Computers, Materials & Continua

Tech Science Press

DOI: 10.32604/cmc.2024.059999

ARTICLE

5DGWO-GAN: A Novel Five-Dimensional Gray Wolf Optimizer for Generative Adversarial Network-Enabled Intrusion Detection in IoT Systems

Sarvenaz Sadat Khatami[1], Mehrdad Shoeibi[2], Anita Ershadi Oskouei[3], Diego Martín[4,*] and Maral Keramat Dashliboroun[5]

[1]Department of Data Science Engineering, University of Houston, Houston, TX 77204, USA

[2]The WPI Business School, Worcester Polytechnic Institute, Worcester, MA 01609, USA

[3]School of Systems and Enterprises, Stevens Institute of Technology, Hoboken, NJ 07030, USA

[4]Department of Computer Science, Escuela de Ingeniería Informática de Segovia, Universidad de Valladolid, Segovia, 40005, Spain

[5]Ernest G. Welch School of Art & Design, Georgia State University, Atlanta, GA 30303, USA

*Corresponding Author: Diego Martín. Email: diego.martin.andres@uva.es

Received: 21 October 2024 Accepted: 05 December 2024 Published: 03 January 2025

ABSTRACT

The Internet of Things (IoT) is integral to modern infrastructure, enabling connectivity among a wide range of devices from home automation to industrial control systems. With the exponential increase in data generated by these interconnected devices, robust anomaly detection mechanisms are essential. Anomaly detection in this dynamic environment necessitates methods that can accurately distinguish between normal and anomalous behavior by learning intricate patterns. This paper presents a novel approach utilizing generative adversarial networks (GANs) for anomaly detection in IoT systems. However, optimizing GANs involves tuning hyper-parameters such as learning rate, batch size, and optimization algorithms, which can be challenging due to the non-convex nature of GAN loss functions. To address this, we propose a five-dimensional Gray wolf optimizer (5DGWO) to optimize GAN hyper-parameters. The 5DGWO introduces two new types of wolves: gamma (γ) for improved exploitation and convergence, and theta (θ) for enhanced exploration and escaping local minima. The proposed system framework comprises four key stages: 1) preprocessing, 2) generative model training, 3) autoencoder (AE) training, and 4) predictive model training. The generative models are utilized to assist the AE training, and the final predictive models (including convolutional neural network (CNN), deep belief network (DBN), recurrent neural network (RNN), random forest (RF), and extreme gradient boosting (XGBoost)) are trained using the generated data and AE-encoded features. We evaluated the system on three benchmark datasets: NSL-KDD, UNSW-NB15, and IoT-23. Experiments conducted on diverse IoT datasets show that our method outperforms existing anomaly detection strategies and significantly reduces false positives. The 5DGWO-GAN-CNN$_{\text{AE}}$ exhibits superior performance in various metrics, including accuracy, recall, precision, root mean square error (RMSE), and convergence trend. The proposed 5DGWO-GAN-CNN$_{\text{AE}}$ achieved the lowest RMSE values across the NSL-KDD, UNSW-NB15, and IoT-23 datasets, with values of 0.24, 1.10, and 0.09, respectively. Additionally, it attained the highest accuracy, ranging from 94% to 100%. These results suggest a promising direction for future IoT security frameworks, offering a scalable and efficient solution to safeguard against evolving cyber threats.

KEYWORDS
Internet of things; intrusion detection; generative adversarial networks; five-dimensional binary gray wolf optimizer; deep learning

1 Introduction

The rapid advancement of the Internet of Things (IoT) has revolutionized numerous sectors by connecting billions of devices that communicate vast amounts of data across distributed and diverse networks. These data are generated by a wide range of sources, including sensors, home automation devices, and industrial equipment, which interact continuously to provide valuable insights and automation [1,2]. However, as the number and diversity of connected devices grow, so does the complexity of IoT networks and, consequently, the potential for security vulnerabilities. The sheer scale and heterogeneity of IoT systems make them a prime target for cyber threats, as each connected device represents a possible entry point for malicious actors. In recent years, cyberattacks have not only increased in frequency but have also become more sophisticated, with attackers developing complex strategies that exploit the dynamic and often decentralized nature of IoT networks. This trend has elevated network security to a critical concern, particularly in IoT-enabled environments where real-time data integrity is paramount.

Within specific IoT applications, such as smart grids [3–5], vehicle-to-grid networks [6–8], smart homes, and healthcare systems, the consequences of security breaches can be severe, impacting not only individual privacy but also broader societal infrastructure and safety. For instance, a security breach in a smart grid could disrupt power distribution, while vulnerabilities in healthcare IoT could expose sensitive patient data or jeopardize life-critical medical devices. Thus, developing effective defense mechanisms is not merely a technical challenge but a necessity for safeguarding critical services. As IoT technology permeates diverse areas, from industrial automation to urban infrastructure, the need for reliable, adaptive, and scalable security frameworks becomes increasingly pressing. Addressing these challenges requires advanced solutions that can quickly detect and respond to abnormal network behaviors, ensuring the security and resilience of IoT applications against the ever-evolving landscape of cyber threats.

Detecting anomalies within IoT networks has emerged as one of the fundamental challenges in securing these complex environments [9–11]. Conventional approaches to intrusion detection have often relied on machine learning (ML) models, such as decision trees (DT) and support vector machines (SVM), which classify network traffic based on predefined rules or statistical patterns [12–14]. While effective in static and well-defined settings, these methods often struggle with the unique demands of IoT networks, which are characterized by dynamic traffic patterns, high variability, and an evolving threat landscape. The adaptability of traditional ML models is limited, and their ability to detect new, unknown types of cyber threats is often insufficient, as they rely on historical data and predefined attack signatures. As a result, these methods may produce high false-positive rates or fail to identify subtle, complex attack patterns within IoT ecosystems.

In recent years, the focus has shifted toward leveraging deep learning (DL) techniques, which have demonstrated superior capabilities in handling large-scale, high-dimensional data and detecting intricate patterns in real-time [15]. Among these, generative adversarial networks (GANs) and deep neural networks (DNNs) have garnered significant interest due to their ability to learn complex data distributions and adapt to the dynamic conditions of IoT networks. GANs, in particular, offer a powerful framework for anomaly detection by generating synthetic data that can help models learn rare or previously unseen attack patterns, effectively enhancing their sensitivity to novel threats. DNNs, with their layered architectures, are well-suited for capturing hierarchical features in network traffic data, allowing them to recognize both obvious and subtle anomalies. By employing these advanced DL models, researchers aim to address the limitations of traditional intrusion detection systems, creating robust and scalable solutions that can adapt to the continuously changing nature of IoT environments. These innovative approaches hold the potential to provide a more reliable defense against cyber threats, ensuring that IoT networks remain secure as they continue to expand and evolve.

1.1 Problem Statement and Motivation

A major challenge in deploying these models in practice is the optimization of their hyperparameters, which significantly contribute to achieving optimal performance. In complex models like GANs, hyperparameters such as learning rate, batch size, and network depth must be carefully tuned. However, gradient-based optimization methods, which are commonly used, often struggle with non-convex problems like GAN training. These methods tend to get trapped in local minima, leading to suboptimal performance and slow convergence. Additionally, determining the correct configuration of hyperparameters is computationally intensive and sensitive to initialization, especially in large-scale IoT networks where real-time response is critical. Poorly optimized GANs can result in ineffective anomaly detection, increasing the risk of undetected threats in IoT environments.

To tackle these issues, this paper introduces an innovative approach to intrusion detection in IoT networks, leveraging the five-dimensional gray wolf optimizer (5DGWO). Unlike traditional gradient-based techniques, the 5DGWO algorithm enhances both exploration and exploitation, helping the model avoid local minima and achieve faster convergence. This optimization approach is particularly beneficial for GANs in dynamic IoT environments, where a balance between accuracy and computational efficiency is essential for timely and effective anomaly detection. The proposed 5DGWO-GAN framework thus addresses critical limitations in current models, offering a more robust, efficient, and scalable solution for safeguarding IoT networks against increasingly sophisticated cyber threats. The 5DGWO algorithm introduces two new wolf types, γ and θ, to enhance exploration and exploitation capabilities. These additional wolves help the GAN escape local minima and converge faster, improving intrusion detection accuracy and reducing false positives. To evaluate our system, we experimented with three network flow data sets considering different scenarios: 1) NSL-KDD [16,17]; 2) UNSW-NB15 [18]; 3) IoT data set [19]; The proposed system framework comprises four key stages: 1) preprocessing, 2) generative model training, 3) autoencoder (AE) training, and 4) predictive model training. The generative models are utilized to assist the AE training, and the final predictive models (including convolutional neural network (CNN), deep belief network (DBN), recurrent neural network (RNN), random forest (RF), and extreme gradient boosting (XGBoost)) are trained using the generated data and AE-encoded features.

1.2 Contributions and Organization of Paper

According to the mentioned gaps, the main contributions of this paper are summarized as follows:

- We propose 5DGWO-GAN, a novel intrusion detection approach for IoT systems based on a developed GAN model. Traditional GANs, while effective at capturing complex data patterns, face significant challenges in hyper-parameter optimization due to the non-convex nature of their loss functions. To address this issue, we introduce 5DGWO, a novel optimization algorithm that significantly improves the tuning of critical hyper-parameters such as learning rate, batch size, and network depth. The 5DGWO method is specifically designed to overcome the limitations of gradient-based optimization techniques, which often struggle with getting trapped in local minima or require excessive computational resources.

- A key innovation of the 5DGWO algorithm is the inclusion of two new types of wolves, γ and θ, which greatly enhance the exploration and exploitation phases of the optimization process. The γ wolves, representing elite members of the pack, focus on refining the search around promising regions of the solution space, accelerating convergence towards optimal solutions. Meanwhile, the random θ wolves are designed to increase the diversity of the search space, enabling the model to escape local minima by exploring less conventional areas. This dual mechanism significantly improves the overall performance of the GAN in terms of convergence speed and model accuracy.
- We utilize generative models to assist in the AE training, and the predictive models are trained using both the generated data and the encoded features from the AE. As part of our contribution, we proposed hybrid algorithms, including DBN_{AE}, CNN_{AE}, 5DGWO-GAN-RNN, 5DGWO-GAN-DBN_{AE}, and 5DGWO-GAN-CNN_{AE}, to enhance the overall detection performance in IoT systems.
- In this paper, we tested the system on three benchmark datasets: NSL-KDD, UNSW-NB15, and IoT-23. The results show that the proposed method achieves superior performance across key metrics such as accuracy, recall, precision, root mean square error (RMSE), and convergence trend, reducing false positives and improving overall detection accuracy.

The remainder of this paper is organized as follows: In Section 2, we review related works in the field of intrusion detection, focusing on traditional ML models and more recent DL approaches, particularly the use of GANs in this domain. Section 3 provides background information on GANs and the datasets used in this study. Section 4 describes the proposed intrusion detection framework based on 5DGWO combined with GANs. Section 5 presents the experimental results, and finally, Section 6 concludes the paper.

2 Related Works

The domain of network intrusion detection systems (NIDS) has received considerable attention, particularly as IoT networks grow, expanding the attack surface for cyber threats. Early studies in NADS primarily relied on traditional ML techniques such as DTs [12], SVMs [13], and RF [20]. These models focused on identifying known attack patterns by analyzing the attributes of network traffic. However, their performance declined when handling more complex and sophisticated threats in IoT environments, which are dynamic and diverse. One major limitation of these traditional ML models is their inability to handle data imbalance effectively, with the vast majority of traffic being normal and only a limited amount showing signs of malicious behavior. DL approaches have emerged as a more robust solution to intrusion detection in NADS, leveraging the power of neural networks to better capture complex, nonlinear relationships within network data [14]. Techniques such as CNNs [21], long short-term memory networks (LSTMs) [22], and DBNs [23] have shown significant promise.

Ingre et al. [24] proposed a NIDS using a multilayer perceptron and achieved accuracies of 81% and 79.9% for binary and multi-class classifications, respectively, on the NSL-KDD dataset. Similarly, Gao et al. [25] proposed a semi-supervised learning method for NIDS, utilizing fuzzy and ensemble learning, reporting an accuracy of 84.54% on the NSL-KDD dataset. Leveraging a DBN model, Alrawashdeh et al. [26] developed an anomaly-based NIDS that showed superior classification performance when tested on subsets of the original dataset. Focusing on a software-defined networking (SDN) environment, Tang et al. [27] presented a DNN model for anomaly detection, demonstrating that it outperformed traditional models such as Naïve Bayes, SVM, and DT. Imamverdiyev et al. [28] introduced a restricted Boltzmann machine (RBM)-based NIDS, showing that the Gaussian-Bernoulli RBM variant surpassed other RBM-based models, such as the Bernoulli-Bernoulli RBM and DBN.

