

Article

Class-Specific GAN-Based Minority Data Augmentation for Cyberattack Detection Using the UWF-ZeekData22 Dataset

Asfaw Debelie ¹, Sikha S. Bagui ^{1,*} , Dustin Mink ²  and Subhash C. Bagui ³ 

¹ Department of Computer Science, The University of West Florida, Pensacola, FL 32514, USA; aad47@students.uwf.edu

² Department of Cybersecurity, The University of West Florida, Pensacola, FL 32514, USA; dmink@uwf.edu

³ Department of Mathematics and Statistics, The University of West Florida, Pensacola, FL 32514, USA; sbagui@uwf.edu

* Correspondence: bagui@uwf.edu

Abstract

Intrusion detection systems (IDS) often struggle to detect rare but high-impact attack behaviors due to severe class imbalance in real-world network traffic. This work proposes a class-specific GAN-based augmentation framework that explicitly targets sparsity in the minority-class in structured cybersecurity datasets. Unlike prior GAN-based approaches that employ global augmentation or anomaly-driven synthesis, separate Generative Adversarial Networks (GANs) are trained independently for each MITRE ATT&CK tactic using only real minority-class samples, enabling focused distribution learning without contamination from benign traffic. Using a relatively new network traffic dataset, UWF-ZeekData22, the proposed framework augments minority classes under conditions of extreme sample sparsity, where traditional classifiers and interpolation-based oversampling methods are ineffective or statistically unreliable. Five traditional classifiers—Logistic Regression, Support Vector Machine (SVM), k-Nearest Neighbors (KNN), Decision Tree, and Random Forest—are evaluated before and after augmentation using stratified 5-fold cross-validation. Experimental results show that class-specific GAN augmentation consistently improves recall and F1-score for rare attack tactics, with the largest gains observed under extreme sparsity where pre-augmentation evaluation was infeasible. Notably, false-negative rates are substantially reduced without degrading majority-class performance, demonstrating that the proposed approach enhances minority-class separability rather than inflating evaluation metrics. These findings demonstrate that class-specific GAN-based augmentation is a practical and robust data-level strategy for improving the detection of rare MITRE ATT&CK-aligned attack behaviors in machine-learning-based IDSs.

Keywords: cybersecurity; intrusion detection; class imbalance; generative adversarial networks; data augmentation; minority-class learning; MITRE ATT&CK; network traffic analysis



Academic Editor: Sotirios K. Goudos

Received: 15 December 2025

Revised: 28 January 2026

Accepted: 9 February 2026

Published: 12 February 2026

Copyright: © 2026 by the authors.

Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

1. Introduction

Modern intrusion detection systems (IDS) increasingly rely on machine learning (ML) models to detect malicious activity in high-volume network traffic [1]. Although these models show strong performance in detecting common behaviors, they perform significantly worse on rare but high-impact cyberattacks. This challenge is primarily due to severe class imbalance in cybersecurity datasets, in which benign traffic overwhelmingly dominates and attack behaviors are observed only sparsely. Minority classes, such as Credential

Access, Persistence, Lateral Movement, or Exfiltration, contain disproportionately few labeled samples, making it difficult for classifiers to learn reliable decision boundaries. As a result, IDS models can exhibit high false-negative rates for the attacks that are most critical to detect.

Traditional approaches for addressing class imbalance include oversampling methods such as SMOTE [2–4] and ADASYN [5,6], as well as hybrid resampling techniques [6,7]. While effective in some domains, these methods rely on interpolation and may distort the underlying feature distributions in structured cybersecurity network-flow data. Such distortion can yield unrealistic synthetic samples or reinforce noise in the minority class. Moreover, interpolation methods may fail to capture complex nonlinear patterns that characterize adversarial behaviors in real network datasets [8].

Generative Adversarial Networks (GANs) are a compelling alternative for data augmentation because they can learn and replicate complex data distributions [9,10]. By training the generator to produce synthetic samples that resemble real minority-class data, GANs can expand the training dataset with diverse samples that help classifiers learn more robust decision boundaries [11,12]. Although GANs have been increasingly applied to cybersecurity problems in recent years [8,13], most existing work focuses on global data augmentation, adversarial traffic generation, or anomaly detection, with comparatively limited attention to class-specific augmentation under extreme minority-class sparsity in structured network traffic data [1,11,14,15].

This study addresses class imbalance by introducing a modular, per-class GAN augmentation pipeline designed for structured, labeled cybersecurity datasets such as UWF-ZeekData22 [16,17]. Each MITRE ATT&CK tactic is treated as an independent binary classification problem against benign traffic. For each tactic, a GAN is trained solely on minority samples to generate synthetic data that improves class representation during training. We evaluate this approach across five widely used machine learning classifiers—Logistic Regression (LR), Support Vector Machine (SVM), K-Nearest Neighbor (KNN), Decision Tree (DT), and Random Forest (RF)—using stratified 5-fold cross-validation and standard metrics such as Precision, Recall, F1-score, and AUC-ROC.

Our results demonstrate that GAN-augmented training improves minority-class detection across the evaluated tactics and classifiers. In several cases, GAN-generated samples substantially reduce false negatives and, under specific classifiers, eliminate minority-class misclassification. Importantly, these improvements occur without degrading performance on the majority class, suggesting that GAN augmentation enhances class separability rather than introducing harmful noise.

Figure 1 illustrates the overall scenario and workflow of the proposed class-specific GAN-based augmentation framework. The figure summarizes how severely imbalanced network traffic data is processed, how independent GANs are trained for individual minority MITRE ATT&CK tactics, and how the resulting synthetic samples are used to augment training data for downstream classifiers.

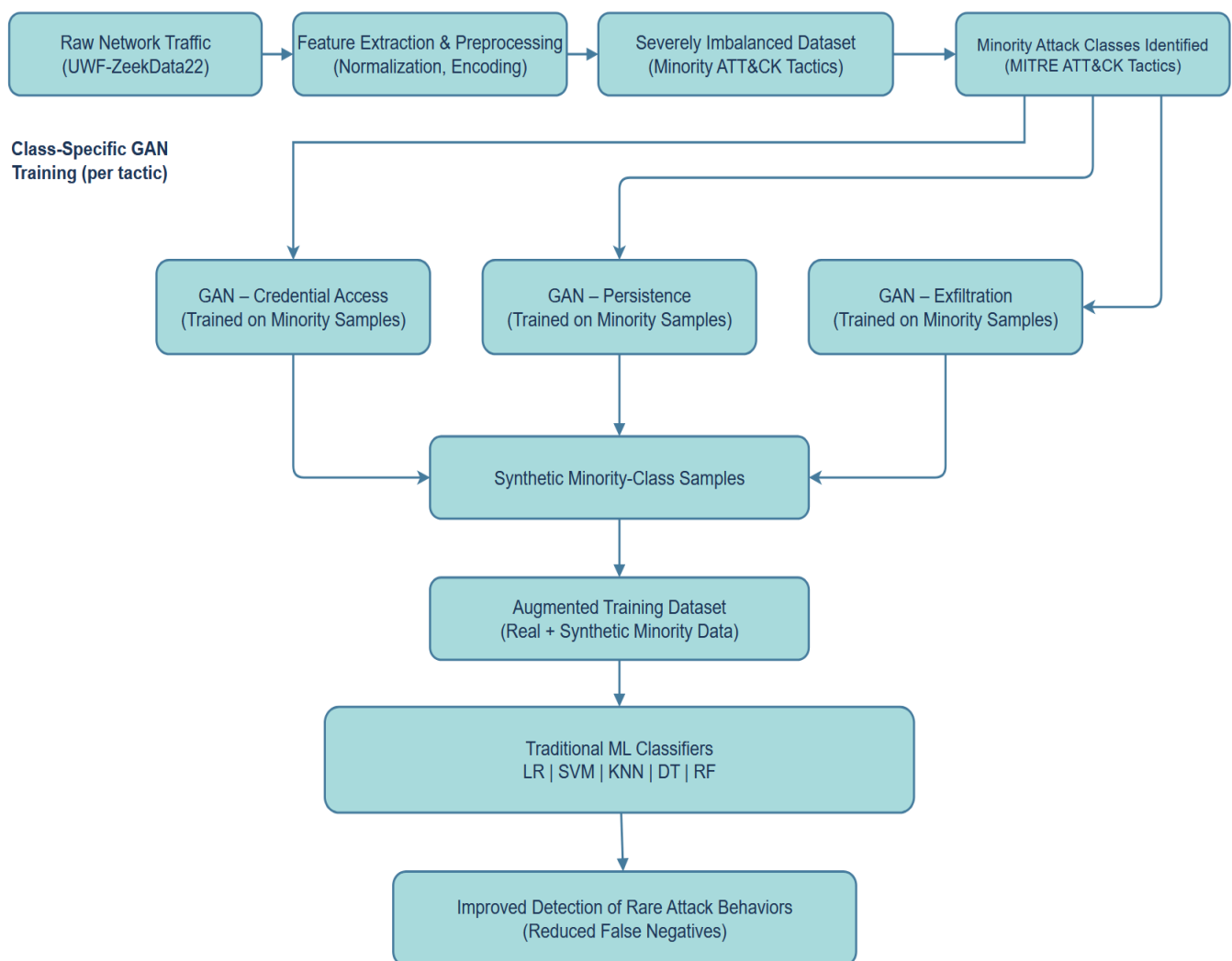
The main contributions of this work are as follows:

1. We propose a class-specific GAN-based data augmentation framework in which separate GANs are trained independently for individual MITRE ATT&CK tactics, enabling focused modeling of rare attack behaviors under extreme class imbalance.
2. We address extreme minority-class sparsity (including cases with fewer than ten real samples) and demonstrate that GAN augmentation enables stable classifier training and evaluation where pre-augmentation assessment is infeasible.
3. We conduct a systematic before-and-after evaluation across five traditional machine-learning classifiers, using stratified 5-fold cross-validation, isolating the impact of augmentation on minority-class detection.

4. We show consistent reductions in false negatives and improvements in recall and F1-score across multiple attack tactics without degrading majority-class performance, validating the robustness of the proposed data-level augmentation strategy.
5. We provide a practical augmentation strategy pipeline aligned with the MITRE ATT&CK framework that can be integrated into real-world IDS workflows.

The rest of this paper is organized as follows: Section 2 introduces GANs. Section 3 reviews related research on oversampling and GAN-based data augmentation. Section 4 covers the dataset and preprocessing steps. Section 5 discusses classifiers and evaluation metrics. Section 6 describes the experimental setup. Section 7 shows the results before and after augmentation, and Section 8 discusses the findings and their implications for cybersecurity detection systems. Section 9 concludes the study and suggests directions for future research.

Scenario Overview of Class-Specific GAN-Based Data Augmentation for Intrusion Detection



Validation folds contain only real data to prevent data leakage.

Figure 1. Overview of the proposed class-specific GAN-based augmentation framework for intrusion detection under extreme imbalance. Separate GANs are trained for each minority MITRE ATT&CK tactic to generate synthetic samples, which are used to augment training data for downstream classifiers.

2. Generative Adversarial Networks

2.1. Introduction

A Generative Adversarial Network (GAN), introduced by Goodfellow et al. [18], is a framework consisting of two neural networks: a generator that produces synthetic samples and a discriminator that distinguishes real data from generated data [19]. The two components of a GAN, the generator and the discriminator, have opposing objectives. The generator G aims to learn the underlying data distribution and to produce realistic samples. At the same time, the discriminator, D , estimates the probability that a given sample originates from the real training data rather than the generator [18]. Figure 2 illustrates the conceptual architecture of a GAN.

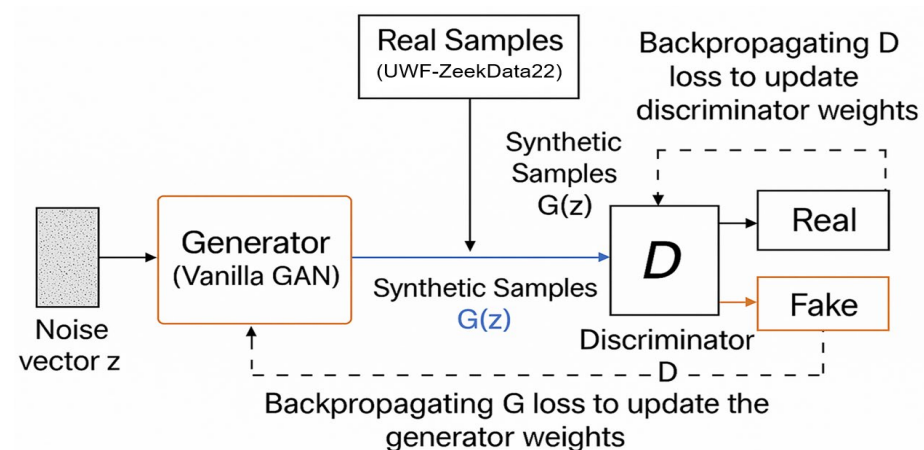


Figure 2. Conceptual diagram of GAN.

GANs are formulated as minmax two-player games [18], in which G seeks to maximize the likelihood that the discriminator misclassifies generated samples. Meanwhile, D aims to distinguish real data from generated data [14]. The discriminator learns the distribution of real data samples and uses this knowledge to differentiate between real and generated data [9]. The adversarial interaction between G and D enables both networks to iteratively improve until the generated samples are statistically indistinguishable from real data. Since their introduction in 2014 [19–21], GANs have outperformed their competitors and achieved success in generative tasks across various fields [9].

In this section, we present the fundamental principles of GANs relevant to understanding their use as data augmentation models for imbalanced cybersecurity datasets.

2.2. Background

Generative Versus Discriminative Models

Discriminative models aim to learn the decision boundary between classes and to predict labels from input variables [9]. They model the conditional probability distribution $p(x|y)$, where y denotes the label and x represents the input. Such models are commonly applied to classification, detection, and segmentation tasks [19].

In contrast, generative models aim to capture the underlying structure of the training data by learning the full data distribution [15]. These models represent the joint probability distribution $p(x, y)$, where x denotes the input and y denotes the label [9]. GANs belong to this class of deep generative models, focusing on learning data distribution rather than explicit decision boundaries.

Figure 3 shows the differences between discriminative and generative models. The left side (Discriminative: $p(y|x)$) focuses on learning the decision boundary between classes. The discriminative model aims to separate the two classes by finding the best boundary.

In contrast, the right side (Generative: $p(x, y)$) emphasizes modeling the joint distribution of each class by conditioning on both x (features) and y (labels), thereby generating new samples that resemble each class. The arrows in Figure 3 illustrate the learning focus of each approach: discriminative models learn a decision boundary between benign and attack traffic ($p(x|y)$), while generative models learn the underlying class distributions ($p(x, y)$) to enable sample generation.

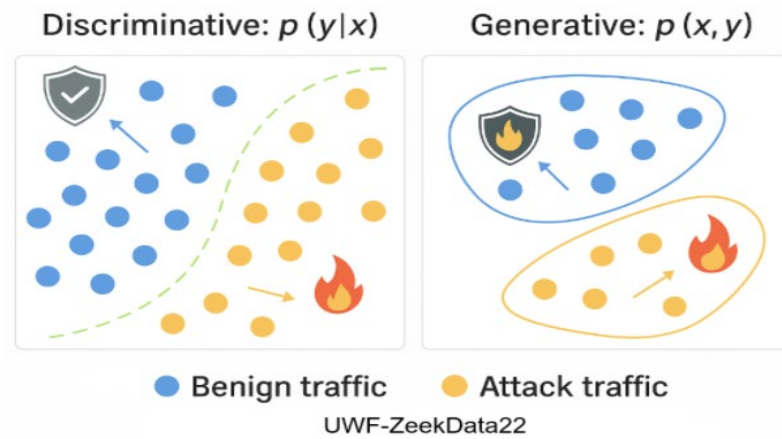


Figure 3. Discriminative versus generative models.

2.3. Components, Structure, and Architecture

As shown in Figure 2, the GAN model consists of two networks: D , the Discriminator, and G , the Generator. These two components are trained as deep neural networks in an alternating manner. The discriminator (D) is trained to differentiate between real and generated data. In contrast, the generator (G) is trained to fool the discriminator by preventing it from distinguishing between real and generated data [22].

Given a noise distribution p_z , G takes a sample z as input and produces a probability distribution p_g that closely matches p_{data} , to the point of being indistinguishable. During training, D aims to maximize the probability of correctly labeling both samples from the training data, p_{data} , and those generated by G , p_g [8]. G and D engage in a two-player minmax game with the value function $V(G, D)$:

$$\min_G \max_D V(D, G) = E_{x \sim p_{data}(x)} [\log D(x)] + E_{z \sim p_z(z)} [\log(1 - D(G(z)))] \quad (1)$$

where D denotes the discriminator, G the generator $p_{data}(x)$ the data distribution, and $p_z(z)$ the prior noise distribution.

D aims to minimize $D(G(z))$ or to maximize $\log D(x) + 1 - D(G(z))$. The generator wants to minimize $E_{z \sim p_z(z)} [\log(1 - D(G(z)))]$ to fool the discriminator into labeling the generated data as real [10,11]. This minmax optimization problem between G and D is solved iteratively until the Nash Equilibrium is reached [9].

These components collectively define the standard adversarial training framework used across all GAN variants evaluated in this study.

The goal of the GAN training process is to maximize $E_{x \sim p_{data}(x)} [\log D(x)]$ to increase the accuracy of discerning real data while G minimizes $\log(1 - D(G(z)))$ to increase the probability of fooling the discriminator [9,10].

2.4. Learning Modes and Algorithms

During training, G and D engage in a min-max game: G attempts to generate data that are similar to the training distribution. At the same time, D tries to distinguish between the generated and real data by assigning higher or lower probabilities [23]. The discriminator

makes this decision by computing a distance measure between the probability distributions of the generated data, p_g , and the real data, p_{data} [9].

Various statistical measures are used to assess the probability distribution. To measure distance, the original GAN by Goodfellow et al. [18] employed a divergence metric based on Kullback–Leibler (KL) and Jensen-Shannon (JS) divergence [9]. However, using JS and KL divergence during training has shown stability issues, such as:

- Mode collapse:—where large probabilities originate from data p_{data} and small probabilities come from p_g .
- Poor convergence:—the discriminator learns to distinguish real and generated data very easily, causing the generator’s optimization to vanish quickly and leading to unstable training [19].

2.5. Objective (Loss) Function

In GANs, the objective function is designed to minimize the divergence between the probability distributions of real data p_{data} and generated data p_g , with the goal of reaching an equilibrium in which the generated data are indistinguishable from the real samples [9]. To achieve this, adversarial optimization is formulated as a zero-sum game between two computing networks, a Generator G and a Discriminator D .

In this formulation, the Discriminator seeks to maximize its ability to correctly distinguish real samples from generated ones, while the Generator seeks to minimize this ability by producing increasingly realistic synthetic data. Both networks optimize the same value function with opposing objectives, such that any improvement in one player’s performance necessarily degrades the performance of the other. This antagonistic interaction naturally defines a minmax optimization problem whose solution corresponds to a Nash equilibrium, at which point neither network can improve its objective without altering the other’s strategy [23].

At equilibrium, the Generator’s distribution converges towards the true data distribution, and the Discriminator becomes unable to reliably distinguish real from synthetic samples. This zero-sum formulation is fundamental to GAN training, as it enforces distribution matching through adversarial pressure.

Training

During training, a GAN is trained on two small batches (minibatches) of data: one with real examples (labeled 1) and the other with fake examples generated by the generator (labeled 0) [23]. The discriminator learns to estimate the likelihood that a sample is real rather than fake, which can be expressed as the probability ratio $p_{data}(x)/p_g(x)$.

