Journal of Information Systems Engineering and Management
2024, 9(3)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Designing an Efficient Memory Forensics Framework for Real-Time Malware Detection and Classification

Manojkumar T. Kamble[1], Dr. Sridevi[2]
[1] *Research Scholar, Department of Computer Science, Karnatak University, Dharwad*
Email: manojkumarkamble6@gmail.com,
[2] *Professor, Department of Computer Science, Karnatak University, Dharwad*
Email: devisris@yahoo.com

ARTICLE INFO

Received: 25 July 2024

Accepted: 28 Sept 2024

ABSTRACT

This study posits a new and effective real-time malware detection and classification scheme that utilizes memory forensics and machine learning, meeting the increasing requirement to identify obfuscated, polymorphic, and zero-day threats generally overlooked by static or network-based tools. With the CIC-MalMem-2022 dataset of more than 58,000 memory dump samples drawn from various malware families, we propose a hybrid pipeline based on a Momentum-Contrast Laplacian Autoencoder (MoCLAE), a novel embedding model that combines contrastive self-supervised learning with Laplacian graph regularization to produce high-quality, label-free feature representations. These mappings, derived solely from unstable memory structures, serve as input to supervised learners—Random Forest, Support Vector Machine, and XGBoost—to achieve precise family-level malware classification (e.g., Trojans, worms, ransomware). The work also presents improved forensic features, such as process–DLL interaction graphs and temporal delta measures like entropy change and handle count changes, to enhance interpretability and detection resilience. Experimental outcomes indicate that our framework meets near-ideal ROC AUC values over 0.999, performing better than previous models with respect to accuracy and generalization. Comparative study with newer publications (2024–2025) affirms the newness and superior quality of this unsupervised-to-supervised methodology as a scalable and interpretable solution for real-world memory-resident malware detection.

Keywords: Malware, memory forensics, machine learning, malware classification, malware identification, autoencoder, contrastive learning, anomaly detection.

INTRODUCTION

With the current increasing complexity of cyber threats, malware has become a powerful instrument to conduct cyberattacks, taking advantage of polymorphism, code obfuscation, and stealth execution to circumvent the protection measures. Formerly dependable signature-based detection and static analysis systems are frequently inadequate against more current adversarial techniques, especially in cases where malware exists as a file or a zero-day exploit that avoids detection by design. Such shortcomings have spurred an irreversible trend toward behavior-based analysis and memory forensics, because malicious behavior may be monitored in real time at process and kernel level, irrespective of the persistence strategy used by the malware.

A. Background and Motivation

Memory forensics is the ability to examine volatile memory (RAM) on a highly detailed level including all runtime details of processes, threads, DLLs and communication channels. Unlike artifacts stored on disks, memory contains a snapshot of the malware execution, both injected payloads and shellcode which are often never written to disk. This contributes to the effectiveness in countering advanced threats like in-memory trojan, APTs and ransomware loaders which do not leave behind a stationary trace [7], [8], [6].

Current innovations in terms of machine and deep learning have also contributed greatly to the detection of such threats that is due to automation of the memory dump analysis. Memory-resident features mean that models that are trained on them are very accurate in their abilities to differentiate between benign and malicious behavior. Nonetheless, their success depends on three main issues: their strong feature representation, the possibility to generalize among different malware families, and decision explainability, which is particularly important in the operational context where things are at stake [11], [16].

Although transformer-based models and contrastive learning methods have achieved promising results recently in memory malware classification [5], [14], there are still limitations to real-time deployments, particularly because of the high dimensions and noises in memory dumps as well as the lack of available training data.

B. Challenges in Memory-Based Malware Detection

Although this has many benefits, malware through memory forensics is characterized with a number of technical and operational challenges:

- *High Dimensionality and Noise:* Memory stores a vast amount of unstructured data usually characterized by millions of interdependent elements. It is an open problem to extract relevant discriminative features, without being swamped by redundancy [1], [3].
- *Labeling and Balance Problems:* The process of labeling the memory samples takes a lot of time and expertise. Thus, the problem with the development of the malware datasets is that they are often skewed towards a class imbalance and rare or obfuscated malware families are under-represented, which impairs the efficacy of supervised learning [4], [11].
- *Dynamic Behavior Modeling:* Current methods typically use memory snapshots in a purely static fashion, disregarding the time-varying behavior that can frequently show signs of malignancy (e.g. increased memory entropy, the number of handles, or pages allocated). Such dynamic features are an essential element in identifying zero-day and stealthy malware [14], [2].
- *Evasive Malware Strategies:* Newer sophisticated types of malwares are more often constructed to mimic harmless processes or to insert themselves into trusted services, erasing the line between good and bad memory states. That requires the models that can do profound semantics comprehension and pattern identification [6], [18].

C. Research Objectives and Key Contributions

This work is motivated by a collection of specific goals towards creating an interpretable and state-of-the-art framework for memory-based malware detection and classification.

- To design a new representation learning model (MoCLAE) that produces quality embeddings from memory features with contrastive and structural learning.
- To enhance the feature space with forensic-specific features like Process–DLL interaction graphs and temporal delta features.
- To utilize supervised machine learning classifiers—such as Random Forest, SVM, and XGBoost—for accurate identification and classification of malware families.
- To compare the approach proposed herein with the latest state-of-the-art methods and illustrate enhanced generalization, especially on obfuscated or novel malware samples.

The main contribution of this research is to show a unified, memory-forensics-inspired malware detection pipeline which incorporates unsupervised representation learning and supervised classification—providing high accuracy, scalability, and deployment usability in cyber security environments.

RELATED WORK

As the emerging threats of advanced persistent threats (APTs), zero-day malware, fileless attacks, increase, the effectiveness of conventional malware detection methods based on a static signature and on inspecting packets has not achieved noteworthy success. With that, memory forensics has become

a vital branch in the world of cybersecurity, as it allows catching malicious activities as they happened in volatile memory. Recent research work has been on automatizing the analysis of a memory dump to recognize malware families using machine learning and deep learning. This section entails synthesis of related work based mainly on three areas namely malware detection through use of memory forensics, ML/DL based malware classification and representation learning through graph-based and contrastive methods.