From the perspective of utilizing both behavioral characteristics and content-based features, Zhong et al. [29] integrated big data with a tree architecture-driven DL system. They combined shallow learning and DL techniques to enhance the system's effectiveness in detecting subtle intrusion patterns. Haghighat et al. [30] introduced an ensemble-based NIDS using DL models and voting mechanisms, demonstrating a significant reduction in false alarms (up to 75%) compared to traditional DL approaches. Additionally, for real-time detection in industrial IoT settings, Yang et al. [31] developed a tree-structured anomaly detection system incorporating window-sliding and model-updating techniques within a locality-sensitive hashing-based iForest model [32,33], enabling effective handling of infinite data streams. In a related effort, Qi et al. [34] introduced a NIDS for multi-aspect data streams by combining locality-sensitive hashing, isolation forests, and principal component analysis (PCA), effectively identifying group anomalies and processing data rows faster than previous models.

When working with time-series data, several researchers have explored the potential of recurrent neural networks (RNNs). Kim et al. [35] designed an LSTM-based NIDS model and demonstrated its efficiency. Similarly, Yin et al. [36] proposed an RNN-based NIDS that achieved accuracies of 83.3% for binary and 81.3% for multi-class classification. Xu et al. [37] developed a NIDS based on RNNs and found that the gated recurrent unit (GRU) was more suitable for intrusion detection tasks than the LSTM unit. In a study focused on SCADA networks, Gao et al. [38] introduced an omni-intrusion detection system by combining LSTM with a feed-forward neural network, showcasing effective intrusion detection regardless of temporal correlations. Their Omni-IDS outperformed previous DL-based approaches through tests conducted on a SCADA testbed.

Beyond supervised learning, unsupervised techniques, particularly autoencoders (AE), have gained traction in anomaly detection. Javaid et al. [39] introduced a sparse AE-based NIDS, achieving 79.1% accuracy for multi-class classification on the NSL-KDD dataset. Similarly, Yan et al. [40] applied a sparse AE to extract high-level feature representations, demonstrating the efficiency of a stacked sparse AE model as a feature extraction technique. Shone et al. [41] introduced a NIDS based on a stacked non-symmetric deep AE, reporting 85.42% accuracy in multi-classification. Ieracitano et al. [42] also contributed to this area by comparing AE and LSTM-based NIDS models, demonstrating that AE-based systems achieved superior performance with accuracies of 84.21% and 87% for binary and multi-class tasks, respectively. Generative models have also been explored to enhance NIDS performance [15].

Early works focused on applying basic GANs, which rely on the Jensen–Shannon divergence or Kullback–Leibler divergence [43–45]. Since then, various GAN models have been developed for specific use cases. Li et al. [46] and Lee et al. [47] utilized Wasserstein GANs to generate synthetic data, while Dlamini et al. [48] introduced a conditional GAN for improving classification accuracy in minority classes. In more specific industrial environments, Li et al. [49] and Alabugin et al. [50] introduced LSTM-GAN and bidirectional GAN-based anomaly detection models, respectively, showing effective performance on the secure water treatment (SWaT) dataset. Siniosoglou et al. [51] developed an anomaly detection system capable of detecting anomalies and classifying attack types, embedding an AE into the GAN architecture by deploying the encoder as a discriminator and the decoder as a generator, proving its efficiency in smart grid settings. The recent work by Park et al. [52] adopts a reconstruction error-based GAN to generate more realistic synthetic data. Specifically, they utilize the boundary equilibrium GAN (BEGAN) model, which combines AE principles with Wasserstein distance to minimize reconstruction error. By incorporating AE models into the detection process, they extract more meaningful features and extend model adaptability, ultimately demonstrating that their framework surpasses existing ML-based NIDS solutions.

Javadpour et al. [53] proposed a novel distributed multi-agent intrusion detection and prevention system (DMAIDPS). The DMAIDPS framework utilizes learning agents that conduct a six-step detection process to classify network activity as either normal or indicative of an attack. Their system was evaluated using the KDD Cup 99 and NSL-KDD datasets. Results demonstrated that the DMAIDPS achieved notable improvements, with average increases of 16.81% in recall, 16.05% in accuracy, and 18.12% in F-score, indicating enhanced detection capabilities. Sangaiah et al. [54] explored the use of IDSs as a critical defense strategy in cloud computing environments, where security challenges are prevalent. Their study emphasized the importance of feature selection in distinguishing between malicious and legitimate activities, as it significantly impacts IDS performance. To enhance classification accuracy, they proposed a hybrid ant-bee colony optimization (HABCO) method, which reframes feature selection as an optimization problem. The effectiveness of HABCO was tested against several methods, including ANNIDS, IDSML, HCRNNIDS, DLIDS, GAPSAIDS, BHSVM, and SVMTHIDS. Results indicated that HABCO achieved superior accuracy compared to these existing approaches, demonstrating its potential for improved intrusion detection. Altamimi et al. [55] Al-Haija proposed an IDS framework designed to enhance the security of IoT communication networks against cyber threats. Their approach incorporates an extreme learning machine (ELM) to improve IDS performance by reducing false positives and providing flexible attack pattern construction. Using two established IoT network datasets their study evaluates the ELM's effectiveness in a supervised learning context. The authors focus on the ELM algorithm's ability to process high-dimensional and imbalanced data, demonstrating its potential to increase IDS accuracy and operational efficiency. Their experiments, conducted on Python and Google Colab, included binary and multi-class classification for both datasets. Results indicate that the ELM-based IDS outperformed other conventional supervised learning models and advanced IDS frameworks in this domain.

3 Backgrounds

In order to develop a robust intrusion detection system, it is essential to understand the underlying technologies and datasets that support such a framework. This section provides an overview of AEs, GANs, and GWO, which form the core of our proposed intrusion detection system and introduces the benchmark datasets used in our evaluation.

3.1 AE Model

AE are essential models in DL and are trained through unsupervised learning. Their primary goal is to reconstruct the input data as accurately as possible. During training, the model parameters are continually adjusted to reduce the reconstruction error. Typically, an AE is composed of two parts: 1) an encoder and 2) a decoder. The encoder is responsible for transforming the raw input data x into a latent representation z, as represented by Eq. (1):

$$z = f(xW + b) \tag{1}$$

where f is the encoder's activation function, W is the weight matrix, and b represents the bias vector. The decoder, on the other hand, attempts to reconstruct z back into an approximation of the original input $\tilde{x}$, described as Eq. (2):

$$\tilde{x} = g(zW' + b') \tag{2}$$

where g is the decoder's activation function, and W' and b' represent the weight matrix and bias vector for the decoder. The AE works by minimizing the reconstruction error L_{RE}, which can be defined as Eq. (3):

$$L_{RE}(x, \tilde{x}; W, W') = \left|x - \tilde{x}\right|_2^2 = |x - g(W' \cdot f(xW + b) + b')| \tag{3}$$

A key feature of AEs is their ability to compress high-dimensional input data into lower-dimensional, meaningful representations. In this study, we employed AEs for feature extraction and dimensionality reduction. While PCA is traditionally used to reduce high-dimensional data, AEs offer the advantage of performing nonlinear transformations on complex datasets. Although this section outlines the basic structure of AEs, these models can be extended with multiple layers and asymmetric architectures.

3.2 Standard GAN Model

Generative models aim to approximate the probability distribution of a given training dataset and strive to produce synthetic data that resembles the real data (i.e., training data). Recently, GANs have emerged as a prominent topic within generative models, leading to numerous studies. Several GAN models have been introduced to enhance both performance and functionality [56]. A GAN consists of two key components: 1) a generator G and 2) a discriminator$D$. The generator G is responsible for generating synthetic data (fake data) that closely mimics real data, while the discriminator D works to distinguish between real and fake data. Thus, these two models have competing goals during training. Formally, let p_z represent the probability distribution of the latent code, and p_{data} be the probability distribution of real data. The objective function $V(D, G)$ of a GAN, which incorporates both a generator G and a discriminator D, is structured as a minimax game and can be described by the Eq. (4):

$$V(D, G) = \underset{G}{min}\,\underset{D}{max}\,\mathbb{E}_{X\sim p_{data}}[log(D_{\theta D}(X))] + \mathbb{E}_{Z\sim p_Z}[log(1 - D_{\theta D}(G_{\theta G}(Z)))] \tag{4}$$

where θ_D and θ_G refer to the parameters of the discriminator D and the generator G, respectively. The discriminator is trained to provide higher confidence scores for real data, whereas the generator learns to produce synthetic data that maximizes the confidence score from the discriminator. Over multiple iterations, the generator and discriminator eventually reach a point of balance, known as a Nash equilibrium, where no further improvements can be made.

Since the original introduction of GANs, various extensions have been developed, refining the objective functions or modifying the architecture. One such variation is the BEGAN model [57], which builds on the principles of AEs and focuses on minimizing reconstruction errors. Unlike traditional GANs, which define their objective function based on the distance between confidence vectors of real and synthetic data, BEGAN's objective is based on the Wasserstein distance between reconstruction error distributions, as shown Eq. (5):

$$\begin{cases} L_D = L(x;\theta_D) - k_t \cdot L\left(G\left(z;\theta_G\right);\theta_D\right) \\ L_G = L\left(G\left(z;\theta_G\right);\theta_D\right) \\ k_{t+1} = k_t + \lambda_k \cdot \left(\gamma \cdot L\left(x;\theta_D\right) - L\left(G\left(z;\theta_G\right);\theta_D\right)\right) \end{cases} \tag{5}$$

Here, the hyper-parameter $\gamma \in [0,1]$ represents the diversity ratio, and λ_k serves as the learning rate for k. The function $L(\cdot)$ denotes the reconstruction error of the autoencoder, and t indicates the iteration step.

3.3 Standard GWO

GWO is an algorithm inspired by the hunting and social hierarchy of grey wolves. It is used to tackle optimization problems by mimicking the structured way in which wolves hunt and interact within their pack. The algorithm assigns potential solutions to a group of wolves, where the positions of wolves represent different solutions. The wolves are classified into four categories based on their hierarchy: alpha (α), beta (β), delta (δ), and omega (ω). Each of these categories plays a unique role in guiding the exploration of the search space, with the alpha wolf typically leading the hunt for optimal solutions while beta and delta wolves assist in decision-making. The GWO algorithm begins with an initialization phase, where the positions of the alpha, beta, and delta wolves are set randomly within the search space. These initial positions symbolize potential solutions to the problem. During the search phase, the other wolves adjust their positions relative to the alpha, beta, and delta wolves. This adjustment follows the pack's natural hierarchy and aims to balance exploration, which involves searching new regions of the solution space, and exploitation, which focuses on refining promising areas. In the social dynamics of grey wolves, the omega wolf plays a significant role despite being the lowest in the hierarchy. The omega is submissive and typically follows the commands of higher-ranking wolves, but its presence is crucial for maintaining pack stability and reducing internal conflicts. This mirrors how the GWO algorithm functions, where weaker solutions still contribute to the overall optimization process by supporting the pack and preventing disarray in the search process.

The GWO's update process is critical for improving the solutions generated by the pack. The positions of the alpha, beta, and delta wolves are updated based on their performance and relative positions to other wolves in the group. This update mechanism allows the algorithm to continuously refine the solutions, drawing inspiration from the wolves' coordinated hunting behavior, where they strategically encircle their prey. This behavior is mathematically modeled in the GWO algorithm, allowing it to find optimal or near-optimal solutions to complex problems effectively. Eqs. (6)–(9) are introduced to mathematically represent this encircling behavior.

$$\vec{D} = \vec{C}\vec{X}_p(t) - \vec{X}(t) \tag{6}$$

$$\vec{X}(t+1) = \vec{C}\vec{X}_p(t) - \vec{A}.\vec{D} \tag{7}$$

$$\vec{A} = 2\vec{a}\vec{r}_1 - \vec{a} \tag{8}$$

$$\vec{C} = 2\vec{r}_2 \tag{9}$$

where $\vec{X}$ is the position vector of a gray wolf, $\vec{X}_p$ is hunting position vector, $\vec{A}$ and $\vec{C}$ are coefficient vectors, $\vec{r}_1$ and $\vec{r}_2$ are random vectors in the interval [0, 1], the $\vec{a}$ vector is linearly reduced from 2 to 0 during the repetition.

When mathematically modeling the hunting strategies of grey wolves, it is assumed that the alpha, along with the beta and delta wolves, have superior knowledge of the prey's likely location. As a result, the algorithm retains the three best solutions found and requires the remaining search agents, including the omega wolves, to update their positions based on the top-performing wolves. This approach is reflected in Eqs. (10)–(12), which are designed to simulate this aspect of the process.

$$\vec{D}_\alpha = \left|\vec{C}_1\vec{X}_\alpha - \vec{X}\right| \quad \vec{D}_\beta = \left|\vec{C}_2\vec{X}_\beta - \vec{X}\right|, \quad \vec{D}_\delta = \left|\vec{C}_3\vec{X}_\delta - \vec{X}\right| \tag{10}$$

$$\vec{X}_1 = \vec{X}_\alpha - \vec{A}_1.(\vec{D}_\alpha) \quad \vec{X}_2 = \vec{X}_\beta - \vec{A}_2.(\vec{D}_\beta), \quad \vec{X}_3 = \vec{X}_\delta - \vec{A}_3.(\vec{D}_\delta) \tag{11}$$

$$\vec{X}(t+1) = \frac{\vec{X}_1 + \vec{X}_2 + \vec{X}_3}{3} \tag{12}$$

Fig. 1 illustrates how a search agent's position is updated in a 2D search space, influenced by the positions of the alpha, beta, and delta wolves. The new position is randomly selected within a circular region, the size of which is determined by the locations of the alpha, beta, and delta wolves. These leading wolves estimate the prey's position, causing the other wolves to periodically adjust their positions within the surrounding area of the prey.