The goal is to train G and D so that the generated data is indistinguishable from the real data; that is, when D evaluates it, D cannot distinguish between the real and generated data. At this point, G and D are said to converge, and the distributions of p_{data} and p_g match, which is expressed as:

$$p_g = p_{data} \quad (2)$$

The minmax objective, $p_g = p_{data}$, occurs at the saddle point [18], where the generator tries to minimize the loss and the discriminator maximizes it. The saddle point (shown in Figure 4) is hard to reach because the loss function is complex, and G and D interact in ways that are difficult to control. As a result, training does not always converge. This causes instability when a single player becomes overly dominant, creating an imbalance. It can also lead to model collapse, in which G may produce only a limited variety of data [23]. In Figure 4, the surface colors represent the loss landscape of the adversarial game. The arrows illustrate alternating gradient updates of the generator (G) and discriminator (D) as they move toward the saddle point, which corresponds to adversarial equilibrium.

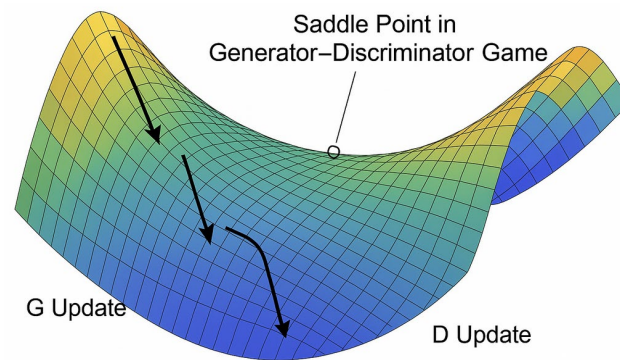


Figure 4. Gradient method on a GAN.

3. Related Works

The application of machine learning to intrusion detection systems (IDSs) has been widely studied, with numerous works documenting the challenges posed by highly imbalanced cybersecurity datasets [24–26]. Traditional oversampling techniques, such as SMOTE [2–4], ADASYN [5,6], and their variants like BSMOTE [27] and SVM-SMOTE [25–27], are widely used to expand minority classes by interpolating synthetic points in feature space [28–31]. While such approaches improve class balance between benign and malicious classes, they often fail to capture the complex, nonlinear distributions of cyberattack behaviors in high-dimensional tabular network traffic data. As a result, these methods may introduce noise or overlapping samples, thereby degrading classifier performance [28–31].

To address these limitations, many studies have employed Generative Adversarial Networks (GANs) [5,8,32,33] to produce synthetic samples for underrepresented classes by learning class distributions and generating high-fidelity data that augment real samples and improve classifier accuracy. GANs have emerged as a powerful alternative to traditional oversampling because they can model high-dimensional data distributions without making explicit assumptions about density or structure [10,18]. Prior work in network security demonstrates that GAN-generated samples can improve detection rates for rare attack types and enhance generalization, particularly when only limited minority-class data are available during training [11,14].

Beyond classical oversampling strategies, GANs have also been explored within intrusion detection system architecture and data preprocessing pipelines. In IDS research, GANs have been used to generate adversarial traffic or to augment rare attack classes, thereby enabling controlled evaluation of detection models under adversarial conditions [34]. In contrast, interpolation-based methods such as SMOTE may lead to overfitting when applied to highly imbalanced cybersecurity datasets [28].

Zhao et al. [14] conducted a comparative study of Vanilla GANs, Wasserstein GANs (WGANs), and Conditional GANs (cGANs) on the CIC-IDS2017 dataset. Their findings suggest that GAN-generated synthetic samples, when incorporated into training pipelines, can improve the performance of intrusion detection systems by increasing detection rates for rare attack types. The study demonstrated that after augmentation, the classification model nearly reached a perfect score. The authors noted that both WGAN and Vanilla GAN models showed significant improvements over cGAN. However, their approach employs a single GAN trained on the full dataset and does not differentiate between individual attack tactics during synthesis. In contrast, our work trains separate GANs for each MITRE ATT&CK tactic, enabling class-specific distribution modeling under extreme sample sparsity and avoiding cross-class interference during data generation.

Krishna et al. [35] presents a Conditional GAN (cGAN)-based framework for malware detection and classification that uses grayscale and RGB image datasets. Their approach

extends cGANs with a class-aware objective and evaluates performance across AC-GAN, DCGAN, and E-GAN architectures using image-based malware representations. However, the dataset used in their study consists of malware images, whereas most real-world intrusion detection systems operate on structured, high-dimensional tabular network traffic. As a result, image-based GAN frameworks are not directly transferable to practical IDS deployments without substantial feature engineering.

Xia et al. [36] proposed an abnormal traffic detection framework for IoT environments based on federated learning and depth wise separable convolutional neural networks (DS-CNNs). Their approach enables collaborative model training across distributed IoT devices while preserving data privacy and employs lightweight convolutional architectures to improve computational efficiency. Their study demonstrates that deep learning-based IDS models can achieve strong detection performance in resource-constrained IoT settings.

However, unlike centralized learning approaches, federated learning mitigates privacy risks by avoiding the direct sharing of raw traffic data; nevertheless, it assumes sufficient labeled data at participating clients and does not explicitly address extreme class imbalance or minority-class data scarcity. In contrast, the present work focuses on data-level augmentation using GANs to improve minority-class representation under severe imbalance in structured network traffic and evaluates augmentation effects across multiple classical classifiers rather than relying on a single deep neural architecture.

Beyond oversampling and standard GAN-based augmentation approaches, alternative strategies have been proposed to address class imbalance in intrusion detection, including cost-sensitive learning and anomaly detection frameworks. Cost-sensitive and class-weighted classifiers adjust loss functions to penalize minority-class errors more heavily; however, such approaches do not increase the diversity of minority-class samples and often remain ineffective under extreme data sparsity [3]. Similarly, GAN-based anomaly detection methods and one-class learning frameworks focus on modeling benign traffic distributions rather than explicitly learning attack-specific patterns, which limit their applicability to supervised multi-tactic intrusion detection tasks [8,16].

More recently, generative models designed for structured tabular data, such as conditional and distribution-aware GAN variants, have been explored to improve the fidelity of synthetic data [4,8,18]. While these models demonstrate improved stability and sample realism, they are typically trained as single global generators and do not explicitly account for per-attack distributional differences within highly imbalanced cybersecurity datasets.

Table 1 summarizes representative prior work on intrusion detection and data augmentation, highlighting differences in datasets, modeling strategies, GAN variants, and augmentation assumptions. As presented in Table 1, most existing approaches employ global data augmentation or anomaly detection, whereas the proposed framework uniquely applies class-specific GAN training to address extreme sparsity in the minority class in structured network traffic.

While classical oversampling techniques such as SMOTE and its variants are widely used, this study does not include a direct empirical comparison with these methods. This choice is intentional. The primary objective of this work is to isolate and evaluate the effectiveness of class-specific GAN-based augmentation under extreme data sparsity, where some minority classes contain fewer than ten real samples. Under these conditions, interpolation-based oversampling methods may be unstable, as they rely on local neighborhood structure that may not exist. By focusing on a before-and-after augmentation comparison, this study directly measures the impact of generative distribution learning on minority-class detection, without confounding effects arising from fundamentally different resampling assumptions.

Table 1. Comparison of Related Work on Data Augmentation and Intrusion Detection.

Study	Dataset	Method Type	GAN Variant	Augmentation Strategy	Class-Specific Modeling	Key Limitation
SMOTE/ADASYN [2–6]	Various IDS	Oversampling	-	Interpolation-based	✗	Fails under extreme sparsity; may introduce overlap
Zhao et al. [14]	CIC-IDS2017	GAN-based IDS	Vanilla GAN, WGAN, cGAN	Global augmentation	✗	Single GAN trained on full dataset
Krishan et al. [35]	Malware image datasets	cGAN-based	AC-GAN, DCGAN, E-GAN	Image-based synthesis	✗	Not applicable to tabular network traffic
Xia et al. [36]	IoT traffic	Deep Learning IDS	-	No augmentation	✗	Assumes sufficient labeled data
GAN anomaly detection [8,16]	Network traffic	One-class GAN	Various	Benign modeling	✗	Not suitable for supervised multi-class IDS
Tabular GANs [4,8,18]	Tabular datasets	GAN-based	cGAN variants	Global tabular synthesis	✗	Do not model per-attack distributions
This work	UWF-ZeekData22	GAN-based augmentation	Vanilla GAN	Per-class augmentation	✓	Focus on classical ML classifiers

From a broader systems perspective, intrusion detection and minority-class attack detection in cyber-physical systems can be viewed as a problem of system resilience and robustness under rare but high-impact faults or adversarial conditions. Resilient and fault-tolerant system design principles emphasize maintaining acceptable system behavior in the presence of uncertainty, failures, or unexpected disturbances [37]. In this context, extreme class imbalance and sparsely observed attack behaviors represent a form of informational fragility, where learning-based detection systems lack sufficient exposure to rare fault modes.

Prior work on resilient system design highlights the importance of anticipating rare events and incorporating mechanisms that improve system robustness to low-frequency, high-consequence scenarios [38,39]. The proposed class-specific GAN-based augmentation framework aligns with these principles by improving the robustness of intrusion detection models to underrepresented attack behaviors, thereby enhancing resilience at the data and model levels rather than relying solely on architectural redundancy or rule-based fault handling [1].

This work extends existing research by applying GAN-based augmentation to a real-world cybersecurity tabular dataset, UWF-ZeekData22 [16,17], using a per-class strategy in which each GAN is trained exclusively on a single minority attack class (e.g., Credential Access). This design enables each generator to learn highly specific feature distributions associated with individual attack behaviors. In addition, the study evaluates the effectiveness of augmentation across multiple traditional classifiers using macro-averaged performance metrics, confusion matrices, and t-SNE visualizations, thereby enabling a comprehensive and reproducible evaluation of GAN-based augmentation for intrusion detection.

4. Data

4.1. Dataset Overview

This study uses the UWF-ZeekData22 [16,17], a real-world network traffic dataset labeled according to the MITRE ATT&CK framework. The dataset contains structured tabular flow-level features extracted from Zeek logs and includes both benign traffic and multiple cyberattack tactics. A key challenge of UWF-ZeekData22 is its extreme class imbalance, in which several attack tactics comprise only a handful of samples (ranging from 1 to 31 instances after preprocessing), whereas benign traffic dominates the dataset.

Such sparsity severely limits conventional machine learning classifiers' ability to learn meaningful decision boundaries for minority attack behaviors.

Each MITRE ATT&CK tactic is treated as an independent binary classification problem against benign traffic. This design reflects practical intrusion-detection deployments, in which security systems are often configured to detect specific attack behaviors rather than to perform full multiclass attribution. Moreover, multiclass learning under extreme imbalance can obscure minority-class patterns, whereas binary decomposition enables focused modeling of each attack tactic and avoids dominance by the benign class during training.

The target variable is the *label_tactic* column, which assigns each record to a MITRE ATT&CK tactic. The defining characteristic of UWF-ZeekData22 is its extreme class imbalance, which closely reflects operational network conditions. While benign traffic dominates the dataset, several ATT&CK tactics are underrepresented, with as few as 1–10 samples in some cases. Minority classes such as Exfiltration, Lateral Movement, Resource Development, Initial Access, Persistence, and Defense Evasion are particularly underrepresented, creating serious challenges for traditional machine learning models, which tend to favor the majority class and misclassify rare cyberattacks as benign.

Although the full UWF-ZeekData22 dataset contains over nine million network flow records (Table 2), this study uses a subset of 428,787 samples due to the significant computational demands of training multiple GAN variants, conducting extensive drift and fidelity analysis, and performing stratified cross-validation across nine minority classes. The subset was created by keeping all minority-class samples and down-sampling only the majority ("none") class. This maintains the extreme imbalance shown in Table 2 while ensuring that every real malicious instance remains available for augmentation. Since GAN-based reconstruction primarily relies on capturing the geometry of minority classes rather than the absolute abundance of benign traffic, this subset provides a computationally manageable yet statistically representative basis for evaluating generative augmentation under realistic imbalance conditions. The subset retains the imbalance needed for evaluating GAN-based augmentation while reducing computational overhead.

Table 2. Distribution of malicious traffic in UWF-ZeekData22 [16,17].

Attack Type	Description	Count
Reconnaissance	Techniques in which adversaries collect information about the target organization, its infrastructure, or its personnel [40].	9,278,722
Discovery	Techniques employed to explore and map the victim's environment include identifying users, systems, or security mechanisms [41].	2086
Credential Access	Method for securing account names, passwords, tokens, or other credentials used to access systems and services [42].	31
Privilege Escalation	Techniques that acquire higher-level permissions enable broader or deeper access to systems and data [43].	13
Exfiltration	Techniques for sealing and removing data from the victim environment using encrypted or covert channels [44].	7
Lateral Movement	Techniques that enable network movement to access remote systems and extend control [45].	4
Resource Development	Techniques where adversaries create resources like infrastructure, accounts, or capabilities that are later used to support operations [46].	3
Initial Access	Techniques used to gain access to the victim's environment by exploiting vulnerabilities through stolen credentials or phishing [47].	1
Persistence	Techniques that sustain an adversary's access to systems through reboots, credential changes, or other disruptions [48].	1
Defense Evasion	Techniques that evade detection or bypass security controls by obfuscating files, disabling defenses, or exploiting trust mechanisms [49].	1

4.2. Preprocessing Pipeline

The dataset, imported from multiple UWF-ZeekData22 parquet files, contains millions of rows. Each record in the dataset describes network traffic and contains numerical, categorical, and Boolean data. Each record includes 23 features: 9 categorical, 11 numerical, 2 Boolean, and one date-time.

To prepare the data for GAN training and classification tasks, the following preprocessing steps were applied:

- **Feature Selection:** columns containing identifiers or non-informative metadata (*uid*, *datetime*, *src_ip_zeek*, *dest_ip_zeek*, *ts*, and *community_id*) were removed.
- **Boolean Conversion:** The Boolean columns (*local_resp* and *local_orig*) were converted to binary integers to increase the compatibility of data with neural network input formats.
- **Handling Missing Values:** Missing values in numerical columns (*duration*, *orig_bytes*, and *resp_bytes*) were imputed using the median value of each column. The missing categorical values (*service* and *history*) were replaced with the placeholder string “unknown”.
- **Encoding Categorical Features:** To convert string values to integers, the categorical features were encoded using *LabelEncoder*. The encoder was saved for later decoding and evaluation.
- **Target Preservation:** To preserve the original class labels, the original labels were preserved for evaluation and reporting.
- **Type Normalization:** All columns were converted to numeric types after encoding to ensure compatibility with GAN and classifier methods.

All preprocessing operations, including feature scaling and normalization, were performed within each training fold only, and the learned parameters were applied to the corresponding validation fold to prevent data leakage during cross-validation.

4.3. Feature Scaling

All features in the dataset were scaled to the range $[-1, 1]$ using *MinMaxScaler* [50] to ensure compatibility with the Generator’s Tanh output activation and to stabilize adversarial training. Mapping real data to the same bounded range as generated samples prevents scale mismatch between real and synthetic data, reduces gradient saturation in the discriminator, and promotes stable convergence during GAN training. Min-max scaling also preserves relative feature relationships without imposing distributional assumptions, which is appropriate for heterogeneous flow-level network traffic features and particularly important under extreme minority-class sparsity. The scalar parameters were computed using training data only and applied consistently across folds to prevent information leakage [12,18].

4.4. Summary

The preprocessing pipeline ensures that the UWF-ZeekData22 dataset is clean, consistent, and well-formatted for GAN-based augmentation and machine learning evaluation. By treating each ATT&CK tactic as a separate binary classification problem, this study directly evaluates how each GAN augmentation influences the detectability of individual cyberattack behaviors under realistic, highly imbalanced conditions.

5. Methodology

This section describes the end-to-end methodology for generating synthetic minority-class data, evaluating sample fidelity, and assessing the impact of GAN-based augmentation on downstream intrusion-detection classifiers. The proposed approach consists of four integrated components: (1) class-specific data filtering, (2) GAN-based data generation

using multiple GAN variants, (3) advanced similarity and drift analysis, and (4) classifier training and evaluation under stratified cross-validation.

Figure 5 presents a detailed flowchart illustrating the complete methodology, including class-specific data filtering, GAN training, synthetic sample generation, data augmentation, and classifier evaluation.

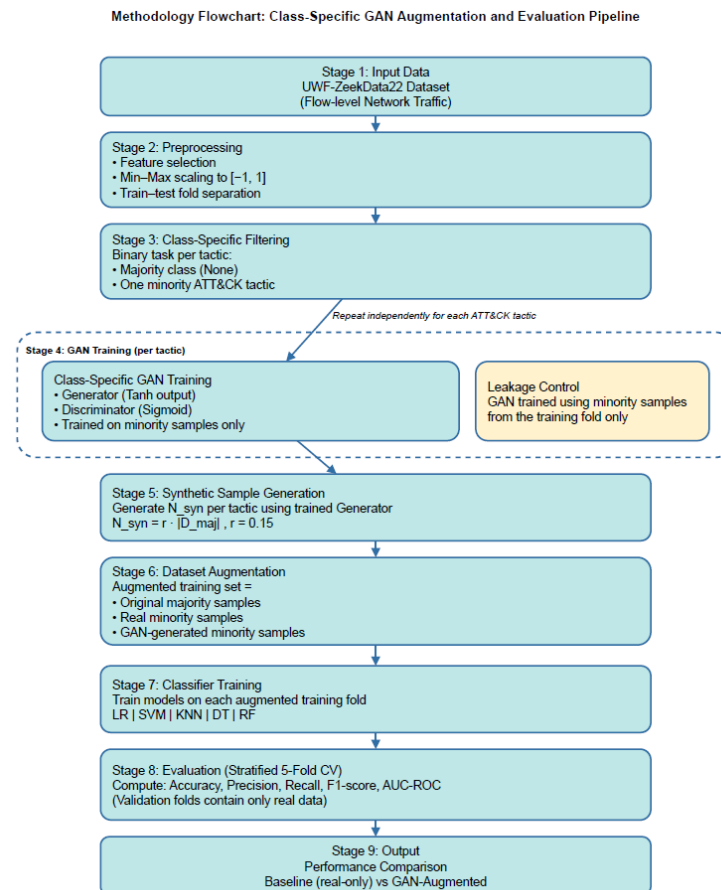


Figure 5. Detailed flowchart of the proposed class-specific GAN-based augmentation and evaluation pipelines. For each MITRE ATT&CK tactic, a separate GAN is trained exclusively on minority-class samples. Synthetic samples are generated and combined with real data to form augmented training sets, which are evaluated using stratified cross-validation across multiple classifiers.

5.1. Class-Specific Augmentation Framework

Given the severe imbalance in the UWF-ZeekData22 dataset, each ATT&CK tactic is treated as a separate binary classification problem. For each experiment, the dataset is filtered to include:

- All samples belonging to the majority class (“none”)
- All samples from a selected minority class (e.g., Credential Access or Persistence)

This creates a binary dataset that isolates the minority class distribution and enables a dedicated GAN model to focus exclusively on reconstructing the structural characteristics of that class. This filtering procedure is repeated independently for each minority class, yielding nine distinct binary learning tasks.

This study employs a class-specific GAN-augmentation framework in which a separate generator-discriminator pair is trained for each minority attack tactic. For each attack tactic, the GAN was trained exclusively on real minority-class samples, without exposure to benign traffic. This ensures that each generator learns only the distribution of its corresponding attack behavior and avoids contamination from majority-class patterns, which could otherwise bias the generation process toward benign characteristics.

5.2. GAN Architecture and Training Procedure

5.2.1. Model Architecture

Each GAN implemented in this study uses a feedforward architecture and operates on the preprocessed feature vector described in Section 4. This consists of the Generator and Discriminator.

Generator (G)

The Generator learns to produce synthetic feature vectors that mimic minority-class samples by mapping the latent noise vector z to the feature space of real data. In this study, the Generator operates on a uniform noise distribution and employs multiple fully connected layers with batch normalization to stabilize training. LeakyReLU activations are used in hidden layers to mitigate vanishing gradients, and a Tanh activation is applied at the output layer to constrain generated features to the normalized range $[-1, 1]$.

