International Journal of Environmental Sciences
ISSN: 2229-7359
Vol. 11 No. 23s, 2025
https://theaspd.com/index.php

Hybrid Dto-Rsa And Gan-Bilstm Based Static And Dynamic Analysis Approach For Efficient Intrusion Detection In Android Application

Sanjeev Kumar[1*], Rajiv Singh[2], Dr Ankit Agarwal[3]
[1*]Banasthali Vidyapith, Rajasthan, India, sanjunic@hotmail.com
[2]Banasthali Vidyapith, Rajasthan, India, jkrajivsingh@gmail.com
[3]Dr Akhilesh Das Gupta Institute of Professional, Delhi, India, cs.ankit11@gmail.com

Abstract
An Intrusion Detection System (IDS) serves as a security mechanism that continuously observes android malware apps activities, aiming to identify unauthorized access, suspicious behavior, or policy breaches. While machine learning and deep learning enhance IDS capabilities, they face challenges such as adapting to evolving attack strategies, ensuring efficient real-time detection, and mitigating adversarial manipulation attempts. To address these challenges, the Hybrid GAN-BiLSTM model integrates adversarial learning for robust feature augmentation and uses Bi-LSTM's contextual patterns, ensuring enhanced resilience against imbalanced data and dynamic malware behaviors. The system processes a static- dynamic malware analysis dataset with advanced techniques such as Recurrent Batch Normalization (RBN) for stability, Stochastic Regression Imputation (SRI) for missing data, and Multiview Subspace Clustering (MSC) to remove redundancy. The feature selection is improved by the hybrid Dipper Throated Optimization - Reptile Search Algorithm (DTO- RSA) technique. To identify malware, a hybrid Generative Adversarial Network- Bidirectional Long Short-Term Memory (GAN-BiLSTM) model is used, which combines sequential pattern analysis and adversarial learning. The system predicts potential threats by refining feature extraction and classification process. The proposed method obtains a PPV of 89.20%, False Discovery Rate of 10.80% and F1_Score of 89.85% for malware prediction respectively. The proposed approach utilizes a GAN-BiLSTM model to enhance malware prediction accuracy by generating synthetic data and leveraging the Bi-LSTM for better feature learning. This advanced method improves the performance of an Intrusion Detection System by effectively handling both temporal features, ensuring more robust anomaly detection.
Keywords: *Intrusion Detection System, RSA, DTO, MSC, SRI, RBN, GAN, BiLSTM*

1. INTRODUCTION

Android malware apps are malicious applications that exploit vulnerabilities, steal data, or disrupt device functionality. Detecting intrusion, them involves analyzing behavioral patterns, permissions, and network activities to identify potential threats. In order to identify any security lapses, including malevolent assaults or policy infractions, it entails tracking and evaluating system activity [1]. IDS can be anomaly-based, which identifies departures from typical activity, or signature-based, which detects established attack patterns [2]. Efficient intrusion detection for static and dynamic analysis ensures rapid and accurate identification of potential threats by leveraging complementary analysis techniques [3]. Static analysis evaluates system configurations and code without execution, detecting vulnerabilities or malicious patterns. Dynamic analysis observes runtime behavior to uncover anomalies or sophisticated attacks [4]. Combining these approaches enhances threat detection, reduces false positives, and strengthens system security. However, it can suffer from high false positive or false negative rates, leading to either unnecessary alerts or missed threats [5]. They may struggle to handle encrypted data or adapt to sophisticated, evolving attack strategies and also assessing encrypted traffic without decryption can limit detection effectiveness. Additionally, maintaining and fine-tuning these systems can be resource- intensive and complex. Alternative drawbacks of intrusion detection systems include limited scalability in handling large networks, which can impact performance [6]. They may also face challenges in effectively integrating with diverse network architectures.

Due to the limitations associated with traditional methods, the focus has shifted toward addressing drawbacks through machine learning techniques. Machine learning (ML) is a data-driven approach that empowers systems to recognize patterns and make informed decisions without explicit programming, leveraging advanced algorithms to enhance performance over time across various domains. Some of the ML techniques with drawbacks in detecting intrusion is Bayesian Linear Regression (BLR) assumes a linear relationship, limiting its effectiveness in capturing complex intrusion patterns, and may struggle with high- dimensional data in dynamic

attack scenarios [7]. Boosted Decision Tree Regression (BDTR) may inadvertently favor certain feature subsets, potentially overlooking critical but less frequent attack indicators. Handling hyperparameter variations in BDTR alongside optimization could introduce instability, impacting the model's robustness against evolving threats [8]. The restricted interpretability of models is one of the difficulties that the Adaptive Neuro Fuzzy Inference System (ANFIS) may encounter, making it challenging to comprehend specific detections. Additionally, the effectiveness of models can degrade over time due to evolving malware techniques, requiring continuous retraining and adaptation [9]. Machine learning techniques often encounter several limitations, leading to a transition toward deep learning methods to better tackle these challenges [10].

Deep learning is a sophisticated branch of machine learning that utilizes deep neural networks to model intricate patterns and relationships in data. By learning hierarchical features, it excels in solving complex problems across various fields with unparalleled accuracy [11]. Convolutional Recurrent Neural Network (CRNN) is difficulty in handling highly imbalanced classes, which can lead to biased detection. Additionally, the dynamic nature of evolving attacks requires frequent updating, creating challenges in maintaining up- to-date models [12]. Cuda-enabled Bidirectional Long Short-Term Memory (Cu-BLSTM) in smart ID networks faces challenges like frequent erroneous detections due to complex attack patterns and difficulty in generalizing across diverse device behaviors. Additionally, models may struggle with scalability as the number of devices and data traffic continues to grow [13]. Gated Recurrent Unit (GRU) for IDS may struggle with elevated false alarms and difficulties in interpreting decisions, reducing trust in automated security systems. Additionally, training on imbalanced datasets could lead to suboptimal detection of rare attack types [14]. When dealing with multi-attribute data, Multi-head Self-Attention based Gated Graph Convolutional Networks (MSA-GCNN) for intrusion detection frequently encounter challenges that result in lengthy training durations and high resource requirements. Additionally, adversarial attacks can significantly degrade the model's performance by exploiting vulnerabilities in the network [15].To overcome these drawbacks, a hybrid GAN- BiLSTM classifier is utilized for malware prediction, combining GAN's ability to generate
high-quality synthetic data with Bi-LSTM's strength in capturing long-term dependencies. This approach enhances data availability, robustness, and predictive accuracy while mitigating interpretability and resource constraints.

Major contribution of the work is given below.

- ❖ A Hybrid Deep Learning algorithm is designed for an Intrusion Detection System to enhance malware prediction in android apps.
- ❖ Recurrent batch normalization, Stochastic Regression imputation (SRI) and Multiview Subspace Clustering (MSC) for Refining data by ensuring consistency, handling missing values, and reducing redundancy.
- ❖ Dipper Throated Optimization (DTO) and Reptile Search Algorithm (RSA) are used to select the best features from pre-processed data, with RSA enhancing the search process for efficient and accurate convergence.
- ❖ A Hybrid Deep Learning GAN-BiLSTM classifier is employed to enhance the accuracy of malware prediction.
- ❖ The performance of the proposed classifier is analyzed by evaluating FDR, FOR, and specificity.

This research work is divided into the following sections: The second section looks at a lot of studies that effectively use deep learning and machine learning techniques for both static and dynamic analysis for intrusion detection systems. Section 3 describes the proposed strategy for the Static and Dynamic Analysis Approach for an Effective Intrusion Detection System. Results of a proposed approach are presented in Section 4. The conclusion of an entire research is described in Section 5.

2. LITERATURE SURVEY

Numerous methods are available for Static and Dynamic feature Analysis for Intrusion Detection System. Some of the current prediction approaches are studied and reviewed below.

Mishra S, Mahanty C, et al [16] have developed a Best First Search-Naive Bayes (BFS-NB) for Intrusion Detection System. The study entails creating a data mining-based intrusion detection model called BFS-NB, which combines a classification method with a feature selection technique. While the Naive Bayes (NB) classifier predicts different network-based assaults, the Best First Search (BFS) approach was used to reduce dimensionality. Accuracy, sensitivity, and time delay were among the characteristics used to assess the model's performance. According to experimental data, BFS-NB provides an efficient intrusion detection system and

operates at peak efficiency. The method's 92.5% accuracy shows how effective the ideal attribute set is in improving categorization. Because of this, BFS-NB is a dependable option for experts.

Mehedi ST, Anwar A, et al [17] have created a Proposed-Residual Neural Network (P- ResNet) for a Reliable Intrusion Detection System for the Internet of Things that was based on transfer learning. With a ResNet-based architecture and efficient attribute selection, this study presents a robust IDS model based on deep transfer learning that was suited for situations with little labeled data. The robustness, effectiveness, and enhanced performance of the model are demonstrated by a thorough experimental evaluation utilizing real-world data. With an accuracy of 0.87%, precision of 0.88%, recall of 0.86%, F1-score of 0.86%, and ROC AUC of 0.83%, the constructed P-ResNet model produces ideal results, guaranteeing reliability in intrusion detection.