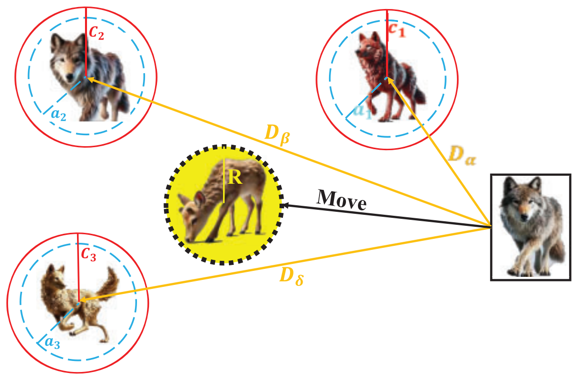


Figure 1: Position update in the standard GWO algorithm

3.4 Dataset Description

In this research, we utilize three well-established network traffic datasets that are frequently used as benchmarks for NIDS evaluations.

1. **NSL-KDD Dataset:** This dataset is an enhanced version of the KDDcup99 dataset [16,17]. It contains two distinct sets: KDDTrain for training, with 125,973 records, and KDDTest for testing, with 22,544 records. Each record is composed of 41 features, which include three nominal, six binary, and 32 numerical attributes that describe various aspects of network traffic. The dataset labels traffic as either normal or belonging to one of four major attack categories: Denial of Service (DoS), Probing, Remote-to-Local (R2L), and User-to-Root (U2R). DoS attacks aim to flood system resources with excess traffic, Probing seeks to gather information about the target system, R2L involves unauthorized remote access to local systems, and U2R attempts to gain root-level access by exploiting system vulnerabilities [16,17].
2. **UNSW-NB15 Dataset:** The UNSW-NB15 dataset [18] was created using the IXIA Perfect-Storm tool and serves as another widely adopted benchmark for intrusion detection system assessments. It comprises 175,341 records for training and 82,332 for testing, with each record characterized by 43 attributes that describe various network flow features. In addition to a binary indicator showing if the traffic is normal or malicious, there is a categorical

label specifying nine different types of attacks: Fuzzers, Analysis, Backdoors, DoS, Exploits, Generic, Reconnaissance, Shellcode, and Worms. These attack types reflect a variety of malicious activities, ranging from fuzzing techniques to identify vulnerabilities, to backdoor access attempts [20].

3. **IoT-23 Dataset:** To assess performance in IoT-specific environments, the IoT-23 dataset [19] was employed. This dataset consists of network traffic collected from real-world IoT scenarios, including 20 sub-datasets that document malicious activity and three sub-datasets representing benign traffic. For this study, we focused on data related to the Mirai botnet (CTU-IoT-Malware-Capture-34-1), which contains 23,145 network flows categorized into four types: Benign, Command & Control (C&C), DDoS, and PortScan. While the benign data captures normal IoT network activity, the other categories encompass various forms of malicious behavior, such as C&C communications and DDoS attacks [19].

4 The Proposed 5DGWO-GAN-Based NIDS

The 5DGWO-GAN framework is designed to improve anomaly detection in IoT systems by combining the strengths of GANs for data generation with the optimization capabilities of the 5DGWO. This framework is composed of four primary stages, each contributing to the system's goal of accurate and efficient intrusion detection. Here's a detailed breakdown of each stage as visualized in Fig. 2:

1. **Preprocessing:** The process begins with preparing the raw IoT data for model training. Datasets like NSL-KDD, UNSW-NB15, and IoT-23 are used, each presenting diverse attack types and network behaviors. This stage involves data cleaning, normalization, and transformation, which are crucial for eliminating inconsistencies and noise in IoT network traffic data. By standardizing features and formatting data appropriately, this step ensures that both generative and predictive models can accurately learn from and distinguish between normal and anomalous patterns. Given the IoT datasets' class imbalance (where certain types of attacks are underrepresented) preprocessing also focuses on balancing data distribution to make it suitable for GAN training and subsequent feature extraction.
2. **Generative Model Training:** In this stage, the GAN model is trained on the preprocessed data to generate synthetic samples that mimic the real data distribution. This synthetic data is particularly valuable for handling class imbalance, as it allows the model to generate samples of underrepresented attack types. Here, the GAN consists of two networks that compete in a minimax game. The GAN framework operates through an interaction between the generator (G) and the discriminator (D) to produce synthetic data that resembles real data. The process begins with real data from the original dataset, which is fed directly into the discriminator for training. The generator, fine-tuned by the 5DGWO optimization, creates synthetic data samples that aim to mimic this real data distribution as closely as possible. These synthetic samples, along with the actual data, are then presented to the discriminator, whose task is to classify each input as either "Real" or "Fake." As the generator produces increasingly realistic samples in its effort to deceive the discriminator, the discriminator simultaneously becomes better at distinguishing genuine data from synthetic data. This adversarial interaction pushes the generator to refine its output, learning to generate highly realistic data that aligns with the patterns in the original dataset. By employing the 5DGWO optimization, key GAN hyper-parameters (such as learning rate, batch size, and network depth) are fine-tuned, resulting in faster convergence and greater model stability. Through this iterative process, the GAN achieves a balanced data generation that effectively addresses class imbalances, producing samples that are representative of both normal and underrepresented attack classes.

3. **AE Model Training:** The AE, consisting of an encoder (E) and a decoder (D), learns to extract core features from both real and synthetic data produced by the 5DGWO-GAN. The AE model's encoder component captures critical features from both real and synthetic data generated by the 5DGWO-GAN, enabling efficient dimensionality reduction and feature extraction. Generative models assist in the training of AEs by supplementing the training data with diverse, high-quality synthetic samples, enabling the AE to achieve a higher level of accuracy and reliability in anomaly detection. By training on a dataset enriched with synthetic samples, the AE learns to represent both normal and attack data patterns effectively, capturing the nuanced differences that characterize each class. These encoded features are then used as inputs for the predictive models, contributing to a more informative feature space that aids in accurate anomaly detection. The AE's role is fundamental to improving the overall system's ability to generalize across different types of network traffic.
4. **Predictive Model Training:** In the final stage, a diverse set of predictive models (including CNN, DBN, RNN, RF, and XGBoost) are trained using both the AE-encoded features and the GAN-generated synthetic data. Each model leverages the feature-rich input space created by the AE, while the synthetic data helps address class imbalance, ensuring that even minority classes are well-represented. This ensemble approach allows the system to capitalize on each model's strengths, achieving robust performance across multiple metrics. Additionally, we proposed hybrid models, such as 5DGWO-GAN-CNN$_{AE}$, 5DGWO-GAN-DBN$_{AE}$, and 5DGWO-GAN-RNN, which combine the advantages of GAN-generated data, AE-encoded features, and the respective model architectures to further enhance detection accuracy and reduce false positives.

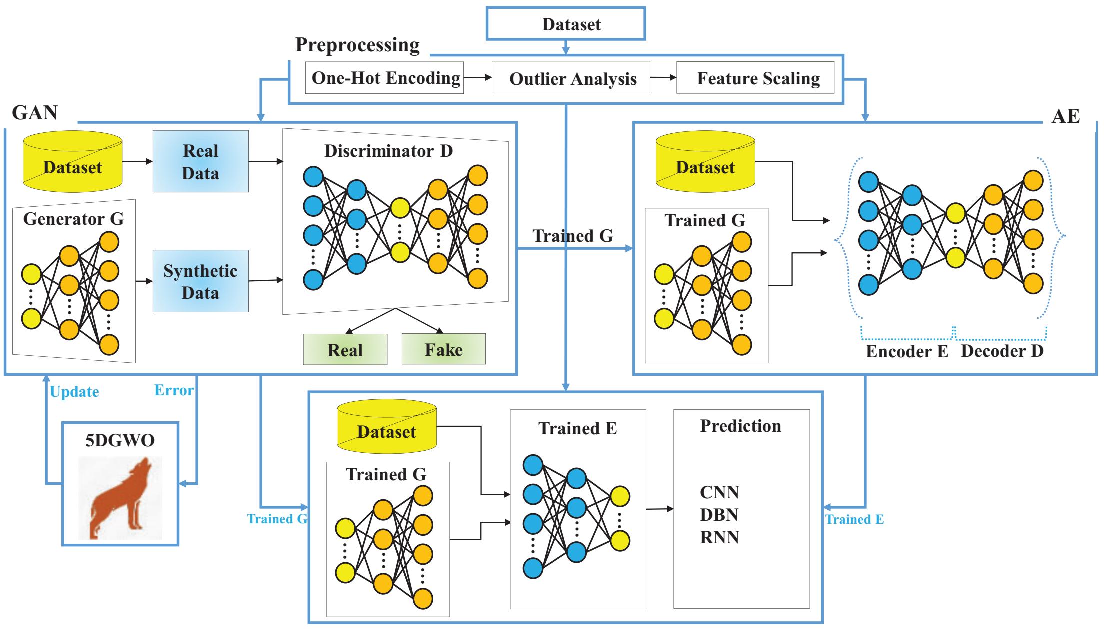


Figure 2: The proposed 5DGWO-GAN-based NIDS architecture consists of four main components: preprocessing, GAN training, AE training, and predictive models. The GAN generates synthetic data, fine-tuned by 5DGWO to improve accuracy and stability. The AE, with its encoder-decoder structure, uses GAN-generated data for feature extraction, facilitating efficient learning. Finally, predictive models classify intrusions based on AE-refined features, boosting detection accuracy and reducing false positives

4.1 Proposed 5DGWO-GAN

In this paper, we present an innovative enhancement to the standard GWO, referred to as the 5DGWO. Our enhancement introduces two new types of wolves, denoted as gamma (γ) and theta (θ), to strengthen the algorithm's overall search intelligence. Gamma wolves, also called elites, focus on improving the exploitation capability of the algorithm, which enhances its ability to converge on optimal solutions more quickly. On the other hand, theta wolves, also referred to as random wolves, are designed to bolster exploration, ensuring that the algorithm can escape local minima and explore a wider range of potential solutions in the search space. The introduction of these two additional types of wolves provides a more dynamic balance between exploration and exploitation, making the 5DGWO more robust in finding global optimal solutions and avoiding premature convergence. The pyramid structure shown in Fig. 3 highlights the new hierarchical order of the 5DGWO, where the newly added γ and θ wolves occupy ranks between alpha and beta wolves in the traditional GWO. The alpha wolves still hold the primary leadership role, directing the overall search, but now they are supported by gamma wolves, which help refine the search for better solutions by focusing more intensely on exploitation. Theta wolves, placed just after delta wolves, assist in the global exploration by adding randomness to their movement, allowing the algorithm to probe new areas of the solution space. This pyramid illustrates the hierarchical positioning of the wolves, showcasing the enhanced leadership roles in this new structure.