The complete Generator architecture and hyperparameters used in all experiments are summarized in Table 3.

Table 3. Generator Architecture and Hyperparameters (Used in Experiments).

Category	Parameter	Value Used *
Input	Latent noise dimension z	32
Architecture	Number of hidden layers	2
Architecture	Hidden units per layer	128
Hidden layers	Linear layer size	$32 \rightarrow 128 \rightarrow 128$
Hidden layers	Normalization	BatchNorm1d after each hidden layer
Hidden layers	Activation function	LeakyReLU ($\alpha = 0.2$)
Hidden layers	Dropout	None
Output layer	Linear output size	$128 \rightarrow D$
Output layer	Activation function	Tanh
Output range	Feature value range	$[-1, 1]$
Output dimension	Feature space size	$D = X_{\text{minority}} \times \text{shape} [1]$

* The same Generator architecture and hyperparameters were used for all minority classes; only the minority-class training data differed between runs.

Discriminator (D)

The Discriminator distinguishes between real and generated samples by mapping input feature vectors to a scalar probability indicating whether a sample is real or synthetic. It consists of fully connected layers with LeakyReLU activations and dropout regularization to reduce overfitting. A Sigmoid activation is applied at the output layer to produce a probability estimate in $[0, 1]$.

The Discriminator architecture and training parameters used in all experiments are summarized in Table 4.

Table 4. Discriminator Architecture and Hyperparameters (Used in Experiments).

Category	Parameter	Value Used *
Input	Feature dimension	D
Architecture	Number of hidden layers	2
Architecture	Hidden units per layer	128
Hidden layers	Linear layer size	$D \rightarrow 128 \rightarrow 128$
Hidden layers	Activation function	LeakyReLU ($\alpha = 0.2$)

Table 4. Cont.

Category	Parameter	Value Used *
Hidden layers	Dropout	0.5
Output layer	Linear output size	128 $\rightarrow$ 1
Output layer	Activation function	Sigmoid
Output	Output interpretation	$P(\text{real} x)$

* The same Discriminator architecture and hyperparameters were used for all minority classes; only the minority-class training data differed between runs.

5.3. GAN Training Dynamics

GAN training is performed independently for each ATT&CK tactic using only minority-class samples from the training fold.

Algorithm 1 summarizes the GAN training procedure applied to each minority ATT&CK tactic independently. Only minority-class samples from the training fold are used to train each GAN, ensuring strict separation between training and validation data and preventing data leakage. Training stability is promoted through feature normalization, mini-batch training, and the use of the Adam optimizer. In this study, each GAN is trained for 2000 epochs with a learning rate of 0.0002. Generator and Discriminator loss trajectories, as well as discriminator output statistics, are monitored to verify stable convergence.

Algorithm 1 Class-Specific GAN Training for Minority Attack Tactics

```

1: Input:
2: Minority – class feature matrix  $X_{min}$  for training fold
3: Noise dimension  $d_z$ 
4: Number of epochs  $E$ 
5: Batch size  $\beta$ 
6: Learning rate  $\eta$ 
7: Output:
8: Trained Generator  $G$  and Discriminator  $D$ 
9: Initialize Generator  $G$  and Discriminator  $D$  with predefined architectures
10: Scale  $X_{min}$  to range  $[-1, 1]$ 
11: Define Binary Cross-Entropy loss  $\mathcal{L}_{BCE}$ 
12: Initialize Adam optimizers for  $G$  and  $D$  with learning rate  $\eta$ 
13: for each epoch  $= 1$  to  $E$  do
14:     Sample a mini-batch of  $\beta$  real minority samples from  $X_{min}$ 
15:     Sample  $\beta$  noise vectors  $z \sim p(z)$ 
16:     Discriminator update:
17:     Generate a synthetic sample  $\check{x} = G(z)$ 
18:     Compute discriminator loss
           
$$\mathcal{L}_D = -E[\log D(x)] - E[\log(1 - D(\check{x}))]$$

19:     Update  $D$  using Adam optimizer
20:     Generator update:
21:     Resample noise vectors  $z \sim p(z)$ 
22:     Generate synthetic samples  $\check{x} = G(z)$ 
23:     Compute generator loss
           
$$\mathcal{L}_G = -E[\log D(\check{x})]$$

24:     Update  $G$  using Adam optimizer
25: end for
26: Return trained Generator  $G$  and Discriminator  $D$ 

```

5.4. Synthetic Sample Generation & Augmentation Ratios

After training, the generator for each tactic is used to produce a set of synthetic minority samples. This study sets the augmentation ratio relative to the majority class to address extreme sparsity across several minority classes.

This study employs the following formulation to determine the number of synthetic samples generated for each minority class:

$$N_{syn} = r \cdot |D_{maj}| \quad (3)$$

where $|D_{maj}|$ is the number of benign samples, and r is the augmentation ratio.

The augmentation ratio was fixed at $r = 0.15$ in this study. This value was selected as a conservative balancing target that increases minority-class representation while avoiding excessive over-synthesis under extreme data sparsity. In the UWF-ZeekData22 dataset, several minority classes contain fewer than ten real samples; aggressively matching the majority class size in such cases would result in synthetic samples dominating the minority distribution, increasing the risk of overfitting to generator artifacts and distorting the original data manifold.

Using a 15% ratio ensures that even the most sparse minority classes can be augmented to form a minimally stable training population, enabling meaningful adversarial training while preserving the statistical integrity of real observations. This augmentation process is applied independently to each minority class identified by a predefined imbalance threshold relative to the majority class size. After training, the generator produces synthetic samples specific to the target majority class, thereby improving class balance and downstream classifier separability without overwhelming real observations.

Evaluating multiple augmentation ratios or adaptive, class-dependent ratio selection strategies through systematic ablation studies is an important direction for future work and will be explored in subsequent studies.

5.5. Classifier Training & Evaluation

To evaluate the impact of GAN-based augmentation on cyberattack detection, five widely used machine learning classifiers were selected. These models cover linear, distance-based, probabilistic, and tree-based paradigms, enabling a comprehensive assessment of how augmentation influences different learning mechanisms. Each classifier is trained separately for each ATT&CK tactic, both with and without synthetic data, under consistent preprocessing and cross-validation procedures.

5.5.1. Logistic Regression (LR)

Logistic Regression is a supervised learning algorithm that trains a linear classifier by estimating the likelihood of an event. It uses the logistic (sigmoid) function to map input features to probabilities, enabling the prediction of whether an unlabeled instance belongs to a particular class [51].

5.5.2. Support Vector Machine (SVM)

SVM is a supervised learning algorithm that constructs an optimal hyperplane to separate data points of different classes. It identifies the decision boundary that maximizes the margin between classes, with support vectors being the data points closest to this boundary [51].

5.5.3. K-Nearest Neighbor (KNN)

KNN is a non-parametric algorithm that classifies an unlabeled example based on the majority class of its k nearest neighbors in the feature space. Each data point is represented

as a position in an n -dimensional space, and classification is performed by measuring the similarity between the new instance and stored training samples [52].

5.5.4. Decision Tree (DT)

A Decision Tree is a supervised learning algorithm that uses a hierarchical structure of nodes and branches to represent decision rules learned from training data. The model begins at the root node and follows decision criteria through internal nodes until it reaches a terminal leaf node [51].

5.5.5. Random Forest (RF)

Random Forest is a supervised learning algorithm that is an ensemble method that constructs multiple decision trees, each trained on a random subset of data and features. By combining the predictions of hundreds of such trees, the ensemble achieves higher accuracy and greater robustness than a single tree [51].

Each classifier was trained using scikit-learn [52], with the random state set to ensure reproducibility. Once the augmented training data is prepared, it is provided directly to the machine-learning classifiers. All five classifiers are trained twice:

1. Baseline: Utilizing only the actual data
2. Augmented: Combining real data with GAN-generated samples

Performance differences between these two settings demonstrate the effect of GAN augmentation on detection capability.

5.6. Methodology Summary

The methodology combines targeted data filtering, adversarial data generation, advanced similarity metrics, classical classification, and ablation-based validation. Together, these elements create a strong framework for examining GAN-based augmentation in highly imbalanced cybersecurity datasets. This modular design allows the pipeline to be replicated for any additional minority class or extended to other tabular cybersecurity datasets.

6. Experimental Setup

This section describes the experimental setup, training parameters, evaluation process, and computational environment used for GAN-based augmentation and classifier benchmarking on the UWF-ZeekData22 dataset. All experiments were conducted using a controlled, repeatable pipeline to ensure a fair comparison across GANs, augmentation ratios, and classifier types. This multi-angle approach provides a more robust and generalizable benchmark for comparing the impact of GAN augmentation on classification performance.

All experiments were conducted using stratified 5-fold cross-validation. For each fold, GAN training and synthetic data generation were performed exclusively on the training partition, and no synthetic samples were introduced into the validation fold, ensuring strict separation between training and evaluation data.

6.1. Computing Environment

All experiments were carried out on a server with the following configuration:

6.1.1. System Configuration

All experiments were conducted in Google Colab Pro+ to enable GPU acceleration and extended compute time. The runtime environment was configured as follows:

- Runtime type: Python 3 (GPU)
- GPU Hardware Accelerator: NVIDIA Tesla T4 (15 GB VRAM)
- System Memory: 12.7 GB RAM

- Disk Storage: ~235 GB (available capacity ~ 39 GB used during experiments)

This setup provided reliable GPU acceleration for GAN training, enabling consistent timing measurements across GAN implementations.

6.1.2. Software Environment

The experiments were implemented using the following software stack:

- Python: 3.12
- PyTorch: 2.9.0—for GAN model implementation and training
- Scikit-learn: 1.6.1—for classifier training and evaluation
- Pandas: 2.2.2.3—for data preprocessing
- Numpy: 2.0.2—for numerical operations
- Matplotlib (3.10.0) and Seaborn (0.13.2)—for result visualization
- t-SNE (1.6.1)—for dimensionality reduction

6.1.3. Implementation Details

To improve computational efficiency, GPU acceleration was used when a CUDA-enabled device was available. Specifically, linear models, support vector machines, k-nearest neighbors, and random forest classifiers were implemented using the RAPIDS cuML library when GPU support was detected at runtime. When GPU resources were unavailable, functionally equivalent CPU-based implementations from scikit-learn were used. The decision tree classifier was implemented on a CPU-based model because there is no cuML equivalent. Notably, GPU acceleration was used solely to accelerate training and inference, without altering the model architecture, loss functions, hyperparameter settings, or evaluation procedures. All experiments were functionally equivalent across GPU and CPU implementations, and results were aggregated across cross-validation folds to minimize any variability introduced by hardware-dependent execution.

6.2. Data Partitioning and Cross-Validation

To ensure accurate and unbiased performance estimates, each binary dataset (majority class + one minority class) was evaluated using stratified 5-fold cross-validation, with identical fold assignments applied consistently across all minority classes (each with its own class-specific GAN), augmentation ratios, and classifiers. Stratification preserves the presence of minority-class samples in each fold, while preprocessing parameters (e.g., feature scaling) are estimated exclusively from the training folds and subsequently applied to the corresponding validation folds. This design prevents information leakage from the validation data into the training process.

During each cross-validation iteration, the model is trained on $k - 1$ folds and evaluated on the remaining fold ($k = 5$). Performance metrics are averaged across all folds to reduce variance and mitigate bias associated with a single train-test split. Confusion matrices and classification reports are generated to assess class-wise performance, with particular emphasis on minority attack classes.

Within each training fold, synthetic minority-class samples generated by the class-specific GAN trained for that minority class are appended only to the training data to form the augmentation training set. The validation fold remains unchanged and consists of real network traffic samples. GANs are trained solely on minority-class samples from the training partition and never observe validation or test data. Consequently, classifiers are evaluated only on unseen real samples, ensuring that performance improvements reflect genuine gains in generalization to rare attack behaviors rather than inflated evaluation results caused by data leakage or memorization of synthetic patterns.

This strict fold-level separation between GAN training, data augmentation, and classifier evaluation ensures that augmentation enhances model learning without artificially inflating reported performance metrics.

6.3. GAN Training Configuration

Each GAN was trained separately for each minority class. The training used the same hyperparameters across all variants unless a variant's specific constraints required otherwise.

Common Hyperparameters

- Batch size: 64
- Learning rate: 0.0002
- Optimizer: Adam ($\beta_1 = 0.5$, $\beta_2 = 0.999$)
- Input noise dimension (z): 32
- Feature normalization: all features scaled to $[-1, 1]$
- Epoch counts: 2000

The GAN models were trained using $[-1, 1]$ feature scaling, a latent noise dimension of 32 with uniform noise sampling, and a training schedule of 2000 epochs with a batch size of 64. The generator used no dropout, while the discriminator used a 0.5 dropout rate to prevent overfitting. Synthetic augmentation was performed with a 0.15 ratio, meaning the number of generated minority samples was set to 15% of the majority class size.

6.4. Synthetic Data Generation and Augmentation Ratios

After the GAN model converged, synthetic samples were produced using the trained generator. For each minority class, the GAN generated synthetic samples at a fixed ratio of the majority class size. Specifically, a ratio $r \in \{0.15\}$ was applied to the number of benign “none” samples, resulting in $r \times |majority|$ synthetic minority instances. This method ensures that even very sparse attack categories are augmented to a density comparable to that of the majority class, enabling stable training of both the GAN and downstream classifiers.

6.4.1. Ratio Definition

As shown in Equation 3, the ratio definition ensures sufficient synthetic density, even for minority classes with 1–10 real samples, thereby stabilizing adversarial learning and improving classifier discrimination.

6.4.2. Motivation

Because several minority attack types are extremely sparse, a ratio based on minority count would produce too few samples for stable GAN training. Using the majority count ensures sufficient synthetic coverage. The resulting synthetic populations (100 k–200 k samples) match the densities necessary for robust generative learning.

6.5. Structural Visualization

t-SNE Visualization of Real vs. Synthetic Samples

To qualitatively assess the geometric relationship between real and GAN-generated minority samples, this study employs a two-dimensional t-distributed Stochastic Neighbor Embedding (t-SNE) projection. t-SNE embeds high-dimensional feature vectors into a shared low-dimensional manifold while preserving local structure, making it well-suited for visual inspection of generative model behavior [53]. The visualization process takes high-dimensional feature vectors from both the real minority class and the synthetic minority samples. It embeds them in a shared 2D manifold, enabling the relative spatial arrangement of the points to be visually examined. To ensure computational feasibility while maintaining

representativeness, the function performs stratified subsampling when the combined sample size exceeds a configurable limit (default 5000 points), preserving the real-to-synthetic ratio during sampling. Real samples are emphasized by plotting them with larger markers and black outlines, while synthetic samples are rendered beneath them using semi-transparent markers. This design choice allows precise observation of whether synthetic samples cluster around real instances, disperse naturally across the minority manifold, or drift into unrealistic regions. By comparing point distributions and overlaps, t-SNE provides an intuitive geometric complement to qualitative fidelity metrics.

6.6. Classifier Training Setup

Five classical classifiers were trained for each minority class and each GAN.

- Logistic Regression
- Support Vector Machine (SVM)
- K-Nearest Neighbor (KNN)
- Decision Tree
- Random Forest

6.6.1. Evaluation Metrics

Various evaluation metrics were used to assess classifier performance in imbalanced settings, including Accuracy, Precision (macro), Recall (macro), F1-Score (macro), AUC-ROC, and Confusion Matrix. For these calculations, we need to identify four terms [54] (p. 365):

- True positives (TP)—refer to positive tuples that were correctly labeled by the classifier.
- True negative (TN)—refers to the negative tuples correctly identified by the classifier.
- False positives (FP)—refer to negative tuples that were incorrectly labeled as positive by the classifier.
- False negative (FN)—refers to positive tuples incorrectly labeled as negative by the classifier.

Given the severe class imbalance in the dataset, the evaluation focused on metrics sensitive to the minority class. Precision, recall, and F1-score were used to assess detection accuracy for rare attack classes, while AUC-ROC provides a threshold-independent measure of separability. Confusion matrices were included to explicitly analyze false negatives and false positives, which are critical in intrusion detection contexts.

Accuracy

Accuracy provides a general measure of overall correctness but can be misleading in highly imbalanced datasets where a model may achieve high accuracy by predicting the majority class [51].

$$Accuracy = \frac{TP + TN}{P + N} \quad (4)$$

Precision

Precision measures the proportion of predicted positive samples that are truly positive [52]. High precision indicates that the classifier produces few false positives for the minority class.

$$Precision = \frac{TP}{TP + FP} \quad (5)$$

Recall

Recall gauges the percentage of true minority-class samples correctly identified [52]. Since false negatives represent missed attacks, recall is a crucial metric in cybersecurity and is a primary focus of this study.

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

F1-Score

The F1-score is the harmonic mean of precision and recall [52]. It provides a balanced measure of classifier performance, especially under skewed distributions. Improvement in the minority-class F1-score after augmentation indicates more effective decision boundaries.

$$F1 = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \quad (7)$$

AUC-ROC

The Area Under the Receiver Operating Characteristic Curve (AUC-ROC) measures a classifier's ability to discriminate between positive and negative classes across various decision thresholds [52]. It facilitates comparisons of models independently of absolute boundaries.

Confusion Matrix Analysis

The confusion matrix assesses the accuracy with which a classifier identifies tuples from different classes [52]. TP and TN denote correct predictions, whereas FP and FN denote errors. The confusion matrix table is shown in Table 5. Reducing FN is especially important in intrusion detection because such errors allow attacks to go undetected. GAN-based augmentation aims to increase TP while decreasing FN without causing an unacceptable increase in FP.

Table 5. Confusion Matrix [52].

		Predicted Label		
		yes	no	Total
True Label	yes	TP	FN	P
	no	FP	TN	N
	total	P'	N'	P + N

6.6.2. Evaluation Protocol

- Binary Classification—Each experiment involved training a binary classifier for each single MITRE ATT&CK tactic against the “none” class.
- Metrics—Accuracy, Precision, Recall, F1-Score, and AUC-ROC were computed for each fold, and the average across folds was reported.
- Visualization—t-SNE plots were used to assess the structure of real vs. synthetic samples quantitatively, and confusion matrices were used to inspect classifier performance.

6.7. Reproducibility and Consistency

To ensure experimental reproducibility, all random seeds were fixed (for PyTorch, NumPy, and scikit-learn), fold splits were kept constant across experiments, identical preprocessing was applied across GAN training, synthetic data were generated only from the training folds, and classifier hyperparameters were fixed across all augmentation conditions.

7. Results

This section presents the comprehensive evaluation of GAN-based augmentation across all attack types in the UWF-ZeekData22 dataset [16,17]. Each minority class is examined individually, and classifier performance is evaluated before and after GAN augmentation, with a detailed analysis of the confusion matrix. Because the experimental design generates a large number of synthetic minority samples ($r \cdot |D_{maj}|$), the augmented datasets substantially increase the separability between the minority and majority classes. Consistent with the methodology, each subsection includes individual table references to maintain explicit clarity and support fine-grained interpretation.

7.1. Class Distribution Before and After Augmentation

The initial class distribution is highly imbalanced, with the “none” class comprising more than 99% of all samples and the rarest tactics (e.g., Credential Access and Privilege Escalation) representing less than 0.02% (Table 2). Figure 6a visualizes this extreme imbalance prior to augmentation, illustrating the dominance of the majority class and the severe sparsity of several ATT&CK tactics.

