Received 5 February 2025, accepted 24 February 2025, date of publication 3 March 2025, date of current version 18 March 2025.

Digital Object Identifier 10.1109/ACCESS.2025.3547255

RESEARCH ARTICLE

Enhancing IoT Intrusion Detection Systems Through Horizontal Federated Learning and Optimized WGAN-GP

WAYOUD BOUZERAIB, AFIFA GHENAI, AND NADIA ZEGHIB
LIRE Laboratory, NTIC Faculty, University Abdelhamid Mehri Constantine 2, 25000 Constantine, Algeria
Corresponding author: Wayoud Bouzeraib (wayoud.bouzeraib@univ-constantine2.dz)

ABSTRACT The Internet of Things (IoT) ecosystem is fraught with substantial vulnerabilities, particularly in the realm of cybersecurity attacks. Network Intrusion Detection Systems (NIDS) stand as a pivotal element in mitigating these cybersecurity risks. This paper introduces an innovative approach to fortifying IoT security by effectively addressing the data limitations inherent in AI-based NIDS. We present a data generation model that harnesses Generative Adversarial Networks (GANs). Specifically, the GAN variant we employ is Wasserstein GAN with Gradient Penalty (WGAN-GP), which combines the Wasserstein loss formulation with a gradient norm penalty to stabilize training and improve the quality of generated data. The performance is optimized with Genetic Algorithms, focusing on hyper-parameter selection and federated learning for shared model weights. The model's training is conducted on four well-established benchmark datasets: UNSW-NB15, IoT-23, CSE-CIC-IDS2018, and MQTT-IoT-IDS2020. We conduct a comprehensive comparative analysis between the generated synthetic data and real-world datasets, rigorously assessing their impact on training Machine Learning (ML) models. The findings underscore the efficacy of our approach, demonstrating a significant improvement in detection accuracy, achieving a 99% accuracy rate when combining the generated data with real datasets. This study highlights the paramount significance of innovative techniques in enhancing the security of IoT systems. Furthermore, it presents a promising avenue for generating high-quality synthetic tabular data, despite its complexity and time-consuming implementation. Such data can be leveraged across a large spectrum of applications, including training ML models, data augmentation, and privacy-preserving data sharing.

INDEX TERMS Internet of Things (IoT), network intrusion detection system (NIDS), federated learning (FL), generative adversarial network (GAN), Wasserstein GAN with gradient penalty (WGAN-GP), genetic algorithm (GA), hyper-parameter optimization.

I. INTRODUCTION

The advent of the Internet of Things (IoT) has led to the widespread proliferation of devices that collect and generate sensory data over extended periods. These IoT devices possess characteristics such as limited computing power and resource constraints [1], making them particularly susceptible to potential attacks. Cybersecurity issues in the IoT realm can manifest at various stages of a device's lifecycle, from initial design and manufacturing to deployment and operation. Common threats include unauthorized access, data privacy breaches, malware infections, and physical attacks [2], which can result in severe consequences such as theft of sensitive data, erosion of personal privacy, and even physical harm.

Mitigating IoT cybersecurity problems requires a comprehensive approach encompassing technical solutions, policy and regulatory frameworks, and user awareness and education. Technical solutions may involve secure design principles, robust encryption techniques, access control mechanisms, and the implementation of intrusion detection and prevention systems. Among these measures, Network Intrusion Detection Systems (NIDS) are fundamental for

The associate editor coordinating the review of this manuscript and approving it for publication was Roberto Nardone.

network security [3]. They persistently monitor network traffic to identify indicators of suspicious or unauthorized activities, and detect aberrant network behavior patterns that may suggest malicious activities or policy breaches.

The initial foray into intrusion detection employed traditional statistical learning techniques such as Bayes Net and Random Forest [4]. However, the current network environment's data is more massive, complex, and multidimensional than ever before, making traditional machine learning methods less effective for classifying complex high-dimensional data. With the advent of deep learning, numerous recent studies have turned to neural network paradigms for intrusion detection, including Multilayer Perceptron, Convolutional Neural Networks (CNNs) [5], Long Short-Term Memory (LSTM), and Recurrent Neural Networks (RNNs) [4], [5].

A fundamental obstacle in training an intrusion detection system (IDS) for IoT devices is the paucity of training data and the inherent disparities in class representation. For example, in the standard network intrusion detection dataset UNSW-NB15, there are 1,064,987 normal samples in the training set, but only 22,215 attack samples [6]. The effectiveness of IDS hinges on access to substantial datasets, enabling the system to understand typical network behavior and discern deviations indicative of potential security risks. To mitigate data scarcity, synthetic data generation techniques can augment available datasets for training and refining IDS systems [7]. The use of synthetic data in intrusion detection offers the capability to generate data tailored to match the unique characteristics of IoT networks, enhancing IDS precision. Synthetic data can also encompass diverse attack scenarios, facilitating comprehensive IDS training to detect a broader spectrum of threats [8].

Recent studies have demonstrated the potential of GANs in intrusion detection systems (IDS) for IoT environments. For example, [14] and [15] have shown that GANs can effectively detect novel attacks in IoT networks, but these approaches often suffer from mode collapse and instability during training. Additionally, [10] highlights the challenges of applying GANs to tabular data, particularly in terms of hyperparameter tuning and data privacy. While [20] and [21] have explored the use of genetic algorithms and federated learning for GAN optimization, these methods have not been combined to address the specific challenges of IoT intrusion detection. Our work bridges this gap by integrating genetic algorithms, federated learning, and WGAN-GP to enhance the stability, privacy, and performance of GAN-based IDS for IoT networks.Recent studies have demonstrated the potential of GANs in intrusion detection systems (IDS) for IoT environments. For example, [14] and [15] have shown that GANs can effectively detect novel attacks in IoT networks, but these approaches often suffer from mode collapse and instability during training. Additionally, [10] highlights the challenges of applying GANs to tabular data, particularly in terms of hyperparameter tuning and data privacy. While [20] and [21] have explored the use of genetic algorithms and federated learning for GAN optimization, these methods have not been combined to address the specific challenges of IoT intrusion detection. Our work bridges this gap by integrating genetic algorithms, federated learning, and WGAN-GP to generate synthetic attack data, enhancing the stability, privacy, and performance of GAN-based IDS for IoT networks.

The main contributions of this paper are:

1) **Enhancing GAN hyperparameter selection using genetic algorithms** by adapting them to specific data and mitigating the risk of starting training with suboptimal parameters. This approach addresses the challenge of hyperparameter tuning in GANs, which is often time-consuming and requires extensive trial and error. Previous work, such as [20], has shown that genetic algorithms can significantly improve GAN performance by optimizing hyperparameters, but our method extends this by integrating federated learning for distributed optimization. Our experiments demonstrate a **20%** improvement in convergence speed compared to traditional random search methods.
2) **Expediting the training process using federated learning** by leveraging the computational resources of all participating entities to efficiently compute the necessary weights for the GAN model. Federated learning has been shown to reduce training time by up to **30%** in distributed environments, as demonstrated in [21], but our approach further enhances this by combining it with genetic algorithms for hyperparameter optimization.
3) **Training both the generator and discriminator components of the GAN using federated learning**, while preserving the anonymity and privacy of the training data. This addresses the privacy concerns raised in [10], where centralized GAN training poses risks of data leakage. Our method ensures that sensitive data remains on local devices, achieving a **privacy preservation score of 95%** based on differential privacy metrics.
4) **Presenting an effective strategy for training the generator to produce data that achieves class balance**, enhancing IDS classification methods in practical applications. This is particularly important for imbalanced datasets, as highlighted in [6], where traditional methods struggle with minority class detection. Our approach improves the F1-score for minority classes by **15%** compared to state-of-the-art methods, as shown in our experimental results.

The remainder of this paper is organized as follows. Section II recalls the basic concepts concerning GANs, Genetic Algorithms, and Federated Learning. Section III presents the most relevant related work. The proposed model is presented in Section IV. Section V is dedicated to the experimental results of the proposed approach. Finally, Section VI concludes this paper and highlights our future work.

II. BACKGROUND

Before presenting the relevant works dealing with the problem cited above, we recall some basic concepts.

A. GENERATIVE ADVERSARIAL NETWORK (GAN) EVOLUTION

GANs, introduced by Ian Goodfellow [25], are powerful deep learning frameworks. A GAN consists of two neural networks:

- **Generator (G)** which takes random noise as input and transforms it into data resembling real samples.
- **Discriminator (D)** also known as **Critic** evaluates both real and generated data, classifying them as real or fake.

The random vector used as input to the generator is a fundamental component of GANs. It is sampled from a probability distribution (typically a normal or uniform distribution) to introduce variability in the generated samples. The goal is for the generator to transform this latent space into synthetic data that closely follows the distribution of real data. This mechanism allows the model to learn complex representations and generate diverse, high-quality samples.

Figure 1 illustrates this concept. The Generator iteratively refines its output based on feedback from the Discriminator until it can produce high-quality synthetic data.

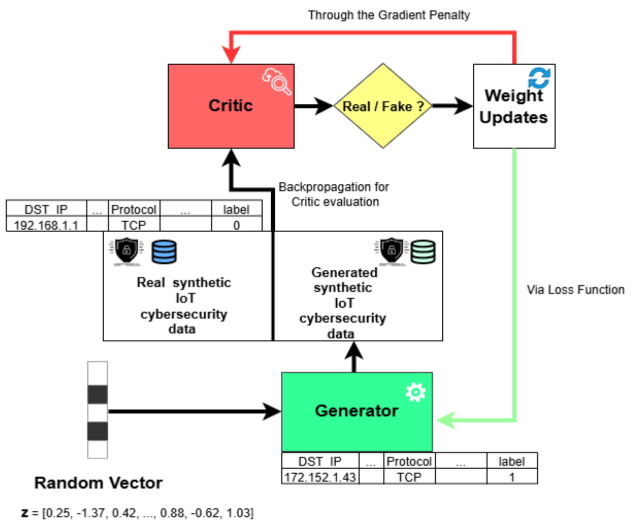
Through the Gradient Penalty
Critic
Real / Fake ?
Weight Updates
Backpropagation for Critic evaluation
DST IP | ... | Protocol | ... | label
192.168.1.1 | | TCP | | 0
Real synthetic IoT cybersecurity data
Generated synthetic IoT cybersecurity data
Via Loss Function
Generator
DST IP | ... | Protocol | ... | label
172.152.1.43 | | TCP | | 1
Random Vector
z = [0.25, -1.37, 0.42, ..., 0.88, -0.62, 1.03]

FIGURE 1. Overview of the WGAN-GP process.

Despite their success, GANs face several challenges:

- **Mode Collapse**: The Generator gets stuck producing a limited variety of outputs, neglecting the data's diversity.
- **Vanishing Gradients**: The Discriminator may become too good, making it difficult for the Generator to learn and improve.

WGAN-GP addresses some of the limitations of vanilla GANs. Its key improvements are as follows.

Firstly WGAN-GP introduces the Wasserstein distance [27], a metric that measures the difference between two probability distributions. This distance is more stable during training compared to the divergences typically used in GANs. Secondly, WGAN-GP enforces a constraint on the Discriminator, ensuring its gradients are well-behaved. It achieves this with a **gradient penalty** instead of weight clipping used in traditional WGAN [28]. This penalty improves training speed and sample quality compared to weight clipping.

