One Size Doesn't Fit All: A Dynamic Heterogeneous Learning Ensemble for Malware Family Classification

Solomon Yekini Sonya[1], Muqi Zou[1], Saastha Vasan[2], Christopher Kruegel[2], Giovanni Vigna[2], and Dongyan Xu[1]

[1] Purdue University
[2] The University of California, Santa Barbara

Abstract. Malware continues to increase in prevalence and sophistication, posing significant challenges to cybersecurity. Leading cyber threat intelligence sources such as AV-TEST and VirusTotal report the discovery of over one million unique malicious files daily. Despite this staggering volume, research shows that the majority of these samples are not fundamentally novel; rather, they are variants of previously observed malware families, often exhibiting shared codebases, behavioral patterns, or structural features. In response, Artificial Intelligence (AI) models are increasingly leveraged to enhance malware classification and remediation efforts. However, while such models trained to classify malware datasets often perform well in controlled environments, research increasingly shows that conventional AI-based malware classifiers struggle to generalize to real-world, highly diverse malware datasets.

We address these limitations by providing three unique contributions to the field of malware family classification. (1) We release a new benchmark dataset called MABEL: ***M***alware ***A***nalysis ***BE***nchmark for AI and Machine ***L***earning. MABEL is a curated dataset containing over 82,000 labeled malware samples spanning 468 families, each described by 600+ structural, behavioral, and metadata features. (2) We introduce a novel heterogeneous ensemble with a dynamic Classification Arbiter agent that leverages the strengths of 61 diverse classifiers to improve accuracy, precision, and generalization. (3) Feedback and granular evaluation of model performance is crucial for explainability and classification optimization. This research provides enhanced classification reporting that identifies which models and features are most effective in classifying specific malware families and highlights areas for targeted model optimization.

To our knowledge, this research represents one of the first to amass such a large, feature-rich dataset with malware attributed to known families and a dynamic heterogeneous ensemble that outperforms existing state-of-the-art models tested on the MABEL dataset. Furthermore, this research introduces an enhanced ensemble paradigm that can be applied to various classification domains.

Keywords: heterogeneous ensemble, malware classification, malware family attribution, malware family classification, classification arbitration, arbiter, dynamic ensemble selection, dynamic inference

1 Introduction

Cybersecurity is increasingly challenged by the rapid advancement and proliferation of malicious software (malware), i.e., software intentionally designed to disrupt, damage, or exploit computers, networks, or digital devices [25,22]. Leading cyber threat intelligence agencies including AV-TEST and VirusTotal report discovering over 1 million unique malicious files daily [4,28,27]. Despite this high volume, research indicates that most new samples are not entirely novel but are variants of previously identified malware families, often sharing code structures, behavioral traits, or other intrinsic characteristics [24,6,10].

Malware family classification, i.e., **malware family attribution**, is the process of assigning samples to known families based on these shared characteristics. Accurate malware family attribution is essential for optimizing remediation strategies, tracking the evolution of threat actors, and deploying targeted defenses against malware campaigns. To support this process at scale, Artificial Intelligence (AI) is increasingly leveraged to enhance malware family attribution. However, while AI-based malware classifiers often achieve strong performance in experimental settings, they often fail to generalize to real-world, highly diverse malware samples [16]. Signature-based detection techniques, dataset generation, and model training remain key challenges for effective real-world malware family attribution.

Signature-based Detection Challenges. The increasing sophistication and volume of new malware samples present a significant challenge to traditional signature and heuristic-based detection methods. Signatures are sets of rules and attributes designed to detect malware [3,23]. While signatures (commonly used by antivirus products) can be effective in detecting known malware samples, they often struggle to generalize and detect new, obfuscated malware samples [7,8,14].

Dataset Challenges. The sheer volume, diversity, and rapid evolution of evasive malware make it difficult for models to consistently learn patterns that generalize well across malware families. Second, obfuscation techniques including packing, encryption, polymorphism, metamorphism, and other evasion techniques significantly increase the complexity of accurate malware family attribution [21]. Third, extracted feature sets are often plagued by sparsity, outdated information, and redundant or noisy attributes, all of which can degrade model performance and hinder generalization [16,17,18].

Modeling Challenges. A common approach in malware classification involves using static, single-model architectures or homogeneous ensembles to classify samples. Such approaches are considered *static* when the prediction strategy remains fixed after training, with no adaptation during inference time based on input context or model confidence. These static approaches often lack adaptability to accommodate the heterogeneity of real-world malware. This rigidity in both representational learning and inference strategies limit model robustness [17,18] and underscores the need for more dynamic, context-aware classification frameworks. These frameworks should incorporate heterogeneous architectures and adaptive inference mechanisms to improve generalization (principles explored in depth throughout this paper.)

Research Contributions. Our research addresses these challenges through the following key contributions. First, we introduce **MABEL** (***M***alware ***A***nalysis ***BE***nchmark for AI and Machine ***L***earning), a curated dataset containing over 82,000 unique malware samples spanning 468 distinct families. Each sample is characterized by 600+ extracted features capturing structural, behavioral, and metadata attributes. Compared to existing datasets, MABEL establishes a new benchmark for real-world malware classification research by offering greater family diversity, deeper feature representation, and improved support for evaluating robust attribution models [19]. Second, we introduce a **dynamic heterogeneous classification ensemble** and a **Classification Arbiter** agent that applies novel arbitration policies and a diverse set of evaluation metrics, including Macro F1-Score, Matthews Correlation Coefficient, and Hamming Loss, to select the most reliable prediction per instance. This dynamic inference strategy capitalizes on the strengths of individual models in the ensemble to improve generalization and robustness across complex malware distributions. Third, we present an enhanced classification reporting system that delivers fine-grained visibility into per-class performance, identifies which models and feature orientations are most effective for each malware family, and flags areas of high uncertainty to guide model optimization.

Results. Our approach demonstrates strong effectiveness in malware family classification, consistently outperforming all baseline models and homogeneous ensembles evaluated on the comprehensive MABEL dataset. We construct a heterogeneous ensemble comprising 61 supervised Machine Learning (ML) models trained from scratch on distinct feature orientations extracted from MABEL, including numeric metadata, disassembly code summaries, API function call sequences, and raw strings. Guided by our conservative ensemble arbitration policy, the Classification Arbiter yields the highest performance of all models evaluated in this research, attaining a Micro F1-Score of **0.9620**, a Weighted F1-Score of **0.9613**, and a Macro F1-Score of **0.7278**. In contrast, ClamAV and the LightGBM performed very poorly to this dataset. ClamAV attained Micro F1-Score of **0.2940**, a Weighted F1-Score of **0.3449**, and a Macro F1-Score of **0.1178** while the LightGBM model trained to the specifications outlined in the EMBER [2] and BODMAS [30] research papers attained Micro F1-Score of **0.2647**, a Weighted F1-Score of **0.2121**, and a Macro F1-Score of **0.0348**. These results highlight the benefits of the feature diversity built into MABEL and the ensemble's ability to deliver precise, balanced predictions across imbalanced class distributions. This affirms the value of dynamic arbitration and multi-modal feature integration for advancing real-world malware family classification.

