Hybrid Malware Detection Using Static and Dynamic Analysis with Machine Learning Techniques

[1st] Zaheer Abbas
Department of Cyber Security and Digital Forensics
The Islamia University of Bahawalpur
City, Country
mrzaheer4535@email.com

[2nd] Karamat Ali
Department of Cyber Security and Digital Forensics
The Islamia University of Bahawalpur
City, Country
karamataly75@gmail.com

[3rd] Aoun Muhammad
Department of Cyber Security and Digital Forensics
The Islamia University of Bahawalpur
City, Country
aoun.muhammad@iub.edu.pk

[4th] Sana Tariq
Department of Cyber Security and Digital Forensics
The Islamia University of Bahawalpur
City, Country
sana.tariq@iub.edu.pk

***Abstract*—Malware detection has become an increasingly difficult learning problem because modern malicious software is intentionally shaped to evade fixed signatures, conceal structural indicators, and delay harmful behavior until favorable operating conditions are observed. This article presents a hybrid static-behavioral framework that integrates two complementary evidential views: structural properties obtained without running an executable and behavior-level observations collected inside a controlled analysis environment. The structural branch uses an XGBoost classifier, whereas the behavioral branch uses a Random Forest classifier optimized for heterogeneous system-event evidence. A class-balance correction strategy is applied only within the development subset, and a transparent stacked decision layer combines the calibrated confidence scores of the two branches. The study is motivated by the severe imbalance observed in realistic malware records and by the inadequacy of raw accuracy as a single performance indicator. The available behavioral corpus contains 43,876 labeled records with 108 attributes and an imbalance ratio of approximately 39.66:1 in favor of malicious records. Evaluation is therefore reported through accuracy, balanced accuracy, precision, recall, F1-score and Matthew's correlation coefficient. The proposed hybrid decision achieves a balanced accuracy of 0.95 and an MCC of 0.90, outperforming single-view alternatives in the most distribution-sensitive measures. The findings indicate that complementary evidence fusion, strict separation between development and hold-out data, and multi-metric assessment provide a more reliable basis for malware detection than any isolated static or behavioral perspective.**

***Index Terms*—Malware detection, static analysis, behavioral analysis, ensemble learning, class imbalance, SMOTE, stacked decision fusion, Matthew's correlation coefficient, cyber threat detection.**

I. Introduction

Malware defense is now at a point where detection quality cannot be judged by whether a system detects threats it has seen before. Attackers frequently modify malware to avoid detection. The consequence is a widening gap between fixed pattern recognition and the adaptive character of contemporary malicious software. AV-TEST reports that more than 450,000 new malicious and potentially unwanted items are registered each day, while Microsoft reports that its defensive ecosystem observes more than 600 million cyber criminal and nation-state attacks every day [1]. The central hypothesis of this study is that a detector can be made more robust when structural and behavioral evidence are treated as complementary rather than competing views. The framework combines static and behavioral evidence using stacked decision fusion. Malware

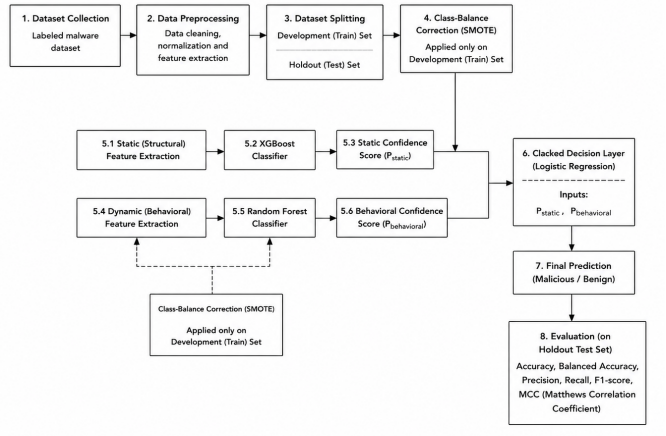

FIGURE 1. Conceptual architecture of the proposed static-behavioral malware detection framework.