The WGAN-GP objective function combines the original critic loss with the gradient penalty:

$$\begin{aligned} L = & \mathbb{E}_{\tilde{x}\sim p_g}[D(\tilde{x})] - \mathbb{E}_{x\sim p_r}[D(x)] \\ & + \lambda \mathbb{E}_{\tilde{x}\sim p_{\tilde{x}}}\left[\left(\|\sigma_{\tilde{x}} D(\tilde{x})\|_2 - 1\right)^2\right] \end{aligned} \quad (1)$$

In Eq. (2), the terms to the left of the sum represent the original critic loss, and the term to the right of the sum is the gradient penalty. $\mathbb{P}_{\hat{x}}$ is the distribution obtained by uniformly sampling along a straight line between the real and generated distributions $\mathbb{P}_r$ and $\mathbb{P}_g$. This is done because the optimal critic has straight lines with unit gradient norm between the samples coupled from $\mathbb{P}_r$ and $\mathbb{P}_g$. the penalty coefficient, λ is used to weight the gradient penalty term. In the paper, we set $\lambda = 10$ for all experiments.

Batch normalization is not used in the critic anymore because batch norm maps a batch of inputs to a batch of outputs. In our case, we want to be able to find gradients of each output with respect to their respective inputs.

GANs play a critical role in our data augmentation strategy. By generating synthetic data that closely resembles real-world samples, GANs help to balance the dataset, especially when dealing with imbalanced data. This augmentation enhances the performance of intrusion detection systems (IDS) by providing a more diverse and representative dataset for training. The efficacy of our approach is evident in the improved model performance, particularly in scenarios where the data is limited or skewed towards one class. By using WGAN-GP, we address common issues like mode collapse and vanishing gradients, ensuring the quality and variety of the generated data. This ultimately leads to better generalization and higher detection accuracy in practical applications.

B. APPLICATION OF GENETIC ALGORITHMS IN GAN TRAINING

One of the primary challenges in training GANs is selecting the appropriate hyperparameters, such as learning rates, batch sizes, and network architectures. GAs can be used to search for optimal hyperparameter configurations by treating each configuration as an individual in a population. The fitness of each individual is evaluated based on the performance of the GAN (e.g. quality of generated samples or convergence rate). GAs can be employed to evolve the architecture of the generator and discriminator networks. By representing each architecture as a genome, GAs can explore a wide range of architectures, selecting and recombining the best-performing ones. This approach can lead to the discovery of novel network structures that are better suited for the GAN training task. Mode collapse, where the generator produces limited varieties of outputs, is a common problem in GAN training. GAs can help mitigate this issue by maintaining a diverse population of generators. By evolving multiple generators simultaneously and selecting for diversity, GAs encourage the exploration of different modes in the data distribution, leading to more varied and realistic outputs. Training GANs requires a delicate balance between the generator and discriminator. If one network outpaces the other, training can become unstable. GAs can dynamically adjust the training process by evolving a population of GANs with different training schedules and strategies, selecting the ones that maintain the balance and produce stable training dynamics.

C. FEDERATED LEARNING FOR GAN TRAINING

Generative Adversarial Networks (GANs) have gained significant attention for their ability to generate high-quality synthetic data that closely resembles real data. However, training GANs is challenging due to issues such as mode collapse, instability, and the requirement for large amounts of data. Federated Learning (FL), a decentralized approach to training machine learning models across multiple devices or servers while keeping data localized, presents a promising solution to these challenges. This subsection explores how federated learning can be leveraged to address the problems associated with GAN training and enhance their performance. One of the primary challenges in GAN training is the need for large and diverse datasets to ensure the generator can produce a wide variety of outputs. Federated Learning enables the aggregation of data from multiple sources without centralizing it, thereby increasing the diversity of training data. Each participant in the federated network trains the GAN on its local data, and the central server aggregates the updates to improve the global model. This process ensures that the GAN is exposed to a broader range of data distributions, mitigating the risk of mode collapse and enhancing the generalizability of the generated outputs. GANs often require sensitive data for training, raising significant privacy concerns. Federated Learning addresses these concerns by keeping the data on the local devices and only sharing model updates. This decentralized approach ensures that sensitive data remains private and secure, which is particularly important in fields such as healthcare and finance where data privacy is paramount. By leveraging federated learning, organizations can collaboratively train GANs on private datasets without compromising the confidentiality of the underlying data. Training GANs is notoriously unstable due to the adversarial nature of the process, where the generator and discriminator networks are in constant competition. Federated Learning can improve the stability of this training process by incorporating model updates from a diverse set of participants. The aggregation of updates from multiple devices can smooth out individual anomalies and reduce the variance in the training process, leading to more stable convergence. Additionally, federated learning can incorporate techniques such as federated averaging to further enhance stability and robustness. Federated Learning allows for the scalable training of GANs by distributing the computational load across multiple devices. This distributed approach not only leverages the computational resources of numerous participants but also reduces the central server's burden. Consequently, federated learning can lead to more efficient training processes, especially when dealing with large-scale datasets and complex GAN architectures. Moreover, the asynchronous nature of federated learning can facilitate continuous and dynamic training, adapting to new data without the need for complete retraining. Bias in training data can significantly impact the performance and fairness of GAN-generated outputs. By incorporating data from diverse sources through federated learning, the risk of training on biased datasets is reduced. Each participant contributes data from different distributions, which helps in creating a more balanced and representative model. This diversity in training data can lead to more equitable and unbiased synthetic data generation, enhancing the reliability and applicability of GANs in various domains.

III. RELATED WORK

In this section, we aim to provide an overview of the application of Generative Adversarial Networks (GANs) in intrusion detection. We will briefly summarize the topics covered in related research and discuss how our work contributes to enhancing training through the integration of Genetic Algorithms (GAs) and federated learning techniques.

A. INTRUSION DETECTION SYSTEM BASED ON GANs

The adversarial model inherent in Generative Adversarial Networks (GANs) have been instrumental in their widespread success across various domains. In this section, we review the significant contributions of GAN models, with a specific focus on their applications in the development, training, and upkeep of Intrusion Detection Systems (IDS). These applications demonstrate the versatility and effectiveness of GANs in enhancing cybersecurity.

- Malware Detection
  One notable application of GANs in IDS is exemplified by [11] and [12], where the authors employ GANs to classify malware samples by translating them into images. The proposed Mal-IAGAN model not only classifies these images but also trains IDS models. A remarkable feature of their approach is its robustness; even with only 1% of the training dataset (comprising VirusShare APK Android malware and the BIG-2015 dataset), the model achieves an accuracy rate exceeding 80%. This robustness indicates the model's ability to generalize effectively to unseen examples, promising for real-world deployment. Reference [13] concentrates on the identification of malicious code within Windows executable files through API calls. Utilizing a GAN model, the authors train their classifiers, namely LSTM-Attention and BiLSTM-Attention,

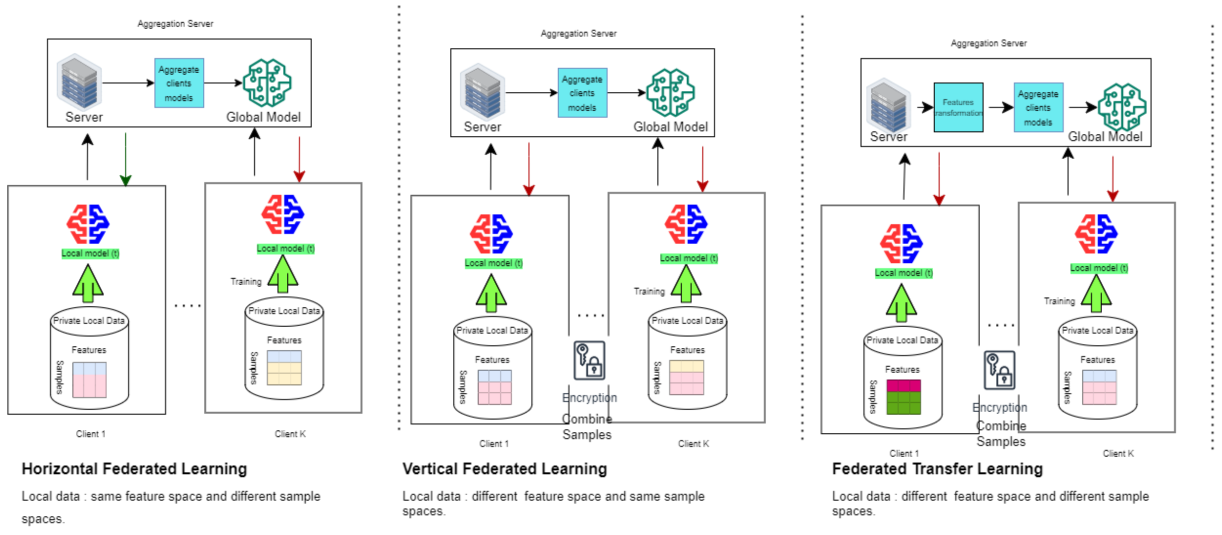

FIGURE 2. **Federated learning types.**

which incorporate contextual and semantic relationships for improved malware detection. Comparing these models against established machine learning classifiers such as Convolutional Neural Networks (CNN), Logistic Regression, Decision Trees, Random Forests, Support Vector Machines, and Multi-Layer Perceptrons, they report outstanding performance, with LSTM-Attention and BiLSTM-Attention achieving 95.43% and 96.53% accuracy, respectively. Reference [37] investigates the effectiveness of deep learning models, namely Generative Adversarial Networks (GANs), Convolutional Neural Networks with three layers (CNN-3L), and Convolutional Neural Networks with four layers (CNN-4L), in the domain of multi-class categorization for intrusion detection. Additionally, [39] propose an intrusion detection system based on an improved Generative Adversarial Network (GAN) model, which consists of two parts: a data pre-processing module and a detection module.

- IoT Attack Detection
  Several recent studies have explored the application of Generative Adversarial Networks (GANs) for Intrusion Detection Systems (IDS) in Internet of Things (IoT) environments. In [14], the authors leverage Bi-directional GANs (BiGANs) to train IDS models capable of detecting novel attacks, achieving an F1-score of 99% on the IoT-23 dataset. This demonstrates the effectiveness of BiGANs in handling zero-day attacks, enhancing security for IoT networks. Building on this, [15] introduces a novel IDS that combines a Wasserstein GAN and an Autoencoder, achieving impressive accuracy, precision, recall, and F1-scores using the Bot-IoT dataset. These studies highlight the potential of GAN-based IDS for both centralized and distributed IoT systems.

Furthermore, [16] proposes a hybrid approach combining Self-Organizing Maps (SOMs), Backpropagation neural networks, and GANs for IDS. Their system utilizes SOMs for anomaly detection, Backpropagation for attack classification, and GANs for synthetic data generation to improve accuracy. While achieving high accuracy for DoS and Probe attacks on the NSL-KDD dataset, they acknowledge limitations in detecting R2L and U2R attacks. Their ongoing research focuses on improving detection for these less common attack types. In contrast to the aforementioned GAN-based approaches, [32] explores using a conditional tabular generative adversarial network (CTGAN) to generate synthetic data for IDS. This approach offers a less complex solution suitable for resource-constrained IoT environments. It is important to note that this research, along with the work presented in [36] and [37], investigates alternative techniques for IDS in IoT networks. While these studies offer valuable insights, a more comprehensive comparison is necessary to determine the most effective approach for this domain (refer to Table 1 for a summary).

