Recurring challenges include dataset bias, generalization to unseen malware, class imbalance, information loss during representation and resizing, model complexity, and the trade-off between static analysis and execution-dependent evidence. The survey identifies the need for careful evaluation of binary-to-image preprocessing, lightweight CNN architectures, reproducible comparisons, and validation across datasets. These findings provide a basis for the project's static, image-based malware-family classification direction while leaving final model configuration to empirical evaluation.

Keywords: Malware Classification, Binary-to-Image Conversion, Convolutional Neural Network, Static Analysis, Malware Families, Deep Learning, Cybersecurity

I. INTRODUCTION

Malware continues to pose a significant challenge to computer and network security. Different malware families can share code, conceal their behavior through obfuscation, or evolve over time, making reliable identification difficult. Malware detection determines whether a sample is malicious, whereas malware-family classification attempts to assign a malicious sample to a more specific family or category. Family-level identification can provide useful context for threat analysis, incident response, and the study of malware relationships.

Machine-learning methods have been applied to malware analysis using a range of representations, including handcrafted attributes, API and DLL behavior, memory-dump features, system-call sequences, network observations, and executable-file bytes. Conventional feature-based methods can provide useful classification performance but depend on the availability and quality of extracted attributes [1]–[3]. Dynamic methods can reveal behavior that is not apparent from a file alone, although they require an execution environment and can introduce additional processing and infrastructure costs [4]–[6].

An alternative is to represent binary data as images and use convolutional neural networks to learn discriminative visual patterns. In this approach, bytes are mapped to pixel intensities and arranged into a two-dimensional grayscale image. CNNs can then learn local patterns and hierarchical features without requiring manual reverse engineering of every sample. Several studies have investigated image-based CNN classification, optimization, ensembles, metadata-derived images, and multimodal representations [7]–[12].

The image-based approach is not without limitations. The conversion from a one-dimensional byte sequence to a two-dimensional image can introduce artificial spatial relationships, and resizing may remove information [13]. Performance on a particular dataset may not translate to other malware distributions, and accuracy alone may conceal poor performance on minority classes. This survey synthesizes sixteen studies to identify the methods explored, reported findings, limitations, and research gaps relevant to a static binary-to-image malware-family classifier.

II. REVIEW METHODOLOGY

This survey synthesizes the sixteen papers supplied with the project literature review report. The original full-text papers were used to check study objectives, model designs, datasets, and reported results where these details were available. The corpus includes direct image-based malware-family classifiers as well as related work on alternative binary representations, behavior-based detection, network traffic, memory dumps, compact data design, and Android malware. No new experiments were performed for this survey.

For synthesis, the papers are grouped into five overlapping themes: (1) image-based malware classification; (2)

 This work is licensed under a Creative Commons Attribution 4.0 International License

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.151106

alternative binary representations and feature selection; (3) hybrid and multimodal detection; (4) memory, network, and system-behavior analysis; and (5) Android malware detection and broader machine-learning frameworks. The comparison considers representation, model family, evaluation setting, reported results where available, and limitations relevant to generalization or practical use.

Numerical results are attributed to the individual papers and are reported in the context in which each study evaluated its method. They are not a controlled head-to-head comparison: datasets, class definitions, preprocessing pipelines, split protocols, and evaluation metrics differ across studies. Where the source papers do not provide a directly comparable numerical result in the reviewed material, the discussion remains qualitative.