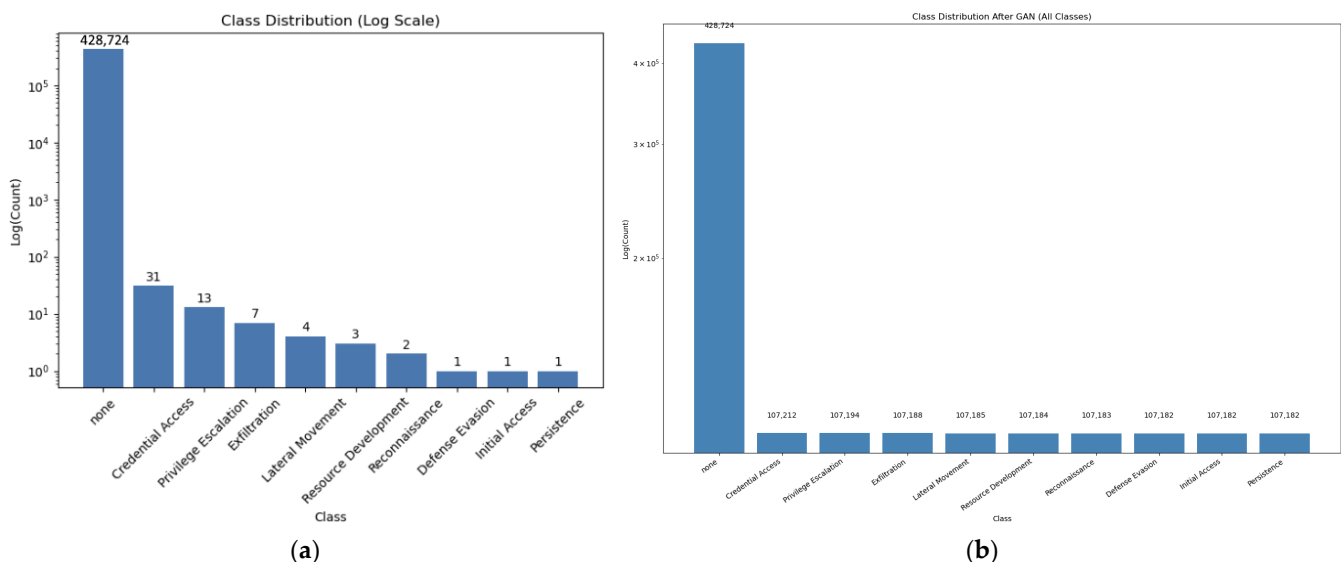


Figure 6. Class Distribution: (a) before GAN; (b) after GAN.

With $r = 0.15$, class-specific GAN augmentation increases the representation of each minority class to approximately 15% of that of the majority class. As shown in Figure 6b, the augmented dataset exhibits a substantially more balanced class distribution, reducing the effect of the majority-minority ratio and improving class separability for downstream classifiers.

7.2. Classifier Performance (Cross-Validation)

Before GAN augmentation, a binary classification task was performed for each minority class versus the majority class (none), and various evaluation metrics were compared. Tables 6–14 present the results of both pre-GAN and post-GAN augmentation, comparing macro-averaged Accuracy, Precision, F1-score, and AUC-ROC across 5-fold stratified cross-validation for five classifiers: Logistic Regression, SVM, KNN, Decision Tree, and Random Forest. For the SVM classifier, AUC-ROC values are not reported because probability calibration was not applied. Under extreme class sparsity, reliable calibration is unstable, rendering AUC-ROC values unreliable.

Table 6. Averaged Metrics Cross-Validation Across Folds for Credential Access vs. None.

Classifier	Augmentation	Accuracy	Precision	Recall	F1-Score	AUC-ROC *
Logistic Regression	Pre-GAN	0.9985	0.6074	0.9826	0.6267	0.9970
	Post-GAN	1.0000	1.0000	1.0000	1.0000	1.0000
SVM	Pre-GAN	1.0000	1.0000	0.9524	0.9741	-
	Post-GAN	1.0000	1.0000	0.9999	1.0000	-
KNN	Pre-GAN	1.0000	0.9750	1.0000	0.9857	1.0000
	Post-GAN	1.0000	1.0000	1.0000	1.0000	1.0000
Decision Tree	Pre-GAN	1.0000	1.0000	0.9833	0.9909	0.9833
	Post-GAN	1.0000	1.0000	1.0000	1.0000	1.0000
Random Forest	Pre-GAN	1.0000	1.0000	1.0000	1.0000	1.0000
	Post-GAN	1.0000	1.0000	1.0000	1.0000	1.0000

* Note: “-” denotes metric unavailable due to insufficient minority-class samples; AUC-ROC is not reported for the SVM classifier because probability estimates were not enabled.

Table 7. Averaged Metrics Cross-Validation Across Folds for Privilege Escalation vs. None.

Classifier	Augmentation	Accuracy	Precision	Recall	F1-Score	AUC-ROC *
Logistic Regression	Pre-GAN	1.0000	0.8917	1.0000	0.9324	1.0000
	Post-GAN	1.0000	1.0000	1.0000	1.0000	1.0000
SVM	Pre-GAN	1.0000	1.0000	0.9500	0.9667	-
	Post-GAN	1.0000	1.0000	1.0000	1.0000	-
KNN	Pre-GAN	1.0000	0.9500	0.9333	0.9267	1.0000
	Post-GAN	1.0000	1.0000	1.0000	1.0000	1.0000
Decision Tree	Pre-GAN	1.0000	1.0000	0.9333	0.9500	0.9333
	Post-GAN	1.0000	1.0000	1.0000	1.0000	1.0000
Random Forest	Pre-GAN	1.0000	1.0000	0.9333	0.9500	1.0000
	Post-GAN	1.0000	1.0000	1.0000	1.0000	1.0000

* Note: “-” denotes metric unavailable due to insufficient minority-class samples; AUC-ROC is not reported for the SVM classifier because probability estimates were not enabled.

Table 8. Averaged Metrics Cross-Validation Across Folds for Exfiltration vs. None.

Classifier	Augmentation	Accuracy	Precision	Recall	F1-Score	AUC-ROC *
Logistic Regression	Pre-GAN	1.0000	0.8167	0.9500	0.8567	0.9986
	Post-GAN	1.0000	1.0000	1.0000	1.0000	1.0000
SVM	Pre-GAN	1.0000	1.0000	0.9000	0.9333	-
	Post-GAN	1.0000	1.0000	1.0000	1.0000	-
KNN	Pre-GAN	1.0000	1.0000	0.9000	0.9333	0.9500
	Post-GAN	1.0000	1.0000	1.0000	1.0000	1.0000
Decision Tree	Pre-GAN	1.0000	1.0000	1.0000	1.0000	1.0000
	Post-GAN	1.0000	1.0000	1.0000	1.0000	1.0000
Random Forest	Pre-GAN	1.0000	1.0000	0.9500	0.9667	1.0000
	Post-GAN	1.0000	1.0000	1.0000	1.0000	1.0000

* Note: “-” denotes metric unavailable due to insufficient minority-class samples; AUC-ROC is not reported for the SVM classifier because probability estimates were not enabled.

Table 9. Averaged Metrics Cross-Validation Across Folds for Lateral Movement vs. None.

Classifier	Augmentation	Accuracy	Precision	Recall	F1-Score	AUC-ROC
Logistic Regression	Pre-GAN	-	-	-	-	-
	Post-GAN	1.0000	1.0000	1.0000	1.0000	1.0000
SVM	Pre-GAN	1.0000	1.0000	1.0000	1.0000	-
	Post-GAN	1.0000	1.0000	1.0000	1.0000	-
KNN	Pre-GAN	1.0000	0.6000	0.6000	0.6000	1.0000
	Post-GAN	1.0000	1.0000	1.0000	1.0000	1.0000
Decision Tree	Pre-GAN	1.0000	1.0000	1.0000	1.0000	1.0000
	Post-GAN	1.0000	1.0000	1.0000	1.0000	1.0000
Random Forest	Pre-GAN	1.0000	1.0000	1.0000	1.0000	1.0000
	Post-GAN	1.0000	1.0000	1.0000	1.0000	1.0000

* Note: “-” denotes metric unavailable due to insufficient minority-class samples; AUC-ROC is not reported for the SVM classifier because probability estimates were not enabled.

Table 10. Averaged Metrics Cross-Validation Across Folds for Resource Development vs. None.

Classifier	Augmentation	Accuracy *	Precision *	Recall *	F1-Score *	AUC-ROC *
Logistic Regression	Pre-GAN	-	-	-	-	-
	Post-GAN	1.0000	1.0000	1.0000	1.0000	1.0000
SVM	Pre-GAN	1.0000	0.8000	0.8000	0.8000	-
	Post-GAN	1.0000	1.0000	1.0000	1.0000	-
KNN	Pre-GAN	1.0000	0.7000	0.7000	0.7000	0.8333
	Post-GAN	1.0000	1.0000	1.0000	1.0000	1.0000
Decision Tree	Pre-GAN	1.0000	0.9000	0.9000	0.9000	0.8333
	Post-GAN	1.0000	1.0000	1.0000	1.0000	1.0000
Random Forest	Pre-GAN	1.0000	0.8000	0.8000	0.8000	1.0000
	Post-GAN	1.0000	1.0000	1.0000	1.0000	1.0000

* Note: “-” denotes metric unavailable due to insufficient minority-class samples; AUC-ROC is not reported for the SVM classifier because probability estimates were not enabled.

Table 11. Averaged Metrics Cross-Validation Across Folds for Reconnaissance vs. None.

Classifier	Augmentation	Accuracy *	Precision *	Recall *	F1-Score *	AUC-ROC *
Logistic Regression	Pre-GAN	-	-	-	-	-
	Post-GAN	1.0000	1.0000	1.0000	1.0000	1.0000
SVM	Pre-GAN	-	-	-	-	-
	Post-GAN	1.0000	1.0000	1.0000	1.0000	-
KNN	Pre-GAN	1.0000	0.8000	0.8000	0.8000	1.0000
	Post-GAN	1.0000	1.0000	1.0000	1.0000	1.0000
Decision Tree	Pre-GAN	1.0000	1.0000	1.0000	1.0000	1.0000
	Post-GAN	1.0000	1.0000	1.0000	1.0000	1.0000
Random Forest	Pre-GAN	1.0000	1.0000	1.0000	1.0000	1.0000
	Post-GAN	1.0000	1.0000	1.0000	1.0000	1.0000

* Note: “-” denotes metric unavailable due to insufficient minority-class samples; AUC-ROC is not reported for the SVM classifier because probability estimates were not enabled.

Table 12. Averaged Metrics Cross-Validation Across Folds for Defense Evasion vs. None.

Classifier	Augmentation	Accuracy *	Precision *	Recall *	F1-Score *	AUC-ROC *
Logistic Regression	Pre-GAN	-	-	-	-	-
	Post-GAN	1.0000	1.0000	1.0000	1.0000	1.0000
SVM	Pre-GAN	-	-	-	-	-
	Post-GAN	1.0000	1.0000	1.0000	1.0000	-
KNN	Pre-GAN	1.0000	0.9000	0.9000	0.9000	-
	Post-GAN	1.0000	1.0000	1.0000	1.0000	1.0000
Decision Tree	Pre-GAN	1.0000	0.9000	0.9000	0.9000	-
	Post-GAN	1.0000	1.0000	1.0000	1.0000	1.0000
Random Forest	Pre-GAN	1.0000	0.9000	0.9000	0.9000	-
	Post-GAN	1.0000	1.0000	1.0000	1.0000	1.0000

* Note: “-” denotes metric unavailable due to insufficient minority-class samples; AUC-ROC is not reported for the SVM classifier because probability estimates were not enabled.

Table 13. Averaged Metrics Cross-Validation Across Folds for Initial Access vs. None.

Classifier	Augmentation	Accuracy *	Precision *	Recall *	F1-Score *	AUC-ROC *
Logistic Regression	Pre-GAN	-	-	-	-	-
	Post-GAN	1.0000	1.0000	1.0000	1.0000	1.0000
SVM	Pre-GAN	-	-	-	-	-
	Post-GAN	1.0000	1.0000	1.0000	1.0000	-
KNN	Pre-GAN	1.0000	0.9000	0.9000	0.9000	-
	Post-GAN	1.0000	1.0000	1.0000	1.0000	1.0000
Decision Tree	Pre-GAN	1.0000	0.9000	0.9000	0.9000	-
	Post-GAN	1.0000	1.0000	1.0000	1.0000	1.0000
Random Forest	Pre-GAN	1.0000	0.9000	0.9000	0.9000	-
	Post-GAN	1.0000	1.0000	1.0000	1.0000	1.0000

* Note: “-” denotes metric unavailable due to insufficient minority-class samples; AUC-ROC is not reported for the SVM classifier because probability estimates were not enabled.

Table 14. Averaged Metrics Cross-Validation Across Folds for Persistence vs. None.

Classifier	Augmentation	Accuracy *	Precision *	Recall *	F1-Score *	AUC-ROC *
Logistic Regression	Pre-GAN	-	-	-	-	-
	Post-GAN	1.0000	1.0000	1.0000	1.0000	1.0000
SVM	Pre-GAN	-	-	-	-	-
	Post-GAN	1.0000	1.0000	1.0000	1.0000	-
KNN	Pre-GAN	1.0000	0.9000	0.9000	0.9000	-
	Post-GAN	1.0000	1.0000	1.0000	1.0000	1.0000
Decision Tree	Pre-GAN	1.0000	0.9000	0.9000	0.9000	-
	Post-GAN	1.0000	1.0000	1.0000	1.0000	1.0000
Random Forest	Pre-GAN	1.0000	0.9000	0.9000	0.9000	-
	Post-GAN	1.0000	1.0000	1.0000	1.0000	1.0000

* Note: “-” denotes metric unavailable due to insufficient minority-class samples; AUC-ROC is not reported for the SVM classifier because probability estimates were not enabled.

To prevent data leakage, GAN-based augmentation was performed independently within each cross-validation fold. For each fold, the GAN was trained exclusively on minority-class samples from the training split, and synthetic samples were generated solely to augment the corresponding training data. Validation folds contained only real samples and were never used during training or data generation, ensuring strict separation between training and evaluation data.

For several extremely underrepresented attack classes, pre-GAN evaluation was infeasible because at least one cross-validation contained no minority-class samples. In such cases, results are denoted by “-”, indicating that the corresponding metric could not be reliably computed and was therefore intentionally omitted.

7.2.1. Minority-Class Performance Improves Pos-GAN

Across most attack categories and classifiers, we see improvements in Precision, Recall, and F1-score after GAN augmentation. The most significant gains are in Logistic Regression, where the F1-score rises from about 0.62 before GAN to 1.00 after GAN (Table 6), and in KNN, with initial F1-scores of 0.60 for Lateral Movement (Table 9), increasing to a perfect 1.00 after GAN.

7.2.2. Consistent Accuracy Across All Tasks

The overall accuracy of all classifiers remained at 1.00, even after the addition of synthetic data to the dataset. This shows that as the detection rate for minority classes increased, the performance of the majority class (none) remained unchanged, a key property of real-world intrusion detection systems. This stability in accuracy, along with better macro-Recall and F1-scores, indicates that improvements in minority classes did not reduce the performance of the majority (“none”) class.

7.2.3. AUC-ROC Indicates Improved Separation

Where applicable, the AUC-ROC scores reached 1.00 or improved after augmentation. For example, in Table 6 for Logistic Regression with Credential Access, the AUC-ROC score rose from 0.9970 to 1.000. These improvements demonstrate increased separation between attack traffic and normal (none) traffic after augmentation.

7.2.4. Patterns by Classifier

Tables 6–14 show the patterns by classifiers. Across classifiers, Decision Tree and Random Forest perform well even without GAN, and GAN further enhances their performance through augmentation. Meanwhile, Logistic Regression and KNN exhibit significant fluctuations in recall and F1-scores due to their sensitivity to imbalanced data. In contrast, SVM generally begins with high F1-scores in the mid-to-high 0.9 range and often reaches 0.99–1.00 after augmentation.

In Tables 6–14, “-” indicates that the corresponding metric could not be reliably computed in the pre-GAN setting because there were insufficient minority-class samples in at least one cross-validation fold. AUC-ROC is not reported for the SVM classifier because probability estimates were not enabled; under extreme class sparsity, probability calibration may be unreliable.

7.2.5. Tactic-Specific Observations

Credential Access

Before augmentation, classifiers such as Logistic Regression had relatively low F1-scores despite high recall. After augmentation, all classifiers achieved an F1-score of 1.00, indicating that the synthetic samples improved the models’ ability to generalize and distinguish the minority class.

Privilege Escalation

This tactic type showed strong recall among classifiers in the pre-GAN phase, but its precision was occasionally weaker (e.g., Logistic Regression Precision = 0.8917). After augmentation, all metrics, including precision, improved across all classifiers, resulting in a perfect F1-score.

Exfiltration

Pre-GAN Logistic Regression and KNN perform slightly worse with F1-scores between 0.85 and 0.93 due to moderate precision. After using a GAN, all classifiers achieved an F1-score of 1.00, underscoring the effectiveness of data augmentation with synthetic samples.

Lateral Movement

KNN performance was weak before augmentation, with an F1-score of 0.6000. Also, Logistic Regression was not evaluated pre-GAN, likely due to the extremely small number of training samples for this class. After GAN augmentation, KNN performance improved significantly (F1-score from 0.6000 to 1.0000), indicating that the model benefited from synthetic data augmentation.

Resource Development

Pre-GAN metrics indicate that several classifiers, such as KNN and Logistic Regression, either performed poorly or returned unavailable results due to limited sample sizes. For example, KNN achieved an F1-score of 0.7000. After GAN augmentation, all classifiers reported an F1-score of 1.0000, indicating that augmentation improved performance and corrected underperforming classifiers due to data scarcity.

Reconnaissance

Classifiers such as Logistic Regression and SVMs were not evaluated pre-GAN due to insufficient training samples for this class. After augmentation, GAN-generated samples improved classifier performance, with all classifiers achieving an F1-score of 1.00. This demonstrates the effectiveness of GAN-generated data augmentation in improving model performance.

Defense Evasion

As with Reconnaissance, Defense Evasion lacked evaluation results from the pre-GAN phase for Logistic Regression and SVM. Post-augmentation showed significant improvement across all classifiers, confirming the effectiveness of GAN in managing underrepresented classes.

Initial Access

Pre-GAN evaluations for Logistic Regression and SVM were missing, likely due to the limited sample size for this class. After augmentation, all classifiers achieved an F1-score of 1.00, demonstrating the benefit of synthetic data generation for minority classes.

Persistence

Pre-GAN evaluations for Logistic Regression and Support Vector Machine (SVM) were unavailable, likely due to the small sample size in this class. After data augmentation, all classifiers achieved an F1-score of 1.00, demonstrating the benefits of synthetic data generation for minority classes.

7.3. Confusion Matrices

The confusion matrix results for each evaluated ATT&CK tactic are presented in Tables 15–21. In intrusion detection systems, false negatives are particularly critical because undetected attacks can lead to severe security consequences; therefore, reduction in false negatives after augmentation represents a meaningful improvement in IDS effectiveness.

Table 15. Pre-GAN vs. post-GAN confusion matrices for Credential Access across classifiers (rows = actual, columns = predicted; CA = Credential Access, None = Majority class).

Classifier	Pre-GAN Confusion Matrix	Post-GAN Confusion Matrix
Logistic Regression	$\begin{bmatrix} 30 & 1 \\ 642 & 428,082 \end{bmatrix}$	$\begin{bmatrix} 107,212 & 0 \\ 0 & 428,724 \end{bmatrix}$
SVM	$\begin{bmatrix} 28 & 3 \\ 0 & 428,724 \end{bmatrix}$	$\begin{bmatrix} 107,195 & 17 \\ 0 & 428,724 \end{bmatrix}$
KNN	$\begin{bmatrix} 31 & 0 \\ 2 & 428,722 \end{bmatrix}$	$\begin{bmatrix} 107,204 & 8 \\ 0 & 428,724 \end{bmatrix}$
Decision Tree	$\begin{bmatrix} 30 & 1 \\ 0 & 428,724 \end{bmatrix}$	$\begin{bmatrix} 107,212 & 0 \\ 0 & 428,724 \end{bmatrix}$
Random Forest	$\begin{bmatrix} 31 & 0 \\ 0 & 428,724 \end{bmatrix}$	$\begin{bmatrix} 107,212 & 0 \\ 0 & 428,724 \end{bmatrix}$

Table 16. Pre-GAN vs. post-GAN confusion matrices for Privilege Escalation across classifiers (rows = actual, columns = predicted; PA = Privilege Escalation, None = Majority class).