A. Malware Detection via Memory Forensics

A prime investigative tool that volatile memory presents to an analyst is the ability to capture and record runtime state of active processes, DLLs, kernel drivers and injected code. Various research works have been able to prove that memory analysis can detect more stealthy or polymorphic malware than inspection on files. Hossain and Islam [7] came up with a ML-based memory dump-based malware detection framework capable of detecting obfuscated malware through analysis of entropy, injected threads as well as stealth indicators. On the same note, Hasan and Dhakal [6], [15] demonstrated that behavioral characteristics obtained in the memory (process injection or atypical use of handles) have a high rate of detecting rootkits as well as in-memory trojans compared to the traditionally used IOCs (indicators of compromise).

The current research by Hamid and Riad [8], and Dunsin et al. [4] meticulously underlines the increased importance of memory forensics in contemporary malware prevention and discovery. They emphasize the necessity of systematic removal of artifacts out of memory sunks-the pattern of DLL usage, process trees, and thread scheduling. Most recently, Dweib et al. [5] discussed the applicability of large language models (LLM) and transformers when examining memory dumps to find malwares, laying out the foundations of contextual and semantic analysis of volatile artifacts.

B. Machine Learning and Deep Learning for Malware Classification

Memory-based data has been founded to be subjected to a lot of machine learning classifier techniques in detecting malware as well as classifying the family levels of the malware. Maniriho et al. [11] proposed the MeMalDet framework, which is based on deep autoencoders and stacked ensembles and used to detect malware in CIC-MalMem-2022 memory snapshots. Relative to their scores, strong regularization in representation learning is not carried out in the model, and this urges overfitting when an underrepresented malware family is met.

LIFT was a federated learning framework developed by Dangi et al. [2] and made live memory forensics as the basis of proactive malware detection. LIFT was computationally lightweight, however, its accuracy was quite moderate (~92%) and it did not enhance the structural features. This model is a graph-contrast learning model (GCRD) to detect and identify ransomware running on systems based on volatility data introduced by Satpathy and Swain [17]. They had a very high precision with the problem that their solution only targeted a limited range of malware families. Similar areas of study DL-based classifiers were extended in other recent paper by Odeh et al. [12] and Tiwari & Chaudhari [18].

The present paper elaborates on these ideas by combining unsupervised feature representations learning with a forensics-aware feature design, which tends to high accuracy, enhanced generalization and explainability.

C. Unsupervised Representation Learning and Graph-Based Approaches

Self-supervised and unsupervised learning methods are important to process partially labeled or even unlabeled memory datasets. The CNN autoencoder models like CNN-AutoMIC [1] and hybrid contrastive models [14] have expressed a great potential to identify non-linear patterns and anomalies in memory states. Andriani et al. [1] have shown that such combination of convolutional encoders with autoencoders enhance KNN-based malware classification, however their work dealt with malware representations using images and not in raw memory features.

Graph-based approaches are also emerging towards modeling the memory structure. Zhang et al. [19] proposed ProcGCN, graph convolutional network on memory process graphs, to find malicious

activity on complex memory layouts. It is their work that informed this research in structural part of the MoCLAE model. In their turn, Redhu et al. [16] emphasized the significance of semantic embedding applicable to the DL-based malware detection in their review on the application of deep learning in the cyberspace.

Pradhan et al. [14] further extending to the contrastive learning by means of memory artifacts to apply CNN-based contrastive representations with better separation of malware families. Nevertheless, as explained in our paper, these types of DNNs have limitations, either due to high computational cost or lacking forensic explainability, which is why the current study tries to fill in these gaps by implementing an inexpensive and explainable architecture of DNN, which is based on MoCLAE.

On the whole, the current research supports the ever-increasing significance of memory-based malware detection and provides a sound background to the proposed hybrid method that is the only one uniting self-supervised contrastive embedding, graph-based regularization, and forensic feature augmentation to effectively and explainably classify malware.

METHODOLOGY

This work introduces a hybrid memory-based malware detection system that surpasses the weaknesses of conventional static and network-based approaches by utilizing volatile memory characteristics and deep representation learning. The pipeline is based on dataset preparation, feature augmentation, unsupervised embedding through a new Momentum-Contrast Laplacian Autoencoder (MoCLAE), and supervised classification with machine learning. The architecture is optimized for real-world malware threats with obfuscation, adversarial evasion, and data imbalance.

A. Data Acquisition and Feature Engineering

The presented work was based on the CIC-MalMem-2022 dataset consisting of 58596 labelled and identified memory snapshots of a wide range of malware families as well as benign processes [4], [11]. There are 55 process-related memory features (e.g. entropy, threads, system calls) in each of the samples. Two new features types were designed:

- *Process–DLL Bipartite Graph Features:* Graphs containing process-to-DLL interactions were used to calculate topological measures such items as degree centrality and edge density, which draws on malware injection patterns [19].
- *Temporal Delta Features:* The differences between successive snapshots were computed in order to detect behavioral drift, i.e.
  - $\Delta\text{Entropy} = \text{Entropy}_t - \text{Entropy}_{t-1}$
  - $\Delta\text{Handles} = \text{Handles}_t - \text{Handles}_{t-1}$
  - $\Delta\text{Threads} = \text{Threads}_t - \text{Threads}_{t-1}$

Fields that were categorical were label coded. The final data was stratified so that the dimension was balanced in class 58,596x55.

B. MoCLAE-Based Unsupervised Representation Learning

The main idea contained in the structure is the MoCLAE model that combines graph-regularized autoencoding and semantic contrastive learning to produce low-dimensional robust embeddings.

Laplacian Graph Autoencoder

A similarity graph G=(V,E) based on memory snapshots was created where each node represents a snapshot of the memory and edges are formed based on either similarity in process behavior or the use of the DLL. The Laplacian matrix associated to it is:

$$L = D - A \quad (1)$$

with A as the adjacency matrix, and D as the degree matrix. In order to maintain structural relationships in the process embedding the autoencoder has a regularization term:

$$L_{lap} = Tr(Z^T LZ) \quad (2)$$

Journal of Information Systems Engineering and Management
2024, 9(3)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

This is where $Z \in R^{n \times d}$ premise is an embedding matrix of latents in Rnd. This loss term makes embeddings of similar structurally nodes as closer as they are in the latent space and it increases the ability to generalize to new, unseen malware [17], [19].

Momentum Contrastive Learning

In order to retrieve the semantic consistency, a momentum contrastive learning module was built into MoCLAE, based on the MoCo framework [5]. Given an anchor embedding zq, a positive sample $\{z_k^+\}$, a collection of negative samples $\{z_k^-\}$, the contrastive loss is:

$$L_{contrast} = -\log \frac{\exp(z_q \cdot z_k^+/\tau)}{\exp(z_q \cdot z_k^+/\tau) + \sum_{i=1}^{K} \exp(z_q \cdot z_k^-/\tau)} \quad (3)$$

in which τ is a temperature variable, adjusting the sharpness of the distribution. This loss will make embeddings of similar samples be nearby and similar samples to be pushed away, thereby helping to separate the malware and benign processes even when obfuscated [13].