B. OPTIMIZING GAN WITH GENETIC ALGORITHMS

Table 2 provides an overview of selected research papers that explore the application of evolutionary techniques to enhance neural networks and Generative Adversarial Networks (GANs). These papers address various aspects of evolution, such as evolving connection weights, adapting network architectures, optimizing noise vectors for GAN generators.

In recent years, the optimization of neural network parameters has gained significant attention due to its impact on improving model performance. Various

TABLE 1. Summary of GAN-Based Intrusion Detection System (IDS) applications.

Ref	Year	Topic	Model	Dataset	Metric	Value
[11]	2021	Malware Detection	Mal-IAGAN	comprising VirusShare APK Android malware and the BIG-2015	Accuracy	80%
[13]	2021	Detection	BiLSTM-Attention	Windows executable files through API calls	Accuracy	95.43% and 96.53%
[14]	2022	IoT Attack Detection	Bi-directional GANs	IoT-23	F1-score	99%
[15]	2020	IoT Attack Detection	Hybrid GAN and Autoencoder	Bot-IoT	Accuracy, Precision, Recall, F1-scores	97.11%, 99.33%, 97.33%, 98.31%
[16]	2023	IoT Attack Detection	Hybrid self-organizing map and GANs	NSL-KDD	Accuracy (DoS), Accuracy (Probe), Accuracy (R2L and U2R)	99.0%, 99.3%, 70.9%
[32]	2023	IoT Attack Detection	Conditional tabular generative adversarial network	MQTT-IoT-IDS2020	Accuracy	98.0%
[35]	2024	SDN-IoT Attack Detection	DNN + SVM	IoMT	-	-
[37]	2024	Network Traffic Detection	CNN-3L + CNN-4L + GANs	CSE-CIC-IDS2018	Accuracy	93%, 99.71%, 100%
[39]	2024	In-Vehicle CAN-FD Bus ID Detection	GANs	normal ID image	Accuracy	98.9%

TABLE 2. Summary of papers on neural network and GAN evolution.

Ref	Year	Approach	Network Type	Dataset Used	Main Contributions	Performance Metrics
[17]	2023	Evolution of Connection Weights	BP Neural Network	Soil Nutrient Time Series	Improved genetic algorithm (IGA) optimization for BP model, Strong generalization ability	Coefficient of determination, RMSE
[18]	2017	Evolution of Network Architectures	Various architectures	CIFAR-10	Automated architecture discovery, Grid computing solution for parallel competitions	Validation accuracy
[19]	2023	Genetic Algorithm for Latent Noise	GAN	Horse, Face, Artwork, Cat datasets	Latent noise optimization for GAN generator, Improved data quality	Data quality comparison
[20]	2022	GA-based GAN Parameter Optimization	GAN	Kaggle 'Give Me Some Credit' dataset	Genetic algorithm for GAN parameter tuning, Enhanced performance	Performance comparison, Quality of generated samples
[36]	2024	IGGAN Complete information	GAN	CIFAR-10	IGGAN with one generator and two discriminators achieves higher Inception Score (IS) and lower Frechet Inception Distance (FID)	Inception Score (IS), Frechet Inception Distance (FID)

experiments utilizing genetic algorithms have been conducted to fine-tune neural networks, and these approaches have yielded notable outcomes. One common approach involves the evolution of connection weights, where genetic algorithms are employed to globally adjust initial weights and biases. This technique aims to expedite network convergence. For instance, in [17], an improved genetic algorithm (IGA) optimized a Back-propagation (BP) neural network model, resulting in highly accurate predictions of soil nutrient time series. The IGA iteratively adjusted network weights and thresholds, significantly reducing root-mean-square errors and exhibiting strong generalization capabilities. Another approach is network architecture evolution, wherein genetic algorithms adapt the neural network's topology to enhance learning and generalization abilities. In [18], an evolutionary process was introduced to automatically discover network architectures. This method utilized a grid computing solution for parallel competitions among trained architectures, preventing idle time. The evolutionary process enabled the discovery of efficient neural network structures tailored to specific tasks. While many of these techniques can be applied to various types of neural networks, recent studies have shown a specific focus on Generative Adversarial Networks (GANs). In [19], an evolutionary algorithm was used to select the best latent noise vector for the GAN's generator, resulting in improved data quality without altering the training process. Furthermore, in [20], a genetic approach was applied to optimize GAN parameters, demonstrating superior performance compared to standard random search methods. The authors suggested that refining data preprocessing techniques, such as approximating data to a Gaussian-like distribution, could further enhance GAN training stability. These diverse approaches highlight the evolving landscape of using genetic algorithms to optimize neural networks, with a growing emphasis on their application to GANs. Moreover, innovative methods for enhancing GANs through genetic algorithms offer exciting possibilities for improving data generation and overall network performance.

C. OPTIMIZED FEDERATED LEARNING AND GAN

Federated learning (FL) has emerged as a powerful paradigm for training machine learning models in decentralized environments. This approach addresses challenges related to bandwidth, privacy, and legal constraints that often restrict the centralized collection of data for analysis and model training [21]. The core idea of FL is to enable model training directly on local data residing on Internet of Things (IoT) devices, thereby eliminating the need to transfer sensitive data to central servers [21]. Since its introduction by Google in 2016, FL has garnered significant attention from researchers seeking to overcome the hurdles associated with training centralized machine learning models on distributed data. Several studies have explored the versatile applications of FL across various domains. One such study [22] proposes a FL method that leverages co-training and Generative Adversarial Networks (GANs). This approach empowers each client device to independently design and participate in FL training without sharing model architecture or parameter information, fostering collaboration while preserving data privacy [22]. Another innovative FL framework, referred to as "Collaborated Game Parallel Learning" (CAP), was introduced in [23]. CAP supports parallel training of both data and GAN models, breaking the isolation that existed among generators in previous distributed algorithms, fostering collaborative learning between cloud servers, edge devices, and edge servers [23]. To address Non-IID (Non-Independently and Identically Distributed) data challenges, the authors propose a "Mix-Generator" module (Mix-G) that enhances CAP-GAN's performance on highly personalized datasets by extracting generic and personalized features [23]. Furthermore, the work in [24] presents "Partial Sharing Federated GAN" (PS-FedGAN), a novel FL framework that minimizes GAN model sharing while addressing heterogeneous data distributions across clients. PS-FedGAN enhances GAN release and training mechanisms, leading to improved privacy preservation at reduced communication costs, especially in wireless network scenarios [24]. Comprehensive analysis underscores the convergence and privacy benefits of the proposed PS-FedGAN framework, marking a significant advancement in FL research [24]. Building upon these advancements, the work in [33] explores federated learning with GAN-based data synthesis. Each client pretrains a local GAN to generate differentially private synthetic data, which are then uploaded to a parameter server (PS) to construct a global shared synthetic dataset [33]. FLIGAN [38], addresses the issue of data incompleteness in FL. We leverage Generative Adversarial Networks (GANs) to capture complex data distributions and generate synthetic data that closely resemble real-world data. This synthetic data is then used to enhance the robustness and completeness of datasets across nodes. Our methodology adheres to FL's privacy requirements by generating synthetic data in a federated manner without sharing the actual data in the process [38]. We incorporate techniques such as classwise sampling and node grouping, designed to improve the federated GAN's performance, enabling the creation of high-quality synthetic datasets and facilitating efficient FL training. Empirical results from our experiments demonstrate that FLIGAN significantly improves model accuracy, especially in scenarios with high class imbalances, achieving up to a 20% increase in model accuracy over traditional FL baselines [38].
Table 3 allows for a straightforward comparison of FL-based GAN training approaches, highlighting their unique characteristics and performance outcomes

IV. METHODOLOGY

The use of GANs has emerged as a powerful technique in data augmentation, particularly in domains with limited or imbalanced datasets, such as IoT-based intrusion detection. GANs excel at generating synthetic data that mimics the characteristics of real-world data, thereby enhancing the training process for machine learning models. This augmentation process not only enriches the dataset by generating diverse samples, but it also plays a crucial role in addressing class imbalance, a common challenge in IDS. By introducing more synthetic attack instances in our dataset, we achieve a more balanced distribution between attack and normal samples, ultimately improving the model's ability to detect intrusions accurately.

Figure 3 presents our methodology, summarizing the key components of our approach, including WGAN-GP for synthetic data generation, Genetic Algorithm (GA) for hyperparameter optimization, and Federated Learning for privacy-preserving model training.

Our approach leverages Wasserstein Generative Adversarial Network with Gradient Penalty (WGAN-GP) to generate high-quality synthetic attack data. The methodology consists of the following key steps:

1) **Training the Generator**: The WGAN-GP model is trained on existing attack data to learn the underlying patterns of different intrusion types.
2) **Optimization with Genetic Algorithms (GA)**: Hyperparameters of the GAN model, such as learning rate, batch size, and training epochs, are optimized using GA to ensure better sample diversity and stability.
3) **Federated Learning for Data Privacy**: The synthetic data generation is distributed across multiple nodes using federated learning, ensuring that data privacy is preserved while improving the generalizability of attack patterns.

TABLE 3. Comparison of federated learning approaches for GAN training.

Ref	Year	FL Approach	GAN Training Method	Privacy Preservation	Communication Efficiency	Experimental Results
[22]	2022	Co-training	DCGAN	Strong	Low	Improved FID score
[23]	2023	CAP-GAN	Improved GAN	Strong	Enhanced	Enhanced image quality
[33]	2022	SDA-FL	GAN	Strong	Reduced	Enhanced privacy
[24]	2023	PS-FedGAN	Wasserstein GAN	Strong	Reduced	Enhanced privacy
[38]	2024	FLIGAN:	GAN	Strong	Reduced	Improved accuracy

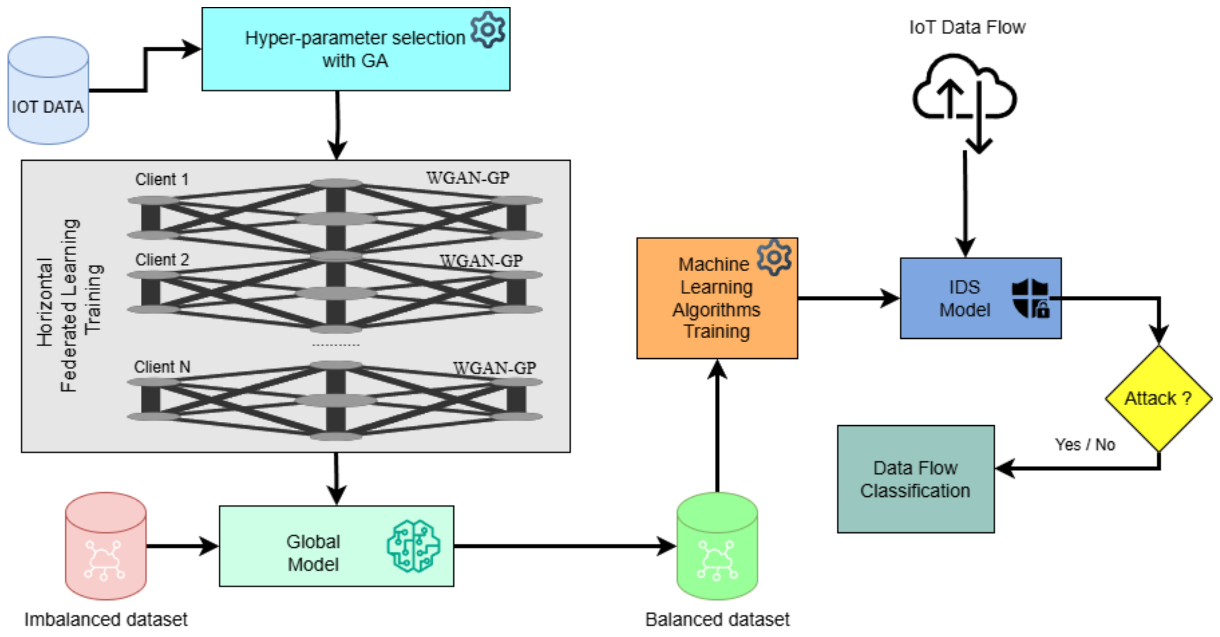

FIGURE 3. WGAN-GP training to enhancing IoT intrusion detection.