Classifier	Pre-GAN Confusion Matrix	Post-GAN Confusion Matrix
Logistic Regression	$\begin{bmatrix} 13 & 0 \\ 4 & 428,720 \end{bmatrix}$	$\begin{bmatrix} 107,194 & 0 \\ 0 & 428,724 \end{bmatrix}$
SVM	$\begin{bmatrix} 12 & 1 \\ 0 & 428,724 \end{bmatrix}$	$\begin{bmatrix} 107,190 & 4 \\ 0 & 428,724 \end{bmatrix}$
KNN	$\begin{bmatrix} 11 & 2 \\ 2 & 428,722 \end{bmatrix}$	$\begin{bmatrix} 107,189 & 5 \\ 2 & 428,722 \end{bmatrix}$
Decision Tree	$\begin{bmatrix} 11 & 2 \\ 0 & 428,724 \end{bmatrix}$	$\begin{bmatrix} 107,193 & 1 \\ 0 & 428,724 \end{bmatrix}$
Random Forest	$\begin{bmatrix} 11 & 2 \\ 0 & 428,724 \end{bmatrix}$	$\begin{bmatrix} 107,194 & 0 \\ 0 & 428,724 \end{bmatrix}$

Table 17. Pre-GAN vs. post-GAN confusion matrices for Exfiltration across classifiers (rows = actual, columns = predicted; EX = Exfiltration, None = Majority class).

Classifier	Pre-GAN Confusion Matrix	Post-GAN Confusion Matrix
Logistic Regression	$\begin{bmatrix} 6 & 1 \\ 5 & 428,719 \end{bmatrix}$	$\begin{bmatrix} 107,185 & 3 \\ 0 & 428,724 \end{bmatrix}$
SVM	$\begin{bmatrix} 5 & 2 \\ 0 & 428,724 \end{bmatrix}$	$\begin{bmatrix} 107,184 & 4 \\ 0 & 428,724 \end{bmatrix}$
KNN	$\begin{bmatrix} 5 & 2 \\ 0 & 428,724 \end{bmatrix}$	$\begin{bmatrix} 107,185 & 3 \\ 2 & 428,722 \end{bmatrix}$
Decision Tree	$\begin{bmatrix} 7 & 0 \\ 0 & 428,724 \end{bmatrix}$	$\begin{bmatrix} 107,187 & 1 \\ 0 & 428,724 \end{bmatrix}$
Random Forest	$\begin{bmatrix} 6 & 1 \\ 0 & 428,724 \end{bmatrix}$	$\begin{bmatrix} 107,187 & 1 \\ 0 & 428,724 \end{bmatrix}$

Table 18. Pre-GAN vs. post-GAN confusion matrices for Lateral Movement across classifiers (rows = actual, columns = predicted; LM = Lateral Movement, None = Majority class).

Classifier	Pre-GAN Confusion Matrix	Post-GAN Confusion Matrix
Logistic Regression	-	$\begin{bmatrix} 107,182 & 3 \\ 0 & 428,724 \end{bmatrix}$
SVM	$\begin{bmatrix} 4 & 0 \\ 0 & 428,724 \end{bmatrix}$	$\begin{bmatrix} 107,182 & 3 \\ 0 & 428,724 \end{bmatrix}$
KNN	$\begin{bmatrix} 0 & 4 \\ 0 & 428,724 \end{bmatrix}$	$\begin{bmatrix} 107,181 & 4 \\ 0 & 428,724 \end{bmatrix}$
Decision Tree	$\begin{bmatrix} 4 & 0 \\ 0 & 428,724 \end{bmatrix}$	$\begin{bmatrix} 107,185 & 0 \\ 0 & 428,724 \end{bmatrix}$
Random Forest	$\begin{bmatrix} 4 & 0 \\ 0 & 428,724 \end{bmatrix}$	$\begin{bmatrix} 107,185 & 0 \\ 0 & 428,724 \end{bmatrix}$

Table 19. Pre-GAN vs. post-GAN confusion matrices for Resource Development across classifiers (rows = actual, columns = predicted; RD = Resource Development, None = Majority class).

Classifier	Pre-GAN Confusion Matrix	Post-GAN Confusion Matrix
Logistic Regression	-	$\begin{bmatrix} 107,183 & 1 \\ 2 & 428,722 \end{bmatrix}$
SVM	$\begin{bmatrix} 1 & 2 \\ 0 & 428,724 \end{bmatrix}$	$\begin{bmatrix} 107,182 & 2 \\ 0 & 428,724 \end{bmatrix}$
KNN	$\begin{bmatrix} 0 & 3 \\ 0 & 428,724 \end{bmatrix}$	$\begin{bmatrix} 107,181 & 3 \\ 0 & 428,724 \end{bmatrix}$
Decision Tree	$\begin{bmatrix} 2 & 1 \\ 0 & 428,724 \end{bmatrix}$	$\begin{bmatrix} 107,184 & 0 \\ 0 & 428,724 \end{bmatrix}$
Random Forest	$\begin{bmatrix} 1 & 2 \\ 0 & 428,724 \end{bmatrix}$	$\begin{bmatrix} 107,183 & 1 \\ 0 & 428,724 \end{bmatrix}$

Table 20. Pre-GAN vs. post-GAN confusion matrices for Reconnaissance across classifiers (rows = actual, columns = predicted; RE = Reconnaissance, None = Majority class).

Classifier	Pre-GAN Confusion Matrix	Post-GAN Confusion Matrix
Logistic Regression	-	$\begin{bmatrix} 107,183 & 0 \\ 0 & 428,724 \end{bmatrix}$
SVM	-	$\begin{bmatrix} 107,183 & 0 \\ 0 & 428,724 \end{bmatrix}$
KNN	$\begin{bmatrix} 0 & 2 \\ 0 & 428,724 \end{bmatrix}$	$\begin{bmatrix} 107,181 & 2 \\ 0 & 428,724 \end{bmatrix}$
Decision Tree	$\begin{bmatrix} 2 & 0 \\ 0 & 428,724 \end{bmatrix}$	$\begin{bmatrix} 107,183 & 0 \\ 0 & 428,724 \end{bmatrix}$
Random Forest	$\begin{bmatrix} 2 & 1 \\ 0 & 428,724 \end{bmatrix}$	$\begin{bmatrix} 107,183 & 0 \\ 0 & 428,724 \end{bmatrix}$

In Tables 18–23, a dash (“-”) in the pre-GAN confusion matrix denotes cases where evaluation could not be performed due to extreme minority-class sparsity. For several attack tactics, the number of real minority samples was insufficient to support stable classifier training or stratified cross-validation, particularly for linear and margin-based models such as Logistic Regression and SVM. As a result, confusion matrices were undefined prior to

augmentation. After GAN-based augmentation, sufficient minority samples are available, enabling stable training and evaluation across all classifiers.

Table 21. Pre-GAN vs. post-GAN confusion matrices for Defense Evasion across classifiers (rows = actual, columns = predicted; DE = Defense Evasion, None = Majority class).

Classifier	Pre-GAN Confusion Matrix	Post-GAN Confusion Matrix
Logistic Regression	-	$\begin{bmatrix} 107,181 & 1 \\ 0 & 428,724 \end{bmatrix}$
SVM	-	$\begin{bmatrix} 107,181 & 1 \\ 0 & 428,724 \end{bmatrix}$
KNN	$\begin{bmatrix} 0 & 1 \\ 0 & 428,724 \end{bmatrix}$	$\begin{bmatrix} 107,181 & 1 \\ 0 & 428,724 \end{bmatrix}$
Decision Tree	$\begin{bmatrix} 0 & 1 \\ 0 & 428,724 \end{bmatrix}$	$\begin{bmatrix} 107,181 & 1 \\ 0 & 428,724 \end{bmatrix}$
Random Forest	$\begin{bmatrix} 0 & 1 \\ 0 & 428,724 \end{bmatrix}$	$\begin{bmatrix} 107,181 & 1 \\ 0 & 428,724 \end{bmatrix}$

Table 22. Pre-GAN vs. post-GAN confusion matrices for Initial Access across classifiers (rows = actual, columns = predicted; IA = Initial Access, None = Majority class).

Classifier	Pre-GAN Confusion Matrix	Post-GAN Confusion Matrix
Logistic Regression	-	$\begin{bmatrix} 107,181 & 1 \\ 0 & 428,724 \end{bmatrix}$
SVM	-	$\begin{bmatrix} 107,181 & 1 \\ 0 & 428,724 \end{bmatrix}$
KNN	$\begin{bmatrix} 0 & 1 \\ 0 & 428,724 \end{bmatrix}$	$\begin{bmatrix} 107,181 & 1 \\ 0 & 428,724 \end{bmatrix}$
Decision Tree	$\begin{bmatrix} 0 & 1 \\ 0 & 428,724 \end{bmatrix}$	$\begin{bmatrix} 107,181 & 1 \\ 0 & 428,724 \end{bmatrix}$
Random Forest	$\begin{bmatrix} 0 & 1 \\ 0 & 428,724 \end{bmatrix}$	$\begin{bmatrix} 107,181 & 1 \\ 0 & 428,724 \end{bmatrix}$

Table 23. Pre-GAN vs. post-GAN confusion matrices for Persistence across classifiers (rows = actual, columns = predicted; PS = Persistence, None = Majority class).

Classifier	Pre-GAN Confusion Matrix	Post-GAN Confusion Matrix
Logistic Regression	-	$\begin{bmatrix} 107,181 & 1 \\ 0 & 428,724 \end{bmatrix}$
SVM	-	$\begin{bmatrix} 107,181 & 1 \\ 0 & 428,724 \end{bmatrix}$
KNN	$\begin{bmatrix} 0 & 1 \\ 0 & 428,724 \end{bmatrix}$	$\begin{bmatrix} 107,181 & 1 \\ 0 & 428,724 \end{bmatrix}$
Decision Tree	$\begin{bmatrix} 0 & 1 \\ 0 & 428,724 \end{bmatrix}$	$\begin{bmatrix} 107,181 & 1 \\ 0 & 428,724 \end{bmatrix}$
Random Forest	$\begin{bmatrix} 0 & 1 \\ 0 & 428,724 \end{bmatrix}$	$\begin{bmatrix} 107,181 & 1 \\ 0 & 428,724 \end{bmatrix}$

7.3.1. Credential Access

Before GAN augmentation, confusion matrices for several classifiers indicate difficulty distinguishing the Credential Access class from the majority class, as shown in Table 15.

Logistic Regression tends to favor the majority class (none), leading to skewed predictions and poor generalization for the minority class. In logistic regression, false positives are prevalent: 642 benign “none” samples are misclassified as Credential Access.

Following GAN augmentation, minority detection improved across all classifiers. Logistic Regression showed the most significant improvement in detection rates for both minority and majority classes, as shown in Table 15. The classifier perfectly distinguished Credential Access from the majority class (none). There were no false negatives; all 107,212 Credential Access samples were correctly classified, and no false positives or misclassified “none” samples occurred. This indicates that GAN-generated samples helped the classifier learn better class boundaries and created a more representative, balanced training dataset.

Decision Trees, already high-performing pre-GAN, benefited from the introduction of GAN-generated Credential Access samples, which eliminated misclassification errors. The model retained its perfect precision on the majority class while overcoming the single false negative in the pre-GAN. The result confirms that classifiers that already perform well can benefit from synthetic augmentation by eliminating rare misclassifications.

Random Forest maintained an F1-score of 1.00 even before augmentation, showcasing its strong resistance to class imbalance. After GAN-based augmentation, the model continues to classify both classes flawlessly, confirming that GAN-based data augmentation does not weaken high-performing models. This result also highlights the generalizability and stability of the proposed pipeline for generating additional training data, particularly for minority classes affected by class imbalance.

7.3.2. Privilege Escalation

GAN-based augmentation substantially improved classifier performance on the Privilege Escalation class, as shown in Table 16. Before augmentation, the Logistic Regression model was effective at identifying Privilege Escalation samples but misclassified four benign records. After augmentation, the classifier improved the decision boundary by reducing false positives, achieving an F1 Score of 1.00.

The linear SVM model missed one Privilege Escalation case. After augmentation, the classifier misclassified only 4 out of over 10,700 Privilege Escalation instances, maintaining near-perfect performance on this substantially expanded minority set. This shows that GAN augmentation improves the SVM’s generalization to underrepresented minority attack types in the cybersecurity dataset.

KNN struggled with both types of errors (false positives and false negatives), likely due to the sparsity of data points for the minority class. Post-GAN, although two false positives remained, the increased diversity of synthetic samples allowed the model to predict all Privileged Escalation instances, with only 5 cases missed out of 107,194. By enriching the local neighborhoods, GAN augmentation makes KNN a more robust classifier, especially for minority classes.

The Decision Tree also improved through augmentation. Before adding synthetic samples, the model exhibited two false negatives due to limited exposure to minority classes. After incorporating more than 100,000 synthetic samples generated by a GAN, the model correctly identified all but one Privilege Escalation case. These findings indicate that decision tree models benefit from GAN-enhanced class balancing, thereby improving their generalizability and stability.

Random Forest, already strong before GAN, still failed to identify two Privilege Escalation instances. Post-GAN, the model eliminated misclassification thanks to the diverse set of synthetic minority-class samples. These results demonstrate the potential of GAN-based augmentation not only to improve underperforming classifiers but also to

enhance the reliability of high-performing models, such as Random Forests, in imbalanced cybersecurity detection tasks.

7.3.3. Exfiltration

GAN-based augmentation improved minority class detection for Exfiltration, as summarized in Table 17. Before augmentation, the model had both false positives and false negatives. After applying GAN augmentation, the model correctly classifies all benign cases and accurately detects most Exfiltration instances, with only three false negatives. This shows that GAN-based augmentation improves minority-class detection for linear classifiers.

SVM classifiers also benefited from GAN-based augmentation. In the pre-GAN phase, the model showed perfect precision but missed two Exfiltration samples due to limited training data for that class. After augmentation, the model recovers nearly all Exfiltration instances with only four false negatives. This demonstrates that GAN augmentation improved the decision boundary, enabling the model to generalize more effectively to minority classes in highly imbalanced settings.

KNN's performance on Exfiltration class detection also improved with GAN-based augmentation. Before augmentation, the model was accurate but had limited recall due to the shortage of samples. After adding over 100,000 synthetic Exfiltration samples, the model recovers nearly all true positives, with only three false negatives. Although two false positives were introduced, the augmentation significantly improved overall classification performance.

The Decision Tree classifier performed very well on Exfiltration, even before augmentation, achieving 100% accuracy on the small minority-class sample set. After adding more than 100,000 synthetic Exfiltration samples, the model still achieved nearly perfect performance, misclassifying only one sample. This indicates that GAN-based augmentation helps maintain high performance, preventing the decision tree from overfitting to the limited dataset in the pre-GAN stage.

The Random Forest classifier, already highly accurate in detecting Exfiltration Pre-GAN, misclassifies only one instance, and continues to perform well after GAN-based augmentation. The model continues to misclassify only a single Exfiltration instance and maintains zero false positives, confirming that GAN-based augmentation preserves its already strong performance.

7.3.4. Lateral Movement

GAN-based augmentation significantly improved the classifiers' reliability for Lateral Movement, as shown in Table 18. Pre-GAN classifiers were limited by a very small number of positive samples. For example, SVM achieved perfect classification, but its confidence was based on just four positive samples, risking overfitting. After GAN augmentation, greater data variability was introduced, enabling the SVM to achieve only three misclassifications among over 100,000 synthetic samples.

KNN particularly benefits from GAN augmentation. Before augmentation, the classifier failed to detect any Lateral Movement instances but correctly identified the majority class. After adding over 100,000 synthetic samples generated by a GAN, the classifier achieved excellent recall and nearly perfect precision. This demonstrates that GAN-based augmentation enhances models such as KNN in their ability to recognize previously undetectable minority-class instances.

Decision Trees also reveal this trend. The model performed well on a very small minority class; however, after using GAN, it demonstrates perfect discrimination with the addition of over 100,000 synthetic Lateral Movement samples. This suggests that Decision

Trees can clearly learn class boundaries while avoiding overfitting when GAN-augmented samples are included.

Although Random Forest already achieves perfect classification with very limited minority data, it benefits greatly from GAN-based augmentation. After incorporating GAN, the classifier correctly distinguished a large, balanced dataset without errors. This demonstrates that Random Forest is well-suited to learning from synthetically balanced data, thereby helping to address real-world class imbalance in cybersecurity datasets.

7.3.5. Resource Development

GAN augmentation proved valuable for Resource Development, where pre-GAN evaluation was infeasible due to the limited number of minority samples, particularly for Logistic Regression. As summarized in Table 19, after augmentation, the classifier correctly identifies nearly all Resource Development and “none” samples, with only three misclassifications in total. This highlights a key contribution of the GAN augmentation framework: before it, evaluation was nonexistent due to severely limited minority-class instances.

SVM also experienced a notable improvement. With only one true positive, the model failed to generalize well. Before using GAN, the classifier lacked sufficient samples to learn effectively. After applying a GAN, the classifier achieved near-perfect performance, demonstrating the value of GAN-generated synthetic data.

KNN highlights the challenge of handling extreme class imbalance. Before GANs, the classifier was ineffective at predicting the minority class because it lacked sufficient neighbors. Post-GAN, with synthetic data generated by the GAN, it nearly achieved near-perfect recall and precision, with only three false negatives among more than 100,000 synthetic Resource Development samples, confirming the usefulness of GANs in addressing locality-sparse issues in KNN.

Decision Trees, which previously misclassified a minority instance, achieved perfect classification after using GAN. After employing GAN, the model accurately classifies all instances without sacrificing performance on the majority class.

Without augmentation, even ensemble models such as Random Forests struggle to generalize from a few minority-class samples. With GAN-based augmentation, the model improves its detection of the minority class while avoiding false positives.

7.3.6. Reconnaissance

Table 20 shows that most classifiers could not be meaningfully evaluated before augmentation. However, after introducing GAN-generated samples, all models achieved near-perfect classification, demonstrating the effectiveness of synthetic data in creating a learnable distribution where none previously existed.

Logistic Regression and SVM couldn’t evaluate pre-GAN, likely due to too few minority-class samples. After augmentation, both achieved perfect classification; all Reconnaissance samples were correctly identified, with no false positives or negatives. This highlights the substantial impact of GAN augmentation in generating learnable data distributions that did not previously exist.

Insufficient data limits the GAN’s performance, preventing the KNN classifier from learning meaningful patterns for the Reconnaissance class. After using a GAN, the model achieves near-perfect results, misclassifying only 2 Reconnaissance instances, and successfully captures the structure of the Reconnaissance class with a balanced training dataset. This demonstrates that GAN augmentation can improve the performance of classifiers such as KNN in detecting rare attack classes.

Before GAN, decision trees successfully predicted Reconnaissance samples despite the very small sample size. However, this could risk overfitting. After GAN augmentation, the

model generalizes well, achieving perfect recall and precision, confirming the effectiveness of GAN-based augmentation.

The Random Forest shows perfect accuracy; however, this performance is based on only two positive samples, providing minimal insight into the model's true ability to generalize for the Reconnaissance class. After applying GAN-based augmentation, the model was tested on a larger, more balanced dataset, yet it still achieved 100% accuracy, demonstrating robustness in learning to recognize the Reconnaissance attack type.

7.3.7. Defense Evasion

Defense Evasion, one of the dataset's most underrepresented classes, renders pre-GAN evaluation infeasible for classifiers such as Logistic Regression and SVM. With only one misclassification, Table 21 shows how GAN-based augmentation affects classifiers such as Logistic Regression, which correctly classify all other samples.

The linear SVM model benefited from synthetic samples generated by a GAN. It accurately identifies nearly every instance of Defense Evasion without affecting its ability to classify the majority class. As with Logistic Regression, this classifier shows no significant pre-GAN performance, likely due to the extreme imbalance or the lack of Defense Evasion samples in the original training data.

