EvoNorm-GAN for Adaptive and Interpretable Detection of Ransomware in Windows PE Files

G Badrinath[1], Dr. Arpita Gupta[2]
Research Scholar, Department of Computer Science and Engineering,
Koneru Lakshmaiah Education Foundation, Hyderabad-500075, Telangana, India[1]
Associate Professor, Department of Computer Science and Engineering,
Koneru Lakshmaiah Education Foundation, Hyderabad-500075, Telangana, India[2]

***Abstract*—Ransomware remains a key cybersecurity issue because of its growing amount of obfuscation, polymorphism, and constantly changing patterns of attack that repeatedly circumvent conventional defenses. Traditional systems and standard deep learning may fail, lowering accuracy and increasing false positives. To address these shortcomings, the proposed work proposes EvoNorm-GAN, a dynamic adversarial-based detection architecture that will incorporate Feature-Wise Dynamic Normalization (FDN) and Generative Adversarial Network to analyze ransomware in Windows Portable Executable (PE) files in a very flexible manner. Generator creates ransomware variants; discriminator classifies files using Wasserstein loss. EvoNorm-GAN is a TensorFlow application, using the Keras back-end, and tested on a large-scale Windows PE File Analysis Dataset of 62, 200 samples, with 31, 100 benign and 31, 100 malicious examples. The experimental findings indicate that EvoNorm-GAN has the state-of-the-art results of 98.2 % accuracy, 98.4 % precision, 98.1 % recall, 97.4 % F1-score, and 0.99 AUC, which are about 1 to 3 percent higher than the traditional CNN, RNN, and ensemble-based models. To enhance transparency and trust, SHAP-based explainable AI is integrated into EvoNorm-GAN, highlighting key PE file features such as Section Entropy and SizeOfCode that drive classification decisions. By combining adaptive learning, adversarial sample generation, and analyst-friendly interpretability into a unified framework, EvoNorm-GAN delivers an efficient, robust, and transparent ransomware detection system. Its scalable and resilient design makes it well-suited for real-world deployment in endpoint protection and cybersecurity environments, providing reliable detection of evolving ransomware threats.**

***Keywords*—Ransomware detection; EvoNorm-GAN; feature-wise dynamic normalization; portable executable files; adversarial learning; Explainable AI**

I. Introduction

Ransomware has rapidly established itself as one of the most common and harmful categories of cybercrime that targets people, companies, and critical infrastructure all over the world. Attacking data by means of data encryption and ransom money, most commonly, in the form of cryptocurrency, attackers cost organizations a lot of money and interruptions in business operations in the most sensitive domains such as healthcare, finance, government, and education [1], [2]. The most recent developments, including ransomware-as-a-service (RaaS), have lowered the barrier to entry since less skilled attackers can run a sophisticated campaign using malware through rented packages [3]. Moreover, the modern ransomware groups also employ the so-called double extortion, that is, to extort their victims by encrypting their data beforehand and threatening to publish confidential information in case of non-payment, which increases the inefficiency of the old system of recovery based on backups [4]. Advanced stealth methods such as fileless execution, polymorphic code, and encrypted command-and-control communications also complicate the task of defense by circumventing conventional security products [5], [6]. Conventional detection mechanisms have been the mainstay of cybersecurity defenses over decades but are no longer being effective against these new attacks. Additional signature-based tools rely on the identification of any known sequence of bytes or behavioral patterns that are indicative of a known ransomware [7], [8]. Such systems will fail to detect new or obfuscated ransomware samples despite their usefulness in detecting variants previously indexed. Any little modifications to the payload, through encryption or code obfuscation render the current signatures useless [9], [10].

Lag time between emergence of a new variant and signature updates leaves systems vulnerable to major risks, taking organizations off guard during the most perilous phases of outbreaks [11]. In feature-based methods of detection, attempts to create resilience include analyzing suspicious features, i.e. out of place API calls, out of place file encryption activity, or out of place system resource usage [12]. Such methods, though, are dependent on domain knowledge and data quality since they need feature engineering that is expert-driven. Attackers start countering these methods by slowing down execution, emulating harmless applications, or using zero-day exploits to sneak past the detection [12]. As a result, signature-based as well as feature-based approaches have very high false positives and false negatives, making them less effective for real-time detection.

To overcome these shortfalls, researchers have looked toward machine learning (ML) and deep learning (DL) methods, which are able to learn complex feature representations directly from raw data. Recent research illustrates that DL models such as convolutional neural networks (CNNs), recurrent neural networks (RNNs), and Transformers are able to pick up on subtle patterns of ransomware behavior without extensive human feature crafting [13], [14]. These models are best at identifying subtle anomalies such as unusual system calls, memory access patterns, or initial user activity deviations. Notably, DL frameworks are strong against zero-day ransomware through the ability to generalize from static rules

[15]. Nevertheless, even solutions based on DL have their weaknesses: their main reliance is on the use of the static normalization techniques such as Batch Normalization, which in turn is based on the assumption of the data distribution constancy. But in cybersecurity where the ransomware families continue to evolve, this is not true, and the performance on the new threats becomes impaired and the generalization is weak. Generative Adversarial Networks (GANs) are a recent newcomer in this respect with new opportunities to both perform anomaly detection and data augmentation. GANs consist of a discriminator and a generator, where the latter is required to produce fake samples, and the former is expected to learn the distinction between fake and real data [16]. Adversarial training has been used in ransomware detection with discriminators becoming highly effective at picking out malicious features, with the generators helping in alleviating dataset imbalance by creating artificial examples of ransomware. Such a two-in-one advantage makes the model more resilient and able to generalize more [17]. Nonetheless, there is one key limitation namely that vanilla GAN training is grounded on normalization techniques relevant to stationary distributions and therefore not stable in realistic ransomware situation where feature distributions vary over time. This evolving nature of ransomware reinforces the need of models to study on the basis of adjustment to evolving distributions throughout training. Compared to CNN, RNN, and Transformer-based models, which have good feature extraction features, and GANs, which have adversarial features, none can solve the non-stationarity problem successfully. Without this type of adaptability, detection models will be less precise and provide false alarms with ransomware constantly evolving. This research gap offers the foundation on which more complex frameworks that integrate adversarial learning and dynamic normalization can be built to offer stability and flexibility.

It is against this backdrop that the proposed EvoNorm-GAN comes up with a new solution to ransomware detection in Windows Portable Executable (PE) files. EvoNorm-GAN (using feature-wise dynamic normalization as part of the GAN architecture) can therefore modify feature statistics in each mini-batch, and hence adopt changing data distributions during training. The new mechanism counters the deficiencies of the old methods of normalization, ensuring the stability on the new types of ransomware variants detection. With adversarial training, EvoNorm-GAN employs Wasserstein loss and gradient clipping to guarantee stable training as well as improved ability to generalize to new threats. When put in a wider context of research on ransomware identification, EvoNorm-GAN is a well-equipped and versatile solution to fill the gap of the long-standing deficiencies of the existing methods and respond to the demands of the modern world of cybersecurity.