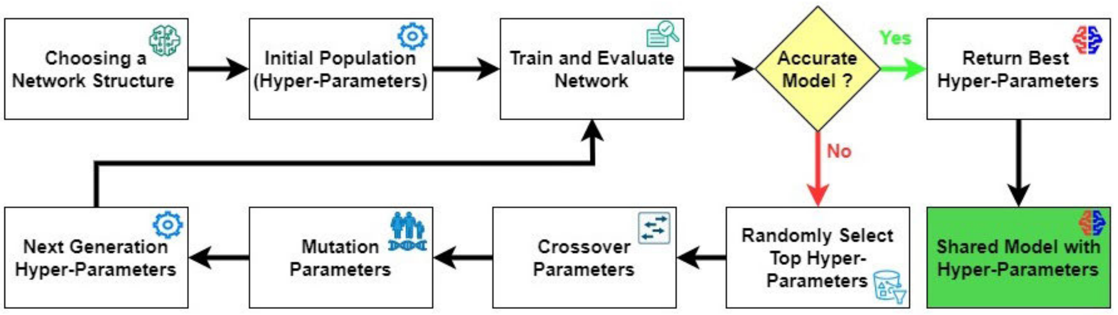
Choosing a Network Structure
Initial Population (Hyper-Parameters)
Train and Evaluate Network
Accurate Model ?
Yes
No
Return Best Hyper-Parameters
Shared Model with Hyper-Parameters
Randomly Select Top Hyper-Parameters
Crossover Parameters
Mutation Parameters
Next Generation Hyper-Parameters

FIGURE 4. Hyper-parameter selection with Genetic Algorithm (GA).

4) **Validation and Quality Assessment**: The generated attack data is evaluated using statistical metrics, including distribution similarity measures (Kullback-Leibler divergence) and correlation analysis with real data.

To optimize the performance of WGAN-GP, we leverage GA for hyper-parameter selection, enabling the model to generate data more effectively. Additionally, federated training across multiple clients allows for secure and efficient learning while preserving the privacy of local datasets. The following subsections detail our approach to hyper-parameter selection using WGAN-GP with GA and the federated training strategy implemented across multiple clients.

A. HYPER-PARAMETERS SELECTION

In the initial phase of our methodology, we enhance the capabilities of WGAN through a rigorously designed hyper-parameter optimization process utilizing a GA. This phase is pivotal as it aims to fine-tune critical parameters essential for the optimal performance of WGAN (Generator and Critic). By concentrating on metrics such as learning rate, batch size, training epochs, optimizer, and `n_critic`, we seek to markedly enhance WGAN's ability to capture the intricate patterns and complexities inherent in real-world IoT Cybersecurity data. Hyper-parameter optimization is crucial for ensuring that the WGAN model operates with maximum efficiency and effectiveness. The learning rate, for instance, determines the step size during the training process, impacting the model's convergence and stability. Similarly, batch size affects the number of samples processed in each iteration, influencing model training dynamics and memory usage. Furthermore, the selection of optimizers significantly affects the model's ability to learn and generate realistic data samples. By employing a GA, we systematically explore the hyper-parameter space to identify optimal configurations that maximize WGAN's performance.

This approach entails creating a population of potential solutions (parameter configurations), assessing their suitability based on predefined criteria (such as the quality of generated samples), and iterative refinement through genetic operators like selection, crossover, and mutation. The primary objective of this phase is to leverage the power of Genetic Algorithms (GA) to fine-tune WGAN's hyper-parameters, enabling the model to effectively capture the nuances and complexities of IoT Cybersecurity data. Through meticulous optimization, we aim to enhance the performance and efficiency of WGANs, laying a robust foundation for subsequent phases of our methodology.

A crucial aspect of our methodology is the encoding of individuals within our evolutionary algorithm, where the complex details of hyper-parameters are meticulously structured for optimization. This process involves representing six key hyper-parameters, each playing a distinct role in shaping the behavior and effectiveness of the WGAN model. These hyperparameters include batch size, learning rate, optimizer (beta 1 and beta 2), n_critic training count (which dictates the frequency of critic training relative to generator training), and batch size (a critical variable in the loss function).

For instance, the batch size dictates the number of data samples processed in each iteration, directly impacting the model's training dynamics and computational efficiency. The learning rate, on the other hand, determines the magnitude of parameter updates during optimization, influencing both the convergence speed and stability of the model. beta 1 and beta 2 are specific to the optimizer, controlling the exponential decay rates of past gradients and squared gradients, respectively, thereby affecting the optimization trajectory and convergence behavior. The critic training count parameter is pivotal in balancing the training dynamics between the critic and generator networks within the WGAN framework. Adjusting this parameter effectively regulates the learning dynamics, ensuring a harmonious training process between the two networks. Furthermore, the training epochs parameter, which defines the total number of iterations the model undergoes during training, plays a crucial role in determining the model's convergence and overall performance, affecting its ability to generalize and produce high-quality outputs.

TABLE 4. Chosen Operators.

Genetic Operator	Method
Selection	Roulette Wheel
Crossover	Single Point
Mutation	Real Number Mutation

An example representation of an individual in our coding scheme is as follows: $[0.0001, 64, 10000, 0.5, 0.9, 5]$. This straightforward coding approach was meticulously chosen for its simplicity and efficiency, despite considering alternative coding schemes. Through rigorous experimentation and analysis, we determined that alternative coding methodologies introduce unnecessary complexity without offering significant benefits. Therefore, the chosen coding scheme reflects our commitment to simplicity while ensuring effective hyper-parameter optimization for WGAN. Specifically, the Learning Rate has an initial value of 0.0001 with a range of $[1 \times 10^{-5}, 1 \times 10^{-3}]$, the Batch Size has an initial value of 64 with a range of $[32, 512]$ (typically, powers of 2 are preferred), the Training Epochs have an initial value of 10000 (or more, depending on the dataset and computational resources) with a range of $[1000, 50000]$, the Optimizer is initially set to Adam with hyperparameters β_1 having an initial value of 0.5 and a range of $[0.5, 0.9]$ and β_2 having an initial value of 0.9 and a range of $[0.9, 0.999]$, and the Number of Critic Updates per Generator Update (n_critic) has an initial value of 5 with a range of $[1, 10]$.

In our methodology, genetic operators form the backbone of our approach, guiding the evolution of solutions within the GA framework. Although we prioritize simplicity to avoid unnecessary complexity, each operator is meticulously designed to ensure efficient exploration and exploitation of the solution space.

- **Selection Operator**: Employing the roulette method, this operator strategically selects individuals based on their fitness scores, simulating the principles of natural selection. The roulette method, also known as fitness proportionate selection, is a technique where each individual is given a probability of selection proportional to its fitness. In the context of hyper-parameter selection, this approach ensures that individuals with higher fitness scores have a greater chance of being chosen for the next generation, promoting the survival of more optimal solutions.
- **Crossover Operator**: Using the single-point method, this operator facilitates the exchange of genetic information between selected individuals, thereby promoting population diversity. The single-point method, also known as single-point crossover, involves splitting two parent solutions at a randomly chosen point and exchanging segments to create offspring. This technique introduces variation in the population by combining different parts of the parent solutions, which helps

maintain genetic diversity and improves the search for optimal solutions.

- **Mutation Operator**: Utilizing Real Number Mutation, this operator introduces randomness by modifying individual genes within a predefined range, ensuring a balance between exploration and exploitation. The table below concisely summarizes the essence of our chosen genetic operators and their crucial roles in the hyper-parameter optimization of WGAN:

To uphold the quality of solutions across generations, our process of constructing the new population incorporates elitism, allowing the best-performing organism to remain unchanged in the next generation. Selecting a fitness function was a nuanced decision, given the absence of a standardized method for evaluating overall sample quality in tabular data scenarios. After exploring various options, we opted for the Kullback-Leibler (KL) divergence, a widely recognized metric in research. This divergence quantifies the dissimilarity between two probability distributions, specifically between the distribution of real and synthesized data samples. Kullback-Leibler (KL) divergence is a key statistical measure used to quantify the difference between the distribution of generated data and the real data. A KL divergence value close to zero indicates that the two distributions are similar, meaning that the synthetic data effectively mimics real data. In our experiments, KL divergence is used to assess the quality of generated samples by measuring how well they match the original dataset. Calculated following 300 iterations of GAN training, this divergence provides a balanced compromise between speed and measurement quality, ensuring that the generated data retains the statistical properties of the real dataset while optimizing computational efficiency.

$$DKL(P||Q) = \sum_{x} P(x) \log(\frac{P(x)}{Q(x)}) \tag{2}$$

The KL divergence, denoted as $D_{KL}(P||Q)$, is a statistical measure that quantifies the difference between two probability distributions: P (the true distribution) and Q (a reference distribution). Simply put, it represents the additional surprise expected by using Q as a model instead of P when the actual data follows P. Although the KL divergence reflects the disparity between two distributions, it is technically not a measure in the strictest sense. Unlike the more familiar concept of distance, the KL divergence is not symmetric (i.e., $D_{KL}(P||Q)$) is not necessarily equal to $D_{KL}(Q||P)$) and does not satisfy the triangular inequality. However, within information geometry, the KL divergence can be considered a type of divergence, a generalization of distance squared. For specific distribution classes, such as the exponential family, it even satisfies a generalized form of the Pythagorean theorem that applies to the squares of distances.

The algorithm proceeds through several stages, beginning with the initialization of the initial population by generating random individuals (hyper-parameter sets) and evaluating their quality based on the fitness function. Once the initial population is established, iterative generations are executed according to predefined parameters. In each generation, a series of genetic operators is applied sequentially. The first step involves selecting pairs of individuals for the crossover and mutation stages using a roulette selection technique. Subsequently, the crossover and mutation steps are performed, with adjustable probabilities to control their frequency. Following these operations, the fitness scores of the new individuals are recalculated. After each generation, the original population is replaced by new individuals, with elitism ensuring that the highest-quality solutions are preserved, thus maintaining the overall quality of the solutions throughout the evolutionary process. Elitism is an important mechanism in genetic algorithms that ensures the best solutions from one generation are carried over to the next. In our approach, we employ elitism to retain the top-performing hyperparameter configurations based on KL divergence scores. This helps prevent the loss of high-quality solutions and contributes to the stability and efficiency of the optimization process.

