Moving Conditional GAN Close to Data: Synthetic Tabular Data Generation and its Experimental Evaluation

Journal:	*Transactions on Big Data*
Manuscript ID	TBD-2023-03-0133.R2
Manuscript Types:	Regular Paper
Keywords:	Synthetic data, Generative adversarial networks, Conditional generative adversarial networks, Privacy enhancing technology, I.2 Artificial Intelligence < I Computing Methodologies, K.4.1.f Privacy < K.4.1 Public Policy Issues < K.4 Computers and Society < K Computing Milieux, H.2.8.c Data and knowledge visualization < H.2.8 Database Applications < H.2 Database Management < H Information Technology and

Moving Conditional GAN Close to Data: Synthetic Tabular Data Generation and its Experimental Evaluation

Abdul Majeed, *Member, IEEE*, and Seong Oun Hwang, *Senior Member, IEEE*

Abstract—Recently, data has ousted oil as the most economical resource in the world, but most companies are reluctant to share customer/user data in pure form and on a large scale due to privacy concerns. Many innovative technologies (e.g., federated learning, split learning) are employed to meet the growing demand for privacy preservation. Despite these technologies, acquiring personal data in order to optimize utility, and then sharing it on a large scale, is still very challenging. Thanks to the rapid development of artificial intelligence (AI), a relatively new and promising solution to resolve these challenges is to generate synthetic data (SD) by mirroring the original dataset's properties. SD is a promising solution to address growing privacy demands as well as the utility/analytics requirements of many industry stakeholders. In this paper, we propose and implement an SD generation method from a real dataset containing both numerical and categorical attributes by using an improved conditional generative adversarial network (CGAN), and we quantify the feasibility of SD on technical and theoretical grounds. We provide a detailed analysis of SD in original and anonymized forms with the help of multiple use cases, whereas prior research simply assumed that privacy issues in SD are small because AI models do not overfit or SD has a poor connection with real data. We provide insights into the characteristics of SD (distributions, value frequencies, correlations, etc.) produced by the CGAN in relation to the real data. To the best of our knowledge, this is the pioneering work that provides an experiment-based analysis of the quality, privacy, and utility of SD in relation to a real benchmark dataset.

Index Terms—synthetic data, generative adversarial networks, privacy, utility, federated learning, privacy-enhancing technology.

1 Introduction

Synthetic data (SD) have become a mainstream privacy enhancing technology (PET) of modern times, that can offer reliable analytical results, just like real data [1]. SD can be produced with machine/deep learning methods or mathematical models by simulating the distributions and structure of real data. SD is a main pillar in health and well-being fields for validating hypotheses and creating new hypotheses related to biomedical research [2]. In the modern era, SD offers many ways to advance science and influence societies, particularly in the era of big data and AI [3]. We highlight the importance of SD in real-world applications in six different ways. (i) SD can be published widely, which may not be the case with real data owing to many privacy issues and legal measures (e.g., the GDPR in Europe) [4]. (ii) SD can enhance the results of AI models by increasing the size, diversity, and quality of training data. (iii) SD has abilities to solve the data island problem (a deficient result from AI models owing to greater disparities in distributions and sizes of data at each site) [5]. This problem was experienced by most hospitals amid the COVID-19 pandemic. (See Fig. 1.) (iv) SD can enable early access when the actual data cannot be retrieved due to unexpected circumstances (taking COVID-19 as an example), or there are too few [6]. (v) SD can be produced in multiple formats, such as electronic health records, medical images, biomedical signals, a time series, or activity data, which can be imperative for analytical tasks, research, or policy-making. (vi) It can become a mainstream PET (like FL) that does not access the real data but still allows analytics of distinct types.

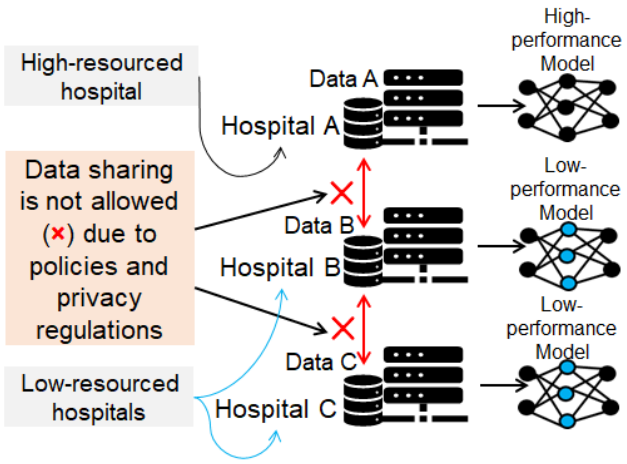


Fig. 1: Illustration of the data island problem (DIP).

