Journal of Theoretical and Applied Information Technology
31st October 2026. Vol.104. No.20
© Little Lion Scientific
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

CTWGAN-GP: SEQUENTIAL INTEGRATION OF CTGAN AND WCGAN-GP FOR PRACTICAL MACHINE LEARNING UTILITY OF SYNTHETIC TABULAR DATA

RAWAN F. ALSHEHRI[1*] , SABAH M. ALZAHRANI[2]
Department of Computer Science, College of Computers and Information Technology, Taif University,
P.O. Box 11099,Taif 21944, Saudi Arabia
E-mail: [1*] rwanf2022@gmail.com, [2]sa.sabah@tu.edu.sa

ABSTRACT

Generating high-quality synthetic tabular data is essential for machine learning applications in data-scarce and privacy-sensitive domains. This study presents CTWGAN-GP, a sequential generative framework that integrates Conditional Tabular GAN (CTGAN) and Wasserstein Conditional GAN with Gradient Penalty (WCGAN-GP) to examine whether multi-stage model integration can improve the quality and utility of synthetic tabular data. Existing approaches typically rely on a single GAN model, overlooking the potential benefits of sequential integration. Experiments were conducted using two benchmark datasets: the Adult dataset, which includes mixed data types, and the Credit Card Fraud dataset, which contains only numerical features. Two sequential configurations were evaluated using machine learning utility metrics, including F1-score and ROC AUC. Results show that CTWGAN-GP improves performance on the Adult dataset, particularly when CTGAN was applied after WCGAN-GP, demonstrating better downstream classification outcomes. Specifically, CTWGAN-GP achieves an F1-score of 0.6454, outperforming CTGAN (0.6067) by approximately 6.4% and WCGAN-GP (0.4195) by approximately 54%. Notably, the strongest sequential configuration achieved its best performance without requiring preprocessing or postprocessing, suggesting that its effectiveness is attributable to the model design rather than external corrections. In contrast, for the numerical-only Credit dataset, WCGAN-GP alone achieves the highest performance, while incorporating CTGAN—whether before or after WCGAN-GP—reduces downstream classification performance. These findings highlight that the effectiveness of sequential GAN frameworks depends strongly on dataset characteristics and that no single architecture consistently generalizes across datasets.

Keywords: *Synthetic Tabular Data; CTGAN; WCGAN-GP; Sequential GAN; Machine Learning Applications.*

1. INTRODUCTION

Synthetic data generation involves modeling and simulating the statistical properties of real-world data to create new datasets that closely resemble the originals [1]. Synthetic data generation techniques are increasingly important in fields such as artificial intelligence (AI) and data analysis, particularly when access to large volumes of high-quality real data is limited due to privacy concerns or scarcity [2]. They aim to replicate meaningful patterns observed in real data across various formats, including images, text, and tabular data [3].

Synthetic datasets can be generated using statistical methods, which rely on mathematical models and probabilistic distributions, or machine learning-based approaches, especially deep neural networks (DNNs), which capture complex real-world distributions [4]–[6]. Machine learning approaches leverage AI and ML algorithms to learn data patterns and produce high-fidelity synthetic samples, helping overcome challenges such as limited data availability and privacy restrictions [7]–[11].

These methods have become essential in research and practical applications, providing high-quality datasets where real data is insufficient or sensitive [6]. Among the most influential techniques for synthetic data generation are Generative Adversarial Networks (GANs), which consist of a generator that creates synthetic data and a discriminator that evaluates its realism [12]. Originally developed for images, GANs have been extended to other domains such as text and tabular data [13], [14]. Notable models for tabular data include Conditional Tabular GAN (CTGAN) [15],

which addresses the unique characteristics and challenges of structured datasets.

Tabular data is organized in rows and columns, where each row represents a single instance and each column represents a feature or attribute describing that instance [16]. Tabular data is widely used in domains such as finance, healthcare, marketing, and retail [17], [18], and it plays a central role in analytics and informed decision-making [19]. Its structured format facilitates understanding relationships between features and outputs, making it suitable for predictive modeling.

However, tabular data presents several challenges, including heterogeneous features, class imbalance, missing values, small sample sizes, and privacy-sensitive content, all of which can reduce the reliability and accuracy of machine learning models [15], [20]–[22]. Generating high-quality synthetic tabular data is therefore crucial for overcoming these barriers and supporting robust real-world ML applications [23]. Standard GAN architectures often struggle with these challenges [24], [25], motivating the development of specialized models such as CTGAN [15] and other tabular-data-specific approaches [26][27], which are better suited to capture complex distributions and feature dependencies in tabular datasets.

Research Problem. Although specialized GAN-based models have been developed to address the challenges of tabular data, research in this area remains ongoing, with many opportunities for further improvement. Most existing approaches rely on a single generative architecture, while multi-model approaches commonly integrate different components within a unified architecture rather than through a sequential training process. To the best of our knowledge, no prior study has systematically investigated the sequential integration of two GAN architectures within a structured multi-stage pipeline for synthetic tabular data generation. Consequently, it remains unclear whether sequentially combining complementary GAN models can improve the quality and utility of synthetic tabular data compared with standalone models, and whether the effectiveness of such a sequential framework depends on the ordering and training design choices of its constituent models. This study addresses this gap by investigating a two-stage framework that sequentially integrates CTGAN and WCGAN-GP and evaluating its utility under different experimental settings.

Research Questions. Based on the identified research problem and gap, this study addresses the following research questions:

- RQ1: How does sequential integration of CTGAN and WCGAN-GP affect the quality and utility of synthetic tabular data compared with the standalone models, as measured by downstream machine learning performance?
- RQ2: How do model ordering and sequential training design choices affect the downstream utility of the generated synthetic data?
- RQ3: What practical utility does the proposed sequential framework provide under different synthetic data generation settings, including synthetic sample size and stage-two training data composition?

To address these research questions, the study focuses on three overarching research objectives. First, it proposes and implements a sequential GAN framework—CTWGAN-GP—that integrates CTGAN and WCGAN-GP in two alternative configurations. Second, it evaluates the effects of model ordering and training design choices on the utility of the generated data. Third, it investigates the practical utility of the generated data under different generation settings, including variations in synthetic sample size and stage-two training data composition.

Scientific Contributions. The main scientific contributions of this study are as follows:

- Sequential GAN integration: A two-stage sequential generative framework, CTWGAN-GP, is proposed by integrating CTGAN and WCGAN-GP, with the generated data from the first stage combined with real data to train the second-stage generator. This provides a structured alternative to standalone GAN generation and to hybrid approaches that combine components within a single architecture.
- Systematic evaluation of model ordering: The study systematically investigates two alternative sequential configurations, WCGAN-GP → CTGAN and CTGAN → WCGAN-GP, to examine whether the ordering of complementary generative models affects the quality and utility of the resulting synthetic tabular data, as reflected

in downstream machine learning performance.

- Dataset-dependent effectiveness: The study demonstrates that the effectiveness of sequential GAN integration depends on dataset characteristics, highlighting the importance of considering data properties when selecting and ordering generative models.
- Analysis of sequential training configurations: Additional controlled experiments examine the effects of preprocessing and postprocessing, synthetic sample size, class-level conditioning, extended second-stage training, and the composition of second-stage training data. These experiments provide further evidence that the utility of sequential GAN pipelines depends on the interaction between model architecture and training configuration.

The remainder of this paper is organized as follows: Section 2 reviews relevant literature, highlighting the research gap and the motivation for this study. Section 3 details the research methodology, including dataset description and preprocessing, model design, implementation, and evaluation procedures. Section 4 reports the experimental results, while Section 5 provides a detailed discussion of the findings. Section 6 presents additional experiments conducted. Section 7 concludes the study, and Section 8 outlines potential directions for future work.

2. RELATED WORKS

Synthetic tabular data generation using Generative Adversarial Networks (GANs) has gained substantial attention in recent years, driven by the growing demand for privacy-preserving data alternatives and the need to augment small or imbalanced datasets. Various GAN-based models have been proposed to address challenges inherent in tabular data, such as mixed feature types, class imbalance, and structural dependencies [28].

The remainder of this section is organized into subsections: Subsection 2.1 provides an overview of GANs and their fundamental principles, while Subsection 2.2 discusses how GAN architectures have been adapted for tabular data, addressing key challenges such as class imbalance, mixed-type features, missing data, and privacy concerns. Section 2.3 concludes by identifying the research gap and the motivation for the proposed sequential GAN framework.

2.1 Generative Adversarial Networks (GANs)

Generative Adversarial Networks (GANs) are a type of deep learning model introduced by Goodfellow et al. in 2014 [12], which represented a significant breakthrough in generative modeling. GANs provide a framework for learning data distributions and generating synthetic samples.

A GAN is composed of two competing neural networks: the generator and the discriminator. The generator creates synthetic data starting from random noise, attempting to imitate real data samples. Meanwhile, the discriminator tries to distinguish between real data (from the original dataset) and fake data (produced by the generator). Both networks are trained simultaneously in a feedback loop: the generator improves its outputs to fool the discriminator, and the discriminator becomes better at identifying fake samples. This adversarial training continues as the two networks iteratively improve their respective objectives, with the generator aiming to produce increasingly realistic samples and the discriminator aiming to distinguish real from generated samples.

This process is illustrated in Figure 1. GANs were initially applied in image generation tasks—such as creating realistic human faces or scenes—and have achieved impressive results in producing high-quality synthetic images [29].

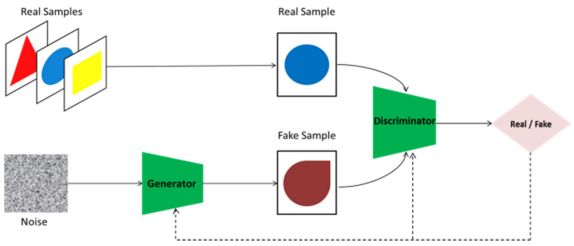


Figure 1 : Illustration of The GAN Training Process.

2.2 Adapting GANs for Tabular Data

GANs have undergone significant adaptations to address the unique challenges involved in generating synthetic tabular data [28]. Unlike image or text data, tabular datasets typically consist of both numerical features (e.g., age, income) and categorical features (e.g., gender, occupation). This mixed structure presents unique modeling challenges. Additionally, tabular data often suffers from class imbalance and may contain missing values. Effective synthetic data generation must also take privacy preservation into account [20], [23].

As a result, several recent innovations have focused on improving the ability of GANs to generate realistic and high-quality tabular data [2].

2.2.1 GANs for Imbalanced Data

Class imbalance is a critical challenge in machine learning, particularly in applications such as fraud detection, medical diagnosis, and risk assessment [30]. It occurs when certain classes are significantly underrepresented compared to others. Training machine learning models on such data often leads to biased learning—where the model becomes overly focused on the majority class and fails to correctly classify minority instances. This imbalance reduces generalization performance, especially when applied to real-world scenarios [21]. A range of GAN-based methods has been proposed to mitigate this issue, each offering distinct mechanisms to improve minority class representation.

