Memory-Resident Malware Detection via a Hybrid Deep Learning Framework

DOI: https://doi.org/10.54654/isj.v3i26.1177

Le Phu Minh*, Do Dinh Quang, Hoang Viet Long, Nguyen Thi Kim Son

***Abstract*— Memory-resident malware detection is a critical cybersecurity challenge, particularly with stealth techniques like Living-off-the-Land (LotL). This paper proposes a hybrid deep learning framework to detect malware from memory-behavior data represented as fixed-length tabular features. The framework emphasizes an effective data processing pipeline rather than a complex model architecture. It has three stages: (1) feature selection and Z-score standardization using Extreme Gradient Boosting (XGBoost) and StandardScaler, (2) data balancing and cleaning using Synthetic Minority Over-sampling Technique (SMOTE) and Edited Nearest Neighbor (ENN), and (3) training a Transformer Encoder-based classifier to extract high-level non-linear representations from the stabilized feature space, utilizing robust Feed-Forward Networks, Layer Normalization, residual connections, and Focal Loss to enhance training stability under class imbalance. Training further employs a StepLR Scheduler and Early Stopping to ensure convergence and prevent overfitting. On the CIC-MalMem-2022 dataset, which comprises one benign class and 15 malware classes, the proposed framework achieves 76.62% Accuracy and 76.35% F1-score, outperforming traditional baselines. These results demonstrate the framework's effectiveness for proactive malware defense based on memory behavioral analysis.**

***Tóm tắt*— Phát hiện phần mềm độc hại cư trú trong bộ nhớ là một thách thức an ninh mạng quan trọng, đặc biệt trước các kỹ thuật tàng hình như Living-off-the-Land (LotL). Bài báo này đề xuất một khung học sâu lai để phát hiện mã độc từ dữ liệu hành vi bộ nhớ được biểu diễn dưới dạng đặc trưng bảng (tabular) có độ dài cố định. Khung mô hình nhấn mạnh một quy trình xử lý dữ liệu hiệu quả thay vì một kiến trúc mô hình quá phức tạp. Khung gồm ba giai đoạn: (1) lựa chọn đặc trưng và chuẩn hóa Z-score bằng Extreme Gradient Boosting (XGBoost) và StandardScaler, (2) cân bằng dữ liệu và làm sạch bằng Synthetic Minority Over-sampling Technique (SMOTE) và Edited Nearest Neighbor (ENN), và (3) huấn luyện bộ phân loại dựa trên Transformer Encoder nhằm trích xuất các biểu diễn phi tuyến tính cấp cao từ không gian đặc trưng đã được ổn định, tận dụng các mạng truyền thẳng (Feed-Forward Networks) mạnh mẽ, kết hợp Layer Normalization, kết nối dư và hàm mất mát Focal Loss để tăng ổn định huấn luyện trong bối cảnh mất cân bằng lớp. Quá trình huấn luyện tiếp tục sử dụng bộ điều chỉnh tốc độ học StepLR và kỹ thuật dừng sớm (Early Stopping) để đảm bảo hội tụ và hạn chế quá khớp. Trên bộ dữ liệu CIC-MalMem-2022 (gồm một lớp lành tính và 15 lớp mã độc), khung đề xuất đạt 76.62% Accuracy và 76.35% F1-score, vượt trội so với các mô hình nền truyền thống. Những kết quả này cho thấy tính hiệu quả của khung mô hình cho phòng thủ chủ động dựa trên phân tích hành vi bộ nhớ.**

***Keywords*— Malware, hybrid framework, CIC-MalMem-2022, deep learning.**

***Từ khóa*— Mã độc; khung mô hình lai, CIC-MalMem-2022, học sâu.**

I. Introduction

Detecting memory-resident malware poses a significant challenge in modern cybersecurity. Unlike traditional malware, this type of malware, such as LotL, exploits legitimate native processes of the operating system, which renders signature-based security methods less effective in detection [1, 2]. Therefore, process behavior analysis methods based on memory data, combined with deep learning, have received considerable attention and are being actively developed [3].

However, three obstacles remain. **(i)** Memory-acquired data is high-dimensional,

This manuscript was received on October 29, 2025. It was reviewed on December 15, 2025, revised on December 19, 2025 and accepted on December 22, 2025.

* Corresponding author.

heterogeneous, and often noisy/redundant, which hinders learning stable, discriminative, and generalizable representations needed for reliable detection [2 - 4]. **(ii)** Class distributions are typically highly imbalanced, a problem where individual malware families form minute minorities, severely degrading the learning process and accuracy [5 - 7]. **(iii)** Finally, as recent surveys [7] indicate, applying complex deep learning architectures (such as Transformers [8]) directly to tabular data often fails. They are highly sensitive to the noisy features and severe class imbalance common in such datasets, frequently underperforming robust tree-based models [7].

In this context, our Hybrid Framework is proposed as a holistic, three-stage solution that effectively integrates a data processing pipeline (Stages 1 and 2) with an effective deep learning classifier (Stage 3). We posit that this synergistic approach is crucial: the pipeline (XGBoost and SMOTE-ENN) first addresses feature noise/redundancy and severe class imbalance, which in turn allows the Transformer-based classifier [8] (Stage 3), when trained using the Focal Loss [9] objective, to learn from a more stable and well-conditioned input space.

Importantly, CIC-MalMem-2022 provides each instance as a structured, fixed-length tabular vector, where each dimension corresponds to a consistent memory-behavior attribute across samples. This property makes the data compatible with an encoder-style architecture without requiring any temporal ordering assumption. After feature selection and Z-score standardization, each sample is mapped into a dense latent representation through a learnable projection layer; stacked Transformer Encoder blocks then leverage their Feed-Forward Networks and Layer Normalization to extract high-level non-linear representations from the stabilized tabular input. Since each instance is treated as a fixed-length vector rather than a sequence, the encoder operates without relying on token-to-token attention mechanisms. This combined approach enhances training stability and performance under the challenges highlighted in [7].