1.1 Motivation and Novelty

Andrew Ng coined the term data-centric artificial intelligence (DC-AI) to develop high-quality and reliable AI systems [7]. DC-AI gives a higher preference to data quality rather than to the code of the AI models. Since the inception of the DC-AI concept, many breakthroughs have been achieved in multiple domains. Hedge achieved 100% accuracy in anomaly detection problems using the DC-AI [8]. Motamedi et al. [9] developed a DC-AI approach to

Abdul Majeed and Seong Oun Hwang are with the Department of Computer Engineering, Gachon University, Seongnam 13120, South Korea.
Manuscript received April 19, 2005; revised August 26, 2015.

training a deep neural network with as little data as possible without degrading accuracy much. Recently, DC-AI-based approaches have contributed significantly to solving longstanding industrial problems [10]. However, in some domains, good-quality data can be absent (or cannot be curated owing to small budgets). To compensate for a deficiency of good data, SD of good quality is vital and has become an integral component for data quality enhancement in AI applications. However, when the quality of real data is poor (incomplete records, skewed distributions, outliers, etc.), generating SD is challenging and can lead to many performance bottlenecks in the generative models. To the best of our knowledge, no framework deals with such problems (e.g., generating SD when the quality of real data is poor), which is the main motivation behind this research. Although many generative methods have been proposed that can create synthetic data from real data, certain challenges remain unaddressed: (i) accurate modeling of both numerical and categorical attributes, (ii) preventing model collapse during simultaneous training of two neural networks, (iii) quantifying the level of privacy versus utility from the SD, (iv) accurately reflecting in the SD all values of attributes from real data, regardless of status (i.e., minor, super-minor, major, and super-major), and (v) preventing imbalanced learning while generating SD.

Despite promising applications, research on SD generation is still in its infancy. Some studies only advocate the use of SD for research purposes [6]. Few studies have evaluated the credibility of SD generated with different AI tools [11]. Some studies have explored cases where SD can act as a replacement for real data in the healthcare sector [12] or in ranking applications [13]. Some studies described different ways to quantify the utility of SD [14], [15]. However, most studies have ignored privacy issues that can emerge from SD when an AI model overfits or when the similarity between the SD and the real data is high. Furthermore, deeper insights on utility in terms of query results, association rules, and value distributions have not been reported. Also, generating high-quality SD (i.e., data that preserves most characteristics of the original data, and that encompasses all attributes, values, and distributions to the fullest extent possible) in the presence of mixed-type attributes in the real data has not been thoroughly investigated. The main contributions of this research are as follows.

- We analyze the performance bottlenecks caused by poor-quality data in generative models, and we identify opportunities to develop a conditional generative adversarial network (CGAN)-based model for generating high-quality tabular SD to fulfill the growing need for privacy, as well as data utility/analytics.
- Our proposed method does not require data conversion from one form to another, and can correctly mirror the properties of the real data, regardless of the nature of the attributes (e.g., categorical or numerical) and the number of columns, whereas existing methods often require data conversion (numerical $\leftrightarrow$ discrete) and yield a higher offset from the real data.
- We introduce various new methods in the CGAN model to address major performance bottlenecks, such as data transformation issues, neural network model collapse, imbalanced learning, and reflecting all values and distributions in the synthetic data.
- We highlight the significance of SD in original and anonymized forms through experimental analysis that remains unexplored in the recent literature (most methods ignore the possibility of privacy breaches from SD based on the assumption that AI models do not overfit, or that similarities among SD and real data are not significantly high) [14].

Extensive experiments were conducted on a real-life benchmark dataset with the help of different use cases to verify the feasibility of the proposed method from both technical and theoretical points of view. The results and comparisons indicate better performance from the proposed method in SD generation, compared to existing SOTA methods.

The remainder of this paper is organized as follows. Section 2 presents related work, and Section 3 presents some preliminaries (e.g., data representations, the GAN architecture, the system model) and the problem formulation. Section 4 demonstrates the proposed method employed to generate high-quality data by leveraging real data. Section 5 evaluates the results obtained from experimentation on a real-world benchmark dataset. Section 6 summarizes the important findings of this research and discusses the implications. We wrap up the paper in Section 7.