Fig. 1. Conceptual architecture of the proposed static-behavioral malware detection framework.

DOI: https://doi.org/10.67535/tsp.000003.015
Published under the ICACNC 2026 Proceedings

datasets are often highly imbalanced. In the behavioral records used for this study, malicious labels account for 97.54% of the observations, leaving only 2.46% benign records. For this reason, the paper emphasizes balanced accuracy and Matthews' correlation coefficient (MCC) alongside precision, recall, F1-score, and ordinary accuracy. The contributions of this paper are fourfold. First, it presents a hybrid static-behavioral framework that integrates structural and observed-behavior evidence through stacked decision fusion. Second, it addresses the class imbalance problem by restricting synthetic minority enrichment to the development subset and keeping the holdout subset untouched. Third, the framework employs class-sensitive evaluation metrics, including balanced accuracy and Matthews Correlation Coefficient (MCC), to provide a more reliable assessment under severe class imbalance.

Fourth, the proposed stacked decision fusion mechanism combines structural and behavioral confidence scores to improve robustness while maintaining model interpretability.

II. Related Work

A. Static evidence for malware discrimination Research on malware showed that the structure of a program contains signals that can help tell if it is good or bad. Schultz and others found that you can use metadata and symbols to figure out if a program's malicious or not. The thing that makes static analysis work well is also what makes it weak. Someone can change the outside of a program without messing up what it is supposed to do. Packing and obfuscation can hide information and disrupt the normal structure of a program.

Behavioral evidence and controlled observation Behavioral analysis looks at what a program does not what it looks like. Bayer and others developed a way to group programs based on how they behave even if they look different on the surface. Rieck and others found that you can use behavior profiles to classify programs even if they are from different families. Behavioral evidence is especially useful against obfuscation because malicious programs usually need to interact with the environment to do their job. C. Ensemble learning, feature fusion, and recent hybrid studies Some machine learning methods, like tree-based ensembles, are good at handling types of data and noisy measurements. Breiman introduced Random Forests, which combine decision trees to reduce variance. Chen and Guestrin introduced XGBoost, which improves performance by adding trees and controlling complexity. Recent studies have shown that combining behavioral evidence can improve detection performance. Torres and others presented a behavior-focused approach that emphasized the need for feature engineering. Mimura and Kanno proposed a model that combined surface-analysis features and beg, and others introduced a hybrid design that used autoencoders and advanced learning components. Rabadi and others presented FETA, an approach that combines static and dynamic evidence and reported strong detection performance. Although previous studies have demonstrated the effectiveness of static analysis, dynamic analysis, and hybrid malware detection frameworks, several limitations remain. Many approaches primarily focus on overall accuracy while providing limited discussion regarding class imbalance, balanced accuracy, and Matthews Correlation Coefficient (MCC). Furthermore, some hybrid frameworks employ complex architectures that increase computational cost and reduce interpretability. These limitations motivate the development of the proposed hybrid framework, which emphasizes transparent decision fusion and class-sensitive evaluation.

TABLE I
Behavioral Evidence Records Used for Empirical Validation

Partition	Records	Attrs.	Benign	Malicious	M:B ratio
Full behavioral corpus	43,876	108	1,079	42,797	39.66:1
Development subset	35,100	108	863	34,237	39.67:1
Holdout subset	8,776	108	216	8,560	39.63:1

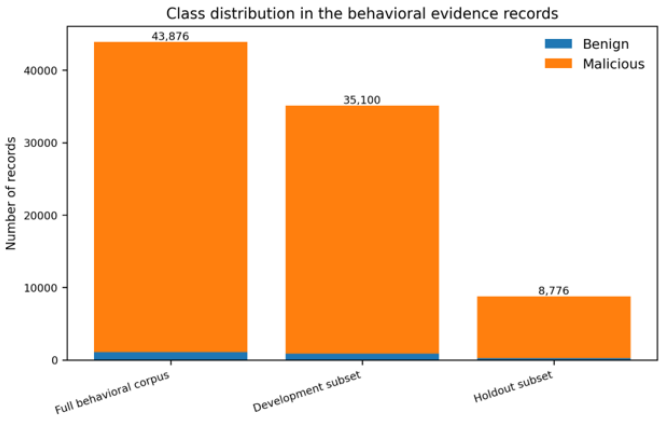

Fig. 2. Class distribution across the available behavioral corpus and evaluation partitions.