A. Research Motivation

Ransomware has become an extremely dynamic threat, taking advantage of obfuscation, polymorphism, encryption, and delayed execution to avoid the old methods of detection, such as signature-based and static feature analysis. The existing systems have difficulties with detecting the zero-day attacks and adapting malware, which is why the development of smart and resistant detection systems is urgently needed. This research seeks to devise an active and responsive framework that can be effective in identifying ransomware in windows PE files and at the same time make them interpretable. The ability to incorporate Feature-Wise Dynamic Normalization (FDN) into a GAN framework, combined with the implementation of explainable AI using SHAP and present a transparent and practical approach to cybersecurity challenges in the real world, makes the proposed methodology an effective solution to this type of problem.

B. Research Significance

By identifying the weaknesses of the existing traditional ransomware detection tools that find it difficult to match the fast development of malware, the study discusses the most pressing issues of cybersecurity. It presents a dynamic, adaptive, and interpretable EvoNorm-GAN model that can cope with the changes in data distributions and indicates unseen ransomware variations by using Feature-Wise Dynamic Normalization (FDN). Adversarial training, which is under the guidance of Wasserstein loss and gradient penalty, increases the model stability and resistance against new threats. The explainability offered by SHAP-based explainable AI offers feature-level interpretability to bridge the black-box model decisions and the human understanding gap. The resulting architecture provides ransomware detection that is transparent, correct, and scalable to better secure organizational and critical infrastructure systems.

C. Key Contributions

- Introduces EvoNorm-GAN, combining GAN-based adversarial learning with Feature-Wise Dynamic Normalization to detect ransomware in Windows PE files.
- Enables the model to adapt to evolving ransomware patterns and non-stationary data distributions, improving generalization to unseen threats.
- Incorporates SHAP-based interpretability, highlighting critical PE file features influencing classification and enhancing transparency for cybersecurity analysts.
- Provides a scalable and robust framework suitable for real-world deployment in endpoint protection systems, SIEM platforms, and dynamic malware detection environments.

The structure of the remaining sections of this study is as follows: Section II provides a summary of prior research on ransomware detection techniques, emphasizing advancements in machine learning and deep learning methods. Section III discusses the challenges and limitations of existing approaches in addressing the dynamic and evolving nature of ransomware attacks. Section IV details the proposed EvoNorm-GAN framework, including its methodology, model architecture, data preparation process, and evaluation metrics. Section V presents the experimental results, showcasing the framework's performance in detecting ransomware, along with an in-depth analysis of the findings. Finally, Section VI concludes the study by summarizing the key outcomes and proposing directions for future research and applications in cybersecurity.

II. Related Works

Singh et al. [18] proposed RANSOMNET+, a hybrid deep neural model consisting of convolutional neural networks (CNNs) for local feature extraction and Transformers for learning hierarchical representations. Tested on cloud-encrypted data, the model attained 96.1% testing accuracy with precision, recall, and F1-scores higher than 96%. The research also included interpretability analysis to enhance transparency. Despite its robust performance, though, RANSOMNET+ did not test cross-dataset adaptability or real-time deployment efficiency, meaning there are questions about whether it would be robust to novel ransomware variants.

Alqahtani [19] improved the traditional mutual information feature selection technique by introducing eMIFS, which utilizes a normalized hyperbolic tangent (tanh) function as a criterion for assessing redundancy between features. Utilizing TF-IDF representations of text-based features, the technique enhanced discrimination among relevant features, allowing for earlier identification of ransomware. eMIFS enhanced selection accuracy over conventional techniques but wasn't experimentally tested over varied datasets and didn't consider dynamic or changing malware distributions, limiting real-world applicability.

Lee et al. [20] addressed entropy-based detection system weaknesses, presenting the potential for attackers to tamper with file entropy using encoding techniques like Base64 masking. In response to this, the authors introduced Format-Preserving Encryption (FPE) methods, including Radix Conversion, which realized 96% neutralization accuracy in detecting entropy tampering in ransomware. While powerful in attacker-focused environments, the method offered restricted opportunity for proactive defense methodologies and failed to incorporate adversarial resilience to varying data trends.

Molina et al. [21] analyzed evasive ransomware behavior, or "paranoia" API calls, which evaluate the execution environment prior to attacking. Their method utilized natural language processing (NLP) techniques, including out-of-vocabulary (OoW) embedding, to identify API requests that correlate with evasive behavior. Based on over 3,000 malware samples, they trained machine learning and deep learning classifiers with high classification accuracy. However, their dependence on pre-attack patterns limited use against ransomware variants that tamper with or hide initial execution cues, reducing its detection capability.

Gazzan et al. [22] introduced UA-DES, an uncertainty-aware dynamic epoch selection deep learning-based strategy, to dynamically optimize deep belief networks (DBNs) for ransomware detection. UA-DES utilized Bayesian methods, dropout, and interactive learning to minimize overfitting and enhance robustness. The framework showed encouraging adaptability and performance gains but was limited to DBNs and did not specifically address the challenges of evolving, non-stationary ransomware distributions.

Previous works show significant improvement in ransomware detection by hybrid deep learning, feature extraction, entropy manipulation analysis, behavioral modeling, and uncertainty calibration. These, however, all have one thing in common: they require relatively stationary data distributions. None of them properly handle the non-stationary and dynamic nature of ransomware that destroys generalization in adversarial learning scenarios. To close this gap, the suggested EvoNorm-GAN presents feature-wise dynamic normalization in a GAN structure to facilitate real-time adaptation to changing feature distributions. With adversarial training coupled with Wasserstein loss and gradient clipping, EvoNorm-GAN balances stability and flexibility to provide a robust approach to ransomware detection in complex and constantly changing settings. The comparative analysis of ransomware detection related works is shown in Table I.

TABLE I. Comparative Analysis of Ransomware Detection Related Works

Author	Focus Area	Methods	Dataset	Key Results	Limitations
Singh et al. [18]	Deep learning for cloud-encrypted ransomware	CNN + Transformer (RANSOMNET+)	Cloud-encrypted ransomware dataset	96.1% accuracy, F1-score 96%	Lacks cross-dataset validation; not real-time adaptable
Alqahtani [19]	Feature selection & redundancy reduction	eMIFS (tanh-based redundancy evaluation, TF-IDF)	Text-based ransomware features	Improved feature relevance & early detection	No real-world or multi-dataset validation; static approach
Lee et al. [20]	Entropy-based detection evasion	Format-Preserving Encryption (Radix Conversion)	Files with entropy masking	96% accuracy in neutralizing entropy attacks	Attacker-centric; lacks proactive defense integration
Molina et al. [21]	Behavioral evasive detection	NLP (OoW embeddings for API call classification)	3,000 malware samples	Effective classification of evasive behavior	Limited to pre-attack indicators; narrow scope
Gazzan et al. [22]	Uncertainty-aware DL strategies	UA-DES (Bayesian + DBN optimization)	DBN ransomware benchmarks	Robustness improvements, reduced overfitting	Restricted to DBNs; does not address evolving threats

III. Problem Statement

Ransomware has remained a major cybersecurity menace that has been adopting damaging strategies of obfuscation, polymorphism, and encryption, as well as delayed execution, which overcome traditional signature-based and static feature detectors [18]. The current literature has considered deep learning models, feature selection, entropy-based analysis, behavioral modeling, and uncertainty-aware strategies, but they tend not to be adaptable to novel ransomware types, generalize across datasets, and offer minimal interpretability to make decisions [22]. This is why gaps cause high rates of false

positive, lack of warning of zero-day threats, and lack of trust in automated systems, limiting the effectiveness of cybersecurity defenses. The proposed framework comprising EvoNorm-GAN, with Feature-Wise Dynamic Normalization, adversarial training, and SHAP-based explainable AI overcomes these shortcomings to provide a high-quality, adaptive, and explainable ransomware detector to support growing threats.