Shareena J, Ramdas A, et al [18] have demonstrated a Deep Neural Network (DNN) for Intrusion Detection System for IOT Botnet Attacks. The dataset was developed in a realistic network environment at the Cyber Range Lab of UNSW Canberra Cyber, incorporating both normal and attack traffic data. A robust and extensible Deep Neural Network (DNN) was designed specifically for IoT networks to efficiently identify botnet attacks. The results of the experiments show that the new DNN performs better than the current systems. With a 94% accuracy rate, the model was a dependable IoT security solution.

Park D, Kim S, et al [19] for host-based intrusion detection, they developed a Siamese Convolutional Neural Network (Siamese-CNN). It assessed a built model using the Leipzig Intrusion Detection Data Set (LID-DS), a host-based intrusion detection dataset. The model uses a Siamese Convolutional Neural Network (Siamese-CNN) with few-shot learning for efficient training on sparse data, and it incorporates pre-processing, vector-to-image processing, and training/testing phases. Based on the similarity scores of cyber-attack samples that have been transformed into pictures, Siamese-CNN recognizes different kinds of attacks. Recall was improved by 6% using the created Siamese-CNN, according to performance comparisons with Vanilla-CNN. The approach's efficacy was validated by the experimental findings, which demonstrated an accuracy of 93%.

Smmarwar SK, Gupta GP, et al [20] provided an example of an effective and optimized Ensemble Learning-based Android Malware Detection (OEL-AMD) system for detecting Android malware. The Detection framework (OEL-AMD) encodes important features and removes non-informative ones using statistical feature engineering. For the best feature selection in both static and dynamic layers, Binary Grey Wolf Optimization (BGWO) was utilized. Hyper-parameter tuning is used to train different base learners in order to improve the performance of the ensemble model. For category categorization, the greatest classification accuracy of 83.49% was attained. The importance of the results was confirmed by statistical testing and comparison with current methodologies. The framework's efficacy in detecting Android malware is encouraging.

2.1Open challenges

From the above mentioned reviews some of the existing Static and Dynamic Analysis for Intrusion Detection System methods are dealing with some of the drawbacks like potential difficulty in handling evolving attack patterns, Additionally, the model may struggle with maintaining high performance under dynamically changing network environments and fluctuating attack strategies [16]. The model's reliance on data distribution may lead to reduced detection effectiveness for infrequent or emerging threats. Moreover, the requirement for significant computational power during training could make real-time deployment challenging on resource-constrained IoT devices [17]. The model struggles to effectively recognize emerging attack patterns, requiring continuous updates to maintain its accuracy. Additionally, scalability issues may arise when the model was deployed in large-scale IoT environments with diverse devices and varying traffic patterns [18]. Moreover, the model's ability to adapt to new attack types in evolving environments may be constrained by the availability of representative data during few-shot learning. The approach could also face inefficiency in real-time applications due to the computational cost involved in processing and analyzing host-based intrusion data [19]. However it make difficult to reliably to identify uncommon attack types which affects detection effectiveness. Furthermore, in a complex network environment, the parameter evaluation method may miss crucial feature connections resulting in less-than-ideal performance [20]. To mitigate these limitations, a hybrid GAN- BiLSTM model is developed for malware prediction, leveraging GANs to generate synthetic balanced data and BiLSTM networks to capture complex temporal dependencies in attack patterns. This combination improves model adaptability, reduces sensitivity to imbalanced data, and enhances real-time prediction capabilities.

3. PROPOSED METHODOLOGY

Number of applications (apps) created for the mobile ecosystem has increased dramatically as smartphones

become more and more integrated into daily life. These applications are made to accommodate a variety of user requirements, such as handling sensitive data, social networking, banking, and communication. Because of this, an increasing number of cybercriminals are focusing on the mobile environment by creating and disseminating harmful programs that either steal data or injure the user of the device. Numerous methods, based on either static or dynamic analysis, have been employed to model Android malware and create detection tools in an effort to combat them. In this paper, a lightweight IDS using hybrid optimization based feature selection and hybrid deep learning approaches GAN- BiLSTM is developed by considering both static and dynamic behavior analysis. The overall workflow of the proposed approach is shown in Figure 1 below.

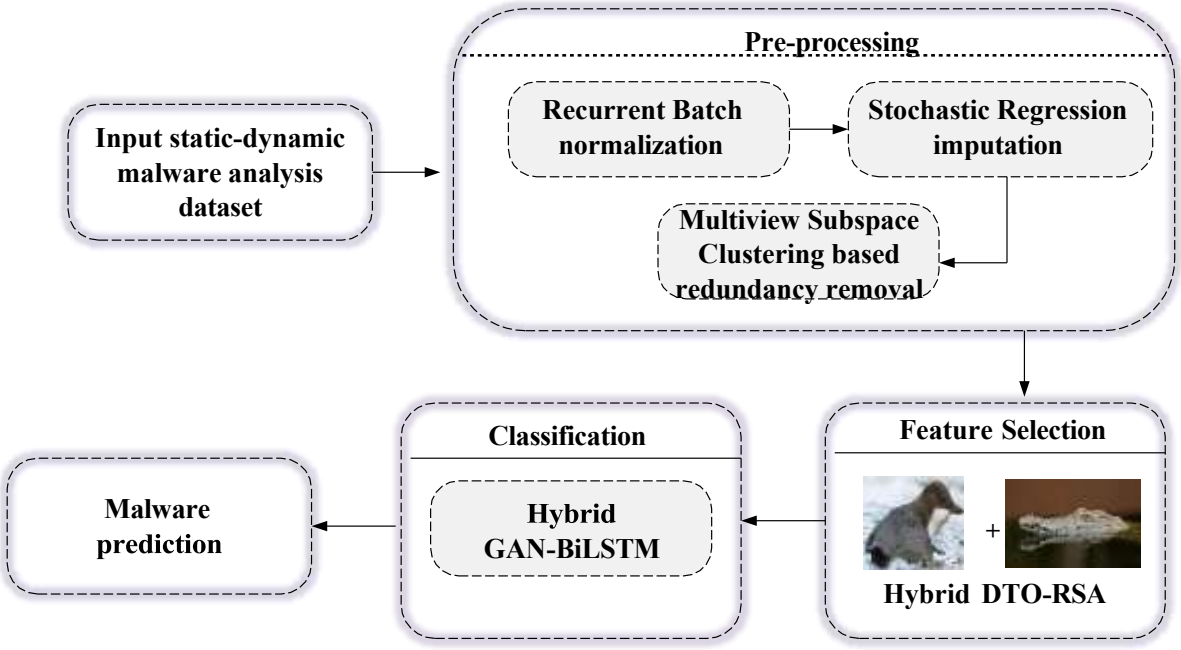

Figure 1: Architecture of the proposed Intrusion Detection System for android app malware prediction.

Both static and dynamic behavior analysis are gathered and utilized as an input dataset in this proposed approach. These collected data are initially preprocessed using data normalization, missing value imputation and redundancy removal approaches for preparing the data better and more appropriate for further processing. In the pre-processing phase, the input raw data undergoes standardization using the Recurrent Batch Normalization (RBN) approach, while missing data is handled with Stochastic Regression Imputation (SRI) for accurate replacement. To address redundancy and high data dimensionality, the Multiview Subspace Clustering (MSC) algorithm is employed for effective redundancy removal. Following pre- processing, feature selection is conducted to identify the most relevant set of features. To tackle the feature selection challenge, a novel hybrid binary meta-heuristic algorithm is proposed, combining Dipper Throated Optimization (DTO) and Reptile Search Algorithm (RSA). The RSA enhances the exploration process, ensuring swift and precise convergence of the optimization algorithm. Ultimately, the selected features are passed to a hybrid classifier, GAN-BiLSTM, for efficient malware prediction. In this hybrid classifier, the fully connected layers of GAN is replaced by BiLSTM.

3.1Pre-processing

Preprocessing transforms raw data into a structured, clean format, enhancing its quality for analysis. The preprocessing phase enhances data quality by applying Recurrent Batch Normalization to normalize inputs and improve stability. Stochastic Regression Imputation addresses missing data effectively, while Multiview Subspace Clustering reduces redundancy, ensuring refined data for further analysis.

3.1.1 Recurrent Batch normalization

Recurrent Batch Normalization (RBN) is employed to standardize raw input data by normalizing features across each batch, ensuring consistent scaling and improved model convergence [21]. This approach enhances the stability of recurrent models by addressing issues like internal covariate shifts. By maintaining uniformity in the input data, RBN accelerates the learning process and optimizes performance.

Normalizing input data in deep neural networks accelerates model convergence, but this effect is limited to the first layer, while inner layers experience changes in input distributions during training, known as internal covariate shift. Batch normalization addresses this issue by normalizing inputs to the inner layers during mini-batch training. For a mini-batch B with m samples, batch normalization is applied independently to each input dimension. This mechanism ensures stable training by mitigating the effects of covariate shifts across layers as shown in eqn (1)

$$\bar{X}_i = \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} \quad (1)$$

Where, $\bar{X}_i$ denotes the batch normalizing transform, x_i represents the input dimension, is the mean and $\mu_B \text{ and } \sigma_B^2$ or the dimension, ϵ denotes the small constant added for stability.