2 Related work

SD has been increasingly used in modern times to fulfill deficiencies in good data, and many techniques have been developed to curate SD. Abay et al. [16] evaluated the utility of SD (T_s), obtained from real data (T), by using various techniques. The authors also developed a DL-based technique to generate T_s that can provide considerable privacy. Li et al. [17] employed a variational autoencoder (VAE) to produce T_s that can maintain an equilibrium between privacy and utility. The VAE proposed in their study outperformed the plain AE in terms of accuracy. However, this method cannot yield desirable performance when data are highly skewed (i.e., the distributions of values in some columns are not uniform). Wang et al. [18] developed a low-cost method named PART-GAN for time series data sharing and augmentation. PART-GAN has abilities to provide considerable privacy and can generate unlimited amounts of data. The authors evaluated PART-GAN on medical datasets for both privacy and utility. Martinsson et al. [19] developed a privacy-preserving method by adding new random information in data. By injecting random information, it offers robust privacy guarantees against strong adversaries. Torfi et al. [20] developed differential privacy combined with a convolutional AE (CAE) and convolutional GAN (CGAN) method to generate T_s. The CAE-CGAN can capture salient features and temporal interactions from T. Appenzeller et al. [21] evaluated three different SD generation methods, and explored various promising use cases of SD in real-life scenarios by amalgamating DP with the CGAN.

Recently, SD has contributed to the medical domain by helping doctors to devise new treatment modalities and to understand the dynamics and clinical properties of different diseases. Elaraby et al. [22] developed a CGAN-based data augmentation approach to enhance performance from

transfer learning in handwritten character recognition tasks. The proposed approach helped reduce the # of epochs and the generalization error when used with three distinct DL models (GoogLeNet, AlexNet, and VGG-16). Lu et al. [23] devised an improved version of the GAN for enhancing accuracy in anomaly detection by enhancing the sample quality as well as stability when training under realistic scenarios. Hammami et al. [24] developed a CycleGAN and YOLO-based technique to detect multiple organs in CT images, which enables robust detection of organs, and has many applications in medical imaging. Zhang et al. [25] developed the RFI-GAN model for generating medical images of good quality. The authors addressed the issues of variable scaling ratios and multiple sections in ultrasound images to prevent illogical image generation.

In recent years, many generative models have been proposed to curate tabular data to be used with ML models for classification and regression tasks [26]. MedGAN [27] is the first state-of-the-art (SOTA) generative model in this line of work. MedGAN can generate SD encompassing high-dimensional discrete variables for medical purposes. TableGAN [28] is another notable development for SD generation in tabular form. TableGAN uses an extra classifier in addition to D and G. It has higher complexity from using the classifier and an additional loss function. CTGAN [29] was devised to process tabular data without modifying the structure, and has yielded promising results in SD generation. However, this model cannot produce SD of good quality when the quality of real data is poor. CW-GAN [30] was proposed as an enhancement of TableGAN to generate SD of good quality and enhance the predictive performance of classifiers. Recently, GANBLR [31], which combines Logistic Regression and Naive Bayes with a GAN-based model, was developed to enhance the interpretation of data. However, the GANBLR model has clear limitations in that it only processes categorical columns. Table 1 highlights the status and comparisons of SOTA generative models. Our proposed method can address the limitations in the existing models and can produce SD in a designated format. Our method can be directly applied to any real-world T, and format conversions are not needed. In addition, it can be applied to poor-quality data and still yields high-quality SD. Lastly, we provide privacy analysis of T_s that remains unexplored.

In the absence of sufficient data, data augmentation techniques are mostly used to complement minor parts of the data by curating more samples. Fig. 2 presents the workflow of the proposed technique and the most commonly used data augmentation techniques. Random undersampling (RUS) downsamples the majority class whereas the minority class is upsampled in random oversampling (ROS) [32]. The synthetic minority oversampling technique (SMOTE) curates more samples for the minority class by computing the distance between the samples of the minority class and by using the k-nearest neighbor strategy [33]. The k-means SMOTE generates new points in each cluster to compensate for a deficiency of samples in the minority class [34]. Fair SMOTE was developed to address the imbalanced learning problem in classifiers by curating more samples for some skewed attributes and target classes [35]. Recently, a new augmentation technique that increases the size of all classes by adding synthetically generated samples from