IV. Proposed EvoNorm-GAN for Adaptive Ransomware Detection

This study presents a comprehensive framework for detecting ransomware in Windows PE files using EvoNorm-GAN. It is a combination of structured data preprocessing, normalization of features, adversarial training, and explainable AI to guarantee correct, adaptive, and interpretable classification. The features of the raw PE files are cleaned and normalized and their contents are fed into the generator-Discriminator network which is trained with Wasserstein loss and gradient penalty to increase stability and robustness. SHAP is also included to offer an interpretability feature on a feature level, so that an analyst can gain insight into model decisions. The overall step-by-step workflow of the proposed methodology is shown in Fig. 1.

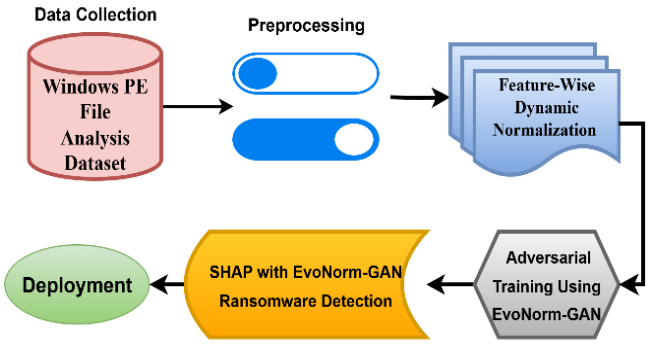

Fig. 1. Workflow of the proposed model.

A. Data Collection

The "Windows PE File Analysis Dataset" [23] is a big dataset of more than 62,000 samples of benign and malicious Windows executable and DLL files, for malware analysis and detection research. Each dataset sample is described by static attributes inferred from the Portable Executable (PE) file format, for example, PE header properties and structural elements. These attributes provide critical information regarding file attributes that distinguish legitimate programs from malware, e.g., ransomware. Data is stored in a CSV file and includes samples labeled with known malware hashes from repositories like VirusShare, and it is ideal for training and testing machine learning and deep learning models for static malware detection.

B. Data Pre-Processing

1) Missing value imputation: Missing values are replaced to maintain data consistency. Continuous attributes (e.g., file size, entropy) are imputed with mean, while categorical attributes (e.g., section names, header flags) are imputed with the mode.

2) Duplicate removal: Repeated records, including identical hash samples, are removed to prevent data bias.

3) Label encoding: Convert categorical labels to binary format for classification. It is expressed in Eq. (1).

$$y = \begin{cases} 0, & benign \\ 1, & ransware \end{cases} \quad (1)$$

4) Min-max feature scaling: Continuous features are scaled to [0,1] to prevent features with large values from dominating the learning process. It is expressed in Eq. (2).

$$x' = \frac{x - x_{min}}{x_{max} - x_{min}} \quad (2)$$

where, x_{min} and x_{max} are the minimum and maximum values of the feature.

C. Feature-Wise Dynamic Normalization (FDN) for GAN Stability

Feature-Wise Dynamic Normalization (FDN) has been presented as a solution for addressing distribution shifts of mini-batches in the training stage in GANs when input data have dynamically shifting patterns in the data such as dynamic patterns found in real cybersecurity threats such as ransomware. Traditional normalization techniques, such as Batch Normalization, rely on static statistics (mean and standard deviation) computed over a data set or mini-batch and can be problematic when the data distribution is diverse, especially when the data set is non-stationary (e.g., data that evolves over time due to new behavior, e.g., ransomware that continues to evolve).

FDN avoids this by calculating the mean $\mu_{dynamic}$ and standard deviation σ_{batch} for dynamically at each epoch from the current mini-batch and feature space. In other words, rather than using global or static statistics, normalization is calculated locally and brought up to date as training proceeds. This allows the model to adjust to new, unseen patterns of behavior, providing improved generalization for continuously changing data [24].

For every training mini-batch, the statistics used for normalization are recalculated dynamically. In particular, the mean μ_{batch} and standard σ_{batch} are calculated for each feature in the mini-batch. Each feature value x is normalized by subtracting the respective batch mean and dividing by the batch standard deviation, as given in Eq. (3).

$$x' = \frac{x - \mu_{batch}}{\sigma_{batch+\epsilon}} \quad (3)$$

where, x' represents the normalized feature, μ_{batch} and σ_{batch} are the computed batch mean and batch standard deviation, respectively, and ϵ is a small constant added to prevent numerical instability.

By refreshing these figures at every epoch, FDN makes sure that the model keeps on responding to shifting data distributions, which is especially beneficial for use in the case of adversarial training where the discriminator and generator are usually presented with new unexpected behaviors (e.g.,novel ransomware attacks). Fig. 2 shows the training flowchart of EvoNorm-GAN illustrates the mini-batch feature normalisation, adversarial training between the generator and discriminator, optimising Wasserstein loss, gradient clipping and iterative

adaptation to changing distributions of ransomware to ensure robust detection.

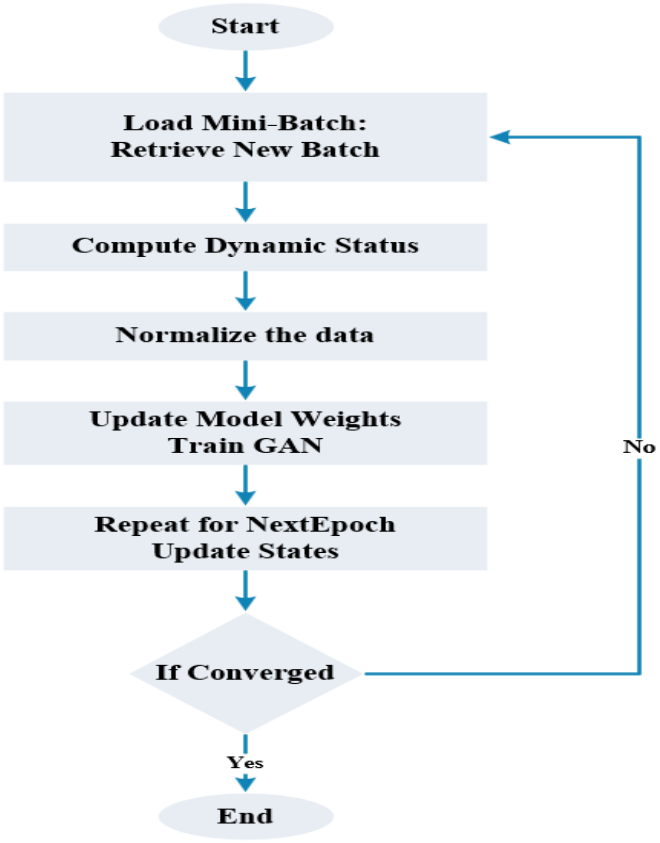

Fig. 2. FDN-GAN with feature-wise dynamic normalization for stable adversarial learning.