2 Background

This section provides a brief background regarding feature extraction methods and malware classification techniques relevant to this research.

2.1 Windows Portable Executable Features

Executable files contain well defined sections that organize code, data, and metadata within specific regions. We parse these sections to extract a rich and diverse set of features for ML-based malware classification. Numeric features, sequential Application Programming Interface (API) import function calls, and strings are three primary feature orientations that we use to train our ensemble. Figure 1 depicts an overview of the Windows Portable Executable (PE) file and highlights where our specific feature orientations are derived.

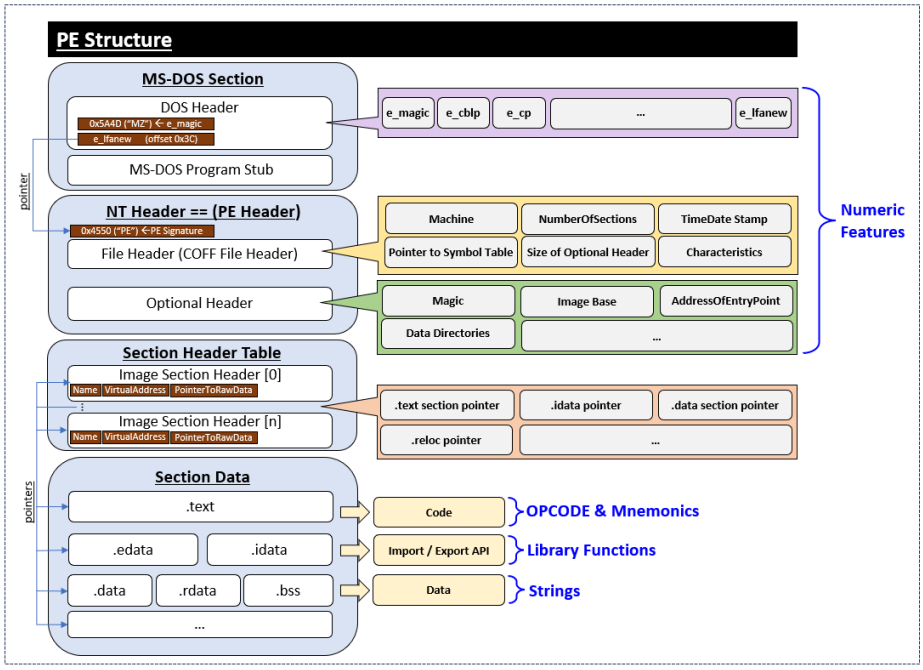


Fig. 1. PE File Format Example [9,31]

2.2 Ensemble Prediction Methods

Ensemble classifiers combine the predictions of multiple models to improve classification accuracy, robustness, and generalization [20]. This section provides a brief background regarding ensemble classification strategies.

Static ensemble methods such as hard voting, soft voting, and weighted voting aggregate predictions across models using majority-vote class selection, averaged probabilities, or performance-weighted influence, respectively [20]. Other static strategies include argmax / argmin and model selection, where an extremal score

or a single model dictates the final prediction [5]. Bagging ensemble techniques reduce variance by training models on bootstrapped subsets and aggregating their outputs. Boosting ensemble techniques, on the other hand, sequentially build a strong learner by focusing on correcting the errors of prior models, thereby reducing both bias and variance over successive iterations. Stacking ensemble techniques train a meta-model that integrates the predictions of multiple base models, allowing the ensemble to capture relationships among model outputs, often yielding greater generalization performance compared to individual models or more simple ensemble strategies [13]. While these approaches offer improved robustness, they remain static at inference time and do not adjust predictions per instance.

Dynamic ensemble selection strategies aim to improve classification performance by adapting model selection at inference time, addressing the limitations of static methods that apply the same models uniformly. Two key techniques are Dynamic Classifier Selection (DCS), which selects the prediction from a single most competent model per instance, and Dynamic Ensemble Selection (DES), which selects a subset of models to jointly predict based on instance-specific evaluation metrics [12]. Our proposed heterogeneous ensemble and Classification Arbiter (detailed later in this paper) are instantiated from this dynamic paradigm.

3 Methodology

We introduce a novel framework that unifies the construction of a large real malware dataset, the development of a dynamic heterogeneous classification ensemble, and the design of a **Classification Arbiter**, culminating in an enhanced classification report. Figure 2 illustrates the end-to-end workflow corresponding to these three core components of our solution. The following subsections outline the methodology used to develop our framework, tracing its design from dataset construction to ensemble modeling and dynamic inference.

3.1 Creation of Malware Dataset (MABEL)

As highlighted in prior studies, a major challenge in advancing malware family attribution models is the limited availability and relevance of existing malware datasets. In this research, we introduce a comprehensive and updated dataset that better captures the complexity and variety of modern, real-world malware.

Amass Malware Corpus. Developing malware datasets like MABEL begins with curating a comprehensive corpus of malware samples suitable for large-scale analysis. Vx-Underground [29], Malware Bazaar [15], and VirusShare [26] are top repositories providing large, up-to-date collections of malware for research and analysis. For the purposes of this research, we exclusively select confirmed malware samples attributed to malware families from Vx-Underground to construct this first release of MABEL. This corpus contains verified malware family

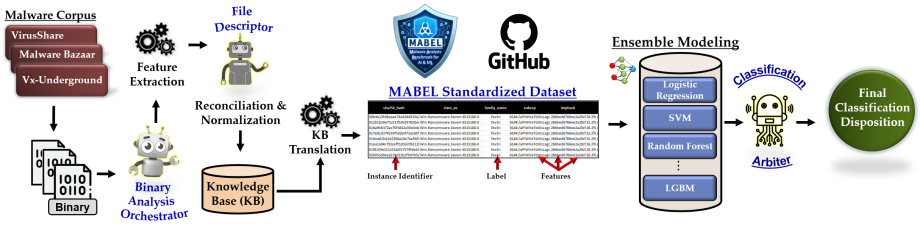


Fig. 2. MABEL Dataset Creation and Modeling Workflow

ground-truth labels, which we annotate as **family_name** within the MABEL dataset.

Automate Malware Analysis and Feature Extraction. Once the corpus is populated with malware, we develop an autonomous, multi-threaded agent called the **Binary Analysis Orchestrator (BAO)** to systematically analyze each binary, extract diverse features, and organize all raw artifacts into a centralized **Knowledge Base (KB)**. Notably, BAO is designed to process both malicious and benign software, enabling the creation of datasets suitable for a variety of classification tasks including binary classification (e.g., *malicious vs. benign*), multi-class classification (e.g., *malware family attribution*), and multi-label classification (e.g., *malware characterization*).