III. LITERATURE REVIEW AND ANALYSIS

Across the sixteen reviewed studies, the basic process follows a broadly similar pipeline despite differences in data type and model design. First, samples are collected from public benchmark datasets of malware binaries, memory dumps, network traffic, or Android applications, such as Malimg, MalNet, Obfuscated-MalMem2022, N-BaIoT, and IoT-23. Second, each sample is converted into a representation suitable for learning: raw bytes are mapped to grayscale images [7]–[9], [11], PE header metadata is rendered as a compact image [10], binaries are resized into one-dimensional signals [13], structural entropy is combined with images [12], or tabular behavioral, memory, permission, and API-call features are extracted [1], [2], [6], [14]. Third, preprocessing steps such as normalization, resizing, balanced sampling, data augmentation, SMOTE, feature selection, and dimensionality reduction are applied to reduce noise, redundancy, and class imbalance [1]–[3], [7], [11]. Fourth, a classifier is trained on the processed data, ranging from conventional machine-learning models and CNNs to hybrid, ensemble, and recurrent architectures such as CNN-SVM, CNN-KNN, CNN-BiLSTM, and CNN-RNN combinations [4], [6], [8]–[11]. Finally, the trained model is evaluated on held-out or, in some studies, separate datasets using metrics such as accuracy, precision, recall, F1-score, and confusion matrices, with attention to limitations including generalization, computational cost, and dependence on dynamic analysis [7], [11], [15], [16].

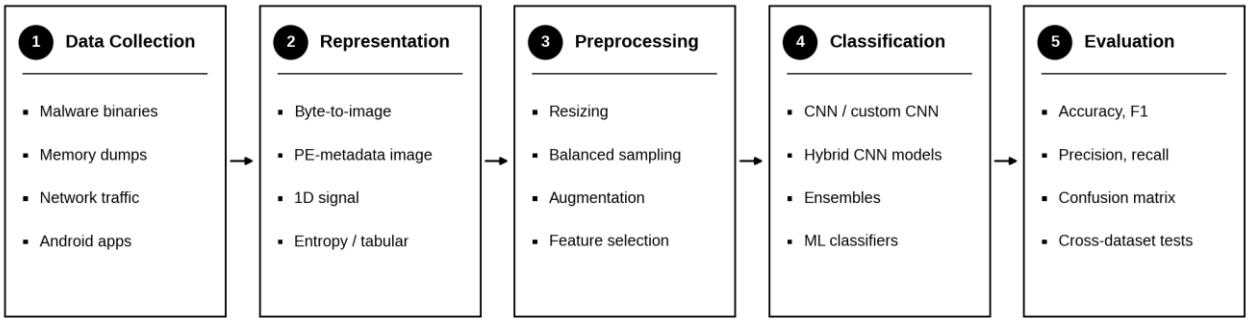


Fig. 1. Basic process followed across the reviewed malware classification studies.

A. IMAGE-BASED MALWARE CLASSIFICATION

Liu et al. proposed the VBDN framework to address class imbalance, overfitting, weak generalization, and processing time in image-based malware detection. Their pipeline combines malware visualization, balanced sampling, data augmentation, and CNN classification. The study reports accuracy above 90% across four open-source datasets, with confusion matrices used to inspect class-level behavior [7]. The multi-dataset evaluation is relevant to generalization, although the individual contribution of preprocessing and augmentation remains an open question.

Duraibi introduced SODCNN-IMC, combining ShuffleNet feature extraction, Snake Optimization for hyperparameter selection, and an attention-based bidirectional LSTM classifier. The study reports a maximum accuracy of 98.42% on Malimg [8]. The result suggests that optimization and sequence-modeling components can improve image-based classification, but the architecture introduces additional complexity. Ablation experiments are needed to establish how much improvement comes from each component.

Musaev et al. proposed an Optimized Epoch Selection Ensemble that combines a custom CNN and fine-tuned MobileNetV2. Candidate epochs are selected based on true-prediction behavior to retain complementary models. The paper reports 99.05% accuracy and F1-score on Malimg, compared with a previously stated result of 98.5% [9]. The method highlights the potential of model diversity, but the use of multiple models increases inference and maintenance costs, and the evaluation is centered on one dataset.

Zhao et al. proposed MCPDS, which generates an Image of PE Metadata (IPM) from the PE header and section table, then combines a shallow CNN with an SVM classifier. On Dataset 2, their PE-metadata representation achieved 0.9580 accuracy and a macro F1-score of 0.9541; average image-generation time was 0.09 seconds [10]. The paper also evaluates inference and resource use on a resource-constrained device. These findings support metadata as a compact



International Journal of Advanced Research in Computer and Communication Engineering
Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026
DOI: 10.17148/IJARCCE.2026.151106

representation, but metadata-only images omit the remainder of the executable and may be affected by changes or manipulation of PE structures.