Before GAN augmentation, KNN failed to recognize the minority class (Defense Evasion), likely due to its underrepresentation in the training data. After GAN augmentation, the KNN model improves in correctly identifying nearly all Defense Evasion cases with only one misclassification. This highlights the benefit of GAN augmentation for models like KNN in detecting minority classes in cybersecurity tasks with class imbalance.

The Decision Tree model, before augmentation, fails to identify the minority class (Defense Evasion), probably because of severe class imbalance and bias toward the majority class. Following GAN augmentation, the model achieves nearly perfect classification performance on the previously underrepresented class. This also demonstrates how GANs can effectively enhance models such as decision trees that may struggle with extreme class imbalance.

Random Forest exhibited the same pattern; it did not detect the minority class before GAN but showed strong recovery after GAN, with only one mistake. This confirms that ensemble models benefit from GAN augmentation when the original dataset is too sparse to provide practical training examples.

7.3.8. Initial Access

Initial Access is also among the most underrepresented classes in the dataset, preventing meaningful pre-GAN evaluation for most classifiers. As summarized in Table 22, none of the models detected the minority class before augmentation. No pre-GAN confusion matrix was produced for Logistic Regression due to the extreme imbalance in the Initial Access dataset before augmentation. Post-GAN, the classifier correctly identifies nearly all Initial Access samples, with only one misclassification. The model also maintains complete accuracy for the majority class while avoiding false positives.

As with Logistic Regression, SVM indicates that no pre-GAN evaluation for Initial Access was likely due to the severe class imbalance in the training dataset. After applying the GAN, the model shows high accuracy, correctly identifying only one Initial Access sample. The majority class is also correctly classified with zero false positives. Both Logistic Regression and SVM misclassified one Initial Access sample, confirming their consistent performance across multiple linear models after the GAN.

Before GAN, KNN failed to detect Initial Access entirely, likely due to insufficient training data for the minority class. In a single instance, KNN could not establish any

neighborhood boundary. After GAN augmentation, the classifier correctly identified nearly all Initial Access samples. This demonstrates that GAN-generated samples constitute a representative, diverse dataset, thereby enabling KNN to form stable neighborhoods.

The Decision Tree classifier completely failed to identify the single Initial Access sample due to severe class imbalance. After augmentation, the classifier accurately identified all Initial Access samples, misclassifying only 1 of over 100,000. This demonstrates the substantial improvement that GAN-augmentation samples provide to the classifier, narrowing the detection gap between the two classes.

Random Forest initially failed to identify the minority class (Initial Access) before using GAN. Due to a single example and strong class imbalance, the classifier defaulted to labeling everything as the majority class. After GAN-based augmentation, the classifier correctly identified nearly all minority class instances while still accurately classifying the majority class.

7.3.9. Persistence

Persistence represented another minority class with very few samples, which impeded meaningful pre-GAN evaluation for most classifiers. As shown in Table 23, no classifier reliably detected the minority class before augmentation; Logistic Regression and SVM yielded no evaluable results, and all other models failed to identify any positive instances.

Logistic Regression and SVM, which could not be meaningfully evaluated before GAN, both achieved near-perfect classification after augmentation, misclassifying only one minority instance while perfectly classifying the majority class.

KNN failed before GAN because it lacked sufficient neighbors and relied solely on the majority class. After applying GAN augmentation, the model correctly identified nearly all Persistence samples, demonstrating that synthetic samples helped it establish effective neighborhood boundaries.

Decision Trees and Random Forest showed similar patterns, both failing to identify Persistence pre-GAN. However, after augmentation, they detected nearly all instances, with just one misclassification. These results confirm that GAN-based augmentation consistently improves a classifier's ability to recognize rare attack classes.

The complete set of confusion matrices from Tables 15–23 shows a significant improvement in classification performance after applying GAN-based augmentation across all attack categories. Before augmentation, classifiers had difficulty identifying minority classes, often completely misclassifying them as the majority class (none). In several cases, models failed to recognize or produce evaluation results for a very small number of minority class samples.

After augmentation, the classifiers showed significant improvements in correctly identifying minority classes while maintaining high accuracy on the majority class. The augmented data enabled the models to learn more effective decision boundaries, reducing false negatives and improving overall detection performance.

Therefore, the confusion matrices demonstrate the effectiveness of the GAN-based augmentation method in improving models' ability to distinguish minority classes from the majority class. Consequently, classification performance improved significantly, with a substantial reduction in bias toward the majority class.

For completeness, the detailed differences in pre- and post-GAN confusion matrices (ΔTP , ΔFN , ΔTN , ΔFP) for each classifier and attack type are presented in Appendix A Tables A1–A9. These tables provide precise numerical evidence of the improvements, particularly in reducing false positives and false negatives for underrepresented classes.

7.4. t-SNE Visualizations

To evaluate the distribution and quality of synthetic samples, t-SNE dimensionality reduction was applied to both real minority-class samples and GAN-generated samples. The t-SNE visualizations are provided for qualitative insight into the alignment between real and synthetic samples. They are not used for interpretation or as a quantitative measure of class separability.

In all visualizations, blue points denote GAN-generated synthetic samples, and red points denote real minority samples. This is shown in Figure 7, where the blue dots indicate synthetic samples generated by the GAN for Credential Access. In contrast, the red dots represent real minority-class samples from the original dataset.

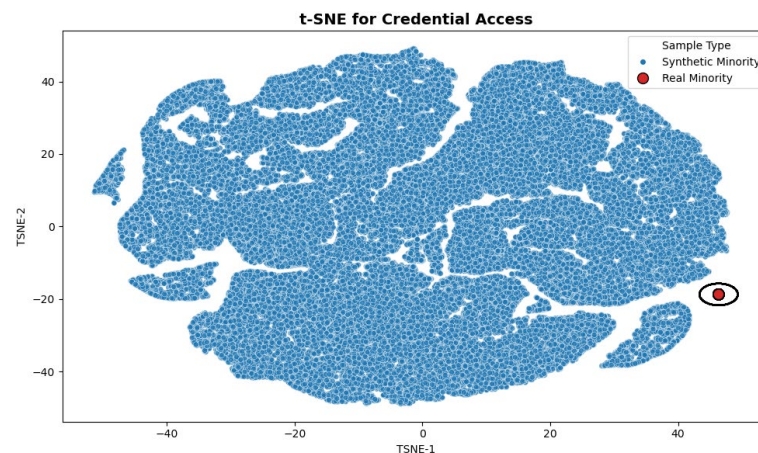


Figure 7. t-SNE dimensionality reduction for Credential Access post-GAN augmentation. The circled points highlight the real minority-class samples.

The graph shows very few red dots, indicating an extremely low number of original sample counts for the Credential Access class, underscoring the class imbalance. The dense blue dots represent many synthetic samples generated via GAN to address this imbalance. The synthetic points form a broad, coherent region around the few real samples, which is consistent with the GAN learning a meaningful approximation of the minority-class structure rather than simply memorizing individual examples.

The t-SNE plot in Figure 8 shows the distribution of real and synthetic minority samples for the Privilege Escalation class after GAN augmentation. Real minority samples (red dots) are few and scattered, consistent with the original data imbalance. The synthetic minority samples (blue dots) are densely packed and form smooth clusters, indicating that the GAN-based augmentation method has successfully generated many samples. Although the real and synthetic samples occupy nearby regions that do not overlap perfectly, many synthetic points lie close to the real samples, suggesting that the GAN has captured key aspects of the Privilege Escalation feature space despite the limited number of real observations.

The t-SNE plot in Figure 9 shows the distribution of real and synthetic Exfiltration samples after GAN-based data augmentation. The blue dots representing the synthetic minority samples form well-defined clusters, indicating that the GAN generated a large, diverse sample set. Only a few genuine Exfiltration samples are visible, underscoring the class imbalance before augmentation. The synthetic samples surround the real samples and appear in similar regions, suggesting that the GAN has effectively learned the overall latent distribution of the minority class. The visual similarity between real and synthetic samples supports the conclusion that the GAN-generated data approximate the structure of the real minority class, as reflected in the improved results shown in the confusion matrices in Tables 10–17.

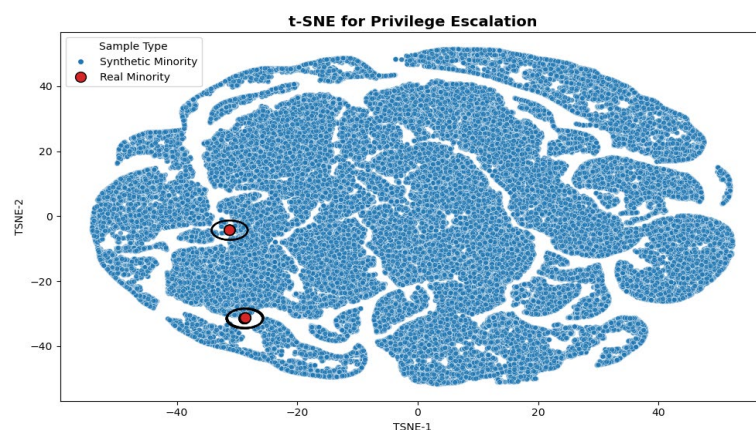


Figure 8. t-SNE dimensionality reduction for Privilege Escalation post-GAN augmentation. The circled points highlight the real minority-class samples.

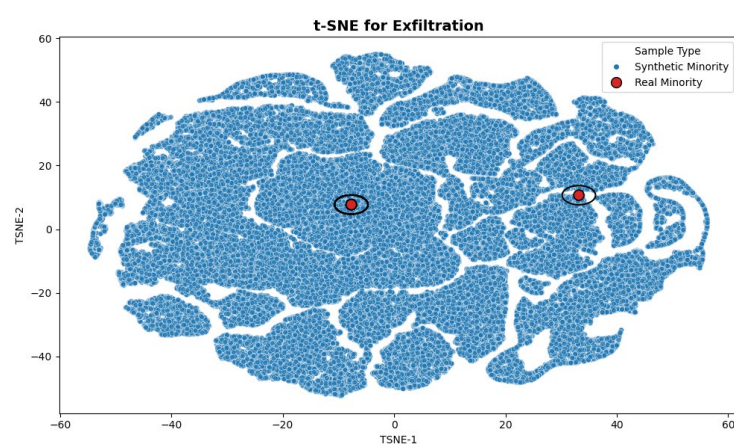


Figure 9. t-SNE dimensionality reduction for Exfiltration post-GAN augmentation. The circled points highlight the real minority-class samples.

The t-SNE plot in Figure 10 shows the Lateral Movement class after GAN-based augmentation with real and synthetic minority-class samples. The synthetic data points (blue) dominate the visualization and are densely and smoothly spread out, indicating that the GAN effectively learned a diverse representation of the minority class. This implies high-quality synthetic data generation, which helps reduce overfitting and enhances classifier performance, as demonstrated in the confusion matrices in Tables 15–23. A small number of real Lateral Movement samples (red) are visible, highlighting the class imbalance.

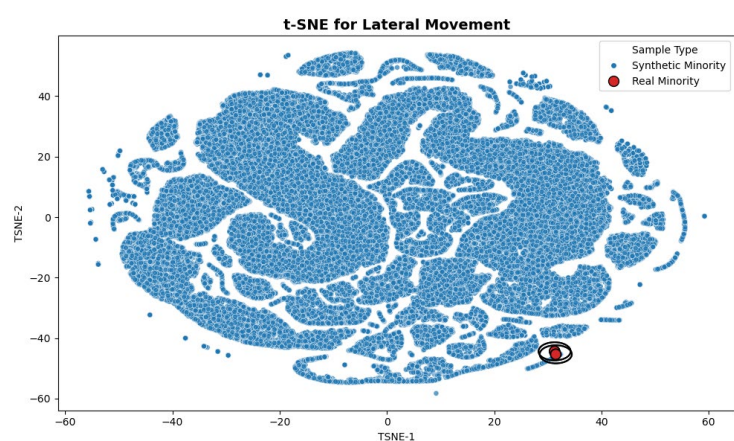


Figure 10. t-SNE dimensionality reduction for Lateral Movement post-GAN augmentation. The circled points highlight the real minority-class samples.

Figure 11 shows the t-SNE visualization of Resource Development after GAN-based augmentation of the minority class. A dense and well-spread distribution of synthetic samples dominates the plot. Their broad coverage indicates that the GAN generated a diverse and representative set of synthetic data for this class. The real samples are limited and mixed with synthetic samples, indicating substantial overlap between the real and generated data.

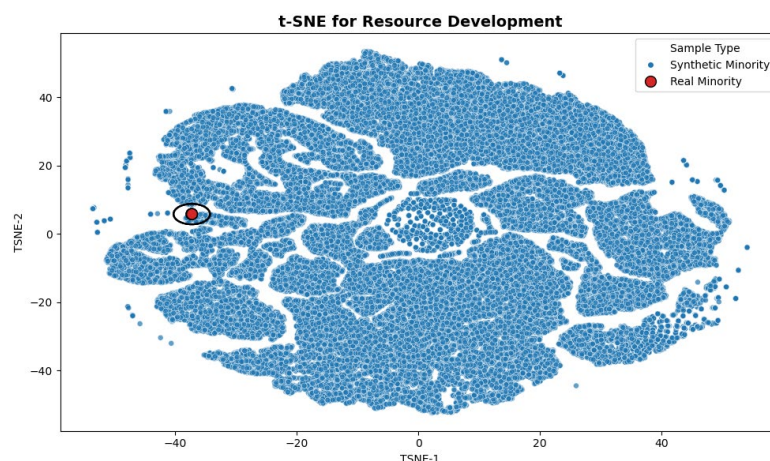


Figure 11. t-SNE dimensionality reduction for Resource Development post-GAN augmentation. The circled points highlight the real minority-class samples.

Figure 12 shows the t-SNE plot for the Reconnaissance class after GAN-based data augmentation. The synthetic data forms several distinct subclusters, suggesting that the GAN has captured complex structure within the Reconnaissance feature space, with the few real samples embedded within these regions.

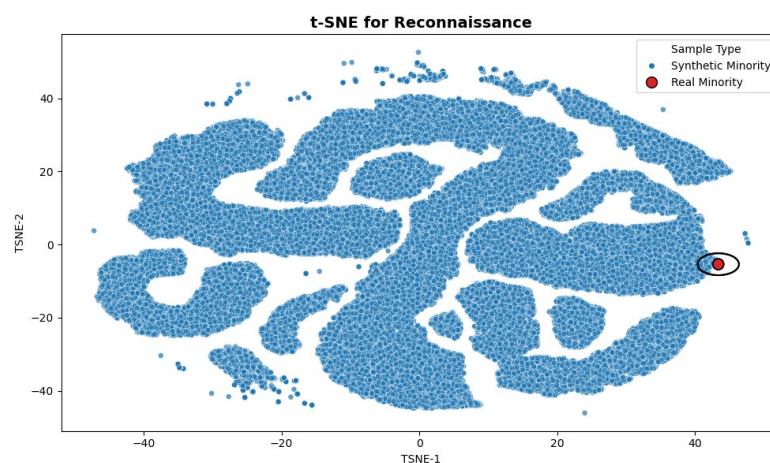


Figure 12. t-SNE dimensionality reduction for Reconnaissance post-GAN augmentation. The circled points highlight the real minority-class samples.

Figure 13 shows the t-SNE visualization for the Defense Evasion class after GAN-based data augmentation. The synthetic samples form a large, tight cluster with a smooth density spread across the t-SNE space. This pattern is consistent with stable data generation without apparent collapse into a single mode while exhibiting diversity in the generated samples.

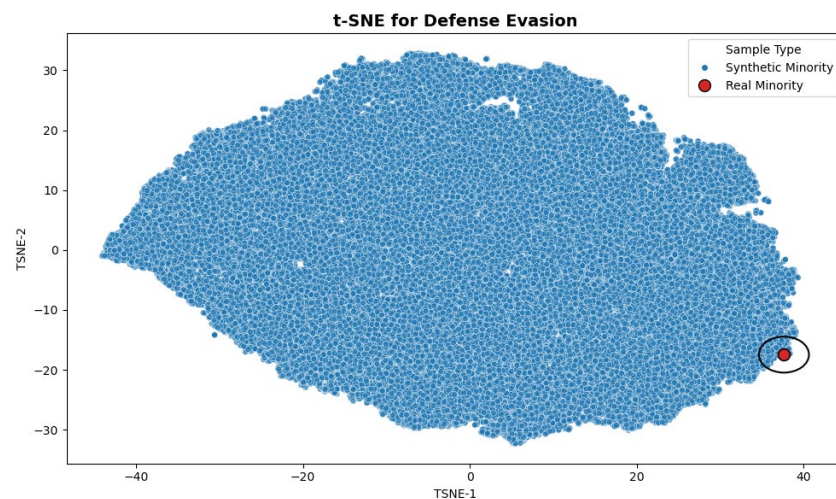


Figure 13. t-SNE dimensionality reduction for Defense Evasion post-GAN augmentation. The circled points highlight the real minority-class samples.

Figure 14 displays the t-SNE plot for Initial Access after GAN augmentation, showing the distribution of both synthetic and real minority samples. The artificial data points form a dense, compact cluster with extensive coverage, indicating that the generator can produce diverse yet consistent samples. Only one real minority point is visible, reflecting the limited number of genuine samples available in this class.

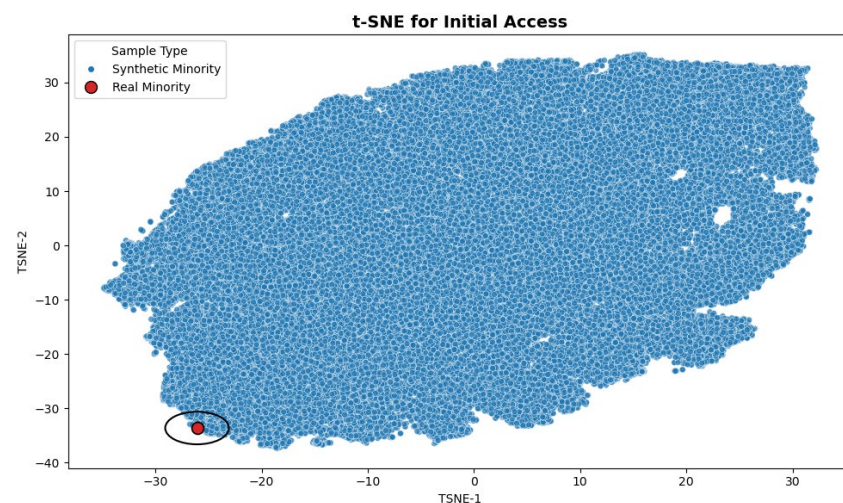


Figure 14. t-SNE dimensionality reduction for Initial Access post-GAN augmentation. The circled points highlight the real minority-class samples.

Figure 15 displays the t-SNE plot for Persistence after GAN-based augmentation, comparing synthetic and real minority samples. The artificial data points form a dense, well-structured cluster that spans the local region of the feature space associated with Persistence, indicating that the generator can produce both diverse and coherent samples. In contrast, only one real Persistence point is visible, emphasizing the extreme imbalance of this class in the original dataset. The proximity of the real point to the synthetic cluster suggests that the GAN approximated the underlying distribution of the minority class despite limited training data.