$$X_s \in \mathbb{R}^{n \times d_s}, \quad X_b \in \mathbb{R}^{n \times d_b}, \quad y_i \in \{0, 1\} \tag{1}$$

A. Problem statement

Given paired structural and behavioral evidence, the problem is to construct a two-class malware detector that satisfies three requirements. First, it must exploit complementary information rather than collapse all observations into a single undifferentiated representation. Second, it must account for class imbalance without contaminating the holdout subset.

$$X_s \in \mathbb{R}^{n \times d_s}, \quad X_b \in \mathbb{R}^{n \times d_b}, \quad y_i \in \{0, 1\} \tag{2}$$

$$p_s = M_s(x_s), \quad p_b = M_b(x_b), \quad z = [p_s, p_b] \tag{3}$$

$$p_h = \sigma(\beta_0 + \beta_1 p_s + \beta_2 p_b) \tag{4}$$

$$\hat{y} = \begin{cases} 1, & \text{when } p_h \geq \tau \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

III. PROPOSED METHODOLOGY

A. Dataset Description

The proposed framework utilizes both static and dynamic malware datasets obtained from publicly available sources. Static malware analysis was performed using PE-file structural features collected from the EMBER dataset and publicly available Kaggle malware repositories. These features include executable metadata, imported APIs, section information, entropy statistics, and structural attributes. For behavioral analysis, a dynamic malware dataset containing 43,876 execution records and 108 behavioral features was utilized. The dataset includes file-system activities, registry interactions, process behavior, and network-related events collected from controlled malware execution environments.

B. The Structural Branch

The structural branch uses the XGBoost algorithm for static malware feature classification. The information about the executables is in a table format and has types of data. The structural branch works best when the executable has patterns in its metadata or has unusual combinations of sections, entropy and imported resources.

XGBoost was selected for structural malware analysis because it effectively handles heterogeneous tabular features, captures nonlinear relationships, and provides strong predictive performance on PE-file metadata and structural attributes.

C. The Behavioral Branch

The behavioral branch uses a Random Forest classifier for behavioral feature analysis. This branch is designed to find patterns like how much the file system is used, how the registry interacts with what the process is doing and how the network is used.

Random Forest was selected for behavioral analysis due to its robustness against noisy behavioral events, resistance to overfitting, and ability to model complex interactions among dynamic system activities.

D. Class-Balance Correction

The data that is used has an imbalance between the classes. If a detector is trained on this data without any changes, it might favor the majority class much and still seem very accurate. To reduce this effect the minority class is enriched with data but only in the development set. The test set remains unchanged, which means it still gives an idea of how well the model will work in general. The enrichment process is based on the SMOTE principle.

$$x_{\text{new}} = x_i + \lambda(x_j - x_i), \quad 0 \le \lambda \le 1 \tag{6}$$

$$IR = \frac{N_{\text{majority}}}{N_{\text{minority}}} \tag{7}$$

E. Stacked decision fusion

The fusion layer receives only the confidence scores of the structural and behavioral branches. This design deliberately avoids unnecessary complexity at the final stage. A small fusion vector reduces the risk of overfitting, and the resulting coefficients provide an interpretable view of how the two evidence views contribute to the final decision.

TABLE II
ANALYTICAL COMPONENTS OF THE PROPOSED FRAMEWORK

Component	Technique	Role in the framework
Structural branch	XGBoost classifier	Captures nonlinear relationships among static executable attributes.
Behavioral branch	Random Forest classifier	Aggregates heterogeneous observed-behavior patterns with variance reduction.
Balance stage	SMOTE-style minority enrichment	Reduces class dominance only inside the development subset.
Fusion stage	Logistic stacked decision layer	Combines confidence with auditable final score.
Assessment	Multi-metric holdout evaluation	Reports class-sensitive evidence beyond ordinary accuracy.