Andriani et al. presented CNN-AutoMIC, combining CNN feature extraction, autoencoder-based dimensionality reduction, and K-nearest-neighbor classification. The study includes cross-dataset testing and reports strong performance across its evaluated test sets [11]. Its emphasis on cross-dataset evaluation is important because same-dataset accuracy alone can overstate practical robustness. However, the feature-compression and KNN stages add components whose individual effects require further isolation.

Kim et al. investigated an attention-based cross-modal CNN that uses both malware images and structural entropy derived directly from non-disassembled files. The method combines these modalities using cross-modal attention and is evaluated on Microsoft Malware Classification, Malimg, and BODMAS datasets [12]. The study reports that combining modalities outperforms single-modality approaches and simple feature fusion. This supports the value of complementary representations, while also raising questions about preprocessing cost and whether the attention mechanism generalizes consistently across datasets.

B. ALTERNATIVE REPRESENTATIONS AND FEATURE SELECTION

Kale et al. combined multi-objective evolutionary feature selection with conventional classifiers using dynamic-analysis attributes derived from API and DLL behavior. Their feature set was reduced from 335 to 200 attributes, and the report records a peak accuracy of 93.33% [1]. The study illustrates how redundant attributes can affect efficiency, but it does not evaluate binary images or CNN-based visual representations. Its relevance to the present project is therefore methodological rather than a direct baseline.

Wilkie et al. question the common practice of converting malware binaries into two-dimensional images, arguing that reshaping and resizing can introduce quantisation loss and artificial spatial dependencies. They instead resize binaries as one-dimensional floating-point signals and evaluate a bespoke 1D CNN using a pre-activation ResNet design with squeeze-and-excitation layers. On MalNet, they report F1-scores of 0.874 for binary classification, 0.503 for malware-type classification, and 0.507 for family-level classification [13]. The paper also reports that the signal approach compares favorably with image-based methods in its evaluations. This makes representation choice a central research question; comparisons between 1D and 2D methods should use carefully matched datasets and evaluation protocols.

Kim et al. studied compact data design for machine-learning malware detection. Their reference setting used 34,111 malware samples and 1,942 input features. The combined reduction removed 1,467 features and 16,339 samples while retaining about 99% accuracy in the reported experiments; the paper reports training was roughly twice as fast for the applicable models [3]. The authors also note that combined reduction does not apply equally well to every model. The findings concern feature/sample compacting rather than binary-to-image classification, but reinforce the importance of measuring efficiency alongside predictive performance.

C. MEMORY, NETWORK, AND SYSTEM-BEHAVIOR ANALYSIS

Hossain and Islam studied obfuscated malware in memory dumps using normalization, categorical encoding, SMOTE, statistical feature selection, and ensemble machine learning. The report records accuracy above 99% across the evaluated binary and multiclass scenarios, including experiments on the Obfuscated-MalMem2022 dataset [2]. This indicates that runtime memory artifacts can retain discriminative information even when static characteristics are concealed. However, the approach relies on tabular memory attributes rather than image representations of complete executable binaries.

Arnold et al. transformed network traffic into visual representations and applied CNNs for edge-oriented IoT botnet detection. They reported detection rates of 100% on N-BaIoT and 99.78% on IoT-23, along with 2.4 times greater throughput and a 21.4% reduction in model size relative to the selected DNN baseline. Accuracy fell below 76% when the task was expanded to specific attack classes [15]. The study supports visualization-based learning for cybersecurity but does not establish performance for malware-binary family classification.

Alharbi and Kashyap developed a dashboard-oriented approach that visualizes malware propagation using system metrics and network logs collected in a virtualized environment. Examples included WannaCry, CryptoLocker, Locky, Petya, and NotPetya [5]. The work supports security analysis and decision-making, but its focus is behavioral visualization rather than automated family classification. Execution conditions and workload can influence the evidence observed.