Conditional Wasserstein GAN (CWGAN) [30] employs an auxiliary classifier loss, enabling the generator to conditionally produce more samples for minority classes. While effective at balancing the dataset, CWGAN may still struggle with extreme imbalance due to its single-generator architecture. Duo-GAN [31] addresses this limitation by using two generators—one for majority and one for minority classes—thus explicitly separating the learning of class distributions. This approach demonstrates improved fidelity for minority samples but introduces additional model complexity and training overhead.

Hybrid architectures, such as CWGAN-GP-PacGAN [32], combine multiple techniques, including conditional generation, auxiliary classification, and PacGAN-style discrimination. These integrated designs show superior performance in both class balance and downstream predictive metrics, particularly in complex datasets like recommendation systems. Similarly, WCGAN-GP [27] enhances training stability and diversity through gradient penalty, addressing common GAN issues such as mode collapse, which indirectly supports better minority representation.

More recent strategies, exemplified by CTAB-GAN+ [33], focus on adaptive training rather than architectural changes. By dynamically reweighting the loss function based on class distribution priors, CTAB-GAN+ effectively prioritizes underrepresented classes, improving robustness in highly imbalanced datasets.

Collectively, these models illustrate a trade-off between architectural complexity, training stability, and class-balancing effectiveness. Single-generator methods such as CWGAN are simpler but may be less robust under extreme imbalance, whereas dual-generator and hybrid models can provide improved minority representation at the cost of increased computational demand. Adaptive approaches such as CTAB-GAN+ highlight the potential of training strategies to complement architectural solutions. This diversity of strategies suggests that different GAN architectures may provide complementary capabilities for handling imbalanced tabular data, which is relevant to the sequential design investigated in this study.

2.2.2 GANs for Imputation and Missing Data

Missing data is a common challenge in real-world tabular datasets [34], and inadequate handling can bias downstream analyses or obscure meaningful patterns. GANs offer a promising solution by learning the underlying patterns in the available data and generating realistic estimates to fill in missing values.

One of the early major contributions in this area was the Generative Adversarial Imputation Net (GAIN) [35]. GAIN introduced a hint mechanism, which gives the discriminator partial information about which entries in the dataset are missing. This helps the generator produce more accurate and realistic imputations by focusing on the observed features.

Building on this, Multiple Imputation via GAN (MI-GAN) [34] extended the idea by modeling missing data under the Missing at Random (MAR) assumption—where the likelihood of a value being missing depends only on other observed values. MI-GAN includes two versions: MI-GAN1 and MI-GAN2. MI-GAN1 focuses on imputing missing values in cases with mostly observed features, which provides stronger statistical guarantees. MI-GAN2 uses an iterative approach to train on all available data, allowing it to generalize across the dataset.

Unlike traditional methods that produce a single estimate, MI-GAN generates multiple imputations for each missing entry. This improves the reliability of the analysis and better accounts for the uncertainty inherent in incomplete data.

Collectively, these methods demonstrate how GAN-based architectures can enhance imputation quality by explicitly modeling feature dependencies and producing multiple plausible estimates. While single-network models like GAIN provide a strong baseline, approaches such as MI-GAN illustrate the benefits of extending architecture and training strategies to better handle complex datasets, laying

the groundwork for more advanced generative imputation models.

2.2.3 GANs for Numerical and Categorical Data

Tabular datasets include both numerical values (e.g., income, age) and categorical values (e.g., gender, occupation), which often interact in complex ways. These mixed data types pose a challenge for generative models, prompting the development of specialized GAN architectures to handle them more effectively [20].

One of the earliest models in this direction was Tabular GAN (TGAN) [36]. It uses a Long Short-Term Memory (LSTM) network—a type of recurrent neural network—to sequentially model tabular data and preserve feature dependencies. This approach improves representation of multimodal numerical distributions but may struggle with highly imbalanced categorical variables. CTGAN [15] addresses this limitation by conditioning on discrete column distributions during sampling, which allows better modeling of categorical dependencies and rare categories. This conditional strategy enhances data realism for mixed-type datasets and has become a benchmark in tabular synthesis evaluation.

Directed Acyclic Tabular GAN (DATGAN) [37] extends this approach by incorporating expert knowledge via a Directed Acyclic Graph (DAG), explicitly encoding relationships between variables to improve structural consistency. CTAB-GAN [13] further unifies numerical and categorical features in a single conditional vector, combining multiple loss functions—including classification and information losses—to address sparsity, imbalance, and training stability.

More recent hybrid approaches further exploit multi-stage or hierarchical designs. T-VAE-GAN [38] combines a Variational Autoencoder (VAE) and GAN hierarchically to improve modeling of complex tabular distributions while enhancing computational efficiency, particularly for mixed numerical and categorical features. Similarly, TabTransGAN [39] leverages transformer-based architectures to capture long-range dependencies across features, improving synthesis fidelity for large and high-dimensional tabular datasets with mixed types.

Some models target specific data types. GANBLR [40] focuses on categorical variables and leverages Bayesian principles to improve interpretability, while medGAN [41] is tailored for multi-label medical records with entirely categorical features.

Overall, TGAN excels at modeling multimodal continuous distributions, and Table/CTAB-GAN approaches maintain correlations and structural fidelity. More recent models, such as T-VAE-GAN and TabTransGAN, employ hierarchical or hybrid architectures to better handle complex mixed-type tabular data. This suggests that integrating multiple stages or combining different types of GANs may further enhance the quality and utility of synthetic tabular datasets. These differences indicate that tabular GANs can exhibit complementary strengths in modeling different aspects of heterogeneous data, providing a rationale for investigating whether such complementary capabilities can be exploited through sequential model integration.

2.2.4 GANs for Privacy and Data Anonymization

Data scarcity in many domains is often linked to privacy concerns, especially when dealing with sensitive personal information [2], [20]. This growing need for privacy has led to the development of privacy-preserving GAN models that generate synthetic data while maintaining the statistical properties of the original dataset. These models aim to ensure data usability without compromising sensitive attributes [23].

Early work such as PNR-GAN [42], based on the Cramér GAN framework [43], demonstrated how GANs can anonymize structured records by retaining behavioral patterns while masking personal identifiers. Subsequent models expanded this idea through architectural or objective-function innovations. Table-GAN [44] employs a Convolutional Neural Network (CNN)-based discriminator and an auxiliary classifier to better preserve semantic relationships and mitigate the risk of attribute inference, while ADS-GAN [45] introduces an identifiability metric that explicitly quantifies reidentification risk in synthetic health records.

More recent approaches focus on enhancing robustness against privacy attacks. CasTGAN [23] adopts a cascaded generator design to reinforce feature dependencies and reduce inference vulnerabilities, whereas PATE-GAN [46] incorporates differential privacy by aggregating outputs from multiple teacher models trained on disjoint data subsets. Multi-objective extensions such as SMOE-CTGAN [47] further balance the trade-off between privacy preservation and statistical utility across diverse tasks.

Taken together, these models address privacy from different perspectives, including anonymization, structural preservation, formal privacy guarantees, and privacy–utility optimization. The diversity of these objectives further illustrates the potential value of combining complementary generative capabilities when designing synthetic data pipelines for practical applications.

Table 1 summarizes the key features, application domains, and generation strategies of the reviewed GAN models. This comparative table reinforces the observation that most models are designed to address isolated challenges, which underscores the need to evaluate a sequential framework that combines multiple strengths.

2.3 Research Gap and Motivation

This section has reviewed a wide range of GAN-based models that address key challenges in synthetic tabular data generation, including class imbalance, missing values, mixed feature types, and data privacy. While these models demonstrate effectiveness within their respective design scopes, most rely on single-architecture solutions. For example, CTGAN [15] is designed to model categorical distributions and conditional relationships in tabular data, whereas WCGAN-GP [27] employs the Wasserstein objective and gradient penalty to improve training stability and distributional learning. Recent hybrid approaches, such as CWGAN-GP-PacGAN [32], demonstrate that combining multiple generative components can provide complementary capabilities. However, these approaches generally integrate their components within a unified architecture rather than applying distinct generative models sequentially through a structured multi-stage training process.

Based on the reviewed literature, a research gap remains in the systematic investigation of sequential GAN integration for synthetic tabular data generation. In particular, limited attention has been given to whether the complementary capabilities of different GAN architectures can be exploited through sequential stages and whether model ordering affects the quality and utility of the resulting synthetic data. This gap motivates the need to empirically evaluate whether sequential integration can provide advantages over standalone models and whether its effectiveness depends on model ordering and training configuration.

This study addresses this gap by investigating CTWGAN-GP, a two-stage sequential framework that integrates CTGAN [15] and WCGAN-GP [27] in two alternative configurations: WCGAN-GP followed by CTGAN, and CTGAN followed by WCGAN-GP. The framework is evaluated against standalone baselines using downstream machine learning performance to assess the quality and utility of the generated synthetic data. The study further examines whether the effectiveness of sequential integration varies according to model ordering, dataset characteristics, and training configuration.

Table 1: Comparison of Tabular Data Generation Models.

Model	Paper's Title	Year	Highlight	Generation Level	Applications	Data Type	Reference
medGAN	Generating Multi-label Discrete Patient Records using GANs	2017	Generates synthetic patient records combining autoencoders with GANs	Record-level	Healthcare	Discrete	[41]
TGAN	Synthesizing Tabular Data using GANs	2018	Generates tables with both continuous and discrete variables	Record-level	Various	Mixed	[36]
Table-GAN	Data Synthesis based on GANs	2018	Privacy-preserving table synthesis using CNN-based discriminator	Record-level	Various	Mixed	[44]
GAIN	Missing Data	2018	Novel method for imputing missing	Cell-level	Various	Mixed	[35]