To tackle these challenges, our framework is implemented as a three-stage malware detection pipeline, which processes and learns effective representations from tabular-structured RAM data, as follows:

- **Stage 1:** Important feature selection is performed using Extreme Gradient Boosting (XGBoost) [10] (based on Gain metric) to reduce dimensionality and remove noisy/redundant features [4, 11], followed by Z-score standardization (StandardScaler) to stabilize training and improve convergence.
- **Stage 2:** Class imbalance is addressed by combining Synthetic Minority Over-sampling Technique (SMOTE) [5] to augment the minority class with Edited Nearest Neighbor (ENN) [5] to remove noise from the majority class, thereby producing a balanced training set.
- **Stage 3:** Given the fixed-length tabular representation of CIC-MalMem-2022, the final stage employs a Transformer Encoder–based classifier [8] as a robust deep learning module (without assuming any temporal ordering). Its architecture is stabilized by Layer Normalization [12] and residual connections [13]. The training process is driven by the Focal Loss [9] objective to improve learning on minority classes under imbalance, supported by a StepLR Scheduler for stable convergence and Early Stopping based on validation monitoring to prevent overfitting.

The model is empirically evaluated on the CIC-MalMem-2022 dataset, which comprises one benign class and 15 malware classes, with the goal of improving classification accuracy on an independent test set, enhancing generalization capability, and increasing the effectiveness of practical deployment in proactive malware detection systems.

II. BACKGROUND

A. Challenges in detecting malware from system memory

Detecting memory-resident malware is a crucial research direction against stealthy techniques, especially LotL attacks that exploit legitimate processes to perform malicious actions without leaving disk traces [2, 4]. Unlike static analysis relying on signatures or fixed traits, memory analysis focuses on real-time execution

behavior, enabling detection of hidden malicious activities within legitimate processes [2, 3].

Extracting and processing system memory data, however, is challenging: it is high-dimensional with noisy, redundant, or unstable features [2, 3]; severely imbalanced between benign and malicious samples, leading to biased learning [5]; and inconsistent across systems, reducing generalization [2].

Moreover, effective classification requires the model to learn complex non-linear relationships from this data, a capacity often beyond traditional models such as Random Forests or SVMs [7].

B. Deep learning models for malware detection

In response to the aforementioned challenges, numerous studies have explored deep learning architectures such as CNNs, RNNs, Autoencoders, and Transformers for behavior feature extraction and malware detection [2, 3, 14]. Among these, Transformer-based architectures [8] have shown significant promise in learning complex, non-linear relationships through stacked encoder blocks, although their effectiveness on tabular data is highly dependent on proper data preprocessing [7].

However, deep learning models also have certain limitations. First, their performance can degrade significantly in the presence of noisy data, limited labeled samples, or uneven data distributions [3, 5]. Second, these models often struggle with output interpretability, which is an essential factor in cybersecurity environments that demand transparency and verifiability [15].

C. Rationale for the proposed framework

To address the aforementioned limitations, this paper proposes a framework that synergistically combines a targeted data pipeline with a robust classifier.

Based on these insights, our framework's contribution is twofold:

- **A Preparatory Data Pipeline:** We incorporate a Data-Centric Pipeline (Stages 1 and 2) to address the known challenges of noise and severe imbalance in raw tabular data. This phase uses XGBoost [10] for feature selection and SMOTE-ENN [5] for data balancing. This preprocessing aims to stabilize the feature space and class distribution before classification.
- **A Robust Classification Stage:** We then employ an architecture (Stage 3) based on Transformer Encoder blocks [8], which is inherently stabilized by Layer Normalization [12] and residual connections [13]. Crucially, we train it with Focal Loss [9]. This classifier is designed to learn complex patterns from the prepared data, and we hypothesize the Focal Loss provides additional robustness against any remaining class imbalance identified in [7] at the objective level.

This synergistic approach allows the robust classification stage [8, 9] to benefit from the prepared data provided by the pipeline [5, 10], achieving maximum performance. As will be demonstrated in our experiments (Section V), this complete hybrid framework proves to be significantly more effective than traditional machine learning approaches for this task.

III. RELATED WORKS

Related works on memory-resident malware detection have explored deep learning, ensemble methods, and data augmentation to improve accuracy and robustness. This section reviews these approaches and notes the gaps addressed by the proposed pipeline-centric framework.

- **Memory-based malware detection:** Utilizing features extracted from memory has proven to be an effective approach. Studies such as XMal [16] and MeMalDet [3] have successfully applied deep learning models to this type of data. Furthermore, specialized memory forensics frameworks [2, 4] have been developed to provide crucial input data for detection systems.
- **Addressing data challenges:** Cybersecurity data, particularly malware data, commonly suffers from feature noise and severe class imbalance, which degrade detection performance. To address these issues, prior studies have explored ensemble learning approaches such as MalHyStack [11], as well as data-level resampling techniques including SMOTE-ENN [5] and ADASYN-based variants [17]. In addition, recent multi-stage frameworks like the CMC framework [18] integrate feature selection with

imbalance-aware preprocessing to enhance multi-class malware classification. A recent forensic-oriented study applied Gray Wolf Optimization for feature selection on the CIC-MalMem-2022 dataset and reported strong performance with Random Forest in both 4-class and 16-class settings [19].