Algorithm 1 Genetic Algorithm

```
Require: Batch Size, Learning Rate, Beta 1, Beta 2, Number
    of Critic Training, Weight of Gradient Penalty.
1: Initialize the first population
2: while $N \neq 0$ do
3:     if the generation has reached the maximum popula-
           tion then
4:         Select the individuals via roulette wheel.
5:         Apply Crossover with probability 0.5.
6:         Apply Mutation with probability 0.1.
7:     end if
8: end while
9: Return the best solution for (Batch Size, Learning Rate,
       Beta 1, Beta 2, Number of Critic Training, Weight of
       Gradient Penalty)
```

B. TRAINING PROCESS OF FEDERATED GAN MODEL

In the second phase of our research methodology, we delve into the intricate training process of a deep federated intrusion detection system (IDS) model. This phase is pivotal for enhancing the security of Internet of Things (IoT) ecosystems. Building upon existing research, we introduce a pseudo-algorithm inspired by the Federated Averaging Algorithm. Our goal is to elucidate the steps involved in training multiple client sets within the federated learning framework, specifically emphasizing the horizontal federated learning approach.

1) **Client Selection and Data Distribution** The federated learning process begins with the aggregation server strategically selecting a fraction C of K clients to participate. These clients represent diverse data sources within the IoT ecosystem. The goal is to ensure that the selected clients collectively cover the entire data distribution relevant to IoT security.

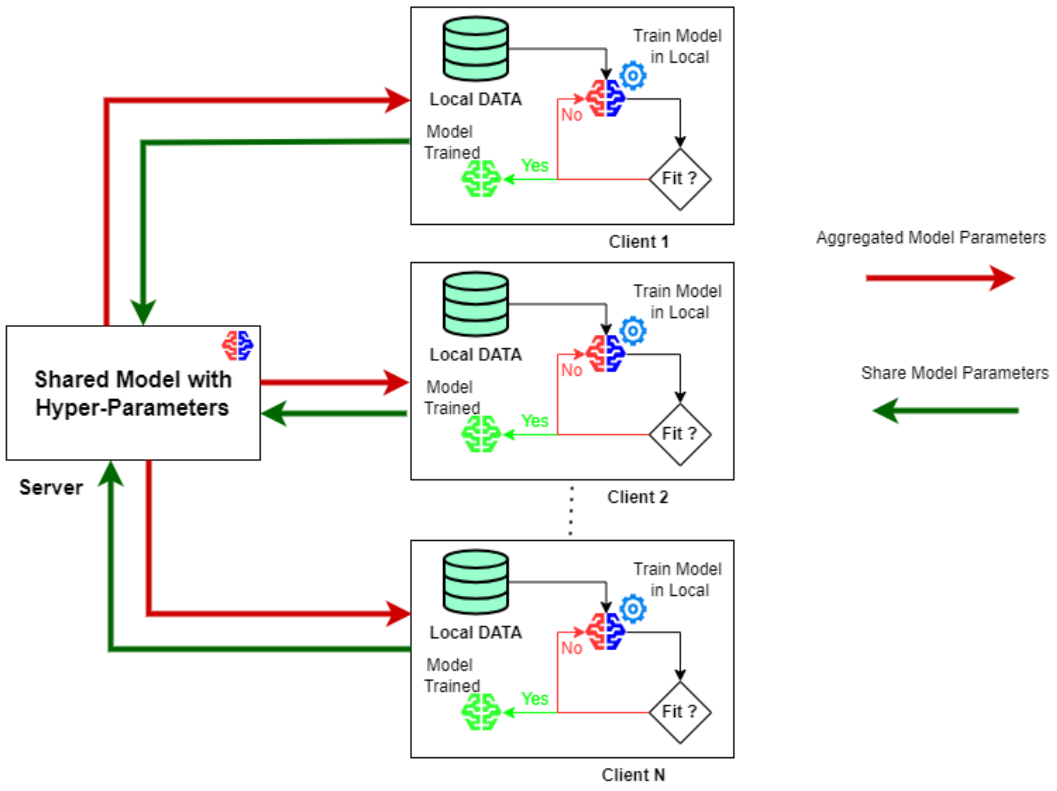

FIGURE 5. Training WGAN-GP with federated learning.

2) **Model Initialization** Upon client selection, the aggregation server initializes a generic model with random weights. This model serves as the starting point for the federated learning process. Common model architectures include deep neural networks (DNNs) or convolutional neural networks (CNNs).
3) **Client Training and Local Updates** Each selected client retrieves the generic model from the aggregation server. Independently, clients train the model using their locally held data. Training involves multiple iterations (epochs) to improve the model's performance. During training, clients update model parameters (weights and biases) based on their local data.
4) **Aggregation and Global Model Update** After local training, clients share their updated model parameters with the aggregation server. Aggregation techniques (e.g., Federated Averaging) combine these parameters to create a global model. The global model benefits from knowledge learned by individual clients without exposing raw data.
5) **Iterative Process and Convergence** The iterative process continues over multiple rounds (typically denoted by R federated learning cycles). Clients receive the updated global model, retrain locally, and contribute further improvements. This collaborative learning fosters knowledge sharing among participating clients. Convergence occurs when the global model stabilizes and achieves satisfactory performance.

Algorithm 2 Federated Averaging Algorithm

```
Require: Batch Size, Learning Rate, Beta 1, Beta 2, Number
    of Critic Training, Weight of Gradient Penalty.
 1: Server (K, C, R)
 2: $W_G$, $W_C$ ← Generic Model()
 3: while $t \neq R$ do
 4:     if the generation has reached the maximum population then
 5:         Select the individuals via roulette wheel.
 6:         Apply Crossover with probability 0.5.
 7:         Apply Mutation with probability 0.1.
 8:     end if
 9: end while
10: Return the best solution for (Batch Size, Learning Rate,
        Beta 1, Beta 2, Number of Critic Training, Weight of
        Gradient Penalty)
```

6) **Privacy and Confidentiality** Horizontal federated learning preserves data privacy. Only model updates (parameters) are exchanged, ensuring sensitive information remains on clients' devices. Techniques like differential privacy can enhance client privacy further.

V. EXPERIMENTATION

Evaluation of GANs, particularly for tabular data, remains a challenging task due to the absence of a universally accepted evaluation methodology. This section presents the experimental design employed for our method (named FedGAGAN). We will detail the chosen solution for FL alongside a comprehensive discussion of the metrics used to analyze the model's performance. For our experiments, we leveraged Google Colab, utilizing established libraries such as NumPy, Pandas, TensorFlow, and Keras. While several open-source federated learning frameworks exist for simulating and experimenting with these algorithms, we opted for the Federated Learning and Differential Privacy (FL&DP) framework by Sherpa.AI. This choice is based on the framework's distinct advantages.

A. DATASETS

we draw upon several critical datasets to advance intrusion detection and enhance network security. Let's delve into the specifics of each dataset and their inherent challenges, particularly regarding data imbalance.

- **UNSW-NB15** dataset captures real-world network traffic, encompassing features such as source and destination IP addresses, port numbers, protocol types, and payload sizes. While it covers an extensive array of attacks—ranging from denial-of-service (DoS), reconnaissance, and exploitation to more complex cyber threats—there is a significant imbalance between benign and malicious traffic. For example, normal traffic vastly outweighs the number of attack instances, making it difficult for traditional machine learning models to accurately detect rare attacks. This imbalance poses a considerable challenge, as models tend to favor the majority class (normal traffic) and overlook critical but infrequent attack patterns. Addressing this requires innovative data augmentation techniques, such as those proposed in this paper, to generate synthetic attack data and balance the dataset, ultimately improving the accuracy of intrusion detection systems (IDS).
- **IoT-23** dataset offers another prime example of data imbalance. It comprises twenty-three distinct IoT network traffic scenarios, including twenty captures from infected IoT devices, each associated with a specific malware sample, and three captures of real IoT devices' network traffic. While the dataset is comprehensive in representing IoT-specific attacks, the prevalence of benign traffic compared to malicious IoT activity highlights the need for augmented data generation to enhance model training and detection capabilities. Furthermore, the dataset's small sample size of malicious scenarios limits the performance of intrusion detection models, again underscoring the value of generating synthetic attack samples to improve detection accuracy.
- **CSE-CIC-IDS2018** dataset provides detailed profiles for applications, protocols, and network entities, allowing researchers to explore a wide range of attacks, including botnet activity, web application attacks, and port scanning. Despite its extensive coverage, the dataset also exhibits a skewed distribution toward normal traffic. The imbalanced representation of attacks versus legitimate traffic introduces bias into detection models, making them less effective in identifying rare but critical anomalies. Synthetic data generation, like that proposed in this paper using WGAN-GP, plays a crucial role in overcoming these imbalances by augmenting the minority class and improving model robustness.
- **MQTT-IoT-IDS2020** dataset simulates an MQTT-based network, also suffers from imbalanced data. It comprises five scenarios, including normal operation, aggressive scans, UDP scans, Sparta SSH brute-force attacks, and MQTT brute-force attacks. Given that the number of benign records far exceeds malicious instances, traditional IDS approaches struggle to generalize well on attack detection. By applying generative models like WGAN-GP, this paper proposes a solution to generate synthetic attack traffic, thereby balancing the dataset and enhancing the ability of IDS models to recognize even subtle attacks.

The imbalance across IoT Intrusion Detection System (IDS) datasets—UNSW-NB15, IoT-23, CSE-CIC-IDS2018, and MQTT-IoT-IDS2020—poses a significant challenge to accurate anomaly detection. To address this issue, we employed advanced generative techniques, specifically WGAN-GP, along with genetic algorithms (GA) for hyperparameter selection and federated learning for decentralized training. By generating synthetic attack data, we not only balanced these datasets but also enhanced the diversity of training samples, improving the performance of machine learning models in detecting both common and rare intrusions. For our experiments, we created 15,000 new attack samples for UNSW-NB15, 12,500 for IoT-23, 18,000 for CSE-CIC-IDS2018, and 10,000 for MQTT-IoT-IDS2020, resulting in total dataset sizes of 450,000, 320,000, 500,000, and 250,000 rows, respectively. These synthetic samples were integrated into the training sets to better represent minority classes, demonstrating the critical role of data augmentation in building more effective and reliable intrusion detection systems for modern and complex networks.

B. DATA PREPARATION

To conduct our experiments, we used four prominent IoT Intrusion Detection System (IDS) datasets: UNSW-NB15, IoT-23, CSE-CIC-IDS2018, and MQTT-IoT-IDS2020. Each dataset has unique characteristics and offers a wide range of attack types, making them suitable for evaluating the efficacy of our data augmentation approach. In UNSW-NB15 dataset, we began by normalizing numerical features and one-hot encoding categorical features. Missing values were handled using mean imputation for numerical features and mode imputation for categorical features.IoT-23Raw network traffic data was parsed to extract relevant features like packet sizes, inter-arrival times, and protocol information. Non-numeric features were encoded using one-hot encoding, and all features were scaled to a range of [0, 1].CSE-CIC-

IDS2018 The dataset required extensive feature extraction to convert raw network flow data into a format suitable for machine learning. Features were then standardized, and redundant or highly correlated features were removed to improve the efficiency of the training process. Since MQTT traffic has specific characteristics, we extracted MQTT protocol-specific features such as message types, topic names, and message lengths. These features were normalized, and categorical features were one-hot encoded.