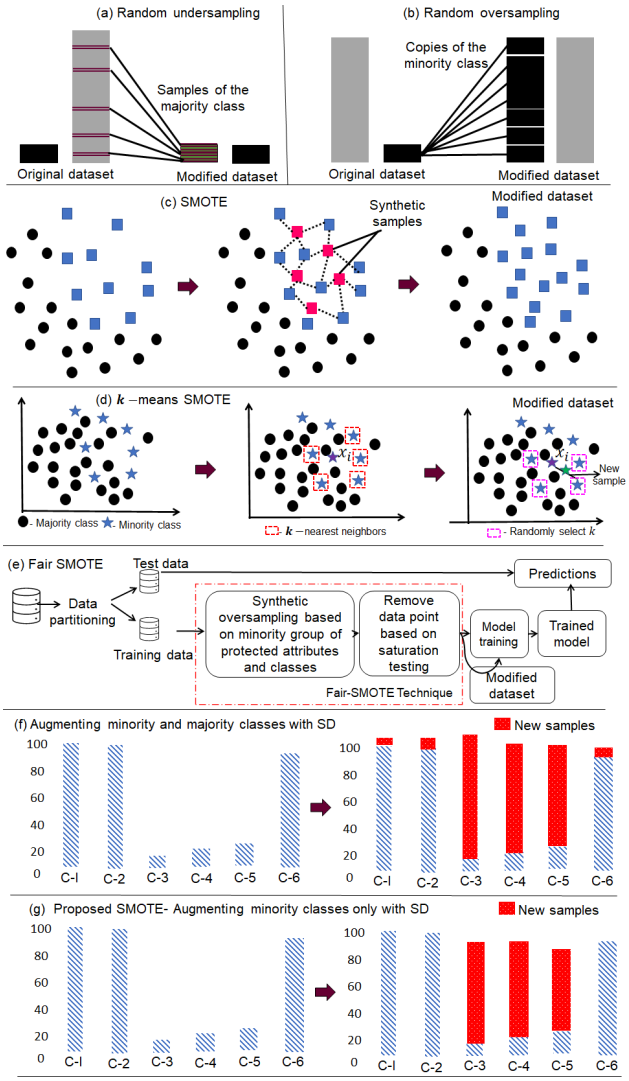


Fig. 2: Overview of the SOTA data augmentation techniques.

the GAN model was proposed [36]. That technique can balance the data, leading to better results than unbalanced data. In this paper, we devise a novel data augmentation technique that adds only required samples in the minority class, leading to better performance than existing SOTA techniques.

3 Preliminaries and Problem Formulation

In this section, we discuss tabular data representation and provide an overview of the GAN model. Later, we provide the system model considered in this work. Lastly, we formulate the problem to be solved with our proposed method.

3.1 Data representation

In this work, we focus on good-quality synthetic data generation to obtain T_s from real data T with the help of a powerful AI-based CGAN model. Let T be a real data table of dimensions $r \times c$, where c refers to the number of columns, and r denotes the number of rows. T consist of both N_c numerical columns, $\{c_1, c_2, \ldots, c_{N_c}\}$, and N_d categorical columns, $\{d_1, d_2, \ldots, d_{N_d}\}$. Each row in T contains full data

TABLE 1: Status and comparisons of SOTA generative models used to curate T_s.

Generative Model	Training method	Limitations
MedGAN [27]	GAN + Auto-encoder	Works well on discrete data only, and results are not generalizable.
TableGAN [28]	Classifier + GAN	No direct application, because data conversion (discrete → numerical) is needed.
CTGAN [29]	WGAN + Condition	Limited application when data size is small, and real data quality is poor.
CW-GAN [30]	Classifier+ WGAN + Condition	Higher offsets in numerical attribute values, and the training process is unstable.
GANBLR [31]	NB + LR	Good for discrete data only, and minor values are not properly reflected in the SD.

about a person, whereas a column contains attribute items (e.g., age/sex). A typical structure of T by using a sample of $20,000$ records is in Eq. 1:

$$T_{users,attributes} = \begin{pmatrix} u_i & a_1 & a_2 & a_3 \cdots & a_p \\ u_1 & v^1_{a_1} & v^1_{a_2} & v^1_{a_3} \cdots & v^1_{a_p} \\ u_2 & v^2_{a_1} & v^2_{a_2} & v^2_{a_3} \cdots & v^2_{a_p} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ u_N & v^N_{a_1} & v^N_{a_2} & v^N_{a_3} \cdots & v^N_{a_p} \end{pmatrix} = \begin{pmatrix} u_i & a_1 = age & a_2 = sex & a_3 = race & a_p = disease \\ 1 & 57 & F & White \cdots & Leukaemia \\ 2 & 34 & M & Black \cdots & Rhinitis \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 20,000 & 35 & F & Black \cdots & Cancer \end{pmatrix} \quad (1)$$

In Eq. 1, $a_1 \in N_c$, whereas $a_2, a_3, a_p \in N_d$. The position of N_c and N_d can be distinct in each real-world T. Each c in T refers to a random variable, and row r_k where $r_k = \{c_{1,k}, c_{2,k}, \ldots, c_{N_c,k}, d_{1,k}, d_{2,k}, \ldots, c_{N_d,k}\}, k \in \{1, 2, \ldots, n\}$ is one complete observation. N_c and N_d can be encoded into different formats when used in a CGAN. Furthermore, T is partitioned into the test set T_{test} and training set T_{train}.