- **Trends in model architectures:** While models have grown in complexity, recent comprehensive surveys on tabular data [7] have provided critical insights. The study [7] highlights that: (1) tree-based models, particularly XGBoost [10], often achieve superior performance, and (2) deep learning models, including Transformer-based architectures [8], often struggle on raw tabular data due to noise and imbalance, despite their theoretical advantages [7]. Concurrently, modern stabilization techniques such as Layer Normalization [12] and residual connections [13], along with specialized loss functions like Focal Loss [9] for imbalance, have been developed. Other approaches focusing on explainability (XAI) [15, 16] and lightweight models [6, 20] have also been explored to meet practical deployment needs.
- **Research gap:** Despite these advancements, most current research tends to focus on one of two directions: (1) developing complex, end-to-end deep learning models, or (2) focusing solely on data preprocessing techniques [5, 17]. Very few studies have proposed a hybrid, pipeline-centric framework. Specifically, a gap exists in systematically combining a data preprocessing pipeline with a robust classification stage. This research investigates a framework that systematically applies a data processing pipeline (Stages 1 and 2) utilizing XGBoost [10] for feature selection and SMOTE-ENN [5] for data balancing. This pipeline is designed to first mitigate the critical weaknesses of noise and imbalance identified in [7], thereby enabling the subsequent Transformer-based classifier [8], trained with Focal Loss [9] and inherently stabilized by Layer Normalization [12] and residual connections [13], to effectively learn from the prepared data.

IV. PROPOSED METHOD

A. Model Overview

The proposed model enhances malware detection and classification in memory environments, particularly under imbalanced data, feature noise, and diverse process behaviors. It consists of three sequential stages, as shown in Figure 1.

- **Stage 1: Dimensionality reduction and data standardization.** XGBoost evaluates feature importance using the Gain metric [4, 11], and SelectFromModel retains the most significant features to reduce noise and improve training efficiency. StandardScaler then applies Z-score standardization (zero mean and unit variance) on the retained features to improve optimization stability and convergence.

- **Stage 2: Data balancing and cleaning.** A combination of SMOTE, which interpolates new samples for the minority class, and ENN, which removes noisy instances from the majority class, is applied. This hybrid approach augments underrepresented data and cleans the dataset, enhancing the model's learning capacity and generalization [5].

- **Stage 3: Optimized Transformer Classifier Training.** A Transformer-based classifier [8] is utilized to learn high-level representations from the preprocessed fixed-length tabular features. Since each sample is a structured vector with consistent feature semantics, the encoder is used to extract robust non-linear representations from fixed-length tabular features, without assuming any temporal ordering or sequence structure. In this configuration, the Transformer Encoder primarily functions as a stabilized residual feature transformation block rather than a sequence modeling architecture. Specifically, each sample is projected into a dense embedding through a learnable linear layer, and the embedding is then refined by the encoder blocks before the final classification head. The detailed training strategy (Focal Loss, Scheduler, and Early Stopping) is presented in Section IV.D.

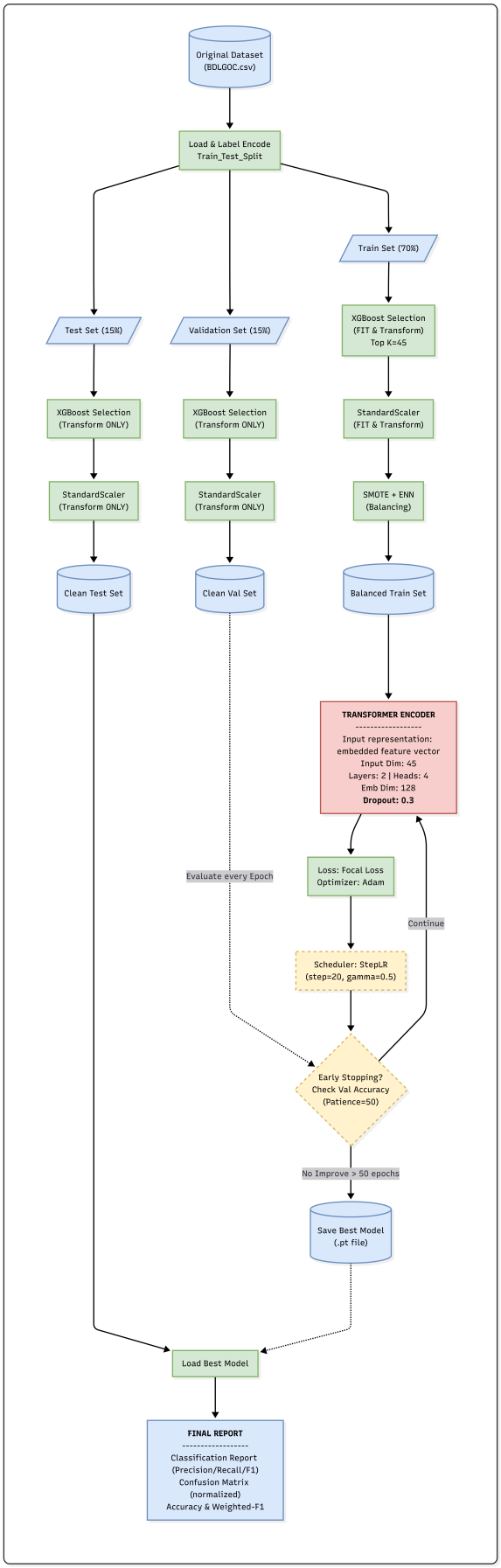


Figure 1. Hybrid pipeline framework detailed architecture

B. *Stage 1: Feature dimensionality reduction and normalization*

In the first stage, raw data is preprocessed to prepare for subsequent steps. The objective of this stage is to reduce data dimensionality, eliminate less informative features, and normalize the remaining ones to ensure effective model training. This process comprises two main steps: feature selection using XGBoost and normalization using Z-score.

Feature selection removes low-discriminative features, reducing dimensionality and model complexity. Subsequent normalization places all features on comparable scales, preventing bias toward large-magnitude variables and improving optimization stability.