MABEL Dataset Construction. The **File Descriptor (FD)** agent plays a central role in constructing the MABEL dataset. It extracts, standardizes, and enriches features from artifact files stored within the KB. In addition to feature extraction, the FD agent performs deep inspection of each sample's disassembly code to reconstruct program flow and generate structural artifacts, which are incorporated into MABEL's feature representation. We publicly release the MABEL [19] dataset through our file-sharing repositories, ensuring the community has access to an up-to-date, feature-rich malware dataset designed for AI-modeling. With the dataset established, the next step of our research is to construct our heterogeneous ensemble to classify malware samples into their respective families using the diverse features present in the MABEL dataset.

3.2 Developing the Heterogeneous Classification Ensemble.

State-of-the-art (SOTA) and traditional approaches to malware classification often rely on a single model or homogeneous ensemble to make final predictions. We find a key limitation of this strategy is its inability to consistently classify highly diverse, real-world datasets with high accuracy across the majority of labels in a dataset. This limitation motivates the development of our **heterogeneous classification ensemble**, which integrates multiple learning models trained on diverse feature orientations to enhance classification accuracy, robustness, and generalization.

Heterogeneous Ensemble Construction. Some malware families are best identified using discrete feature orientations. For instance, certain samples are most effectively classified using numeric attributes and disassembly code summaries, while other families are more accurately identified via their sequence of API import function calls or unique string patterns extracted from the binaries. To robustly cover this diversity, we train a dedicated group of models on each feature orientation, ensuring that classification is optimized for the unique characteristics of each representation.

As of this writing, the ensemble consists of **61 trained models**, with approximately 20 models dedicated to each of the three feature orientations. The core of our heterogeneous ensemble includes a diverse set of the following classifiers: Logistic Regression, Perceptron and Multilayer Perceptron (MLP), Naive Bayesian Classifier (NBC), Stochastic Gradient Descent (SGD) Classifier in One-vs-Rest (OvR) configuration, Support Vector Classifier (SVC), Bagging Classifiers, Decision Trees, Random Forest, Extra Trees, AdaBoost, Passive-Aggressive Classifier, and Ridge Classifier. Several models are instantiated with varied hyperparameters to enable more specialized learning across the distinct feature types present in MABEL.

The ensemble is intentionally designed to be extensible, supporting seamless integration of new classifiers as malware behaviors evolve and detection methods advance. **Most importantly, this ensemble architecture is *domain-agnostic*, making it transferable to other complex classification tasks beyond malware family attribution.**

3.3 Developing the Classification Arbiter

The **Classification Arbiter** agent is a key component of the heterogeneous ensemble. This arbiter dynamically selects the most accurate prediction for each instance based on per-class evaluations of each model's output. **It is important to note that the arbiter dynamically adapts its selection strategy on a per-instance basis**. In some cases, the most reliable prediction comes from a single model (that may not always be the ensemble's top performer overall) while in other cases, the arbiter's prediction is decided via consensus by selecting the label with the highest agreement across models. This flexibility enables tailored predictions for diverse malware family classes, enhancing attribution accuracy and generalization.

We develop and implement **ensemble arbitration policies** to govern the arbiter's prediction selection process. By incorporating arbitration policies and per-class evaluation metrics such as Macro F1-Score, Precision, Hamming Loss, etc, the arbiter ensures that its final decisions are informed by the strengths of each model and the relevance of feature orientations. We define and evaluate three distinct ensemble arbitration policies: **electoral**, **objective**, and **conservative**; each governed by its own decision logic for resolving conflicting model predictions and guiding the Classification Arbiter's final output.

Electoral Arbitration Policy. This policy implements hard (majority) voting across the entire ensemble to determine each prediction. Figure 6 in Ap-

pendix 1 illustrates the flow and decision rules guiding the arbiter's evaluation process according to this arbitration policy. The strength of this policy lies in its simplicity and resilience to outlier predictions. By leveraging consensus from diverse knowledge representations, this policy minimizes the impact of any single poorly performing model and mitigates localized perturbations and adversarial manipulations targeting classifier inconsistencies. Although this approach may struggle with rare or underrepresented classes, it performs robustly across the majority of well-represented classes.

Objective Arbitration Policy. This policy applies principles of Dynamic Classifier Selection (DCS), where the arbiter dynamically selects a **single** classifier from the ensemble to make the final prediction for each instance. Selection relies on per-class, model-specific evaluation scores e.g., Macro F1-Score, captured during training and validation. This approach prioritizes classifiers with specialized strengths in certain labels, improving class-level precision. As shown in Figure 6 in Appendix 1, the decision logic aligns inference with model-specific expertise, enabling more tailored predictions than global, *one-size-fits-all* strategies. However, performance may degrade if training metrics poorly reflect generalization or when no classifier demonstrates strong competence for a given class.

Conservative Arbitration Policy. This policy leverages principles of Dynamic Ensemble Selection (DES) by combining elements of both electoral and objective strategies with order-of-precedence rules, first evaluating high-confidence predictions and subsequently enforcing ensemble consensus thresholds to prioritize certainty and maximize accuracy. As shown in Figure 7 in Appendix 2, the arbiter integrates performance thresholds and agreement-based heuristics to dynamically mediate between model-specific expertise and ensemble consensus, resulting in more reliable and stable predictions.

Even with dynamic selection policies, certain instances remain challenging for ML-based classification. To address these cases, the conservative policy introduces a user-defined *distrust threshold*, which establishes a minimum acceptable evaluation score (e.g., Macro F1-Score ≥ 0.80) that models must achieve for their predictions to be considered reliable. At inference, if no higher-order heuristic applies and no model exceeds this threshold for its predicted label on a given instance, the arbiter defaults to majority voting while simultaneously generating an alert to flag the prediction as low-confidence. These alerts highlight (1) mislabeled or highly ambiguous samples and (2) potential novel malware that significantly deviates from previously learned patterns, warranting additional investigation and training.

4 Experimental Evaluation

In this section, we describe our experimental setup and rigorously evaluate the performance of our heterogeneous classification framework and its arbitration policies using the MABEL dataset. Models are assessed using a comprehensive

suite of evaluation metrics to enable a multidimensional analysis of classification effectiveness, generalizability, and robustness.

4.1 Dataset

MABEL is constructed from a real-world malware corpus containing 468 unique families. As expected when dealing with real-world malware, the distribution of samples per family is highly imbalanced: some families in MABEL include thousands of unique samples, while others contain three or fewer samples. To permit a stratified training/testing strategy, families with fewer than two unique instances were excluded, resulting in 375 families used for training. The dataset is then partitioned into three distinct subsets according to feature orientation: (1) numeric metadata and opcode statistics, (2) sequential API import function calls, and (3) extracted string features. Each partition is used to train constituent models within the heterogeneous ensemble, enabling model specialization across feature modalities. We implement a 3-fold stratified 80%–15% train–validation split, with 5% of the data reserved as a hold-out test set for final evaluation. Independent groups of models are trained on each partitioned subset, and a master performance report is generated to annotate per-class performance across all models. This master performance report supports the Classification Arbiter in dynamically assessing per-class model strengths and guiding ensemble decision-making. The 5% hold-out test set comprises 3,821 samples randomly selected from the MABEL dataset, ensuring they were excluded from model training. These test instances span 251 unique malware families and are used to assess model performance.