RBN is employed in intrusion detection systems to standardize raw input data, stabilizing training and ensuring consistent scaling across batches. This approach addresses internal covariate shifts, enhancing the detection of complex malware patterns. By normalizing data inputs, RBN improves the convergence speed and accuracy of models in malware prediction tasks as shown in eqn (2)

$$y_i = \gamma \bar{x}_i + \beta \quad (2)$$

Where, γ and β represents the parameters learned during training along with parameter, y_i is the final transformed output.

3.1.2 Stochastic Regression imputation

Stochastic Regression Imputation (SRI) is used in intrusion detection systems to handle missing data in the input dataset, ensuring data completeness for malware prediction [22]. By estimating missing values through regression models with added randomness, SRI preserves data variability and reduces bias. This enhances the system's ability to accurately detect and predict malware threats.

SRI is applied in intrusion detection systems to address missing data challenges in malware prediction. By leveraging ordinary least squares (OLS) regression and incorporating a random residual component, SRI preserves variability and minimizes bias. This approach enhances the dataset's reliability, supporting accurate malware detection as shown below in eqn (3)

$$\hat{y}_{ik} = \sum_{j=1}^{k-1} \beta_j x_{ij} + z_i \quad (3)$$

Where, $\hat{y}_{ik}$ denotes the variable of the missing value, z_i represents the random draw, $k-1$ is the imputation for $k-th$ variable, β_j denotes the elimated regression coefficients for the j- th variable, $j = 1$ represents the summation begins with the first variable in the dataset, x_{ij} is the observed value.

3.1.3 Multiview Subspace Clustering

Multiview Subspace Clustering (MSC) is employed in intrusion detection systems as a redundancy removal algorithm to address high data dimensionality and redundancy issues in malware prediction [23]. By integrating information from multiple feature views, this technique identifies meaningful subspaces that capture the most relevant patterns. It reduces noise and redundancy, improving the clarity and efficiency of data analysis. This optimization enhances the system's capability to detect and predict malware threats effectively.

MSC identifies shared subspaces across multiple views to reduce redundancy and high dimensionality. This approach enhances data representation and improves malware prediction in intrusion detection systems as shown below in eqn (4)

$$\min_{Z} f(Z) - ||X - XZ||_F^2 + \lambda \Omega(Z) \quad (4)$$

Where, Z denotes the Subspace Structure, X represents the subspace clustering, F is the cluster indicator matrix, λ denotes the trade-off parameter, $\Omega(Z)$ is the certain regularization term.

Feature Selection

Feature selection in intrusion detection systems for malware prediction involves identifying and retaining the most relevant attributes from the dataset that contribute to distinguishing between malicious and non-malicious activities. This process enhances the model's efficiency by reducing dimensionality, improving accuracy, and minimizing redundancy. For the feature selection involves using the Hybrid DTO-RSA approach, which integrates optimization techniques for selecting the most relevant features. This step ensures improved performance by reducing redundancy and retaining significant attributes for malware prediction.

3.1.4 Hybrid Dipper Throated- Reptile Search Optimization Algorithm

In order to provide a balanced search procedure for the best answers, the Hybrid Dipper Throated-Reptile Search Optimization Algorithm (DTO-RSA) combines the exploration and exploitation stages of the Dipper Throated and Reptile Search Optimization algorithms. An entire flock of birds swimming and looking for food is modeled by the Dipper-Throated Optimization (DTO) [24] technique. The position and velocity of birds are represented using matrices. During the exploration phase, the position of the swimming agent and the velocity of the flying agent are updated. In the exploitation phase, the position of the flying agent is refined for optimal solutions. The Reptile Search Algorithm (RSA) mimics crocodiles' hunting behaviors in nature. It balances exploration and exploitation using four movement types: high walking, belly walking, hunting coordination, and hunting cooperation [25, 31, 33]. These movements help optimize the search process effectively. RSA enhances problem solving by leveraging crocodiles' strategic hunting patterns.

Step 1: Initialization

The population of the Dipper Throated is used to initialize the features, and the mathematical formula for this initialization function is described in eqn (5)

$$D = D_1, D_2, \ldots \ldots, D_n \quad (5)$$

The features are represented by the symbol D, with upper bound is 123 and lower bound is 43.

Step 2: Fitness function

The fitness function for this optimization strategy aims to minimize the error value, thereby selecting the most suitable nodes for participation using eqn (6) and eqn (7).

$$fitness\ function = minimize(error) \quad (6)$$

$$E = {}^{1}/_{2}\ \sum_{i=1}^{I}(H_n - K_{ii})^2 \quad (7)$$

In this illustration, H_n denotes the expected value, K_{ii} denotes the actual value, and I represents the total number of data points.

Step 3: Updation

The fitness value is adjusted according to the reptile search behavior during both the exploration and exploitation phases. Exploration is performed through the encircling phase, and exploitation is handled by the hunting phase. The search is driven by four types of crocodile movements: high walking, belly walking, hunting coordination, and hunting cooperation. These movements guide the optimization process during the exploration and exploitation stages. The behaviors ensure the optimization algorithm adapts effectively during both phases. In the exploitation phase, crocodiles perform local searches, leveraging hunting coordination and cooperation. Hunting coordination is applied when $t \leq \frac{3T}{4}$ and hunting cooperation takes over when $t > T$ and $t \in \left[\frac{T}{2}, \frac{3T}{4}\right]$ these behaviors are modeled in the Reptile Search Optimization Algorithm to balance global and local search.

Exploration phase *(Encircling phase)*

Crocodiles surround the prey in the search area during the exploring stage. Their belly and high walking motions, which are precisely engineered for efficient encirclement, accomplish this. The number of iterations in the procedure determines the walking mode. The specific mathematical expressions governing these walking modes are outlined in eqn. (8) and (9). These equations guide the crocodiles' movements to ensure a thorough exploration of the search space. The goal is to effectively search and encircle the prey during the exploration phase.

$$X_{(i,j)}(t+1) = Best_j(t) - \eta_{(i,j)}(t) \times \beta - R_{(i,j)}(t) \times rand, \quad if\ t \leq \frac{T}{4} \quad (8)$$

International Journal of Environmental Sciences
ISSN: 2229-7359
Vol. 11 No. 23s, 2025
https://theaspd.com/index.php

$$X_{(i,j)}(t+1) = Best_j(t) \times X_{r1,j} \times ES(t) \times rand, if\ t \leq \frac{T}{2}\ and\ t > \frac{T}{4} \quad (9)$$

Where, $X_{(i,j)}$ denotes the generated position of $i-th$ individual in the $j-th$ dimension, $Best_j(t)$ represents the current best position with the population, t is the current iteration number, $\eta_{(i,j)}(t)$ denotes the hunting operator, β represents the constant value, $R_{(i,j)}(t)$ represents the reduce function, T is the maximum iteration number, $r1$ denotes the random integer between $[1, N]$, $ES(t)$ represents the probability ratio called evolutionary sence lying in $[-2,2]$.

- ***Exploitation phase*** *(Hunting phase)*

In the exploitation phase, crocodiles utilize both hunting coordination and cooperation for local search. Hunting coordination involves working in sync to capture prey efficiently. Cooperation allows crocodiles to share information about prey locations, enhancing their hunting success. This phase focuses on refining strategies by leveraging the group's collective strength. The collaboration and coordination between crocodiles help them secure food with precision. Their teamwork ensures the optimization of resources within a localized area as shown below in eqn (10) and (11).

$$X_{(i,j)}(t+1) = Best_j(t) - P_{(i,j)}(t) \times rand\ if\ t \leq \frac{3T}{4}\ and\ t > \frac{T}{2} \quad (10)$$

$$X_{(i,j)}(t+1) = Best_j(t) - \eta_{(i,j)}(t) \times \epsilon - R_{(i,j)}(t) \times rand,\ \ if\ t \leq T\ and\ t > \frac{3T}{4} \quad (11)$$

Where, $X_{(i,j)}$ indicates the $i-th$ individual's produced location in the $j-th$ dimension, $Best_j(t)$ indicates the population's best position at the moment, t is the number of iterations in progress, $\eta_{(i,j)}(t)$ indicates the hunter, $R_{(i,j)}(t)$ symbolizes the reduction function, , T is the most iterations possible, $P_{(i,j)}(t)$ indicates the size of the steps, $rand$ symbolizes the random number, which is usually in the range [0,1]ϵ is the constant term.

Step 4: Termination

After identifying the optimal solution, the process concludes. The flowchart for Hybrid HTO- RSA Algorithm is presented in Figure 2.

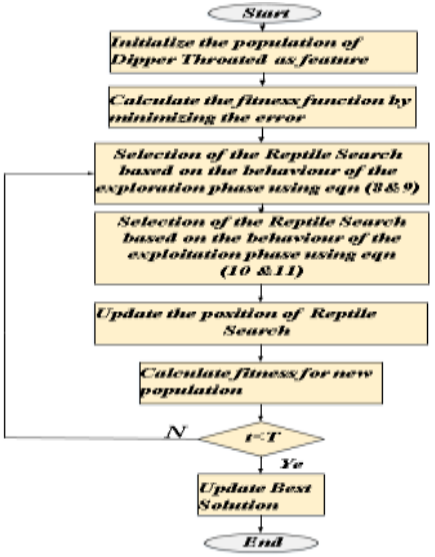

Figure 2: Flowchart for Hybrid HTO-RSA Algorithm

The figure 2 describes an optimization process using the Hybrid HTO-RSA Algorithm. It begins by initializing the population and computing the fitness function, followed by exploration and exploitation phase selection based on specific equations. The positions are updated iteratively until the stopping criterion is met, at which point the best solution is determined.