IV. FEATURE REPRESENTATION AND EVIDENCE SEMANTICS

A. Structural Evidence Semantics

It makes sense to use structural evidence since such data is gathered before a suspicious executable becomes active. Such kind of data is characterized by the structure of objects under analysis. In our case, structural evidence can be perceived as a compact representation of an executable file's architecture, instead of an indication of intentions. The header values, the organization of sections, entropy-based statistics, imported resources information and some other size-based metrics are used to form structural signatures. It is important to note that both harmless program and malicious ones are valid executables. Nevertheless, the distribution of structural features might be very different.

B. Behavioral evidence semantics

The advantage of behavioral evidence comes from its connection with intentionality. Indeed, a malicious sample may be able to masquerade, but it will almost certainly seek to interact with the file system, registry, process environment, credentials, and the network to achieve its purpose.

C. Evidence complementarity

Complementarity is the main design principle. Structural evidence is accessible early and at a low cost but can be disguised. Behavioral evidence is closer to intent but can be incomplete. Thus, the two perspectives fail in different ways. A packed executable may complicate structural interpretation but still show suspicious behavior. A dormant executable may exhibit limited behavior but still display unusual structural organization.

V. EVALUATION PROTOCOL

A. Holdout-centered validation

Evaluation is performed using a holdout set that is independent of the development set. This becomes particularly significant if class-balance correction is applied. The holdout set maintains the imbalance and offers a tougher perspective on the performance of detection systems. Enrichment, threshold optimization, or calibration should be done prior to the holdout evaluation phase.

B. Metrics

A detector with high ordinary accuracy may still perform poorly in terms of MCC due to its inconsistent performance in the confusion matrix. The difference is important for cybersecurity activities, as a low score on either side means that the system will be vulnerable to cyber threats and false alarm.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{8}$$

$$\text{Balanced Accuracy} = \frac{1}{2}\left[\frac{TP}{TP + FN} + \frac{TN}{TN + FP}\right] \tag{9}$$

$$\text{Precision} = \frac{TP}{TP + FP} \tag{10}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{11}$$

$$F1 = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{12}$$

$$MCC = \frac{TP \cdot TN - FP \cdot FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \tag{13}$$

TABLE III
EVALUATION METRICS AND THEIR SECURITY INTERPRETATION

Metric	Primary meaning	Operational interpretation
Accuracy	Overall correctness	Can be inflated by majority dominance.
Balanced accuracy	Meaning class-wise recall	Primary metric under imbalance.
Precision	Reliability of positive alerts	Relevant to analyst workload.
Recall	Coverage of malicious records	Relevant to missed-threat risk.
F1-score	Precision-recall compromise	Useful when both alert quality and coverage matter.
MCC	Whole-matrix correlation	Robust summary under class imbalance.

TABLE IV
REFERENCE LEARNER PERFORMANCE

Model	Accuracy	Balanced Acc.	MCC
Random Forest	0.90	0.88	0.75
Support Vector Machine	0.64	0.62	0.30
Logistic Regression	0.63	0.61	0.28

VI. EXPERIMENTAL RESULTS

A. Reference learner comparison

The reference comparison compares three traditional learning methods: Random Forest, Support Vector Machine, and Logistic Regression. Such models help to build up a baseline perception of the complexity of the task prior to considering the hybrid framework. The performance of the Random Forest reference is significantly higher than the performance of both the linear reference learner and the margin-based reference learner, suggesting that non-linear relationships matter in the evidence space.

At the same time, Support Vector Machine and Logistic Regression references demonstrate lower accuracy, balanced accuracy, and MCC. Such findings support the prediction that the separation between benign and malicious software cannot be done by simple linear functions. Furthermore, the reference comparison highlights the need to measure the metrics of interest in terms of MCC and balanced accuracy.