4.2 Optimal NLP N-Gram Specification

We implement a custom Natural Language Processing (NLP) pipeline incorporating tailored preprocessing, stemming, lemmatization, vectorization, and embedding techniques to transform the sequential API function call and string raw textual features for downstream malware family classification. We use N-Grams, specifically unigrams, bigrams, and trigrams, to reduce the feature space of textual data to the most valuable and informative token sequences for training. We conduct a validation study using the Extra Trees classification model on API import function calls and string-based feature representations to identify the optimal N-Gram configuration for training and evaluation. This analysis allowed us to empirically assess the trade-off between model performance and resource consumption as the textual feature space increases. Figure 8 in Appendix 3 illustrates the results of this evaluation. Based on these findings, we selected **20,000 features for N-Gram(1,3)** as the optimal parameters for our NLP vectorization.

4.3 Classification Evaluation Metrics

We evaluate each model's performance using several metrics that capture class balance, confidence calibration, misclassification patterns, and alignment with

ground-truth labels. To identify a representative evaluation metric for each feature orientation, we apply Mean Absolute Correlation (MAC), which measures how consistently each metric correlates with all others across the ensemble [1]. This analysis is conducted at the conclusion of ensemble validation and informs the Classification Arbiter's per-instance model selection by ensuring it relies on the most representative metric for each feature orientation.

Figure 3 presents MAC results for the numeric feature partition. Based on these scores, we group evaluation metrics into performance tiers. In this analysis, Tier 1 metrics include F1-Score, F2-Score, Specificity, False Positive Rate, Jaccard Index, and Kappa Statistic. These metrics demonstrate the highest alignment within this partition and are best suited for guiding arbitration. Metrics in Tiers 2–5, including Precision, False Discovery Rate, and Log Loss, exhibit lower MAC scores, largely due to sensitivity to class imbalance or volatility among rare classes. For this dataset, relying on metrics below Tier 1 would yield less generalizable evaluations and could degrade the Classification Arbiter's decision-making process.

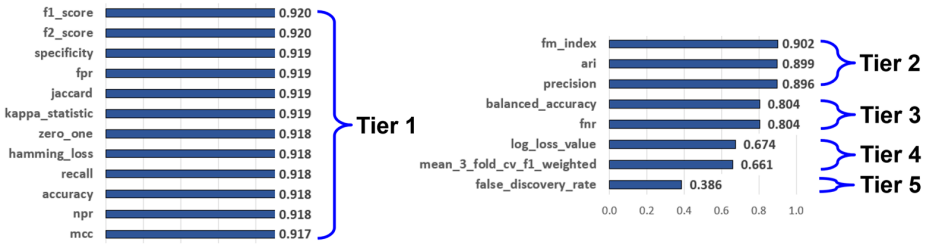


Fig. 3. Mean Absolute Correlation of Evaluation Metrics

We repeat this process across the remaining two feature orientations. F1-Score consistently ranks in Tier 1 as the most representative evaluation metric and is therefore selected as the default evaluation criterion for arbitration. This ensures that the Classification Arbiter is guided by a stable and informative metric when making dynamic per-instance decisions across the heterogeneous ensemble. **MAC is a key process that should be repeated when this framework is applied to different domains to ensure the arbiter uses the most appropriate best evaluation metric for per-class model evaluation and decision-making.**

4.4 Results

We benchmark the performance of our three ensemble arbitration policies—**electoral**, **objective**, and **conservative** against two established baselines: the state-of-the-art EMBER [2]/BODMAS [30] LightGBM malware classifier and ClamAV, a widely used signature-based endpoint detection system.

To evaluate classifier robustness across varying class distributions, we conduct nine experiments by incrementally increasing the minimum number of instances per malware family in the test set (from 1 to 9). This stratified analysis reveals how performance scales with class representation and identifies conditions where classifiers perform best or degrade. We use Macro F1-Score as the primary evaluation metric. Macro F1-Score treats all classes equally, making it especially suitable for imbalanced classification tasks.

Figure 4 reports Macro F1-Score performance across varying class frequency thresholds. The **conservative arbitration policy** consistently outperforms both the LightGBM model trained on MABEL and ClamAV's signature-based detection results.

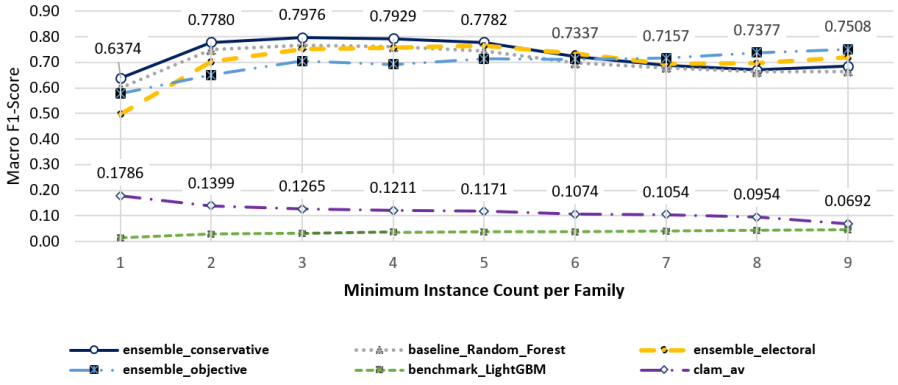


Fig. 4. Overall Model Performance

Table 1 summarizes the average classification performance of each model across Macro, Weighted, and Micro F1-Scores. As revealed earlier, the heterogeneous ensemble implementing the **conservative arbitration policy** achieves the highest scores across all three metrics, with an average Macro F1-Score of 0.73, Weighted F1-Score of 0.96, and Micro F1 of 0.96. This model outperforms all baselines and other ensemble strategies tested on MABEL. Both the **electoral** and **objective** ensemble policies perform competitively, though they fall short of the **conservative model**. Their results reflect the trade-offs between strict consensus-based voting and class-specialized model selection strategies. In contrast, traditional solutions like **ClamAV** and the **LightGBM** benchmarks perform very poorly across all metrics in this dataset, underscoring the limitations of signature-based detection and boosting models in the context of imbalanced, high-variance datasets.

Table 2 concretely shows the number of correct and incorrect classifications across different models. Our top-performing **conservative** arbitration policy highlights the value of an adaptive evaluation and selection strategy where the

Table 1. Mean F1-Score Summary Across Models

Model	Mean Macro F1	Mean Weighted F1	Mean Micro F1
ensemble_conservative	**0.7278**	**0.9613**	**0.9620**
ensemble_electoral	0.7020	0.9251	0.9308
ensemble_objective	0.6948	0.8895	0.9090
clam_av	0.1178	0.3449	0.2940
benchmark_LightGBM	0.0348	0.2121	0.2647

combination of per-class expertise and model consensus results in higher and stable performance on the MABEL dataset.