	Imputation using GANs		values				
PNR-GAN	Airline Passenger Name Record Generation using GANs	2018	Generates privacy-preserving synthetic PNRs	Record-level	Airline Industry	Mixed	[42]
CTGAN	Modeling Tabular Data using Conditional GAN	2019	Models complex tabular data with conditional sampling	Record-level	Various	Mixed	[15]
PATE-GAN	Generating Synthetic Data with Differential Privacy	2019	Combines GANs with PATE for privacy-preserving synthesis	Record-level	Various	Mixed	[46]
ADS-GAN	Anonymization Through Data Synthesis Using GANs	2020	Generates EHR-like data to minimize re-identification risk	Record-level	Healthcare	Mixed	[45]
WCGAN-GP	Synthesising Tabular Datasets Using Wasserstein Conditional GANs	2020	Stabilizes GAN training with gradient penalty	Record-level	Various	Mixed	[27]
CWGAN	Conditional Wasserstein GAN-based Oversampling	2021	Addresses class imbalance with conditional generation	Record-level	Finance	Mixed	[30]
MI-GAN	Multiple Imputation via GAN for Blockwise Missing Values	2021	Handles high-dimensional missing values with two methods	Cell-level	Healthcare	Mixed	[34]
CTAB-GAN	Effective Table Data Synthesizing	2021	Improves mixed-variable encoding with specialized loss functions	Record-level	Various	Mixed	[13]
Duo-GAN	Synthesizing Heavily Unbalanced Data	2021	Two generators for positive and negative records	Record-level	Finance	Mixed	[31]
DATGAN	Integrating Expert Knowledge into GANs	2022	Uses DAGs to preserve variable relationships	Record-level	Various	Mixed	[37]
CWGAN-GP-PacGAN	Hybrid GAN for Imbalanced Data	2022	Combines WCGAN-GP, auxiliary classifier, and PacGAN discriminator	Record-level	Recommendation	Mixed	[32]
GANBLR	Interpretabl	2023	Bayesian	Record-	Various	Categori	[40]

	e Tabular Data Generation		networks improve interpretability	level		cal	
CasTGAN	Cascaded GAN for Realistic Tabular Data	2024	Series of generators for feature-level synthesis	Feature-level	Various	Mixed	[23]
SMOE-CTGAN	Multi-Objective Evolutionary GAN	2024	Balances utility and disclosure risk with multi-objective optimization	Record-level	Census	Mixed	[47]
CTAB-GAN+	Enhancing Tabular Data Synthesis	2024	High-utility, privacy-preserving synthesis with gradient penalty	Record-level	Various	Mixed	[33]
T-VAE-GAN	Synthetic Tabular Data Generation Using a VAE-GAN Architecture	2025	Combines VAE and GAN hierarchically to improve tabular distribution modeling	Hybrid-level	Various	Mixed	[38]
TabTrans GAN	Hybrid GAN with Transformer Architecture for Tabular Data Synthesis	2025	Integrates Transformer and GAN to enhance semantic and distributional data synthesis	Hybrid-level	Various	Mixed	[39]

3. MATERIALS AND METHODS

3.1 Research Design

This research adopts an experimental design to evaluate the effectiveness of the proposed sequential GAN-based framework, Conditional Tabular Wasserstein GAN with Gradient Penalty (CTWGAN-GP), for synthetic tabular data generation. Based on the research gap identified in the literature review, the central objective is to investigate whether sequentially combining two complementary GAN models, Conditional Tabular GAN (CTGAN) [15] and Wasserstein Conditional GAN with Gradient Penalty (WCGAN-GP) [27], can improve the quality and utility of synthetic tabular data, as assessed through downstream machine learning performance.

The experiments are conducted on benchmark tabular datasets exhibiting representative characteristics such as class imbalance and mixed feature types. The models are trained and evaluated under controlled experimental conditions using consistent data-generation procedures and evaluation metrics.

To investigate the effect of model ordering, the proposed sequential framework is implemented in two configurations:

- WCGAN-GP → CTGAN: WCGAN-GP is first trained on the real training data. The resulting synthetic data are then combined with the original real training data and used to train CTGAN, which subsequently generates the final synthetic dataset.
- CTGAN → WCGAN-GP: CTGAN is first trained on the real training data. The resulting synthetic data are then combined with the original real training data and used to train WCGAN-GP, which subsequently generates the final synthetic dataset.

This staged design allows the study to investigate whether the complementary characteristics of the two generative models can be leveraged through sequential processing. WCGAN-GP provides training stability and improves the diversity of numerical features [27], while CTGAN effectively models complex dependencies among categorical features [15]. Accordingly, the sequential configurations provide a controlled basis for

examining whether the order in which these models are applied affects the quality and utility of the resulting synthetic data.

The research is structured into three main phases:

- Sequential Model Training: The first GAN is trained on the real training data. Its generated synthetic samples are then combined with the real training data and used to train the second GAN.
- Synthetic Data Generation: The final synthetic dataset is generated by sampling from the second GAN in the sequence.
- Evaluation: The generated synthetic data are used to train a machine learning classifier, which is subsequently evaluated on real test data. Metrics such as F1-score and ROC AUC are used to assess downstream classification performance and, consequently, the practical utility of the synthetic data.

The proposed sequential configurations are evaluated against standalone CTGAN [15] and WCGAN-GP [27] baselines to determine whether sequential integration provides measurable improvements in synthetic data quality and utility.

In addition to the primary comparison between the sequential configurations and standalone models, several controlled experiments are conducted to further investigate the behavior of the proposed framework. These experiments examine the effects of synthetic data volume, extended training epochs, excluding real data from the second-stage training (ablation), class-level conditioning, and different preprocessing and postprocessing strategies.

3.2 Datasets

This research employs two widely used tabular datasets to evaluate the performance of generative models on real-world classification tasks: the Credit Card Fraud Detection dataset [48] and the Adult Income dataset [49]. These datasets were selected based on their relevance in prior studies and their representation of common challenges in tabular data, such as class imbalance, feature heterogeneity, and structural skewness.

3.2.1 Adult Dataset

The Adult Income dataset, obtained from the UCI Machine Learning Repository [49], is a benchmark dataset commonly used for evaluating machine learning models on structured data. It contains 48,840 records and 14 features, including both numerical (such as age, fnlwgt, and capital-gain) and categorical (such as workclass, education, and occupation) variables. The binary target variable income indicates whether an individual's income exceeds $50K per year.

Statistical Summary:

- The dataset is pre-divided into a training set (34,561 instances) and a test set (14,279 instances).
- It includes 6 numerical features and 8 categorical features.
- Several categorical features contain missing values, including workclass (1,836), occupation (1,843), and native-country (583).
- Numerical attributes such as capital-gain and capitalloss exhibit extreme outliers (e.g., capital-gain max = 99,999), while fnlwgt has a large standard deviation (105,550).
- The dataset is moderately imbalanced: 75.92% of individuals earn <= $50K, and 24.08% earn more.
- Some categorical features are strongly skewed (e.g., workclass: Private = 73.87%), while others include rare classes (e.g., occupation: Armed-Forces = 0.03%).
- As shown in Figure 2, the correlation heatmap indicates generally weak linear correlations among the numerical features.

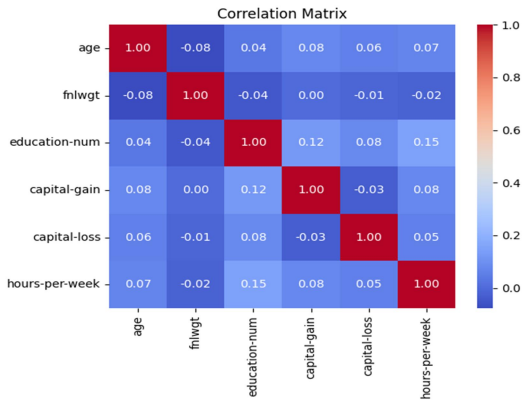


Figure 2: Correlation heatmap of the Adult dataset.

3.2.2 Credit Card Fraud Detection Dataset

The Credit Card Fraud Detection dataset, sourced from Kaggle [48], is widely used for evaluating machine learning models under severe class imbalance. It contains 284,807 transaction records

and 31 features, including V1–V28 (obtained through PCA transformation), Time, and Amount. The binary target variable, Class, indicates whether a transaction is fraudulent (1) or legitimate (0). Only 492 transactions are classified as fraudulent (0.17%), making the dataset highly imbalanced.

Statistical Summary:

- The features V1–V28 are centered around zero due to the PCA transformation.
- The average target class value is 0.0017, confirming the high class imbalance.
- Features such as V1, V2, V3, V4, V7, and V9 show significant mean differences between classes, indicating high discriminative potential.
- Features like V22 and V23 show minimal variation across classes.
- As shown in Figure 3, the PCA-transformed features exhibit generally weak pairwise correlations, reflecting the transformed nature of the feature space.

Table 2 provides a comparative statistical overview of the datasets, while Table 3 summarizes the rationale for their selection.

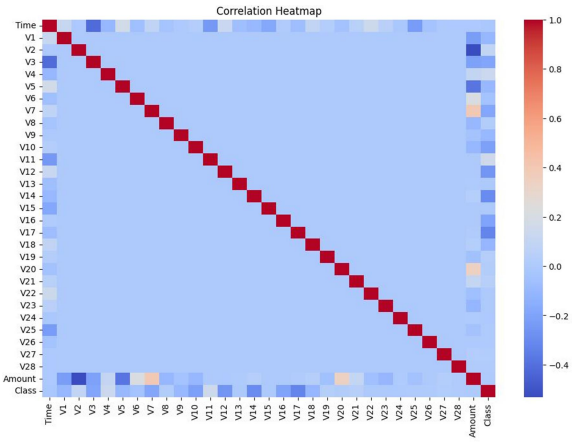


Figure 3 :Correlation heatmap of the Credit Card Fraud Detection dataset.

Table 2: Summary of Statistical Properties for the Credit Card Fraud Detection and Adult Datasets.

Dataset	Credit Card Fraud Detection	Adult (Census Income)
Source	Kaggle [48]	UCI [49]
Domain	Finance	Social Science
Number of Instances	284,807	48,840
Number of Features	31	14
Target Variable	Fraudulent (Class 1) vs. Non-fraudulent (Class 0)	Income (≤50K, >50K)
Class Imbalance	Highly imbalanced (0.17% fraudulent, 99.83% non-fraudulent)	Moderately imbalanced (75.92% ≤50K, 24.08% >50K)
Data Types	Numeric (V1–V28, Time, Amount)	Numeric (6 columns) and Categorical (8 columns)
Missing Values	-	workclass (1,836), occupation (1,843), native-country (583)
Correlation Insights	Weak correlations among features	Weak correlations among features

Table 3: Reasons for selecting datasets

Dataset	Credit Card Fraud Dataset	Adult Dataset
Imbalanced Data	Real-world fraud detection imbalance; useful for testing balancing	Reflects real-world imbalances; useful for testing balancing
Privacy Preservation	Features are anonymized via PCA transformation, ensuring privacy while preserving statistical patterns	-
Diverse Data Types	-	Contains numerical and categorical features; useful for handling mixed data
Missing Values	-	Contains missing values in categorical features; evaluates GANs' ability to impute and generate realistic distributions
Extreme Values	-	Contains extreme outliers (e.g., capital-gain max 99,999); evaluates GAN performance in

		robust data generation
Generalizability Testing	Evaluates GANs' ability to generalize across domains by testing synthetic data in financial prediction tasks	Suitable for social science domain; tests cross-domain generalizability
Established benchmark dataset	Widely used in tabular data modeling	Widely used in tabular data modeling

3.3 Datasets Preprocessing

To ensure fair evaluation and reliable training, each dataset underwent a dedicated preprocessing pipeline tailored to its structure and characteristics. The preprocessing was designed to align with established practices in generative modeling and to maintain consistency with the protocols followed in prior studies, particularly the original WCGAN-GP implementation [27].