3.2 The generative adversarial network

A generative adversarial network contains two neural network modules: generator G and discriminator D. Training of both these modules is performed in an adversarial manner. Random noise (a.k.a. latent code), denoted as z, serves as input to G in order to produce samples with the approximate distribution of T_{train}. In contrast, D takes $G(z)$ produced by G and T_{train} as input, optimizing it to distinguish $G(z)$ from T_{train} as shown in Eq. 2.

$$\begin{aligned} G &: z \rightarrow G \rightarrow G(z) \\ D &: (G(z), T_{train}) \rightarrow D \rightarrow F/R \end{aligned} \quad (2)$$

In Eq. 2, z is the latent code that is fed into the G, and $G(z)$ denotes the synthetic data produced by the G against z. $G(z)$ (synthetic data) and T_{train} (real data) are both fed into the D for distinguishing real samples from fake ones. F and R denote the fake and real samples, respectively.

During training, both G and D try to outperform each other. The objective function of the GAN is given in Eq. 3:

$$\begin{aligned} \min_G \max_D V(D,G) = E_{x \sim h_T(x)}[logD(x)] \\ + E_{z \sim h_z(z)}[log(1 - D(G(z)))] \end{aligned} \quad (3)$$

where h_T denotes the distributions of real data, and h_z is the distribution of latent code. During the update phase, the first item of the objective function forces D to yield higher scores with x, while the second item forces D to yield lower scores for z. Specifically, D wants to make the $G(z) \approx 0$, by doing so D can accurately distinguish between real and fake samples. In contrast, when updating G, the goal is to yield higher scores for D on z. With the help of joint training, T_s can be obtained for downstream tasks.

The two famous enhancements of the GAN are styleGAN and CGAN. The former is used for multimedia data, and the latter is used for tabular data. In this work, we use a CGAN to produce high-quality SD from T. The CGAN concept is the same as the original GAN, but condition y is attached to both modules. The y in CGAN carries additional information, such as labels for N_d, T modality, class labels, and/or network parameters. Due to the addition of y, the objective function of a CGAN will be slightly modified, as shown in Eq. 4:

$$\begin{aligned} \min_G \max_D V(D,G) = E_{x \sim h_T(x)}[logD(x|y)] \\ + E_{z \sim h_z(z)}[log(1 - D(G(z|y)))] \end{aligned} \quad (4)$$

where y denotes the condition in the CGAN. The CGAN converges when both G and D reach a Nash equilibrium.

3.3 The system model

In this paper, we consider a data-driven scenario/application with multiple actors (record owners, data owners, data analysts, and an adversary), as shown in Fig. 3. For analytical/data-mining purposes, analysts are interested in a fine-grained T from data owners who gather a huge amount of data from individuals. The data owners (hospitals, banks, insurance companies, etc.) are reluctant to share individual data in pure form due to privacy risks, even if analysts are honest. See "a" in Fig. 3. Another solution to this deadlock between data owners and analysts is to share data in an anonymized form. In this case, a copy of the data can still be obtained by an adversary who might be able to compromise the privacy of a user (or a set of users) by linking the data to auxiliary sources. See "b" in Fig. 3. This method cannot contribute to responsible data science, and data owners may hesitate to share anonymized data if, time and again, explicit privacy issues occur from the release of anonymized data. A relatively new solution to this problem is to mirror the properties of the original data and generate a copy similar to the real data by using AI methods ($AI\ algorithms \rightarrow data$) as illustrated with "c" in Fig. 3. By protecting privacy, we can explore opportunities to make use of AI-based methods to address the issues of data sharing on a wide scale. Our goal is to implement a practical method that can generate synthetic data of very good quality for analysts by learning the attributes and their distributions from the original data. We keep the design of the proposed method as flexible as possible to allow analysts to choose the amount of data they wish to generate to meet the growing demand for analytics (or data sharing) while respecting privacy issues. We assume the real dataset has already been collected from relevant users.

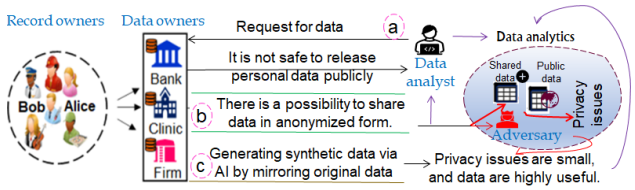


Fig. 3: Overview of system model considered in our study.

3.4 Problem formulation

Consider real-world dataset T where $T = r \times c$. In T, label c_i for the ith column can be either categorical or numerical, denoting a piece of information about the subjects. Let us say a_i is N_c, and another attribute, a_j, is N_d. The format of any row r is as follows:

$$r_x = \{a_i \in N_c, a_j \in N_d, a_k \in N_c, \ldots, a_p \in N_d\} \quad (5)$$