Table 2. Number of correct/incorrect classifications by policy for 3,821 test set samples

Heterogeneous Ensemble	Correct Classifications	Mis-Classifications	Correct %
Conservative Policy	**3591**	**230**	**93.98%**
Electoral Policy	3444	377	90.14%
Objective Policy	3385	436	88.58%

In summary, these results demonstrate the effectiveness of our heterogeneous ensemble on the MABEL dataset, with the conservative arbitration policy achieving the highest overall performance. More broadly, our findings indicate that dynamic, instance-specific ensemble strategies improve both reliability and generalization, highlighting their value in addressing complex malware classification tasks

4.5 Enhanced Classification Reporting

We develop an enhanced classification reporting system to provide fine-grained insight into model performance across all classes in the dataset. This includes tabular and visual summaries that identify top-performing classifiers per class and highlight which feature orientations are most effective at classifying each malware family. Figure 5 visualizes an example of these per-class performance distributions. This type of enhanced reporting improves interpretability by revealing model specialization across feature orientations.

The top-right chart in this figure shows that the **Babadeda** family is perfectly classified using *strings*, whereas the middle-right chart indicates that the **Icefrog** family is most effectively detected via API function call sequences. Other families are better classified using numeric metadata and opcode statistics as illustrated in the figure. We apply the **distrust metric** (described in 4.5) to flag classes of high uncertainty where model predictions are less reliable. Malware family classes exhibiting low detection confidence like ***dexbia*** (located at the bottom-right of this figure) are marked with an asterisk ('*') to prompt targeted retraining, label refinement, or manual review. Overall, these results

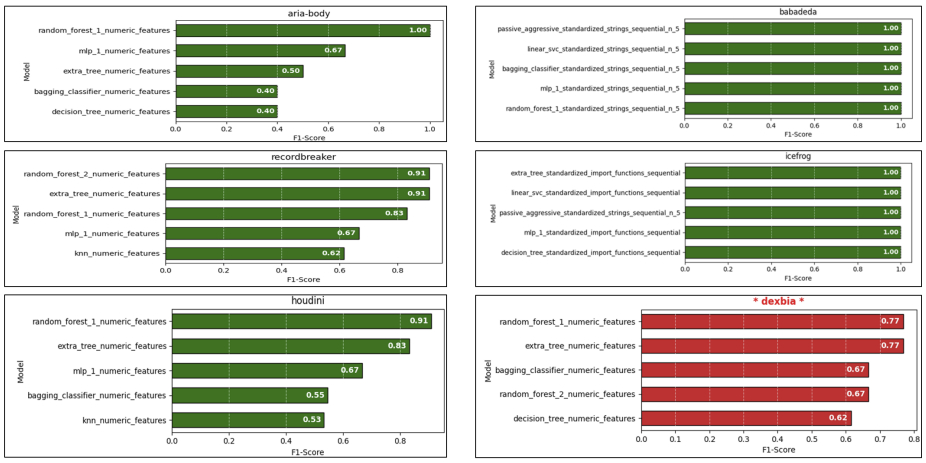


Fig. 5. Enhanced Classification Reporting

demonstrate the complementary strengths of feature-specific models and their collective reliability in robust malware family classification.

4.6 Case Study

We present a representative case study illustrating how the heterogeneous ensemble, guided by the **conservative arbitration policy**, improves classification accuracy through dynamic, instance-specific decision-making. The arbiter not only provides per-instance predictions but also generates interpretability feedback justifying each decision. Figure 9 in Appendix 4 illustrates the arbiter's decision pathway for a specific sample belonging to the *Grimplant* malware family. The figure also presents each model's predicted label and per-class F1-Score corresponding to the predicted label, overall performance average, and ensemble voting proportions—enabling a comprehensive breakdown of the arbiter's rationale and model agreement for the selected instance. We can see that many models incorrectly classify this instance belonging to the Koadic family. Koadic and Grimplant malware families share several functional similarities: both are Windows-based backdoor Remote Access Trojans (RATs) designed to maintain command-and-control (C2) channels, perform data exfiltration, and maintain persistence often through Windows registry modifications. These overlapping capabilities likely contribute to the observed misclassification, as many individual models may conflate behavioral features and structural artifacts common to both families. However, the Classification Arbiter dynamically updates its decision strategy, leveraging per-class performance in this case to correctly identify the instance as belonging to the Grimplant family despite conflicting model predictions.

The arbiter's decision process in this case unfolds in three steps: First, the arbiter initially defaults to the prediction from the top-performing overall model,

Random Forest, which incorrectly classifies the sample as belonging to the *Koadic* malware family. Next, the arbiter assesses the ensemble-wide vote distribution. *Koadic* remains the majority-voted label, receiving 38% of model votes. Finally, the arbiter overrides both the top-model and majority vote prediction by selecting *Grimplant*(the correct ground-truth label) based on the recommendation of a single Logistic Regression model in the ensemble. The arbiter's justification reveals why it diverged from the top overall performing model and consensus. During validation, the Logistic Regression model previously achieved a **perfect F1-Score** when classifying instances belonging to this specific malware family. Consequently, the arbiter correctly selects the prediction from this single model rather than relying on majority voting or the prediction from the single best overall performing model.

Interestingly, although Logistic Regression performs poorly overall on the MABEL dataset (mean F1 ≈ 0.39), the arbiter leveraged this model's per-class strength and determined it exceeded the defined ACCEPTANCE_THRESHOLD for *Grimplant* family prediction. Thus, the arbiter returns Grimplant as the final disposition for this instance. This case exemplifies the value of class-specific performance metrics and dynamic selection demonstrating how even weak global models can contribute critical predictions when appropriately contextualized.

5 Discussion

Malware family classification remains an inherently difficult task due to the large number of families exhibiting overlapping structural and behavioral characteristics, the incompleteness of extracted features (often caused by anti-analysis techniques), and the complexity of real-world multi-class classification where dataset class imbalance and misclassifications between similar families are common and expected. Within this context, we achieve a promising advancement in scalable, multi-class malware family attribution employing a heterogeneous classification ensemble guided by centralized arbitration. The heterogeneous ensemble capitalizes on diverse representational strengths by integrating models trained on complementary feature orientations. Arbitration policies mediate these specializations to produce stable, accurate predictions across the highly imbalanced MABEL malware dataset.

Among the arbitration strategies tested, the **conservative policy** yielded the highest classification performance, offering reliable instance-specific decision-making. Its ability to override majority votes or top-model predictions in favor of historically well-performing, class-specific model recommendations highlights the utility of dynamic ensemble arbitration with evaluation thresholds. This approach proves especially valuable in high-risk settings, such as malware classification, where false negatives and misattributions carry significant operational costs.

