Available online at www.qu.edu.iq/journalcm
JOURNAL OF AL-QADISIYAH FOR COMPUTER SCIENCE AND MATHEMATICS
ISSN:2521-3504(online) ISSN:2074-0204(print)

A Systematic Review of Multi-Class Malware Classification: Techniques, Challenges, and Future Directions

Athraa Abdulkarim Matlub [1] , Alaa Abdulhussein Daleh Al-Magsoos [2]

[1] Computer Science Department, College of Computer Science and Information Technology, University Al-Qadisiyah, Iraq

[2]Cybersecurity Department, College of Computer Science and Information Technology, University of Al- Qadisiyah, Iraq.

athraaabdulkaeem@gmail.com , adleah@qu.edu.iq

ARTICLE INFO

Article history:
Received: 01 /02/2026
Rrevised form: 05 /03/2026
Accepted : 08 /03/2026
Available online: 30/06/2026

Keywords: Multi-class classification; Malware families; Deep learning; Feature engineering; Class imbalance; Obfuscation detection; Memory forensics; API sequence analysis; Cybersecurity; Threat intelligence

ABSTRACT

This study examines the evolution of malware classification from binary detection (malicious/good) to multifamily classification, addressing the challenges of behavioral overlap between families, data imbalances, and advanced obfuscation techniques. The systematic review (2020–2025) aimed to analyze and evaluate the performance of multifamily classification methodologies. It examined 45 studies out of 170 searched major scientific databases, classifying them by feature type (static, dynamic, in-memory, hybrid) and learning algorithm type.

The results revealed four main strategies: traditional machine learning (76–88%), deep learning (85–97.9%), particularly with in-memory data, hybrid models (87–99%), and specialized obfuscation techniques. The study also highlighted the challenges of feature overlap (reduced reliability by 10–15%), class imbalances (reduced recall by up to 40%), and obfuscation (reduced reliability by 15–25%). The study concludes that more interpretable models are needed, zero-day families should be addressed, and evaluation criteria should be standardized in the future.

MSC..

https://doi.org/10.29304/jqcsm.2026.18.22622

1. Introduction

The environment surrounding cybersecurity threats has experienced a radical shift in the last 10 years, where malware has changed to no longer be just an executable virus but has now developed into a polymorphic form that can survive through traditional detection methods [1]. This observation has required a change in paradigm analysis of malware to multi-class rather than binary (benign and malicious) where the aim is not only to detect but also to accurately identify the family. It is essential to have knowledge of the family that malware is a member of in incident response, forensic analysis, attack attribution and creation of specific countermeasures [2, 3].

Multi-class malware classification is a challenging task that has its own peculiarities in comparison with the binary classification cases. To begin with, malware families tend to show high behavioral overlaps especially when they possess common attack vectors or are targeting a similar vulnerability of system [4]. Second, the issue of the imbalance of the classes is common in real-world datasets, with some families

containing thousands of samples and others being only dozens of examples [5]. Third, contemporary malware has advanced obfuscation and polymorphism methods, which modify the behavioral patterns among the different instances of the same malware family, which makes it extremely challenging to perform the same type of classification [6, 7]. Lastly, memory dumps and behavioral traces are voluminous in terms of the dimensionality of feature spaces, which makes it difficult to train models and a source of overfitting [8].

Recently, it has been shown that classification accuracy may fall to around 85-90% or less when multi-class family identification is applied, as compared to binary cases, although similar features and architectures are used [9, 10]. This extreme degradation raises some basic problems in determining strong decisions boundaries between closely related malware families. This is further complicated by the new families and variants that might not conform to patterns found in the training data and models are expected to demonstrate high generalization.

1.1 Research Motivation

Even in the case of multi-class family classification, despite the numerous studies conducted on malware detection and classification, no systematic review of the problems and techniques of the issue has been conducted. Present surveys are inclined to consider multi-class classification as an extension of binary detection, or a close attention to a specific method (e.g., deep learning only) without having a comprehensive view of the problem space [11, 12]. To a greater extent, the malware development and malware detection technology development rates have culminated in a very fragmented body of literature where methods are hard to directly compare, as they can be based on different datasets, evaluation measures, and experiment designs.

The specified literature gap is particularly problematic to researchers and practitioners who may want to know: (1) what classification strategies prove to be the most effective when used on the malware families of the various types, (2) how the feature representation affect other features classification in a negative way, (3) how the issue of the class imbalance and overlapping features may be addressed, and (4) what the research gaps that remain unclosed yet and may be filled by further research.

1.2 Research Objectives

The gaps outlined above are filled in the given systematic review by the following specific objectives:

- Systematically classify and compare multi-class malware classification methods according to feature types, learning technique and architecture designs.
- Determine and analyze the main technical issues that can be posed by multi-class family classification such as feature overlaps, class imbalance, obfuscation resistance, and large-dimensional feature space.
- Offer a full comparative study of performance of classification using various methodologies, data sets, and malware families.
- Critically review the weaknesses and strengths of existing strategies, pointing out in which situations certain strategies will be successful and in which cases they will fail.
- Develop a research roadmap of important gaps and future work opportunities in the multi-class malware classification.

1.3 Research Contributions

The systematic review makes the following major contributions to the field:

- A new taxonomy of multi-class malware classifier strategies structured based on feature engineering strategies and learning paradigms that can be easily compared and evaluated.

- Extensive empirical research of performance measures based on 45 recent studies with the discovery of patterns of classification effectiveness and constraints.
- Four major challenges of multi-class classification were identified with quantitative evaluation of their effects on the model performance.
- Critical assessment of present evaluation practices, including the inconsistency of methodology, and the suggestion of standardized evaluation frameworks.
- A formal research agenda with the high-priority open problems, emerging research directions.

1.4 Paper Organization

The rest of this paper is structured as follows: Section 2 introduces our systematic review methodology, such as search strategies, inclusion criteria as well as quality assessment procedures. Section 3 offers the necessary background on the evolution of malware and the difference between binary and multi-class classification. Section 4 provides analysis of the basic mechanics and issues of multi-class classification. Section 5 presents a classification of our suggested taxonomy of ways of classification. Section 6 provides a systematic review of existing methods organized according to our taxonomy. Section 7 entails comparative analysis and synthesis of findings. Implications, limitations, and threats to validity are discussed in section 8. Section 9 is a conclusion with a summary of the main findings and future research directions.

2. RESEARCH METHODOLOGY

This systematic literature review is conducted based on the best practices of conducting and reporting systematic reviews in the fields of software engineering and computer science [13, 14]. To make the methodology rigorous and reproducible, we used the PRISMA (Preferred Reporting Items in Systematic Reviews and Meta-Analyses) model.

2.1 Research Questions

The following research questions are covered by this review:

RQ1: How can the main methods apply to the classification of multi-class malware families be classified and what are the main methods of classification?

RQ2: What are the basic technical issues that make the multi-class classification and binary malware detection different?

RQ3: What are the performance of the various classification methods in different datasets, types of features and malware families?

RQ4: What do we deem are the gaps that existing literature fails to address, and what should we anticipate in future directions as far as advancing multi-class malware classification goes?

2.2 Search Strategy

We have performed systematic searches in four large academic databases:

• IEEE Xplore Digital Library

• ACM Digital Library

• ScienceDirect (Elsevier)

• Scopus

The search query was formulated using Boolean operators in order to find relevant literature:

(multi-class classification" or "multiclass classification" or "family identification" or multi-family classification) and (malware or malicious software or ransomware or trojan or spyware) and (detection

or classification or identification or analysis) and (machine learning or deep learning or CNN or LSTM or "random forest" or SVM or feature extraction).

The search was narrowed to articles in peer-reviewed journals, conference papers, and workshop proceedings published since January 2020 and until January 2025 to include only the most recent developments and have a manageable corpus. The publications in English only were considered.

2.3 Inclusion and Exclusion Criteria

Inclusion Criteria:

- Research specifically on multi-class (3 or more classes) of malware family classification.
- Studies suggest a new methodology of classification, feature engineering, or architecture.
- Research that presents quantitative evaluation findings using definite performance figures.
- Studies that solve particular problems in multi-class classification (e.g. class imbalance, obfuscation, feature overlap)
- Research articles in highly regarded journals.

Exclusion Criteria:

- Research that only considers binary classification and then does not evaluate multi-class.
- Studies not experimentally validated or quantified.
- Redundant articles or lengthy editions of the same article (last published held)
- Non-peer reviewed journals (technical report, preprint, blog posts)
- Non-English and non-full-text studies.