3.3.1 Adult Dataset

Prior to model training on the Adult dataset, a preprocessing procedure was applied to handle missing values and prepare the data for the respective generative models. Missing values were imputed using mode substitution, replacing each missing entry with the most frequent value within its corresponding feature. This approach is particularly suitable for categorical variables and helps preserve their observed distributions.

For WCGAN-GP, categorical attributes were label-encoded using LabelEncoder, and all numerical and encoded categorical features were standardized using StandardScaler. This preprocessing follows the procedure adopted in the original WCGAN-GP study [27] and places the input features on a comparable numerical scale.

CTGAN, in contrast, was trained directly on the cleaned dataset without label encoding or feature scaling, using the original categorical representations. This distinction follows the different data-handling requirements of the two models and preserves the categorical information used by CTGAN for conditional generation.

The dataset was originally distributed into separate training and test sets by the UCI Machine Learning Repository [49]. To prevent data leakage, the two subsets were processed independently. The training set contained 34,561 records, while the test set contained 14,279 records.

3.3.2 Credit Card Fraud Detection Dataset

The Credit dataset did not contain any missing values and was already numerically encoded, as all features were transformed using PCA. Therefore, no imputation or encoding was necessary. The dataset was split into a training set (70%) and a test set (30%) while maintaining the original class distribution.

Feature standardization was applied only to the training data prior to training WCGAN-GP or any sequential framework that included WCGAN-GP. This step was included to replicate the preprocessing strategy used in the original WCGAN-GP [27] study and to improve numerical stability during training.

CTGAN was trained directly on the raw training data without applying any additional preprocessing beyond the initial data cleaning.

3.4 CTWGAN-GP Sequential GAN Framework

The CTWGAN-GP framework is a two-stage generative framework that integrates two distinct GAN models, WCGAN-GP and CTGAN, sequentially. The framework is implemented in two model-ordering variants: WCGAN-GP → CTGAN and CTGAN → WCGAN-GP. In each configuration, the synthetic data generated by the first-stage model are combined with the real training data and used to train the second-stage model, which generates the final synthetic dataset. The overall framework and both ordering variants are illustrated in Figure 4.

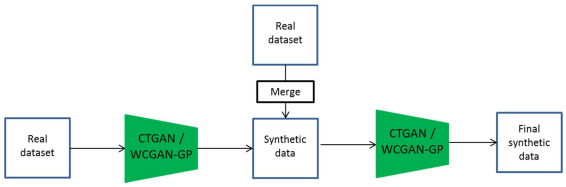


Figure 4: CTWGAN-GP sequential Framework with interchangeable ordering of WCGAN-GP and CTGAN.

3.4.1 Conditional Tabular GAN (CTGAN)

CTGAN [15] is a conditional generative adversarial network specifically designed to synthesize tabular datasets containing both continuous and categorical variables, including imbalanced class distributions. It incorporates mode-specific normalization, conditional generation, and training-by-sampling to model heterogeneous tabular features and address imbalanced categorical distributions [15]. The architecture of CTGAN is illustrated in Figure 5. Further technical details are provided in the original CTGAN study [15].

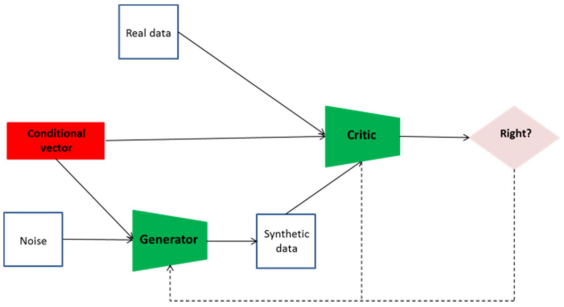


Figure 5: Structure of CTGAN.

3.4.2 Wasserstein Conditional GAN with Gradient Penalty (WCGAN-GP)

WCGAN-GP [27] is a conditional generative adversarial model that builds upon the Wasserstein GAN with Gradient Penalty (WGAN-GP) [50]. It uses the Wasserstein objective and gradient penalty to improve training stability and distributional learning for tabular data [27]. The structure of WCGAN-GP is illustrated in Figure 6. Further technical details are provided in the original WCGAN-GP study [27].

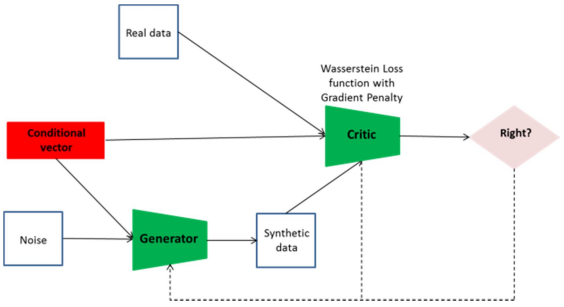


Figure 6: Structure of WCGAN-GP.

3.5 Model Implementation and Architecture

3.5.1 CTGAN

The CTGAN model follows a fully connected feed-forward structure designed for tabular data, where spatial locality is absent. The generator takes as input a concatenation of a latent noise vector z and a conditional vector representing the class label. This input is passed through two hidden layers, each followed by batch normalization and ReLU activation functions. The output layer uses a mixture of activation functions: continuous columns are modeled with tanh, while categorical and discrete variables are handled using the Gumbel-Softmax function for differentiable sampling.

The critic (or discriminator) similarly consists of two hidden layers with LeakyReLU activations and dropout regularization. It is augmented with the PacGAN framework (packing size = 10), allowing the critic to evaluate groups of 10 samples simultaneously to reduce mode collapse. The conditional vector is also concatenated with the inputs to guide learning.

The detailed architectural specifications of the CTGAN generator and critic are presented in Tables 4 and 5, respectively.

Table 4: General architecture of the CTGAN Generator.

Component	Description
Input	Concatenation of a random noise vector (z) and a conditional vector encoding the target category
Hidden Layers	Two fully connected layers (256 units each) with batch normalization and ReLU activation
Output	• Continuous values: generated using `tanh` • Discrete values: generated using Gumbel-softmax for differentiable sampling

Table 5: General architecture of the CTGAN Critic (Discriminator).

Component	Description
Input	Pack of 10 samples (PacGAN), each consisting of a full data row concatenated with its condition vector
Hidden Layers	Two fully connected layers (256 units each) with Leaky ReLU activation (α = 0.2)
Regularization	Dropout applied after each hidden layer to reduce overfitting
Output	A single scalar value estimating the Wasserstein distance

3.5.2 WCGAN-GP

The WCGAN-GP model implemented in this research follows the conditional Wasserstein GAN architecture with a gradient penalty, specifically adapted for tabular data synthesis. The model consists of two neural networks: a generator and a critic, both conditioned on class labels to facilitate class-aware data generation.

The generator receives two inputs: a random noise vector of 32 dimensions and a label vector.

These are concatenated and passed through a series of fully connected layers with increasing dimensionality: 128, 256, and 512 units, respectively. Each layer is followed by a LeakyReLU activation with a negative slope of 0.2 and a dropout layer with a rate of 0.3 to reduce overfitting. The output layer is a dense layer with linear activation that generates a synthetic sample matching the structure of the real dataset.

The critic accepts either real or synthetic data samples, each concatenated with the corresponding class label. It follows a mirrored architecture with three fully connected layers of 512, 256, and 128 units, respectively, and employs LeakyReLU activations throughout. Unlike traditional GAN discriminators, the critic outputs a scalar Wasserstein score representing the relative realism of the input samples. A gradient penalty term with $\lambda = 10$ is included in the loss function to enforce the 1-Lipschitz constraint.

Tables 6 and 7 summarize the architecture of the WCGAN-GP generator and critic, respectively.

Table 6: General architecture of the WCGAN-GP Generator.

Component	Description
Input	Concatenation of a noise vector (z) and an integer-encoded class label
Hidden Layer 1	Dense layer with 128 units, followed by LeakyReLU ($\alpha = 0.2$) and dropout
Hidden Layer 2	Dense layer with 256 units, followed by LeakyReLU ($\alpha = 0.2$) and dropout
Hidden Layer 3	Dense layer with 512 units, followed by LeakyReLU ($\alpha = 0.2$) and dropout
Output	Dense layer with linear activation, producing output with the same dimensionality as the real data

Table 7: General architecture of the WCGAN-GP Critic.

Component	Description
Input	Concatenation of a data sample (real or generated) and an integer-encoded class label
Hidden Layer 1	Dense layer with 512 units, followed by LeakyReLU ($\alpha = 0.2$) and dropout
Hidden Layer 2	Dense layer with 256 units, followed by LeakyReLU ($\alpha = 0.2$) and dropout
Hidden Layer 3	Dense layer with 128 units, followed by LeakyReLU ($\alpha = 0.2$) and dropout
Output	Dense layer without activation (linear output), returning a single scalar value

3.6 Training Procedure

The standalone CTGAN and WCGAN-GP models were first trained independently on their respective training data to establish the baseline configurations. The same model-specific training procedures were then used within the sequential CTWGAN-GP configurations.

3.6.1 CTGAN Training

The CTGAN model was trained following the settings proposed by Xu et al. [15]. The generator and critic were updated alternately using the Adam optimizer with a learning rate of 2×10^{-4} and $\beta_1 = 0.5$, $\beta_2 = 0.9$. The model was trained for 300 epochs with a batch size of 500. During training, the conditional vector encoding the class label was concatenated to both the generator input and the critic input, allowing the model to generate class-specific samples. A Gumbel-softmax trick was applied to enable differentiable sampling for categorical features.

Once training was completed, the CTGAN generator was used to produce synthetic samples matching the structure and schema of the original training data.

3.6.2 WCGAN-GP Training

The WCGAN-GP model was trained using the Wasserstein loss with gradient penalty to enforce the 1-Lipschitz constraint. The generator was updated once for every five critic updates, as suggested in the original WCGAN-GP framework [27]. The gradient penalty coefficient λ was set to 10. Both networks used the Adam optimizer with a learning rate of 0.0001 and parameters $\beta_1 = 0.5$ and $\beta_2 = 0.9$. A batch size of 64 was used, and the model was trained for 300 epochs. The class label vector was concatenated to both real and generated samples to condition the generation process. Dropout layers were included in the generator to reduce overfitting. After training, the generator was used to sample synthetic data with the same dimensionality and structure as the original dataset.

Because the input data underwent normalization and encoding during preprocessing, the generated data were subjected to inverse transformations to recover the original feature scale and categorical labels. These transformations were applied column-wise using the corresponding scalers and encoders from the preprocessing stage, ensuring that the final synthetic data were returned to the original data representation.

3.7 Evaluation Strategy

To assess the effectiveness of the proposed CTWGAN-GP framework in generating high-quality synthetic tabular data, we conducted a series of experiments using two labeled real-world datasets. The sequential configurations were compared with two standalone baselines: CTGAN [15] and WCGAN-GP [27].