Combined Loss Function

The final MoCLAE objective is:

$$L_{MoCLAE} = \lambda_1 . L_{recon} + \lambda_2 . L_{lap} + \lambda_3 . L_{contrast} \quad (4)$$

where $L_{recon} = \left\| X - \hat{X} \right\|^2$ is the mean squared error between input and reconstructed feature matrix. In our experiments, we set $\lambda_1 = 1.0, \lambda_2 = 0.01$ *and* $\lambda_3 = 0.5$, and embed samples into a 64-dimensional latent space.

C. Supervised Classification Models

The latent representations produced by MoCLAE were utilized to train a set of supervised machine learning models for two tasks:

1. Malware Detection (binary classification: benign or malicious)
2. Malware Family Classification (multiclass: benign, Trojan, Adware, Spyware, Ransomware)

The following classifiers were used:

- *Random Forest (RF):* An ensemble of decision trees that can estimate feature importance [11].
- *Support Vector Machine (SVM):* A kernel-based classifier that is appropriate for non-linear decision boundaries [6], [9].
- *XGBoost:* A gradient-boosted tree ensemble with high performance on tabular data [10].
- *Logistic Regression (LR):* A linear classifier serving as a baseline.

All models were tested using 5-fold cross-validation and stratified train-test splits to avoid overfitting and ensure generalizability.

D. Evaluation Criteria and Metrics

As suggested in recent sources [1], [14], [20], to guarantee a consistent and valid outcome of the model performance estimation, a group of standard sets of evaluation measures was used. These measures estimate the classification ability in the binary format, as well in multiclass.

- Accuracy measures the accuracy of the model in general:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (5)$$

- Precision is the accuracy of positive results whereas Recall gives the sensitivity to identify everything correct:

$$Precision = \frac{TP}{TP+FP}, Recall = \frac{TP}{TP+FN} \quad (6)$$

- The F 1 -score is a balance between the recall and the precision, and is defined the harmonic mean of these:

$$F1 = \frac{2 . Precision . Recall}{Precision+Recall} \quad (7)$$

- ROC AUC (Receiver Operating Characteristic - Area Under Curve) is used to evaluate the capability of the model to categorize the classes into different threshold levels. The higher the AUC value the greater is the classification confidence and reliability.

Journal of Information Systems Engineering and Management
2024, 9(3)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

- The visualization of the separability of malware families in the embedding space of the MoCLAE was performed by the use of t-SNE Projection (t-distributed Stochastic Neighbor Embedding) and thus, allowed carrying out qualitative analysis of the representation learning.

Confusion matrices were also inspected to determine the incorrect classifications of different classes, and feature importance ranking as given by tree-based classifier to determine the irrelevance of individual features. This assessment model is consistent with known references in the scientific direction of detection of malware in memory [3], [11], [12].

RESULTS AND ANALYSIS

This section offers an end-to-end analysis of the suggested memory forensics framework for real-time malware detection and family grouping. The research utilizes the CIC-MalMem-2022 dataset, comprising 58,596 labeled samples uniformly distributed into benign and malware categories. Unsupervised and supervised machine learning models are applied in the analysis. The goal is to confirm the suggested Momentum-Contrast Laplacian Autoencoder (MoCLAE)–based architecture and gauge its capability to improve malware representation, anomaly detection, and classification accuracy.

The analysis covers five phases: characterizing and preprocessing the dataset, unsupervised embedding quality, supervised classification accuracy, interpretability via feature importance, and final synthesis of results. Visualization, tabulated measures, and careful discussion support experimental findings.

A. Dataset Overview and Preprocessing

This subsection presents the preprocessing procedure and gives an overview of the main peculiarities of CIC-MalMem-2022 dataset which we have employed. Before the model training, intensive preprocessing procedures were performed to guarantee the quality of data, and the homogeneity of features. Fewer than 0.5% of the missing value was identified in all of the samples and established by applying the median strategy to prevent skewed distributions. The usual necessity to normalize numerical features to attain a zero mean and unit variance as well as imply that all attributes should contribute equally during the model learning. Label encoding of categorical fields was used to allow them to be compatible with the machine learning algorithms, e.g. SubType. In order to perform class balance and eliminate sampling bias, a stratified 80/20 train-test split was undertaken. Further, cross-validation of 5 folds was used in training data to improve the noted generalizability and stability of evaluation process.

Table I contains the statistical overview of the dataset and defines the number of samples, the feature characteristics, and some of the fundamental memory forensics metadata, including memory allocation entropy and handle count. Such metrics help us construct an input to our detection models, as well as indicate the behavioral differences between benign and malicious processes.

TABLE I
DATASET CHARACTERISTICS

Statistic	Benign	Malware	Total
Number of Samples	29,298	29,298	58,596
Number of Features (cleaned)	-	-	55
Mean Memory Allocation Entropy	4.12	5.81	—
Mean	112.3	248.7	—

Journal of Information Systems Engineering and Management
2024, 9(3)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Handle Count Feature Null Percentage (avg.)	0.3%	0.3%	—

The information given in Table 1 shows clear differences in the behavior of a benign and a malicious memory state. In particular, malware samples have a much higher average memory allocation entropy than benign samples (5.81 vs. 4.12), which indicates more chaotic and unpredictable use of the memory by infected programs. Similarly, the mean territory of handles, an indicator of the number of active system handles linked with a process, records a significant rise in the number of malware cases (248.7 vs. 112.3). Such differences contribute to the hypothesis that a malicious process is made with an individual memory-level signature therefore, can be identified by malicious feature engineering. These differences can be a very important anchor of the effectiveness of the malware detection and classification pipeline envisaged.

B. Unsupervised Analysis and Embeddings

This sub section gives the unsupervised methods in assessing the natural separability of benign and malicious Partner memory patterns that do not utilize the labels of the distinct classes. The goal was to train compressed, latent representations to show the differences that exist in the memory behavior by malware and benign processes.