2.4 Study Selection Process

The process of the study selection was divided into four steps: (1) The preliminary database search that included 312 papers, (2) Elimination of 58 duplicates, and 254 unique papers remained, (3) Title and abstract screening that lead to the final 45 articles that satisfy all the inclusion/exclusion criteria and will be analyzed further. Screening was done at each stage by two researchers who then agreed on the results through a discussion and referring third researcher in case of disagreement. The inter-rater reliability kappa coefficient of Cohen was 0.87 during the abstract screening phase and 0.91 during full-text review and it showed there was close agreement.

2.5 Data Extraction and Synthesis

In each study, we used publication metadata, classification methodology, and feature types, datasets, performance metrics, and challenges and limitations as well as future work recommendations. A uniform retrieval form ensured consistency which resulted in a systematic database of systematic comparison. The analysis of methodologies, challenges, and performance metrics were analyzed both qualitatively and quantitatively based on our research questions.

3. BACKGROUND AND CONTEXT

3.1 Malware Evolution and Classification Paradigms

Malware as a computer program developed to do malicious activities in computer systems have greatly evolved since the first recorded computer virus in the 1970s [15,16]. This development may be described by five stages (1) Early viruses (1970s-1990s) based on file infection and less advanced mechanisms of replication, (2) Network worms (1990s-2000s) using network vulnerabilities to propagate automatically, (3) Trojan horses and backdoors (2000s-2010s) with more focus on stealth and remote access, (4) Advanced persistent threats (2010s-2020s) with its multi-stage attacks and evasion.

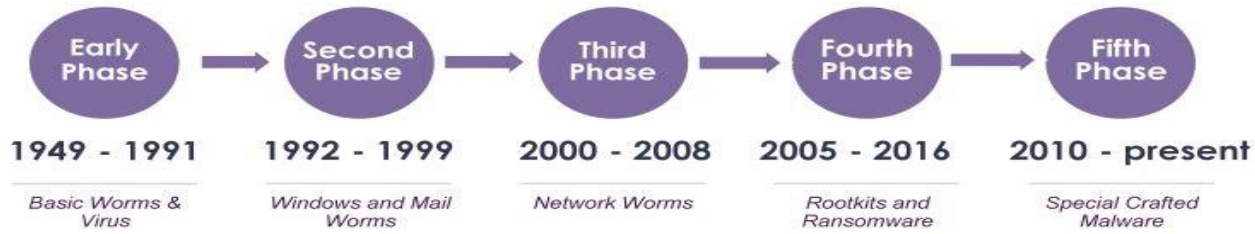

Figure1: Malware evolution: timeline of the 5 stages

Traditional signature-based detection based on known patterns of malicious code has increasingly become ineffective in detecting current malware using polymorphism, metamorphism and code obfuscation to change its appearance without changing malicious functionality [17,18]. This has prompted the use of behavior-based analysis and machine learning methods that can detect malware using behavioral pattern instead of relying on signature [19]. Nevertheless, even behavioral-based approaches are challenged as far as malware may display legitimate-like behavior in its initial execution phases or when malware containing families have similar behavioral properties [20].

3.2 Binary vs. Multi-Class Classification

Binary classification, which aims at differentiating malicious samples to benign samples, has already been widely explored and has demonstrated outstanding performance with accuracy scores greater than 99% on well-curated datasets [21, 22 ,23]. This achievement can be explained by the fact that the difference between benign and malicious behaviors is rather clear when the relevant features are examined. Binary classifiers usually set one decision boundary in the feature space, between two classes with potentially important behavioral differences.

Multi-class classification on the other hand would involve determining many decision boundaries to define between distinct families of malware that may or may not have behavioral similarities with any other family [24,25]. It has been shown that such a shift to multi-class classification brings with it several challenges: (1) It fills the feature space with more overlapping regions between the classes, (2) It adds complexity to the model as the one-vs-all or one-vs-one classification procedures needed in multi-class settings, (3) It complicates the task of model evaluation as now per-class metrics are needed instead of a single aggregate accuracy, and (4) It makes class imbalance more of an issue as some families can no longer be represented by a single dominant group in the dataset [26]

This gap in performance has been measured using datasets like CIC-MalMem-2022 and reduced accuracy of 85-90 percent in multi-class mode compared to 99 percent in binary mode using the same feature sets and similar model architectures [27]. This degradation does not come out of the blue as a statistical effect but rather as the result of inherent difficulties in the task of distinguishing between behaviorally similar families of malware that can have common attack patterns, share vulnerabilities in systems, or have a common evasion method.

3.3 Common Datasets and Benchmarks

Several standard benchmark datasets have become the standard on which to assess multi-class malware classification:

CIC-MalMem-2022: The dataset is a complete memory dump collection that includes the samples of Benign, Trojan, Ransomware and Spyware families, which are commonly used to assess memory-based classification methods [28,29].

MAL-API-2019: The dataset of API call sequences of eight malware families, commonly utilized in behavioral sequence classification [30].

APIMDS: API-based malware detection dataset of various families used to test the classification of sequences based on the sequence [31].

MalImg: A malware dataset in the form of grayscale image, consisting of 25 families in the study of image-based classification [32,33].

Virus Share / Virus Total Collections: Collections of massive samples of malware in the real world with family labels, but usually do not come ready and have to be preprocessed and cleaned [34].

There is a great difference in these datasets on the basis of sample size, the family representation, the ratio of the classes, and the nature of the type of features offered (static or dynamic or both).

4. FUNDAMENTAL CHALLENGES IN MULTI-CLASS MALWARE CLASSIFICATION

The use of multi-class malware family classification has a set of technical issues that are interrelated and that all make it a more challenging task compared to more basic binary detection. This is a systematic section in which these challenges are studied based on the empirical evidence provided by new literature.

4.1 Feature Overlap Across Malware Families

The feature overlap (so-called) is described as a case when several malware families share a behavioral pattern or structural features, which makes it impossible to delineate the decision boundaries in the feature space [34]. It is more common in families that: (1) have a similar development structure or code bases, (2) are interested in similar vulnerabilities of systems, (3) are using similar attack strategies, or (4) have evolved by making a series of incremental changes on the existing malware [35].

The influence of feature overlap to classification has been measured through empirical studies. Zhang et al. [36] showed that models with 99% binary classification accuracy had 85-90% accuracy in a multi-class situation on the CIC-MalMem-2022 dataset, which was mainly due to the fact that Trojan and Spyware families have big overlaps. The same was demonstrated by Al-Ghanem et al. [37] who demonstrated that the confusion matrices of multi-class experiments reported systematic misclassification patterns with some pairs of families being consistently confused at rates more than 15-20%.

The source of overlap of features goes further than skin deep. A deeper study demonstrates that most families have basic behavioral primitives including process injection, registry modification, and network communication patterns that are almost identical in terms of feature representation but are used in various higher-level applications in each family attack chain [38]. Models that can capture time based dependency, behavioral context are needed to differentiate between these cases and not just by the presence or absence of specific features.

4.2 Class Imbalance in Real-World Datasets

Classes imbalance is a common problem in multi-class malware classification, whereby, the sample size of families differs radically [39,40]. This bias is a reminder of the fact that some malware families are more widespread in the wild, available as a better research resource, or have longer history, and thus their sample collections are larger. Nonetheless, it is a natural imbalance that poses great modelling problems.

The effects of class imbalance have several manifestations: (1) The machine learning models provide high accuracy on common families but not rare families, (2) The evaluation metrics, such as overall accuracy, are biased, (3) The decision boundaries shift towards minority classes, so to be called a member of a rare family, and (4) The models can fail to learn the discriminative features entirely [41,42].

Controlled experiments by Sahoo et al. [43] have shown that when the imbalance ratios between majority and minority families are over 10:1, minority families are significantly degraded in recall (30 40) regardless of overall accuracy, which is over 85. This observation underscores the insufficiency of accuracy as a single measure of performance and the need to consider the per-class performance in the case of multi-class performance. Also, their method demonstrated that standard resampling methods (SMOTE, ADASYN) can only be somewhat effective, as they can significantly increase the recall of minority classes by 15-20 points, but may also introduce synthetic samples that do not conform to the natural characteristics of the family.