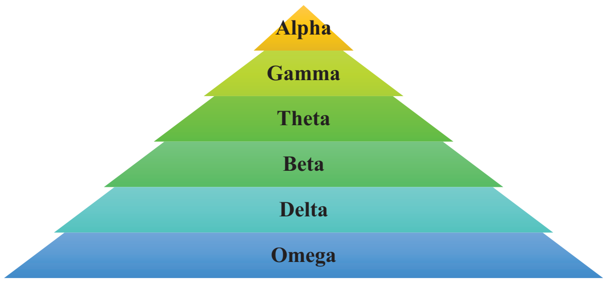


Figure 3: The hierarchy of gray wolfs in the proposed 5DGWO

In Eqs. (13)–(15), the new update formulas for the proposed 5DGWO algorithm are presented, which incorporate the gamma and theta wolves into the standard GWO equations. Eq. (13) shows how the distances to $\alpha, \beta, \delta, \gamma$, and θ wolves are calculated, using the current position of the prey and their relative positions. Eq. (14) details how the positions of the search agents (wolves) are updated based on the alpha, beta, delta, gamma, and theta wolves. The additional gamma and theta wolves add layers of refinement and randomness to the movement of the wolves, enhancing both exploitation and exploration processes. Finally, Eq. (15) shows the overall position update rule, where the position of each wolf at time $t+1$ is calculated as the average of the updated positions of all the wolves, ensuring a balanced and well-distributed movement across the search space.

$$\vec{D}_\alpha = \left|\vec{C}_1\vec{X}_\alpha - \vec{X}\right|, \quad \vec{D}_\beta = \left|\vec{C}_2\vec{X}_\beta - \vec{X}\right|, \quad \vec{D}_\delta = \left|\vec{C}_3\vec{X}_\delta - \vec{X}\right| \quad \vec{D}_\gamma = \left|\vec{C}_4\vec{X}_\gamma - \vec{X}\right|, \quad \vec{D}_\theta = \left|\vec{C}_5\vec{X}_\theta - \vec{X}\right| \tag{13}$$

$$\vec{X}_1 = \vec{X}_\alpha - \vec{A}_1.\left(\vec{D}_\alpha\right), \quad \vec{X}_2 = \vec{X}_\beta - \vec{A}_2.\left(\vec{D}_\beta\right), \quad \vec{X}_3 = \vec{X}_\delta - \vec{A}_3.\left(\vec{D}_\delta\right) \quad \vec{X}_4 = \vec{X}_\gamma - \vec{A}_4.\left(\vec{D}_\gamma\right),$$
$$\vec{X}_5 = \vec{X}_\theta - \vec{A}_5.\left(\vec{D}_\theta\right) \tag{14}$$

$$\vec{X}(t+1) = \frac{\vec{X}_1 + \vec{X}_2 + \vec{X}_3 + \vec{X}_4 + \vec{X}_5}{5} \tag{15}$$

Fig. 4 visualizes how the positions of wolves are updated in the search space with the introduction of γ and θ wolves. Each wolf moves in response to the position of the prey, indicated by the central point (R), but the wolves now interact with γ and θ wolves to determine their new positions. The new wolves improve the algorithm's strategic flexibility by allowing more diverse movement patterns. Gamma wolves (C_4) and theta wolves (C_5) are shown adjusting their positions to assist in both refining the search (exploitation) and expanding the search boundaries (exploration), ensuring that the wolves do not get trapped in local minima and continue seeking the global optimum. The introduction of gamma wolves, or elite wolves with superior fitness function values, significantly enhances the exploitation capability of the 5DGWO algorithm. Exploitation refers to the algorithm's ability to fine-tune and intensify its search in regions of the search space where optimal or near-optimal solutions are likely to be found. By prioritizing the guidance of wolves with higher fitness values, the algorithm becomes more focused on refining promising areas of the solution space. The gamma wolves, being elites with enhanced knowledge of the solution landscape, help the pack zero in on more optimal solutions, thus accelerating convergence. Their role is particularly crucial during the later stages of the optimization process when the algorithm needs to exploit known good areas and make incremental improvements. This selective guidance by elite wolves ensures that the pack does not waste resources on less promising regions, thereby improving the efficiency and effectiveness of the exploitation process.

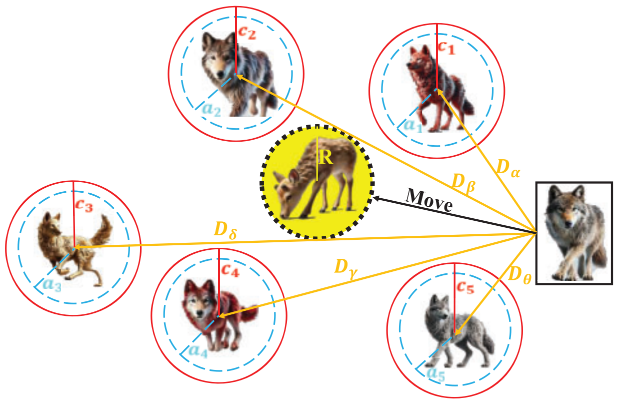
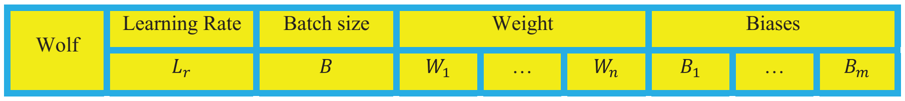


Figure 4: The position update in the proposed 5DGWO algorithm. Here, different types of wolves collaborate to locate the optimal solution, represented by the target prey (R). Each wolf's position is adjusted based on its distance from the target, allowing them to converge towards the prey. This configuration emphasizes the exploration and exploitation capabilities of the 5DGWO, where the additional wolf types, γ and θ, enhance the search diversity and adaptability, helping to escape local minima and improving convergence efficiency

Theta wolves, or random wolves, play a key role in strengthening the exploration capabilities of the 5DGWO algorithm. Exploration refers to the ability of the algorithm to search across diverse areas of the solution space, ensuring that it does not get trapped in local optima. By introducing a degree of randomness into their movement, theta wolves allow the algorithm to break free from repetitive or confined search patterns. These wolves encourage the algorithm to explore less visited regions of the search space, helping to discover new potential solutions that may have been overlooked by the more deterministic search agents. This added exploration flexibility is critical, especially in complex or multi-modal optimization problems, as it helps maintain a balance between exploitation and exploration, ultimately preventing premature convergence and ensuring a more comprehensive search of the solution landscape. By integrating these two new types of wolves, the 5DGWO becomes more adept at finding optimal solutions in complex optimization problems, offering a more intelligent and adaptable search mechanism compared to the standard GWO. This improvement is crucial for enhancing convergence speed while maintaining the algorithm's ability to explore a diverse set of potential solutions.

In the proposed 5DGWO-GAN framework, the 5DGWO algorithm is utilized to optimize the hyper-parameters of the GAN architecture. Fig. 5 illustrates the structure of a wolf in the 5DGWO-GAN algorithm. Each wolf represents a candidate solution in the optimization process, encoded with a specific set of hyper-parameters. The structure includes components for learning rate (L_r), batch size (B), weights (W_n), and biases (B_m), which collectively define the configuration of a GAN model. By adjusting these parameters, each wolf iteratively moves toward the optimal hyper-parameter combination that minimizes the objective function. The objective function of the proposed 5DGWO-GAN algorithm, as defined in Eq. (5), guides the wolves toward the global optimum. The 5DGWO enhances the optimization process by introducing two additional wolf types (gamma and theta) which contribute to a more intelligent exploration-exploitation balance, leading to improved convergence and stability during GAN training. By emphasizing exploitation, gamma wolves ensure that the optimal weights, biases, and learning rates are fine-tuned to accelerate the convergence of the GAN training process. This is particularly beneficial in DL, where well-optimized hyper-parameters can significantly enhance model performance and reduce training time. On the other hand, theta wolves enhance the exploration aspect of the optimization. These wolves introduce randomness into the search process, allowing the algorithm to explore less examined regions of the hyper-parameter space. This prevents the GAN from becoming trapped in local minima, a common issue in DL models. Theta wolves help ensure that the algorithm explores a wide range of potential hyper-parameter configurations, which is essential for achieving a global optimal solution, especially in complex GAN architectures like the one proposed for intrusion detection in IoT systems.



Wolf	Learning Rate	Batch size	Weight			Biases		
	L_r	B	W_1	…	W_n	B_1	…	B_m

Figure 5: The structure of a wolf in 5DGWO-GAN

Fig. 6 illustrates an example of updating positions in the 5DGWO algorithm. Each row represents a wolf (agent) with a specific role each assigned unique hyper-parameter configurations, including learning rate, batch size, and other parameters. The "New Wolf" displays updated parameters, aiming to reduce the RMSE, which serves as the evaluation metric for model performance. Alpha, with the lowest RMSE of 12.59, currently represents the best-performing solution, while omega, with the highest RMSE of 32.74, is the least effective. This structure demonstrates how 5DGWO optimizes model performance by iteratively adjusting the positions (parameters) of each wolf to minimize RMSE, allowing the algorithm to converge towards an optimal set of parameters for better accuracy and stability. The new wolf's RMSE of 4.32 suggests a substantial improvement, validating the effectiveness of this update strategy in refining model performance.

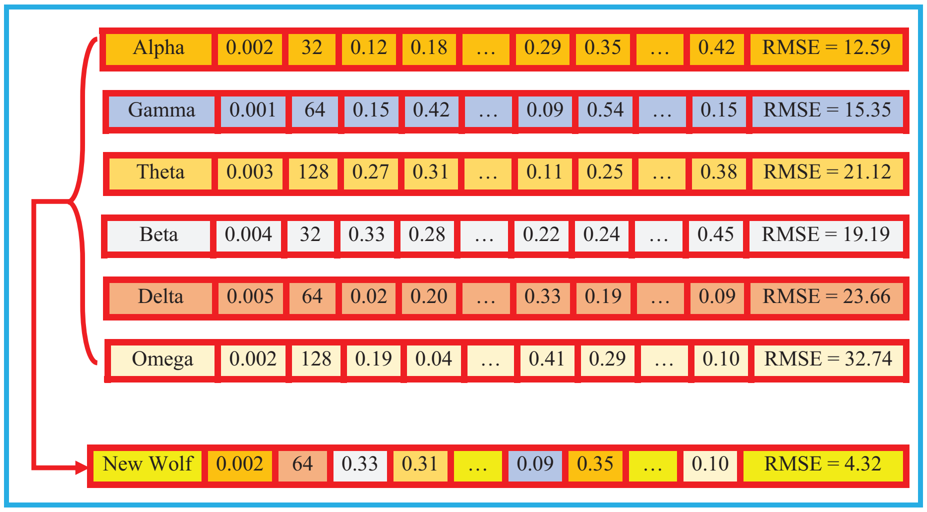


Figure 6: An example of position updates in the 5DGWO algorithm (for training GAN). This figure presents the structure and parameter configurations of different wolves in the 5DGWO algorithm, each with a unique set of hyper-parameters such as learning rate, batch size, weights, and biases

The 5DGWO algorithm's ability to optimize hyper-parameters such as the learning rate, batch size, weight initialization, and bias adjustments allows for the automatic tuning of the GAN's training process. This leads to a more robust architecture that is better suited for handling the intricacies of IoT data in network intrusion detection tasks. Despite the potential for increased computational costs due to the iterative nature of the optimization, the benefits of using 5DGWO to optimize GAN parameters (such as improved convergence speed and model stability) outweigh the drawbacks. The proposed 5DGWO-GAN system thus demonstrates how nature-inspired algorithms can be effectively applied to optimize DL architectures in real-world applications.

4.2 Preprocessing

Before the DL models are constructed and trained, the raw dataset is refined through a preprocessing module that involves three key steps: 1) outlier analysis, 2) one-hot encoding, and 3) feature scaling. The outlier analysis step removes anomalous data points that could hinder model training. Outliers are typically identified by examining the statistical distribution of data using robust scaling methods. Common techniques for detecting outliers include the interquartile range (IQR) and median absolute deviation (MAD), with this study opting for MAD. For a numeric attribute $A = \{x_1, x_2, \ldots, x_n\}$, the MAD is calculated as Eq. (16):

$$f_A(x_i) = \tilde{x}_i = \frac{x_i - min\, x_j}{max\, x_j - min\, x_j} \tag{16}$$

where (x_i) represents the i-th attribute value within (A). While many DL approaches incorporate feature extraction techniques (such as PCA or Pearson correlation) at this stage to enrich the model with informative features, we do not adopt such computational processes here. This is because our framework integrates an AE that serves the role of feature extraction. In fact, the use of computational feature extraction yielded minimal performance improvements compared to models without it. Detailed information on the AE's role as a feature extractor is provided later.

4.3 Synthetic Data Generation Using the 5DGWO-GAN Model

The synthetic data generation component involves building and training generative models using the refined dataset from the preprocessing phase. For this purpose, we employ a state-of-the-art GAN model called 5DGWO-GAN, which relies on AEs and an objective function based on reconstruction error. Before training 5DGWO-GAN, the system partitions the dataset according to its class labels and then builds separate generative models for each class. After training, each model is responsible for producing synthetic data that corresponds to a particular class. One critical aspect to consider when applying GANs to NIDS is determining the stopping criteria for training, as it directly affects the quality of the synthetic data used to train the detection model. Given that GANs are based on a zero-sum game, monitoring convergence during training is challenging. In conventional GANs, synthetic data quality is often monitored through visual inspection, but this is impractical for NIDS due to the non-visual nature of the data. Fortunately, 5DGWO-GAN simplifies this process by allowing the approximation of training convergence through equilibrium tracking. The measure of convergence (M) in 5DGWO-GAN is defined as Eq. (17):

$$M = L(x) + |\gamma L(x) - L(G(z))| \tag{17}$$

where ($L(\cdot)$) denotes the reconstruction error function, and (γ) is the diversity ratio. Once the convergence measure (M) falls below a predefined threshold, the generative model's training process is terminated. In our experiments, we set this threshold to 0.058. After training, the system generates class-specific synthetic data using the trained generator, which is then merged with the original dataset for subsequent stages.

4.4 Training the AE and Detection Model

The next step involves training the AE, which provides feature extraction and dimensionality reduction. In our framework, the AE shares the same architecture as the 5DGWO-GAN discriminator. Since the 5DGWO-GAN discriminator functions as an AE, it seamlessly integrates into our detection model, handling the same data format. After training the AE, the system uses the trained encoder as the feature extractor. The trained encoder is then used in the detection models as a feature extractor, with its parameters fixed during the training of the detection models. For classification, we implemented DBN, CNN, and LSTM models. The CNN model, traditionally used for image analysis, was adapted with one-dimensional convolutional layers to handle network traffic data directly. Given the LSTM's ability to analyze temporal features, we chose not to combine it with the AE, as this could obscure time-related data. For each model, the output layer was structured with binary fields for anomaly detection tasks and multi-class fields when identifying specific attack types.