Gutierrez et al. proposed a real-time automatic malware detector combining static features with runtime behavioral features. CNNs support visual/static analysis, while recurrent neural networks process sequential behavior. The report records 98% precision, 97% recall, and an F1-score of 0.975 [4]. The hybrid approach shows the value of combining evidence sources, but dynamic analysis introduces execution and infrastructure requirements that differ from the static-only scope of the present project.

D. ANDROID MALWARE DETECTION AND BROADER FRAMEWORKS

Nasser et al. proposed DL-AMDet, combining a CNN-BiLSTM model for static Android application characteristics

DOI: 10.17148/IJARCCE.2026.151106

with a deep autoencoder for dynamic system-call behavior. The report records 99.935% accuracy on the evaluated datasets [6]. The two-stage design illustrates how static and dynamic evidence can complement one another, but it requires application execution and is tied to Android-specific features.

Smmarwar et al. reviewed Android malware detection frameworks using machine-learning and deep-learning techniques. Their review identifies recurring challenges involving zero-day and obfuscated malware, dataset quality, class imbalance, computational cost, and limited generalization [16]. These concerns also apply more broadly to malware classification, although Android application features cannot be transferred directly to arbitrary executable files.

Rashid et al. presented a hybrid Android malware classification framework using permissions, intents, and API-call information across multiple public datasets. The paper reports 98.2% accuracy and a 7.5% improvement over the cited DeepAMD baseline, with experiments covering 15 malware families and approximately 45,000 applications [14]. The cross-dataset emphasis is useful, but the feature set is platform-specific and does not directly address binary-to-image classification of desktop malware.

E. COMPARATIVE ANALYSIS

Table I summarizes the sixteen studies using the representation or approach, evaluation context, and principal limitation reported in the supplied literature review. Results are included only where the report provides a specific metric; qualitative results are retained where no comparable number is stated.

TABLE I COMPARISON OF REVIEWED MALWARE CLASSIFICATION STUDIES

Study / Approach	Dataset or input	Reported result in source report	Key limitation / gap
Kale et al.; evolutionary feature selection + ML	Dynamic API/DLL features	Peak accuracy 93.33%; 335 to 200 attributes	No image representation; dynamic-analysis dependency
Arnold et al.; traffic visualization + CNN	N-BaIoT, IoT-23	100% and 99.78% detection; 2.4× throughput	Fine-grained attack accuracy below 76%; not binary-family classification
Hossain & Islam; memory features + ensembles	Obfuscated-MalMem2022 memory dumps	Above 99% in evaluated scenarios	Tabular memory features, not binary images
Wilkie et al.; 1D CNN signal representation	MalNet byte sequences	F1: 0.874 binary; 0.503 type; 0.507 family	Questions 2D reshaping; 1D vs 2D needs matched evaluation
Liu et al.; VBDN image-CNN pipeline	Four open-source datasets	Above 90% accuracy across datasets	Individual effects of augmentation and balancing unclear
Duraibi; SODCNN-IMC + Snake Optimization	Malimg	98.42% accuracy	Complex architecture; component contributions need isolation
Musaev et al.; epoch-selection ensemble	Malimg	99.05% accuracy and F1	Centered on Malimg; ensemble increases cost
Zhao et al.; PE metadata image + CNN/SVM	PE header and section metadata	Acc. 0.9580; macro F1 0.9541; image generation 0.09 s	Metadata-only input omits remaining binary content
Andriani et al.; CNN-AutoMIC	MaleVis and cross-dataset tests	Strong cross-dataset performance reported	Multiple stages; broader robustness and ablations needed
Gutierrez et al.; static CNN + recurrent behavior model	Static and runtime features	Precision 98%, recall 97%, F1 0.975	Dynamic analysis increases processing and deployment cost
Kim et al.; cross-modal CNN + attention	Microsoft Malware Classification, Malimg, BODMAS	Fusion outperforms single-modality approaches	Fusion complexity; modality contribution and robustness need study
Alharbi & Kashyap; system/network visualization	Virtualized system metrics and network logs	Visualized multiple malware propagation cases	Visualization, not quantitative family classification
Nasser et al.; DL-AMDet	Android static features + dynamic system calls	99.935% accuracy	Execution-dependent and Android-specific