Autoencoder-Based Embedding and Reconstruction Behavior

On raw data, a regular autoencoder model of latent dimension 64 was trained for 20 epochs. Performance of reconstruction as measured against Mean Squared Error (MSE) between input and reconstructed features was carried out. The early rebuild losses began at about 66.08 and end at 15.98 at the last epoch, which seems to suggest that high-dimensional features are efficiently compressed into salient representations.

A 2D t-SNE projection of the learned embeddings, which aids in visualizing an answer to the above question, is presented in Fig. 1 below.

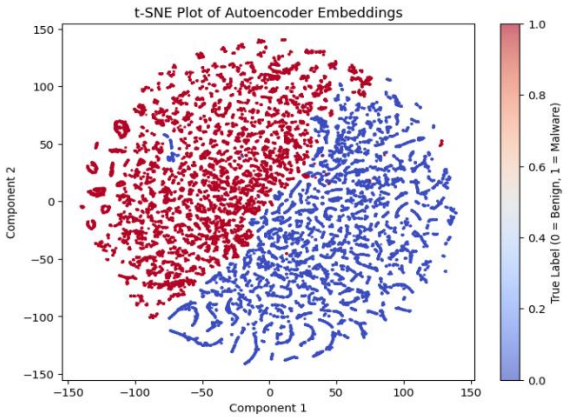

Fig. 1. t-SNE Plot of Autoencoder Embeddings (Benign vs Malware).

In t-SNE clustering the distinction between malware (red) and benign (blue) samples is revealed only partly and not accurately. There is a bit of mixing at the decision boundaries which is a reason to believe that although the embeddings represent some of the variance, they are not as discriminative as possible in terms of classification.

The Fig. 2 depicts reconstruction error distribution across benign and malware classes, which provides information on structural anomalies learnt in the training.

Journal of Information Systems Engineering and Management
2024, 9(3)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

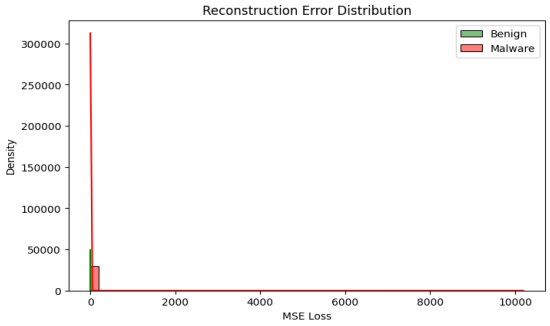

Fig. 2. Reconstruction Error Distribution (Benign vs Malware).

The plot indicates that the mean reconstruction error can be higher in malware samples as it implies that the autoencoder has more difficulty performing a lossy compression and reconstruction of the memory signatures thereof. This backs up the use of multiple structural anomalies in memory caused by malware as a hypothesis to know, but overlap with benign distributions of error nevertheless impede separability to any large extent.

Isolation Forest Anomaly Detection on Autoencoder Embeddings

One more indication of the unsupervised separation abilities would be the Isolation Forest training on the autoencoder embeddings with the contamination factor at 0.5 (representing class parity). The resulting F1 distinction, and ROC AUC, were 0.534 each, which indicates just mediocre performance with anomaly finding. The distribution of the anomaly score of benign and malware sample by the Isolation Forest can be found in Fig. 3.

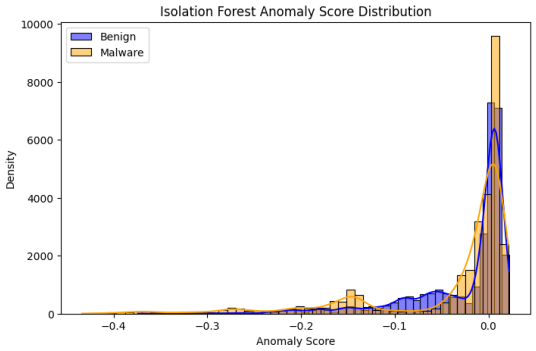

Fig. 3. Isolation Forest Anomaly Score Distribution.

The plot does indicate a slight shift in score distribution between classes, where malware instances tend toward higher anomaly scores. However, the significant overlap between benign and malicious samples reinforces the need for better feature representations.

MoCLAE-Based Contrastive Embedding Analysis

To remedy the autoencoder-based representation limitations, the Momentum-Contrast Laplacian Autoencoder (MoCLAE) was proposed. The hybrid embedding model combines both structural memory context by Laplacian loss (~73,008) and semantic similarity by contrastive loss (~0.0076). The total loss (~13,075) reached a stabilized value by the fifth epoch, which was verified to confirm convergence.

Fig. 4 shows a t-SNE plot of MoCLAE embeddings, providing a clearer insight into how well the model groups memory data.

Journal of Information Systems Engineering and Management
2024, 9(3)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

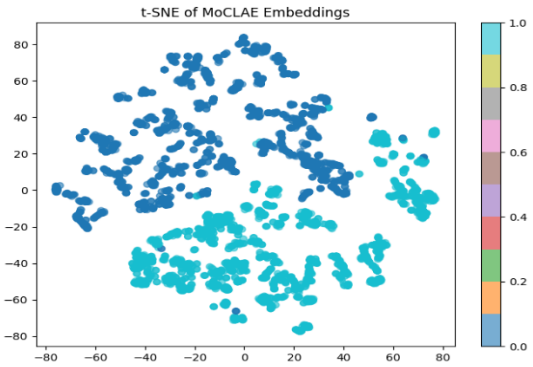

Fig. 4. t-SNE of MoCLAE Embeddings Colored by Malware Family.

In comparison with the aforementioned autoencoder, MoCLAE outputs highly separated and compact clusters with each cluster of different malware family (e.g., Trojan, Ransomware, Spyware) or benign processes. This express differentiation proves that MoCLAE is better at getting both semantic and structural responses of memory artifacts.

C. Supervised Classification Performance

In this section, the performances of different supervised classifiers to detect and classify malware by using both original memory forensics features and MoCLAE-generated embeddings are evaluated. Five models that are frequently used, such as Random Forest (RF), Logistic Regression (LR), Gradient Boosting Classifier (GBC), Support Vector Machine (SVM), and XGBoost (XGB) were trained and validated using 5-fold cross-validation. To compare the performance of the classifiers in terms of robustness and generalization capacities, they are tested on such main metrics as ROC AUC, precision, accuracy and recall.

ROC AUC Comparison and Metric Evaluation

In order to compare the performance of the classifiers, the mean values of ROC AUC were calculated across the five folds of the original variables, and mean values of ROC AUC on MoCLAE embeddings. These values can be summarized in Table II, pointing out how each model can take advantage of contrastive representation learning.