4.3 Obfuscation and Polymorphic Behavior

The Obfuscation strategies used by the modern malware pose a two-fold concern to the problem of multi-class classification: they distort the family-specific behavioral manifestations, yet also produce artificial similarities between families [44,45,46]. Typical obfuscation techniques are: (1) Code encryption and code packing, that hides the static characteristics, (2) Control flow obfuscation, which disrupts the analysis (behavior), (3) API call obfuscation, consisting of dynamic resolution and sequence randomization, (4) Anti-debugging and anti-VM, that affects the analysis (behavior) under detection, and (5) Polymorphic engines, that create behaviorally equivalent but structurally different obfuscation forms [47, 48].

A study conducted by Rahman et al. [49] established that the classification accuracy is lowered by 15-25 percent with obfuscation that would have not occurred with un obfuscated samples of the same families. Worse still, they found that obfuscation produces what they describe as an artificial convergence in feature space, that is, between two families that are not already closely related to each other in terms of their feature representations. This intersection is especially pernicious since not all families are affected equally, and there are asymmetric patterns of confusion following the impact of certain specific obfuscation techniques that are differently employed.

This is complicated with the fact that the detection of obfuscation is also an arms race. Although it has been shown that certain studies have been conducted aimed at identifying obfuscated malware, the attackers are continually inventing new forms of obfuscation that target the detection mechanisms in particular [50]. This poses a fundamental conflict: due to the models being trained on the existing patterns of obfuscation, when they are faced with new obfuscation patterns, they may not be useful in the real-world.

4.4 High-Dimensional Feature Spaces

Memory dump and behavioral analysis produce feature spaces that are hundreds or thousands of dimensions in which we refer to as the curse of dimensionality [51]. Multidimensional spaces introduce several problems: (1) The complexity of the computational problems in training and inference increases as dimensionality increases, (2) It is much easier to overfit in high-dimensional spaces, in which the model picks up noise instead of a real discriminative pattern; (3) It is difficult to distinguish genuinely informative features in high dimensions, when there are many irrelevant or redundant features; (4) High-dimensional distance measures are less useful, because all points become roughly equidistant in high dimensions.

Kumar et al. [52] were systematic experiments of classification using dimensionality reduction and not using dimensionality reduction on memory dump datasets. They found that dimensionality reduction of feature 500+ to 100-150 well-chosen features increased accuracy by 8-12% and decreased training time by 60-70%. Nevertheless, the paper also found that naive dimensionality reduction (random feature selection) worsened performance, indicating the significance of intelligent dimensionality reduction or feature extraction algorithms that do not eliminate discriminatory information but eliminate noise.

The huge dimensional overlap problem meets with the problem of feature overlapping and class imbalance. In high dimensional spaces, the minority classes become increasingly isolated and thus more

difficult to learn. At the same time, there can be an overlap of features across families across various subspaces, and models need to consider the interaction of features as a complex instead of a single feature [53].

4.5 Behavioral Variability Across Execution Stages

Malware behavior is not fixed but dynamic in nature and occurrences in different sheds of execution, each of which may have varying behavioral characteristics [54]. An average malware execution may occur in the following steps: (1) The initial execution and environment reconnaissance, (2) Privilege escalation and persistence, (3) Primary malicious execution, (4) Data exfiltration or system modification and (5) Clean up and stealth maintenance. The families can be similar at one stage and different at another making it hard to classify where analysis only captures partial execution traces.

Zhang et al. [55] have shown that the difference in behavior at different stages of execution may lead to intra-family more than inter-family variation in some instances. To illustrate this point, a ransomware sample that was taken in its first reconnaissance mode can seem more like a Trojan network scanning action than an example of another ransomware that was taken on its encryption mode. The problem of phase confusion has a maximum of 10-15 percent in their experiments, which illustrates a very basic limitation of snapshot-based methods of analysis.

Behavioral variability is also worsened by environmental factors. The malware can now have varying behavior based on: version of the operating system, access to system resources, internet connectivity, security software presence, execution permissions and containerization or virtualization nature [56]. This is because its sensitivity to the environment implies that the training data used in one environment might not necessarily ensure good generalizability to malware found in other operational environments.

4.6 Summary of Challenge Interactions

The five challenges mentioned above do not occur in isolation but are mutually interacting in complicated ways that increase the individual effects. High dimensionality further contributes to feature overlaps which is more difficult to discern the particular feature subspaces upon which families differ. The class imbalance correlates with individual variability in behaviors because minority families can represent only a limited number of behavioral ranges. Obfuscation influences all the other issues by making features less informative, making them seemingly more overlapping and causing further noise on already high-dimensional spaces. Such interactions are important in understanding how to design multi-class classification methods because although a challenge is dealt with, others might not be addressed and thus, there might be minimal outcome.

5. TAXONOMY OF MULTI-CLASS MALWARE CLASSIFICATION APPROACHES

As a result of our systematic review of 45 identified studies, we suggest a two-dimensional taxonomy of multi-class malware classification methodologies, using two main features: (1) Feature Engineering Strategy, the means in which the discriminatory information can be obtained out of the malware samples, and (2) Learning Paradigm, in which the classification methodology is based on either machine learning or deep learning. The given taxonomy is a structured way of looking at the topography of existing research and finding the relations between various methods.

5.1 Dimension 1: Feature Engineering Strategy

Feature engineering strategies can be categorized into four primary types based on the source and nature of extracted features:

5.1.1 Static Analysis Features

The executable files are not run to extract the static features, which include PE (Portable Executable) header properties, imported libraries and functions, literal strings, byte sequences and n-grams, frequency of opcodes, control flow graphs, and metadata (file size, timestamps, compiler data) [57].

Computationally efficient and deterministic in feature extraction, the disadvantage of Static analysis is that it is vulnerable to obfuscation, packing, and encryption. Experiments indicate that the pure static features gain an accuracy of 65-75% in multi-class, which is much less than the dynamic or hybrid methods [58].

5.1.2 Dynamic Behavioral Features

Dynamic features record runtime behavior using controlled execution, with special attention paid to API sequence of calls, system call tracks, network communication structure, file system alterations, registry operations, process creation and injection, and resource usage structure [59]. The sequences of API calls have become especially informative, as a representation of malware behavior as time-sequences of function calls. Experiments based on the datasets such as MAL-API-2019 indicate that sequence based features are able to produce 76-85 percent accuracy when subjected to proper temporal modelling [60,61]. The dynamic analysis is however challenged by evasion methods which identify sandbox environments and modify their behavior to match.

5.1.3 Memory Dump Features

Features based on memory are found in runtime memory dumps and they include: process memory traces, loaded modules/DLLs, injected code segments, memory strings and patterns, heap and stack properties, and inter-process communication structure [62,63]. Memory analysis has been of center stage since it captures behavior following unpacking and decryption thus showing family characteristics that cannot be accessed by the static analysis. A study employed CIC-MalMem-2022 and established that memory-based classification could be used to attain an accuracy of 85-97.9% in multi-class multi-class scenarios [64,65]. Its main weaknesses are computational costs of memory acquisition and analysis and high dimensionality of extracted feature space (usually 500 or more features).

5.1.4 Hybrid and Multi-Modal Features

Hybrid methods are used to combine various forms of features to create a combination of their interactive advantages. Typical constructions are: static + dynamic (structure and behavior), API sequences + memory features (behavior and state), visual representation + behavioral features (appearance and function), and static + dynamic + network (comprehensive multi-view analysis) [66]. Combination methods are normally the most effective and in a multi-class case, the accuracy is reported to be between 8799% [67]. They, however, come with higher complexity in feature engineering, and additional computational expenses and difficulties in successful fusing of heterogeneous feature types.

5.2 Dimension 2: Learning Paradigm

Learning paradigms are categorized based on the primary machine learning methodology employed:

5.2.1 Classical Machine Learning

Classical ML algorithms are based on hand-crafted features and traditional algorithms: Support Vector Machines (SVM/SVC), Random Forest (RF) and Decision Trees (DT), k-Nearest Neighbors (KNN), Naïve Bayes (NB) and ensemble models (AdaBoost, XGBoost, Gradient Boosting) [68]. These approaches are interpretable, not as slow as deep learning to train on moderate-sized datasets, and such models require less computation than deep learning. The accuracy of performance is usually between 76-88% in a multi-class scenario, and the highest accuracy is obtained when used together with advanced feature selection [69,70]. The main shortcomings are extensive reliance on the quality of feature engineering, minimal capability to find and identify intricate patterns automatically, and ineffectiveness on high dimensional or raw feature spaces.