1. *Feature Selection using XGBoost*

XGBoost is used to evaluate feature importance via the Gain metric, which reflects the loss reduction when a feature splits a decision tree [4, 11]. Let the initial input be the feature matrix $X \in \mathbb{R}^{N \times d}$, where:

- N is the number of training samples,
- d is the number of original features,
- the corresponding labels are $Y \in \{0, 1, \ldots, C-1\}^N$, with C being the number of classes.

Feature selection is performed on the raw training data before normalization and data balancing, based on the Gain metric, defined as:

$$\text{Gain}(j) = \frac{1}{N_j} \sum_{\text{splits on } j} \Delta\text{Impurity},$$

where:

- $\text{Gain}(j)$ measures the loss improvement when feature X_j is used for splitting,
- N_j is the number of times feature X_j is selected for splitting across all trees,
- ΔImpurity denotes the reduction in node impurity.

Once the Gain values for all features are computed, the features are ranked in descending order, and the top n most important features are retained for training using the selection mechanism:

$$X' = \text{SelectFromModel}(X, \text{Gain}, \text{max\_features} = n)$$

This selection process reduces input dimensionality, enhancing both the efficiency and accuracy of the subsequent Transformer-based classifier.

2. Z-score Normalization

After selecting the most informative features, the input data is standardized using the Z-score method, bringing all features to a normalized distribution with a mean of 0 and a standard deviation of 1. This normalization plays a crucial role in reducing scale discrepancies among features, thereby preventing the learning algorithm from being biased toward features with large absolute values [3]. The Z-score transformation is defined as:

$$x'_{ij} = \frac{x_{ij} - \mu_j}{\sigma_j},$$

where:

$$\mu_j = \frac{1}{N}\sum_{i=1}^{N} x_{ij}, \quad \sigma_j = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_{ij} - \mu_j)^2}.$$

Here, μ_j is the mean of the jth feature over the entire dataset, and σ_j is the standard deviation of the jth feature, reflecting the degree of data dispersion around the mean.

This transformation ensures that each feature follows a standardized distribution with mean 0 and standard deviation 1, thereby eliminating the influence of differing units or scales and accelerating convergence during model training.

C. Stage 2: Data balancing using SMOTE and ENN

Class imbalance in the dataset causes machine learning models to be biased toward majority classes, reducing detection performance for rare but highly dangerous malware types. To address this issue, we employ a two-step resampling strategy combining SMOTE and ENN.

First, SMOTE is applied to generate new samples for minority classes by interpolating between existing data points. This method enhances the representation of underrepresented classes, thereby improving the model's generalization capability [5].

Next, ENN is used to remove noisy or ambiguous samples, particularly from majority classes. Specifically, for each sample x_i, the k nearest neighbors are examined. If the label of x_i differs from the majority label among its neighbors, the sample is removed from the training set. This approach cleans the data and produces clearer class boundaries [5].

The combination of SMOTE and ENN yields a more balanced, less noisy training set, thereby improving classification performance across all classes in the malware detection task.

D. Stage 3: Optimized Transformer classifier training and hyperparameter optimization

After dimensionality reduction and data balancing, the proposed model is employed for training and optimization. Because the pipeline outputs a structured, fixed-length tabular vector with consistent feature semantics, the Transformer Encoder can be applied without requiring any temporal sequence assumption. In this work, the encoder blocks leverage the architecture's deep Feed-Forward Networks and Layer Normalization to extract robust non-linear representations from the denoised and standardized tabular input. The model's architecture is based on the Transformer Encoder [8] to learn robust representations from the preprocessed tabular features produced by the preceding pipeline stages. Each sample is projected into a dense embedding through a learnable linear layer, and the embedding is then refined by the encoder blocks before the final classification head.

The architecture comprises multiple Transformer Encoder layers, each incorporating Layer Normalization, Dropout, and residual connections to enhance stability during training.

Specifically, hyperparameters (e.g., learning rate, dropout rate, number of Transformer layers, and embedding dimension) were set empirically based on preliminary experiments. Model selection during training is performed via Early Stopping on the validation accuracy.

During training, several auxiliary techniques are integrated to improve learning efficiency and mitigate overfitting, including:

- **Focal Loss:** directs the model's focus toward hard-to-classify samples, beneficial in imbalanced datasets.
- **Early Stopping:** halts training when performance on the validation set shows no

improvement over consecutive epochs, preventing overfitting.
- **StepLR Scheduler:** periodically reduces the learning rate, facilitating more effective convergence.

Through the combination of a Transformer-inspired encoder architecture and a carefully designed training strategy, the model achieves robust classification performance in complex, imbalanced multi-class scenarios such as CIC-MalMem-2022.

V. EXPERIMENTS AND EVALUATION

A. Dataset and experimental environment

In this study, we use the CIC-MalMem-2022 dataset, which contains 58,596 memory-behavior samples labeled into one benign class and 15 malware classes. The benign class comprises 29,298 files, whereas each malware family has fewer than one-tenth as many. Malware families and counts are: Transponder (2,410), Gator (2,200), Shade (2,128), CWS (2,000), Scar (2,000), 180Solutions (2,000), Ako (2,000), Refroso (2,000), Conti (1,988), Emotet (1,967), Maze (1,958), Zeus (1,950), Pysa (1,717), Reconyc (1,570), and TIBS (1,410).

We performed experiments within the Google Colab Pro environment (Python 3), utilizing a high-RAM runtime equipped with 51.0 GB of system RAM and an NVIDIA T4 GPU (15.0 GB of VRAM).

B. Experimental Implementation

In the first stage of the pipeline, the XGBoost algorithm is employed to assess feature importance and select the most influential features for model learning. Based on the Gain metric, the top 45 most important features are retained to reduce noise and optimize the input space. The choice of 45 features is determined after experimenting with multiple values, demonstrating this setting as the optimal balance between performance and model stability. Due to the low randomness of XGBoost when using a fixed random_state, the selected feature set is highly repeatable and reliable across runs.