To evaluate the practical utility of the generated synthetic data, we adopted a classifier-based evaluation framework that focuses on downstream machine learning performance. The core idea is to measure how well a classifier trained on synthetic data can generalize to real, unseen samples. This strategy serves as a practical proxy for determining the utility of synthetic data for downstream classification tasks.

3.7.1 Evaluation Metrics

To evaluate the performance of the classifier trained on synthetic data, we report five standard classification metrics: Accuracy, Precision, Recall, F1-Score, and ROC AUC Score. While Accuracy provides a broad measure of correctness, it is often less informative in imbalanced settings. Therefore, F1-Score and ROC AUC are particularly emphasized as more stable and reliable indicators under class imbalance, which is inherent in both the Adult and Credit Card Fraud datasets.

The evaluation metrics used in this research are formally defined as follows:

- Accuracy: Measures the overall correctness of the classifier and is calculated as:

$$\text{Accuracy} = \frac{TP+TN}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \quad (1)$$

where TP = True Positives, TN = True Negatives, FP = False Positives, and FN = False Negatives.

- F1 Score: Provides the harmonic mean of precision and recall and is calculated as:

$$\text{F1 Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \quad (2)$$

- Precision: Indicates the proportion of correctly predicted positive instances among all instances predicted as positive and is calculated as:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \quad (3)$$

- Recall: Reflects the proportion of correctly predicted positive instances among all actual positive instances and is calculated as:

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \quad (4)$$

- ROC AUC Score: Measures the area under the Receiver Operating Characteristic (ROC) curve, which plots the True Positive Rate (TPR) against the False Positive Rate (FPR) at various classification thresholds. It is formally defined as:

$$\text{AUC} = \int_0^1 \text{TPR(x)}\, dx \quad (5)$$

where TPR = $\frac{TP}{TP+}$ and FPR = $\frac{FP}{FP+TN}$. The AUC value ranges from 0 to 1, with higher values indicating better model performance.

3.7.2 Utility-based Evaluation

To assess the practical usefulness of the synthetic data, we adopted a utility-based evaluation framework based on the well-established Train on Synthetic, Test on Real (TSTR) strategy. In this setup, a machine learning classifier is trained on synthetic data and then evaluated on real, unseen test data. This approach serves as a rigorous benchmark for measuring how well synthetic data preserves the discriminative patterns of the original distribution in downstream classification tasks. The TSTR method has been widely used in prior studies, including the original CTGAN paper [15].

Figure 7 illustrates the TSTR evaluation pipeline used in our experiments.

For the classifier, we employed Extreme Gradient Boosting (XGBoost) [51] due to its strong performance on tabular data and its ability to handle heterogeneous tabular data efficiently. XGBoost also provides fast training performance, which is essential when conducting repeated evaluations across multiple synthetic datasets. Accordingly, XGBoost was used consistently as the downstream classifier for evaluating the synthetic datasets generated by all sequential and standalone configurations.

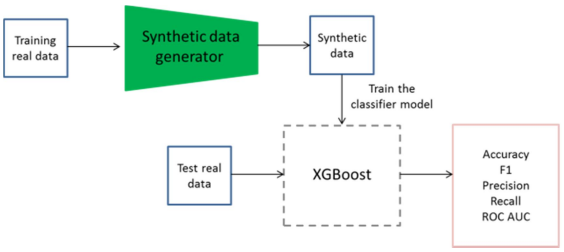


Figure 7: Evaluation flows for ML utility of classification.

3.7.3 Comparative Baseline Experiments

To assess the performance of the proposed CTWGAN-GP framework, we conducted comparative experiments with two standalone generative models, CTGAN [15] and WCGAN-GP [27], which also form the core components of the proposed sequential framework. This comparative setup enables us to evaluate the individual baseline performance of each model and determine whether their sequential integration within CTWGAN-GP leads to measurable improvements in synthetic data quality and utility.

To ensure a fair and reproducible comparison, we implemented both baseline models within the same experimental environment. All models were trained using the same training–testing split for each dataset and the same number of training epochs (300), ensuring consistency across evaluations.

The CTGAN model was trained using the official implementation provided by the Synthetic Data Vault (SDV) library (version 1.4.0), while WCGAN-GP was manually implemented in TensorFlow/Keras based on the architectural and training specifications outlined in its original publication. No modifications were made to their core structures beyond those necessary for compatibility with our experimental setup.

This re-implementation and unified evaluation setup was necessary because the original studies used different data splits, evaluation strategies, or reported results that were not directly comparable to our experimental setup. Therefore, our baseline results reflect performance under a unified and reproducible setting.

3.8 Computational Environment and Tools

Experiments were conducted on Google Colab Pro, utilizing NVIDIA Tesla T4 (15.3 GB VRAM) and NVIDIA A100 (40 GB VRAM) GPUs, with approximately 53.5 GB of system RAM, CUDA 12.4, and Python 3.11.12. The CTGAN model was implemented using the SDV library (v1.4.0), while WCGAN-GP was developed from scratch using Keras and TensorFlow (v2.15.0), following the architecture described in [27]. Data preprocessing and evaluation were performed using scikit-learn (v1.2.2), pandas (v1.5.3), numpy (v1.24.3), and matplotlib (v3.7.1).

4. RESULTS

This section presents the experimental results of the baseline models (CTGAN and WCGAN-GP) and the proposed sequential framework (CTWGAN-GP) across two datasets: Adult Income [49] and Credit Card Fraud Detection [48]. All models were evaluated using the Train-on-Synthetic-Test-on-Real (TSTR) strategy, with XGBoost as the classifier. Each model generated synthetic data with the same number of samples as the original training set: 34,561 for the Adult dataset and 199,364 for the Credit dataset. All classifiers were trained and evaluated under identical settings to ensure a fair comparison.

4.1 Adult Dataset Results

Table 8 summarizes the classification performance using synthetic data generated by each model on the Adult dataset. The classifier trained on real data achieved the highest overall performance, as expected. Among the synthetic models, CTGAN outperformed WCGAN-GP in terms of F1-score and ROC AUC.

The proposed CTWGAN-GP (WCGAN-GP → CTGAN) variant achieved an F1-score of 0.6454, outperforming both baseline models—CTGAN (0.6067) and WCGAN-GP (0.4195). This result indicates that the sequential configuration provided better downstream classification performance and practical utility than either standalone baseline. In contrast, the CTGAN → WCGAN-GP configuration performed worse than CTGAN in terms of F1-score (0.5650 vs. 0.6067) and ROC AUC (0.8100 vs. 0.8815), but outperformed WCGAN-GP on both metrics, achieving a higher F1-score (0.5650 vs. 0.4195) and ROC AUC (0.8100 vs. 0.7611).

4.2 Credit Card Fraud Detection Dataset Results

Table 9 shows the classification performance for synthetic data generated on the Credit Card Fraud Detection dataset. As observed for the Adult dataset, models trained on real data achieved the best performance across all metrics.

Among synthetic methods, CTGAN achieved the highest recall (0.7973), but at the cost of low precision (0.3460), resulting in a relatively modest F1-score (0.4826). WCGAN-GP delivered a more

Table 8:Adult Dataset Classification Performance Using Real and Synthetic Data with XGBoost..

Method	Accuracy	Precision	Recall	F1-Score	ROC AUC
Real Data	0.8698	0.7620	0.6526	0.7031	0.9259
CTGAN	0.8327	**0.6825**	0.5460	0.6067	0.8815
WCGAN-GP	0.7956	0.6377	0.3125	0.4195	0.7611
(CTWGAN-GP) WCGAN-GP → CTGAN	**0.8334**	0.6491	**0.6417**	**0.6454**	**0.8834**
(CTWGAN-GP) CTGAN → WCGAN-GP	0.7899	0.5529	0.5777	0.5650	0.8100

Bold values indicate the best performance among synthetic data models.

Table 9: Credit Card Dataset Classification Performance Using Real and Synthetic Data with XGBoost.

Method	Accuracy	Precision	Recall	F1-Score	ROC AUC
Real Data	0.9995	0.9407	0.7500	0.8346	0.9692
CTGAN	0.9970	0.3460	**0.7973**	0.4826	**0.9657**
WCGAN-GP	**0.9992**	**0.8279**	0.6824	**0.7481**	0.9424
(CTWGAN-GP) WCGAN-GP → CTGAN	0.9985	0.5781	0.5000	0.5362	0.8891
(CTWGAN-GP) CTGAN → WCGAN-GP	0.9987	0.7200	0.3649	0.4843	0.9003

Bold values indicate the best performance among synthetic data models.

balanced performance with an F1-score of 0.7481, indicating stronger downstream classification performance than CTGAN.

In contrast, the CTWGAN-GP variants demonstrated reduced effectiveness. The WCGAN-GP → CTGAN configuration achieved a recall of 0.5000 and an F1-score of 0.5362, while the reverse order (CTGAN → WCGAN-GP) performed slightly lower in terms of F1-score but achieved better precision. These results suggest that, for the highly imbalanced Credit Card Fraud dataset, standalone WCGAN-GP provided stronger downstream machine learning utility than either sequential configuration.

5. DISCUSSION

5.1 Adult Dataset Analysis

Table 8 provides a comparative evaluation of the classifier performance using synthetic data generated by standalone and sequential GAN models, with evaluation performed on the real test split of the Adult dataset. This dataset is characterized by a mixture of numerical and categorical features, moderate class imbalance (approximately 76% negative class), and weak feature correlations. These characteristics make it a suitable benchmark for assessing a model's ability to preserve both structure and class distribution fidelity.

Among the synthetic models, the proposed sequential configuration CTWGAN-GP (WCGAN-GP → CTGAN) achieved superior performance in Recall (0.6417), F1-score (0.6454), and ROC AUC (0.8834). This result suggests that the effectiveness of the sequential framework is influenced by the complementary characteristics of the two models and their ordering. WCGAN-GP, used as the first stage, generates diverse synthetic samples with continuous feature realism by leveraging the Wasserstein loss and gradient penalty, which are designed to improve training stability and distributional learning [27]. However, WCGAN-GP lacks mechanisms for modeling complex categorical distributions. Feeding its output—combined with the real data—into CTGAN allows CTGAN to refine the categorical feature relationships through its conditional vector and training-by-sampling strategy [15]. This sequential hand-off may therefore allow the second-stage CTGAN to further model categorical relationships while building on the representation learned by the first-stage model, which may explain the improved downstream classification performance.

In contrast, the reverse configuration CTWGAN-GP (CTGAN → WCGAN-GP) performed worse on most metrics, including F1-score (0.5650) and ROC AUC (0.8100). While CTGAN can effectively model categorical structures in the first stage, its synthetic outputs—when merged with real data and passed to WCGAN-GP—may lack sufficient

diversity in the numerical domain due to mode collapse. WCGAN-GP, although strong in stabilizing training for continuous features, lacks the same conditional mechanisms as CTGAN and may not effectively enhance the categorical information provided by CTGAN. This ordering thus may limit the overall data utility and generalization. However, this variant still outperformed standalone WCGAN-GP, providing evidence that sequential training can provide gains even with a less effective model ordering.