5 Experimental Results

In this section, the performance of the algorithms RF, XGBoost, RNN, DBN, CNN, DBN_{AE}, CNN_{AE}, 5DGWO-GAN-RNN, 5DGWO-GAN-DBN_{AE}, and 5DGWO-GAN-CNN_{AE} is evaluated on the NSL-KDD, UNSW-NB15, and IoT-23 datasets within a simulated environment. The experiments are designed to benchmark the anomaly detection capabilities of these models in realistic IoT environments, offering insights into their performance under diverse conditions. In this paper, we employed four key metrics to assess the performance of the proposed models: RMSE, accuracy, precision, and recall. Accuracy measures the proportion of correct predictions and is widely used to evaluate the effectiveness of AI models. Precision, in the context of a particular class, is defined as the percentage of correct positive predictions relative to all positive predictions made by the model.

Recall, however, represents the ratio of true positive instances that the model successfully detected. The formulas for these metrics are as Eqs. (18)–(21):

$$Accuracy = \frac{TP + TN}{TP + FN + FP + TN} \tag{18}$$

$$Precision = \frac{TP}{TP + FP} \tag{19}$$

$$Recall = \frac{TP}{TP + FN} \tag{20}$$

$$RMSE = \left(\frac{1}{N}\sum_{i=1}^{N}[P_i - O_i]^2\right)^{\frac{1}{2}} \tag{21}$$

where N is the number of observations; P_i is the calculated parameter; O_i is the observed parameter; TN = true negative; TP = true positive; FN = false negative; and FP = false positive. We used these metrics to assess the performance of each model on the experimental datasets. Although the models were constructed with a stable architecture, some variability in results persisted. To address this, we trained each model independently 20 times and reported the outcomes for the model that achieved the highest detection rate on the test dataset.

5.1 Tuning Algorithms

In the implementation of our proposed 5DGWO-GAN framework, we meticulously calibrated the hyper-parameters of each algorithm to optimize performance for intrusion detection in IoT systems. This calibration process was carried out through an extensive trial-and-error optimization approach, in which each parameter was adjusted iteratively, observing its impact on model performance in terms of accuracy, convergence stability, and computational efficiency. The final calibration parameters are summarized in Table 1, with the initial parameter ranges specified based on prior research and refined through systematic experimentation to identify the best-performing values. In our trial-and-error calibration approach, we tested various values for each parameter individually to identify the optimal settings for our 5DGWO-GAN framework. During this process, we kept all other parameters constant while varying only the target parameter, allowing us to isolate its impact on the model's performance. For each parameter, we evaluated its effect based on key performance metrics, such as accuracy for classification tasks or RMSE for regression-oriented components. By systematically testing a range of values for each parameter, we could determine which setting produced the best results according to these metrics.

Table 1: Parameter setting of proposed models through the trial and error method

Method	Parameter	Value
GAN	Learning rate	0.002
	Number of neurons in hidden layers	80
	Batch size	64
	Momentum term	0.05
	Activation function	ReLU

(Continued)

Table 1 (continued)

Method	Parameter	Value
	Convergence threshold	0.059
	Epochs	300
	Optimizer	5DGWO
5DGWO	C	0.7
	A	0.3
	α	[0, 2]
	Population size	110
	Iteration	300
AE	Learning rate	0.001
	Hidden layer sizes	60
	Batch size	128
	Activation function	ReLU
DBN	Number of hidden layers	8
	Number of neurons in hidden layers	44
	Learning rate	0.06
	Activation	Tanh and sigmoid
	Optimizer	SGD
RF	Number of estimators	100
	Maximum depth of trees	12
	Minimum samples per split	6
RNN	Number of hidden layers	12
	Number of neurons in hidden layers	36
	Learning rate	0.04
	Dropout rate	0.2
	Activation	Tanh and sigmoid
	Optimizer	SGD
CNN	Number of convolution layers	4
	Kernel size	3×3
	Pooling type	Max pooling (2×2)
	Number of neurons	26
XGBoost	Learning rate	0.2
	Max depth	6
	Number of estimators	100

For the GAN model, key parameters were tuned with particular attention to their impact on the training dynamics and stability of the generated data distribution. We experimented with the learning rate over a range of [0.001, 0.005], with 0.002 emerging as the optimal setting, balancing convergence speed and training stability. Batch size was tested in increments of 32, ranging from 32 to 128; a batch size of 64 provided the best memory efficiency without compromising data richness for training. Hidden layers were tested with neuron counts from 50 to 120, and a configuration of 80 neurons per layer was selected to ensure adequate capacity for feature extraction without overfitting. Additional parameters, such as the momentum term (0.05) and the *ReLU* activation function, were introduced to enhance stability, as these values were found to effectively minimize fluctuations in training loss. The 5DGWO optimizer was configured with a convergence threshold of 0.059 and a maximum of 300 epochs, as these parameters maintained model accuracy and prevented premature convergence. The AE model's hyper-parameters were similarly refined using trial-and-error, targeting optimal reconstruction accuracy. Initial tests set the learning rate between [0.0005, 0.002], with 0.001 found to offer the best trade-off between training speed and stability. Hidden layer sizes were progressively reduced to capture essential features while preserving important information; this approach prevented over-parameterization and maintained model efficiency. A batch size of 128 was chosen to optimize convergence speed without overloading computational resources. These configurations were iteratively tested and fine-tuned to produce an AE model that achieves high reconstruction accuracy and stable training.

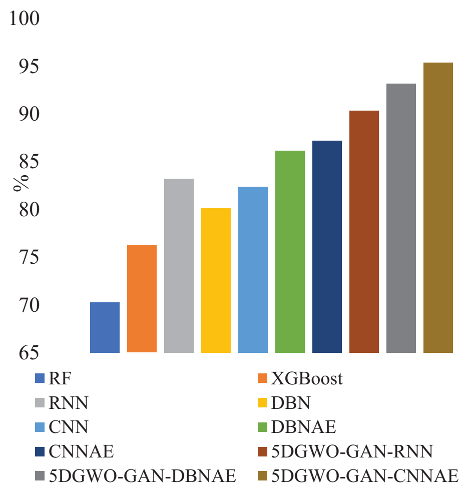
To ensure balanced exploration and exploitation within the 5DGWO algorithm, control parameters C and A were optimized with initial values tested in the range [0.2, 0.9]. Values of 0.7 and 0.3, respectively, were found most effective in balancing search depth and breadth. The population size was tested from 80 to 150 wolves, with 110 selected to provide adequate diversity without incurring excessive computational cost. The number of iterations was set to 300 after testing a range of 200 to 500 iterations, allowing sufficient convergence time without compromising efficiency. The parameter α was optimized within [0, 2], providing necessary flexibility in guiding wolves while achieving fast convergence. These settings allowed 5DGWO to adapt dynamically to the GAN's complex parameter space, refining the search with each iteration. For DBN, a learning rate of 0.06 was selected after trials ranging from 0.01 to 0.1, paired with 8 hidden layers of 44 neurons each, a balance that provided effective depth for feature extraction while avoiding overfitting. The RNN was calibrated with a dropout rate tested in the range [0.1, 0.3], and a rate of 0.2 was finalized to prevent overfitting. Activation functions *Tanh* and Sigmoid were chosen for their non-linear properties, which improved sequential data processing. For the CNN, two convolutional layers with 3×3 kernels and max pooling layers were optimized to maintain key features while reducing dimensionality, preserving essential patterns in data. Finally, XGBoost underwent learning rate tuning in the range [0.05, 0.2], with 0.1 chosen to balance depth and convergence speed. A max tree depth of 6 was selected to prevent overfitting while maintaining model complexity. These parameter settings were fine-tuned using a combination of grid search and manual adjustment through trial-and-error, and they represent the best performing configurations after extensive testing.

5.2 Algorithm Results on NSL-KDD Dataset

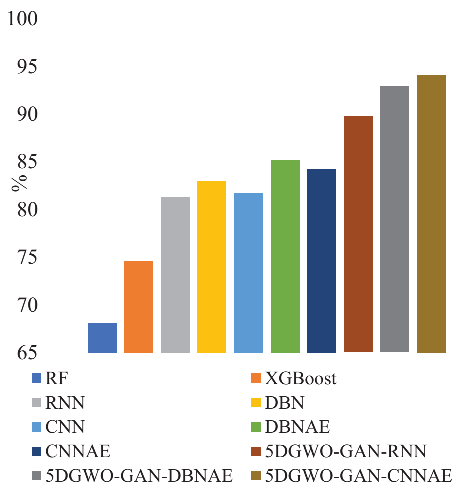
In this dataset, we investigated both binary and multiclass classification tasks. In our experiments with models such as 5DGWO-GAN-CNN$_{AE}$, 5DGWO-GAN-DBN$_{AE}$, and 5DGWO-GAN-RNN, we generated synthetic data for each class using the generative model and incorporated it into the training dataset. The results presented in Table 2 highlight a clear performance distinction between traditional ML models, DL approaches, and the more advanced 5DGWO-GAN-enhanced models in binary classification. It's important to highlight that, in the binary classification task, the attack class data are inherently treated as anomalies and labeled as abnormal. Fig. 7a provides a comparison of binary classification accuracy on the NSL-KDD dataset. In general, the models demonstrated relatively high recall rates for the normal class data, while for the abnormal class, the models exhibited higher precision values.

Table 2: The comparison of various models for binary classification on the NSL-KDD dataset

Models	Accuracy	RMSE	Normal		Abnormal	
			Recall	precision	Recall	precision
RF	70.25%	13.27	92.21%	61.49%	77.98%	76.24%
XGBoost	76.21%	10.25	93.24%	67.93%	82.93%	75.19%
RNN	83.21%	5.29	95.41%	74.39%	83.64%	85.67%
DBN	80.14%	5.84	93.29%	73.29%	80.19%	80.27%
CNN	82.34%	6.32	95.19%	71.36%	81.32%	82.39%
DBN_{AE}	86.14%	3.41	96.19%	83.19%	84.29%	88.94%
CNN_{AE}	87.19%	2.96	96.29%	82.14%	85.14%	89.63%
5DGWO-GAN-RNN	90.32%	1.87	98.29%	86.19%	87.39%	93.28%
5DGWO-GAN-DBN_{AE}	93.19%	1.04	99.24%	90.29%	88.96%	96.71%
5DGWO-GAN-CNN_{AE}	95.34%	0.24	99.63%	91.32%	90.24%	98.32%



(a)



(b)

Figure 7: A comparison of classification accuracy on the NSL-KDD dataset: (a) binary and (b) multiclass

Traditional methods like RF and XGBoost show relatively lower accuracy (70.25% and 76.21%, respectively) and higher RMSE values, indicating less precision and greater variability in predictions. While they manage decent recall values for both normal and abnormal classes, their precision, particularly for normal instances, is notably lower, suggesting a higher rate of false positives. In contrast, DL methods such as RNN, DBN, and CNN exhibit substantial improvements in accuracy, with values above 80%, as well as significantly lower RMSE scores. However, these models still struggle with maintaining a consistent balance between recall and precision, particularly in the abnormal class, where misclassifications are more frequent.

The introduction of AE-enhanced models (DBN_{AE} and CNN_{AE}) results in marked improvements in both accuracy and precision across all classes, with much lower RMSE values (as low as 2.96 for CNN_{AE}). These models benefit from the AE's ability to better capture complex data patterns, improving both recall and precision, particularly for the abnormal class. However, the highest-performing models in the experiment are those optimized using 5DGWO-GAN. These models, particularly 5DGWO-GAN-CNN_{AE}, achieve the best overall performance, with accuracy reaching 95.34%, the lowest RMSE of 0.24, and near-perfect recall and precision scores for both normal and abnormal classes. This clearly demonstrates that the 5DGWO-GAN optimization significantly enhances model performance by effectively tuning hyper-parameters and improving the detection of both normal and abnormal instances, making it the most robust solution for the NSL-KDD intrusion detection dataset.

The results in Table 3 provide a comprehensive comparison of various models for multiclass classification on the NSL-KDD dataset. In contrast to the binary classification, the system was able to identify the specific type of threat associated with the data and subsequently generate synthetic data of varying magnitudes based on the population weights. Fig. 7b presents a comparison of multiclass classification accuracy on the NSL-KDD dataset. RF and XGBoost show relatively lower performance compared to DL-based approaches. RF achieves an accuracy of 68.12% with a high RMSE of 22.18, reflecting substantial variability in its predictions. XGBoost performs slightly better, with an accuracy of 74.59% and a lower RMSE of 19.96. However, both models exhibit weaknesses in recall, particularly for difficult-to-detect classes like R2L and U2R. When comparing DL models such as RNN, DBN, and CNN, we observe a marked improvement in both accuracy and RMSE. These models not only increase accuracy (achieving over 80% for all three) but also show a better balance between recall and precision across different attack types. RNN achieves an accuracy of 81.34% with an RMSE of 12.57, offering higher recall for the R2L (22.74%) and U2R (8.51%) classes than the traditional models. DBN and CNN further improve these results, with accuracies of 82.96% and 81.74%, respectively, and lower RMSE values. These models still struggle with detecting minority classes like U2R but perform better in capturing the patterns of the Probe and DoS classes.