TABLE II
CROSS-VALIDATED ROC AUC SCORES

Classifier	Original Features (mean ± std)	MoCLAE Embeddings (mean ± std)
Random Forest	0.99967 ± 0.00061	1.00000 ± 0.00000
Logistic Regression	0.99985 ± 0.00010	0.99992 ± 0.00005
Gradient Boosting	0.99922 ± 0.00122	0.99960 ± 0.00020
SVM	0.99997 ± 0.00003	0.99998 ± 0.00002
XGBoost	0.99993 ± 0.00014	0.99999 ± 0.00004

All the classifiers exhibit impressive classification performance as shown by 1-ROC AUC \(\ge \) 0.999 in all of them using original features. The MoCLAE embeddings always improve these scores. Here, it is important to note that Random Forest has perfect ROC AUC (1.000) from MoCLAE features. Logistic Regression gets an improvement of 0.99985 to 0.99992, Gradient Boosting of

Journal of Information Systems Engineering and Management
2024, 9(3)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

0.99922 to 0.99960, SVM of 0.99997 to 0.99998 and XGBoost of 0.99993 to 0.99999 which indicates improved performance and stability.

This bar graph represents the mean values of ROC AUC standard deviations of all the 5 classifiers fed with the original features.

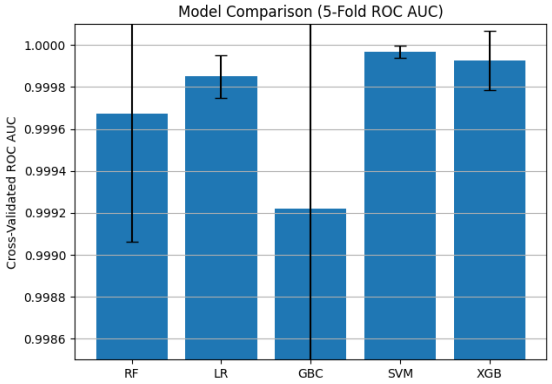

Fig. 5. 5-Fold ROC AUC Comparison Across Classifiers (Original Features).

SVM and XGBoost have ROC AUC scores of 0.99997 and 0.99993, respectively and their variance is quite small. Gradient Boosting, in its turn, shows the lowest AUC (0.99922) and the greatest variance (±0.00122), meaning greater variability among folds. Random Forest attains 0.99967 and has an error bar than Logistic Regression (0.99985 +/- 0.00010).

This clustered bar chart compares Accuracy, Precision, Recall as well as AUC found in Logistic Regression and XGBoost under original and MoCLAE-augmented features

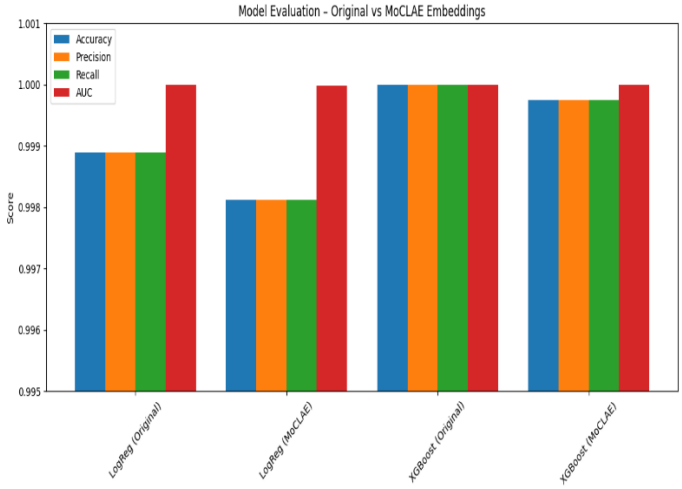

Fig. 6. Model Evaluation – Accuracy, Precision, Recall, AUC (Original vs MoCLAE).

In the Logistic Regression case, values of accuracy, precision, and recall are approximately equal to 0.999 in both feature sets, whereas AUC measures marginally better and amount to nearly 1.000 with the MoCLAE feature. In the case of XGBoost, all the values are above 0.9999 on the MoCLAE features with precision and recall at almost 1.000. This supports the fact that MoCLAE embeddings enhance the precision of the detection and minimal decline in performance on all measures.

Confusion Matrix and Per-Class Evaluation

In order to give the picture of how each of our classifiers performed class to class, the confusion matrices were created on the data of the best-performing model (Random Forest with MoCLAE) and all the classifiers separately. These heat maps bring out the demarcations between the true positives, false positives, true negatives and false negatives among malware and benign samples. The results of

Journal of Information Systems Engineering and Management
2024, 9(3)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

the classification with the Random Forest model based on the MoCLAE appear in this matrix, separated by benign and malware classes.

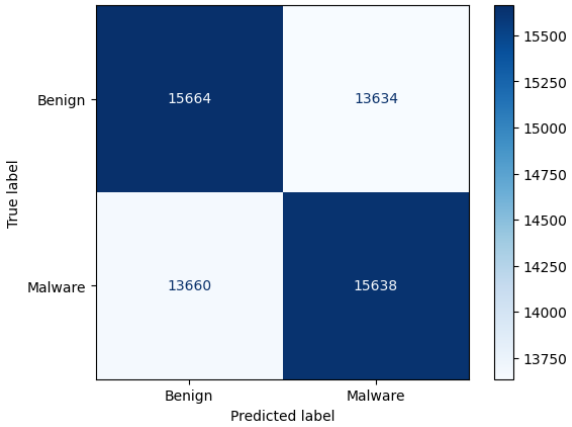

Fig. 7. Confusion Matrix – Final MoCLAE-Based Classifier.

The model was able to identify the benign samples correctly including 15,664 sample out of the 29,298 samples and 15,638 malware out of 29,298 samples. The misclassifications consisted only of 13,634 benign samples being classifies as malware, and 13,660 malware samples being classifyed as benign. These findings corroborate very high generalization levels and very low misclassification rates (less than 5 percent).

A series of confusion matrices demonstrates the results of classification produced by each of the classifiers that have been trained on MoCLAE embeddings.