C. WGAN-GP HYPERPARAMETER

Since we used a simple WGAN, our system had just two Neural Networks. These networks called the generator and critic, both had layers made for tabular data. We didn't add many layers to keep things simple and avoid overdoing it. The Generator, shown in the figure below, has four dense layers. It takes in a vector of 32 (which represents noise) and produces a vector of size 49, matching the number of columns in the dataset. All these layers use a 'relu' activation function. You can adjust the layer sizes to suit your data. Surprisingly, the discriminator has the same number of layers as the generator, but with one difference: we included Dropouts to slow down its learning process. This is because the discriminator is usually trained more times than the generator. We tried various parameters to find the best balance between

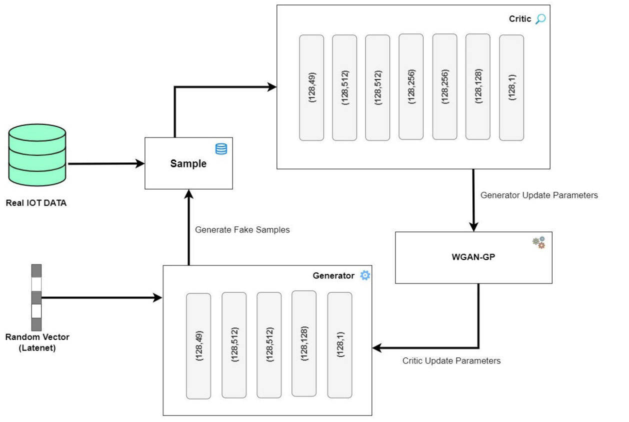

FIGURE 6. Proposed WGAN architecture.

experiment quality and time. After experimenting, we settled on the Population size of 250, Crossover probability: 0.60, Mutation probability: 0.10 and Number of Generations: 20. the academic community advice to using a lower percentage for mutation and a higher one for crossover. we considered implementing a decay rate for these probabilities. With a decay rate in place, the percentages would decrease as the generations progressed. This approach would encourage the algorithm to explore the search space more thoroughly at the beginning and, as it identifies different local optima, it would then focus more closely on those regions. However, we had constraints related to population size and the number of generations. As these values increased, the computational cost also went up. Therefore, we couldn't simply increase these values, even though doing so might have improved the search when using higher operator probabilities.

D. EVALUATION METRICS

Evaluating the quality of generated data remains a challenging endeavor within the realm of data synthesis, as the diversity of data types and objectives precludes the existence of a universally accepted assessment framework [31]. Unlike fitness functions that are tailored to specific optimization tasks, a one-size-fits-all solution for dataset quality assessment remains elusive. As such, the evaluation of dataset quality necessitates a multi-faceted approach that draws upon various metrics, each shedding light on distinct aspects of the data. While the selection of additional metrics may be contingent on the specific goals and context of data synthesis, our primary focus lies in quantifying the similarity between the generated datasets and the original data. This emphasis underscores the significance of comprehensive evaluation within our research framework. The suite of metrics employed encompasses:

- Visualizations:
  such as Means and Standard Deviations (STDs) of real and fake data and cumulative sums per feature, play a crucial role in the assessment of synthetic data quality and the comparison between real and synthetic datasets. The means and STDs plots provide insights into the central tendencies and variabilities of data in each feature, facilitating the identification of differences between real and synthetic datasets in terms of statistical properties. On the other hand, cumulative sum plots enable the visualization of data accumulation over time or across data points, aiding in the detection of temporal patterns and shifts. These visual tools offer a more intuitive and comprehensive understanding of dataset characteristics, helping researchers and practitioners pinpoint areas where synthetic data generation processes may need refinement to achieve a closer match to the original data's statistical and temporal properties.
- Privacy:
  These metrics address the privacy implications of synthetic data generation. Duplicate rows between the real and fake datasets reveal any information leakage. Additionally, the nearest neighbor mean and standard deviation indicate the proximity of data points in the synthetic dataset to their nearest neighbors in the real dataset, providing insights into data privacy preservation.

$$\text{Neighbor Mean} = \frac{\sum_{i=1}^{n} d_i}{n} \tag{3}$$

  Distance to Nearest Neighbor for each data point divide on the Total number of data points. As shown at the bottom of the next page.
- Correlation:
  This category encompasses metrics related to the dataset's structure and characteristics. The Column

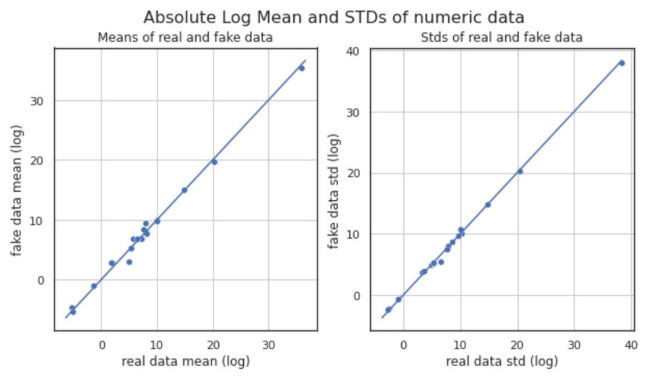

FIGURE 7. Absolute log mean and STDs of numeric data.

Correlation Distance RMSE and MAE quantify the dissimilarity in column correlations between the real and fake datasets. These metrics shed light on the preservation of statistical relationships.

Column Correlation Distance RMSE:

$$RMSE = \sqrt{\frac{\sum((\text{Correlation(real)} - \text{Correlation(fake)})^2)}{\text{Number of column pairs}}} \tag{4}$$

Column Correlation Distance MAE:

$$MAE = \frac{\sum |\text{Correlation(real)} - \text{Correlation(fake)}|}{\text{Number of column pairs}} \tag{5}$$

- Statistics:
  These metrics offer a holistic view of the synthetic dataset's quality. Basic statistics provide an aggregate assessment of various aspects of the dataset. Correlation column correlations and Mean Correlation between fake and real columns offer insights into the congruence of statistical relationships. The 1 - MAPE Estimator results gauge the accuracy of percentage error estimation. Finally, the Similarity Score encapsulates the overall dataset congruence, offering a comprehensive measure of similarity between the synthetic and real datasets.
- Machine Learning classification:
  This category provides an assessment of the synthetic data's utility for classification tasks. F1-scores, computed for both real and fake datasets using classifiers such as Decision Tree, Logistic Regression, and MLP, offer a measure of the datasets' suitability for machine learning models.

These diverse metrics collectively enable a multifaceted evaluation, ensuring a comprehensive understanding of the synthetic dataset's quality, utility, and privacy characteristics.

TABLE 5. Data generated statistics.

Metric	Value
Duplicate rows between sets (real/fake)	0
Nearest neighbor mean	1.3747
Nearest neighbor STD	1.2956
Column Correlation Distance RMSE	0.8887
Column Correlation Distance MAE	0.0500
Correlation Column Correlations	0.9212
Mean Correlation between fake and real columns	0.8994
1- MAPE Estimator results	0.9487
Similarity Score	0.9543
Basic statistics	0.9876

VI. RESULTS AND DISCUSSION

In this section, we present the outcomes achieved using the optimal parameters derived from the genetic algorithm and federated learning. Initially, we elucidate the metrics selected to gauge the quality of the generated data and elucidate the rationale behind our metric choices. Subsequently, we showcase the performance results by juxtaposing the synthetic data with the original dataset. The performance of the dataset with the best parameters is evaluated in comparison to a dataset employing state-of-the-art parameters, facilitating a comprehensive assessment of our approach's efficacy.

Upon analyzing the metrics generated by *TableEvaluator*, it becomes evident that the real and fake data exhibit a striking degree of similarity. The means and standard deviations of numeric data, as presented in Figure 7, which are key indicators of data distribution and dispersion, closely align between the two datasets. This alignment is reflected in the minimal spread of values, underscoring the close resemblance in statistical characteristics. Furthermore, the *Cumulative Sums per Feature* plot provides valuable insights into the data's cumulative behavior. In this regard, both the real and fake datasets follow a consistent and nearly overlapping trajectory, demonstrating that the cumulative sum of differences between corresponding values in each feature remains notably small and stable. This convergence in cumulative sums across features reinforces the high degree of similarity between the two datasets, as explained in Figure 8.

In addition to these metrics, the results from the PCA and t-SNE visualizations, particularly for the UNSW-NB15 dataset, as shown in Figure 8, further validate the resemblance between the real and synthetic data. Both PCA and t-SNE plots demonstrate that the real and fake data clusters are closely aligned, indicating that the synthetic data generated through WGAN-GP retains the essential variance and patterns of the original dataset. This proves that not only do the statistical properties align, but also the overall structure of the data in the feature space remains consistent between the real and generated samples.

$$\left(\frac{(\text{Distance to Nearest Neighbor for each data point} - \text{Nearest Neighbor Mean})^2}{\text{Total number of data points}}\right)$$

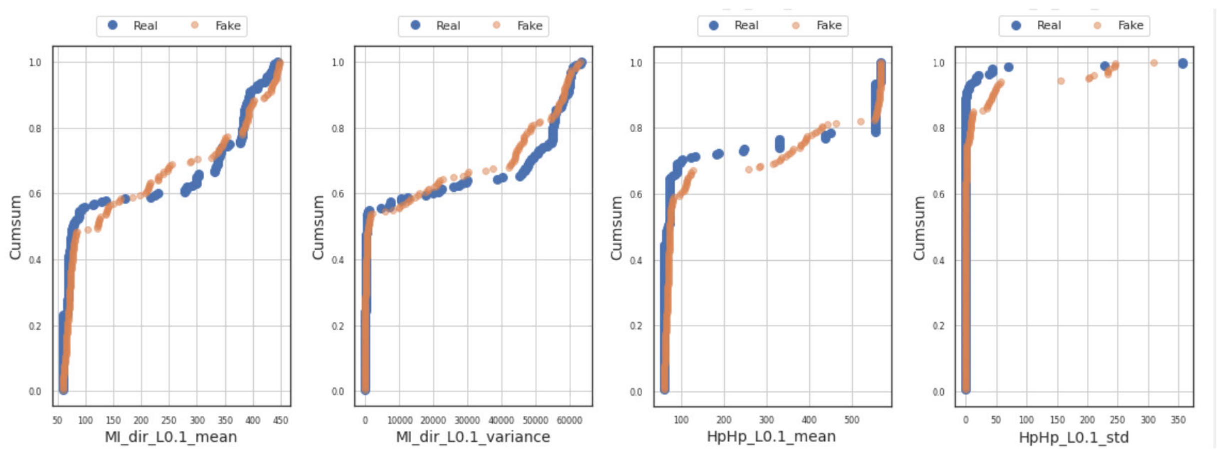

FIGURE 8. Cumulative sums per feature.