Table 3: The comparison of various models for multiclass classification on the NSL-KDD dataset

Models	Accuracy	RMSE	DoS		Probe		R2L		U2R	
			Recall	precision	Recall	precision	Recall	precision	Recall	precision
RF	68.12%	22.18	72.36%	85.11%	41.77%	77.88%	12.62%	55.42%	2.01%	60.13%
XGBoost	74.59%	19.96	77.81%	82.45%	49.09%	81.63%	18.07%	65.36%	4.27%	32.22%
RNN	81.34%	12.57	83.98%	79.87%	47.45%	79.94%	22.74%	69.77%	8.51%	70.50%
DBN	82.96%	11.58	85.76%	81.94%	45.91%	76.50%	20.26%	67.70%	10.36%	72.84%
CNN	81.74%	10.02	86.21%	82.12%	42.37%	81.45%	30.90%	72.89%	14.60%	69.19%
DBN_{AE}	85.19%	6.47	90.22%	88.37%	88.65%	90.08%	41.48%	80.34%	18.07%	75.64%
CNN_{AE}	84.27%	5.86	91.75%	89.76%	90.18%	92.77%	49.71%	81.30%	20.84%	77.46%
5DGWO-GAN-RNN	89.74%	2.95	97.8/%	94.88%	95.22%	95.46%	82.61%	90.25%	37.26%	82.28%
5DGWO-GAN-DBN_{AE}	92.88%	1.89	98.47%	96.27%	97.38%	96.18%	92.57%	96.62%	34.37%	85.88%
5DGWO-GAN-CNN_{AE}	94.12%	0.86	99.09%	97.64%	98.94%	97.83%	96.05%	93.11%	36.86%	88.41%

The inclusion of AEs in DBN_{AE} and CNN_{AE} significantly enhances performance, as demonstrated by their higher accuracies (85.19% for DBN_{AE} and 84.27% for CNN_{AE}) and substantially lower RMSE scores (6.47 and 5.86, respectively). These models also show much better recall for the minority classes, particularly with DBN_{AE} reaching 18.07% for U2R and CNN_{AE} achieving 20.84%. The 5DGWO-GAN optimized models deliver the most robust results, particularly in handling all classes more effectively. 5DGWO-GAN-RNN, 5DGWO-GAN-DBN_{AE}, and 5DGWO-GAN-CNN_{AE} achieve the highest overall accuracies, with 5DGWO-GAN-CNN_{AE} reaching an impressive 94.12%. These models also demonstrate the lowest RMSE values (as low as 0.86 for 5DGWO-GAN-CNN_{AE}), indicating minimal prediction errors. Furthermore, they significantly outperform other models in detecting the difficult R2L and U2R classes, with 5DGWO-GAN-CNN_{AE} achieving a recall of 96.05% for R2L and 36.86% for U2R, which are far superior to other models. The incorporation of 5DGWO optimization ensures more balanced detection across all classes, making these models the most effective for multiclass intrusion detection in the NSL-KDD dataset.

5.3 Algorithm Results on UNSW-NB15 Dataset

To evaluate model performance on a dataset with a wider variety of classes, we also performed experiments using the UNSW-NB15 dataset as an additional multiclass classification scenario. This dataset includes ten different classes, one normal class, along with three major and six minor attack classes. Similar to previous experiments, synthetic data was generated for each class using the generative model and integrated into the training data. Table 4 summarizes the experimental results for the multiclass classification scenario on the UNSW-NB15 dataset. RF and XGBoost perform moderately well, especially in the Generic class, with accuracies of 72.96% and 73.47%, respectively. However, these models struggle in more complex attack categories such as DoS and Shellcode, where their performance drops significantly.

Table 4: The comparison of various models for multiclass classification on the UNSW-NB15 dataset

Models	Generic	Exploit	Fuzzers	DoS	Reconnaissance	Analysis	Backdoors	Shellcode	Worms
RF	72.96%	44.19%	61.14%	13.08%	47.44%	68.20%	77.24%	50.17%	55.33%
XGBoost	73.47%	40.73%	66.58%	14.16%	52.50%	69.11%	80.80%	51.81%	54.26%
RNN	78.83%	54.03%	74.76%	25.29%	50.63%	74.48%	85.01%	55.20%	61.55%
DBN	81.24%	56.25%	72.35%	27.30%	55.08%	76.75%	84.33%	59.35%	63.90%
CNN	80.90%	55.50%	66.70%	28.66%	54.64%	78.94%	87.76%	57.69%	62.04%
DBN_{AE}	85.05%	58.40%	76.92%	30.53%	66.43%	83.97%	90.95%	65.88%	68.64%
CNN_{AE}	86.64%	60.65%	77.28%	33.74%	68.28%	86.60%	91.15%	63.18%	70.27%
5DGWO-GAN-RNN	90.76%	67.51%	92.17%	39.99%	75.73%	88.31%	95.34%	71.51%	75.39%
5DGWO-GAN-DBN_{AE}	93.34%	71.27%	94.43%	41.28%	77.90%	90.88%	98.28%	74.24%	78.62%
5DGWO-GAN-CNN_{AE}	95.68%	70.38%	96.22%	44.01%	80.07%	93.70%	97.72%	79.08%	81.26%

In contrast, DL models like RNN, DBN, and CNN exhibit better overall performance, particularly in detecting attacks like Exploit and Fuzzers, with RNN achieving 74.76% in Fuzzers and 25.29% in DoS. The introduction of AE-enhanced models like DBN_{AE} and CNN_{AE} brings a noticeable improvement in classification accuracy. DBN_{AE} consistently outperforms traditional models, especially in detecting major attack types such as Generic (85.05%), Fuzzers (76.92%), and DoS (30.53%), while CNN_{AE} further strengthens the model's ability to detect classes like Reconnaissance (68.28%) and Shellcode (63.18%). The standout performers, however, are the 5DGWO-GAN-optimized models, with 5DGWO-GAN-CNN_{AE} emerging as the top-performing model. It achieves the highest accuracy across several classes, including Generic (95.68%), Fuzzers (96.22%), and DoS (44.01%). These models demonstrate their strength particularly in the major classes, but also in minor and complex classes such as Shellcode (79.08%) and Worms (81.26%). 5DGWO-GAN-DBN_{AE} also delivers high performance, achieving significant improvements in classes like Exploit (71.27%) and Backdoors (98.28%).

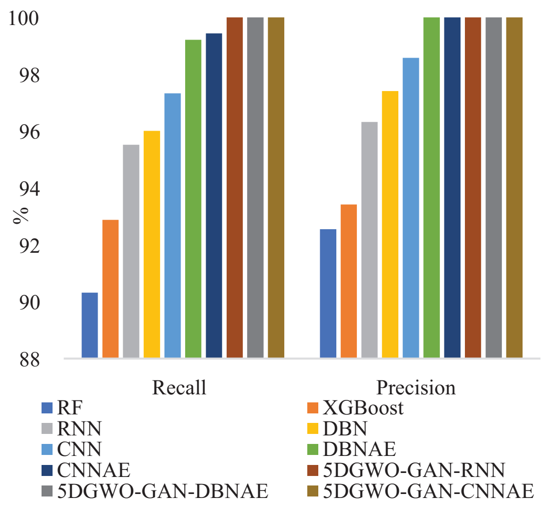
For the major attack classes, such as Generic, Exploit, and Fuzzers, both simple and advanced DL models performed comparably, with our proposed models showing improvements even in RNN-based systems. The generative model-enhanced approaches significantly improved classification in the Generic and Fuzzers classes. For the minor classes, the proposed models displayed moderate improvements in performance across the board. By testing on the more diverse UNSW-NB15 dataset, it was evident that the proposed models enhanced classification performance, particularly for the major classes. Additionally, the generative model showed its effectiveness in boosting performance for minor and rare attack classes. Despite these gains, challenges remain, particularly in detecting certain classes, where detection rates remained relatively low.

5.4 Algorithm Results on IoT-23 Dataset

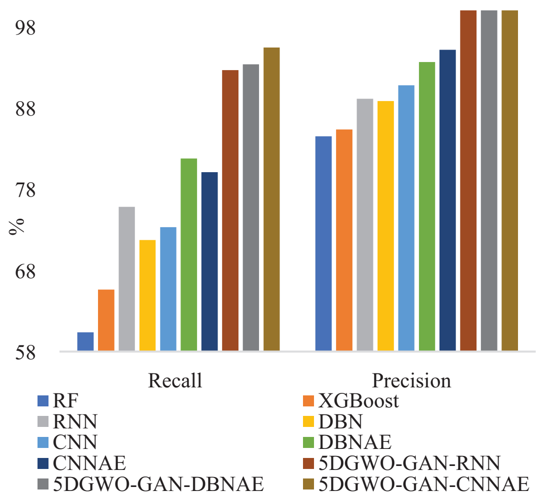
To assess the effectiveness of the proposed models in IoT environments, we carried out experiments using the IoT-23 dataset. Specifically, we utilized data from the Mirai botnet scenario (CTU-IoT-Malware-Capture-34-1) and intentionally created an extreme data imbalance scenario to challenge the models. Table 5 outlines the results from the multiclass classification task on the IoT-23 dataset, while Fig. 8 provides a visual comparison of the experimental outcomes. In general, all models achieved over 80% accuracy, with even the simpler models showing perfect classification for the DDoS class. This strong performance can likely be attributed to the simplicity of the IoT dataset, where certain features, such as "history," contain highly indicative information about the attacks.

Table 5: The comparison of various models for multiclass classification on the IoT-23 dataset

Models	Accuracy	RMSE	DDoS		C&C		PortScan	
			Recall	precision	Recall	precision	Recall	precision
RF	81.73%	10.13	90.32%	92.55%	60.37%	84.50%	84.86%	70.34%
XGBoost	85.18%	9.12	92.88%	93.42%	65.64%	85.33%	86.12%	72.83%
RNN	90.51%	7.79	95.52%	96.32%	75.83%	89.12%	91.44%	77.58%
DBN	92.46%	6.32	96.01%	97.41%	71.74%	88.84%	90.90%	79.20%
CNN	91.24%	5.41	97.33%	98.58%	73.32%	90.79%	92.73%	80.13%
DBN_{AE}	93.72%	3.01	99.21%	100.00%	81.76%	93.65%	97.62%	83.99%
CNN_{AE}	94.30%	2.20	99.44%	100.00%	80.08%	95.15%	96.23%	85.85%
5DGWO-GAN-RNN	96.69%	1.14	100.00%	100.00%	92.64%	100.00%	100.00%	92.47%
5DGWO-GAN-DBN_{AE}	98.55%	0.76	100.00%	100.00%	93.35%	100.00%	100.00%	96.36%
5DGWO-GAN-CNN_{AE}	99.20%	0.09	100.00%	100.00%	95.42%	100.00%	100.00%	95.42%



(a)



(b)

Figure 8: A visual comparison of proposed models on the IoT-23 dataset: (a) DDoS, and (b) C&C

RF and XGBoost models demonstrate moderate performance, with accuracies of 81.73% and 85.18%, respectively. In contrast, DL models like RNN, DBN, and CNN show a substantial improvement in both accuracy and RMSE. For instance, RNN reaches an accuracy of 90.51%, with notably better performance in detecting C&C attacks (75.83% recall), though precision in PortScan detection (77.58%) still shows room for improvement. DBN and CNN further enhance performance, achieving accuracies of 92.46% and 91.24%, respectively, and better balancing precision and recall across all attack types. However, the standout models are those enhanced with AE, specifically DBN_{AE} and CNN_{AE}, which significantly boost detection rates. Both models achieve over 93% accuracy and nearly perfect recall and precision for DDoS and C&C classes, demonstrating the added value of AE-based feature extraction. The 5DGWO-GAN-optimized models demonstrate superior performance across all metrics, with 5DGWO-GAN-RNN reaching an accuracy of 96.69% and near-perfect recall and precision in detecting all attack classes. The 5DGWO-GAN-DBN_{AE} and 5DGWO-GAN-CNN_{AE} models further push the boundaries, achieving over 98.5% accuracy, with perfect 100% recall and precision in detecting DDoS. These models also excel in detecting PortScan attacks, where 5DGWO-GAN-CNN_{AE} attains 100% for recall. This highlights the effectiveness of the 5DGWO-GAN models, particularly in IoT-based environments where a diverse range of attack types demands both precise detection and broad generalization capabilities.