DOI: 10.17148/IJARCCE.2026.151106

Smmarwar et al.; comprehensive review	Android malware literature	Identifies recurring dataset and generalization issues	Review findings; not a new binary-image classifier
Kim et al.; compact data design + ML	34,111 samples; 1,942 features	~99% accuracy; reduced 1,467 features and 16,339 samples; ~2× faster for applicable models	Reduction effects depend on model; not image-specific
Rashid et al.; hybrid Android DNN	Five public datasets; ~45,000 apps	98.2% accuracy	Platform-specific features; transfer to other malware settings unclear

IV. DISCUSSION

A. REPRESENTATION AND MODEL CHOICE

The reviewed literature does not establish one universally superior representation or architecture. Binary-to-image CNNs provide a visual route to learning patterns from executable bytes, while 1D CNNs preserve the sequential order of bytes [7]–[11], [13]. Metadata-only images reduce representation cost but discard content outside selected PE structures [10]. Feature-based and behavior-based methods can use information not visible in static images, but may require feature extraction, execution, or specialized collection environments [1], [2], [4]–[6]. The appropriate approach therefore depends on the classification target, available data, computational constraints, and intended deployment.

B. DATASET BIAS AND GENERALIZATION

A repeated issue is the gap between high performance on a selected dataset and reliable performance on unseen samples or different malware distributions. Several image-based studies use Malimg, and strong reported results on this dataset are encouraging but do not alone establish generalization [8]–[10]. Cross-dataset evaluation in CNN-AutoMIC and the multi-dataset work of Liu et al. are relevant steps [7], [11]. Differences in family labels, sample provenance, duplicate files, class balance, and preprocessing should be documented when comparing results.

C. CLASS IMBALANCE AND METRIC SELECTION

Accuracy can obscure weaknesses in minority classes, especially when family distributions are imbalanced. The reviewed image-based literature includes balanced sampling and augmentation [7], while other studies report F1-score or macro F1 in addition to accuracy [9], [10], [13]. A robust evaluation should include per-class precision, recall and F1-score, macro-averaged measures, and a confusion matrix. These measures can help distinguish strong overall classification from consistent performance across malware families.

D. COMPLEXITY AND PRACTICAL DEPLOYMENT

Optimization, ensembles, recurrent modules, autoencoders, and cross-modal attention can improve results in particular settings, but each adds computational or implementation cost [8]–[12]. The literature on real-time and hybrid detection also emphasizes the costs of runtime analysis and the need to balance performance with processing requirements [4], [6]. For a static image-based project, a useful design objective is to determine whether architectural complexity produces a repeatable improvement over a well-tuned baseline CNN.

V. RESEARCH GAPS

The reviewed studies reveal several gaps relevant to static malware-family classification using binary-to-image conversion:

1) Representation sensitivity: The conversion of bytes into a two-dimensional image introduces design choices such as width, padding, truncation, and resizing. The 1D CNN study questions whether reshaping preserves useful sequential information [13]. Controlled experiments are needed to determine how image-generation choices affect family-level performance.

2) Limited cross-dataset evidence: High scores reported on individual datasets do not guarantee transfer to different malware sources, collection periods, or family distributions. Cross-dataset validation and clearly documented data splits are necessary to estimate generalization [7]–[11].

3) Unclear component contributions: Some approaches combine feature extraction, optimization, attention, recurrent layers, autoencoders, or ensembles [8]–[12]. Ablation studies are needed to isolate the contribution of each component and assess whether added complexity is justified.

4) Class imbalance and evaluation: Balancing and augmentation are used in image-based classification [7], but reported metrics and class definitions vary. Evaluation should report class-wise and macro-averaged metrics alongside accuracy to expose minority-family errors.