Considering the mixed data types in T, generating a mix of continuous and categorical columns in T_s is challenging. For example, one needs to use both $tanh$ and $softmax$ functions of a GAN on the output to convert $T \rightarrow T_s$. Secondly, numerical values in c can have different cardinalities, which cannot be denoted with $[-1, 1]$ in most cases, and if denoted can lead to the vanishing gradient problem in AI methods. Therefore, transformations are inevitable while using a numerical c of T, which can be tricky in most cases. Third, modeling all modes of continuous columns while transforming $T \rightarrow T_s$ is very challenging. Lastly, some columns in T can have very skewed distributions (e.g., one value can be 80%/90% of T). In such cases, minority classes can be overlooked by G while converting $T \rightarrow T_s$. Consider a T that contains numerical, categorical, and/or mixed attributes. It poses distinct challenges in its conversion to replicated form because of a large distribution of values, mixed-attribute types, multimodal distributions, and non-Gaussian-like distributions. Let T_s be the SD made from T for the downstream task. With T_s, the privacy breach issue can be reduced/resolved to a greater extent compared to T; ML models can be trained, and/or T_s can be distributed on a large scale. However, proving the efficacy of T_s in real-world decision-making scenarios may be subject to bias and inappropriate conclusions. The chosen problem to be formulated is: *how to convert T with mixed attributes (e.g., where categorical attributes can have skewed distributions, and where numerical attributes can have multimodal distributions) into T_s in an automated way by rigorously replicating joint distributions from T such that the conclusions or analysis results drawn from T_s are aligned with, or are approximately similar to, T while the structural similarity between T and T_s remains high. Furthermore, any legitimate information-consumers or data-analysts who see T and T_s should not be able to distinguish between the real data and the synthetic data without prior knowledge*. In most data-driven applications, analysts are mostly concerned with results (extracting pictures from the data). Hence, as long as T_s fulfills this requirement, it can be used as a proxy for T in most data-driven applications.

4 Synthetic Data Generation by Leveraging a Conditional GAN Model

In this section, we discuss how to generate a high-quality T_s by mirroring T using a CGAN (a powerful AI tool). The major steps in this process are data representation/transformation of numerical and categorical columns, designing a fully connected network structure, model training, loss function updating, and T_s generation by combining hot vectors and a ReLU. Fig. 4 presents a schematic of the proposed CGAN model used to generate SD of high quality. The key reason to use the CGAN model is to enable SD generation even when the real data have highly skewed distributions. In many real-world datasets, a single value in discrete columns might account for 80-90% of the data, with the rest having many diverse values. In these circumstances, traditional GAN models cannot generate SD of good quality, and likely face an imbalanced learning situation where only the majority value will be properly learned. The proposed model prevents such situations by imposing a condition on the values in discrete columns, and therefore, the quality of the SD is better. Furthermore, our model applies to any dataset, skewed or balanced. Table 2 shows the key notations in our T_s generation method.

TABLE 2: Main notations used in the proposed method.

Symbols	Description
T, T_s, x	Real data, synthetic data, sample of real data
N_c, N_d, c_i	Numerical columns, categorical columns, ith column
G, D, con	Generator, discriminator, condition of the CGAN
z, $T \rightarrow T_s$	Latent code or prior noise, T's conversion to T_s
$\text{gumbel}_\tau(x)$	Gumbel softmax on vector x with parameter τ
$x_1 \oplus x_2, \ldots$	Concatenation of vectors x_1 and x_2
$\text{FC}_{u \rightarrow v}(x)$	Linear transformation to get v from u
$\text{leaky}_\gamma(x)$	Leaky ReLU activation with ratio γ on x

From Fig. 4, there are three key parts in the proposed CGAN model. A concise description of each is given below.

I. *Column representation and transformation:* In this part, pre-processing is applied to the data, and numerical and categorical columns are converted to their respective representations. Later, the categorical columns are represented with hot vectors, whereas numerical attributes are transformed into modes and normalized values using the variational gaussian mixture (VGM) model. In addition, data are cleaned of vulnerabilities (e.g., missing values, outliers, inconsistent values) to prevent unnecessary processing and to generate a T_s that correctly mimics the properties of T. Technical details on this part are given in Section 4.1.

II. *Structure of the network:* In this part, the structure for both G and D of the network is established. Specifically, we use two fully connected networks with a PacGAN strategy in D to maintain stable training and prevent imbalanced learning. In addition, a ReLU and a Leaky ReLU are used as activation functions. Some other components, such as batch normalization, dropouts, data extractors (i.e., Gumbel, softmax), and tanh also contribute to the network structure. Technical details are given in Section 4.2.