Among the standalone models, CTGAN outperformed WCGAN-GP, especially in Precision (0.6825) and ROC AUC (0.8815). This aligns with CTGAN's design: it uses a conditional generator that enables it to focus on underrepresented classes and employs a training-by-sampling strategy to improve exposure to minority categories during training [15]. However, its lower Recall (0.5460) and F1-score (0.6067) indicate that its generated data did not preserve all class-discriminative patterns equally well for downstream classification.

On the other hand, WCGAN-GP recorded the lowest performance across most metrics, notably Recall (0.3125) and F1-score (0.4195). This lower performance may be related to the mixed numerical and categorical structure of the Adult dataset and the absence of CTGAN's specialized mechanisms for categorical and imbalanced data. Its architecture is therefore less suited to some of the characteristics of this mixed-type dataset, which may have limited the downstream utility of its generated data.

These results highlight the importance of model ordering in sequential architectures and provide evidence that the CTWGAN-GP framework, when configured appropriately, can improve synthetic data utility for mixed-type tabular datasets.

5.2 Credit Dataset Analysis

Table 9 presents the classification performance on synthetic data generated for the Credit Card Fraud Detection dataset, which is highly imbalanced (only 0.17% fraud cases) and consists entirely of numerical features derived via PCA transformation. Among the synthetic models, the standalone WCGAN-GP achieved the best overall performance, with an F1-score of 0.7481 and high precision (0.8279), closely approaching the results obtained from the real data. This strong performance can be attributed to WCGAN-GP's design, which emphasizes training stability and the ability to model smooth, continuous feature distributions through the use of the Wasserstein loss and gradient penalty. Given that the Credit dataset lacks categorical features and exhibits highly imbalanced numerical data, WCGAN-GP appears well suited for capturing its underlying structure.

In contrast, CTGAN, which was originally designed to handle mixed data types and complex categorical dependencies, achieved the highest recall (0.7973) but suffered from low precision (0.3460), resulting in a lower F1-score (0.4826). This trade-off suggests that CTGAN generated a large number of positive (fraudulent) instances, increasing recall but also introducing many false positives, which negatively affected overall utility. This behavior is typical when the model attempts to mitigate extreme class imbalance without effective regularization for numeric feature boundaries.

The proposed sequential model (CTWGAN-GP), in both ordering variants, demonstrated lower performance compared to the standalone WCGAN-GP. In the WCGAN-GP → CTGAN configuration, the model achieved an F1- score of 0.5362 and a recall of 0.5000. This degradation occurs because CTGAN's conditional sampling and mode-specific normalization—mechanisms designed primarily for mixed-type tabular data—introduce discretization effects that distort the smooth continuous distributions previously learned by WCGAN-GP. As a result, the WCGAN-GP → CTGAN variant loses part of the numerical stability established in the first stage. Similarly, the CTGAN → WCGAN-GP variant achieved an F1-score of 0.4843, indicating that the initial CTGAN-generated samples lacked the statistical robustness needed to support the second-stage WCGAN-GP training.

These results suggest that in fully numerical and highly imbalanced datasets, such as the Credit dataset, the direct application of WCGAN-GP outperforms more complex sequential architectures. Therefore, the effectiveness of CTWGAN-GP is context-dependent and should be evaluated in relation to the structural composition of the target dataset.

Nevertheless, both CTWGAN-GP variants achieved slightly higher F1-scores than standalone CTGAN (0.5362 and 0.4843 vs. 0.4826). Although the improvements were small, this result suggests that sequential integration can provide limited benefits over standalone CTGAN, even when it does not outperform the WCGAN-GP baseline.

5.3 Cross-Dataset Comparison and Interpretation

The experimental results demonstrate that the effectiveness of the proposed CTWGAN-GP framework varies according to the structural

characteristics of the dataset. On the Adult dataset, which contains mixed numerical and categorical features, moderate class imbalance, and missing values, the WCGAN-GP → CTGAN configuration outperformed both standalone models, indicating that sequential integration can be beneficial when the characteristics of the two models are complementary. In contrast, on the Credit Card Fraud Detection dataset, which consists entirely of numerical PCA-transformed features and exhibits extreme class imbalance, standalone WCGAN-GP achieved the highest performance, suggesting that the additional architectural complexity of sequential integration does not necessarily provide an advantage for purely numerical data.

These results indicate that the effectiveness of sequential GAN integration is dataset-dependent rather than universally superior to standalone generation. The benefits of combining models appear to depend on whether their respective capabilities align with the characteristics of the target dataset. The results also show that WCGAN-GP → CTGAN generally outperformed the reverse ordering, particularly on the Adult dataset, where CTGAN could further model categorical relationships after the first-stage generation. However, the advantage of this ordering was less pronounced on the Credit dataset, where the standalone WCGAN-GP remained the strongest approach.

6. ADDITIONAL EXPERIMENTS

6.1 Effect of Preprocessing and Postprocessing on Model Performance

To explore whether preprocessing and postprocessing steps can improve model performance, a set of controlled experiments was conducted using the Adult dataset. The goal was to test whether data transformations applied before training and corrections applied after generation could enhance the overall quality and utility of synthetic data.

The preprocessing pipeline included: (1) log-transforming skewed numerical features to stabilize variance and approximate normality; and (2) grouping rare categories in the native-country feature to reduce sparsity and minimize the impact of extreme low-frequency values, which was particularly important because WCGAN-GP was observed to be sensitive to outliers and rare categorical values, potentially resulting in unstable training or unrealistic outputs.

For postprocessing, two types of transformations were introduced after synthetic data generation: (1) rounding integer-like columns—such as age or education-num—to ensure semantic validity; and (2) clipping negative values to zero, particularly for columns where negative values are logically invalid (e.g., hours-per-week). These actions were specifically designed to address artifacts observed in WCGAN-GP's output, where the lack of explicit output constraints sometimes led to implausible numerical values.

Table 10 demonstrates a clear contrast in how preprocessing and postprocessing steps influenced the independent models versus the sequential CTWGAN-GP configurations. For standalone models, particularly WCGAN-GP, these processing steps significantly enhanced performance. WCGAN-GP saw its recall increase from 0.3125 to 0.5580 and F1-score from 0.4195 to 0.5406, highlighting how input normalization, category grouping, and output correction can improve its downstream performance. CTGAN also experienced slight improvements in recall and F1-score after postprocessing, indicating improved alignment between its generated outputs and the classification task.

In contrast, the sequential CTWGAN-GP configurations responded inconsistently. Surprisingly, both pipeline variants—CTGAN → WCGAN-GP and WCGAN-GP → CTGAN—performed better without any preprocessing or postprocessing. The WCGAN-GP → CTGAN configuration, in particular, saw a substantial drop in F1-score from (0.6454) to (0.4939) after applying the same data corrections, despite an increase in recall. Our experiments show that preprocessing and postprocessing steps, which improve standalone models, can reduce performance in the sequential CTWGAN-GP pipeline. This indicates that multi-stage architectures have unique dynamics, where transformations beneficial for one stage may interfere with the subsequent stage's ability to model the generated data effectively.

Most notably, the WCGAN-GP → CTGAN configuration achieved the best overall performance without any processing, with an F1-score of (0.6454) and ROC AUC of (0.8834). This outperformed all standalone models, even when they were improved with processing. It suggests that the strength

of the proposed CTWGAN-GP model comes from its architecture itself, not from external data corrections. These findings highlight that while

preprocessing and postprocessing are beneficial, their success depends on model compatibility.

Table 10: Effect of Preprocessing and Postprocessing on Model Performance (Adult Dataset).

Method	Setting	Accuracy	Precision	Recall	F1-Score	ROC AUC
CTGAN	Without Processing	0.8327	0.6825	0.5460	0.6067	0.8815
CTGAN	With Postprocessing	0.8286	0.6547	0.5809	0.6156	0.8748
WCGAN-GP	Without Processing	0.7956	0.6377	0.3125	0.4195	0.7611
WCGAN-GP	With Pre/Post-processing	0.7759	0.5242	0.5580	0.5406	0.7955
(CTWGAN-GP) CTGAN → WCGAN-GP	Without Processing	0.7899	0.5529	0.5777	0.5650	0.8100
(CTWGAN-GP) CTGAN → WCGAN-GP	With Pre/Post-processing	0.7904	0.6140	0.3040	0.4066	0.7712
(CTWGAN-GP) WCGAN-GP → CTGAN	Without Processing	0.8334	0.6491	0.6417	0.6454	0.8834
(CTWGAN-GP) WCGAN-GP → CTGAN	With Pre/Post-processing	0.6265	0.3632	0.7715	0.4939	0.7558

To the best of our knowledge, this is the first comparative study to quantify how pre- and post-processing differently impact standalone vs. sequential GAN models for tabular data.

6.2 Effect of Increasing the Synthetic Data Size

To test the ability of the proposed CTWGAN-GP model to generate larger volumes of synthetic data, we conducted an experiment where the final generator was configured to produce twice the number of samples relative to the original training dataset using the Adult dataset. The architecture, training configuration, and data composition remained unchanged; only the number of generated records at the end was increased. Instead of generating synthetic data equal in size to the original training set, the synthetic data size was doubled (i.e., 65,122 rows vs. 34,561).

This design decision was motivated by the limitations observed in prior studies, where evaluations often use synthetic datasets with a size similar to that of the original data [15], [27], [23], [33]. Such setups do not fully explore the generative capacity or scalability of the models. By generating a larger synthetic dataset, we aimed to evaluate whether the proposed CTWGAN-GP model could maintain its performance when tasked with producing a larger synthetic dataset, especially under class imbalance and structural diversity.

In the WCGAN-GP → CTGAN configuration, the classifier trained on double-sized synthetic data achieved an F1- score of (0.6165) and ROC AUC of (0.8706), showing robustness to the increased volume. In contrast, the CTGAN → WCGAN-GP configuration resulted in an F1-score of (0.5754) and ROC AUC of (0.8120).

These results suggest that the CTWGAN-GP model, particularly when WCGAN-GP precedes CTGAN, can generate a larger synthetic dataset while retaining relatively strong downstream utility,

although performance decreased compared with the original synthetic data size. This finding indicates potential for data augmentation in tabular domains where larger synthetic datasets may be useful, particularly when data scarcity or privacy constraints limit access to real records (see Table 11).