5) Efficiency and reproducibility: Image generation, model training, ensemble inference, and multimodal

 This work is licensed under a Creative Commons Attribution 4.0 International License

DOI: 10.17148/IJARCCE.2026.151106

preprocessing have different computational costs. Papers should report preprocessing time, training cost, inference time, and model size where possible, using consistent experimental conditions.

6) Static-only scope: Hybrid studies demonstrate the potential value of runtime behavior [4], [6], but dynamic analysis introduces execution dependencies and additional infrastructure. A static-only approach should explicitly assess what can be achieved from binary-derived images and acknowledge that runtime behavior is outside its scope.

VI. PROPOSED RESEARCH DIRECTION

Based on the reviewed literature, the project is directed toward static, image-based malware-family classification. The intended pipeline converts raw binary bytes into grayscale images and uses a CNN to learn visual patterns associated with malware families. This direction aligns with studies that classify malware images directly [7]–[12], while avoiding the execution stage required by dynamic-analysis methods [1], [2], [4]–[6].

The final CNN architecture and preprocessing configuration should be selected through experiments rather than assumed in advance. A suitable evaluation plan would establish a reproducible baseline, document the binary-to-image conversion process, and compare candidate CNN configurations under the same data splits. Where feasible, experiments should evaluate image dimensions and preprocessing choices, track class imbalance, and use per-class metrics in addition to overall accuracy. A compact architecture should be considered if it offers comparable performance with lower computational cost.

The project's scope is static classification from binary-derived images. It does not require malware execution, sandbox-based behavioral collection, or disassembly as part of the core classification method. This boundary should be stated clearly when comparing the project with research that uses dynamic behavior, memory dumps, Android-specific attributes, or network observations.

VII. FUTURE RESEARCH DIRECTIONS

Future work should evaluate the proposed image-based classifier on the selected dataset using reproducible training and test partitions, with careful checks for data leakage and duplicate samples. Further experiments can compare baseline CNNs with lightweight or transfer-learning architectures and determine whether added optimization or ensemble stages provide measurable gains. Cross-dataset testing, when compatible datasets and labels are available, would provide stronger evidence of generalization than a single-dataset result.

The binary-to-image pipeline should also be evaluated systematically. Possible experiments include comparing image widths, handling of incomplete rows, normalization choices, and resizing strategies, while preserving a clear record of the transformation. Performance should be assessed with accuracy, precision, recall, macro F1-score, confusion matrices, and inference time. Finally, error analysis should examine which families are most frequently confused and whether the confusion is associated with class imbalance, shared visual patterns, or data limitations.

VIII. CONCLUSION

This literature survey examined sixteen studies covering image-based malware classification, alternative binary representations, feature selection, memory and network analysis, hybrid deep learning, and Android malware detection. The reviewed work shows that CNN-based malware visualization is a promising direction, with several studies reporting strong results on their chosen datasets. At the same time, the literature highlights persistent concerns involving representation choices, dataset bias, class imbalance, generalization, model complexity, and the comparability of reported metrics.

For the present project, binary-to-image conversion and CNN-based static malware-family classification provide a focused research direction. The key requirement is not simply to obtain a high accuracy on one dataset, but to establish a transparent and reproducible pipeline and evaluate its strengths and limitations using appropriate metrics. The final model and preprocessing decisions should therefore be determined by controlled experiments grounded in the available data.

REFERENCES

[1] G. Kale, G. E. Bostancı, and F. V. Çelebi, "Evolutionary Feature Selection for Machine Learning Based Malware Classification," Engineering Science and Technology, an International Journal, vol. 56, 101762, 2024. doi: 10.1016/j.jestch.2024.101762.

[2] M. A. Hossain and M. S. Islam, "Enhanced Detection of Obfuscated Malware in Memory Dumps: A Machine Learning Approach for Advanced Cybersecurity," Cybersecurity, vol. 7, article 16, 2024. doi: 10.1186/s42400-024-00205-z.