The dataset is split into three subsets: 70% for training (41,017 samples), 15% for validation (8,789 samples), and 15% for independent testing (8,790 samples). Crucially, all subsequent preprocessing steps are performed only on the training set to prevent data leakage and ensure objective evaluation. First, XGBoost feature selection and StandardScaler are fit on the training data, and then applied to the validation and testing sets. Finally, the SMOTE-ENN data balancing technique [5] is applied exclusively to the processed training set. This combined approach increases the number of samples for minority classes while removing noisy samples from majority classes, there by enhancing the model's discriminative capability. The changes in sample counts after each stage are illustrated in Figure 2.

The proposed model is trained for up to 500 epochs using the Focal Loss objective. To ensure successful training and prevent overfitting, we employ an Early Stopping mechanism, which halts training after 50 consecutive epochs without improvement in the Validation Accuracy.

C. Results and Discussion

1. Training Stability and Convergence Analysis

To validate the reliability of the training process, we first analyze the model's learning dynamics. Figure 3 illustrates the progression of Training and Validation Loss over 500 epochs. The curves exhibit a steady descent and converge effectively, with a minimal gap between training and validation loss, indicating that the model successfully learns discriminative patterns without suffering from significant overfitting.

Figure 4 monitors the Validation Accuracy and F1-score throughout training. Early Stopping is triggered based on validation accuracy (patience = 50), while validation F1-score is monitored and reported for a balanced assessment. Both metrics show consistent improvement and stabilization, confirming the efficacy of the proposed optimization strategy (StepLR and Focal Loss).

2. Comparative Performance Analysis

The quantitative performance of the proposed framework on the independent test set is presented in Figure 5, which provides a comparative evaluation against existing state-of-the-art methods on the CIC-MalMem-2022 dataset.

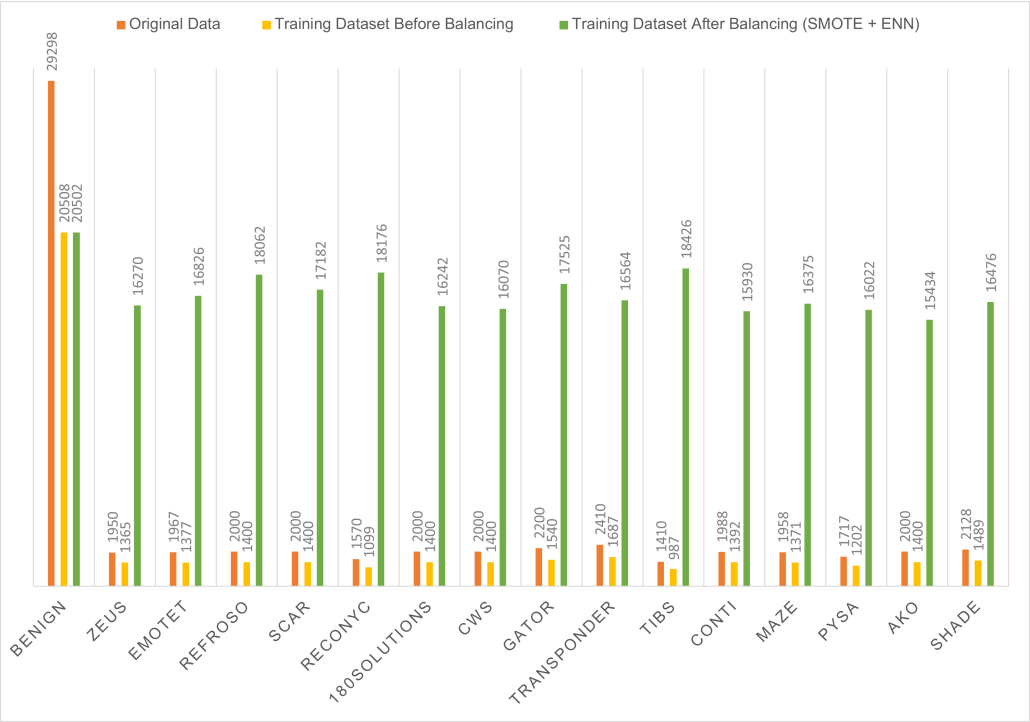


Figure 2. Class distribution before and after Balancing (SMOTE-ENN) on training set

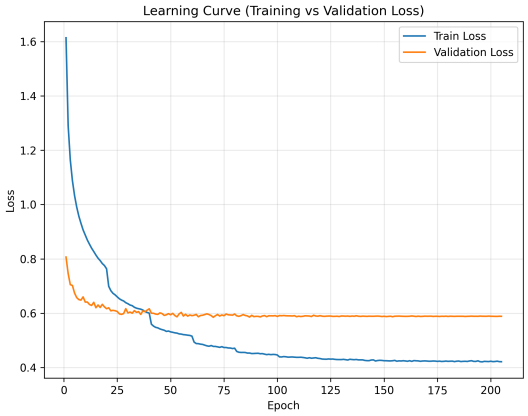


Figure 3. Training and Validation Loss over epochs, demonstrating model convergence