We designed the ensemble to be both modular and extensible, enabling broader applicability across domains characterized by class imbalance, noise, and heterogeneous features, e.g., as fraud detection, medical diagnostics, or net-

work intrusion classification. Our architecture supports flexible arbitration policies, validation-informed metric calibration, and dynamic distrust thresholds, fostering interpretability and robust generalization. Recognizing the complexity of coordinating 61 classifiers across diverse feature orientations, we designed a streamlined training and evaluation workflow to ensure usability and facilitate adoption by the research community. This workflow simplifies model retraining and supports the straightforward addition, removal, or updating of constituent models with minimal overhead. Additionally, we will be sharing our pre-trained models for community use. Furthermore, the Classification Arbiter is engineered to seamlessly interface with an arbitrary number of classification models, providing both flexibility and extensibility while preserving a consistent and efficient deployment process.

Moreover, the enhanced classification reporting system provides transparency at the per-class level, revealing which models and feature domains contribute most to performance and highlighting classes where models consistently misclassify instances. The integrated distrust metric helps identify predictions with low confidence, offering actionable insight for targeted retraining, expert review, or label correction. **Taken together, these contributions position our framework as a general-purpose solution for ensemble-based classification in challenging, high-stakes classification domains.** Future work will apply our ensemble to new domains and explore dynamic adjustment of arbitration weights, thresholds, and further automation of trust calibration to adaptively evolve these decision-making strategies.

6 Related Work

The field of malware classification and family attribution has seen significant advancements in recent years, driven by the application of AI, ML, and Deep Learning (DL) models. Several techniques and datasets have been developed to address the growing complexity and diversity of malware. In this section, we discuss two exemplary research publications that include release of new datasets and trained models for malware detection.

BODMAS [30] introduced a large-scale, time-stamped malware dataset comprising 57,293 labeled malware samples across 581 malware families and 77,142 benign Windows PE files. BODMAS was designed to support malware family attribution and temporal drift analysis. BODMAS adopts the same feature schema as EMBER 2018, standardizing cross-dataset experimentation and evaluation. Similar to EMBER 2017-2018 [2], the authors of BODMAS trained Light Gradient Boosted Machine (LGBM) models on the EMBER 2017–2018 dataset and evaluated performance on the BODMAS dataset in order to investigate generalization of the trained models to the BODMAS dataset. Results revealed significant performance degradation—highlighting the impact of concept drift and the limitations of static models.

EMBER 2017-2018 introduced a publicly available dataset containing over 1.5 million Windows PE files for training and evaluating binary classification

ML models [2]. This benchmark provided a standardized dataset that facilitated reproducibility and helped evaluate model performance in the field of malware detection. Building on the EMBER 2017-2018 dataset, EMBER 2024 [11] was recently released and significantly expands the scope and scale of the benchmark. Notably, EMBER's malware family attribution models achieved a **Macro F1-Score of 0.4371**. Although MABEL and EMBER are different datasets, the EMBER evaluation results corroborate the difficulty of real-world malware family attribution yet highlights the potential of our heterogeneous classification ensemble to these highly diverse datasets. Instead of relying on fixed LightGBM classifiers trained on prior datasets, our approach dynamically leverages multiple learning paradigms and diverse feature orientations (numeric, opcode, API call sequences, and strings) to enhance both generalization and instance-level adaptability. This highlights the novelty of our research to the challenges of modern malware family classification.

7 Conclusion

In this work, we introduce the MABEL (***M***alware ***A***nalysis ***BE***nchmark for AI and Machine ***L***earning) dataset. MABEL is an extensive, multi-modal malware benchmark curated to support rigorous evaluation of classification models across malware family attribution tasks. MABEL comprises over 82,000 labeled malware samples spanning hundreds of families and includes a rich set of extracted features across numeric metadata, opcode summaries, API function call sequences, and raw strings. This dataset enables a more realistic and challenging evaluation of classification performance under conditions of feature heterogeneity and class imbalance common in real-world, operational malware detection.

Building on this foundation, we introduce a novel heterogeneous ensemble classification framework coordinated by a centralized **Classification Arbiter**. This arbiter agent dynamically selects the most appropriate prediction per instance using validation-informed performance criteria and one of three ensemble arbitration policies we introduce in this research: *electoral*, *objective*, or *conservative*. Extensive evaluation on the MABEL dataset shows our ensemble consistently outperforms strong baselines, including the EMBER/BODMAS LightGBM classifier and ClamAV, a traditional signature-based detection engine. The **conservative arbitration policy** yields the most reliable and accurate predictions, illustrating the efficacy of dynamic decision-making in ensemble learning.

Our framework contributes a modular and interpretable architecture suitable for other domains that demand instance-specific model selection, such as fraud detection, medical diagnosis, and cybersecurity analytics. Moreover, our inclusion of a distrust metric and enhanced classification reporting equips practitioners with actionable insights for model optimization and human-in-the-loop validation. By uniting model diversity, confidence-aware arbitration, and transparency, this research advances the field of ensemble-based classification and introduces a robust foundation for future adaptive learning systems in adversarial and high-stakes environments.