3.2Classification

In data mining, classification is the process of categorizing data into predefined classes or groups based on a set

of features. For an Intrusion Detection System (IDS) aimed at malware prediction, classification techniques help in distinguishing between malicious and benign activities by analyzing patterns in android malware application activity. The classification phase in the given flow employs a hybrid GAN-BiLSTM model. Bi-directional Long Short- Term Memory (BiLSTM) is used in this model in place of the fully-connected layers of the Generative Adversarial Network (GAN), improving intrusion detection performance and system efficiency. The GAN generates synthetic data to address class imbalance, enhancing the training process. The BiLSTM component leverages temporal dependencies in malware features for accurate malware prediction.

3.2.1 GAN

Generative Adversarial Networks (GANs) for classification the generator and the discriminator working in opposition to classify data by generating synthetic samples and distinguishing from real data [26]. In the context of an Intrusion Detection System (IDS) for malware prediction, GANs can be used to generate synthetic attack data, enhancing the training dataset for the classifier. This helps in addressing data imbalance issues, improving the classifier's ability to identify novel malware attacks. GANs generate realistic attack samples to mitigate data imbalance and improve model generalization. The discriminator refines IDS accuracy by distinguishing between normal and malicious behavior.

In order to trick the discriminator, the generator produces realistic data, while the discriminator improves its capacity to discern between produced and actual data. This interaction is the fundamental idea of GANs. The distribution of actual data is closely mirrored by this adversarial dynamic, which forces the generator to provide better results. The training process is guided by the discriminator's loss function, which measures how well it can distinguish between genuine and bogus data, as seen in eqn (12).

$$L_D = -E_{X_r \sim p_d}[\log D_\phi(X_r)] - E_{Z \sim pz}[\log(1 - D_\phi(G_\theta(Z)))] \quad (12)$$

Where, L_D denotes the loss function of the discriminator, $E_{X_r \sim p_d}$ represents the first term of the probability of the real samples, X_r is the true data distribution, $D_\phi(X_r)$ denotes the probability of the real sample, $G_\theta(Z)$ represents the probability of the fake sample generated by the generator.

The loss function of the generator in a GAN is designed to measure how well the generated data fools the discriminator. It aims to minimize the difference between the discriminator's classification of generated data as fake and real data. By optimizing this loss, the generator continuously improves its ability to create more convincing data samples as shown below in eqn (13)

$$L_G = E_{Z \sim pz}[\log(1 - D_\phi(G_\theta(Z)))] \quad (13)$$

Where, L_G represents the loss generator, $E_{Z \sim pz}$ denotes the sampled from the prior distribution, D_ϕ is the discriminator model, $G_\theta(Z)$ represents the generator model parameterized by θ.

The adversarial training process enhances the detection system's ability to identify even sophisticated malware patterns effectively as shown below in eqn (14)

$$\min_\theta \max_\phi \{L_{gan} = E_{X_r \sim p_d}[log D_\phi(X_r)] + E_{Z \sim p_z}[\log(1 - D_\phi(G_\theta(Z)))]\} \quad (14)$$

Where, L_{gan} denotes the loss function of gan, $E_{X_r \sim p_d}$ represents the expected value of the discriminator output, $D_\phi(X_r)$ is the discriminator predicts the probability of the sample, $G_\theta(Z)$ denotes the generator of the fake sample.

3.2.2 BiLSTM

An RNN type known as a Bidirectional Long Short-Term Memory (BiLSTM) network analyzes data both forward and backward, fully capturing temporal relationships [27]. Their bidirectional processing enables the detection of subtle anomalies often missed by unidirectional models. This makes BiLSTMs ideal for identifying sophisticated malware with high accuracy and minimal false positives.

The LSTM unit determines which information is essential to retain through the forget layer. This layer evaluates the current input x_t and the previous hidden state h_{t-1} to decide the relevance of past data. Using a sigmoid activation function, it outputs a value between 0 and 1 to adjust the previous cell state c_{t-1}. This process helps the LSTM focus on crucial information while discarding irrelevant details as shown below in eqn (15)

$$f_t = \sigma(w_f . x_t + u_f . h_{t-1} + b_f) \quad (15)$$

Where, f_t denotes the forget layer, σ represents the sigmoid function, x_t and h_{t-1} is the output value between 0 and 1, u_f denotes the weighted matrix is applied to the previous hidden state, b_f represents the bias term. The output gate decides how much memory content should be passed to the next hidden state. By evaluating the current input and previous information, it regulates the flow of data. This ensures only the most relevant information is propagated to the subsequent state as shown below in eqn (16)

$$h_t = o_t . \tanh(c_t) \quad (16)$$

Where, h_t represents the hidden state, o_t denotes the output gate, $\tanh$ is the function creates a vector of new candidate value, c_t represents the new cell state.

By adding two distinct networks that analyze data both forward and backward, BiLSTM enhances the LSTM model. The model can incorporate context from both past and future sequences because of this bidirectional nature. The forward hidden state is computed as part of this process to enhance predictive performance as shown below in eqn (17)

$$\overleftarrow{h} = o_t . \tanh(c_t) \quad (17)$$

Where, $\overleftarrow{h}$ denotes the backward hidden state, o_t represents the output gate, c_t is the cell state. As seen below, Algorithm 1 describes the pseudo code for malware prediction.

Algorithm 1: *Pseudo code for malware prediction*

```
Input: Static Dynamic Malware analysis dataset # Perform
pre-processing (SD)
{
UI= RBN (SD) // Normalizes input data to stabilize training and improve model convergence using eqn (2)
HK= SRI (UI)// Fills missing data with estimated values for completeness and reliability using eqn (3)
QW= MSC (HK)// Removes redundant information by clustering data into meaningful subspaces using eqn
(4)
}
# Feature Selection (O)
PI= DTO-RSA (O)// Selects the most relevant features using an optimized search process
{
Calculate Fitness Function F=
min(error)
}
# Classification
AS= Hybrid GAN-BiLSTM (PI) // to predict malware using eqn (14)and (17) If (PI=0)
class 0= Adware Else
(PI=1)
class 1=Backdoor Else if
(PI=2)
class 2=File Infector Else (PI=3)
class 3=No_Category Else
(PI=4)
Class 4=PUA Else
(PI=5)
Class 5= Ransomware
Else (PI=6)
Class 6= Riskware Else (PI=7)
Class 7=Scareware Else (PI=8)
Class 8=Trojan Else (PI=9)
Class 9=Trojan_Banker Else (PI=10)
Class 10=Trojan_Dropper Else (PI=11)
Class 11=Trojan_SMS Else (PI=12)
Class 12= Trojan_Spy Else (PI=13)
Class 13= Zeroday}
Output: Accurately Predict the malware attack. END
```

4. RESULT AND DISCUSSION

Implementing the proposed Hybrid GAN-BiLSTM model into practice for malware prediction. The dataset is made clean, consistent, and ready for additional analysis by preprocessing. Recurrent Batch Normalization standardizes the data for stable model training, Stochastic Regression Imputation handles missing values to ensure data completeness, and Multiview Subspace Clustering removes redundant information to reduce complexity and improve data quality. The Hybrid GAN-BiLSTM model combines synthetic data generation by GAN to address class imbalance and BiLSTM to capture temporal and contextual relationships, ensuring accurate and robust malware prediction. The proposed framework is implemented with Python 3.8 and hardware used in system are Nvidia GeForce GTX 1650, Intel Core i5 CPU, and 16GB of RAM. Hybrid GAN-BiLSTM and DTO-RSA.

Table 1: simulation parameters for Hybrid GAN-BiLSTM and DTO-RSA

Parameters	Algorithms	Values
optimizers	GAN	SGD
learning_rate		0.01
momentum		0.9
epochs		200
batch_size		16
optimizers	BILSTM	Adam
learning_rate		0.01
activation		'softmax'
epochs		50
batch_size		8
lower_bound	DTO-RSA	0
upper_bound		1
pop_size		50
max_iterations		100

In table 1, the simulation parameters for Hybrid GAN-BiLSTM and DTO-RSA including optimizers ,learning_ rate, momentum, epochs, batch_ size, optimizers, learning_ rate, activation, epochs, batch_size, lower_ bound, upper_ bound, pop_ size and max_ iterations.