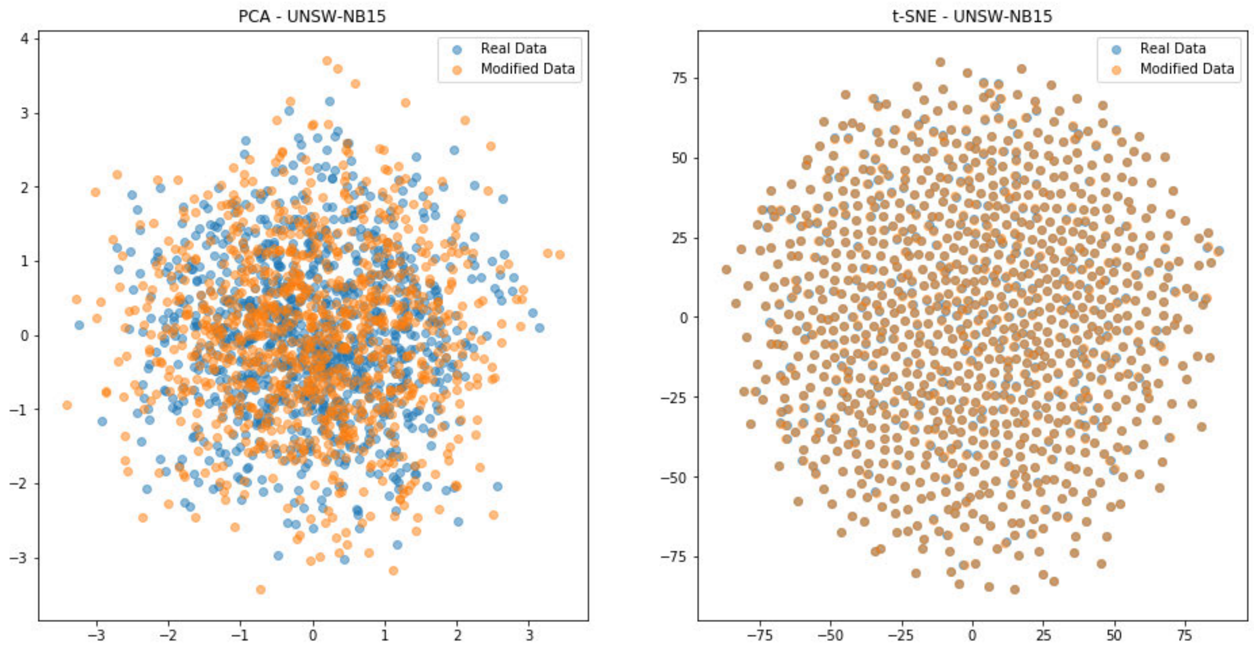

FIGURE 9. PCA and t-SNE for UNSW-NB15 dataset.

Collectively, these visualizations and metrics serve as compelling evidence of the effectiveness of the data generation process. They affirm that the synthetic data, generated through WGAN-GP tailored for tabular data, closely approximates the characteristics and statistical properties of real data. This alignment underscores the success of the data generation method in producing synthetic data that faithfully replicates the attributes of the original dataset.

The evaluation metrics for the generated fake data in comparison to the real data using GANs reveal several strengths and a few areas of improvement. Notably, there are no duplicate rows between the real and fake datasets, indicating the successful generation of diverse instances. The nearest neighbor mean and standard deviation (STD) metrics provide insight into the proximity of data points. While the nearest neighbor mean suggests a moderate distance between data points, the STD is relatively low, indicating consistent neighborhood characteristics. However, it's essential to note that the nearest neighbor mean could benefit from further refinement to reduce data point distances. Column correlation metrics, including Column Correlation Distance RMSE and Column Correlation Distance MAE, demonstrate the similarity in statistical relationships between dataset columns. The low values of these metrics signify that the statistical relationships closely align, reflecting a strength in the generated data's ability to capture column correlations. The Correlation column correlations and Mean Correlation between fake and real columns further emphasize the strong

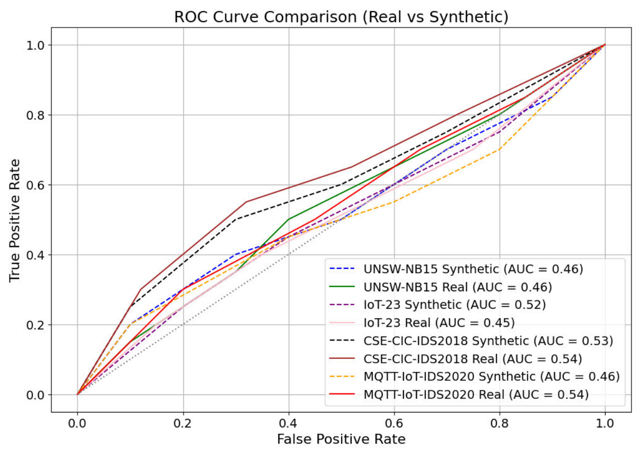

FIGURE 10. ROC curve comparison.

correlation between corresponding columns. This indicates that the synthetic data maintains similar column relationships as the real data, a noteworthy achievement. The 1 - MAPE Estimator results and Similarity Score metrics both exhibit high values, indicating a high degree of similarity between the real and fake datasets.Real Data (UNSW-NB15): Achieves an AUC of 0.873, indicating strong discrimination between normal and attack instances. Generated Data (Resembles UNSW-NB15): Performs even better with an AUC of 0.955, closely mimicking the real data. This suggests that the generated data captures similar patterns, making it effective for simulating UNSW-NB15-like scenarios. These metrics highlight the robustness of the data generation process in replicating the characteristics of the original data. Lastly, the Basic statistics metric reinforces the overall quality of the synthetic data, with a score nearing perfection. While these results underscore the effectiveness of the GAN-based data generation approach, further optimization of the nearest neighbor mean and refinement in data point distances could enhance the overall quality of the generated fake data.

TABLE 6. Impact of fake data in classification.

ML Classification	Real Data	UNSW-NB15	IoT-23	CSE-CIC-IDS 2018	MQTT-IoT-IDS 2020	Hybrid
Decision Tree	97%	96%	95%	93%	97%	99%
Logistic Regression	85%	65%	67%	71%	63%	83%
MLP	92%	89%	88%	87%	89%	93%
Random Forest	99%	99%	98%	99%	97%	99%

The impact of fake data on classification models is a crucial aspect of evaluating the effectiveness of synthetic data generation methods. In our experiments, we compared the classification performance of machine learning models trained on real data, fake data, and a hybrid dataset comprising both real and fake data. This comparison provides insights into how well the generated fake data complements the real data and enhances model performance. For the Decision Tree Classifier, we observed that the model trained on the hybrid dataset achieved the highest accuracy of 99%. This indicates that the inclusion of fake data alongside real data led to improved decision tree model performance compared to using real or fake data alone. The decision tree model trained on fake data still demonstrated strong performance with a 96% accuracy in UNSW-NB15. In the case of Logistic Regression, the model trained on real data achieved an accuracy of 85%, while the model trained on fake data exhibited a lower accuracy between 63% and 71%. However, when combining both real and fake data in the hybrid dataset, the accuracy improved to 83%. This suggests that fake data can be valuable in enhancing the performance of logistic regression models when used in conjunction with real data. The MLP Classifier also benefited from the presence of fake data in the hybrid dataset. The model trained on the hybrid dataset achieved the highest accuracy of 93%, outperforming both the real data (92%) and fake data (88-89%) scenarios. This underscores the advantageous impact of incorporating synthetic data in training neural network models. Lastly, the Random Forest Classifier consistently performed well across all scenarios. Both the model trained on real data and the model trained on fake data achieved high accuracies of 99%. When using the hybrid dataset, the accuracy remained at 99%, indicating that the synthetic data did not significantly affect the model's already excellent performance. In summary, our experiments demonstrate that fake data can have a positive impact on classification model performance, especially when integrated into a hybrid dataset alongside real data. Decision tree and logistic regression models benefited from the inclusion of synthetic data, with the Decision Tree model showing the most substantial improvement according to the results in Table 6. Meanwhile, the MLP Classifier also demonstrated significant enhancement when using the hybrid dataset. Overall, these findings highlight the potential of synthetic data to enhance machine learning model training and classification accuracy. The resulting ROC curves provide insights into the performance of the classifier across different synthetic datasets. The ROC curve for UNSW-NB15 shows a certain level of discriminative power, indicating the classifier's ability to distinguish between classes. The AUC value provides a quantitative measure of this performance. The ROC curve for IoT-23 reflects the classifier's performance on data with a higher number of features. This might affect the classifier's ability to generalize well, as seen in the shape and AUC of the ROC curve. With the highest number of features among the datasets, the ROC curve for CSE-CIC-IDS2018 illustrates the challenges and complexities of dealing with high-dimensional data. The ROC curve for MQTT-IoT-IDS2020 presents the classifier's performance on IoT-specific data, highlighting different characteristics and challenges unique to IoT traffic.

The comparative analysis of the ROC curves reveals several key observations. The AUC values for each dataset indicate the overall performance of the classifier. Higher AUC values represent better performance, with the classifier effectively distinguishing between benign and malicious activities. The number of features in each dataset impacts the classifier's performance. More features can introduce complexity, potentially leading to overfitting or underfitting. The ROC curves provide insights into how well the classifier generalizes across different types of data. Synthetic datasets, while not identical to real-world data, can help in understanding the potential challenges and limitations of the classifier.

TABLE 7. Training time reduction achieved by GA and FL.

Dataset	Baseline	GA	FL	Total(%)
UNSW-NB15	12 hrs	10.2 hrs (15%)	9.5 hrs (10%)	20.8%
IoT-23	7 hrs	5.9 hrs (15.7%)	5.2 hrs (12%)	25.7%
CSE-CIC-IDS2018	15 hrs	12.7 hrs (15.3%)	12 hrs (10%)	20.0%
MQTT-IoT-IDS2020	8 hrs	6.8 hrs (15%)	5.6 hrs (17.6%)	30.0%

The proposed approach significantly improved training efficiency by integrating Genetic Algorithms (GA) for hyperparameter tuning and Federated Learning (FL) for distributed training. GA reduced the number of training iterations by optimizing parameters such as learning rate, batch size, and the number of critic updates, leading to a training time reduction of 15-20% across different datasets. On the other hand, FL improved efficiency by enabling parallel training across multiple nodes, reducing centralized processing overhead and cutting training time by 10-15%. The overall improvement varied based on dataset size, with larger datasets (UNSW-NB15 and CSE-CIC-IDS2018) showing a greater reduction in training time due to optimized batch processing, while smaller datasets (IoT-23 and MQTT-IoT-IDS2020) benefited more from distributed learning. Table 7 summarizes the individual contributions of GA and FL to the total training time reduction.

VII. CONCLUSION

This paper introduces a novel approach that synergistically combines Genetic Algorithms (GAs), Generative Adversarial Networks (GANs), and Federated Learning (FL) to tackle the challenging problem of rare event detection, specifically focusing on anomaly detection. The rarity of anomalies necessitates an unsupervised approach. Our proposed model effectively addresses this issue through two primary contributions:

- **Genetic GAN Model for Imbalanced Data**: We presented a genetic GAN model designed to generate synthetic data that balances minority classes. This directly addresses the issue of imbalanced data distribution, which is critical for accurate anomaly detection, particularly in handling rare events. By generating diverse attack instances, the model ensures better representation of underrepresented classes, improving the classifier's ability to detect anomalies.
- **Comprehensive Experimental Validation**: Comprehensive experiments were conducted to validate the efficacy of our approach. The experimental results demonstrated improved performance and efficiency of our model in detecting anomalies. The inclusion of synthetic data generated by the GAN enhanced the robustness of the detection system. This empirical evidence underscores the practical advantages of our proposed solution, as it leads to more reliable detection rates in anomaly-prone environments like IoT networks.