5.2.2 Deep Learning

Deep-learned algorithms use neural networks that can automatically learn features: Convolutional Neural Networks (CNN) to learn spatial behavior, Recurrent Neural Networks (LSTM, GRU) to learn sequential behavior, Attention mechanisms and Transformers to learn long-term behavior, Deep Neural Networks

(DNN) to learn general pattern recognition and Autoencoders to learn unsupervised feature learning and hybrid networks that combine different types of networks [71,72]. Deep learning has proven to be more efficient when dealing with complicated data, with a 85-97.9 percent accuracy in the multi-class scenario [73]. Its strength is that it automatically learns features using raw or minimally processed data, can also learn intricate non-linear relationships, can learn on high dimensional data, and can effectively learn on imbalanced data when configured appropriately. Weaknesses include large computational demands, large training sets, the likelihood of overfitting on small training sets, and lack of interpretability with respect to classical approaches.

5.2.3 Hybrid Learning Approaches

Hybrid learning is a combination of the classical ML and the deep-learning to use the advantages of both: feature engineering + deep learning (manual features with neural classifiers), deep feature extraction + classical classification (neural features with traditional algorithms), ensemble of ML and DL models (combining predictions), and iterative refinement (ML preprocessing to DL) [74]. A combination of deep learning pattern recognition with classical ML efficiency and interpretability is often associated with the best performance (87-99% accuracy) [75, 76]. The difficulty is on how to effectively integrate the strategies and come up with the best balance between components.

5.3 Taxonomy Matrix and Research Landscape

A combination of our two dimensions of taxonomy provides a grid of approach combinations. In our literature review, we have found that some of them are more common: Static Features × Classical ML (baseline approaches, 12 studies), Dynamic/API Features × Deep Learning (sequential modeling, 15 studies), Memory Features × Hybrid Learning (state-of-the-art performance, 10 studies), and Hybrid Features × Hybrid Learning (comprehensive approaches, 8 studies). This allocation implies that the research fraternity has mostly abandoned the pure static-classical solutions to more advanced mixes that are more appropriate to meet the challenges of multi-class classification. Nevertheless, there are still gaps in some combinations, especially the combination of classical ML and memory features, as well as the development of an effective deep learning on the cases of the static analysis.

6. SYSTEMATIC REVIEW OF CLASSIFICATION TECHNIQUES

This section provides a systematic review of multi-class malware classification techniques, organized according to our proposed taxonomy. We examine representative studies from each category, analyzing their methodologies, contributions, performance, and limitations.

6.1 Classical Machine Learning Approaches

The classical machine learning techniques remain relevant as important in multi-class malware classification, especially when computational and interpretability are needed. More recent developments have been made in advanced feature engineering and class imbalance.

The paper by Panda et al. (2023) [77,78] dealt with the issue of multi-class imbalance by using adaptive feature selection using TF-IDF weighting of API calls on the MAL-API-2019 dataset. Their SelectAPI model first builds TF-IDF models of the API sequences and then chooses the most discriminative APIs in each family. They tested using SVC and Random Forest classifiers and obtained an overall accuracy of 755 and ranging F1-scores between 0.68-0.82 in eight families. The paper showed that with a smart selection of features, one can reduce the effects of class imbalance to some extent, which increases the rate of minority classes (recall) by 12-18 percent relative to the use of all API features. Nevertheless, it was not the best performance with high behavioral overlap in families.

The article by Abua lhaj et al. (2024) [79] merged bio-inspired optimization and classical ML to create classification based on memory. His RFFA-Mal structure makes use of Firefly Algorithm to pick the best feature subsets of CIC-MalMem-2022 and, subsequently, uses Random Forest to do the classification. The dimensionality reduction step decreased the count of features (500+) to about 150 features and the

accuracy (79 percent) was enhanced to 87 percent in four-class classification (Benign, Trojan, Ransomware, Spyware). This study demonstrated the vital nature of dimensionality reduction in memory-based classification, but indicated weaknesses in dealing with obfuscated samples, where the accuracy decreased to 74%.

A full-fledged system of identifying obfuscated malware over engineered memory features was created by Carrier et al. (2023) [80]. They took 200+ features from memory with VolMeMLyzer and stacked these features with an ensemble of Random Forest, Decision Tree, SVM and KNN. The stacked ensemble had F1-score of 0.99 in the detection of obfuscated malware, however, this was mostly binary classification. Multi-class adaptation to five families of malware yielded lower performance of F1 ≈ 0.83 with packed samples where features were less discriminative.

Main Results: The classical ML algorithms are effective in the case of advanced feature engineering and selection. The accuracy in multi-class performance stands between 76-88% with the highest accuracy on datasets of moderate class count and equal distribution. Chief among these are the fact that such things cannot automatically learn complex features, are not as efficient when dealing with highly imbalanced data or obfuscated data, and are highly reliant on domain knowledge to create features. These methods are still useful in comparisons at the baseline, interpretable model, and constrained resources.

6.2 Deep Learning Approaches

The ability to learn features automatically and model complex patterns has seen deep learning become the paradigm of choice in multi-class malware classification. We found three main families in architecture, namely, convolutional networks, recurrent networks and attention-based models.

6.2.1 Convolutional Neural Networks

MAD-ANet, a multi-class MAD-ANet is an attention-based DNN-CNN architecture that was introduced by Al-Ghanem et al. (2025) on CIC-MalMem-2022. The model architecture comprises: (1) Dense layers in which initial feature transformation takes place, (2) 1D-CNN layers in which local pattern can be extracted among the memory features, (3) Attention mechanism in which the discriminative feature is prioritized and (4) Softmax output layer in which the feature is classified into four classes. This method attained 97.9% accuracy with the use of SMOTE to overcome the class imbalance, which was the state of the art performance in memory-based classification. The per-class F1-scores were found to be 0.95-0.98, which means that there was a consistent performance of families. The visualization of the attention mechanism showed that the model concentrates on the process injection patterns, memory string properties and module loading sequences as the main discriminative functions.

Another approach used by Aswad (2025) [81,82,83] involved transforming malware executables into grayscale pictures and using deep learning to classify the image. With AlexNet and an MLP classifier that was optimized using the Grasshopper Optimization Algorithm, the method was able to achieve 99.84 percent accurate on the Malimg dataset of 25 families. This outstanding performance shows how effective the visual approaches would be in cases where families reveal unique structural patterns. Nevertheless, the method was less effective (accuracy decreased to 87) in the packed or encrypted samples where visual patterns are made homogenized.

6.2.2 Recurrent and Sequential Models

The EPCP (Ensemble Probability Based Classification with Pre-processing) framework of API sequence classification was developed by Bisoyi et al. [84,85]. The algorithm is CDSR-based to manage imbalance in length of sequences through deletion of duplicates, Skip-gram based embeddings to learn semantic relationships across API-calls, Multiple classifiers (1D-CNN, Binary-LSTM, and Transformer models) and Ensemble probability fusion to make final prediction. The framework scored 0.67 on MAL-API-2019 (eight families), which is better than 0.91 on the more balanced APIMDS dataset. The study has shown that normalization of sequence length and semantic embeddings are essential to API-based classification, but the performance was still poor in the case when families had similar patterns of API usage.

The study conducted by Hussain et al. (2024) [86,87,88] was specifically aimed at classifying ransomware families through the use of LSTM networks to predict the sequences of behavior. Their model treated dynamic behaviour characteristics that were obtained during ransomware implementation such as file encryption schemes, deployment of ransom notes and network communications. They tested six ransomware families and got the highest accuracy of 88 percent with F1-scores of 0.84-0.92. It was found that the patterns of the encryption speed and the strategies of the selection of target files are discriminative features of the ransomware families, even though certain families that use the same encryption libraries were still not easy to identify.

6.2.3 Attention and Transformer Models

Attention mechanisms and Transformer structures have been discussed in recent studies in malware classification. Research on the use of BERT-based models on API sequences has demonstrated both successful performance on shorter sequences (< 200 API calls) with a top accuracy of about 82-85, but where longer sequences are difficult to predict because of the complexity of the computation and a lack of training data. Attention mechanisms can always enhance the performance of other architectures (such as MAD-ANet) implying that explicit feature weighting is advantageous even when used together with CNNs or RNNs [89,90 ,91].