Appendix 1: Electoral and Objective Arbitration Policies

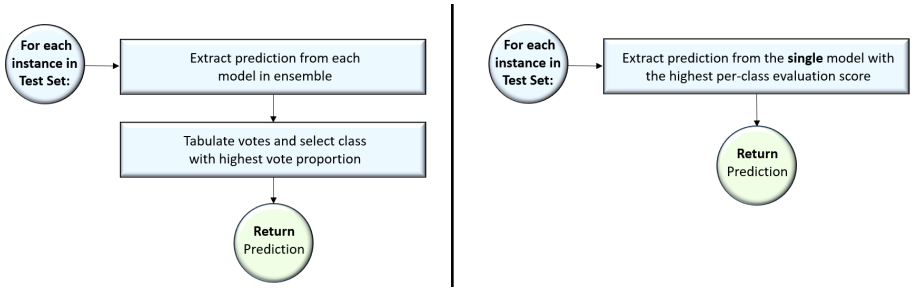


Fig. 6. Electoral (left) and Objective (right) Ensemble Arbitration Policy

Appendix 2: Conservative Ensemble Arbitration Policy

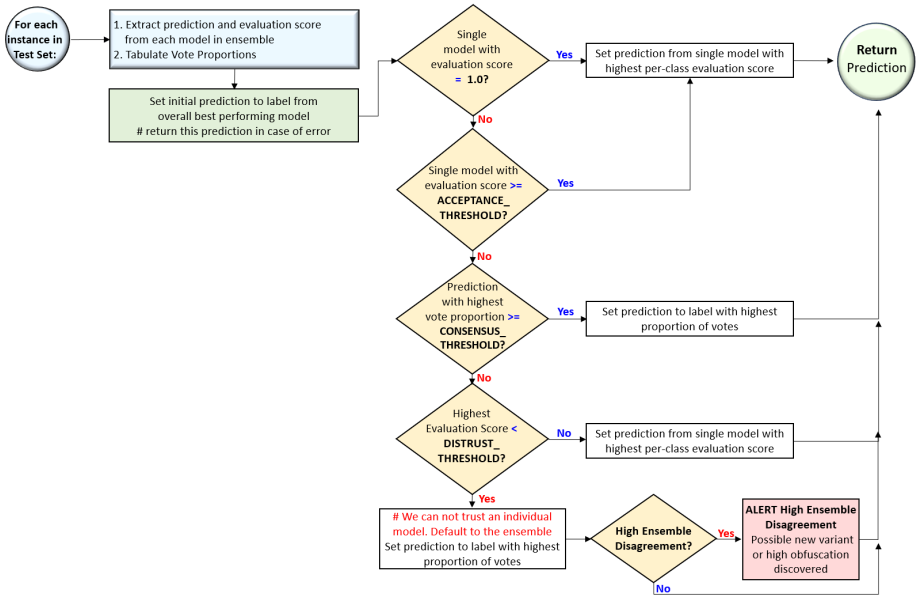


Fig. 7. Conservative Ensemble Arbitration Policy

Appendix 3: NGram(1,3) Hyperparameter Searching

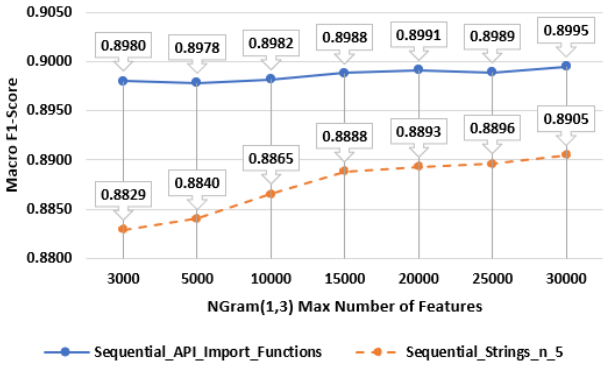
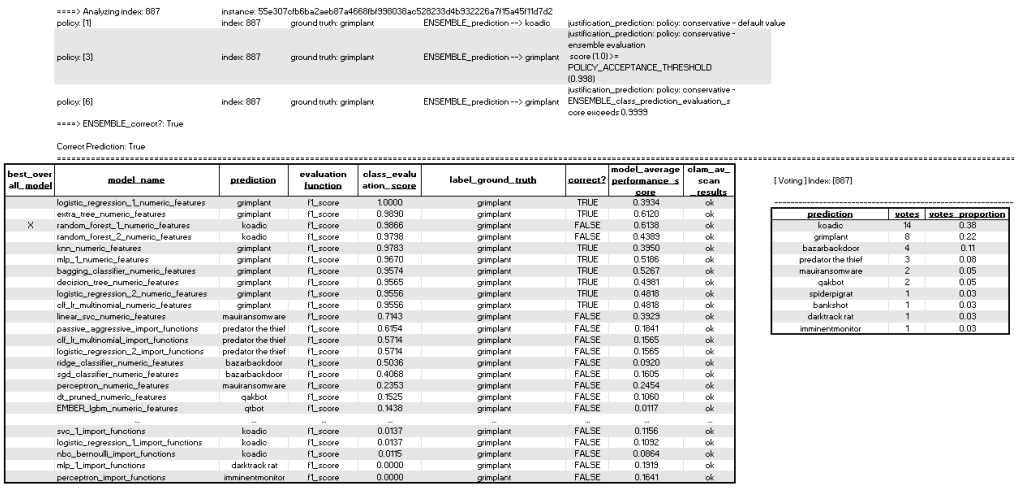


Fig. 8. NGram(1,3) Feature Hyperparameter Searching

Appendix 4: Case Study Ensemble Decision Flow



```
====> Analyzing index: 887   instance: 55e307cfb6ba2aeb87a4668fbf998038ac528233d4b932226a7f15a45f11d7d2
policy: [1]   index: 887   ground truth: grimplant   ENSEMBLE_prediction --> koadic   justification_prediction: policy: conservative - default value
policy: [3]   index: 887   ground truth: grimplant   ENSEMBLE_prediction --> grimplant   justification_prediction: policy: conservative - ensemble evaluation score (1.0) > POLICY_ACCEPTANCE_THRESHOLD (0.998)
policy: [6]   index: 887   ground truth: grimplant   ENSEMBLE_prediction --> grimplant   justification_prediction: policy: conservative - ENSEMBLE_class_prediction_evaluation_score exceeds 0.9999
====> ENSEMBLE_correct?: True
Correct Prediction: True
```

best_overall_model	model_name	prediction	evaluation function	class_evaluation_score	label_ground_truth	correct?	model_average performance_score	class_av_ scan results
	logistic_regression_1_numeric_features	grimplant	f1_score	1.0000	grimplant	TRUE	0.3934	ok
	extra_tree_numeric_features	grimplant	f1_score	0.9830	grimplant	TRUE	0.6120	ok
X	random_forest_1_numeric_features	koadic	f1_score	0.9866	grimplant	FALSE	0.6138	ok
	random_forest_2_numeric_features	koadic	f1_score	0.9736	grimplant	FALSE	0.4389	ok
	knn_numeric_features	grimplant	f1_score	0.9783	grimplant	TRUE	0.3950	ok
	mlp_1_numeric_features	grimplant	f1_score	0.9670	grimplant	TRUE	0.5186	ok
	bagging_classifier_numeric_features	grimplant	f1_score	0.9574	grimplant	TRUE	0.5267	ok
	decision_tree_numeric_features	grimplant	f1_score	0.9565	grimplant	TRUE	0.4361	ok
	logistic_regression_2_numeric_features	grimplant	f1_score	0.9556	grimplant	TRUE	0.4818	ok
	clf_lr_multinomial_numeric_features	grimplant	f1_score	0.9556	grimplant	TRUE	0.4818	ok
	linear_svc_numeric_features	mauiransomware	f1_score	0.7143	grimplant	FALSE	0.3329	ok
	passive_aggressive_import_functions	predator the thief	f1_score	0.6154	grimplant	FALSE	0.1841	ok
	clf_lr_multinomial_import_functions	predator the thief	f1_score	0.5714	grimplant	FALSE	0.1565	ok
	logistic_regression_2_import_functions	predator the thief	f1_score	0.5714	grimplant	FALSE	0.1565	ok
	ridge_classifier_numeric_features	bazarbackdoor	f1_score	0.5036	grimplant	FALSE	0.0920	ok
	sgd_classifier_numeric_features	bazarbackdoor	f1_score	0.4068	grimplant	FALSE	0.1605	ok
	perceptron_numeric_features	mauiransomware	f1_score	0.2353	grimplant	FALSE	0.2454	ok
	dt_pruned_numeric_features	qakbot	f1_score	0.1525	grimplant	FALSE	0.1060	ok
	EMBER_lgbm_numeric_features	qbot	f1_score	0.1438	grimplant	FALSE	0.0117	ok
	...	...	...	...	...	...	...	...
	svc_1_import_functions	koadic	f1_score	0.0137	grimplant	FALSE	0.1156	ok
	logistic_regression_1_import_functions	koadic	f1_score	0.0137	grimplant	FALSE	0.1032	ok
	nbc_bernoulli_import_functions	koadic	f1_score	0.0115	grimplant	FALSE	0.0864	ok
	mlp_1_import_functions	darktrack rat	f1_score	0.0000	grimplant	FALSE	0.1919	ok
	perceptron_import_functions	imminentmonitor	f1_score	0.0000	grimplant	FALSE	0.1641	ok