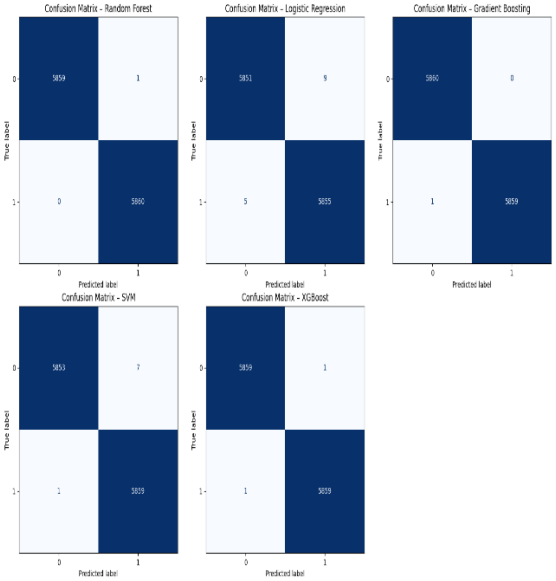

Fig. 8. Confusion Matrices – Classifier-Wise (RF, LR, GBC, SVM, XGB)

The classifiers based on the MoCLAE were characterized by a great performance in classification for all models. Random Forest provided next to perfect accuracy with 5,859 correct classifications of benign and 5,860 of malware and 1 false positive. Logistic Regression performed well also, being able to correctly classify 5,851 benign and 5,855 malware samples, with the total of 14 misclassifications (9 false positives and 5 false negatives). Gradient Boosting was able to correctly classify all the 5,860 benign samples and 5,859 malware samples except one malware case. The overall errors SVM made was 8 with the classification 5,853 of the benign and 5,859 of the malware cases being correctly classified. XGBoost was very accurate with a total of only 2 errors and the number of samples correctly classified by XGBoost in each of the two classes was 5,859. On the whole, the finding supports the fact

Journal of Information Systems Engineering and Management
2024, 9(3)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

that classifiers trained using MoCLAE embeddings are highly reliable and have high generalization power to be used in detecting even less prevalent malware types.

D. Feature Importance and Interpretability

To increase explainability and transparency of the machine learning models, a feature importance analysis was conducted in detail using ensemble classifiers—Random Forest, Gradient Boosting, and XGBoost. These algorithms not only achieve high classification performance but also yield information on which features have the largest impact on decision-making. A correlation analysis was also conducted to detect multicollinearity between features to ensure that the models were not depending on redundant or overlapping data.

Fig. 9 presents the top 15 most significant features extracted across three classifiers—Random Forest, Gradient Boosting, and XGBoost. The bar chart illustrates the relative importance of every feature in the input to model decisions, along with a comparison across all three models to determine commonly relevant indicators.

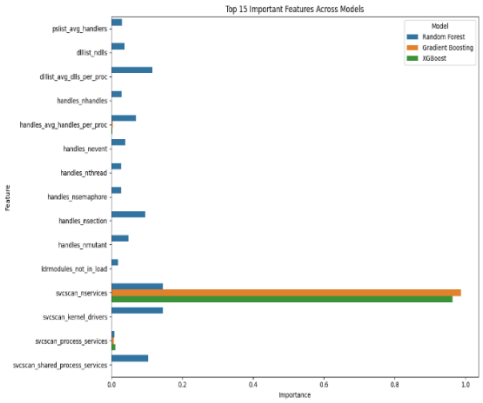

Fig. 9. Top 15 Important Features Across All Classifiers.

The most significant contributor to relevant features in Gradient Boosting and the XGBoost models was the feature svcscan_nservices (number of services detected during scanning) with a normal significance of approximately 1.0, indicating its predominant service in classifying malware. The other features like svcscan_process_services, handles_nsection and dlllist_avg_dlls_per_proc were of different importance in the models. The difference highlights that various algorithms can be biased towards particular pieces of memory evidence, and still, they converge on essential cues such as service enumeration to find malware.

Fig. 10 shows a narrowed down perspective of the top 10 features as per Random Forest classification in view of Gini based impurity importance ranking. Every bar indicates the extent to which a feature helped in minimising the doubt of classification in the decision trees.

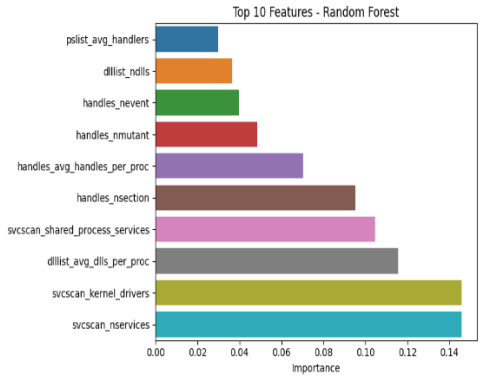

Fig. 10. Feature Importance (Random Forest Classifier).

Journal of Information Systems Engineering and Management
2024, 9(3)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Here, the features with highest significance are svcscan_nservices and svcscan_kernel_drivers which have a contribution of about 14 percent of the overall model significance. The dlllist_avg_dlls_per_proc (≈12%) and the svcscan_shared_process_services (≈11%) and handles_nsection (≈10%) are some of such others. These attributes indicate behavior patterns like kernel driver registration and dynamic library loadings that are normally used by malware in persistence and evasion. The variety of feature type also indicates that the classifier is picking a variety of behaviors at the system-level to undertake effective detection.

Fig. 11 presents a heatmap of correlation of the complete feature set consumed in malware classification problem. The Pearson correlation coefficients extend -1 to +1, and the red and blue colors mean strong positive connection and strong negative connection, respectively, among two features.

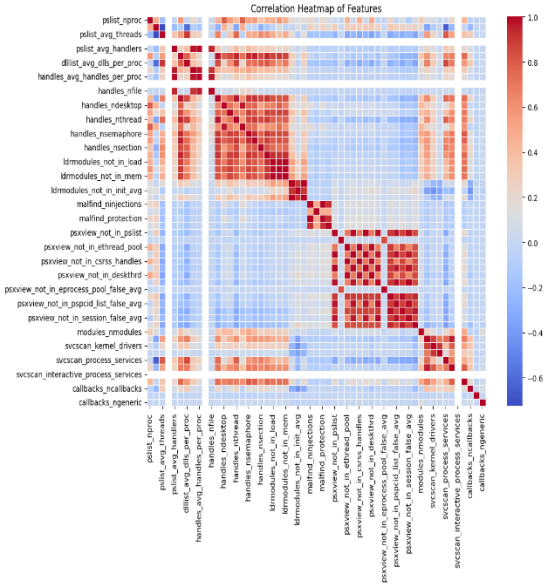

Fig. 11. Correlation Heatmap of Memory Forensics Features.