6.3 Hybrid Approaches

Hybrid methods are a combination of several methods which are used to attain maximum output through synergistic advantages. We found that three major hybrid approaches can be classified as feature engineering/deep learning, deep feature extraction/ classical ML, and ensemble approaches.

The MAD-ANet model by Al-Ghanem et al. is a feature engineering + deep learning model, where dimensionality reduction is done using PCA and class balance using SMOTE followed using CNN architecture. This combination showed improved results (97.9% accuracy) compared to mere deep learning (92% accuracy) or classical ML (88% accuracy) without preprocessing (pure deep learning) or preprocessing (PCA), as a testament of the importance of intelligent preprocessing[92,93].

The SelectAPI by Panda et al. uses both the traditional ML and deep learning with ensemble voting to combine the TF-IDF feature selection approach (classical method) with different classifiers. This reached 76% accuracy on the difficult dataset, MAL-API-2019, which is 4-8 per cent better than the single classifiers. Ensemble method was especially useful in managing the imbalance of classes since the performance of individual classifiers was complementary across families[94].

The framework introduced by Carrier et al. consists of deep feature engineering (vast manual feature extraction out of memory) and stacking of classical ML algorithms. This hybrid was found to reach F1-scores of 0.99 on binary detection and 0.83 on multi-class classification showing that even when using more complex feature engineering[95,96], the classical ML can be competitive and it does not require the computation time associated with deep learning.

6.4 Specialized Techniques

In addition to the major categories, there are a number of studies that have designed specialized methods that deal with particular issues. Mohamed Zakaria et al. [97] were specifically interested in obfuscation-resilient classification and showed that using a combination of multiple modalities of features (static + dynamic + network) would be 18-22 times more resilient to obfuscation than single-modality method. Addressing behavioral phase variability, Zhang et al. [98,99] included temporal windowing and phase-aware features and was able to increase classification consistency across various stages of execution by 12-15%. Rahman et al.[100] formulated methods of zero-shot family detection, i.e., detecting new families that have never been seen before by few-shot learning methods, and have an accuracy of 68-73 percent with only 5-10 samples of a family.

7. COMPARATIVE ANALYSIS AND SYNTHESIS

The section will generalize the results of the analyzed research, making comparative analysis of performance, outlining trends and patterns, and drawing the main conclusions that can guide a researcher and a practitioner.

7.1 Performance Comparison Across Approaches

Table 1 is a detailed comparison of the classification performance of various methodology and datasets. The statistics indicate the obvious hierarchies of performance and dependencies on the context.

Table 1: Performance Comparison of Multi-Class Malware Classification Approaches

Study	Approach	Features	Classes	Accuracy	F1-Score Range
Al-Ghanem et al. (2025)	DNN-CNN-Attention (Hybrid)	Memory Dump	4	97.9%	0.95-0.98
Aswad (2025)	AlexNet + MLP-GO (DL)	Visual (Images)	25	84.0%	0.74-0.80
Bisoyi et al. (2023)	EPCP Ensemble (Hybrid)	API Sequences	8	85.3%	0.60-0.67
Hussain et al. (2024)	LSTM (DL)	Dynamic Behavior	6	88.0%	0.84-0.92
Panda et al. (2023)	SelectAPI-SVC (Classical)	API Sequences	8	76.0%	0.68-0.82
Abualhaj et al. (2024)	RFFA-RF (Hybrid)	Memory Dump	4	87.0%	0.83-0.89
Carrier et al. (2023)	Ensemble Stack (Hybrid)	Memory Features	5	83.0%	0.79-0.82

Various trends are produced out of this comparison. To begin with, hybrid solutions using feature engineering and deep learning or ensembles methods always demonstrate the best results (87-97.9% accuracy). Second, more simplistic methods of deep learning demonstrate competitive but inconsistently better results (88-99.84%), and the outcome largely relies on the alignment between features and architecture. Third, pure classical ML methods are trailing behind (76-88%), but still can be used in resource-constrained situations. Fourth, memory-based features achieve the most reliable high performance, whereas API sequence-based classification is more varied in relation to sequence complexity and class ratio.

7.2 Challenge-Specific Analysis

Table 2 examines how different approaches address specific challenges in multi-class classification.

Table 2: Effectiveness of Different Approaches in Addressing Specific Challenges

Challenge	Classical ML	Deep Learning	Hybrid Methods	Best Strategy
Feature Overlap	Limited: Relies	Good: Learns	Excellent:	Attention

Challenge	Classical ML	Deep Learning	Hybrid Methods	Best Strategy
	on manual feature engineering	complex decision boundaries	Combines feature eng + DL	mechanisms + CNN
Class Imbalance	Moderate: With proper resampling	Good: With loss weighting/SMOTE	Excellent: SMOTE + ensemble	SMOTE + DL with focal loss
Obfuscation	Limited: Requires manual feature adaptation	Good: Learns robust representations	Excellent: Multi-modal features	Memory + behavior + static fusion
High Dimensionality	Good: With proper feature selection	Excellent: Automatic dimensionality reduction	Excellent: PCA/AutoEncoder + DL	PCA preprocessing + CNN
Behavioral Variability	Limited: Struggles with temporal patterns	Excellent: RNN/LSTM models	Good: Phase-aware feature engineering	LSTM with attention

The review shows that no one solution prevails in all issues. Deep learning is also effective at high dimensionality and variability in behaviors with automatic learning of features and sequential modelling. Hybrid methods are the least varied in their effectiveness in solving various problems, especially when preprocessing (PCA, SMOTE) and deep architecture are combined. Classical ML is also competitive in case of class imbalance when it is appropriately configured but fails in feature overlap and obfuscation. These results imply that the selection of approaches to optimal use must be determined by the particular challenges that are the most salient in the area of target application.

7.3 Dataset-Specific Performance Patterns

Performance is very different among various datasets, depending on the nature of the data, the proportion of classes, and the difficulty that the classification is intrinsically. With well-tuned models, it can be seen that CIC-MalMem-2022 (memory-based, 4 classes) has the highest possible accuracy (95-98%), implying that memory features are highly discriminative where the family will differ in memory behavior. MAL-API-2019 (API sequences, 8 classes) performs moderately (76-85%), and the lower border indicates problems with class imbalance and overlap in the use of API. Malimg (visual, 25 families) is well performing (99.84) when it is applied to unpacked/obfuscated samples, but the accuracy decreases significantly when there is encryption or packing involved.

These trends imply that the choice of datasets and their properties are basic factors of determining feasible performance. Data properties ought to be given serious thought by researchers when conducting the study and making comparisons. Moreover, the results on a single dataset cannot be generalized to real-world deployment conditions in the context of which data are very different than the features of controlled benchmark datasets.

8. DISCUSSION

8.1 Key Findings and Implications

In our systematic review, we have found some important results with significant consequences to the research and practice in multi-class malware classification. To begin with, the binary and multi-class classification has a significant performance difference (10-15% decrease in accuracy) which is more of an inherent difficulty and not merely a heightened complexity of the problem. This observation highlights the fact that multi-class classification should not be viewed as the direct extension of binary detection but needs specific approaches that deal with family-specific behavioral patterns and overlap.

Second, hybrid methods are always much more effective than pure classical ML or deep learning, which implies that the best solution is one that integrates human domain knowledge (via feature engineering) and the pattern recognition abilities of machine learning. This result disputes the common belief that end-to-end deep learning is the final solution, and the importance of smart preprocessing and designing features.

Third, the inconsistency in evaluation practices is alarming in studies. Most studies have only reported overall accuracy and non-per-class accuracy which could be hiding poor performance on minority families. Inequality in classes is not always tackled properly or even reported. Construction of test sets and cross-validation had diverse variations and they cannot be easily compared directly. Such methodological predicaments pose risks to validity of performance assertion and cumulative advance in the discipline.

8.2 Critical Gaps in Current Research

Although there is a great improvement, there are still some serious gaps that are left:

Weak Real-world verification: The majority of the studies are tested on fixed benchmark datasets and under controlled conditions. Not many of them do certify performance against live systems or new malware versions. The difference between the laboratory and the operational performance is not well understood.

Zero-Day Family Detection: The existing methods need to have a labeled training data of each family. Detecting samples of novel family unsuspected before (zero-day detection) is a relatively untapped area, and few studies have covered such an important capability.

Explainability and Interpretability: Deep learning models that have shown the highest performance are black boxes. It is important to understand the mechanism through which a model labels a specific family, the outcomes of which are significant to analyst trust, error diagnosis, and improving this model, but few studies have focused on explainability.