III. *Training process and generation of a T_s using T:* Here, a condition is established concerning discrete columns that give useful hints to G and D that prevent imbalanced learning. The two models are trained in parallel by respecting

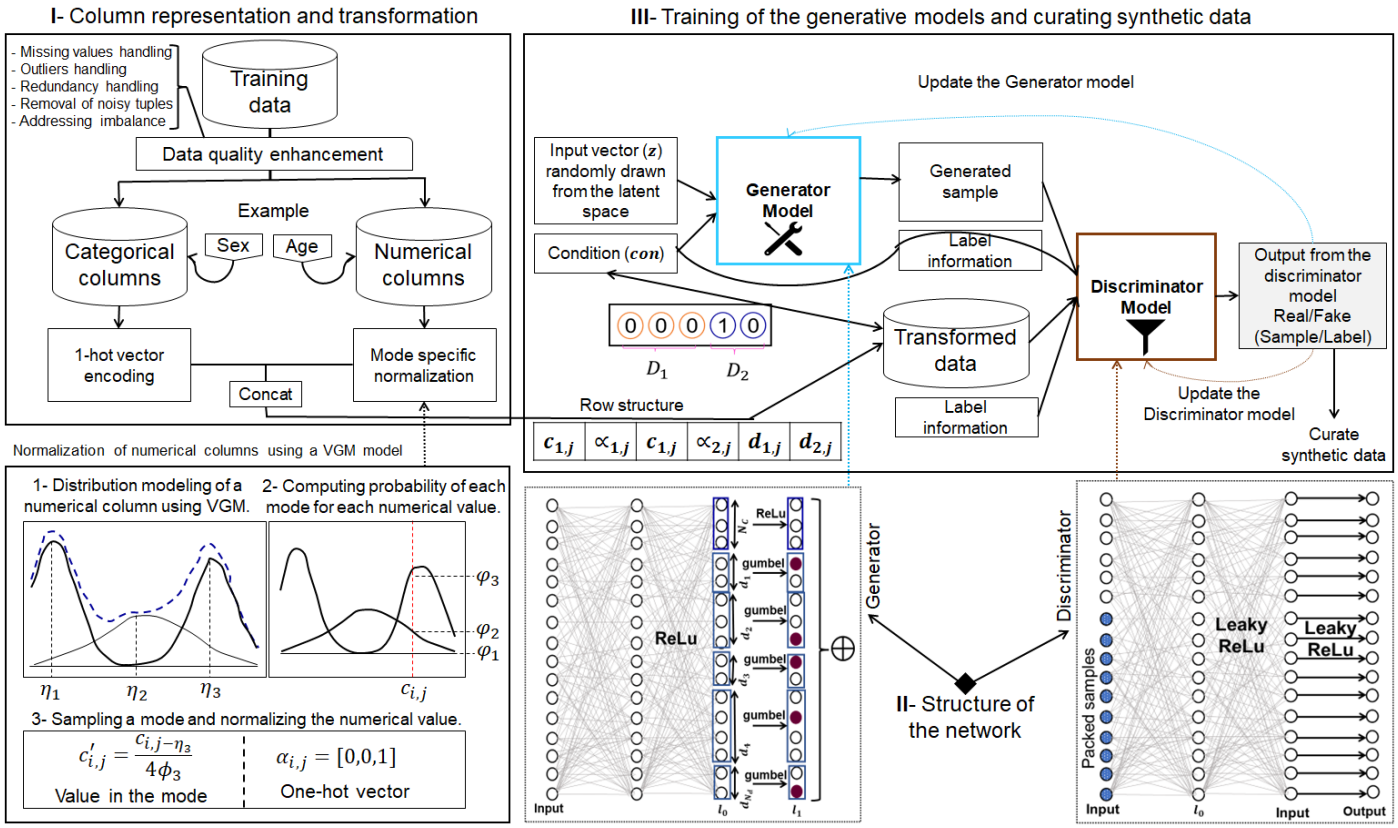


Fig. 4: Schematic of the proposed CGAN model for generating high-quality synthetic data.

the condition established. After training, a T_s that closely represents T is obtained, which is used for downstream tasks. Technical details of this part are in Section 4.3.

4.1 Column representation and transformation

In this section, we first discuss the modeling and transformation of the columns. Subsequently, we discuss the mechanism used to establish the condition that improves the learning abilities of the generator.

4.1.1 *Numerical column modeling and transformation*

The structure of any row, r_x, in T can be expressed as

$$T : r_x = \{a_i \in N_c, a_j \in N_d, a_k \in N_c, \ldots, a_p \in N_d\} \quad (6)$$

As shown in Eq. 6, any column can be either N_c or N_d. Both types pose distinct challenges for representation in GAN environments. Previous research used one of several methods (e.g., min-max normalization, quantile transformation, normalization) in order to transform and represent N_c. In this work, we use mode-specific normalization to address multimodal distribution issues in the CGAN. In contrast, we transform and represent N_d with hot vectors to effectively resolve imbalance issues. As stated earlier, N_c cannot be bounded by [-1,1], and distributions can be highly complicated. To address this challenge, we used a VGM model [37] to transform N_c and accurately represent their values. The three-step process used to model each numerical column is shown in the bottom left portion of Fig. 4. A concise description of each step is given below.