5.5 Convergence Curve and Inference Time of Algorithm

The convergence curves in Fig. 9 clearly demonstrate the superior performance of the 5DGWO-GAN-CNN_{AE} model across all three datasets (NSL-KDD, UNSW-NB15, and IoT-23). This model consistently achieves the fastest and most stable convergence, as indicated by its rapidly decreasing RMSE values, which approach near-zero much earlier than other models. The success of this model can be attributed to the integration of the 5DGWO, which optimizes the hyper-parameters of the GAN architecture, thereby enhancing the quality of synthetic data generation and improving the model's learning process. Furthermore, the combination of CNN and AE in the architecture plays a critical role in effective feature extraction and dimensionality reduction, allowing the model to capture complex patterns in the data with greater precision. The 5DGWO-GAN-CNN$_{AE}$ model's remarkable convergence can also be linked to its ability to balance the strengths of each component. The GAN architecture, optimized by 5DGWO, ensures high-quality synthetic data generation, which enhances the model's training data. The CNN extracts spatial features efficiently, while the AE component helps generalize the model by reducing overfitting and improving its adaptability to various datasets. This powerful combination allows the model to learn more quickly and accurately than traditional models such as RF, XGBoost, or even baseline DL models like RNN and CNN.

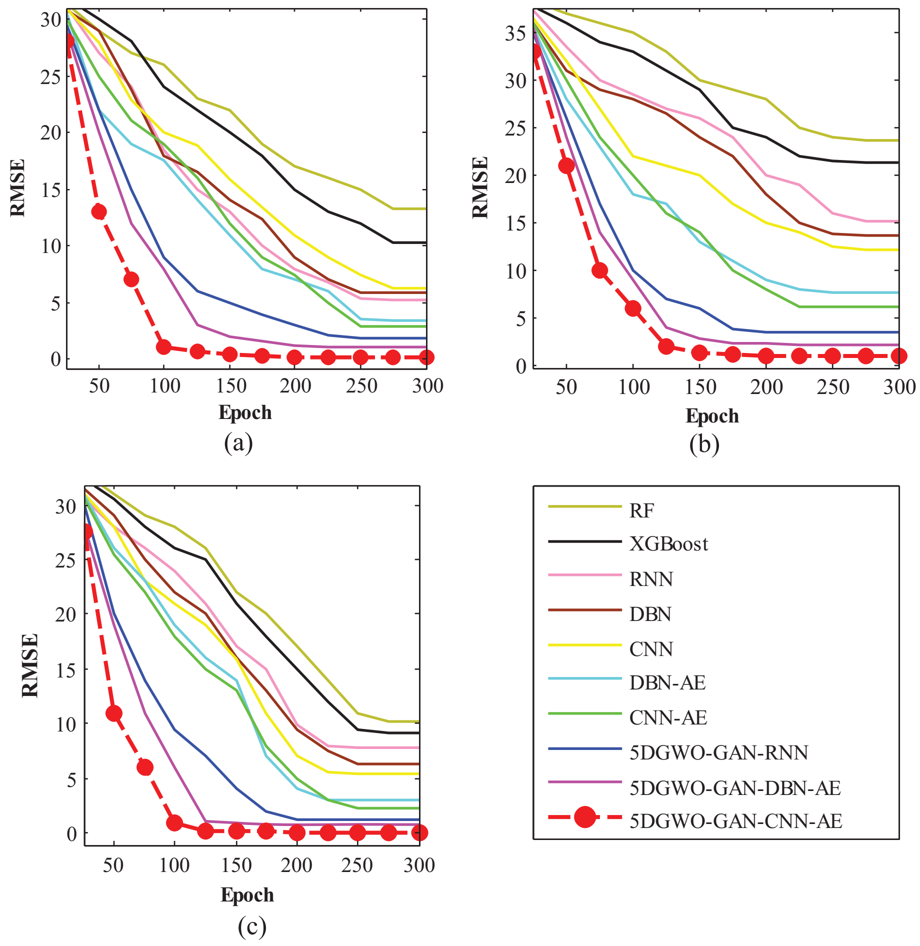


Figure 9: The convergence curve of models: (a) NSL-KDD, (b) UNSW-NB15, and (c) IoT-23 dataset

Table 6 provides a comprehensive overview of the average inference time (in seconds) for each model on the NSL-KDD (binary classification) and IoT-23 datasets under various RMSE termination conditions. Analysis of the results shows that traditional ML models such as RF and XGBoost exhibit the longest inference times across both datasets, especially under tighter RMSE thresholds. DL models like RNN, DBN, and CNN demonstrate relatively lower inference times compared to RF and XGBoost but still require significantly more time as the RMSE condition becomes stricter. AE-enhanced models (DBN$_{AE}$ and CNN$_{AE}$) provide a noticeable reduction in inference time, benefiting from the feature extraction capabilities of the AE that streamline the data processing pipeline.

However, the models optimized with the 5DGWO-GAN framework, particularly 5DGWO-GAN-CNN_{AE}, consistently achieve the lowest inference times across all RMSE conditions. This reduction in processing time can be attributed to the hyper-parameter optimization provided by 5DGWO, which not only enhances model accuracy but also contributes to computational efficiency. For instance, 5DGWO-GAN-CNN_{AE} records an inference time of only 49s for RMSE < 15 on the NSL-KDD dataset, demonstrating its practical relevance and effectiveness for real-time IoT applications where latency is crucial.

Table 6: Comparison of algorithm inference time based on RMSE termination conditions

Models	NSL-KDD dataset (binary)			IoT-23 dataset		
	RMSE < 15	RMSE < 9	RMSE < 3	RMSE < 15	RMSE < 9	RMSE < 3
RF	657	–	–	612	–	–
XGBoost	591	–	–	524	–	–
RNN	329	612	–	309	563	–
DBN	372	621	–	357	603	–
CNN	335	607	–	321	584	–
DBN_{AE}	214	493	–	205	483	–
CNN_{AE}	196	406	983	189	428	960
5DGWO-GAN-RNN	120	212	684	125	201	621
5DGWO-GAN-DBN_{AE}	88	145	403	79	132	392
5DGWO-GAN-CNN_{AE}	49	102	281	35	92	268

6 Conclusions

In this paper, we proposed a novel intrusion detection framework, 5DGWO-GAN-CNN_{AE}, designed for detecting anomalies in IoT networks. Our approach addresses the challenges of hyper-parameter optimization in GANs, which are essential for accurately identifying cyberattacks in dynamic environments. To enhance the performance of GANs, we introduced the 5DGWO, which incorporates two additional types of wolves, gamma and theta, to improve both the exploration and exploitation phases of the optimization process. The inclusion of these wolves enables better tuning of critical hyper-parameters, improving both convergence speed and detection accuracy. Our experiments were conducted on three benchmark datasets (NSL-KDD, UNSW-NB15, and IoT-23) covering a diverse range of network traffic scenarios. The results consistently demonstrated that the 5DGWO-GAN-CNN_{AE} model outperformed traditional ML models and baseline DL models across all key performance metrics, including accuracy, recall, precision, and RMSE. In the binary classification task on the NSL-KDD dataset, the 5DGWO-GAN-CNN_{AE} model achieved a 95.34% accuracy, with the lowest RMSE of 0.24. In multiclass classification tasks, this model continued to outperform, especially in detecting difficult-to-detect classes like R2L and U2R.

Further results from the UNSW-NB15 and IoT-23 datasets confirmed that the 5DGWO-GAN optimized models exhibited superior convergence and detection performance. The 5DGWO-GAN-CNN_{AE} model achieved near-perfect accuracy and recall rates in detecting DDoS and C&C attacks on the IoT-23 dataset, with 100% recall for the DDoS class. These results confirm that the inclusion of generative models (5DGWO-GANs), combined with CNN and AE components, allows for more efficient feature extraction and dimensionality reduction, leading to higher detection accuracy in both major and minor attack classes. The optimization provided by 5DGWO helps to overcome common challenges in GAN training, such as getting stuck in local minima or suffering from slow convergence. The ability to generate synthetic data through GANs also enriches the training dataset, enabling the model to generalize better to unseen attack patterns, which is crucial in the dynamic and evolving nature of IoT systems.

The proposed 5DGWO-GAN framework is designed with scalability in mind, addressing the high-volume, complex, and real-time requirements of IoT networks. By employing mini-batch processing and leveraging 5DGWO optimization, the model efficiently manages memory, reduces computational load, and adapts well to large-scale datasets. These strategies enhance the model's robustness and stability, maintaining performance as data volume and complexity increase. Extensive testing on NSL-KDD, UNSW-NB15, and IoT-23 datasets demonstrates the model's ability to generalize from test scenarios to comprehensive, real-world IoT applications. The framework's resource-efficient design enables deployment across a range of IoT devices, from powerful servers to edge devices, supporting both centralized and distributed environments. The efficient structures of CNN and AE layers further enhance real-time detection capabilities, making the model well-suited for the dynamic and high-throughput conditions of IoT networks. However, despite the significant improvements in detection accuracy and efficiency, some limitations remain. One challenge is the computational complexity of training GANs and the associated 5DGWO optimization, which can be resource-intensive, particularly for large-scale IoT environments. Furthermore, while our approach has demonstrated effectiveness on benchmark datasets, real-world IoT environments are even more heterogeneous and could present additional challenges in terms of data diversity and scalability. Although generative models help with data scarcity, class imbalance remains a challenge, especially when rare attack types are underrepresented. Generating high-quality synthetic data for these minority classes without overfitting is complex and can affect detection accuracy. Additionally, GANs are highly sensitive to hyper-parameter tuning, and finding the optimal settings across different datasets is difficult.

Looking forward, several directions for future research can be pursued. First, the scalability of the 5DGWO-GAN framework can be further optimized to handle real-time intrusion detection in large-scale IoT networks. Incorporating more advanced generative models and exploring distributed GANs could also help to address scalability concerns. In addition, expanding the system to work with even larger and more diverse datasets, including real-time traffic from a wider range of IoT devices, will be important for improving the system's robustness and adaptability to emerging threats. Furthermore, integrating our framework with edge computing techniques could allow for faster, more localized detection, making the system more practical for real-world deployment in resource-constrained IoT environments.

Acknowledgement: The authors would like to thank the editors and reviewers for their invaluable feedback and constructive comments, which have significantly contributed to improving the quality of this paper.

Funding Statement: The work described in this paper has been developed within the project PRESECREL (PID2021-124502OB-C43). We would like to acknowledge the financial support of the Ministerio de Ciencia e Investigación (Spain), in relation to the Plan Estatal de Investigación Científica y Técnica y de Innovación 2017–2020.

Author Contributions: The authors confirm contribution to the paper as follows: study conception and design: Sarvenaz Sadat Khatami, Mehrdad Shoeibi, Anita Ershadi Oskouei, Diego Martín; data collection: Sarvenaz Sadat Khatami, Mehrdad Shoeibi, Anita Ershadi Oskouei, Maral Keramat Dashliboroun; analysis and interpretation of results: Sarvenaz Sadat Khatami, Anita Ershadi Oskouei, Diego Martín; draft manuscript preparation: Sarvenaz Sadat Khatami, Mehrdad Shoeibi, Diego Martín, Maral Keramat Dashliboroun; supervision: Diego Martín. All authors reviewed the results and approved the final version of the manuscript.

Availability of Data and Materials: The data that support the findings of this study are available from the corresponding author, upon reasonable request.

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest to report regarding the present study.

References

[1] P. Sun *et al.*, "A survey of IoT privacy security: Architecture, technology, challenges, and trends," *IEEE Internet Things*, vol. 11, no. 21, pp. 34567–34591, 2024. doi: 10.1109/JIOT.2024.3372518.

[2] F. Najafi, M. Kaveh, D. Martin, and M. R. Mosavi, "Deep PUF: A highly reliable DRAM PUF-based authentication using deep convolutional neural networks," *Sensors*, vol. 21, no. 6, 2021, Art. no. 2009. doi: 10.3390/s21062009.

[3] M. Kaveh, Z. Yan, and R. Jäntti, "Secrecy performance analysis of RIS-aided smart grid communications," *IEEE Trans. Ind. Inform.*, vol. 20, no. 3, pp. 5415–5427, 2023. doi: 10.1109/TII.2023.3333842.

[4] M. Kaveh and M. R. Mosavi, "A lightweight mutual authentication for smart grid neighborhood area network communications based on physically unclonable function," *IEEE Syst. J.*, vol. 14, no. 3, pp. 4535–4544, 2020. doi: 10.1109/JSYST.2019.2963235.

[5] M. Kaveh, M. R. Mosavi, D. Martín, and S. Aghapour, "An efficient authentication protocol for smart grid communication based on on-chip-error-correcting physical unclonable function," *Sustain. Energy Grid. Netw.*, vol. 36, 2023, Art. no. 101228. doi: 10.1016/j.segan.2023.101228.

[6] M. Ali *et al.*, "A smart digital twin enabled security framework for vehicle-to-grid cyber-physical systems," *IEEE Trans. Inf. Forensics Secur.*, vol. 18, pp. 5258–5271, 2023. doi: 10.1109/TIFS.2023.3305916.

[7] M. Kaveh, D. Martin, and M. R. Mosavi, "A lightweight authentication scheme for V2G communications: A PUF-based approach ensuring cyber/physical security and identity/location privacy," *Electronics*, vol. 9, no. 8, 2020, Art. no. 1479. doi: 10.3390/electronics9091479.