[Voting] Index: [887]

prediction	votes	votes_proportion
koadic	14	0.38
grimplant	8	0.22
bazarbackdoor	4	0.11
predator the thief	3	0.08
mauiransomware	2	0.05
qakbot	2	0.05
spiderpigrat	1	0.03
bankshot	1	0.03
darktrack rat	1	0.03
imminentmonitor	1	0.03

Fig. 9. Case Study: Ensemble Dynamic Selection Over Majority Vote

References

1. Anahideh, H., Nezami, N., Asudeh, A.: Finding representative group fairness metrics using correlation estimations. In: Expert Systems with Applications (2025),

https://openreview.net/forum?id=DEX3gP6RWm
2. Anderson, H.S., Roth, P.: Ember: An open dataset for training static pe malware machine learning models. arXiv preprint arXiv:1804.04637 (2018)
3. ANY.RUN: Malware signatures explained: How security tools spot threats (2024), https://any.run/cybersecurity-blog/malware-signatures-explained/, accessed: 2025-06-03
4. AV-TEST: Malware statistics. https://www.av-test.org/en/statistics/malware/, accessed: Mar. 13, 2025
5. Barbieri, M.M., Berger, J.O.: Optimal predictive model selection. The Annals of Statistics **32**(3), 870–897 (2004), https://arxiv.org/pdf/math/0406464.pdf
6. Brezinski, K., Ferens, K.: Metamorphic malware and obfuscation: A survey of techniques, variants, and generation kits. Security and Communication Networks **2023** (2023). https://doi.org/10.1155/2023/8227751, https://doi.org/10.1155/2023/8227751
7. Carlin, D., O'Kane, P., Sezer, S.: A cost analysis of machine learning using dynamic runtime opcodes for malware detection **85**, 138–155. https://doi.org/10.1016/j.cose.2019.04.018
8. Cheng, L., Zhang, Y., Han, Z., Deng, Y., Sun, X., Feng, D.: Evaluating and comparing the quality of access control in different operating systems. Computers & Security **47**, 26–40 (2014)
9. Gibert, D., Mateu, C., Planes, J.: The rise of machine learning for detection and classification of malware: Research developments, trends and challenges. Journal of Network and Computer Applications **153**, 102526 (2020)
10. Hsiao, S.W., Sun, Y.S., Chen, M.C.: Virtual machine introspection based malware behavior profiling and family grouping. arXiv preprint arXiv:1705.01697 (2017), https://arxiv.org/abs/1705.01697
11. Joyce, R.J., Miller, G., Roth, P., Zak, R., Zaresky-Williams, E., Anderson, H., Raff, E., Holt, J.: Ember2024 - a benchmark dataset for holistic evaluation of malware classifiers. In: Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining (2025)
12. Ko, A.H., Sabourin, R., Britto Jr, A.d.S.: Dynamic classifier selection: Recent advances and perspectives. Pattern Recognition **41**(12), 3460–3475 (2008)
13. Kuncheva, L.I.: Combining Pattern Classifiers: Methods and Algorithms. Wiley, Hoboken, NJ (2004)
14. Li, B., Roundy, K., Gates, C., Vorobeychik, Y.: Large-scale identification of malicious singleton files. In: Proceedings of 7th ACM Conference on Data and Application Security and Privacy (2017)
15. MalwareBazaar: Malwarebazaar: A repository for sharing malware samples (2025), https://bazaar.abuse.ch
16. Maniriho, P., Mahmood, A.N., Chowdhury, M.J.M.: A systematic literature review on windows malware detection: Techniques, research issues, and future directions. Journal of Systems and Software **192** (2024)
17. Maniriho, P., Mahmood, A.N., Chowdhury, M.J.M.: A systematic literature review on windows malware detection: Techniques, research issues, and future directions. Journal of Systems and Software **209**, 111921 (March 2024). https://doi.org/10.1016/j.jss.2023.111921
18. Nelson, T., O'Brien, A., Noteboom, C.: Machine learning applications in malware classification: A meta-analysis literature review. International Journal on Cybernetics and Informatics (IJCI) **12**(1) (Feb 2023). https://doi.org/10.5121/ijci.2023.120109

19. NSF Action AI Institute: Mabel: [m]alware [a]nalysis [be]nchmark for artificial intelligence and machine [l]earning; malware dataset family - ready for modeling (2024), https://github.com/action-ai-institute/MABEL-dataset/tree/main/release/malware_family/ready_for_modeling
20. Raschka, S.: Python Machine Learning. Packt Publishing, Birmingham, UK (2015)
21. Sasa Software: Payload obfuscation: How attackers hide malware in plain sight (2024), https://www.sasa-software.com/learning/payload-obfuscation-how-attackers-hide-malware/, accessed: 2025-06-16
22. Scarfone, K., Mell, P.: Guide to malware incident prevention and handling for desktops and laptops. Tech. Rep. NIST SP 800-83 Revision 1, National Institute of Standards and Technology (2013), https://nvlpubs.nist.gov/nistpubs/specialpublications/nist.sp.800-83r1.pdf, definition of malware on p. 2
23. SentinelOne: What is a malware file signature and how does it work? (2024), https://www.sentinelone.com/blog/what-is-a-malware-file-signature-and-how-does-it-work/, accessed: 2025-06-03
24. Sun, M., Li, X., Lui, J.C.S., Ma, R.T.B., Liang, Z.: Monet: A user-oriented behavior-based malware variants detection system for android. IEEE Transactions on Information Forensics and Security **11**(11), 2278–2290 (2016). https://doi.org/10.1109/TIFS.2016.2646641, https://doi.org/10.1109/TIFS.2016.2646641
25. Tahir, R.: A study on malware and malware detection techniques. International Journal of Education and Management Engineering (IJEME) **8**(2), 20–30 (2018). https://doi.org/10.5815/ijeme.2018.02.03, https://www.mecs-press.net/ijeme/ijeme-v8-n2/IJEME-V8-N2-3.pdf
26. VirusShare: Virusshare: A repository of malware samples for researchers (2025), https://virusshare.com
27. VirusTotal: 2023 emerging threats report. https://assets.virustotal.com/reports/2023emerging.pdf
28. VirusTotal: How it works. https://docs.virustotal.com/docs/how-it-works (2024), accessed: Mar. 12, 2025
29. VX-Underground: Vx-underground: The largest collection of malware source code, samples, and papers (2025), https://vx-underground.org
30. Yang, L., Ciptadi, A., Laziuk, I., Ahmadzadeh, A., Wang, G.: Bodmas: An open dataset for learning-based temporal analysis of pe malware. https://liminyang.web.illinois.edu/data/DLS21_BODMAS.pdf (2021)
31. Zealots, T.: Pe (portable executable) structure – malware analysis part 2 (2023), https://tech-zealots.com/malware-analysis/pe-portable-executable-structure-malware-analysis-part-2/, accessed: 2025-06-03