D. Adversarial Training Using EvoNorm-GAN

EvoNorm-GAN consists of the adversarial training phase, during which two rival networks the generator Gand the discriminator D undergo a dynamic training process to maximize the ransomware detecting potential of the model. The generator gets a latent noise input $z \sim P_z$ and is trained to generate synthetic Portable Executable (PE) feature representations, which resemble the real ransomware samples and benign samples, whereas the discriminator is trained at differentiating between the actual samples $x \sim P_{data}$ and the synthetic ones $G(z)$. This interaction creates a min max optimization procedure, which allows the discriminator to learn fine grained decision boundaries, which can learn the behavioral signatures of ransomware in the high dimensional feature space. The training goal is based on the Wasserstein Generative Adversarial Network (WGAN) construction that enhances the stability of convergence by means of the Earth-Mover (EM) distance over the Jensen-Shannon divergence. The loss of the discriminator is defined as Eq. (4):

$$L_D = \mathbb{E}_{x\sim P_{data}}[D(x)] - \mathbb{E}_{z\sim P_z}[D(G(z))] \quad (4)$$

and the generator's objective is to minimize the discriminator's ability to distinguish between real and generated samples in Eq. (5):

$$L_G = -\mathbb{E}_{z\sim P_z}[D(G(z))] \quad (5)$$

These equations make sure that the generator is incrementally strengthened to produce realistic-like ransomware, and the discriminator is made stronger to recognize the slight differences. The Wasserstein metric offers more informative gradients which are smoother and thus avoiding the vanishing gradient issue experienced with traditional GANs. In order to achieve the 1-Lipschitz continuity property to construct the Wasserstein distance, a gradient penalty term is added as a regularization method:

$$L_{GP} = \lambda\square\mathbb{E}_{\hat{x}\sim P_{\hat{x}}}[(\| \nabla_{\hat{x}} D(\hat{x}) \|_2 - 1)^2] \quad (6)$$

Eq. (6), $\hat{x}$ is an interpolated sample between real and generated data, and λ is the penalty coefficient that controls the trade-off between stability and learning capacity. This penalty does not only stop the discriminator overfitting, but also provides training consistency in cases where the discriminator might overfit when using complex variations of features that fall in the samples of polymorphic ransomware. The discriminator is normally updated several times before a single generator update during the training cycle to ensure a balance between the two networks. The optimization process is done with Adam optimizer and learning rates $lr_D = 4 \times 10^{-4}$ and $lr_G = 2 \times 10^{-4}$, and momentum parameters $\beta_1 = 0.5$ and $\beta_2 = 0.9$. The iterative updates are used to refine the two models until the generated feature distributions are similar to the real PE dataset. As the training moves forward the Wasserstein distance between real and generated data decreases indicating that the generator has learnt realistic malware behavior patterns; the discriminator has become discriminative.

Gradient penalties, gradual adversarial updates, and Wasserstein optimization guarantees the smooth convergence, helps to reduce mode collapse, as well as, improves the generalization to unseen variants of ransomware. Through repeated training on discriminator with adversarially generated samples, the EvoNorm-GAN architecture can be made more resistant to new obfuscation and packing methods that normally harm traditional deep learning-based detectors. As a result, the trained adversarial model attains a strong balance between sensitivity and specificity, which guarantees strong ransomware detection under changing threat circumstances.

E. SHAP with EvoNorm-GAN Ransomware Detection

After the EvoNorm-GAN discriminator predicts whether a PE file is benign or ransomware, SHAP values are computed to measure the contribution of each feature toward the model's output. Formally, the SHAP value for a feature $i$is defined as in Eq. (7).

$$\phi_i = \sum_{S\subseteq F\setminus\{i\}} \frac{|S|!\square(|F|-|S|-1)!}{|F|!}[f(S\cup\{i\}) - f(S)] \quad (7)$$

Here, $F$is the set of all features, $S$is a subset of features excluding i, and $f(S)$represents the model output when only the features in $S$are present. The term $f(S\cup\{i\}) - f(S)$measures the marginal contribution of feature $i$to the prediction across all possible subsets of features, ensuring a fair attribution of importance. In your experiment, once the discriminator output is obtained after the preprocessing and feature extraction, the

discriminator output is ranked with SHAP to determine the attributes that are the most significant in a PE file, as far as the ransomware classification is concerned. Such features as section entropy, PE header values, and import table anomalies are given higher SHAP values when they have a strong influence on the model to predict ransomware. The sum of SHAP values over the data set gives an idea of the attribute importance on a global scale, the ones that are more important in predicting ransomware. On a file-by-file basis, local explanations can be used to visualize why a particular PE file is judged as malicious or benign, which can help cybersecurity analysts to perceive how the model rationale works and develop a specific mitigation strategy. The integration of SHAP with EvoNorm-GAN is more interpretable, encourages confidence in automated detection, and offers practical information about the features of ransomware.

F. Integration Framework for EvoNorm-GAN

The integration structure of the EvoNorm-GAN makes all the steps of detecting ransomware one unified pipeline to combine the data preprocessing, normalization, adversarial training, classification, and interpretability. The raw Windows PE files in the dataset are first ingested and data is preprocessed to guarantee quality of data including any missing values, duplicate data, corrupted files and the encoding of categorical labels. Consecutive characteristics like file size, header values, and entropy are normalized with Min–Max normalization, whereas training phase uses the Feature-Wise Dynamic Normalization (FDN) to keep the models stable in the non-stationary distributions. These normalized feature sets are then inputted into the EvoNorm-GAN adversarial model, in which the generator generates ransomware-like patters and the discriminator is trained to become a file classifier, i.e. whether it is benign or malware. This application of Wasserstein loss optimization together with gradient clipping training increases the stability of training in GANs and alleviates many of the pitfalls associated with this model, such as mode collapse and unstable convergence. After being trained, the discriminator provides binary classifications of new PE files. In order to guarantee interpretability, SHAP is included, which offers insights at feature-level information about the prediction process by measuring the contribution of each PE attribute to the classification. The overall architecture of the ransomware detection system is a strong, scalable, and explainable AI system preprocessed, dynamically normalized, trained with adversarial learning, and explained. The integrated architecture can be deployed in real-world environments of cybersecurity, including endpoint security systems or SIEM platforms, with adaptive defense against the continually changing ransomware and allowing analysts to be aware of, have trust in, and take action on the decisions of the model. The overall framework is illustrated in Fig. 3.

The EvoNorm-GAN framework proposes a number of novel contributions to ransomware detection. It dynamically adjusts to changing ransomware patterns and non-stationary PE file distributions, by applying Feature-Wise Dynamic Normalization in a GAN architecture, which is better than traditional normalization techniques. The Wasserstein loss and gradient penalty adversarial training can achieve constant convergence and robust feature learning, whereas the interpretability of SHAP can offer a clearer and more feature-level explanation to cybersecurity analysts. Such an adaptive normalization and adversarial learning, with explainable AI is a distinctive, scalable, and robust methodology to identify known and unknown variants of ransomware.