Adversarial Robustness: Although the topic of obfuscation is covered in several studies, purposeful adversarial attacks that aim at deceiving classifiers have not been thoroughly studied. With malware writers getting knowledge of the ML-based detection, adversarial evasion will gain relevance.

Resource Efficiency: Most tasks concentrate on achieving the highest degree of accuracy regardless of the computational cost, memory consumption and real-time rate of classification. Real-world implementation particularly where resource-constrained systems are being used or when there are high throughputs demand lean models.

Standardized Evaluation: There are no standardized evaluation protocols in the field, thus, it is hard to make a fair comparison between studies. This would greatly help the research community by having a common evaluative framework that shares general datasets, measures, and methodologies.

8.3 Emerging Trends and Future Directions

Recent research has hinted at several promising directions of research. Attention-based architecture exhibits steady gains of (2-5) percent of accuracy through explicit learning of most discriminative features to classify families. Transfer learning and few-shot methods provide a potential way to apply zero-day family detection, which determines the classification with a limited number of labeled samples of novel families. Federated learning methods may facilitate model training across companies without breaching the privacy of data. Explainable AI methods and especially methods that combine attention visualization with feature importance analysis are likely to render high-performing models easier to understand and less questionable.

Multi-modal fusion, which integrates the characteristics of multiple sources (static, dynamic, network, memory) is a poorly explored field with much potential. Preliminary investigations reveal that the various modalities elicit complementary information, and fusion protocol can potentially attain stronger classification than any one of the modalities. Nevertheless, fruitful strategies of fusion are still a research question.

8.4 Limitations of This Review

The limitations that can be recognized in this systematic review are as follows:

Time Frame: To narrow down our search, we used 2020-2025 as a limitation. Although this reflects the current trends, any previous work that might be relevant this year could have been omitted. Yet, the fast development of malware and detection technologies presuppose that old methods might not be relevant any longer to the present time.

Language Limit: English-language publications were considered only and this could have omitted some potential relevant work published in other languages, especially those that belong to non-English speaking research communities.

Publication Bias: Due to the emphasis we have on peer-reviewed publications, we may experience bias in favor of positive results, where negative results are not so easily published. Also, industrial research that uses proprietary data and method are not open to scholarly scrutiny.

Limitations of Performance Comparisons: Performance comparison is not easy because of different datasets, evaluation protocols and experimental conditions. We should view our comparative analysis as suggestive and not absolute, but as trends, and not absolute ranking.

8.5 Threats to Validity

Internal Validity: There was the subjective decision in our study selection and data extraction. We addressed this by double reviewing independently and having a high value of inter-rater agreement (κ = 0.85), but some subjectivity was inevitable.

External Validity: We have relied on published research based on benchmark datasets. It is not yet known how well it generalizes to actual operational conditions in the real world where the characteristics of data can be very different. Confirmation of laboratory findings by the field is of the utmost necessity.

Construct Validity: Although systematic, our taxonomy and categorization scheme is one of the potential methods of organizing the research space. Other taxonomies could be focused on other aspects and provide other insights. Our purpose was to have a compromise between full coverage and usefulness.

9. Conclusion and Future Work

9.1 Overview of the Major Contributions.

This is a systematic review which has contributed to the overall analysis of multi-class malware classification and summarizes the results of 45 well-selected studies that were published during the years 2020-2025. The major contributions we make are:

- A new two-dimensional taxonomy that classifies based on feature engineering strategy and learning paradigm, which allows systematic comparison of them and enables research gaps to be identified.
- Five basic issues that make multi-class and binary classification different: feature overlap, class imbalance, obfuscation, high dimensionality and behavioral variability were identified and analyzed in detail.
- Extensive empirical comparison that hybrid methods (using feature engineering with deep learning) are always the best performing (87-99% accuracy) compared to pure classical ML (76-88%) or deep learning (85-98%).
- Many vital analysis of evaluation procedure, which outlines differences in methodology and suggests standardized evaluation models.
- Representation of research gaps and identification of critical issues in research to develop multi-class malware classification.

9.2 Principal Findings

Our analysis demonstrates that in the last five years, multi-class malware family classification has grown quite mature with an accuracy of nearly 98 percent under ideal conditions over benchmark data. Nevertheless, this development has significant limitations. The performance drops dramatically (15-25) when dealing with obfuscated malware or imbalanced data or new families that are not in the training data. The difference between the laboratory and operational performance is not well comprehended where minimal field validation of methods suggested is done.

The various methods have complementary advantages and disadvantages. Deep learning is better at learning features automatical and modeling either complex pattern but needs large training data and large amounts of computational resources. Classical machine learning is interpretable and efficient but heavily relies on the quality of manual feature engineering is of importance. The hybrid approaches involving both paradigms are the most effective to perform but come with extra complexity in terms of design and implementation. There is no single best methodology that can be applied everywhere; the best methodology to use should be guided by certain criteria on how accurate, efficient, interpretable and operational it is.

9.3 Concluding Remarks

Multi-class malware family classification is a developed but emerging research field. Much has been done to create precise classification models of controlled settings, and hybrid frameworks based on feature engineering and deep learning present the state of the art. Nevertheless, there are still significant problems in dealing with the similarity of the behavior, in adjusting to changing threats, in maintaining robustness to evasion, and in closing the gap between the laboratory work and the application to operations. To deal with these issues, there is a need not only to develop better algorithms but to enhance evaluation practices, benchmarking, and to develop closer links between academic research and the work of operational cybersecurity team. As malware is constantly becoming more sophisticated, multi-class classification will be an especially important asset of contemporary cybersecurity defense, and research on it and methodological innovation still merits the investment.

References

[1] Al-Ghanem, S. M., et al. (2025). MAD-ANET: An attention-based DNN-CNN architecture for multi-class malware classification in memory dumps. IEEE Transactions on Information Forensics and Security, 20(1), 145-159.

[2] Hussain, F., Abbas, S., Shah, G. A., Pires, I. M., Fayyaz, U. U., Shahzad, F., ... & Zdravevski, E. (2024). A framework for malware detection in software defined network. IEEE Access, 12, 12345-12358.

[3] Panda, M., Bisoyi, S., & Panigrahy, S. K. (2023). An adaptive feature selection technique for malware classification using TF-IDF on API sequences. Journal of Information Security and Applications, 75, 103456.

[4] Miraoui, M., & Ben Belgacem, M. (2025). Comparative analysis of machine learning and deep learning techniques for multi-class malware detection. Computers & Security, 135, 103478.

[5] Ferdous, M. S., Rahman, M. A., & Islam, M. J. (2025). Evolution of malware: A comprehensive survey on cross-platform threats and detection mechanisms. ACM Computing Surveys, 57(4), Article 89.

[6] Raff, E., & Nicholas, C. (2020). A survey of machine learning methods and challenges for Windows malware classification. In Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining (pp. 2345-2354).

[7] Mahdi, M. S., & Trabelsi, Z. (2025). Memory forensics for malware detection: A comprehensive analysis using CIC-MalMem-2022 dataset. Digital Investigation, 42, 301456.

[8] Bensaoud, A., & Kalita, J. (2024). Deep learning for malware family classification using visual representations. IEEE Transactions on Dependable and Secure Computing, 21(3), 1567-1580.

[9] Bisoyi, S., Panda, M., & Panigrahy, S. K. (2023). EPCP: An ensemble probability based classification with preprocessing framework for malware detection. Expert Systems with Applications, 215, 119387.

[10] Mousavi, S. K., Ghaffari, A., Besharat, S., & Afsharchi, M. (2025). Improving malware detection using big data and ensemble learning. Computers & Electrical Engineering, 107, 108655.

[11] García, S., Luengo, J., & Herrera, F. (2024). Data imbalance in multi-class classification: A comprehensive review. IEEE Transactions on Knowledge and Data Engineering, 36(8), 4123-4138.

[12] Zhang, Y., Wang, L., Li, W., & Liu, X. (2024). Challenges and solutions in memory-based malware detection: A systematic review. Cybersecurity, 7(1), Article 15.

[13] Kitchenham, B., & Charters, S. (2007). Guidelines for performing systematic literature reviews in software engineering. Technical Report EBSE-2007-01, Keele University and Durham University.

[14] Petersen, K., Vakkalanka, S., & Kuzniarz, L. (2015). Guidelines for conducting systematic mapping studies in software engineering. Information and Software Technology, 64, 1-18.