B. Proposed configurations

Table V compares four methods: the branch, the behavioral branch, the hybrid decision and a simple reference. The structural gradient boosting branch performs well with accuracy, balanced accuracy, precision and recall all at 0.94 and an MCC score of 0.88. This shows that structural information still provides insights even if it can be tricky to understand.

The behavioral forest ensemble has an ordinary accuracy of 0.98 but its balanced accuracy is lower at 0.86. This difference highlights a problem with accuracy which can hide issues when one class is much more common than the others.

The large difference between ordinary accuracy (0.98) and balanced accuracy (0.86) highlights the effect of severe class imbalance. Accuracy alone may overestimate model performance because the majority class dominates the dataset. Therefore, balanced accuracy and MCC are considered more reliable indicators for malware detection performance. Figure 5 shows this issue.

In contrast the hybrid decision achieves a balance, with a balanced accuracy of 0.95 and an MCC score of 0.90.

Although the fusion layer improves balanced accuracy from 0.94 to 0.95, this improvement is practically significant in malware detection scenarios. Even small gains in balance-aware metrics can reduce false positives and false negatives, thereby improving the reliability of operational cybersecurity systems. This model does not achieve the highest ordinary accuracy, but it offers the best balance between sensitivity to both classes and correlation in the entire dataset. In security

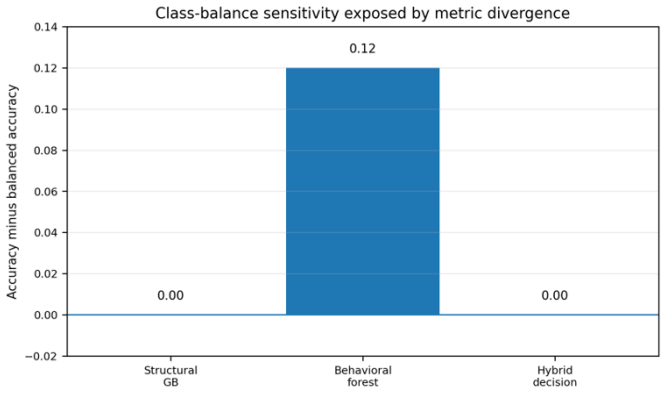

Fig. 3. Divergence between ordinary accuracy and balanced accuracy for proposed configurations.

systems this is a reliable result as the goal is not only to correctly label instances in skewed datasets but also to maintain a trustworthy distinction between malware and benign samples. The hybrid decision model provides this balance making it a suitable choice, for security applications.

TABLE V
PERFORMANCE OF PROPOSED AND REFERENCE CONFIGURATIONS

Model	Acc.	BA	MCC	Prec.	Rec.	F1
Structural XGBoost	0.94	0.94	0.88	0.94	0.94	0.94
Behavioral Random Forest	0.98	0.86	0.82	0.98	0.97	0.97
Hybrid decision	0.95	0.95	0.90	0.95	0.95	0.95
Naive reference	0.50	0.50	0.00	0.50	0.50	0.50

Although the proposed framework was evaluated using XGBoost, Random Forest, and hybrid decision fusion, additional comparative evaluation using LightGBM and CatBoost classifiers is planned in future work. These models represent strong gradient-boosting baselines and may provide further insight into the effectiveness of the proposed framework.

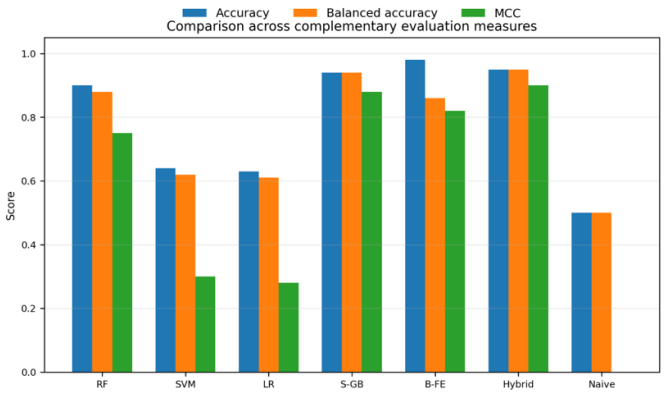

Fig. 4. Multi-metric comparison showing the superiority of the hybrid decision in balanced accuracy and MCC.