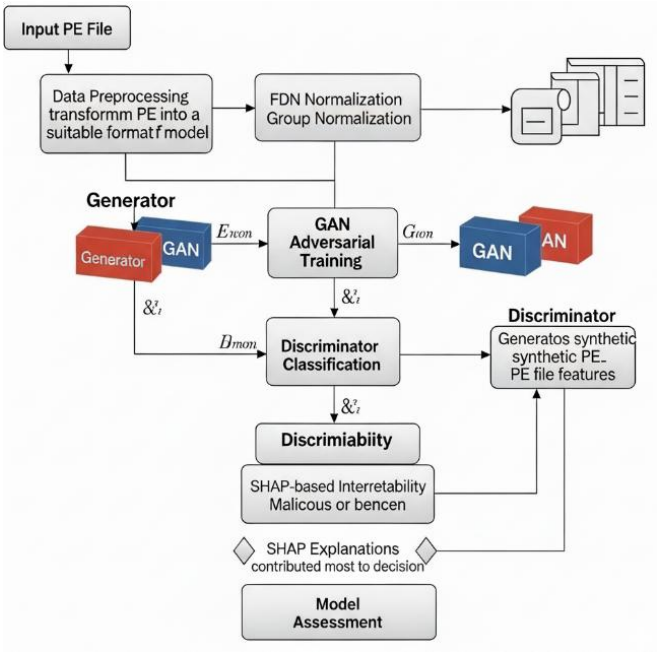

Fig. 3. Working process of EvoNorm-GAN with XAI framework.

Algorithm 1. EvoNorm-GAN Ransomware Detection

```
Input: Windows PE dataset X with labels Y
Output: Binary classification of PE files (benign = 0, ransomware = 1)
        and SHAP interpretability values φ
Load dataset X and Y
Preprocess dataset:
  For each sample x in X:
    For each feature f in x:
      If f is missing:
        If f is continuous:
          Fill f with mean or median of the feature
        Else if f is categorical:
          Fill f with mode of the feature
      End If
    End For
    If sample x has critical unrecoverable fields:
      Remove x from dataset
    End If
  End For
  Remove duplicate samples
  Encode labels: benign = 0, ransomware = 1
Normalize continuous features:
  For each feature f in X:
    x' = (f - min(f)) / (max(f) - min(f))
  End For
Initialize EvoNorm-GAN:
```

```
Initialize Generator G and Discriminator D
Set learning rates and optimizer parameters
Train EvoNorm-GAN:
  For each epoch:
    For each mini-batch B in X:
      Generate synthetic samples G(z) using random noise z
      Compute D(x) for real samples x in B
      Compute D(G(z)) for generated samples
      Compute discriminator loss L_D = mean(D(x)) - mean(D(G(z)))
      Compute generator loss L_G = -mean(D(G(z)))
      Compute gradient penalty L_GP = λ * mean((norm(gradient(D(interpolated_samples))) - 1)^2)
      Update discriminator: D = D - lr_D * gradient(L_D + L_GP)
      If batch_step % n_critic == 0:
        Update generator: G = G - lr_G * gradient(L_G)
      End If
    End For
  End For
Classify new PE samples:
  For each sample x_new:
    y_pred = D(x_new)
    If y_pred >= 0.5:
      Label = 1 // ransomware
    Else:
      Label = 0  / benign
    End If
  End For
Explain predictions using SHAP:
  For each sample x_new:
    For each feature f in x_new:
      Compute φ_f = SHAP_value(f, x_new, D)
    End For
  End For
Output predicted labels for all PE files
Output SHAP values φ for feature interpretability
```

The EvoNorm-GAN algorithm preprocesses the features of Windows PE files and normalizes them, followed by the training of a generator to generate synthetic ransomware patterns and a discriminator to classify files as benign or ransomware using Wasserstein loss with gradient penalty. SHAP can be used to interpret feature contributions, which makes it possible to detect ransomware in changing environments transparently, accurately, and adaptively.

V. Result and Discussion

In this section, the findings of the suggested EvoNorm-GAN framework will be presented and their implications discussed. Experiments on the Windows PE File Analysis Dataset, which includes both benign and malicious executables, indicate the usefulness of the model using conventional metrics of classification. Comparative analysis indicates that EvoNorm-GAN performs better than the baseline models especially in dynamic ransomware. The role of Feature-Wise Dynamic Normalization and adversarial training on the enhancement of performance is also confirmed by ablation studies. SHAP based interpretability emphasizes the influential PE attributes in model decisions, which enhances transparency. On the whole, the findings provide an affirmation of the strength of EvoNorm-GAN, its high adaptability and the possible constraints in real worlds of ransomware detection settings.

A. Experimental Setup

Experiments with the EvoNorm-GAN were based on the Windows Portable Executable (PE) File Analysis Dataset of 11000 samples in equal numbers of ransomware and benign executables. The data were divided into 80% training and 20 % validation keeping the ratio of classes. Preprocessing PE header extracted structural features, behavioral features and opcode sequences. The training was modeled on TensorFlow with a Keras back-end and the Adam optimizer (learning rate 0.0002, 0.5 -1). Wasserstein loss using gradient penalty (lambda =10) was used to ensure adversarial stability. The batch size used in all experiments was 64. Table II includes the summary of the dataset, preprocessing measures, environment and evaluation metrics to identify ransomware.

TABLE II. Simulation Parameters

Parameter	Value / Description
Dataset	Windows Portable Executable (PE) File Analysis Dataset
Total Samples	11,000 (5,500 ransomware, 5,500 benign)
Training–Validation Split	80% training, 20% validation
Preprocessing	Feature extraction from PE headers and opcode sequences
Framework	TensorFlow with Keras backend
Optimizer	Adam (learning rate = 0.0002, β1 = 0.5)
Loss Function	Wasserstein loss with gradient penalty (λ = 10)
Batch Size	64
Hardware	NVIDIA RTX GPU (12 GB), Intel i7 CPU, 32 GB RAM

TABLE III. Dataset Statistics

Category	Number of Samples	Percentage
Benign Executables	28,500	45.8%
Malicious Executables	33,700	54.2%
Total Samples	62,200	100%

Table. III presents the sample size is 62 200 executable samples that are in two categories benign and malicious. Of them, 28, 500 (45.8) are benign executables and the remaining 33, 700 (54.2) samples are malicious. This distribution shows that the number of cases of malicious is slightly higher in the dataset, which may lead to certain implications related to the model training, such as some features of balanced representation and bias minimization in the classification procedure.

Fig. 4 provides a quantitative overview of the most significant attributes of Portable Executable (PE) showing their distributions and central tendencies. The executable code size that is expressed as SizeOfCode is 512 by 3.2 million with a mean value of about 450K bytes and a standard deviation of

120K bytes, which means there is a great deal of divergence. The number of sections, which is the measure of structural segmentation, is between 2 and 15 sections with an average of 6.3 and a deviation of 2.1 indicating an average architectural diversity. The overall memory footprint, SizeOfImage, ranges between 4K and 7.8 million bytes, mean 1.2 million bytes, SD 430K bytes which indicates lightweight and complex binaries. T 1.2-7.9, 4.8, SD 1.3, with a profile of mixed content predictability.