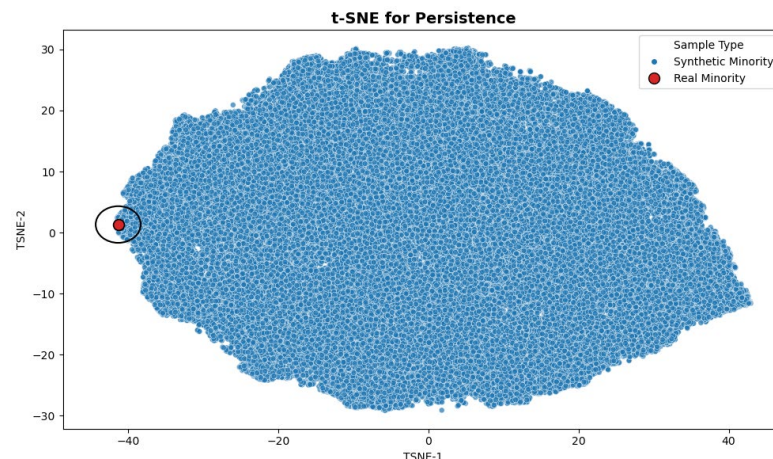


Figure 15. t-SNE dimensionality reduction for Persistence post-GAN augmentation. The circled points highlight the real minority-class samples.

In conclusion, the t-SNE visualizations for the minority classes, shown in Figures 5–13, illustrate the effectiveness of the GAN-based augmentation strategy in generating realistic samples for underrepresented attack classes. Across all figures, the synthetic minority samples (blue) form dense, continuous clusters, consistent with the GAN capturing key aspects of the minority-class distributions. The t-SNE visualizations collectively reinforce the conclusions from the confusion matrices: GAN-based augmentation balances class distribution and maintains feature-space structure, enhancing classifier generalization and detection accuracy across minority cyber threat classes.

7.5. GAN Training Behavior

The plots in Figure 16 show the training dynamics of the GAN model for the Credential Access class, including generator/discriminator loss behavior and discriminator output trajectories.

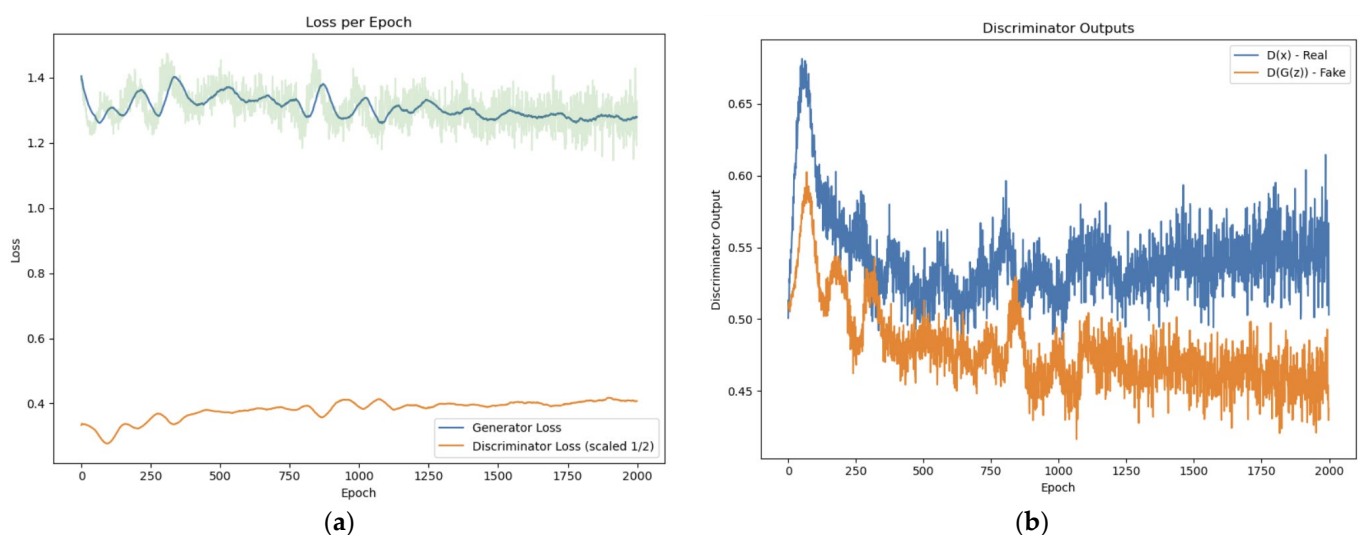


Figure 16. GAN training: (a) GAN training loss for class Credential Access; the green shaded region indicates epoch-wise loss variability (smoothed trend shown in solid line); (b) Discriminator outputs for real samples $D(x)$ and synthetic samples $D(G(z))$ during GAN training.

Figure 16a, Loss per Epoch, shows the Generator loss fluctuates and gradually settles near 1.25–1.30, while the Discriminator loss stabilizes around 0.35–0.42. These trajectories indicate that neither network dominates the other during training and that both networks

continue to update throughout the training. These losses do not exhibit divergence or collapse to zero, consistent with stable adversarial learning.

Figure 16b shows the discriminator output during training. The discriminator outputs for real samples $D(x)$ remain between approximately 0.5 and 0.57, while outputs for synthetic samples $D(G(z))$ remain between 0.43 and 0.48. These values remain separated, without converging to 0 or 1, consistent with adversarial training, in which the generator steadily improves. At the same time, the discriminator maintains a moderate ability to distinguish real and synthetic examples.

Overall, the generator and discriminator maintain distinct loss and output patterns throughout training, and neither network overwhelms the other.

7.6. Training Time Results

Training times were recorded on Google Colab Pro+ using the computational configurations described in Section 6.1.1. All models were trained using GAP acceleration. The results show substantial variation in runtime across GAN variants. The GAN required 44 min.

7.7. Summary

Overall, the experimental results demonstrate that class-specific GAN augmentation substantially improves minority-class detection across multiple MITRE ATT&CK tactics and classifiers. The most consistent gains are observed in recall and F1-score, indicating a reduction in false negatives for rare attack behaviors. These improvements are achieved without degrading the performance of the majority class, suggesting that the synthetic samples enhance class separability rather than introducing noise. While performance varies across classifiers and attack types, the results collectively support the effectiveness of GAN-based augmentation for addressing extreme class imbalance in intrusion detection datasets.

8. Discussions

The results presented in Section 7 indicate that Generative Adversarial Networks (GANs) provide a transformative solution for addressing severe class imbalance in cybersecurity datasets, including UWF-ZeekData22 [16,17]. This section discusses the broad implications of these findings, summarizes insights across classifiers and attack types, and demonstrates that class-specific GAN augmentation enhances minority-class detection in highly imbalanced intrusion-detection scenarios.

8.1. Addressing Extreme Imbalance in Cybersecurity

A significant challenge in IDS is the inherently imbalanced distribution of real-world network traffic. In UWF-ZeekData22, several attack types have fewer than ten samples, making it impossible for traditional classifiers to learn reliable decision boundaries. Before augmentation, Logistic Regression and SVM consistently failed to detect minority attacks, especially Lateral Movement, Resource Development, Defense Evasion, Initial Access, and Persistence. At the same time, KNN misclassified many minority instances due to the sparsity of local neighborhoods. Decision Trees and Random Forests often performed well, even with very small sample sizes [53].

GAN-based augmentation and the ratio-driven approach used in this study fundamentally alter this scenario by generating tens of thousands of synthetic minority samples per minority class. This transforms an otherwise intractable classification task into one in which all models can effectively evaluate minority detection. The results confirm that GANs can reconstruct minority manifolds even with extremely limited real data, thereby enabling the practical deployment of IDS models in environments with severe class imbalance.

The effectiveness of the proposed framework is attributable to its class-specific design. By training independent GANs exclusively on minority-class samples for each MITRE ATT&CK tactic, the generators learn focused, attack-specific feature distributions without interference from benign traffic. This isolation is critical under extreme sparsity, where exposure to majority-class patterns could dominate the learning process. The consistent gains observed across diverse classifiers suggest that the generated samples meaningfully enrich the minority-class feature space rather than merely replicating existing instances.

8.2. Comparison to Traditional Oversampling

Traditional oversampling methods, such as SMOTE, rely on local neighborhood interpolation to generate synthetic samples [55]. Because they depend solely on minority-class nearest neighbors, they often fail to capture the dataset's global structure, limiting the variability and diversity of the generated samples. Moreover, SMOTE selects k-nearest neighbors based solely on Euclidean distance, without accounting for decision boundaries or potential overlap with the majority class. This can result in synthetic samples being placed near ambiguous or noisy regions of the feature space. In contrast, GAN-based methods learn the underlying data distribution and therefore produce higher-quality, more diverse synthetic instances that better represent the true minority-class manifold.

8.3. Effectiveness of GAN-Based Augmentation

Across all evaluated classifiers, precision, recall, and F1-score consistently improved after GAN-based augmentation. The most notable improvements are seen in the confusion matrix analysis, where previously misclassified minority classes under the baseline models show a significant increase in true positives. Before augmentation, the classifiers showed a strong bias toward the majority “none” class, achieving high accuracy on benign traffic but performing poorly on rare attack types, likely due to extreme class imbalance.

After augmentation, however, the confusion matrix showed a significant improvement in minority-class detection. As summarized in Tables 15–23, classifier performance neared complete separation, with very few false negatives and only a small number of false positives. This indicates that GAN-generated synthetic samples introduce valuable within-class variability, helping classifiers generalize rather than memorize.

These findings are supported by the detailed difference tables in Appendix A Tables A1–A9, which outline the specific changes in TP, FN, TN, and FP across all tactics. These results verify that GAN augmentation significantly improves minority-class detection while maintaining the majority-class performance. The most notable improvements are observed among the most underrepresented classes, particularly Initial Access, Persistence, and Defense Evasion.

8.4. Visualization Insights

The visual analyses presented in this study provide strong evidence that GAN-generated minority samples exhibit realistic structure, meaningful diversity, and close alignment with the true minority distributions. However, these visualizations are intended to provide qualitative insight and do not constitute quantitative evidence of separability.

The t-SNE projection of the augmented minority classes in that study, as shown in Figures 7–15, indicates that the synthetic samples (blue) generated by the GAN populate the minority-class distribution despite the extremely low number of real samples (red). This suggests that the generator did not simply memorize the original data points but instead captured the underlying structure of the minority-class distribution.

Moreover, the training of the Vanilla GAN shown in Figure 16a,b shows that the Generator and the Discriminator engage in a balanced adversarial game, with the Generator loss stabilizing around ~1.25 and experiencing mild oscillations. Meanwhile, the Discriminator

loss stays moderate and does not drop to zero. Both networks evolve together, and their loss patterns show that neither dominates the other, thereby preventing issues such as model collapse, gradient vanishing, or uninformative discriminator feedback. The separation between $D(x)$ (real samples) and $D(G(z))$ (fake samples) indicates that the Generator has learned to produce realistic samples, thereby enhancing the classifiers' performance after augmentation.

8.5. Implications for Intrusion Detection Systems (IDS)

The ability to enhance detection of rare ATT&CK tactics has significant implications for operational IDS environments, including improved detection of high-impact threats in which many rare categories align with high-risk adversary objectives (e.g., Credential Access, Exfiltration, Persistence). Improving recall for these behaviors can directly decrease undetected intrusions. Other implications concern the practical application and scalability of the augmentation pipeline, which uses a class-specific augmentation framework integrated into existing IDS workflows by extracting minority-attack samples, training a small GAN model, generating synthetic samples, and retraining or fine-tuning the classifiers.

8.6. Principle-Level Validity of GAN-Generated Network Traffic

Unlike image data, network traffic exhibits strong logical coupling and physical constraints among features, arising from protocol semantics, timing behavior, and flow-level dependencies. Prior studies have emphasized that network traffic features are not independent but are governed by protocol behavior and operational constraints, distinguishing cybersecurity data from image and signal domains [1,52]. A valid concern when applying generative models to cybersecurity data is whether synthetic samples comply with protocol logic or represent physically plausible network behavior.

In this study, GAN-based augmentation is performed at the flow-feature level rather than at the packet or protocol-message level. The UWF-ZeekData22 dataset comprises structured, aggregated features extracted from Zeek logs, such as byte counts, durations, connection states, and categorical protocol attributes [16,17]. Consequently, the GAN learns the joint statistical distribution of valid flow-level representations rather than attempting to generate raw packets or protocol sequences, which require explicit protocol modeling and state-machine enforcement [8].

Protocol consistency is implicitly enforced through several mechanisms. First, all training data originate from real network traffic and therefore represent valid protocol executions observed in operational environments [1]. As a result, the generator is constrained to learn feature combinations that already satisfy protocol logic within the empirical distribution of observed flows. Second, feature preprocessing removes identifiers and non-informative metadata while preserving semantically meaningful attributes, a practice commonly adopted in flow-based intrusion detection research to maintain local coherence while reducing noise [52]. Third, normalization and bounded activation functions restrict generated values to feasible numeric ranges observed during training, which has been shown to improve stability and realism in generative modeling of structured tabular data [9,10,37].

Rather than guaranteeing formal protocol correctness, which would require packet-level synthesis, explicit protocol grammar constraints, and simulation-based validation, this augmentation framework aims to generate statistically realistic feature vectors that lie on the same manifold as real minority-class flows. This level of fidelity is sufficient and appropriate for training downstream intrusion-detection classifiers, which operate exclusively in feature space and do not explicitly reason about protocol semantics [1,8,52].

The effectiveness of this approach is empirically supported by consistent improvements in recall and F1-score across multiple classifiers, as well as by distributional alignment observed in t-SNE visualizations and confusion matrix analysis [51]. Nevertheless, ensuring strict protocol compliance in generative network traffic remains an open research challenge and a direction for future work, particularly for packet-level generation, simulation-based IDS evaluation, and protocol-aware generative modeling [8,11].

8.7. Interpretation of Perfect Post-Augmentation Performance

The occurrence of perfect F1-scores in some post-augmentation should be interpreted in the context of extreme class sparsity and binary classification. For several attack tactics, the pre-GAN setting contained very few minority samples; after GAN-based augmentation, minority-class samples became more compact and separable in feature space. In such cases, classifiers—particularly tree-based models—may achieve perfect discrimination on held-out validation folds consisting only of real samples.

8.8. Limitations

Despite these promising results, several limitations should be noted. First, GAN training under extreme data sparsity remains challenging and may be sensitive to architectural choices and training hyperparameters. Second, this study focuses on a single real-world dataset; although UWF-ZeekData22 is representative of modern IDS challenges, the results may not generalize uniformly to other network environments. Third, the evaluation emphasizes classical machine learning classifiers; the interaction between GAN-based augmentation and deep learning-based IDS models remains an open area of future investigation. In addition, a systematic experimental comparison between GAN-based augmentation and classical oversampling methods, such as SMOTE, is an important direction for future work.

While stratified cross-validation and strict fold-level data separation mitigate data leakage, perfect classification performance may in part reflect overfitting to highly structured synthetic distributions. Evaluating generalization across independent datasets and conducting formal statistical significance testing across cross-validation folds represent important directions for future work. Although this study focuses on UWF-ZeekData22 due to its realism and extreme class imbalance, extending the evaluation to additional benchmark datasets such as CIC-IDS2017 and NSL-KDD will be explored in future work to further assess generalizability.

This study employs Vanilla GAN architecture to isolate the impact of class-specific adversarial augmentation under extreme class sparsity and to establish a clear baseline under extreme class imbalance. While the current work focuses on this baseline setting, subsequent work is underway that extends the proposed framework to evaluate more advanced GAN variants, including Conditional GANs (cGAN), Wasserstein GANs (WGAN), and WGAN-GP, on the same UWF-ZeekData22 dataset. These extensions aim to systematically assess the impact of alternative adversarial objectives and conditioning mechanisms on minority-class data augmentation.

8.9. Summary of Key Insights

GANs improve minority-class detection across all classifiers and tactics. False negatives are significantly reduced, sometimes eliminated. Precision is not degraded. Improvements generalize across classifier types, and class-specific augmentation is a practical technique for real-world IDS pipelines. These findings confirm that GAN-based augmentation is a viable and effective strategy for enhancing cyberattack detection in highly imbalanced datasets.

These findings motivate further exploration of class-specific generative augmentation strategies, including their extension to additional datasets, alternative generative models, and real-time intrusion detection scenarios.

9. Conclusions and Future Work

This study addressed the challenge of extreme class imbalance in cybersecurity intrusion detection by proposing a class-specific GAN-based data augmentation framework. Using the UWF-ZeekData22 dataset, each MITRE ATT&CK tactic was formulated as an independent binary classification task, and separate GANs were trained exclusively on minority-class samples to generate realistic synthetic attack data. This design enables focused modeling of rare attack behaviors that are difficult for traditional classifiers to learn under severe data sparsity.

Experimental results demonstrate that GAN-augmented training consistently improves minority-class detection across multiple classifiers, with particularly notable gains in recall and F1-score. The observed reduction in false-negative rates highlights the practical relevance of the proposed approach for intrusion detection scenarios, where missed attacks can have serious consequences. Importantly, these improvements were achieved without degrading the performance of the majority class, indicating that the samples generated enhance class separability rather than introducing noise or distributional drift.

While the proposed framework shows strong potential, several directions for future research remain. First, a deeper investigation into the fidelity and stability of GAN-generated samples, using multidimensional drift and related distributional metrics, will be conducted and form the basis of ongoing work. Second, alternative GAN architectures, including conditional GANs (cGANs), Wasserstein GANs (WGANs), and WGAN-GP, will be systematically compared to identify the most effective generative models for different attack types. Third, extending the framework toward real-time or continual augmentation pipelines may support adaptive learning in operational environments where attack distributions evolve over time. Finally, applying the proposed methodology to additional cybersecurity datasets would further validate its generalizability. Incorporating explainable AI (XAI) techniques to analyze how classifiers respond to synthetic versus real samples also represents an important avenue for future work.

Overall, this study demonstrates that class-specific generative augmentation is a practical and flexible strategy for mitigating severe class imbalance in intrusion detection systems, proving a foundation for improving the detection of rare but high-impact cyberattacks.

Author Contributions: Conceptualization, A.D., S.S.B., D.M. and S.C.B.; methodology, A.D. and S.S.B.; software, A.D.; validation, S.S.B., D.M. and S.C.B.; formal analysis, A.D. and S.S.B.; investigation, A.D.; resources, S.S.B., D.M. and S.C.B.; data curation, A.D. and D.M.; writing—original draft preparation, A.D.; writing—review and editing, S.S.B., D.M. and S.C.B.; visualization, A.D.; supervision, S.S.B., D.M. and S.C.B.; project administration, S.S.B., D.M. and S.C.B.; funding acquisition, S.S.B., D.M. and S.C.B. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by 2021 NCAE-C-002: Cyber Research Innovation Grant Program, Grant Number: H98230-21-1-0170. This research was also partially supported by the Askew Institute at the University of West Florida.

Data Availability Statement: The datasets are available at datasets.uwf.edu (accessed on 8 August 2025).

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

IDS	Intrusion Detection System
ATT&CK	Adversarial Tactics, Techniques, and Common Knowledge
GAN	Generative Adversarial Network
SMOTE	Synthetic Minority Oversampling Technique
D	Discriminator
G	Generator
ReLU	Rectified Linear Unit
SVM	Support Vector Machine
KNN	k-Nearest Neighbor
DT	Decision Tree
RF	Random Forest
AUC-ROC	Area Under the Receiver Operating Characteristic Curve
t-SNE	t-Distributed Stochastic Neighbor Embedding
TN	True Negative
FN	False Negative
FP	False Positive
TP	True Positive
cGAN	Conditional GAN
WGAN	Wasserstein GAN
WGAN-GP	Wasserstein GAN with Gradient Penalty
XAI	Explainable AI

Appendix A

The appendix offers a detailed comparison of classifier performance before and after GAN-based augmentation across all attack types. Specifically, Tables A1–A9 show the differences in confusion matrix results (ΔTP , ΔFN , ΔTN , ΔFP) for each classifier. These tables supplement the main findings by clarifying the specific improvements from augmentation, such as increases in true positives and decreases in false negatives.

Table A1. Pre-GAN vs. Post-GAN Confusion Matrices for Credential Access vs. None.