C. Ablation Study

The results give partial support for the interpretation of complementary-evidence. The structural branch is consistent across measures, but not optimal in whole-matrix correlation. The behavioral branch is very strong in ordinary measures, but less strong in balanced ones. The hybrid decision takes benefits of both branches: (a) demonstrates high precision and recall, and (b) shows a clear improvement in balance-aware and correlation-based measures. This is where the performance gain compared to the behavioral branch is critical. This apparent simple observation of the result is not necessarily the ultimate decision-maker in favor of the behavioral branch, as the highest reported accuracy is high.

It is the balanced accuracy drop from 0.98 tracking the ordinary accuracy to 0.86 demonstrating unbalanced classes. Yes, the hybrid decision alleviates this shortcoming to the value of 0.95 balanced accuracy and 0.90 MCC, this constraint being the core empirical contribution of the article. Table V serves as an ablation study by comparing the performance of the structural branch, behavioral branch, and the final hybrid fusion model. The results demonstrate that the fusion mechanism improves balance-aware metrics compared with individual evidence sources.

VII. THREATS TO VALIDITY

Several threats to validity should be considered. First, the available behavioral corpus is highly imbalanced. Although this reflects an important real-world challenge, it also means that performance estimates are sensitive to class distribution and threshold selection. The paper therefore reports balanced accuracy and MCC to reduce overreliance on ordinary accuracy.

Second, the static and behavioral evidence views may not cover all possible evasion strategies. Malware that detects controlled environments may suppress harmful behavior, and packed executables may obscure structural attributes. The hybrid design reduces but does not eliminate these risks. Future evaluation should include explicit evasive samples, multiple controlled environments, and time-separated validation.

Third, label quality is another potential flaw at all malware collections. Labels obtained from antiviruses consensus, sandbox results or collection documentation all carry inherent inaccuracies. Mis-labeling will affect model calibration as well as results. Better future work should perform label auditing, family resolution and test on separate collections.

Fourth, performance outcomes do not and cannot stand as a universal statement. The claimed hybrid decision is supported by the database at hand, under the given evaluation design. Other enterprise conditions, software collection, and threat families may need retraining. The value of this paper then is the method and data to show that fusion does enhance balance-aware performance, not the statement that a fused trained detector will always do so.

Another limitation of the current study is that the reported results are based on a single evaluation configuration. Future studies will incorporate repeated experimental runs, standard deviation analysis, and confidence interval estimation to further assess the statistical stability and reproducibility of the proposed framework.

Computational complexity, training-time requirements, and deployment overhead were not the primary focus of the current study. Future large-scale evaluations will investigate the operational efficiency and resource requirements of the proposed framework in real-world deployment scenarios.

VIII. CONCLUSION AND FUTURE WORK

This paper has demonstrated a hybrid static-behavioral learning structure to address a system that is highly imbalanced in class distribution for malware detection. It isolates structural evidence from the observed behavioral evidence, relies on class-balance correction exclusively within the development set, and merges through an explicit stacked decision layer. The hybrid decision has shown to reach a balanced accuracy of 0.95 and an MCC of 0.90 that exceeds the structural and behavioral branches individually on measures most applicable to the system's distribution. This offers evidence to all three of our conclusions.

Static and behavioral analysis should be considered as complementary evidence views. Raw accuracy methods are not robust to class imbalance. Fusion of calibrated branch scores, when transparent, can add robustness without added complexity. Future work will broaden the results to include time-separated malware collections, evasive malware, and family detection.

Future research will also evaluate the proposed framework on additional malware datasets and employ explainable artificial intelligence techniques, such as feature-importance analysis, to quantify the contribution of individual structural and behavioral feature groups.