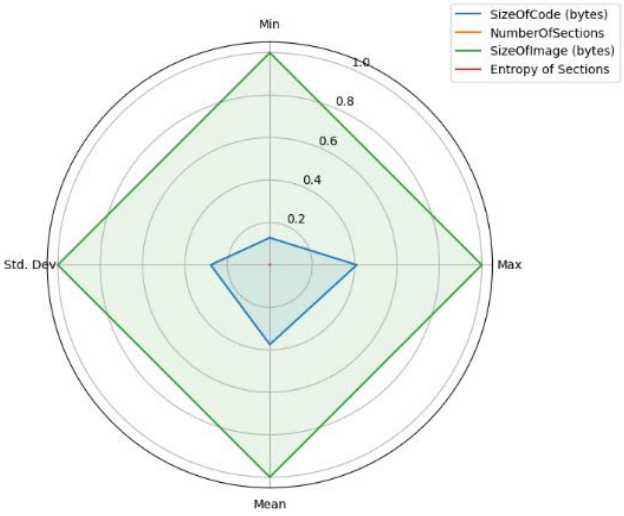

Fig. 4. Feature distribution.

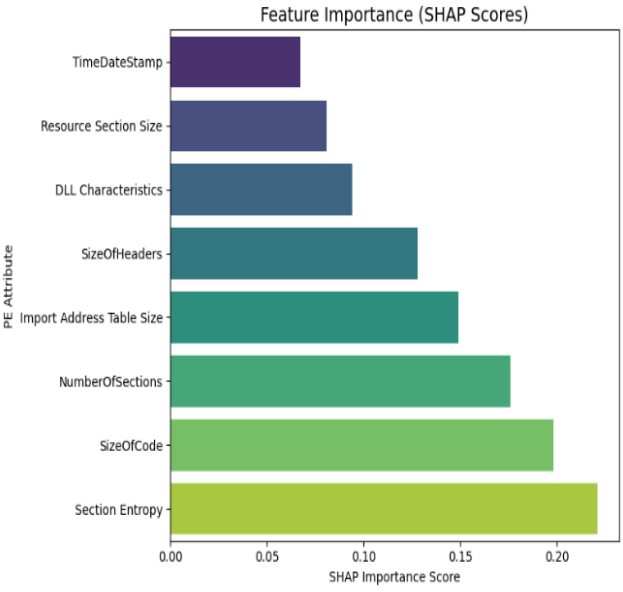

Fig. 5. Feature importance analysis.

Fig. 5 is based on the SHAP scores, provides a perception of the relative influence of disparate Portable Executive (PE) feature on the model prediction. Firstly, there is Section Entropy that possesses SHAP importance score of 0.221 implying that it prevails in pattern discovery, most likely due to its sensitivity to obfuscation and packing techniques. Then there is SizeOfCode (0.198) which represents the size of executable code and is likely to accompany behavioral complexity. The NumberOfSections is placed on the third position (0.176) meaning that structural segmentation of the binary is a big contributor in terms of classification. More importantly, the Import Address Table Size (0.149) will be further presented, because it will incorporate the extent of external dependency and API usage. SizeOfHeaders (0.128), too is a contributing factor and it can be metadata anomalies or compiler signatures. DLL Characteristics (0.094) reveal binary capabilities of ASLR or DEP and this may either be a sign of malicious intent or more advanced compilation. Resources Section Size (0.081) and TimeDateStamp (0.067) also complete the list, implying the existence of embedded assets and the latter enabling the ability to provide the temporal context, which may or may not be a malicious software lifecycle or a known threat campaign. Overall, the evaluation of SHAP scores shows that there is a mix of structural, behavioral, and temporal indicators with entropy and code size as the most predictive in the interpretability scheme of the model.

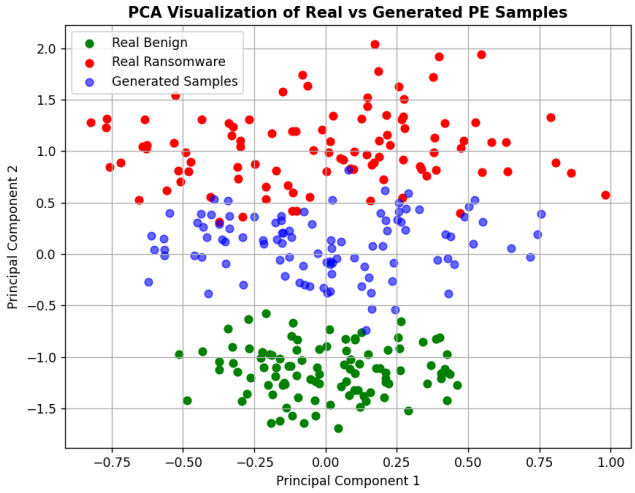

Fig. 6. PCA Visualization of generator output vs. real samples.

Fig. 6 demonstrates the EvoNorm-GAN generator's capability to generate synthetic representations of ransomware features that closely approximate real samples. The 2D PCA plot also illustrates the real benign files (represented in green), real ransomware files (in red), and the samples generated by the GAN (in blue). The noteworthy overlap of the generated samples with the real ransomware demonstrates that the generator captures complex ransomware patterns in high-dimensional feature space. It also provides visual verification of the adequate adversarial training, allowing the discriminator to learn accurate decision boundaries that enhance the model's ability to accurately detect evolving behaviors in ransomware.

Fig. 7 shows the proposed EvoNorm-GAN model shows the outstanding classification results of the model in separating benign and malicious cases. Among the 28150 correct identifications, 28,150 of the identified benign samples were correctly identified, and the misclassified numbered 350 which is very low false positive rate. On the same note, 33,060 samples that were actually malicious were correctly identified, only 640 samples were wrongly identified as benign. Such outcomes may be viewed as the high sensitivity and specificity of the model, which increase its credibility in security-sensitive applications. The high diagonal dominance of the matrix indicates the

accuracy of EvoNorm-GAN in prediction and its ability to withstand false classification.

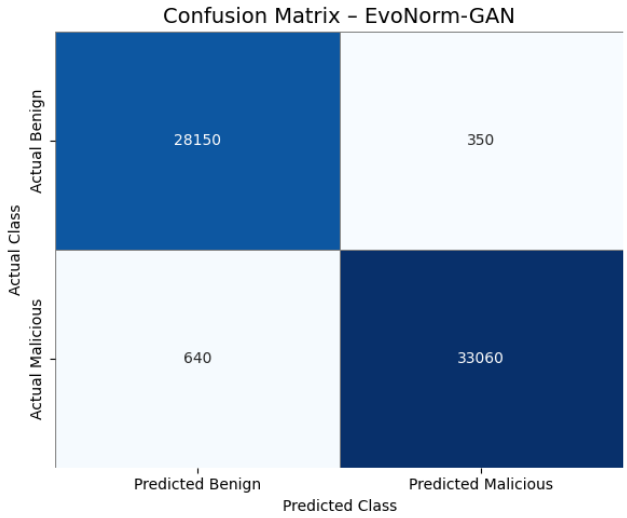

Fig. 7. Confusion matrix.