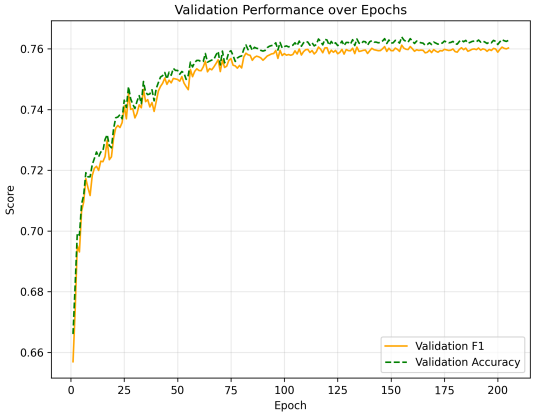
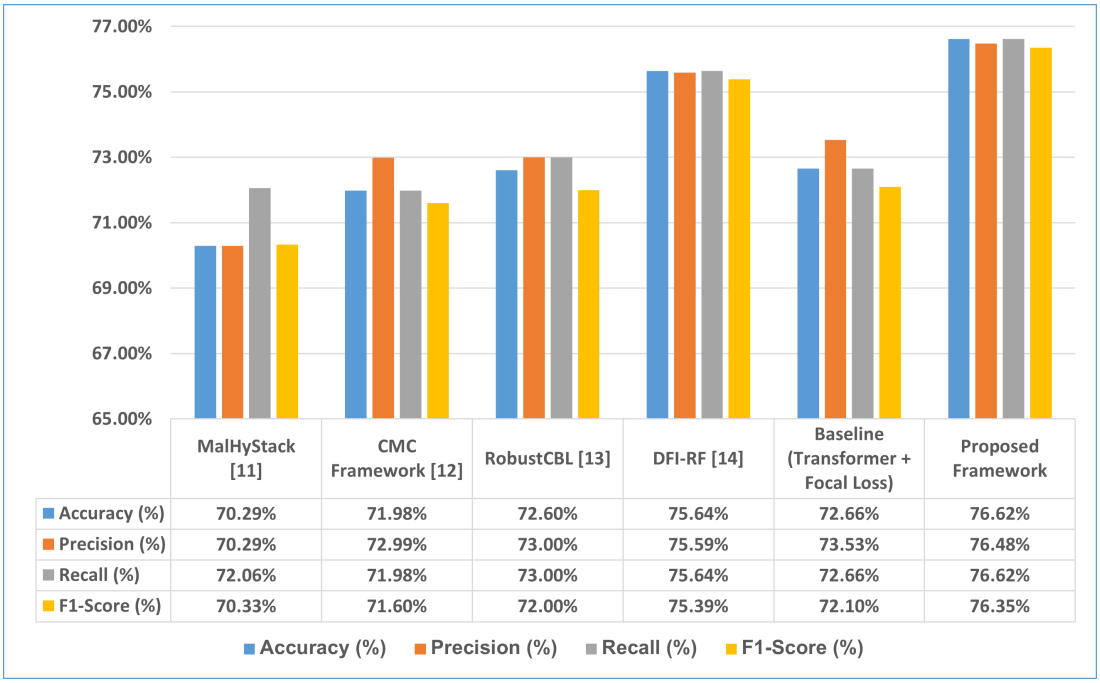


Figure 4. Validation Accuracy and F1-score monitoring for Early Stopping

As illustrated, our framework demonstrates superior efficacy, particularly in terms of the F1-score, outperforming traditional baselines and recent ensemble methods. Specifically:

- Compared to the previous state-of-the-art model, DFI-RF, our proposed framework achieves a consistent improvement, with the F1-score increasing by 0.96 percentage points (from 75.39% to 76.35%) and Accuracy increasing by 0.98 percentage points (from 75.64% to 76.62%).
- Conventional models such as MalHyStack and RobustCBL exhibited limited capability in distinguishing complex malware families, yielding F1-scores fluctuating approximately around 70%.
- The CMC Framework, despite incorporating feature selection (AFSP) and imbalance handling, attained an F1-score of only 71.6%, which is significantly inferior to the performance of our approach [18].

Furthermore, to isolate the impact of the data processing pipeline, we performed an ablation study comparing the full framework against a baseline (Transformer + Focal Loss without XGBoost feature selection and without SMOTE-ENN balancing; only Z-score



	MalHyStack [11]	CMC Framework [12]	RobustCBL [13]	DFI-RF [14]	Baseline (Transformer + Focal Loss)	Proposed Framework
Accuracy (%)	70.29%	71.98%	72.60%	75.64%	72.66%	76.62%
Precision (%)	70.29%	72.99%	73.00%	75.59%	73.53%	76.48%
Recall (%)	72.06%	71.98%	73.00%	75.64%	72.66%	76.62%
F1-Score (%)	70.33%	71.60%	72.00%	75.39%	72.10%	76.35%

Figure 5. Performance comparison of proposed framework and existing models on CIC-MalMem-2022

standardization is applied for numerical stability). As shown in Figure 5, the baseline achieved an F1-score of 72.10%, whereas the full Proposed Framework reached 76.35%. This 4.25% improvement substantiates the necessity of the hybrid pipeline, confirming that feature selection and SMOTE-ENN are critical for optimizing the input space before classification.

The robustness of our framework can be attributed to the synergistic integration of three key factors:

- **Deep Representation Learning:** The deployment of a Transformer Encoder-based classifier [8], stabilized by Layer Normalization [12] and residual connections [13], enables the extraction of high-level non-linear representations from the tabular feature space.
- **Imbalance Mitigation:** The utilization of Focal Loss [9] effectively addresses class imbalance by reshaping the loss function to prioritize hard-to-classify examples from minority classes.
- **Data Integrity Enhancement:** The application of SMOTE and ENN [5] synthesizes representative minority samples while eliminating noise from majority classes, thereby establishing a clearer decision boundary for the classifier.

These enhancements enable the model to achieve not only high accuracy but also a well-balanced trade-off between Precision and Recall. This balance is critical in multi-class malware detection, where the data distribution is inherently uneven, affirming the practical viability of the proposed approach.

VI. CONCLUSION AND FUTURE WORK

This study proposes a hybrid malware detection framework that effectively integrates a data processing pipeline utilizing XGBoost [10] and SMOTE-ENN [5] with a robust Transformer Encoder classifier [8] trained with Focal Loss [9].