[3] S.-K. Kim, X. Feng, H. Al Hamadi, E. Damiani, C. Y. Yeun, and S. Nandyala, "Advanced Machine Learning Based Malware Detection Systems," IEEE Access, 2024. doi: 10.1109/ACCESS.2024.3434629.

[4] R. Gutierrez, W. Villegas-Ch., L. Naranjo Godoy, A. Mera-Navarrete, and S. Luján-Mora, "Application of Deep

DOI: 10.17148/IJARCCE.2026.151106

Learning Models for Real-Time Automatic Malware Detection," IEEE Access, 2024. doi: 10.1109/ACCESS.2024.3436588.

[5] F. Alharbi and G. S. Kashyap, "Empowering Network Security through Advanced Analysis of Malware Samples: Leveraging System Metrics and Network Log Data for Informed Decision-Making," International Journal of Networked and Distributed Computing, vol. 12, pp. 250–264, 2024. doi:10.1007/s44227-024-00032-1.

[6] A. R. Nasser, A. M. Hasan, and A. J. Humaidi, "DL-AMDet: Deep Learning-Based Malware Detector for Android," Intelligent Systems with Applications, vol. 21, 200318, 2024. doi: 10.1016/j.iswa.2023.200318.

[7] Y. Liu, H. Fan, J. Zhao, J. Zhang, and X. Yin, "Efficient and Generalized Image-Based CNN Algorithm for Multi-Class Malware Detection," IEEE Access, 2024. doi: 10.1109/ACCESS.2024.3435362.

[8] S. Duraibi, "Enhanced Image-Based Malware Classification Using Snake Optimization Algorithm With Deep Convolutional Neural Network," IEEE Access, 2024. doi: 10.1109/ACCESS.2024.3425593.

[9] A. Musaev, A. Anorboev, and J. M. Youn, "Optimized Epoch Selection Ensemble: Integrating Custom CNN and Fine-Tuned MobileNetV2 for Malimg Dataset Classification," IEEE Access, 2025. doi: 10.1109/ACCESS.2025.3547791.

[10] Y. Zhao, C. Guo, Y. Ping, Y. Chen, Y. Cui, and G. Shen, "MCPDS: Image-Based Malware Classification Method Using PE Metadata Alone," Cybersecurity, vol. 9, article 34, 2026. doi: 10.1186/s42400-025-00455-5.

[11] S. Andriani, S. Galantucci, A. Iannacone, A. Maci, and G. Pirlo, "CNN-AutoMIC: Combining Convolutional Neural Network and Autoencoder to Learn Non-Linear Features for KNN-Based Malware Image Classification," Computers & Security, vol. 156, 104507, 2025. doi: 10.1016/j.cose.2025.104507.

[12] J. Kim, J.-Y. Paik, and E.-S. Cho, "Attention-Based Cross-Modal CNN Using Non-Disassembled Files for Malware Classification," IEEE Access, 2023. doi: 10.1109/ACCESS.2023.3253770.

[13] J. Wilkie, H. Hindy, I. Andonovic, C. Tachtatzis, and R. Atkinson, "Signal-Based Malware Classification Using 1D CNNs," Cybersecurity, vol. 9, article 36, 2026. doi: 10.1186/s42400-025-00454-6.

[14] M. U. Rashid, S. Qureshi, A. Abid, S. S. Alqahtany, A. Alqazzaz, M. ul Hassan, M. S. Al Reshan, and A. Shaikh, "Hybrid Android Malware Detection and Classification Using Deep Neural Networks," International Journal of Computational Intelligence Systems, vol. 18, article 52, 2025. doi: 10.1007/s44196-025-00783-x.

[15] D. Arnold, M. Gromov, and J. Saniie, "Network Traffic Visualization Coupled With Convolutional Neural Networks for Enhanced IoT Botnet Detection," IEEE Access, 2024. doi: 10.1109/ACCESS.2024.3404270.

[16] 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, 100130, 2024. doi: 10.1016/j.teler.2024.100130.