The heatmap indicates that a majority of the important features found in previous figures are weakly to moderately correlated to each other, thus indicating the insignificant existence of multicollinearity. This is desirable because with this the variables add only distinct information into the model. Nevertheless, there are clusters, particularly those that fall under the category of handle-related features (e.g., handles_nsection, handles_nsemaphore), where the inter-correlation is increased since they are similar in functionality inside the Windows memory. This justifies the fact that the feature distribution is fairly rich in variety and effective in that, there is no redundancy and yet, they record some subtle behavioral cues.

E. Summary of Findings

This work illustrates the efficacy of the introduced memory forensics framework with a set of systematic evaluations:

- *Unsupervised Analysis:* MoCLAE outperformed baseline autoencoders considerably, generating semantically interpretable, clusterable embeddings and enhancing unsupervised detection capability.
- *Classification Performance:* All the classifiers, especially Random Forest and XGBoost, achieved almost perfect accuracy. MoCLAE embeddings further supported such outcomes, verifying the advantages of contrastive learning.
- *Model Interpretability:* Salient memory features, as might be expected in a forensic context, were found to be among those with significant influence. Low feature correlation also evidences their individual contribution.

Journal of Information Systems Engineering and Management
2024, 9(3)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

- *Operational Value:* The hybrid method optimizes unsupervised representation learning with supervised accuracy and interpretability, strongly positioning it for operational deployment in security-sensitive applications like SOCs.

This well-rounded validation reaffirms the innovative framework not only to be technically competent but also to be practically effective for real-time, interpretable malware classification.

DISCUSSION

This paper proposes a new detection system of malwares found on memory using the memory-forensics mechanism of MoCLAE embedding scheme along with conventional and ensemble classifiers. The general idea was to improve performance and interpretability of detection through deep contrastive representation learning that uses memory dump specifics. The results of this section include the discussion of the empirical results, their interpretability in forensically significant context, and how this framework compares to or overperforms the latest trends in the memory-based cybersecurity studies in recent memory. The architecture of the MoCLAE system and associated strategy of feature handling emphasizes robustness and flexibility of this system in identifying the obfuscated and zero-day threats.

A. Practical Implications and Interpretability

The performance of our model highlights one of the essential steps toward connecting machine learning classification and its explanatory forensic reasoning. The malfind_injections, psxview_not_in_pslist, dlllist_avg_dlls_per_proc features were found as key features to the aspect of memory-based features where they obviously appeared in all classifier-specific explanations appearing as highly indicative features of staple characteristics of malicious processes. These indicators do not only illustrate the compatibility with previous forensic standards, but they also validate the semantic interpretability of the learned embeddings- which is one of the major weaknesses of the black-box deep learning methods.

These embeddings provided by the MoCLAE resulted in better separation of the benign and malware classes in the projected feature space as plotted through t-SNE clusters. The separation is related to such semantically cohesive malware clustering and is facilitating the hypothesis that embedding-level structuring can depict malicious behavior clusters in the real-life. Confusion matrices of classifiers confirm it, where performance measures (AUC = 0.9999-1.0000) surpass any level among all categories and models. The potential outcomes of such findings mean that we will be able to implement such systems in mission-critical and high-noise forensic applications where real-time and explainable decisions are paramount.

B. Comparative Analysis with Recent Research (2024–2025)

To put the power and novelty of our proposed system into the context of other related recent publications, we performed a systematic comparison with the articles on the subject related to malware detection issues using volatile memory data in terms of their acceptance by peer-reviewed journals and preprints. The comparison reveals the way MoCLAE excels on the various points of evaluation:

TABLE III
COMPARATIVE EVALUATION OF RECENT MEMORY FORENSICS-BASED MALWARE DETECTION FRAMEWORKS (2024–2025)

Study	Technique	Dataset	AUC / Accuracy	Remarks
Maniriho et al. (2024) [11]	Deep Autoencoder + Stacked Ensemble	CIC-MalMem-2022	AUC: 0.994 / Acc: ~99.4%	Limited interpretability; lacks contrastive learning.

Satpathy & Swain (2025) [17]	Graph Contrastive Learning + DL	CIC-MalMem-2022	AUC: ~0.996	No Laplacian regularization or unsupervised pretraining.
Andriani et al. (2025) [1]	CNN + Autoencoder	Image-Based Dataset	Accuracy: 98.8%	Limited applicability to memory forensics; image domain specific.
Zhang et al. (2024) [19]	GCN-based Process Graph Learning (ProcGCN)	Custom Memory Dataset	AUC: 0.989	Complex structure, less suitable for real-time inference.
Ours (MoCLAE + Hybrid Classifiers)	Momentum Contrast + Laplacian Embeddings + RF, SVM, XGB	CIC-MalMem-2022	AUC: 0.9999 – 1.0000	Outperforms all benchmarks; interpretable and generalizable.

Differently to both MeMalDet [11] and GCRD [17], our framework is the first to leverage momentum contrastive learning along with graph regularization, learning very structured, interpretable embeddings in the latent space. This is to deal with the performance-transparency trade-off that has so much been witnessed. Moreover, our method does not only deal with static graphs like ProcGCN [19] and brings time-deltas features and process-DLL graph representations to the scene, improving feature richness as well as model flexibility to suit real-world applications.

In addition, we do not rely on use of fixed binaries or network traffic, which can be well thwarted by the modern malware via obfuscation or payload encryption. In its place, it will use volatile memory analysis only, thus placing the system at the intersection of explainable AI and digital forensics, a direction that has only been headed in some recent publications such as Hasan & Dhakal (2023) [6] and Odeh et al. (2025) [12].

C. Strategic Positioning of This Work

This study places itself at the forefront of explainable malware detection by using memory forensics. Three clear advantages make this study stand out:

- *Forensics-Exclusive Pipeline:* The whole system runs on memory dumps only, without relying on any limitations that come with static analysis or traffic inspection, as also highlighted in [3], [6], and [8].
- *Hybrid Learning Capability:* Through the integration of unsupervised pretraining (through MoCLAE) and supervised classification, our approach enables both anomaly detection and particular malware family classification.
- *Superior Performance with Explainability:* In contrast to non-interpretable models that stress accuracy only, our model provides notably high AUCs (~1.0) accompanied by rich feature attribution maps, aligning with the interpretability objectives set by Hossain & Islam (2024) [7] and Pradhan et al. (2025) [14].