Classifier	TP (Pre)	TP (Post)	ΔTP	FN (Pre)	FN (Post)	ΔFN	TN (Pre)	TN (Post)	ΔTN	FP (Pre)	FP (Post)	ΔFP
Logistic Regression	30	107,212	+107,182	1	0	−1	428,082	428,724	+642	642	0	−642
SVM	28	107,195	+107,167	3	17	+14	428,724	428,724	0	0	0	0
KNN	31	107,204	+107,173	0	8	+8	428,722	428,724	+2	2	0	−2
Decision Tree	30	107,212	+107,182	1	0	−1	428,724	428,724	0	0	0	0
Random Forest	31	107,212	+107,181	0	0	0	428,724	428,724	0	0	0	0

Table A2. Pre-GAN vs. Post-GAN Confusion Matrices for Privilege Escalation vs. None.

Classifier	TP (Pre)	TP (Post)	ΔTP	FN (Pre)	FN (Post)	ΔFN	TN (Pre)	TN (Post)	ΔTN	FP (Pre)	FP (Post)	ΔFP
Logistic Regression	13	107,194	+107,181	0	0	0	428,720	428,724	+4	4	0	−4
SVM	12	107,190	+107,178	1	4	+3	428,724	428,724	0	0	0	0
KNN	11	107,189	+107,178	2	5	+3	428,722	428,722	0	2	2	0
Decision Tree	11	107,193	+107,182	2	1	−1	428,724	428,724	0	0	0	0
Random Forest	11	107,194	+107,183	2	0	−2	428,724	428,724	0	0	0	0

Table A3. Pre-GAN vs. Post-GAN Confusion Matrices for Exfiltration vs. None.

Classifier	TP (Pre)	TP (Post)	ΔTP	FN (Pre)	FN (Post)	ΔFN	TN (Pre)	TN (Post)	ΔTN	FP (Pre)	FP (Post)	ΔFP
Logistic Regression	6	107,185	+107,179	1	3	+2	428,719	428,724	+5	5	0	−5
SVM	5	107,184	+107,179	2	4	+2	428,724	428,724	0	0	0	0
KNN	5	107,185	+107,180	2	3	+1	428,724	428,720	−4	0	4	+4
Decision Tree	7	107,187	+107,180	0	1	+1	428,724	428,724	0	0	0	0
Random Forest	6	107,187	+107,180	1	1	0	428,724	428,724	0	0	0	0

Table A4. Pre-GAN vs. Post-GAN Confusion Matrices for Lateral Movement vs. None.

Classifier	TP (Pre)	TP (Post)	ΔTP	FN (Pre)	FN (Post)	ΔFN	TN (Pre)	TN (Post)	ΔTN	FP (Pre)	FP (Post)	ΔFP
Logistic Regression	-	107,182	+107,182	-	3	+3	-	428,724	+428,724	-	0	0
SVM	4	107,182	+107,178	0	3	+3	428,724	428,724	0	0	0	0
KNN	0	107,181	+107,181	4	4	0	428,724	428,724	0	0	0	0
Decision Tree	4	107,185	+107,181	0	0	0	428,724	428,724	0	0	0	0
Random Forest	4	107,185	+107,181	0	0	0	428,724	428,724	0	0	0	0

Table A5. Pre-GAN vs. Post-GAN Confusion Matrices for Resource Development vs. None.

Classifier	TP (Pre)	TP (Post)	ΔTP	FN (Pre)	FN (Post)	ΔFN	TN (Pre)	TN (Post)	ΔTN	FP (Pre)	FP (Post)	ΔFP
Logistic Regression	-	107,183	+107,183	-	1	+1	-	428,722	+428,722	-	2	0
SVM	1	107,182	+107,182	2	2	0	428,724	428,724	0	0	0	0
KNN	0	107,181	+107,181	3	3	0	428,724	428,724	0	0	0	0
Decision Tree	2	107,184	+107,184	1	0	−1	428,724	428,724	0	0	0	0
Random Forest	1	107,183	+107,183	2	1	−1	428,724	428,724	0	0	0	0

Table A6. Pre-GAN vs. Post-GAN Confusion Matrices for Reconnaissance vs. None.

Classifier	TP (Pre)	TP (Post)	ΔTP	FN (Pre)	FN (Post)	ΔFN	TN (Pre)	TN (Post)	ΔTN	FP (Pre)	FP (Post)	ΔFP
Logistic Regression	-	107,183	+107,183	-	0	0	-	428,724	+428,724	-	0	0
SVM	-	107,183	+107,183	-	0	0	-	428,724	+428,724	-	0	0
KNN	0	107,181	+107,181	2	2	+2	428,724	428,724	0	0	0	0
Decision Tree	2	107,183	+107,181	0	0	0	428,724	428,724	0	0	0	0
Random Forest	2	107,183	+107,181	0	0	0	428,724	428,724	0	0	0	0

Table A7. Pre-GAN vs. Post-GAN Confusion Matrices for Defense Evasion vs. None.

Classifier	TP (Pre)	TP (Post)	ΔTP	FN (Pre)	FN (Post)	ΔFN	TN (Pre)	TN (Post)	ΔTN	FP (Pre)	FP (Post)	ΔFP
Logistic Regression	-	107,181	+107,181	-	1	+1	-	428,724	+428,724	-	0	0
SVM	-	107,181	+107,181	-	1	+1	-	428,724	+428,724	-	0	0
KNN	0	107,181	+107,181	1	1	0	428,724	428,724	0	0	0	0
Decision Tree	0	107,181	+107,181	1	1	0	428,724	428,724	0	0	0	0
Random Forest	0	107,181	+107,181	1	1	0	428,724	428,724	0	0	0	0

Table A8. Pre-GAN vs. Post-GAN Confusion Matrices for Initial Access vs. None.

Classifier	TP (Pre)	TP (Post)	Δ TP	FN (Pre)	FN (Post)	Δ FN	TN (Pre)	TN (Post)	Δ TN	FP (Pre)	FP (Post)	Δ FP
Logistic Regression	-	107,181	+107,181	-	1	+1	-	428,724	+428,724	-	0	0
SVM	-	107,181	+107,181	-	1	+1	-	428,724	+428,724	-	0	0
KNN	0	107,181	+107,181	1	1	0	428,724	428,724	0	0	0	0
Decision Tree	0	107,181	+107,181	1	1	0	428,724	428,724	0	0	0	0
Random Forest	0	107,181	+107,181	1	1	0	428,724	428,724	0	0	0	0

Table A9. Pre-GAN vs. Post-GAN Confusion Matrices for Persistence vs. None.

Classifier	TP (Pre)	TP (Post)	Δ TP	FN (Pre)	FN (Post)	Δ FN	TN (Pre)	TN (Post)	Δ TN	FP (Pre)	FP (Post)	Δ FP
Logistic Regression	-	107,181	+107,181	-	1	+1	-	428,724	+428,724	-	0	0
SVM	-	107,181	+107,181	-	1	+1	-	428,724	+428,724	-	0	0
KNN	0	107,181	+107,181	1	1	0	428,724	428,724	0	0	0	0
Decision Tree	0	107,181	+107,181	1	1	0	428,724	428,724	0	0	0	0
Random Forest	0	107,181	+107,181	1	1	0	428,724	428,724	0	0	0	0

References

1. Sommer, R.; Paxson, V. Outside the Closed World: On Using Machine Learning for Network Intrusion Detection. In Proceedings of the 2010 IEEE Symposium on Security and Privacy, Oakland, CA, USA, 16–19 May 2010; pp. 305–316. [\[CrossRef\]](#)
2. Jabeen, U.; Singh, K.; Vats, S. Credit Card Fraud Detection Scheme Using Machine Learning and Synthetic Minority Oversampling Technique (SMOTE). In Proceedings of the 2023 5th International Conference on Inventive Research in Computing Applications (ICIRCA), Coimbatore, India, 3–5 August 2023; pp. 122–127. [\[CrossRef\]](#)
3. Ahsan, M.; Gomes, R.; Denton, A. SMOTE Implementation on Phishing Data to Enhance Cybersecurity. In Proceedings of the 2018 IEEE International Conference on Electro/Information Technology (EIT), Rochester, MI, USA, 3–5 May 2018; pp. 0531–0536. [\[CrossRef\]](#)
4. Massaoudi, M.; Refaat, S.S.; Abu-Rub, H. Intrusion Detection Method Based on SMOTE Transformation for Smart Grid Cybersecurity. In Proceedings of the 2022 3rd International Conference on Smart Grid and Renewable Energy (SGRE), Doha, Qatar, 20–22 March 2022; pp. 1–6. [\[CrossRef\]](#)
5. Haruna, U.S.; Mahmoud, A.A.; Danlami, M.; Sharifai, A.G. Enhancing Network Intrusion Detection for Big Datasets with Hybrid Deep Learning: A SWIN Transformer and VGG19. In Proceedings of the 2024 1st International Conference on Cyber Security and Computing (CyberComp), Melaka, Malaysia, 6–7 November 2024; pp. 13–18. [\[CrossRef\]](#)
6. Acharya, T.; Annamalai, A.; Chouikha, M.F. Addressing the Class Imbalance Problem in Network-Based Anomaly Detection. In Proceedings of the 2024 IEEE 14th Symposium on Computer Applications & Industrial Electronics (ISCAIE), Penang, Malaysia, 24–25 May 2024; pp. 1–6. [\[CrossRef\]](#)
7. Rahma, F.; Rajasa, M.C.; Rachmadi, R.F.; Pratomo, B.A.; Purnomo, M.H. Resampling Effects on Imbalanced Data in Network Intrusion Classification. In Proceedings of the 2024 International Electronics Symposium (IES), Denpasar, Indonesia, 6–8 August 2024; pp. 534–540. [\[CrossRef\]](#)
8. Dunmore, A.; Jang-Jaccard, J.; Sabrina, F.; Kwak, J. A comprehensive survey of generative adversarial networks (GANs) in cybersecurity intrusion detection. *IEEE Access* **2023**, *11*, 123456–123470. [\[CrossRef\]](#)
9. Mohammed, M.; Ammar, M. A comprehensive review of generative adversarial networks: Fundamentals, applications, and challenges. *WIREs Comput. Stat.* **2024**, *16*, e1629.
10. Gui, J.; Sun, Z.; Wen, Y.; Tao, D.; Ye, J. A Review on Generative Adversarial Networks: Algorithms, Theory, and Applications. *IEEE Trans. Knowl. Data Eng.* **2023**, *35*, 3313–3332. [\[CrossRef\]](#)
11. Ring, R.; Wunderlich, S.; Grube, T. Flow-based network traffic generation using generative adversarial networks. In Proceedings of the IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), Paris, France, 29 April–2 May 2019; pp. 140–145.
12. Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; et al. Scikit-learn: Machine learning in Python. *J. Mach. Learn. Res.* **2011**, *12*, 2825–2830.
13. Agrawal, G.; Kaur, A.; Myneni, S. A Review of Generative Models in Generating Synthetic Attack Data for Cybersecurity. *Electronics* **2024**, *13*, 322. [\[CrossRef\]](#)

14. Zhao, X.; Fok, K.W.; Thing, V.L.L. Enhancing Network Intrusion Detection Performance using Generative adversarial networks. *Comput. Secur.* **2024**, *145*, 104005. [CrossRef]
15. Hojjatinia, S.; Monshizadeh, M.; Khatri, V.; Mähönen, P.; Yan, Z. Improving IoT Intrusion Detection Using GAN-Based Synthetic Traffic Augmentation. In Proceedings of the 2025 IEEE 11th International Conference on Network Softwarization (NetSoft), Budapest, Hungary, 23–27 June 2025; pp. 164–168. [CrossRef]
16. Bagui, S.S.; Mink, D.; Bagui, S.C.; Ghosh, T.; Plenkers, R.; McElroy, T. Introducing UWF-ZeekData22: A Comprehensive Network Traffic Dataset Based on the MITRE ATT&CK Framework. *Data* **2023**, *8*, 18. [CrossRef]
17. Available online: <https://datasets.uwf.edu> (accessed on 8 August 2025).
18. Goodfellow, I.J.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; Bengio, Y. Generative adversarial nets. In Proceedings of the 28th International Conference on Neural Information Processing Systems, Cambridge, MA, USA, 8–13 December 2014.
19. Wenzel, M.T. Generative Adversarial Networks and Other Generative Models. In *Machine Learning for Brain Disorders*; MEVIS: Bremen, Germany, 2019.
20. Jetchev, N.; Bergmann, U.; Vollgraf, R. Texture synthesis with spatial generative adversarial networks. *arXiv* **2016**, arXiv:1611.08207.
21. Wang, T.-C.; Liu, M.-Y.; Zhu, J.-Y.; Tao, A.; Kautz, J.; Catanzaro, B. High-resolution image synthesis and semantic manipulation with conditional GANs. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 8798–8807.
22. Wang, Z.; She, Q.; Ward, T.E. Generative Adversarial Networks in Computer. *ACM Comput. Surv.* **2022**, *52*, 38.
23. Saxena, D.; Cao, J. Generative Adversarial Networks (GANs). *ACM Comput. Surv.* **2022**, *54*, 42. [CrossRef]
24. He, H.; Garcia, E.A. Learning from Imbalanced Data. *IEEE Trans. Knowl. Data Eng.* **2009**, *21*, 1263–1284. [CrossRef]
25. Bagui, S.S.; Mink, D.; Bagui, S.C.; Subramaniam, S. Determining Resampling Ratios Using BSMOTE and SVM-SMOTE for Identifying Rare Attacks in Imbalanced Cybersecurity Data. *Computers* **2023**, *12*, 204. [CrossRef]
26. Bagui, S.S.; Mink, D.; Bagui, S.C.; Subramaniam, S. Resampling to Classify Rare Attack Tactics in UWF-ZeekData22. *Knowledge* **2024**, *4*, 96–119. [CrossRef]
27. Bagui, S.S.; Mink; Bagui, S.C.; Subramaniam; Wallace, D. Resampling Imbalanced Network Intrusion Datasets To Identify Rare Attacks. *Future Internet* **2023**, *15*, 130. [CrossRef]
28. Chawla, V.; Bowyer, K.W.; Hall, L.O.; Kegelmeyer, W.P. SMOTE: Synthetic minority over-sampling technique. *J. Artif. Intell. Res.* **2002**, *16*, 321–357. [CrossRef]
29. Chawla, N.V. Data mining for imbalanced datasets: An overview. In *Data Mining and Knowledge Discovery Handbook*; Springer: Berlin/Heidelberg, Germany, 2006; pp. 853–867. [CrossRef]
30. Han, H.; Wang, W.Y.; Mao, B.H. Borderline-SMOTE: A new over-sampling method in imbalanced data sets learning. In Proceedings of the International Conference on Intelligent Computing (ICIC), Hefei, China, 23–26 August 2005; pp. 878–887.
31. He, H.; Bai, Y.; Garcia, E.A.; Li, S. ADASYN: Adaptive synthetic sampling approach for imbalanced learning. In Proceedings of the IEEE International Joint Conference on Neural Networks (IJCNN), Hong Kong, China, 1–6 June 2008; pp. 1322–1328.
32. Ullah, I.; Mahmoud, Q.H. A Framework for Anomaly Detection in IoT Networks Using Conditional Generative Adversarial Networks. *IEEE Access* **2021**, *9*, 165907–165931. [CrossRef]
33. Dutta, I.K.; Ghosh, B.; Carlson, A.; Totaro, M.; Bayoumi, M. Generative Adversarial Networks in Security: A Survey. In Proceedings of the 2020 11th IEEE Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON), New York, NY, USA, 28–31 October 2020; pp. 0399–0405. [CrossRef]
34. Lin, Z.; Shi, Y.; Xue, Z. IDSGAN: Generative adversarial networks for attack generation against intrusion detection. In Proceedings of the 26th Pacific-Asia Conference on Knowledge Discovery and Data Mining, Chengdu, China, 16–19 May 2022; pp. 1–7.
35. Kumar, K.; Mandoria, H.L.; Singh, R.; Dwivedi, S.P. Paras Malware Detection and Classification using Generative Adversarial Network. *Int. J. Comput. Sci. Inf. Technol.* **2024**, *16*, 93–110.
36. Xia, Q.; Dong, S.; Peng, T. An Abnormal Traffic Detection Method for IoT Devices Based on Federated Learning and Depthwise Separable Convolutional Neural Networks. In Proceedings of the 2022 IEEE International Performance, Computing, and Communications Conference (IPCCC), Austin, TX, USA, 11–13 November 2022; pp. 352–359. [CrossRef]
37. Xu, L.; Skoularidou, M.; Cuesta-Infante, A.; Veeramachaneni, K. Modeling Tabular Data Using Conditional GAN. *arXiv* **2019**, arXiv:1907.00503. [CrossRef]
38. Zhang, W.J.; Lin, Y. On the principle of design of resilient systems-application to enterprise information systems. *Enterp. Inf. Syst.* **2010**, *4*, 99–110. [CrossRef]
39. Woods, D.D. Four concepts for resilience and the implications for the future of resilience engineering. *Reliab. Eng. Syst. Saf.* **2015**, *141*, 5–9. [CrossRef]
40. MITRE ATT&CK. Reconnaissance, Tactic TA0043—Enterprise. 2 October 2020. Available online: <https://attack.mitre.org/tactics/TA0043/> (accessed on 8 August 2025).

41. MITRE ATT&CK. Discovery, Tactic TA0007—Enterprise. 17 October 2018. Available online: <https://attack.mitre.org/tactics/TA0007/> (accessed on 8 August 2025).
42. MITRE ATT&CK. Credential Access, Tactic TA0006—Enterprise. 17 October 2018. Available online: <https://attack.mitre.org/tactics/TA0006/> (accessed on 8 August 2025).
43. MITRE ATT&CK. Privilege Escalation, Tactic TA0004—Enterprise. 17 October 2018. Available online: <https://attack.mitre.org/tactics/TA0004/> (accessed on 8 August 2025).
44. MITRE ATT&CK. Exfiltration, Tactic TA0010—Enterprise. 17 October 2018. Available online: <https://attack.mitre.org/tactics/TA0010/> (accessed on 8 August 2025).
45. MITRE ATT&CK. Lateral Movement, Tactic TA0008—Enterprise. 17 October 2018. Available online: <https://attack.mitre.org/tactics/TA0008/> (accessed on 8 August 2025).
46. MITRE ATT&CK. Resource Development, Tactic TA0042—Enterprise. 30 September 2020. Available online: <https://attack.mitre.org/tactics/TA0042/> (accessed on 8 August 2025).
47. MITRE ATT&CK. Initial Access, Tactic TA0001—Enterprise. 17 October 2018. Available online: <https://attack.mitre.org/tactics/TA0001/> (accessed on 8 August 2025).
48. MITRE ATT&CK. Persistence, Tactic TA0003—Enterprise. 17 October 2018. Available online: <https://attack.mitre.org/tactics/TA0003/> (accessed on 8 August 2025).
49. MITRE ATT&CK. Defense Evasion, Tactic TA0005—Enterprise. 17 October 2018. Available online: <https://attack.mitre.org/tactics/TA0005/> (accessed on 8 August 2025).
50. Prabhu, H.; Valadi, J.; Arjunan, P. Generative Adversarial Network with Soft-Dynamic Time Warping and Parallel Reconstruction for Energy Time Series Anomaly Detection. *arXiv* **2024**, arXiv:2402.14384. [[CrossRef](#)]
51. Guller, M. *Big Data Analytics with Spark*; Apress: New York, NY, USA, 2015; pp. 160–165.
52. Han, J.; Kamber, M.; Pei, J. *Data Mining Concepts and Techniques*, 3rd ed.; Elsevier Inc.: Waltham, MA, USA, 2012.
53. Quinlan, J.R. Induction of decision trees. *Mach. Learn.* **1986**, *1*, 81–106. [[CrossRef](#)]
54. van der Maaten, L.; Hinton, G. Visualizing data using t-SNE. *J. Mach. Learn. Res.* **2008**, *9*, 2579–2605.
55. Sharma, A.; Singh, P.K.; Chandra, R. SMOTified-GAN for class imbalanced pattern classification problems. *IEEE Access* **2022**, *10*, 30655–30665. [[CrossRef](#)]

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.