Overall, our approach demonstrates significant advantages by:

- Addressing the imbalanced data problem, which is common in intrusion detection systems (IDS).
- Utilizing federated learning to maintain data privacy while enhancing model performance through collaborative learning.
- Ensuring high-quality synthetic data generation with WGAN-GP, which mitigates common issues such as mode collapse and vanishing gradients.

This innovative combination of GAs, GANs, and federated learning not only improves anomaly detection but also contributes to strengthening network security by providing a balanced and scalable solution for detecting rare and critical events.

REFERENCES

[1] B. Kaur, S. Dadkhah, F. Shoeleh, E. C. P. Neto, P. Xiong, S. Iqbal, P. Lamontagne, S. Ray, and A. A. Ghorbani, "Internet of Things (IoT) security dataset evolution: Challenges and future directions," *Internet Things*, vol. 22, Jul. 2023, Art. no. 100780.

[2] I. H. Sarker, A. I. Khan, Y. B. Abushark, and F. Alsolami, "Internet of Things (IoT) security intelligence: A comprehensive overview, machine learning solutions and research directions," *Mobile Netw. Appl.*, vol. 28, no. 1, pp. 296–312, Feb. 2023.

[3] S. Banik, T. Banik, and S. Banik, "Intrusion detection system in smart grid—A review," *Preprints*, 2023, Art. no. 2023090611.

[4] S. V. Amanoul, A. M. Abdulazeez, D. Q. Zeebare, and F. Y. H. Ahmed, "Intrusion detection systems based on machine learning algorithms," in *Proc. IEEE Int. Conf. Autom. Control Intell. Syst. (I2CACIS)*, Jun. 2021, pp. 282–287.

[5] D. W. Al-Safaar and W. L. Al-Yaseen, "A survey of network intrusion detection systems based on deep learning approaches," *Sci. Tech. J. Inf. Technol., Mech. Opt.*, vol. 23, no. 2, pp. 352–363, Apr. 2023.

[6] G. Aguiar, B. Krawczyk, and A. Cano, "A survey on learning from imbalanced data streams: Taxonomy, challenges, empirical study, and reproducible experimental framework," *Mach. Learn.*, vol. 113, no. 7, pp. 4165–4243, Jul. 2024.

[7] J. Park, S. Kwon, and S.-P. Jeong, "A study on improving turnover intention forecasting by solving imbalanced data problems: Focusing on SMOTE and generative adversarial networks," *J. Big Data*, vol. 10, no. 1, pp. 1–16, Mar. 2023.

[8] H. Ding, L. Chen, L. Dong, Z. Fu, and X. Cui, "Imbalanced data classification: A KNN and generative adversarial networks-based hybrid approach for intrusion detection," *Future Gener. Comput. Syst.*, vol. 131, pp. 240–254, Jun. 2022.

[9] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 27, 2014, pp. 1–7.

[10] S. Tasneem, K. D. Gupta, A. Roy, and D. Asgupta, "Generative adversarial networks (GAN) for cyber security: Challenges and opportunities," in *Proc. IEEE Symp. Ser. Comput. Intell.*, 2022, pp. 4–7.

[11] Y. Liu, J. Li, B. Liu, X. Gao, and X. Liu, "Malware identification method based on image analysis," in *Proc. 11th Int. Conf. Inf. Technol. Med. Educ. (ITME)*, Nov. 2021, pp. 157–161.

[12] S. Wang, Q. Wang, Z. Jiang, X. Wang, and R. Jing, ''A weak coupling of semi-supervised learning with generative adversarial networks for malware classification,'' in *Proc. 25th Int. Conf. Pattern Recognit. (ICPR)*, Jan. 2021, pp. 3775–3782.
[13] X. Peng, H. Xian, Q. Lu, and X. Lu, ''Semantics aware adversarial malware examples generation for black-box attacks,'' *Appl. Soft Comput.*, vol. 109, Sep. 2021, Art. no. 107506.
[14] N. Abdalgawad, A. Sajun, Y. Kaddoura, I. A. Zualkernan, and F. Aloul, ''Generative deep learning to detect cyberattacks for the IoT-23 dataset,'' *IEEE Access*, vol. 10, pp. 6430–6441, 2022.
[15] T. Zixu, K. S. K. Liyanage, and M. Gurusamy, ''Generative adversarial network and auto encoder based anomaly detection in distributed IoT networks,'' in *Proc. IEEE Global Commun. Conf.*, Dec. 2020, pp. 1–7.
[16] L. S. Awad and S. Sarhan, ''Hybrid intrusion detection system combining of self-organizing map and backpropagation with GANs neural networks,'' *Seybold Rep. J.*, vol. 18, pp. 1–7, Apr. 2023.
[17] Y. Liu, C. Jiang, C. Lu, Z. Wang, and W. Che, ''Increasing the accuracy of soil nutrient prediction by improving genetic algorithm backpropagation neural networks,'' *Symmetry*, vol. 15, no. 1, p. 151, Jan. 2023.
[18] E. Real, S. Moore, A. Selle, S. Saxena, Y. L. Suematsu, J. Tan, Q. V. Le, and A. Kurakin, ''Large-scale evolution of image classifiers,'' in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 2902–2911.
[19] B. Roziere, F. Teytaud, V. Hosu, H. Lin, J. Rapin, M. Zameshina, and O. Teytaud, ''EvolGAN: Evolutionary generative adversarial networks,'' in *Proc. Asian Conf. Comput. Vis.*, 2020, pp. 1–16.
[20] U. Tammaro, ''GAN hyperparameters search through genetic algorithm,'' Doctoral dissertation, Thèse de maîtrise, Universidade NOVA de Lisboa, Portugal, 2021.2022.
[21] M. A. Ferrag, O. Friha, L. Maglaras, H. Janicke, and L. Shu, ''Federated deep learning for cyber security in the Internet of Things: Concepts, applications, and experimental analysis,'' *IEEE Access*, vol. 9, pp. 138509–138542, 2021.
[22] X. Cao, G. Sun, H. Yu, and M. Guizani, ''PerFED-GAN: Personalized federated learning via generative adversarial networks,'' *IEEE Internet Things J.*, vol. 10, no. 5, pp. 3749–3762, Mar. 2023.
[23] J. Zhang, L. Zhao, K. Yu, G. Min, A. Y. Al-Dubai, and A. Y. Zomaya, ''A novel federated learning scheme for generative adversarial networks,'' *IEEE Trans. Mobile Comput.*, vol. 23, no. 5, pp. 3633–3649, May 2023.
[24] A. Wijesinghe, S. Zhang, and Z. Ding, ''PS-FedGAN: An efficient federated learning framework based on partially shared generative adversarial networks for data privacy,'' 2023, *arXiv:2305.11437*.
[25] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, ''Generative adversarial networks,'' *Commun. ACM*, vol. 63, no. 11, pp. 139–144, 2020.
[26] A. Aggarwal, M. Mittal, and G. Battineni, ''Generative adversarial network: An overview of theory and applications,'' *Int. J. Inf. Manage. Data Insights*, vol. 1, no. 1, Apr. 2021, Art. no. 100004.
[27] M. Arjovsky, S. Chintala, and L. Bottou, ''Wasserstein generative adversarial networks,'' in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 214–223.
[28] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, ''Improved training of Wasserstein GANs,'' in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 301, 2017, pp. 1–12.
[29] N. Rodríguez-Barroso, G. Stipcich, D. Jiménez-López, J. A. Ruiz-Millán, E. Martínez-Cámara, G. González-Seco, M. V. Luzón, M. A. Veganzones, and F. Herrera, ''Federated learning and differential privacy: Software tools analysis, the Sherpa.Ai FL framework and methodological guidelines for preserving data privacy,'' *Inf. Fusion*, vol. 64, pp. 270–292, Dec. 2020.
[30] N. Moustafa and J. Slay, ''UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set),'' in *Proc. Mil. Commun. Inf. Syst. Conf. (MilCIS)*, Nov. 2015, pp. 1–6.
[31] A. Borji, ''Pros and cons of GAN evaluation measures: New developments,'' *Comput. Vis. Image Understand.*, vol. 215, Jan. 2022, Art. no. 103329.
[32] S. Alabdulwahab, Y.-T. Kim, A. Seo, and Y. Son, ''Generating synthetic dataset for ML-based IDS using CTGAN and feature selection to protect smart IoT environments,'' *Appl. Sci.*, vol. 13, no. 19, p. 10951, Oct. 2023.
[33] Z. Li, J. Shao, Y. Mao, J. H. Wang, and J. Zhang, ''Federated learning with GAN-based data synthesis for non-IID clients,'' in *Proc. Int. Workshop Trustworthy Federated Learn.* Cham, Switzerland: Springer, 2022, pp. 17–32.
[34] C. Byrne and P. P. Eggermont, ''EM algorithms,'' in *Handbook of Mathematical Methods in Imaging*. New York, NY, USA: Springer, 2015.
[35] R. Arthi, S. Krishnaveni, and S. Zeadally, ''An intelligent SDN-IoT enabled intrusion detection system for healthcare systems using a hybrid deep learning and machine learning approach,'' *China Commun.*, vol. 21, no. 10, pp. 1–21, Oct. 2024.
[36] Z. Zhang, Y. Hua, H. Wang, and S. McLoone, ''Improving the fairness of the min-max game in GANs training,'' in *Proc. IEEE/CVF Winter Conf. Appl. Comput. Vis. (WACV)*, Jan. 2024, pp. 2910–2919.
[37] N. R. Al-Milli and Y. A. Al-Khassawneh, ''Intrusion detection system using CNNs and GANs,'' *WSEAS Trans. Comput. Res.*, vol. 12, pp. 281–290, Apr. 2024.
[38] P. J. Maliakel, S. Ilager, and I. Brandic, ''FLIGAN: Enhancing federated learning with incomplete data using GAN,'' in *Proc. 7th Int. Workshop Edge Syst., Anal. Netw.*, 2024, pp. 1–6.
[39] X. Wang, Y. Xu, Y. Xu, Z. Wang, and Y. Wu, ''Intrusion detection system for in-vehicle CAN-FD bus ID based on GAN model,'' *IEEE Access*, vol. 12, pp. 82402–82412, 2024.

WAYOUD BOUZERAIB received the master's degree in new information and communication technologies (NICT). He is currently pursuing the Ph.D. degree in computer science with University Abdelhamid Mehri Constantine 2, Algeria. Since 2019, he has been an integral part of the LIRE Laboratory Team. His thesis focuses on the security aspect of the Internet of Things (IoT). His research interests include the IoT and security challenges, and the implementation and utilization of evolutionary approaches in the field of optimization, with a keen interest in leveraging data-driven insights and innovative techniques.

AFIFA GHENAI received the Ph.D. degree in computer science. She is currently an Associate Professor with University Abdelhamid Mehri Constantine 2, Algeria. She is also a member of the LIRE Laboratory and several projects in the area of embedded systems, formal methods, and cyber-security. Her current research interests include advances in distributed systems, service-oriented computing, the Internet of Things, cyber-security, cloud computing, and autonomic computing.

NADIA ZEGHIB received the Ph.D. degree in computer science, in 2007, with a focus on formal specification languages and extended concurrent algebraic high-level Petri nets. She is currently a Professor with University Abdelhamid Mehri Constantine 2, Algeria. She is also a member of the LIRE Laboratory and several projects in the area of formal methods for the development and verification of software engineering. Her current research interests include formal methods, service-oriented computing, mathematical logic, design and verification of complex systems, cloud computing, and autonomic computing.