6.3 Balanced Synthetic Generation (Class-Level Conditioning)

To investigate the effect of generating class-balanced synthetic data in the second stage, we modified the CTWGANGP pipeline to use the sample_from_conditions function available in SDV 1.4+. Specifically, the CTGAN model was instructed to generate an equal number of samples for each class label (income) in the Adult dataset, resulting in a perfectly balanced synthetic dataset with respect to class labels. Compared to the CTWGAN-GP (WCGAN-GP → CTGAN), which learns and replicates the natural class imbalance of the training data, this variant achieved a notably higher recall (0.8500) but at the cost of reduced precision (0.5172), as shown in Table 12. The resulting F1-score (0.6431) and ROC AUC (0.8828) were comparable to the original configuration, suggesting a trade-off between minority class sensitivity and predictive reliability.

Although the generated dataset was numerically balanced, the precision drop indicates that the synthetic samples for the minority class (>50K) may not have fully captured the true distributional

Table 11: Performance of CTWGAN-GP Model with Double Synthetic Sample Size (Adult Dataset).

Configuration	Number of Generated Records	Accuracy	Precision	Recall	F1-Score	ROC AUC
(CTWGAN-GP) WCGAN-GP → CTGAN	34,561 rows	0.8334	0.6491	0.6417	0.6454	0.8834
(CTWGAN-GP) WCGAN-GP → CTGAN	65,122 rows	0.8289	0.6553	0.5822	0.6165	0.8706
(CTWGAN-GP) CTGAN → WCGAN-GP	34,561 rows	0.7899	0.5529	0.5777	0.5650	0.8100
(CTWGAN-GP) CTGAN → WCGAN-GP	65,122 rows	0.7827	0.5344	0.6232	0.5754	0.8120

Table 12: Effect of Balanced Synthetic Data Generation Using Fixed Class-Level Conditions in CTGAN (Adult Dataset).

Configuration	Accuracy	Precision	Recall	F1-Score	ROC AUC
CTWGAN-GP (WCGAN-GP → CTGAN)	0.8334	0.6491	0.6417	0.6454	0.8834
CTWGAN-GP (WCGAN-GP → CTGAN) (Balanced Output)	0.7771	0.5172	0.8500	0.6431	0.8828

boundaries. Instead, the model may have over-generated ambiguous or noisy instances that overlapped with the majority class (<=50K), increasing false positives during classification. This reflects a known limitation of class-conditional generation: quantitative balance does not necessarily imply qualitative representativeness.

In summary, enforcing class balance during generation can help increase recall by improving minority class coverage, but it also requires careful control to avoid precision degradation. This result underscores the importance of combining conditional sampling with distribution-aware refinements to maintain both class balance and data fidelity.

6.4 Effect of Increasing Training Epochs in the Second Stage (CTGAN)

To assess whether prolonged training of the second-stage model in the CTWGAN-GP pipeline can enhance performance, a new experiment was conducted using the Adult dataset and the WCGAN-GP → CTGAN configuration, where CTGAN was trained for 3000 epochs instead of the default 300.

The motivation for this test stems from the observation that the WCGAN-GP → CTGAN sequence consistently outperformed other configurations. The hypothesis was that allowing CTGAN to train longer on the data produced by the first-stage WCGAN-GP model might lead to more refined and representative synthetic outputs, particularly benefiting minority class modeling.

As shown in Table 13, the extended training improved both recall and F1-score, with the final F1-score reaching 0.6532, compared with 0.6454 in the default configuration. Recall also increased to 0.6576, indicating better sensitivity to the positive class. Although the ROC AUC remained almost unchanged (0.8833 vs. 0.8834), the overall balance between precision and recall improved, indicating that the extended second-stage training provided a modest improvement in downstream performance without reducing ROC AUC.

These results suggest that prolonging training in the CTGAN stage can yield modest yet meaningful performance gains, particularly within the WCGAN-GP → CTGAN configuration. This reinforces the value of considering training duration as an important design parameter in sequential GAN architectures, particularly when different models are assigned distinct roles across stages.

6.5 Ablation Study: Removing Real Data from Stage Two

To evaluate the importance of combining real data with synthetic data in the second stage of the CTWGAN-GP (WCGAN-GP → CTGAN) model, we conducted an ablation experiment in which the CTGAN model was trained solely on synthetic data generated by the first-stage WCGAN-GP, without incorporating any real samples. In contrast, the proposed CTWGAN-GP (WCGAN-GP →

CTGAN) trains CTGAN on a hybrid dataset composed of both real and synthetic records.

Table 13: Effect of Increasing Epochs in the Second Stage (CTGAN) in the CTWGAN-GP Pipeline (Adult Dataset).

Configuration	CTGAN Epochs	Accuracy	Precision	Recall	F1-Score	ROC AUC
(CTWGAN-GP) WCGAN-GP → CTGAN (Default)	300	0.8334	0.6491	0.6417	0.6454	0.8834
(CTWGAN-GP) WCGAN-GP → CTGAN (Extended Training)	3000	0.8351	0.6490	0.6576	0.6532	0.8833

Table 14: Ablation Study on Removing Real Data in Stage Two of CTWGAN-GP (Adult Dataset).

Configuration	Accuracy	Precision	Recall	F1-Score	ROC AUC
(CTWGAN-GP) WCGAN-GP → CTGAN	0.8334	0.6491	0.6417	0.6454	0.8834
(CTWGAN-GP) WCGAN-GP → CTGAN (No Real Data in Stage Two)	0.8270	0.6528	0.5720	0.6098	0.8715

As shown in Table 14, removing the real data from the second-stage training led to a noticeable performance decline. The F1-score dropped from (0.6454) in the proposed configuration to (0.6098), and the ROC AUC decreased from (0.8834) to (0.8715).

This suggests that including real data in the second-stage training contributes to the downstream utility of the sequential framework. The inclusion of real data provides CTGAN with direct access to the original data distribution in addition to the synthetic samples generated by WCGAN-GP, which may help preserve structural and categorical patterns during the second-stage training. In contrast, although the performance decreased compared to the proposed configuration, the results remain relatively strong. This indicates that training CTGAN solely on WCGAN-GP output can still produce useful synthetic data, but does not achieve the same downstream utility as the hybrid real–synthetic training configuration.

7. CONCLUSIONS

This study proposed and evaluated CTWGAN-GP, a sequential generative framework that integrates CTGAN and WCGAN-GP to investigate whether integrating two distinct GAN models in a multi-stage pipeline can improve the quality and practical utility of synthetic tabular data. The study systematically examined the effects of model ordering and training configuration across datasets with different structural characteristics, comparing the sequential configurations with standalone CTGAN and WCGAN-GP using downstream machine learning utility.

The experimental results demonstrate that sequential integration can provide a meaningful advantage when the participating models are compatible with the target dataset. On the Adult dataset, the WCGAN-GP → CTGAN configuration achieved the highest F1-score and ROC AUC among the evaluated synthetic-data approaches, while the reverse ordering achieved lower performance, highlighting the importance of model ordering in sequential generative pipelines. In contrast, standalone WCGAN-GP achieved the strongest performance on the highly imbalanced, fully numerical Credit Card Fraud Detection dataset, indicating that the benefits of sequential integration depend on the structural characteristics of the dataset and the compatibility between the integrated models.

Several additional experiments were conducted to further examine the robustness and practical behavior of the proposed framework under different training and data-generation settings. These experiments investigated training duration, synthetic data volume, class-level conditioning, preprocessing and postprocessing, and the role of real data in second-stage training. Overall, the findings showed that sequential performance can be improved through appropriate training adjustments, while some strategies that benefit standalone models may not provide the same benefits within a sequential pipeline. This further highlights the importance of carefully designing and tuning each stage of sequential generative frameworks.

From a practical and professional perspective, the findings have implications for organizations and practitioners that use machine learning with tabular data, particularly where obtaining sufficient real training data is difficult, costly, or subject to access

constraints. The CTWGAN-GP framework provides a sequential approach for generating synthetic tabular data and assessing its practical utility through downstream machine learning performance. This approach may support the preliminary development and evaluation of machine learning models when access to real data is limited, although synthetic data should be validated for the specific application before operational use. The findings further suggest that data scientists and machine learning practitioners should consider dataset characteristics when selecting model ordering and training configurations, rather than assuming that a single sequential configuration will be optimal across datasets.

More broadly, these findings provide an empirical basis for further investigation of sequential GAN architectures and highlight multi-stage generative pipelines as a promising direction for future research in synthetic tabular data generation.

8. FUTURE WORK AND OPEN RESEARCH ISSUES

The findings of this study provide empirical evidence on the practical utility of sequential GAN integration while also highlighting several directions that warrant further investigation. In particular, the observed dependence of performance on dataset characteristics, model ordering, and training configuration raises open research questions regarding the generalizability, scalability, and broader applicability of the CTWGAN-GP framework. These questions motivate the following directions for future research.

Although the Adult and Credit datasets used in this study contain a moderate number of features, they do not represent extremely high-dimensional tabular settings. Evaluating the scalability of the sequential CTWGAN-GP framework on datasets with substantially larger numbers of features remains an important direction for future work.

Additionally, the generalizability of the framework could be assessed on more diverse datasets from different domains, such as medical records and financial risk control, and on datasets with varying characteristics, including extreme sparsity, a large number of classes, or multi-class structures. Evaluating the framework under such conditions would help determine whether the dataset-dependent effects observed in this study extend beyond the datasets evaluated here.

Finally, the generated data should be evaluated using complementary measures beyond downstream machine learning utility, such as privacy preservation, statistical similarity, and domain-specific quality indicators. Future studies should also assess performance across multiple classifier types to provide a more comprehensive and classifier-agnostic evaluation of synthetic data quality.

Data Availability Statement: This research used the Credit Card Fraud Detection dataset [48] and the Adult Income dataset [49], which are publicly available datasets.

Acknowledgments: Some AI-assisted tools were used for partial proofreading and grammatical refinement. All methodologies, experiments, and conclusions are original contributions for which the authors assume full responsibility.

Abbreviations

Key abbreviations used in this manuscript include:

Abbreviation	Definition
AI	Artificial Intelligence
ML	Machine Learning
DNN	Deep Neural Networks
GAN	Generative Adversarial Network
CTWGAN-GP	Conditional Tabular Wasserstein GAN with Gradient Penalty
CTGAN	Conditional Tabular GAN
WCGAN-GP	Wasserstein Conditional GAN with Gradient Penalty
TGAN	Tabular GAN
medGAN	Medical Generative Adversarial Network
GAIN	Generative Adversarial Imputation Networks
PNR-GAN	Passenger Name Records-GAN
PATE-GAN	Private Aggregation of Teacher Ensembles Generative Adversarial Network
ADS-GAN	Anonymized Data Synthesis GAN
CWGAN	Conditional Wasserstein GAN
MI-GAN	Multiple Imputation via Generative Adversarial Network
DATGAN	Directed Acyclic Tabular GAN
CasTGAN	Cascaded Generative Adversarial Network
SMOE-CTGAN	Smart Multi-Objective Evolutionary CTGAN
TSTR	Train on Synthetic, Test on Real
XGBoost	Extreme Gradient Boosting

REFERENCES:

[1] J. Eno and C.W. Thompson, "Generating Synthetic Data to Match Data Mining Patterns," *IEEE Internet Computing*, Vol. 12, No. 3, May 2008, pp. 78–82.