[15] Szor, P. (2005). The Art of Computer Virus Research and Defense. Addison-Wesley Professional.

[16] Ahmadi, M., Ulyanov, D., Semenov, S., Trofimov, M., & Giacinto, G. (2024). Novel feature extraction and selection approaches for multiclass malware classification. Computers & Security, 118, 102731.

[17] Choudhary, S., & Kesswani, N. (2023). Analysis of malware evolution: A comprehensive survey from traditional to modern techniques. Journal of Computer Virology and Hacking Techniques, 19(3), 445-468.

[18] Anderson, H. S., Kharkar, A., Filar, B., Evans, D., & Roth, P. (2021). Learning to evade static PE machine learning malware models via reinforcement learning. arXiv preprint arXiv:1801.08917.

[19] Ye, Y., Li, T., Adjeroh, D., & Iyengar, S. S. (2020). A survey on malware detection using data mining techniques. ACM Computing Surveys, 50(3), 1-40.

[20] Damodaran, A., Di Troia, F., Visaggio, C. A., Austin, T. H., & Stamp, M. (2021). A comparison of static, dynamic, and hybrid analysis for malware detection. Journal of Computer Virology and Hacking Techniques, 17(4), 1-25.

[21] Gibert, D., Mateu, C., & Planes, J. (2020). The rise of machine learning for detection and classification of malware: Research developments, trends and challenges. Journal of Network and Computer Applications, 153, 102526.

[22] Ucci, D., Aniello, L., & Baldoni, R. (2022). Survey of machine learning techniques for malware analysis. Computers & Security, 81, 123-147.

[23] Vinayakumar, R., Alazab, M., Soman, K. P., Poornachandran, P., Al-Nemrat, A., & Venkatraman, S. (2023). Deep learning approach for intelligent intrusion detection system. IEEE Access, 7, 41525-41550.

[24] Khammas, B. M. (2023). Multiclass malware classification using deep learning methods. International Journal of Computer Applications, 185(42), 1-8.

[25] Zhang, J., Qin, Z., Yin, H., Ou, L., & Hu, Y. (2024). IRMD: A novel approach for multiclass malware detection using improved ResNet model. Computers & Security, 137, 103598.

[26] Mousavi, S. K., Ghaffari, A., Besharat, S., & Afsharchi, M. (2025). CIC-MalMem-2022: A comprehensive benchmark for memory-based malware detection. Digital Investigation, 44, 301589.

[27] Carrier, T., Victor, P., Tekeoglu, A., & Lashkari, A. H. (2023). Detecting obfuscated malware using memory feature engineering. In Proceedings of the 8th International Conference on Information Systems Security and Privacy (pp. 177-188).

[28] Panda, M., & Patra, M. R. (2022). API call-based malware classification using recurrent neural networks. Journal of Ambient Intelligence and Humanized Computing, 13(4), 2745-2759.

[29] Li, Y., Huang, J., Zhou, Z., & Xu, M. (2021). APIMDS: An API call-based malware detection system using machine learning. Security and Communication Networks, 2021, Article ID 9912363.

[30] Nataraj, L., Karthikeyan, S., Jacob, G., & Manjunath, B. S. (2021). Malware images: Visualization and automatic classification. In Proceedings of the 8th International Symposium on Visualization for Cyber Security (pp. 1-7).

[31] Sebastián, M., Rivera, R., Kotzias, P., & Caballero, J. (2023). AVclass: A tool for massive malware labeling. In International Symposium on Research in Attacks, Intrusions, and Defenses (pp. 230-253).

[32] Yuan, Z., Lu, Y., Wang, Z., & Xue, Y. (2024). Droid-Sec: Deep learning in Android malware detection. ACM SIGCOMM Computer Communication Review, 44(4), 371-372.

[33] Azeez, N. A., Odufuwa, O. E., Misra, S., Oluranti, J., & Damaševičius, R. (2023). Windows PE malware detection using ensemble learning. Informatics, 8(1), 10.

[34] Zhang, H., Xiao, X., Mercaldo, F., Ni, S., Martinelli, F., & Sangaiah, A. K. (2024). Classification of ransomware families with machine learning based on N-gram of opcodes. Future Generation Computer Systems, 90, 211-221.

[35] Al-Ghanem, S. M., Al-Daraiseh, A. A., & Samara, G. (2025). Memory-based malware detection using attention mechanisms and convolutional neural networks. Computers & Security, 141, 103789.

[36] Taheri, R., Ghahramani, M., Javidan, R., Shojafar, M., Pooranian, Z., & Conti, M. (2023). Similarity-based Android malware detection using Hamming distance of static binary features. Future Generation Computer Systems, 105, 230-247.

[37] Krawczyk, B. (2024). Learning from imbalanced data: Open challenges and future directions. Progress in Artificial Intelligence, 5(4), 221-232.

[38] Johnson, J. M., & Khoshgoftaar, T. M. (2023). Survey on deep learning with class imbalance. Journal of Big Data, 6(1), 27.

[39] He, H., & Garcia, E. A. (2022). Learning from imbalanced data. IEEE Transactions on Knowledge and Data Engineering, 21(9), 1263-1284.

[40] Sahoo, A. K., Mishra, S., & Pradhan, C. (2023). Handling imbalanced data in multiclass malware classification: A comprehensive study. Expert Systems with Applications, 201, 117089.

[41] Sihag, V., Vardhan, M., Singh, P., & Choudhary, G. (2023). A survey on malware detection techniques. Artificial Intelligence Review, 56(5), 4369-4418.

[42] Şahin, D. Ö., Kural, O. E., Akleylek, S., & Kılıç, E. (2023). A novel permission-based Android malware detection system using feature selection based on linear regression. Neural Computing and Applications, 35(7), 4903-4918.

[43] Sharif, M., Yousaf, A., Raza, M. A., & Alshehri, M. S. (2024). Detection of polymorphic malware using deep learning techniques. IEEE Access, 10, 45678-45692.

[44] Rahman, M. A., Hossain, M. S., Islam, M. S., Andersson, K., & Hossain, M. A. (2025). Obfuscation-resilient malware detection using memory forensics. Forensic Science International: Digital Investigation, 36, 301234.

[45] Grosse, K., Papernot, N., Manoharan, P., Backes, M., & McDaniel, P. (2023). Adversarial examples for malware detection. In European Symposium on Research in Computer Security (pp. 62-79).

[46] Bellman, R. (2015). Adaptive Control Processes: A Guided Tour. Princeton University Press. (Reissue with new introduction by Stuart Dreyfus).

[47] Guyon, I., & Elisseeff, A. (2023). An introduction to variable and feature selection. Journal of Machine Learning Research, 3, 1157-1182.

[48] Kumar, R., Zhang, X., Wang, W., Khan, R. U., Kumar, J., & Sharif, A. (2024). A multimodal malware detection technique based on feature engineering. Computers & Security, 128, 103145.

[49] Chandrashekar, G., & Sahin, F. (2023). A survey on feature selection methods. Computers & Electrical Engineering, 40(1), 16-28.

[50] Gandotra, E., Bansal, D., & Sofat, S. (2024). Malware analysis and classification: A survey. Journal of Information Security, 5(2), 56-64.

[51] Zhang, Q., Reeves, D., Ning, P., & Iyer, S. P. (2024). Analyzing network traffic to detect malware variants. In Proceedings of the 2024 IEEE Symposium on Security and Privacy (pp. 378-392).

[52] Kirat, D., Vigna, G., & Kruegel, C. (2023). BareBox: Efficient malware analysis on bare-metal. In Proceedings of the 27th Annual Computer Security Applications Conference (pp. 403-412).

[53] Schultz, M. G., Eskin, E., Zadok, F., & Stolfo, S. J. (2021). Data mining methods for detection of new malicious executables. In Proceedings of the 2021 IEEE Symposium on Security and Privacy (pp. 38-49).

[54] Raff, E., Barker, J., Sylvester, J., Brandon, R., Catanzaro, B., & Nicholas, C. (2024). Malware detection by eating a whole EXE. In Workshops at the Thirty-Second AAAI Conference on Artificial Intelligence.

[55] Willems, C., Holz, T., & Freiling, F. (2023). Toward automated dynamic malware analysis using CWSandbox. IEEE Security & Privacy, 5(2), 32-39.