[8] F. R. Ghadi, M. Kaveh, and D. Martín, "Performance analysis of RIS/STAR-IOS-aided V2V NOMA/OMA communications over composite fading channels," *IEEE Trans. Intell. Vehicles*, vol. 9, no. 1, pp. 279–286, 2023. doi: 10.1109/TIV.2023.3337898.

[9] Q. Abu Al-Haija, S. Altamimi, and M. AlWadi, "Analysis of extreme learning machines (ELMs) for intelligent intrusion detection systems: A survey," *Expert. Syst. Appl.*, vol. 253, no. 2, 2024, Art. no. 124317. doi: 10.1016/j.eswa.2024.124317.

[10] G. Li and J. J. Jung, "Deep learning for anomaly detection in multivariate time series: Approaches, applications, and challenges," *Inf. Fusion*, vol. 91, no. 2, pp. 93–102, 2023. doi: 10.1016/j.inffus.2022.10.008.

[11] M. M. Inuwa and R. Das, "A comparative analysis of various machine learning methods for anomaly detection in cyber attacks on IoT networks," *Internet Things*, vol. 26, no. 71, 2024, Art. no. 101162. doi: 10.1016/j.iot.2024.101162.

[12] M. Douiba, S. Benkirane, A. Guezzaz, and M. Azrour, "An improved anomaly detection model for IoT security using decision tree and gradient boosting," *J. Supercomput.*, vol. 79, no. 3, pp. 3392–3411, 2023. doi: 10.1007/s11227-022-04783-y.

[13] M. Ghiasi *et al.*, "A comprehensive review of cyber-attacks and defense mechanisms for improving security in smart grid energy systems: Past, present and future," *Electr. Power Syst. Res.*, vol. 215, 2023, Art. no. 108975. doi: 10.1016/j.epsr.2022.108975.

[14] S. Agrawal and J. Agrawal, "Survey on anomaly detection using data mining techniques," *Procedia Comput. Sci.*, vol. 60, no. 3, pp. 708–713, 2015. doi: 10.1016/j.procs.2015.08.220.

[15] W. Lim, K. Y. S. Chek, L. B. Theng, and C. T. C. Lin, "Future of generative adversarial networks (GAN) for anomaly detection in network security: A review," *Comput. Secur.*, vol. 139, no. 1, 2024, Art. no. 103733. doi: 10.1016/j.cose.2024.103733.

[16] S. Hettich and S. D. Bay, "KDD cup 1999 data," 1999. Accessed: Dec. 04, 2024. [Online]. Available: http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html

[17] D. D. Protić, "Review of KDD Cup'99, NSL-KDD and Kyoto 2006+ datasets," *Vojnotehnički Glasnik/Military Tech. Courier*, vol. 66, no. 3, pp. 580–596, 2018. doi: 10.5937/vojtehg66-16670.

[18] S. M. Kasongo and Y. Sun, "Performance analysis of intrusion detection systems using a feature selection method on the UNSW-NB15 dataset," *J. Big Data*, vol. 7, no. 1, 2020, Art. no. 105. doi: 10.1186/s40537-020-00379-6.

[19] A. Parmisano, S. Garcia, and M. J. Erquiaga, "A labeled dataset with malicious and benign IoT network traffic," 2020. Accessed: Dec. 04, 2024. [Online]. Available: https://www.stratosphereips.org/datasets-iot23

[20] P. Biswas and T. Samanta, "Anomaly detection using ensemble random forest in wireless sensor network," *Int. J. Inform. Technol.*, vol. 13, no. 5, pp. 2043–2052, 2021. doi: 10.1007/s41870-021-00717-8.

[21] H. Liu and H. Wang, "Real-time anomaly detection of network traffic based on CNN," *Symmetry*, vol. 15, no. 6, 2023, Art. no. 1205. doi: 10.3390/sym15061205.

[22] Y. Imrana, Y. Xiang, L. Ali, and Z. AbdulRauf, "A bidirectional LSTM deep learning approach for intrusion detection," *Expert. Syst. Appl.*, vol. 185, no. 8, 2021, Art. no. 115524. doi: 10.1016/j.eswa.2021.115524.

[23] I. Sohn, "Deep belief network-based intrusion detection techniques: A survey," *Expert. Syst. Appl.*, vol. 167, 2021, Art. no. 114170. doi: 10.1016/j.eswa.2020.114170.

[24] B. Ingre and A. Yadav, "Performance analysis of NSL-KDD dataset using ANN," in *Proc. Int. Conf. Signal Process. Commun. Eng. Syst.*, Andhra Pradesh, India, Jan. 2015, pp. 92–96.

[25] Y. Gao, Y. Liu, Y. Jin, J. Chen, and H. Wu, "A novel semi-supervised learning approach for network intrusion detection on cloud-based robotic system," *IEEE Access*, vol. 6, pp. 50927–50938, 2018. doi: 10.1109/ACCESS.2018.2868171.

[26] K. Alrawashdeh and C. Purdy, "Toward an online anomaly intrusion detection system based on deep learning," in *Proc. IEEE 15th Int. Conf. Mach. Learn. Appl. (ICMLA)*, Anaheim, CA, USA, 2016, pp. 195–200.

[27] T. A. Tang, L. Mhamdi, D. McLernon, S. A. R. Zaidi, and M. Ghogho, "Deep learning approach for network intrusion detection in software defined networking," in *Proc. Int. Conf. Wirel. Netw. Mobile Commun. (WINCOM)*, 2016, pp. 258–263.

[28] Y. Imamverdiyev and F. Abdullayeva, "Deep learning method for denial-of-service attack detection based on restricted Boltzmann machine," *Big Data*, vol. 6, no. 2, pp. 159–169, 2018. doi: 10.1089/big.2018.0023.

[29] W. Zhong, N. Yu, and C. Ai, "Applying big data based deep learning system to intrusion detectio," *Big Data Min. Anal.*, vol. 3, no. 3, pp. 181–195, 2020.

[30] M. H. Haghighat and J. Li, "Intrusion detection system using voting based neural network," *Tsinghua Sci. Technol.*, vol. 26, no. 4, pp. 484–495, 2021.

[31] Y. Yang *et al.*, "ASTREAM: Data-stream-driven scalable anomaly detection with accuracy guarantee in IIoT environment," *IEEE Trans. Netw. Sci. Eng.*, vol. 10, no. 5, pp. 3007–3016, Mar. 2022. doi: 10.1109/TNSE.2022.3157730.

[32] F. T. Liu, K. M. Ting, and Z. -H. Zhou, "Isolation-based anomaly detection," *ACM Trans. Knowl. Discov. Data.*, vol. 6, no. 1, pp. 1–39, 2012. doi: 10.1145/2133360.2133363.

[33] X. Zhang *et al.*, "LSHiForest: A generic framework for fast tree isolation-based ensemble anomaly analysis," in *Proc. IEEE 33rd Int. Conf. Data Eng. (ICDE)*, Apr. 2017, pp. 983–994.

[34] L. Qi, Y. Yang, X. Zhou, W. Rafique, and J. Ma, "Fast anomaly identification based on multi-aspect data streams for intelligent intrusion detection toward secure Industry 4. 0," *IEEE Trans. Ind. Informat.*, vol. 18, no. 9, pp. 6503–6511, 2022. doi: 10.1109/TII.2021.3139363.

[35] J. Kim, J. Kim, H. L. T. Thu, and H. Kim, "Long short-term memory recurrent neural network classifier for intrusion detection," in *Proc. Int. Conf. Platform Technol. Serv. (PlatCon)*, 2016, pp. 1–5.

[36] C. Yin, Y. Zhu, J. Fei, and X. He, "A deep learning approach for intrusion detection using recurrent neural networks," *IEEE Access*, vol. 5, pp. 21954–21961, 2017. doi: 10.1109/ACCESS.2017.2762418.

[37] C. Xu, J. Shen, X. Du, and F. Zhang, "An intrusion detection system using a deep neural network with gated recurrent units," *IEEE Access*, vol. 6, pp. 48697–48707, 2018. doi: 10.1109/ACCESS.2018.2867564.

[38] J. Gao *et al.*, "Omni SCADA intrusion detection using deep learning algorithms," *IEEE Internet Things*, vol. 8, no. 2, pp. 951–961, 2021. doi: 10.1109/JIOT.2020.3009180.

[39] A. Javaid, Q. Niyaz, W. Sun, and M. Alam, "A deep learning approach for network intrusion detection system, EAI Endorsed Trans," *Secur. Saf.*, vol. 16, no. 9, 2016, Art. no. e2.

[40] B. Yan and G. Han, "Effective feature extraction via stacked sparse autoencoder to improve intrusion detection system," *IEEE Access*, vol. 6, pp. 41238–41248, 2018. doi: 10.1109/ACCESS.2018.2858277.

[41] N. Shone, T. N. Ngoc, V. D. Phai, and Q. Shi, "A deep learning approach to network intrusion detection," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 2, no. 1, pp. 41–50, 2018. doi: 10.1109/TETCI.2017.2772792.

[42] C. Ieracitano, A. Adeel, F. C. Morabito, and A. Hussain, "A novel statistical analysis and autoencoder driven intelligent intrusion detection approach," *Neurocomputing*, vol. 387, no. 1, pp. 51–62, 2020. doi: 10.1016/j.neucom.2019.11.016.

[43] J. Y. Kim, S. J. Bu, and S. B. Cho, "Malware detection using deep transferred generative adversarial networks," in *Proc. Int. Conf. NeuralInf Process.*, 2017, pp. 556–564.

[44] M. H. Shahriar, N. I. Haque, M. A. Rahman, and M. Alonso, "G-IDS: Generative adversarial networks assisted intrusion detection system," in *Proc. IEEE 44th Annu. Comput. Softw. Appl. Conf. (COMPSAC)*, Jul. 2020, pp. 376–385.

[45] I. Yilmaz, R. Masum, and A. Siraj, "Addressing imbalanced data problem with generative adversarial network for intrusion detection," in *Proc. IEEE 21st Int. Conf. Inf. Reuse Integr. Data Sci. (IRI)*, Las Vegas, NV, USA, 2020, pp. 25–30.

[46] D. Li, D. Kotani, and Y. Okabe, "Improving attack detection performance in NIDS using GAN," in *Proc. IEEE 44th Annu. Comput. Softw. Appl. Conf. (COMPSAC)*, Jul. 2020, pp. 817–825.

[47] W. Lee, B. Noh, Y. Kim, and K. Jeong, "Generation of network traffic using WGAN-GP and a DFT filter for resolving data imbalance," in *Proc. Int. Conf. Internet Distrib. Comput. Syst.*, Oct. 2019, pp. 306–317.

[48] G. Dlamini and M. Fahim, "DGM: A data generative model to improve minority class presence in anomaly detection domain," *Neural Comput. Appl.*, vol. 33, no. 20, pp. 13635–13646, 2021. doi: 10.1007/s00521-021-05993-w.

[49] D. Li, D. Chen, J. Goh, and S. K. Ng, "Anomaly detection with generative adversarial networks for multivariate time series," 2018, *arXiv:1809.04758*.

[50] S. K. Alabugin and A. N. Sokolov, "Applying of generative adversarial networks for anomaly detection in industrial control systems," in *Proc. Global Smart Ind. Conf. (GloSIC)*, Nov. 2020, pp. 199–203.

[51] I. Siniosoglou, P. Radoglou-Grammatikis, G. Efstathopoulos, P. Fouliras, and P. Sarigiannidis, "A unified deep learning anomaly detection and classification approach for smart grid environments," *IEEE Trans. Netw. Service Manage*, vol. 18, no. 2, pp. 1137–1151, 2021. doi: 10.1109/TNSM.2021.3078381.

[52] C. Park *et al.*, "An enhanced AI-based network intrusion detection system using generative adversarial networks," *IEEE Internet Things*, vol. 10, no. 3, pp. 2330–2345, 2022. doi: 10.1109/JIOT.2022.3211346.

[53] A. Javadpour, P. Pinto, F. Ja'fari, and W. Zhang, "DMAIDPS: A distributed multi-agent intrusion detection and prevention system for cloud IoT environments," *Cluster Comput.*, vol. 26, no. 1, pp. 367–384, 2023. doi: 10.1007/s10586-022-03621-3.

[54] A. K. Sangaiah *et al.*, "A hybrid heuristics artificial intelligence feature selection for intrusion detection classifiers in cloud of things," *Cluster Comput.*, vol. 26, no. 1, pp. 599–612, 2023. doi: 10.1007/s10586-022-03629-9.

[55] S. Altamimi and Q. Abu Al-Haija, "Maximizing intrusion detection efficiency for IoT networks using extreme learning machine," *Discover Internet Things*, vol. 4, no. 1, 2024, Art. no. 5. doi: 10.1007/s43926-024-00060-x.

[56] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *Proc. 34th Int. Conf. Mach. Learn. (ICML)*, 2017, pp. 214–223.

[57] D. Berthelot, T. Schumm, and L. Metz, "BEGAN: Boundary equilibrium generative adversarial networks," 2017, *arXiv:1703.10717*.