[2] A. Figueira and B. Vaz, "Survey on Synthetic Data Generation, Evaluation Methods and GANs," *Mathematics*, Vol. 10, No. 15, August 2022, p. 2733.

[3] A. Creswell, T. White, V. Dumoulin, K. Arulkumaran, B. Sengupta, and A.A. Bharath, "Generative Adversarial Networks: An Overview," *IEEE Signal Processing Magazine*, Vol. 35, No. 1, January 2018, pp. 53–65.

[4] J. Sivakumar, K. Ramamurthy, M. Radhakrishnan, and D. Won, "GenerativeMTD: A Deep Synthetic Data Generation Framework for Small Datasets," *Knowledge-Based Systems*, Vol. 280, November 2023, p. 110956.

[5] M. Goyal and Q.H. Mahmoud, "A Systematic Review of Synthetic Data Generation Techniques Using Generative AI," *Electronics*, Vol. 13, No. 17, 2024, p. 3509.

[6] A. Pathare, R. Mangrulkar, K. Suvarna, A. Parekh, G. Thakur, and A. Gawade, "Comparison of Tabular Synthetic Data Generation Techniques Using Propensity and Cluster Log Metric," *International Journal of Information Management Data Insights*, Vol. 3, No. 2, November 2023, p. 100177.

[7] *Artificial Intelligence: A Modern Approach*, [Online]. Available: https://thuvienso.hoasen.edu.vn/handle/123456789/8967.

[8] T.M. Mitchell, "Machine Learning," McGraw-Hill, New York, 1997.

[9] I. Joshi, M. Grimmer, C. Rathgeb, C. Busch, F. Bremond, and A. Dantcheva, "Synthetic Data in Human Analysis: A Survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 46, No. 7, July 2024, pp. 4957–4976.

[10] "Multi-Label Tabular Synthetic Data Generation for Bundle Recommendation Problem," *IEEE Conference Publication*, [Online]. Available: https://ieeexplore.ieee.org/document/10434763.

[11] E. Papadaki, A.G. Vrahatis, and S. Kotsiantis, "Exploring Innovative Approaches to Synthetic Tabular Data Generation," *Electronics*, Vol. 13, No. 10, 2024, p. 1965.

[12] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative Adversarial Nets," *Advances in Neural Information Processing Systems*, Vol. 27, 2014.

[13] Z. Zhao, A. Kunar, R. Birke, and L.Y. Chen, "CTAB-GAN: Effective Table Data Synthesizing," *Proceedings of the 13th Asian Conference on Machine Learning*, November 2021, pp. 97–112.

[14] S. Islam, M.T. Aziz, H.R. Nabil, J.R. Jim, M.F. Mridha, M.M. Kabir, N. Asai, and J. Shin, "Generative Adversarial Networks (GANs) in Medical Imaging: Advancements, Applications, and Challenges," *IEEE Access*, Vol. 12, 2024, pp. 35728–35753.

[15] L. Xu, M. Skoularidou, A. Cuesta-Infante, and K. Veeramachaneni, "Modeling Tabular Data Using Conditional GAN," *Advances in Neural Information Processing Systems*, Vol. 32, 2019.

[16] J. Yoon, Y. Zhang, J. Jordon, and M. van der Schaar, "VIME: Extending the Success of Self- and Semi-supervised Learning to Tabular Domain," *Advances in Neural Information Processing Systems*, Vol. 33, 2020, pp. 11033–11043.

[17] J. Fonseca and F. Bacao, "Tabular and Latent Space Synthetic Data Generation: A Literature Review," *Journal of Big Data*, Vol. 10, No. 1, July 2023, p. 115.

[18] M. Hernadez, G. Epelde, A. Alberdi, R. Cilla, and D. Rankin, "Synthetic Tabular Data Evaluation in the Health Domain Covering Resemblance, Utility, and Privacy Dimensions," *Methods of Information in Medicine*, Vol. 62, 2023, pp. e19–e38.

[19] J.D. Kelleher, B.M. Namee, and A. D'Arcy, "Fundamentals of Machine Learning for Predictive Data Analytics, Second Edition: Algorithms, Worked Examples, and Case Studies," MIT Press, October 2020.

[20] N. Hollmann, S. Müller, L. Purucker, A. Krishnakumar, M. Körfer, S.B. Hoo, R.T. Schirrmeister, and F. Hutter, "Accurate Predictions on Small Data with a Tabular Foundation Model," *Nature*, Vol. 637, No. 8045, January 2025, pp. 319–326.

[21] G.Y. Wong, F.H. Leung, and S.-H. Ling, "A Novel Evolutionary Preprocessing Method Based on Over-Sampling and Under-Sampling for Imbalanced Datasets," *IECON 2013 – 39th Annual Conference of the IEEE Industrial Electronics Society*, November 2013, pp. 2354–2359.

[22] C. Chai, J. Wang, Y. Luo, Z. Niu, and G. Li, "Data Management for Machine Learning: A Survey," *IEEE Transactions on Knowledge and Data Engineering*, Vol. 35, No. 5, May 2023, pp. 4646–4667.

[23] A. Alshantti, D. Varagnolo, A. Rasheed, A. Rahmati, and F. Westad, "CasTGAN: Cascaded Generative Adversarial Network for Realistic Tabular Data Synthesis," *IEEE Access*, Vol. 12, 2024, pp. 13213–13232.

[24] S. Popov, S. Morozov, and A. Babenko, "Neural Oblivious Decision Ensembles for Deep Learning on Tabular Data," *arXiv*, September 2019.

[25] R. Shwartz-Ziv and A. Armon, "Tabular Data: Deep Learning Is Not All You Need," *Information Fusion*, Vol. 81, May 2022, pp. 84–90.

[26] S. Ö. Arik and T. Pfister, "TabNet: Attentive Interpretable Tabular Learning," *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35, No. 8, May 2021, pp. 6679–6687.

[27] S. McKeever and M.S. Walia, "Synthesising Tabular Datasets Using Wasserstein Conditional GANS with Gradient Penalty (WCGAN-GP)," *AICS 2020*, 2020.

[28] V. Borisov, T. Leemann, K. Seßler, J. Haug, M. Pawelczyk, and G. Kasneci, "Deep Neural Networks and Tabular Data: A Survey," *IEEE Transactions on Neural Networks and Learning Systems*, Vol. 35, No. 6, June 2024, pp. 7499–7519.

[29] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, "Analyzing and Improving the Image Quality of StyleGAN," *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 8110–8119.

[30] J. Engelmann and S. Lessmann, "Conditional Wasserstein GAN-based Oversampling of Tabular Data for Imbalanced Learning," *Expert Systems with Applications*, Vol. 174, July 2021, p. 114582.

[31] F. Ferreira, N. Lourenço, B. Cabral, and J.P. Fernandes, "When Two Are Better Than One: Synthesizing Heavily Unbalanced Data," *IEEE Access*, Vol. 9, 2021, pp. 150459–150469.

[32] W. Shafqat and Y.-C. Byun, "A Hybrid GAN-Based Approach to Solve Imbalanced Data Problem in Recommendation Systems," *IEEE Access*, Vol. 10, 2022, pp. 11036–11047.

[33] Z. Zhao, A. Kunar, R. Birke, H. Van der Scheer, and L.Y. Chen, "CTAB-GAN+: Enhancing Tabular Data Synthesis," *Frontiers in Big Data*, Vol. 6, January 2024.

[34] Z. Dai, Z. Bu, and Q. Long, "Multiple Imputation via Generative Adversarial Network for High-dimensional Blockwise Missing Value Problems," December 2021.

[35] J. Yoon, J. Jordon, and M. Schaar, "GAIN: Missing Data Imputation Using Generative Adversarial Nets," *Proceedings of the 35th International Conference on Machine Learning*, July 2018, pp. 5689–5698.

[36] L. Xu and K. Veeramachaneni, "Synthesizing Tabular Data Using Generative Adversarial Networks," November 2018.

[37] G. Lederrey, T. Hillel, and M. Bierlaire, "DATGAN: Integrating Expert Knowledge into Deep Learning for Synthetic Tabular Data," March 2022.

[38] D. Anshelevich and G. Katz, "Synthetic Tabular Data Generation Using a VAE-GAN Architecture," *Knowledge-Based Systems*, Vol. 326, September 2025, p. 113997.

[39] H. Zhang, Y. Jing, F. Zhang, Z. Li, X.S. Wang, Z. Chen, and C. Lv, "TabTransGAN: A Hybrid Approach Integrating GAN and Transformer Architectures for Tabular Data Synthesis," *Information Processing & Management*, Vol. 62, No. 5, September 2025, p. 104220.

[40] Y. Zhang, N. Zaidi, J. Zhou, and G. Li, "Interpretable Tabular Data Generation," *Knowledge and Information Systems*, Vol. 65, No. 7, July 2023, pp. 2935–2963.

[41] E. Choi, S. Biswal, B. Malin, J. Duke, W.F. Stewart, and J. Sun, "Generating Multi-label Discrete Patient Records Using Generative Adversarial Networks," *Proceedings of the 2nd Machine Learning for Healthcare Conference*, November 2017, pp. 286–305.

[42] A. Mottini, A. Lheritier, and R. Acuna-Agost, "Airline Passenger Name Record Generation Using Generative Adversarial Networks," July 2018.

[43] M.G. Bellemare, I. Danihelka, W. Dabney, S. Mohamed, B. Lakshminarayanan, S. Hoyer, and R. Munos, "The Cramer Distance as a Solution to Biased Wasserstein Gradients," May 2017.

[44] N. Park, M. Mohammadi, K. Gorde, S. Jajodia, H. Park, and Y. Kim, "Data Synthesis Based on Generative Adversarial Networks," July 2018.

[45] J. Yoon, L.N. Drumright, and M. van der Schaar, "Anonymization Through Data Synthesis Using Generative Adversarial

Networks (ADS-GAN)," *IEEE Journal of Biomedical and Health Informatics*, Vol. 24, No. 8, August 2020, pp. 2378–2388.

[46] J. Jordon and J. Yoon, "PATE-GAN: Generating Synthetic Data with Differential Privacy Guarantees," 2019.

[47] N. Ran, B.I. Nasution, C. Little, R. Allmendinger, and M. Elliot, "Multi-Objective Evolutionary GAN for Tabular Data Synthesis," April 2024.

[48] "Credit Card Fraud Detection," [Online]. Available: https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud.

[49] R.K. Barry Becker, "Adult," 1996.

[50] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A.C. Courville, "Improved Training of Wasserstein GANs," *Advances in Neural Information Processing Systems*, Vol. 30, 2017.

[51] "XGBoost Documentation — XGBoost 2.1.1 Documentation," [Online]. Available: https://xgboost.readthedocs.io/en/stable/.