Experimental results on the CIC-MalMem-2022 dataset demonstrate the effectiveness of the proposed framework, achieving an accuracy of 76.62% and an F1-score of 76.35%, outperforming many previous approaches. The proposed method effectively addresses key challenges in malware detection, including data imbalance, noise, and diversity in attack behaviors, while maintaining flexibility and strong generalization capability for tabular data.

However, the current evaluation is limited to CIC-MalMem-2022. Future work will extend experiments to additional datasets and real-world environments to better validate the model's generalization capability.

While our Proposed Framework achieves the best overall performance, lightweight models like XGBoost or shallow CNNs may be more practical in resource-limited settings. Future work will systematically compare such baselines to analyze the accuracy–efficiency trade-off.

In the future, potential research directions include:

- Performing ablation studies for each pipeline component (XGBoost feature selection, SMOTE-ENN, Focal Loss) to quantify their individual contributions to Accuracy/F1.
- Deploying and evaluating the model in real-world environments such as IoT/Edge.
- Incorporating semi-supervised or continual learning approaches to enable rapid adaptation to new malware variants.
- Leveraging self-supervised pre-training on large benign memory datasets (e.g., denoising or contrastive learning) to learn foundation representations before fine-tuning on labeled malware data.
- Expanding to larger and more diverse datasets to validate generalization performance.
- Optimizing architecture and training processes to reduce computational cost, enabling efficient deployment in resource-constrained environments.

Overall, this study provides an effective and robust approach to intelligent malware detection, particularly well-suited for rapid and detailed behavioral analysis in modern security systems.

REFERENCES

[1] A. Bensaoud, J. Kalita, and M. Bensaoud, "A survey of malware detection using deep learning," *Machine Learning with Applications*, vol. 16, p. 100546, Jun. 2024. doi: doi.org/10.1016/j.mlwa.2024.100546.

[2] T. Leng, Y. Pan, L. Zhao, A. Yu, Z. Zhu, L. Cai, and D. Meng, "MemInspect: Memory Forensics for Investigating Fileless Attacks," in *Proceedings of the IEEE 22nd International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, Exeter, United Kingdom, 2023, NJ: IEEE, pp. 946–955. doi: doi.org/10.1109/TrustCom60117.2023.00134.

[3] P. Maniriho, A. N. Mahmood, and M. J. M. Chowdhury, "MeMalDet: A memory analysis-based malware detection framework using deep autoencoders and stacked ensemble under temporal evaluations," *Computers & Security*, vol. 142, p. 103864, 2024. doi: doi.org/10.1016/j.cose.2024.103864.

[4] T. Carrier, P. Victor, A. Tekeoglu, and A. H. Lashkari, "Detecting Obfuscated Malware using Memory Feature Engineering," in *Proceedings of the 8th International Conference on Information Systems Security and Privacy (ICISSP)*, Online Streaming, 2022, Portugal: SCITEPRESS, pp. 177–188. doi: doi.org/10.5220/0010908200003120.

[5] S. R. Safavian and D. Landgrebe, "A Hybrid SMOTE-ENN Method for Malware Detection in Imbalanced Datasets," *Computers & Security*, vol. 128, p. 102931, 2023. doi: doi.org/10.1016/j.cose.2023.102931.

[6] A. Gaur, P. Mishra, P. Vinod, A. Singh, V. Varadharajan, U. Tupakula, and M. Conti, "vDefender: An explainable and introspection-based approach for identifying emerging malware behaviour at hypervisor-layer in virtualization environment," *Computers and Electrical Engineering*, vol. 120, Part B, p. 109742, 2024. doi: doi.org/10.1016/j.compeleceng.2024.109742.

[7] V. Borisov, T. Leemann, K. Seßler, J. Haug, M. Pawelczyk, and G. Kasneci, "Deep Neural Networks and Tabular Data: A Survey," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 35, no. 6, pp. 7499–7515, 2024. doi: doi.org/10.1109/TNNLS.2022.3229161.

[8] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention Is All You Need," in *Proceedings of the 31st Conference on Neural Information Processing Systems (NIPS)*, Long Beach, CA, USA, 2017, NY: Curran Associates, Inc., pp. 5998–6008. doi: doi.org/10.48550/arXiv.1706.03762.

[9] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal Loss for Dense Object Detection," in *2017 IEEE International Conference on Computer Vision (ICCV)*, Venice, Italy, 2017, NJ: IEEE, pp. 2999–3007. doi: doi.org/10.1109/ICCV.2017.324.

[10] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Francisco, CA, USA, 2016, NY: ACM, pp. 785–794. doi: doi.org/10.1145/2939672.2939785.

[11] K. S. Roy, T. Ahmed, P. B. Udas, M. E. Karim, and S. Majumdar, "MalHyStack: A hybrid stacked ensemble learning framework with feature engineering schemes for obfuscated malware analysis," *Intelligent Systems with Applications*, vol. 20, p. 200283, 2023. doi: doi.org/10.1016/j.iswa.2023.200283.

[12] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *arXiv preprint* arXiv:1607.06450, 2016. doi: doi.org/10.48550/arXiv.1607.06450.

[13] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, USA, 2016, NJ: IEEE, pp. 770–778. doi: doi.org/10.1109/CVPR.2016.90.

[14] Y. Song, D. Zhang, J. Wang, Y. Wang, Y. Wang, and P. Ding, "Application of deep learning in malware

detection: a review," *Journal of Big Data*, vol. 12, no. 99, pp. 1–23, 2025. doi: doi.org/10.1186/s40537-025-00999-6.

[15] A. Galli, V. La Gatta, V. Moscato, M. Postiglione, and G. Sperlì, "Explainability in AI-based behavioral malware detection systems," *Computers & Security*, vol. 143, p. 103842, 2024. doi: doi.org/10.1016/j.cose.2024.103842.