4.1Dataset Description:

The Static-Dynamic Malware analysis dataset includes 200K labeled android malware apps categorized into families, alongside 200K benign apps sourced to maintain balance. Fourteen malware types are identified, including adware, backdoor, ransomware, riskware, trojans, and zero-day threats [28]. Eight categories sensitive data collection, media, hardware, activities, internet connection, C&C, antivirus, and storage/settings—are used by a thorough taxonomy to group malware families. This classification enhances understanding of malware behaviors and potential risks. The taxonomy aids in identifying specific traits associated with diverse malware categories. It provides insights into malware targeting strategies across varied functional domains. This structured approach supports robust threat detection and prevention. The dynamic-static analysis dataset comprises a total of 1,971 samples, with 1,576 allocated for training and 395 for testing. This division ensures a balanced evaluation of model performance.\

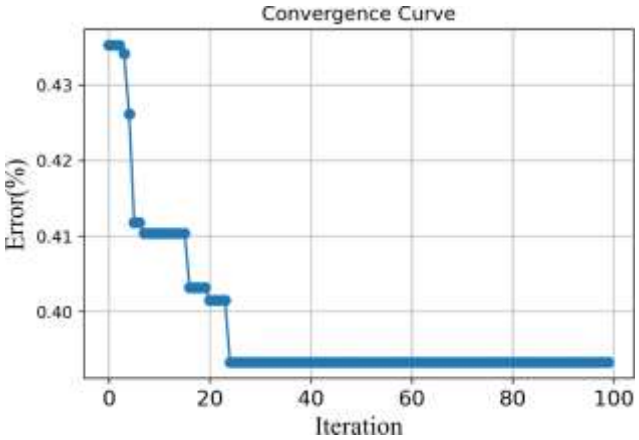

Figure 3: The convergence plot for DTO-RSA

Convergence plot for DTO-RSA is presented in Figure 3 demonstrating how the target result progresses with each iteration. The graph reveals that the error value reaches 0.393276 % by the 23th iteration and remains constant through the 100th iteration.

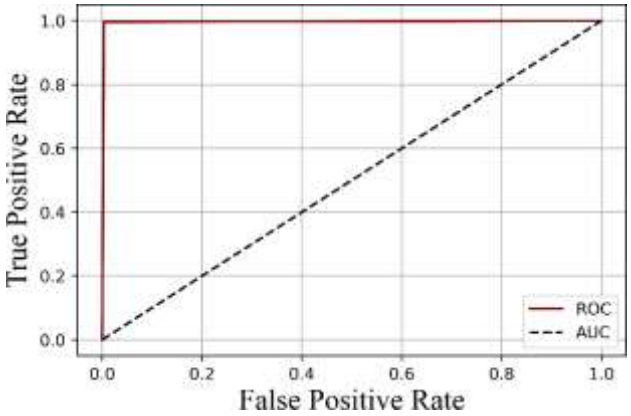

Figure 4: Proposed methods for AUC and ROC plots

The ROC curve of trade-off between a classification model's True Positive Rate (TPR) and False Positive Rate (FPR) is shown in Figure 4. The curve approaching (1, 1) with nearly no false positives indicates near-perfect performance. The dashed diagonal represents a random classifier, showing that the model significantly outperforms chance.

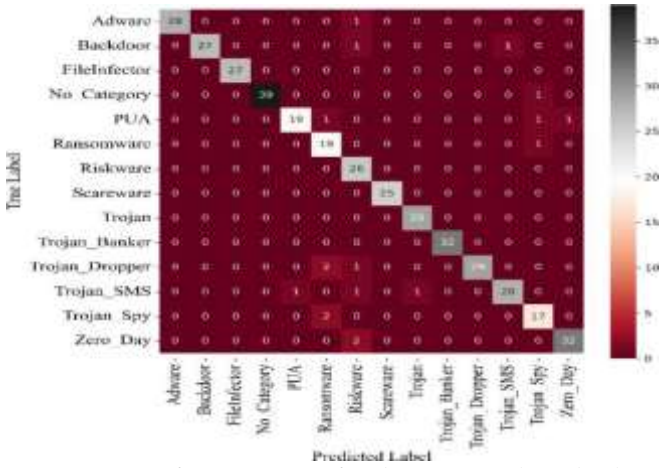

Figure 5: Confusion matrix for the proposed method.

Confusion matrix is displayed in Figure 5 for the proposed method, serving as a key tool for evaluating classification accuracy. This analysis focuses on 14 distinct classes related to malware prediction. Class 0 (Adware) is represented by 28 instances, while Class 1 (Backdoor) has 27 instances. Class 2 (File Infector)

includes 27 instances, whereas Class 3 (No_Category) contains 39 instances. Class 4 (PUA) consists of 19 instances, while Class 5 (Ransomware) also has 19 instances. Class 6 (Riskware) is represented by 26 instances, whereas Class 7 (Scareware) includes 25 instances. Class 8 (Trojan) consists of 29 instances, while Class 9 (Trojan_Banker) has 32 instances. Class 10 (Trojan_Dropper) is represented by 29 instances, whereas Class 11 (Trojan_SMS) contains 28 instances. Class 12 (Trojan_Spy) consists of 17 instances, while Class 13 (Zeroday) includes 32 instances.

4.2Comparison Analysis

The performance of the GAN-BiLSTM model is evaluated against algorithms like Deep Neural Network (DNN) [18], Differential Privacy-Recursive Feature Elimination with Cross Validation-Feed Forward Neural Network (DP-RFECV-FNN) [29, 34], Propose-Residual Neural Network (P-ResNet) [17], and Convolutional Neural Network-Long Short-Term Memory (CNN-LSTM) [30, 33] for malware prediction. The evaluation involves metrics such as accuracy, Positive Predictive Value (PPV), Selectivity, Negative Prediction Value (NPV), Miss Rate, Hit Rate, False Omission Rate (FOR), False Discovery Rate (FDR), Fall out, F1_Score, Error, MK, Phi Coefficient, Kappa, Hamming loss and FM to comprehensively assess each model. This analysis highlights the strengths and limitations of the GAN-BiLSTM model compared to its counterparts. By comparing these parameters, the model's effectiveness in accurately predicting malware is determined. Additionally, the evaluation emphasizes the robustness of GAN-BiLSTM in handling complex patterns. These insights offer a clear understanding of its predictive capabilities.

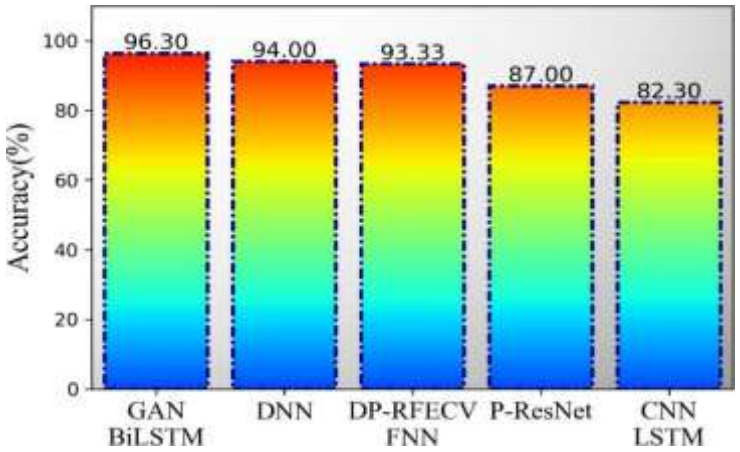

Figure 6: Comparison of the accuracy of the proposed method with other existing algorithms.

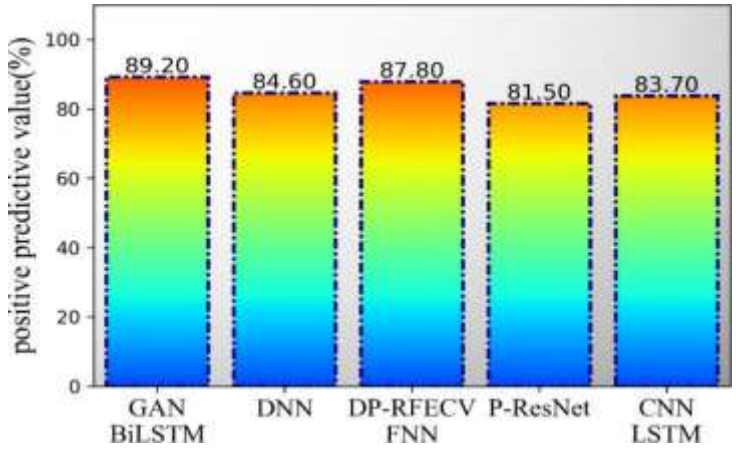

Figure 7: Evaluation of the PPV of GAN-BiLSTM in relation to other existing algorithms.

Comparison of accuracy for proposed GAN-BiLSTM and existing model are depicted in figure 6 and PPV among different classifiers are illustrated in figure 7. The proposed GAN- BiLSTM classifier outperforms the others, achieving an accuracy of 96.30% and a PPV of 89.20%. In contrast, existing methods such as DNN, DP-RFECV-FNN, P-ResNet, and CNN- LSTM show lower accuracies of 94%, 93.33%, 87%, and 82.30%, with corresponding PPV

values of 84.60%, 87.80%, 81.50%, and 83.70%. This highlights the significant performance enhancement offered by the GAN-BiLSTM algorithm. The GAN component generates synthetic samples to enhance the model's ability to detect rare malware instances effectively.