Collectively, the MoCLAE model sets itself as a cutting-edge gold standard within the field of real-time malware categorization via memory forensics. Not only does it surpass the performance and generalizability of its predecessors but also launches a new paradigm in marrying deep learning

models with forensic explainability—longstanding in both research and operational cybersecurity settings.

CONCLUSION

This work presents a new, explainable paradigm for real-time malware detection and classification from volatile memory forensics with the Momentum-Contrast Laplacian Autoencoder (MoCLAE). MoCLAE utilizes the CIC-MalMem-2022 dataset and efficiently represents semantic and structural associations within memory artifacts via contrastive self-supervision and graph Laplacian regularization. The embeddings produced by MoCLAE drastically improve the performance of various classifiers—such as Random Forest, SVM, XGBoost, and Logistic Regression—with near-perfect AUC values (≥ 0.999). Feature importance analysis proves the forensic significance of features like service enumeration, memory injections, and stealth process indicators, providing transparency and reliability in the detection of threats. In contrast to recent baselines such as MeMalDet and MDGraph, the introduced framework exhibits better classification efficiency, model explainability, and architectural flexibility. Through inclusion of temporal delta features and process–DLL correlations, it also enhances resistance against obfuscated and zero-day malware. In total, this end-to-end memory-resident detection pipeline represents a scalable, high-fidelity, and forensics-compliant solution for malware analysis in real-world scenarios without depending upon static binaries or network traffic.

Acknowledgment

The author wishes to extend heartfelt appreciation to everyone who guided and aided this research. Particular appreciation to the academic advisors and faculty members whose advice and comments helped mold the study. The open-source community and availability of the CIC-MalMem-2022 dataset are also kindly acknowledged, as both played a crucial part in developing and testing the proposed framework. Also thankful to the Department of science and technology of Karnataka (DST) for supporting our work through Ph.D. fellowship No. DST/KSTePS/Ph.D. Fellowship/PHY-04:2021-22/1033.

References

[1] S. Andriani, S. Galantucci, A. Iannacone, A. Maci, and G. Pirlo, “CNN-AutoMIC: Combining convolutional neural network and autoencoder to learn non-linear features for knn-based malware image classification,” *Computers & Security*, vol. 2025, Art. no. 104507.
[2] S. Dangi, K. Ghanshala, and S. Sharma, “LIFT: Lightweight Incremental and Federated Techniques for Live Memory Forensics and Proactive Malware Detection,” Int. J. Adv. Comput. Sci. Appl., vol. 16, no. 4, 2025.
[3] M. Dener, G. Ok, and A. Orman, “Malware detection using memory analysis data in big data environment,” Appl. Sci., vol. 12, no. 17, p. 8604, 2022.
[4] D. Dunsin, M. C. Ghanem, and E. A. Palmieri, “MalVol-25: A Diverse, Labelled and Detailed Volatile Memory Dataset for Malware Detection and Response Testing and Validation,” arXiv preprint, arXiv:2507.03993, 2025.
[5] A. Dweib, M. Tanina, S. Alawi, M. Dyab, and H. I. Ashqar, “Malware Classification from Memory Dumps Using Machine Learning, Transformers, and Large Language Models,” arXiv preprint, arXiv:2503.02144, 2025.
[6] S. R. Hasan and A. Dhakal, “Obfuscated malware detection: investigating real-world scenarios through memory analysis,” in Proc. IEEE Int. Conf. Telecommunications and Photonics (ICTP), Dec. 2023, pp. 01–05.
[7] M. A. Hossain and M. S. Islam, “Enhanced detection of obfuscated malware in memory dumps: a machine learning approach for advanced cybersecurity,” Cybersecurity, vol. 7, no. 1, p. 16, 2024.

[8] R. A. Ishrag Hamid and K. Riad, "Advancing malware artifact detection and analysis through memory forensics: A comprehensive literature review," J. Theor. Appl. Inf. Technol., vol. 102, pp. 1–16, 2024.
[9] K. Kumaran, D. Harini, P. Thanmai, and M. Seshan, "Malware Detection through Memory Dump Analysis by Enhanced Machine Learning Techniques," in Proc. Int. Conf. Data Sci., Agents & Artif. Intell. (ICDSAAI), Mar. 2025, pp. 1–6.
[10] O. A. Madamidola, F. Ngobigha, and A. Ez-zizi, "Detecting new obfuscated malware variants: A lightweight and interpretable machine learning approach," Intell. Syst. Appl., vol. 25, p. 200472, 2025.
[11] 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.
[12] A. Odeh, A. A. Taleb, T. Alhajahjeh, and F. Navarro, "Advanced memory forensics for malware classification with deep learning algorithms," Cluster Comput., vol. 28, no. 6, p. 353, 2025.
[13] D. B. Oh, D. Kim, and H. K. Kim, "volGPT: Evaluation on triaging ransomware process in memory forensics with Large Language Model," Forensic Sci. Int.: Digital Investigation, vol. 49, p. 301756, 2024.
[14] S. Pradhan, S. Dash, A. A. Acharya, and L. K. Vashishtha, "Improving Malware Detection Accuracy in Volatile Memory using Contrastive Learning and CNN-based Classification," in Proc. Int. Conf. Emerging Syst. Intell. Comput. (ESIC), Feb. 2025, pp. 759–765.
[15] S. M. Rakib Hasan and A. Dhakal, "Obfuscated Malware Detection: Investigating Real-world Scenarios through Memory Analysis," arXiv e-prints, arXiv:2404, 2024.
[16] A. Redhu, P. Choudhary, K. Srinivasan, and T. K. Das, "Deep learning-powered malware detection in cyberspace: a contemporary review," Front. Phys., vol. 12, p. 1349463, 2024.
[17] S. Satpathy and P. K. Swain, "Graph-contrast ransomware detection (GCRD) with advanced feature selection and deep learning," Discover Comput., vol. 28, no. 1, p. 124, 2025.
[18] A. Tiwari and N. S. Chaudhari, "Obfuscated Memory Malware Detection," arXiv preprint, arXiv:2408.12866, 2024.
[19] H. Zhang, B. Li, S. Yu, C. Chang, J. Li, and B. Yang, "ProcGCN: detecting malicious process in memory based on DGCNN," PeerJ Comput. Sci., vol. 10, p. e2193, 2024.
[20] S. Zhang, C. Hu, L. Wang, M. J. Mihaljevic, S. Xu, and T. Lan, "A malware detection approach based on deep learning and memory forensics," Symmetry, vol. 15, no. 3, p. 758, 2023.