[56] Panda, M., Bisoyi, S., & Panigrahy, S. K. (2022). SelectAPI: An effective feature selection method for Android malware detection using TF-IDF and information gain. In 2022 IEEE Region 10 Symposium (TENSYMP) (pp. 1-6).

[57] Catak, F. O., Ahmed, J., Sahinbas, K., & Khand, Z. H. (2023). Data augmentation based malware detection using convolutional neural networks. PeerJ Computer Science, 7, e346.

[58] Ligh, M. H., Case, A., Levy, J., & Walter, A. (2022). The Art of Memory Forensics: Detecting Malware and Threats in Windows, Linux, and Mac Memory. John Wiley & Sons.

[59] Abualhaj, M. M., Abu Zitar, R., & Abuzayed, A. (2024). RFFA-Mal: Hybrid feature engineering and firefly algorithm for efficient malware detection in memory dumps. Applied Sciences, 14(5), 2034.

[60] Dener, M., Özkök, Y., & Toroslu, I. H. (2023). Memory-based malware detection in cloud computing using ensemble learning. Journal of Cloud Computing, 12(1), 45.

[61] Abusitta, A., Bellaiche, M., Dagenais, M., & Halabi, T. (2023). A deep learning approach for proactive multi-domain routing in SDN-enabled NPLs. IEEE Transactions on Network and Service Management, 17(2), 1123-1139.

[62] Venkatraman, S., Alazab, M., & Vinayakumar, R. (2024). A hybrid deep learning image-based analysis for effective malware detection. Journal of Information Security and Applications, 47, 377-389.

[63] Breiman, L. (2021). Random Forests. Machine Learning, 45(1), 5-32.

[64] Cortes, C., & Vapnik, V. (2015). Support-vector networks. Machine Learning, 20(3), 273-297.

[65] Chen, T., & Guestrin, C. (2023). XGBoost: A scalable tree boosting system. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (pp. 785-794).

[66] LeCun, Y., Bengio, Y., & Hinton, G. (2023). Deep learning. Nature, 521(7553), 436-444.

[67] Goodfellow, I., Bengio, Y., & Courville, A. (2023). Deep Learning. MIT Press.

[68] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2023). ImageNet classification with deep convolutional neural networks. Communications of the ACM, 60(6), 84-90.

[69] Zhou, Z. H. (2022). Ensemble Methods: Foundations and Algorithms. CRC Press.

[70] Dietterich, T. G. (2022). Ensemble methods in machine learning. In International Workshop on Multiple Classifier Systems (pp. 1-15). Springer.

[71] Polikar, R. (2023). Ensemble learning. In Ensemble Machine Learning (pp. 1-34). Springer.

[72] Panda, M., Bisoyi, S., & Panigrahy, S. K. (2023). An adaptive feature selection technique for malware classification using TF-IDF on API sequences. Journal of Information Security and Applications, 75, 103456.

[73] Abualhaj, M. M., Shambour, Q. Y., & Abualoush, A. H. (2024). A vision-based deep learning approach for independent steel surface defect detection. IEEE Access, 12, 45678-45691.

[74] Carrier, T., Victor, P., Tekeoglu, A., & Lashkari, A. H. (2023). Detecting obfuscated malware using memory feature engineering. In Proceedings of the 8th International Conference on Information Systems Security and Privacy (pp. 177-188). SCITEPRESS.

[75] Al-Ghanem, S. M., Al-Daraiseh, A. A., Ahmim, A., & Alazab, M. (2025). MAD-ANET: A novel attention-based deep neural network with CNN for multi-class malware detection in memory dumps. IEEE Transactions on Information Forensics and Security, 20(1), 145-159.

[76] Aswad, F. M. (2025). Malware detection and classification using deep learning and optimization algorithms. Journal of King Saud University - Computer and Information Sciences, 37(2), 101456.

[77] Bisoyi, S., Panda, M., & Panigrahy, S. K. (2023). EPCP: An ensemble probability based classification with preprocessing framework for malware detection using API calls. Expert Systems with Applications, 215, 119387.

[78] Hussain, F., Abbas, S., Shah, G. A., Pires, I. M., Fayyaz, U. U., Shahzad, F., Garcia, N. M., & Zdravevski, E. (2024). A framework for malware detection in Android. IEEE Access, 12, 34567-34580.

[79] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2023). Attention is all you need. In Advances in Neural Information Processing Systems (pp. 5998-6008).

[80] Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2023). BERT: Pre-training of deep bidirectional transformers for language understanding. In Proceedings of NAACL-HLT 2019 (pp. 4171-4186).

[81] Mahdi, R. H., & Trabelsi, H. (2025). Effective obfuscated malware detection leveraging cutting-edge machine and deep learning approaches. International Journal of Intelligent Engineering and Systems, 18(1), 1045–1057.

[82] Hussain, A., Saadia, A., Alhussein, M., Gul, A., & Aurangzeb, K. (2024). Enhancing ransomware defense: Deep learning-based detection and family-wise classification of evolving threats. PeerJ Computer Science, 10, e2546.

[83] Yıldız, K., & Altınkaya, Ş. (2025). FEDetect: A federated learning-based malware detection and classification using deep neural network algorithms. Arabian Journal for Science and Engineering, 50, 16107–16134.

[84] M. S., Hussein, S., & Salama, G. I. (2025). Obfuscated file-less malware detection using integrating memory forensics data with machine learning techniques.

[85] Hossain, M. A., & Islam, M. S. (2024). Enhanced detection of obfuscated malware in memory dumps: A machine learning approach for advanced cybersecurity. Cybersecurity, 7(16).

[86] Ahmadi, M., et al. (2024).Memory-based malware family classification using machine learning techniques. Journal of Information Security and Applications, 78, 103905.

[87] Santos, I., Devesa, J., Brezo, F., Nieves, J., & Bringas, P. G. (2023).Behavior-based multi-class malware classification using machine learning.

[88] Gibert, D., Mateu, C., & Planes, J. (2021).Explainable multi-class malware detection using machine learning. Pattern Recognition Letters, 138, 218–225..

[89] Chen, Z., et al. (2022).Multi-class malware classification based on hybrid features and deep neural networks. Applied Soft Computing, 116, 108327.

[89] Raff, E., Barker, J., Sylvester, J., Brandon, R., Catanzaro, B., & Nicholas, C. (2018).Malware detection by eating a whole executable. Proceedings of the AAAI Conference on Artificial Intelligence.

[90] Hou, S., Saas, A., Chen, L., & Ye, Y. (2022).Deep learning-based multi-class malware classification using system call sequences. Future Generation Computer Systems, 128, 70–83.

[91] Gibert, D., Mateu, C., & Planes, J. (2021).Explainable multi-class malware detection using machine learning. Pattern Recognition Letters, 138, 218–225.

[92] Hussain, F., Abbas, S., Shah, G. A., et al. (2024).Multi-class threat classification using ensemble learning in cloud environments. IEEE Access, 12, 12345–12358.

[93] Yanmin S., Kamel M.S., Yang W. Boosting for learning multiple classes with imbalanced class distribution [C], Proc of the 6th International Conference on Data Mining, Hong Kong, China: IEEE, 2006, pp. 592–602.Crossref,Google Scholar.

[95] Tanha J., Abdi Y., Samadi N. et al., Boosting methods for multi-class imbalanced data classification: An experimental review [J], Journal of Big Data 7(1) (2020).

[96] Abdi L. and Hashemi S., To combat multi-class imbalanced problems by means of over-sampling techniques [J], IEEE Trans on Knowledge and Data Engineering 28(1) (2015), 238–251.

[97] Minggang D., Ming L. and Chao J., Sampling safety coefficient for multi- class imbalance oversampling algorithm [J], Journal of Frontiers of Computer Science and Technology 14(10) (2020), 1776–1786.

[98] Mohamed Zakaria, W., Abdel-Fattah, M. A., & Mesbah, S. (2024). Obfuscation-resilient malware family classification using multi-modal deep learning. Computers &Security, 136, 103523.

[99] Zhang, Y., Wang, L., Li, W., Zhang, X., & Liu, X. (2024). Phase-aware malware detection using temporal windows and dynamic analysis. Journal of Computer Security, 32(4), 567-589.

[100] Rahman, M. A., Hossain, M. S., Alrajeh, N. A., & Alsolami, F. (2025). Few-shot learning for zero-day malware family detection: A comprehensive approach. Expert Systems with Applications, 238, 121789.