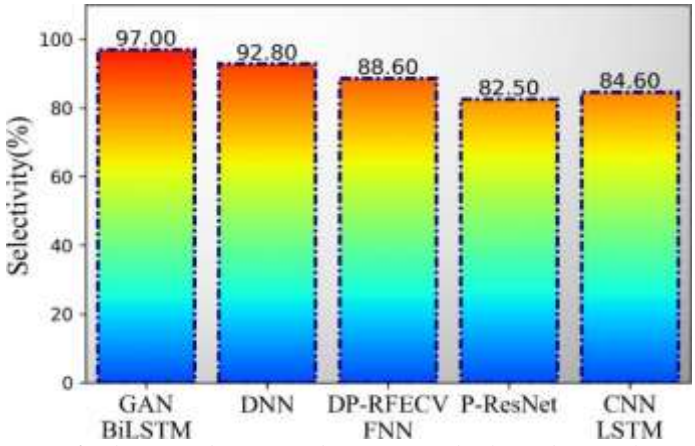

Figure 8: Comparison of Selectivity between the Proposed Algorithm and Existing Algorithms

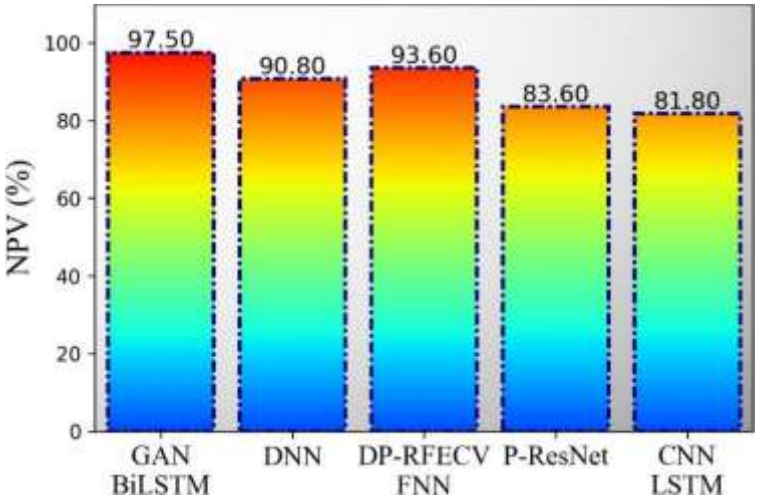

Figure 9: Comparison of the proposed algorithm's NPV with that of other algorithms

A comparison of Selectivity is displayed in figure 8. The proposed GAN-BiLSTM classifier outperformed existing methods, achieving a Selectivity of 97%, compared to 92.80% for DNN, 88.60% for DP-RFECV-FNN, 82.50% for P-ResNet, and 84.60% for CNN-LSTM.

Figure 9 presents a comparison of NPV, where the GAN-BiLSTM classifier achieved 97.50%, surpassing DNN (90.80%), DP-RFECV-FNN (93.60%), P-ResNet (83.60%), and CNN-LSTM (81.80%). Compared to other existing algorithms, GAN-BiLSTM demonstrates better performance the BiLSTM effectively captures long-term dependencies and temporal patterns in sequential data, leading to more accurate malware behavior prediction.\

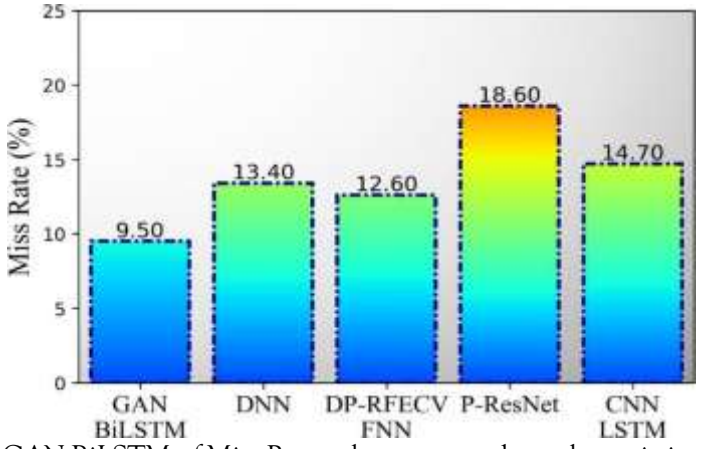

Figure 10: GAN-BiLSTM of Miss Rate value compared to other existing algorithms

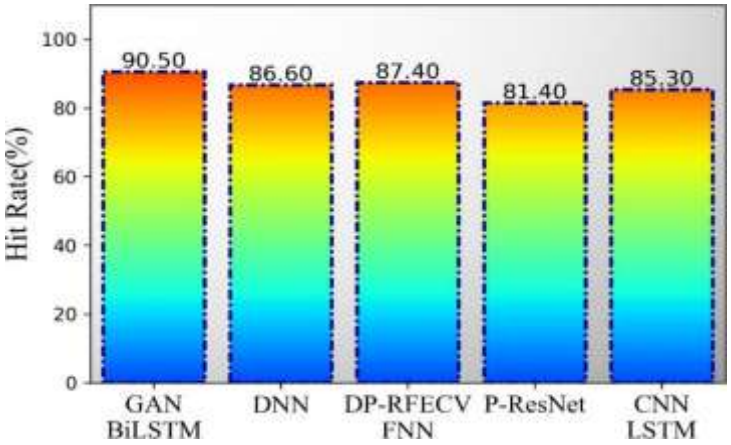

Figure 11: Hit Rate value using alternative existing algorithms for GAN-BiLSTM

Compare Miss Rate and Hit Rate values for proposed GAN-BiLSTM and existing methods in figure 10 and figure 11. The proposed GAN-BiLSTM method achieves a Miss Rate of 9.50% and a Hit Rate of 90.50%. This demonstrates its effectiveness, as it has significantly lower Miss Rate and Higher Hit Rate values compared to existing methods. For instance, algorithms like DNN, DP-RFECV-FNN, P-ResNet, and CNN-LSTM have Miss Rate values of 13.40%, 12.60%, 18.60%, and 14.70%, with Hit Rate values of 86.60%, 87.40%, 81.40%, and 85.30%, respectively. The GAN-BiLSTM approach performs better than these current algorithms in comparison. GANs dynamically adapt to the changing nature of malware patterns, enabling the model to stay effective against evolving threats.

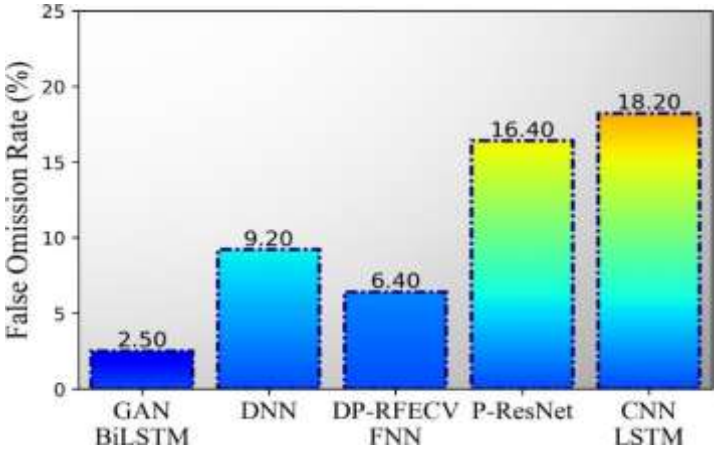

Figure 12: Comparison of the FOR value of GAN-BiLSTM with various existing techniques.

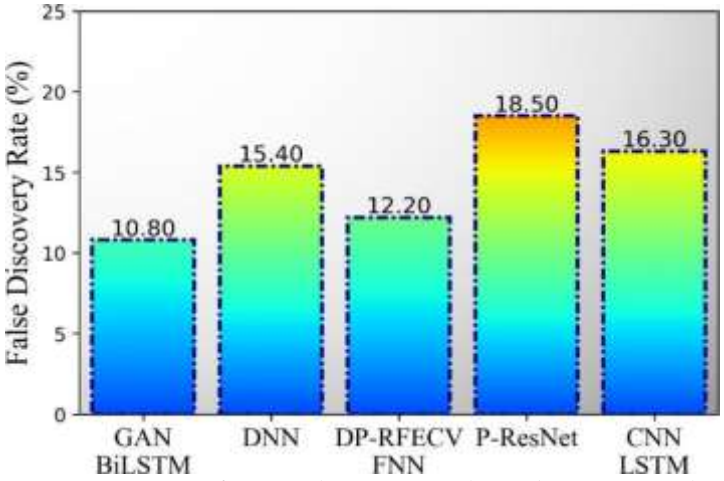

Figure 13: GAN-BiLSTM of FDR value compared to other existing algorithms

The comparison of FOR values is presented in figure 12. Unlike existing methods, the proposed GAN-BiLSTM classifier achieved a lower FOR value of 2.50%, compared to DNN at 9.20%, DP-RFECV-FNN at 6.40%, P-ResNet at 16.40%, and CNN-LSTM at 18.20%.

Figure 13 shows the comparison FDR, with the GAN-BiLSTM classifier scoring a lower value of 10.80%, outperforming DNN at 15.40%, DP-RFECV-FNN at 12.20%, P-ResNet at 18.50%, and CNN-LSTM at 16.30%. Compared to other existing algorithms, the GAN- BiLSTM provides a better performance. The model is well-suited for handling large, diverse datasets, making it effective in real-world intrusion detection tasks.