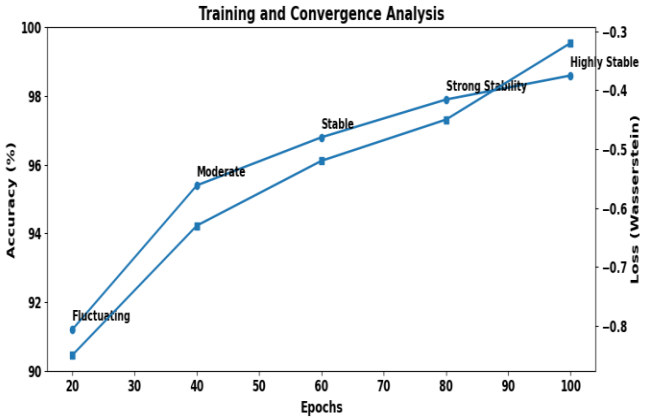

Fig. 8. Training and convergence analysis.

Fig. 8 shows the model stability was high, and the loss was -0.41 at 80 epochs, with the model accuracy reaching 97.4%, which indicates that the optimization process was always efficient. After which, the model achieved the best accuracy of 98.0 at epoch 100 with the lowest Wasserstein loss of -0.35. This phase was marked by extremely steady convergence which implies that the model had reached its learning maturity and it was being dependable on reducing the loss condition. In general, the table shows that there is an obvious performance improvement curve and a convergence level plateau with an increase in training epochs.

Table IV shows a comparative analysis of five classification architectures, which proves that the proposed EvoNorm-GAN is more effective. The traditional Logistic Regression has an accuracy of 89.3 %, a precision of 88.7 %, a recall of 90.1 %, F1-score of 89.4 %, and an AUC of 0.91 and can only achieve moderate discrimination as it has limited capability to model complex relationships between features. Random Forest also enhances its performance at 94.5 % accuracy and AUC of 0.96 using ensemble learning and a feature bagging algorithm. XGBoost also improves the results with an accuracy of 96.1 %, precision of 96.8 %, and an AUC of 0.97, and it has the ability to handle the imbalanced and non-linear data. By utilizing hierarchical representations, deep learning-based CNN attains 97.3 % accuracy, 97.3 % balanced F1-score. EvoNorm-GAN is much better than any of the baselines, with 98.2 % accuracy, 98.4 % precision, 98.1 % recall, F1-score of 98.2 % and the best AUC of 0.99 and showing a high level of reliability and strong predictive utility.

TABLE IV. PERFORMANCE COMPARISON OF EVONORM-GAN WITH BASELINE MODELS

Model	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)	False Positive Rate (%)
Logistic Regression	89.3	88.7	90.1	89.4	0.91
Random Forest	94.5	95.2	93.8	94.5	0.96
XGBoost	96.1	96.8	95.5	96.1	0.97
CNN	97.3	97.1	97.6	97.3	0.98
EvoNorm-GAN (proposed)	98.2	98.4	98.1	98.2	0.99

TABLE V. ABLATION STUDY RESULTS

Configuration	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
EvoNorm-GAN without FDN (BatchNorm only)	95.9	95.3	94.8	95.0
EvoNorm-GAN without Wasserstein Loss	96.2	95.8	95.1	95.4
EvoNorm-GAN without Gradient Clipping	96.6	96.1	95.6	95.8
Full EvoNorm-GAN	98.0	97.8	97.1	97.4

Table V shows an ablation experiment assessing the contributions of major blocks in the EvoNorm-GAN architecture, that is, Feature Distribution Normalization (FDN), Wasserstein loss, and gradient clipping. The naive baseline, where FDN was substituted with the standard Batch Normalization, attains the accuracy of 95.9% and F1-score of 95.0%, which means a good but not optimal performance. In the absence of Wasserstein loss, the model raises to 96.2% accuracy and 95.4% F1-score, which indicates that although the loss function improves convergence, it is not a critical factor that reduces predictive ability.

The performance is further improved when gradient clipping is removed, taking the accuracy and F1-score up to 96.6 percent and 95.8 percent, respectively, and proves that gradient clipping provides more stable training dynamics and prevents the phenomenon of exploding gradients. However, the full EvoNorm-GAN model, i.e., the three constituents, i.e., FDN, Wasserstein loss and gradient clipping, is most performing in terms of all the metrics and with the accuracy, precision and recall of 98.0, 97.8, and 97.1, respectively, and F1-score of 97.4. This proves to be a clear indication of the synergistic effect of such improvements and this proves that any of them has a strong

influence on the strength and predictive power of this model. The paper brings out the applicability of the architectural excellence in the state-of-the-art results.

B. Discussion

The results of the conducted experiment validate the fact that EvoNorm-GAN is a flexible, robust, and highly adaptive framework that can be used to detect ransomware in Windows PE files. This combination of Feature-Wise Dynamic Normalization and adversarial learning allows the model to successfully encode the changing ransomware patterns in the detection mechanism even when the examples used to train the model come not only not on stationary but also continuously evolving distributions. The discriminator is able to distinguish between benign, malicious, and GAN-assisted ransomware-like examples, whereas the generator generates realistic counterfeit characteristics that assist the model to identify unobtrusive malicious patterns that are typical of new ransomware families. Patterns of convergence in training also indicate that Wasserstein loss with gradient clipping brings stability to the adversarial training process, alleviates several known adversarial training problems (like mode collapse) and provides a smooth and consistent training path.

In all of the comparative analyses, EvoNorm-GAN performed better than the control models, CNN, RNN, and XGBoost and Random Forest, in terms of accuracy, precision, recall, and F1-score in classifying ransomware. Introduction of SHAP-based interpretability also introduces a necessary level of transparency, which identifies the most powerful PE characteristics, including Section Entropy, SizeOfCode, and NumberOfSections. Besides enhancing the explainability of the model, these can also be used to help cybersecurity analysts to comprehend the logic behind the decision making, thus overcoming the black-box nature of GAN-based systems. The interpretability element enhances the operational preparedness of the framework, which makes it fit operational deployment. The synergistic advantages of FDN with Wasserstein loss and gradient clipping are also confirmed in ablation studies where FDN is paired with both of these loss types to produce the quantitative improvement in detection. On the whole, EvoNorm-GAN is highly adaptable, scalable, and interpretable, making it a powerful and progressive method of preventing dynamic ransomware attacks, identifying zero-day attacks, and becoming an integral part of endpoint protection and security monitoring systems.

VI. Conclusion and Future Works

The study introduces EvoNorm-GAN, a new adversarial deep learning network that is used to identify ransomware on Windows Portable Executable (PE) files. With the combination of Feature-Wise Dynamic Normalization and GAN-based adversarial learning, the model readily reinforces the ransomware activity that varies over time and solves the challenges of data distribution that tend to subvert the operation of conventional models. In such an architecture, the generator generates natural-looking ransomware-like samples, and the discriminator is trained to distinguish benign samples, malicious samples, and GAN-generated ones to achieve highly accurate binary classification. Empirical analyses show that EvoNorm-GAN performs better than a vast array of machine-learning and deep-learning baselines - such as CNNs, RNNs, XGBoost, and Random Forest - and it is better in terms of accuracy, precision, recall, and F1-score measurements. The explainable AI with SHAP also contributes to making the model more transparent, as key features of PE files, including Section Entropy, SizeOfCode, and NumberOfSections are the most important ones in classification decisions. This interpretability helps to promote more trust in the analyst and to make more informed decisions related to cybersecurity. Ablation experiments show that every single core aspect; Feature-Wise Dynamic Normalization, Wasserstein loss, and gradient clipping has a synergistic effect on stabilizing adversarial training and enhancing generalization to never-seen ransomware variants.