[16] M. M. Alani, A. Mashatan, and A. M. Miri, "XMal: A lightweight memory-based explainable obfuscated-malware detector," *Computers & Security*, vol. 133, p. 103409, 2023. doi: doi.org/10.1016/j.cose.2023.103409.

[17] T. A. Tuan, P. S. Nguyen, P. N. Van, N. D. Hai, P. D. Trung, N. T. K. Son, and H. V. Long, "A novel framework for cross-platform malware detection via AFSP and ADASYN-based balancing," *Computers and Electrical Engineering*, vol. 128, p. 110625, 2025. doi: doi.org/10.1016/j.compeleceng.2025.110625.

[18] P. S. Nguyen, P. N. Van, H. V. Long, and P. D. Trung, "An Efficient Framework for Multi-Class Malware Classification in Cloud Environments," *Journal of Science and Technology on Information Security*, vol. 1, no. 24, pp. 1–10, 2023. doi: doi.org/10.54654/isj.v1i24.1092.

[19] M. M. Abualhaj, S. Al-Khatib, N. Al Shafi, I. Qaddara, and A. Hyassat, "Utilizing gray wolf optimization algorithm in malware forensic investigation," *Journal of Computer and Cognitive Engineering*, vol. 00, no. 00, pp. 1–12, 2025. doi: doi.org/10.47852/bonviewJCCE52025053.

[20] S. S. Shafin, G. Karmakar, and I. Mareels, "Obfuscated memory malware detection in resource-constrained IoT devices for smart city applications," *Sensors*, vol. 23, p. 5348, 2023. doi: doi.org/10.3390/s23115348.

About the Authors

Le Phu Minh
Workplace:
Academy of Security Technology and Engineering, Vietnam
Email: lephuminht07@gmail.com
Education: Studying in depth about Information Security at the Academy of Security Technology and Engineering, Vietnam.
Recent research interests: Mathematical modeling, Machine Learning, Deep Learning, Artificial Intelligence and applications in Cybersecurity.

Tên tác giả: **Lê Phú Minh**
Cơ quan công tác: Học viện Kỹ thuật và Công nghệ an ninh, Việt Nam
Email: lephuminht07@gmail.com
Quá trình đào tạo: Theo học chuyên sâu về An toàn thông tin tại Học viện Kỹ thuật và Công nghệ an ninh, Việt Nam.
Hướng nghiên cứu hiện nay: Mô hình toán học, học máy, học sâu, trí tuệ nhân tạo và ứng dụng trong lĩnh vực an toàn thông tin.

Do Dinh Quang
Workplace:
Academy of Security Technology and Engineering, Vietnam
Email: quangkma@gmail.com
Education: Graduated with an Engineering degree in Information Security from the Vietnam Academy of Cryptography Techniques, Hanoi, in 2011; received an MSc in Information Security from the same institution in 2017.
Recent research interests: Applying Machine Learning, Deep Learning, and Artificial Intelligence in cybersecurity to detect and prevent malware, enhance digital security, and strengthen system defenses against evolving cyber risks.

Tên tác giả: **Đỗ Đình Quang**
Cơ quan công tác: Học viện Kỹ thuật và Công nghệ an ninh, Việt Nam
Email: quangkma@gmail.com
Quá trình đào tạo: Nhận bằng Kỹ sư An toàn thông tin (2011) – Học viện Kỹ thuật Mật mã; Thạc sĩ An toàn thông tin (2017) – Học viện Kỹ thuật Mật mã.
Hướng nghiên cứu hiện nay: Ứng dụng học máy, học sâu và trí tuệ nhân tạo trong an toàn thông tin; phát hiện/ngăn chặn mã độc; tăng cường phòng thủ hệ thống trước rủi ro mạng tinh vi.

Hoang Viet Long
Workplace:
Academy of Security Technology and Engineering, Vietnam
Email: longhv08@gmail.com
Education: Received PhD diploma in Computer Science at Hanoi University of Science and Technology in 2011, specializing in fuzzy computing and soft computing techniques with applications to electronics engineering. He has been promoted to Associate Professor since 2018.
Recent research interests: Mathematical modeling, Machine Learning, Deep Learning with applications in Cybersecurity.

Tên tác giả: **Hoàng Việt Long**
Cơ quan công tác: Học viện Kỹ thuật và Công nghệ an ninh, Việt Nam
Email: longhv08@gmail.com
Quá trình đào tạo: Nhận bằng Tiến sĩ chuyên ngành Khoa học máy tính tại Trường Đại học Bách Khoa Hà Nội năm 2011, chuyên sâu về fuzzy computing và soft computing trong kỹ thuật điện tử. Được phong học hàm Phó Giáo sư từ năm 2018.
Hướng nghiên cứu hiện nay: Các mô hình toán học, học máy, học sâu và ứng dụng trong an toàn thông tin.

Nguyen Thi Kim Son
Workplace: School of Information and Communications Technology, Hanoi University of Industry, Vietnam
Email: sonntk@haui.edu.vn
Education: Received her Ph.D. from Hanoi National University of Education in 2010 and was appointed Associate Professor in 2019.
Recent research interests: Data science, information systems, and artificial intelligence in education, underpinned by a strong foundation in applied mathematics.

Tên tác giả: **Nguyễn Thị Kim Sơn**
Cơ quan công tác: Khoa Công nghệ Thông tin và Truyền thông, Trường Đại học Công nghiệp Hà Nội
Email: sonntk@haui.edu.vn
Quá trình đào tạo: Tiến sĩ – Trường ĐH Sư phạm Hà Nội (2010); Phó Giáo sư (2019).
Hướng nghiên cứu hiện nay: Khoa học dữ liệu, hệ thống thông tin và trí tuệ nhân tạo trong giáo dục, trên nền tảng vững chắc về Toán ứng dụng.