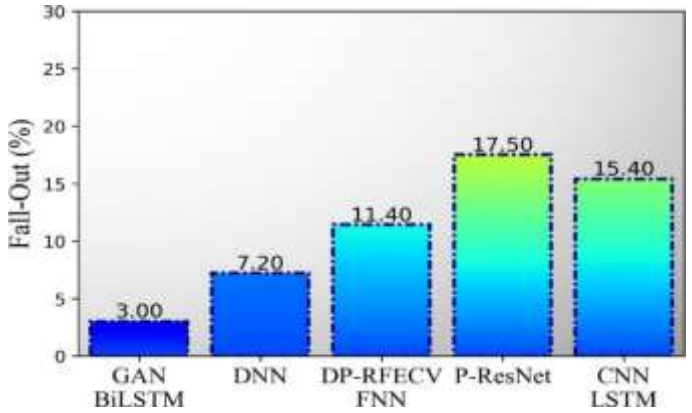

Figure 14: Comparison of Fall-Out for GAN-BiLSTM with various existing algorithms.

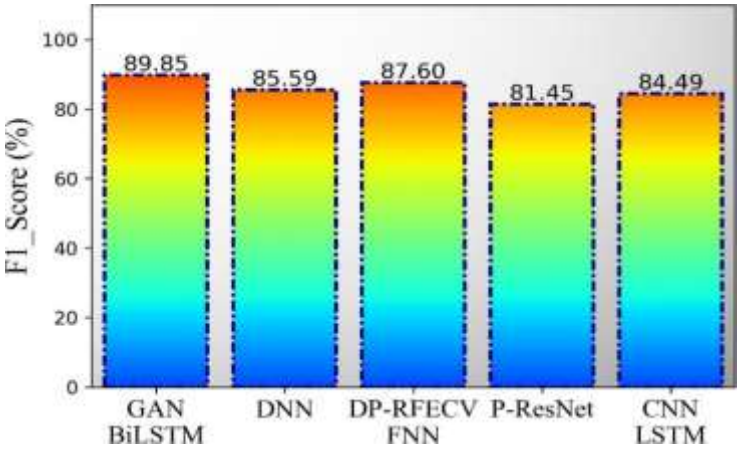

Figure 15: Comparison of the F1_Score value of GAN-BiLSTM with various existing techniques.

Display the comparisons of Fall Out and F1_Score in figure 14 and figure 15. The recommended GAN-BiLSTM approach achieves a Fall Out of 3% and an F1_Score of 89.85%, demonstrating its effectiveness with significantly higher values of fall out and lower values of F1_Score than existing methods. In contrast,

algorithms such as DNN, DP-RFECV- FNN, P-ResNet, and CNN-LSTM have Fall Out values of 7.20%, 11.40%, 17.50%, and 15.40%, and F1_Score VALUES of 85.59%, 87.60%, 81.45%, and 84.49%, respectively. Compared to these existing algorithms, GAN-BiLSTM shows superior performance. Integration of GAN and BiLSTM ensures superior accuracy, leveraging both generative capabilities and sequence modeling.

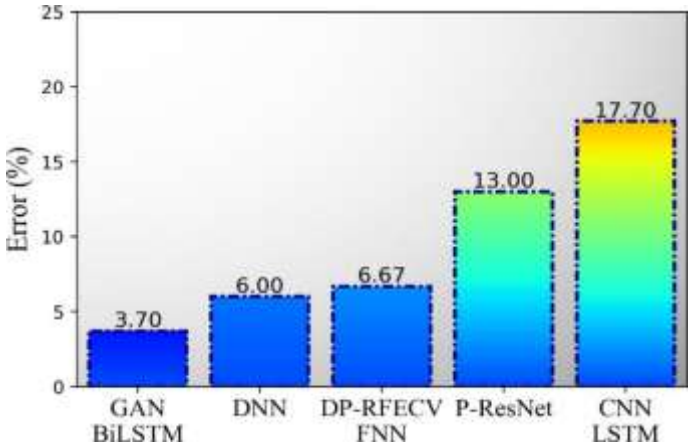

Figure 16: Error evaluation for GAN-BiLSTM with other existing techniques

Compares the Error in figure 16, showing that the proposed GAN-BiLSTM classifier outperformed existing methods with a perfect Error of 3.70%, compared to 6% for DNN, 6.67% for DP-RFECV-FNN, 13% for P-ResNet, and 17.70% for CNN-LSTM. When compared to these current algorithms, GAN-BiLSTM performs better the GAN-BiLSTM architecture can identify subtle anomalies that evasion techniques often exploit, strengthening the detection system.

Table 2: Performance Metrics Comparison for both (GAN-BiLSTM) and existing techniques

Parameter	Methods				
	Proposed (GAN- BiLSTM)	DNN	DP-RFECV- FNN	P-ResNet	CNN-LSTM
MK (%)	94.40	88.60	86.90	74.90	77.60
Phi coefficient (%)	95.10	87.80	91.20	86.30	79.50
Kappa (%)	94.80	89.60	83.40	78.40	77.50
Hamming loss (%)	4.90	12.20	8.80	13.70	20.50
FM (%)	91.80	87.30	81.80	74.80	81.50

Compares the performance attributes of the proposed GAN-BiLSTM with those of the existing methods (DNN, DP-RFECV-FNN, P-ResNet, and CNN-LSTM) in table 2. When compared to alternative methods, the suggested method performs better, achieving greater MK, Phi coefficient, Kappa, and FM as well as reduced error rates (Hamming loss). This demonstrates how successful GAN-BiLSTM is in predicting malware.

5. CONCLUSION

An effective malware detection framework integrates both static and dynamic behaviour of android malware app. Static methods analyze the code's structure and behavior before execution, while dynamic methods monitor its operations during runtime. However, challenges such as maintaining detection speed in real-time, evading detection by modifying malware behavior, and defending against adversarial strategies complicate the process. By improving the model's training and increasing detection precision, a hybrid classifier that combines Generative Adversarial Networks (GANs) and Bi-directional Long Short-Term Memory (BiLSTM) networks tackles these problems. Preprocessing is the initial stage for improving the quality of the raw data for further analysis. Recurrent batch normalization (RBN) is used to normalize the input, while stochastic regression imputation (SRI) is used to handle missing data. Multiview Subspace Clustering (MSC) reduces dimensionality in order to reduce duplication and excessive dimensionality. Following preprocessing, feature selection is the process of determining which properties of the processed data are most pertinent. A hybrid binary optimization

method that combines Reptile Search Algorithm (RSA) and Dipper Throated Optimization (DTO) is introduced for feature extraction. By increasing search efficiency, the RSA guarantees quicker and more accurate convergence. Following their selection, the best features are fed into the GAN-BiLSTM combination model, which is intended for malware detection. This model improves the prediction performance and overall system efficiency by substituting Bi-directional Long Short-Term Memory (BiLSTM) for the fully linked layers of the Generative Adversarial Network (GAN). Using the proposed approach, 96.30% Accuracy, 89.20% PPV, 97.50% NPV, 10.80% FDR, 2.50% FOR, 89.85% F_1 score were obtained. The proposed approach outperforms the existing methods, as shown by a graphical comparison of the results. The results indicate that the GAN-BiLSTM outperforms other existing methods, offering a more effective solution.

5.1Limitation and Future Scope

The limitation lies in the difficulty of balancing the different methodologies, which may hinder the overall system's stability and performance during integration. The approach might also struggle with the balance between accuracy and generalization, especially in dynamic environments. The future scope aims to leverage quantum computing for exponential data processing speeds, enabling real-time decision-making in complex systems. Additionally, integrating bio-inspired AI could enhance adaptability and cognitive capabilities. Self-adaptive AI systems could autonomously refine workflows, ensuring continual optimization and enhanced malware detection. Additionally, the integration of edge intelligence may facilitate real-time analysis and detection in decentralized environments.