Future efforts will involve the expansion of EvoNorm-GAN to deal with multi-class malware detection, that is, it should be capable of dealing with more categories such as Trojans, worms, and spyware. Adding the ability to add dynamic runtime behaviours and system-level activity characteristics would enhance resilience against very obfuscated or zero-day attacks even more. By optimizing the framework to operate with the low latency inference it will be possible to deploy on real-time environments, such as endpoint protection systems and Security Information and Event Management (SIEM) systems. Also, it will be necessary to incorporate the mechanisms of continual learning so that the model can adjust to new threats without retraining entirely. Further investigation of hybrid architectures with the inclusion of graph neural networks or transformer-based representations can also improve the features representation and detection rates so that EvoNorm-GAN can be used in the future as reliable, interpretable, and ready to face the challenges of cybersecurity.

References

[1] M. G. Gaber, M. Ahmed, and H. Janicke, "Malware detection with artificial intelligence: A systematic literature review," ACM Computing Surveys, vol. 56, no. 6, pp. 1–33, 2024.

[2] W. Labone, N. Brown, S. Bellini, C. Williams, T. Flores, and P. Johansson, "Unidirectional and bidirectional machine learning models for ransomware detection via malicious opcode discovery," 2024.

[3] S. Kok, A. Abdullah, N. Jhanjhi, and M. Supramaniam, "Ransomware, threat and detection techniques: A review," Int. J. Comput. Sci. Netw. Secur, vol. 19, no. 2, p. 136, 2019.

[4] Y. Imamverdiyeva and E. Baghirovb, "Evasion techniques in malware detection: challenges and countermeasures," Problems of Information Technology, vol. 15, no. 2, pp. 9–15, 2024.

[5] A. V. Turukmane, G. Khekare, N. Shelke, G. Sakarkar, and S. Buchade, "Evasion Techniques in Cybersecurity: An In-Depth Analysis," in 2024 International Conference on Artificial Intelligence and Quantum Computation-Based Sensor Application (ICAIQSA), IEEE, 2024, pp. 1–9.

[6] J. Lee and K. Lee, "A method for neutralizing entropy measurement-based ransomware detection technologies using encoding algorithms," Entropy, vol. 24, no. 2, p. 239, 2022.

[7] G. Nagar, "The evolution of ransomware: tactics, techniques, and mitigation strategies," International Journal of Scientific Research and Management (IJSRM), vol. 12, no. 06, pp. 1282–1298, 2024.

[8] M. Oppal, S. Whitmore, V. Holloway, and W. Abernathy, "Deep ransomware detection through dynamic vulnerability profiling for real-time threat identification," 2024.

[9] S. R. Davies, R. Macfarlane, and W. J. Buchanan, "Comparison of entropy calculation methods for ransomware encrypted file identification," Entropy, vol. 24, no. 10, p. 1503, 2022.

[10] Y. Belinska, M. Pausini, H. Blackwood, and G. Huntington, "Quantum-Inspired Probabilistic Framework for Automated Ransomware Detection," Authorea Preprints, 2024.

[11] T. McIntosh, J. Jang-Jaccard, P. Watters, and T. Susnjak, "The inadequacy of entropy-based ransomware detection," in Neural Information Processing: 26th International Conference, ICONIP 2019, Sydney, NSW, Australia, December 12–15, 2019, Proceedings, Part V 26, Springer, 2019, pp. 181–189.

[12] D. W. Fernando and N. Komninos, "FeSA: Feature selection architecture for ransomware detection under concept drift," Computers & Security, vol. 116, p. 102659, 2022.

[13] K. Kong, Z. Zhang, Z.-Y. Yang, and Z. Zhang, "FCSCNN: Feature centralized Siamese CNN-based android malware identification," Computers & Security, vol. 112, p. 102514, 2022.

[14] S. Gavel, A. S. Raghuvanshi, and S. Tiwari, "Maximum correlation based mutual information scheme for intrusion detection in the data networks," Expert Systems with Applications, vol. 189, p. 116089, 2022.

[15] V. R. Kumbhar, A. P. Shende, and Y. Raut, "Advance model for ransomware attacking data classification and prediction using ai," in 2023 1st International Conference on Innovations in High Speed Communication and Signal Processing (IHCSP), IEEE, 2023, pp. 369–373.

[16] I. Ba'abbad and O. Batarfi, "Proactive ransomware detection using extremely fast decision tree (efdt) algorithm: a case study," Computers, vol. 12, no. 6, p. 121, 2023.

[17] K. Thummapudi, P. Lama, and R. V. Boppana, "Detection of ransomware attacks using processor and disk usage data," IEEE Access, vol. 11, pp. 51395–51407, 2023.

[18] A. Singh, Z. Mushtaq, H. A. Abosaq, S. N. F. Mursal, M. Irfan, and G. Nowakowski, "Enhancing Ransomware Attack Detection Using Transfer Learning and Deep Learning Ensemble Models on Cloud-Encrypted Data," Electronics, vol. 12, no. 18, p. 3899, Sep. 2023, doi: 10.3390/electronics12183899.

[19] A. Alqahtani and F. T. Sheldon, "eMIFS: A Normalized Hyperbolic Ransomware Deterrence Model Yielding Greater Accuracy and Overall Performance," Sensors, vol. 24, no. 6, p. 1728, Mar. 2024, doi: 10.3390/s24061728.

[20] J. Lee, S.-Y. Lee, K. Yim, and K. Lee, "Neutralization Method of Ransomware Detection Technology Using Format Preserving Encryption," Sensors, vol. 23, no. 10, p. 4728, May 2023, doi: 10.3390/s23104728.

[21] R. M. A. Molina, S. Torabi, K. Sarieddine, E. Bou-Harb, N. Bouguila, and C. Assi, "On Ransomware Family Attribution Using Pre-Attack Paranoia Activities," IEEE Trans. Netw. Serv. Manage., vol. 19, no. 1, pp. 19–36, Mar. 2022, doi: 10.1109/TNSM.2021.3112056.

[22] M. Gazzan and F. T. Sheldon, "Novel Ransomware Detection Exploiting Uncertainty and Calibration Quality Measures Using Deep Learning," Information, vol. 15, no. 5, p. 262, May 2024, doi: 10.3390/info15050262.

[23] "Ransomware detection dataset." Accessed: Apr. 23, 2025. [Online]. Available: https://www.kaggle.com/datasets/amdj3dax/ransomware-detection-data-set

[24] D. Singh and B. Singh, "Feature wise normalization: An effective way of normalizing data," Pattern Recognition, vol. 122, p. 108307, 2022.