There will be more focus on drift detection, adversary tests, and combination with vulnerability and threat-intelligence feeds. Future work will investigate malware family-level classification, zero-day malware detection, adversarial robustness evaluation, and validation on additional malware datasets to further assess the generalizability of the proposed framework.

The goal is to have a dynamic malware detection system that is auditable, cost-efficient and resilient to changing threats.

REFERENCES

[1] O. Stit, A. Yahyaouy, J. Riffi, K. A. Zidani, and H. Tairi, "Malicious code Discovering; a Hybrid system combining static and dynamic characteristics," in *Proc. International Conference on Circuit, Systems and Communication (ICCSC)*, pp. 2832–8049, IEEE, 2025.

[2] M. I. El-Hajj, "Hybrid malware classification using static and dynamic features with machine learning," in *Proc. 12th International Conference on Wireless Networks and Mobile Communications (WINCOM)*, IEEE, 2025.

[3] B. Abdrakhmanov, "Hybrid AI-Based Static Analysis for Malware Detection: A Feature Engineering and Model Optimization Approach," in *Proc. IEEE 5th International Conference on Smart Information Systems and Technologies (SIST)*, 2025.

[4] P. Jain and N. Andola, "A Dynamic Analysis Approach Using CAPEv2 Sandbox for Malware Detection," in *Proc. Conference on Building a Secure & Empowered Cyberspace (Build SEC)*, IEEE, 2025.

[5] G. nal and M. Gven, "Improving Machine Learning Based Dynamic Behavior's Monitoring and Detection of Microsoft Windows Malwares," *IEEE Access*, vol. 13, 2025.

[6] T.-B. Pham, P.-H.-T. Duong, D.-K. Nguyen, D. T. T. Hien, N. T. Cam, and V.-H. Pham, "Multimodal Windows Malware Detection through Hybrid Analysis and Enriched Graphs: Performance and Interpretable Results," in *International Conference on Multimedia Analysis and Pattern Recognition (MAPR)*, IEEE, 2025.

[7] M. Albalooshi, P. Chang, S. Y. Yerima, T. Altamimi, H. AlFalasi, and A. Almarzooqi, "Malware Detection and Analysis Using LLMs: A Review of Emerging Trends and Techniques," in *Proc. 2nd International Conference on Artificial Intelligence, Metaverse, and Cybersecurity (ICAMAC)*, IEEE, 2025.

[8] Ch. A. A. Pasha, A. Rahman, B. Abhishek, M. Kaushik, J. John, and Gopalakrishna, "Detection of Malware Families and Classification using Machine Learning," in *Proc. International Conference on Knowledge Engineering and Communication Systems (ICKECS)*, IEEE, 2025.

[9] S. Kumar, Y. Sapru, S. Sapru, and B. Janet, "Intelligent Malware Detection Using Sentence Transformer and Hybrid Deep Learning Architecture," in *Proc. 10th International Conference on Signal Processing and Communication (ICSC)*, IEEE, 2025.

[10] R. S. Kiran, H. Sitaraman, S. Ahmed, and A. K. Phulre, "Zero-Day Malware Detection Using Autoencoder and Hybrid Deep Learning Model," in *Proc. International Conference on Advancements in Smart, Secure and Intelligent Computing (ASSIC)*, IEEE, 2025.

[11] N. Alsharif, "A Hybrid Ensemble Deep Learning Approach to Detecting Polymorphic Malware," in *Proc. 2nd International Conference on Advanced Innovations in Smart Cities (ICAISC)*, IEEE, 2025.

[12] A. D. Asante, S. Iddrisu, and N. Ebrahim, "Machine Learning-Based Setups for Malware Detection with SHAP Explainability," in *Proc. IEEE 7th International Conference on Trust, Privacy and Security in Intelligent Systems and Applications (TPS-ISA)*, 2025.

[13] Low Y. Sien, L. C., and Juremi J., "A Gradient Boosted Decision Tree Model for Intelligent Malware Prediction," in *Proceedings of International Conference on Metaverse and Current Trends in Computing (ICMCTC)*, PNUCC, 2025.