REFERENCE

1. Vinayakumar, Ravi, Mamoun Alazab, K. Padannayil Soman, Prabaharan Poornachandran, Ameer Al-Nemrat, and Sitalakshmi Venkatraman., "Deep learning approach for intelligent intrusion detection system" ,Ieee Access, vol.7, pp. 41525- 41550, April 2019, doi:10.1109/ACCESS.2019.2895334.
2. Halbouni, Asmaa, Teddy Surya Gunawan, Mohamed Hadi Habaebi, Murad Halbouni, Mira Kartiwi, and Robiah Ahmad. ,"CNN-LSTM: hybrid deep neural network for network intrusion detection system", IEEE Access,vol.10,pp. 99837-99849,September 2022,doi:10.1109/ACCESS.2022.3206425.
3. Schmitt, Marc. ,"Securing the Digital World: Protecting smart infrastructures and digital industries with Artificial Intelligence (AI)-enabled malware and intrusion detection", Journal of Industrial Information Integration,vol.36,pp. 100520, December 2023,doi:10.1016/j.jii.2023.100520.
4. Guizani, Nadra, and Arif Ghafoor. ,"A network function virtualization system for detecting malware in large IoT based networks" , IEEE Journal on Selected Areas in Communications, vol. 38, no. 6, pp. 1218-1228, April 2020, doi:10.1109/JSAC.2020.2986618.
5. Gad, Abdallah R., Ahmed A. Nashat, and Tamer M. Barkat. ,"Intrusion detection system using machine learning for vehicular ad hoc networks based on ToN-IoT dataset", IEEE Access, vol.9, pp. 142206-142217, October 2021, doi:10.1109/ACCESS.2021.3120626.
6. Kanimozhi, V., and T. Prem Jacob. ,"Artificial Intelligence outflanks all other machine learning classifiers in Network Intrusion Detection System on the realistic cyber dataset CSE-CIC-IDS2018 using cloud computing", ICT Express, vol.7, no.3, pp. 366-370, September 2021, doi:10.1016/j.icte.2020.12.004.
7. Yang, Jin, Tao Li, Gang Liang, Wenbo He, and Yue Zhao. "A simple recurrent unit model based intrusion detection system with DCGAN." IEEE Access, vol. 7, pp. 83286-83296, June 2019, doi:10.1109/ACCESS.2019.2922692.
8. Alem, S., Espes, D., Nana, L., Martin, E. and De Lamotte, F., “A novel bi-anomaly- based intrusion detection system approach for industry 4.0”, Future Generation Computer Systems, vol. 145, pp. 267-283, August 2023, doi:10.1016/j.future.2023.03.024.
9. Gassais, Robin, Naser Ezzati-Jivan, Jose M. Fernandez, Daniel Aloise, and Michel R. Dagenais. ,"Multi-level host-based intrusion detection system for Internet of things", Journal of Cloud Computing, vol. 9, no. 1, pp. 62, November 2020, doi:10.1186/s13677-020-00206-6.
10. Asif, Muhammad, Sagheer Abbas, M. A. Khan, Areej Fatima, Muhammad Adnan Khan, and Sang-Woong Lee. ,"MapReduce based intelligent model for intrusion detection using machine learning technique", Journal of King Saud University- Computer and Information Sciences, vol. 34, no. 10, pp. 9723-9731, November 2022, doi:10.1016/j.jksuci.2021.12.008.
11. Sun, Zhenyang, Gangyi An, Yixuan Yang, and Yasong Liu. ,"Optimized machine learning enabled intrusion detection 2 system for internet of medical things", Franklin Open, vol. 6, pp. 100056, March 2024, doi:10.1016/j.fraope.2023.100056.
12. Kaushik, Baijnath, Reya Sharma, Kulwant Dhama, Akshma Chadha, and Surbhi Sharma. ,"Performance evaluation of learning models for intrusion detection system using feature selection" ,Journal of Computer Virology and Hacking Techniques, vol. 19, no. 4, pp. 529-548, January 2023, doi:10.1007/s11416-022-00460-z.
13. Al-Ambusaidi, Mohammed, Zhang Yinjun, Yar Muhammad, and Abid Yahya. ,"ML- IDS: an efficient ML-enabled intrusion detection system for securing IoT networks and applications", Soft Computing, vol. 28, no. 2, pp. 1765-1784, December 2023, doi:10.1007/s00500-023-09452-7.
14. Muthanna, Mohammed Saleh Ali, Reem Alkanhel, Ammar Muthanna, Ahsan Rafiq, and Wadhah Ahmed Muthanna Abdullah., "Towards SDN-enabled, intelligent intrusion detection system for internet of things (IoT)", IEEE Access, vol. 10, pp. 22756-22768,

February 2022, doi:10.1109/ ACCESS.2022.3153716.

15. Yang, Liyan, Yubo Song, Shang Gao, Aiqun Hu, and Bin Xiao. ,"Griffin: Real-time network intrusion detection system via ensemble of autoencoder in SDN", IEEE Transactions on Network and Service Management, vol. 19, no. 3, pp. 2269-2281, May 2022, doi:10.1109/TNSM.2022.3175710.
16. Mishra, Sushruta, Chandrakanta Mahanty, Shreela Dash, and Brojo Kishore Mishra.,"Implementation of BFS-NB hybrid model in intrusion detection system",In Recent Developments in Machine Learning and Data Analytics: IC3 2018, vol. 740, pp. 167-175, 2019, doi:10.1007/978-981-13-1280-9_17.
17. Mehedi, Sk Tanzir, Adnan Anwar, Ziaur Rahman, Kawsar Ahmed, and Rafiqul Islam. ,"Dependable intrusion detection system for IoT: A deep transfer learning based approach", IEEE Transactions on Industrial Informatics, vol. 19, no. 1, pp. 1006- 1017, April 2022, doi:10.1109/TII.2022.3164770.
18. Shareena, Jishma, Aiswarya Ramdas, and Haripriya AP. ,"Intrusion detection system for iot botnet attacks using deep learning", SN Computer Science, vol. 2, no. 3, pp. 1- 8, April 2021, doi:10.1007/s42979-021-00516-9.
19. Park, Daekyeong, Sangsoo Kim, Hyukjin Kwon, Dongil Shin, and Dongkyoo Shin. ,"Host-based intrusion detection model using siamese network", IEEE Access, vol. 9, pp. 76614-76623, May 2021, doi:10.1109/ACCESS.2021.3082160.
20. Smmarwar, Santosh K., Govind P. Gupta, Sanjay Kumar, and Prabhat Kumar. "An optimized and efficient android malware detection framework for future sustainable computing." Sustainable Energy Technologies and Assessments, vol. 54, pp. 102852, December 2022, doi:10.1016/j.seta.2022.102852.
21. Fekri, Mohammad Navid, Harsh Patel, Katarina Grolinger, and Vinay Sharma. ,"Deep learning for load forecasting with smart meter data: Online Adaptive Recurrent Neural Network", Applied Energy, vol. 282, pp. 116177, January 2021, doi:10.1016/j.apenergy.2020.116177.
22. Amitha, Puranik, V. S. Binu, and Biju Seena., "Estimation of missing values in aggregate level spatial data" ,Clinical Epidemiology and Global Health, vol. 9, pp. 304-309, March 2021, doi:10.1016/j.cegh.2020.10.003.
23. Chen, Man-Sheng, Ling Huang, Chang-Dong Wang, Dong Huang, and S. Yu Philip. "Multiview subspace clustering with grouping effect." IEEE Transactions on Cybernetics, vol. 52, no. 8, pp. 7655-7668, December 2020, doi:10.1109/TCYB.2020.3035043.
24. Alhussan, Amel Ali, Doaa Sami Khafaga, El-Sayed M. El-Kenawy, Abdelhameed Ibrahim, Marwa Metwally Eid, and Abdelaziz A. Abdelhamid., "Pothole and plain road classification using adaptive mutation dipper throated optimization and transfer learning for self-driving cars", IEEE Access, vol. 10, pp. 84188-84211, August 2022, doi:10.1109/ACCESS.2022.3196660.
25. Zhou, Liping, Xu Liu, Ruiqing Tian, Wuqi Wang, and Guowei Jin. ,"A multi-strategy enhanced reptile search algorithm for global optimization and engineering optimization design problems", Cluster Computing, vol. 28, no. 2, pp. 1-41, December 2024, doi: 10.1007/s10586-024-04819-3.
26. Su, Qichen, Haza Nuzly Abdull Hamed, Mohd Adham Isa, Xue Hao, and Xin Dai., "A GAN-based data augmentation method for imbalanced multi-class skin lesion classification", IEEE Access, vol. 12, pp. 16498 - 16513, January 2024, doi:10.1109/ACCESS.2024.3360215.
27. Zrira, Nabila, Assia Kamal-Idrissi, Rahma Farssi, and Haris Ahmad Khan., "Time series prediction of sea surface temperature based on BiLSTM model with attention mechanism", Journal of Sea Research, vol. 198, pp. 102472, April 2024, doi:10.1016/j.seares.2024.102472.
28. Alberto Zorzetto (2022)Kaggle:[https://www.kaggle.com/datasets/albertozorzetto/cic- andmal-2020-dynamic-static-analysis?select=Static+Analysis.tar.] Accessed on: 29- 01-2025].
29. Nawshin, Faria, Devrim Unal, Mohammad Hammoudeh, and Ponnuthurai N. Suganthan. ,"AI-powered malware detection with Differential Privacy for zero trust security in Internet of Things networks", Ad Hoc Networks, vol. 161, no. 1, pp. 103523, August 2024, doi:10.1016/j.adhoc.2024.103523.
30. Jain, Shreeya, Pranav M. Pawar, and Raja Muthalagu., "Hybrid intelligent intrusion detection system for internet of things." Telematics and Informatics Reports", vol. 8, pp. 100030, December 2022,doi:10.1016/j.teler.2022.100030.
31. Saif, Sohail, Priya Das, Suparna Biswas, Manju Khari, and Vimal Shanmuganathan. "HIIDS: Hybrid intelligent intrusion detection system empowered with machine learning and metaheuristic algorithms for application in IoT based healthcare." Microprocessors and Microsystems (2022): 104622.
32. Bhati, Nitesh Singh, Manju Khari, Vicente García-Díaz, and Elena Verdú. "A review on intrusion detection systems and techniques." International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems 28, no. Supp02 (2020): 65-91.
33. Bhati, Nitesh Singh, and Manju Khari. "An ensemble model for network intrusion detection using adaboost, random forest and logistic regression." Applications of Artificial Intelligence and Machine Learning: Select Proceedings of ICAAAIML 2021. Singapore: Springer Nature Singapore, 2022. 777-789.
34. Kumar, Kapil, and Manju Khari. "Efficient Intrusion Detection and Trust Management in Federated Learning Systems." 2025 2nd International Conference on Computational Intelligence, Communication Technology and Networking (CICTN). IEEE, 2025.