- *Distribution modeling of a column*: Given numerical column C_i, the VGM model fits a Gaussian mixture and estimates the number of modes, m_i, in C_i. As shown in Part 1 in the bottom half of Fig. 4's Section I, the VGM finds three modes denoted with η_i, where $i = 1, 2, 3$. The learned GM is given in Eq. 7:

$$P_{C_i}(c_{i,j}) = \sum_{l=1}^{3} \mu_l N(c_{i,j}; \eta_l, \phi_l) \quad (7)$$

  where ϕ_l denotes the standard deviation of a mode, and μ_l denotes the weight of the respective mode.
- *Computing probabilities of the modes*: After finding the modes in C_i, the probability of each value (say, $c_{i,j} \in C_i$) concerning each mode is computed. As shown in Part 2 in the bottom half of Fig. 4 Part I, three probability densities are φ_1, φ_2, and φ_3. Each φ is computed using Eq. 8:

$$\varphi_l = \mu_l N(c_{i,j}; \eta_l, \phi_l) \quad (8)$$

- *Sampling a mode and normalizing the value*: Pick one mode from the given φ_l values, and normalize value $c_{i,j}$. As shown in Part 3 in the bottom half of Fig. 4 Part I, if φ_3 is chosen as the sample mode, its value can be expressed in the form of a one-hot vector, $\alpha_{i,j} = [0, 0, 1]$. The scaler value, $c'_{i,j}$, in the normalized form in m_3 can be obtained using the formula in Part 3 in the bottom half of Fig. 4 Part I. The output of the three-step process is given in Eq. 9:

$$C_i : c_{i,j} = c_{i,j}^{'} \oplus \alpha_{i,j} \tag{9}$$

where $c_{i,j}^{'}$ is the normalized value in a mode, and $\alpha_{i,j}$ is the hot vector of the mode.

Although mode-specific normalization proposed in [29] yielded 25.70% better performance than the $min-max$ normalization technique, the resulting T_s had two key problems: (i) some values in the numerical columns were negative, even if the entire column in real data had all positive values, and (ii) the offset in some numerical values was significantly higher than in the real data. Furthermore, if the data size is very small, many values in the numerical columns of SD tend to lie outside the desirable range, leading to a higher loss of statistical information. In this paper, we address the above-cited limitations in the CTGAN model by improving the mode-specific normalization process, as well as choosing the optimal values for cluster size and weight threshold. Also, the decision process regarding the distribution selection and value representation is bounded by a new valid-component indicator in the numerical columns transformer to better approximate the real data. The final representation of row r after transformation of numerical columns is expressed in Eq. 10:

$$r_e = c_{1,j}^{'} \oplus \alpha_{1,j} \oplus c_{2,j}^{'} \oplus \alpha_{2,j}, \ldots, c_{N_c,j}^{'} \oplus \alpha_{N_c,j} \oplus d_{1,j} \oplus d_{2,j} \ldots d_{N_d,j} \tag{10}$$

where $d_{i,j}$ is the hot vector form of a categorical value.

4.1.2 Solution to class imbalance in categorical columns

Setting up a suitable condition in N_d can assist G to address class-imbalance problems to generate a T_s that is very close to T. Without conditions, G can ignore minor categories in N_d, and the obtained T_s offers poor utility. Let v^* be the value from the k^*th column and row r_i in a categorical column (i.e., D_{k^*}) to be matched to a new generated sample denoted r_s. In this case, G can be perceived as a conditional distribution of r_i in the k^*th column, which can be formally expressed in Eq. 11:

$$r_s \sim P_G(r_i | D_{k^*=v^*}) = P(r_i | D_{k^*=v^*}) \tag{11}$$

The integration of such a condition forces G to produce high-quality data. By enforcing the condition, original distributions can be reconstructed as formalized in Eq. 12:

$$P(r_i) \sum_{v \in D_{k^*}} P_G(r_i | D_{k^*=v^*}) P(D_{k^*=v}) \tag{12}$$

The conditions (denoted with con) are incorporated in the CGAN implementation via hot vectors. As shown in Eq. 10, all N_d are transformed as hot vectors. Therefore, a conditional vector con is created for each categorical column. Thus, the overall transformation of categorical columns $D_1 \in T$ can be expressed as $(D_1 \rightarrow d_1 \rightarrow [d_1^{(v)}]$, where $v = 1, \ldots, |D_1|)$. For any kth categorical column, $con_k = [con_k^{(v)}]$ can be the masked vectors encompassing conditions for a particular value. Therefore, the condition can be formalized in Eq. 13:

$$con_k^{(v)} = \begin{cases} 1, & \text{if } k = k^* \text{ and } v = v^* \\ 0, & \text{otherwise} \end{cases} \tag{13}$$

Finally, conditional vectors, con, can be made: $con = con_1 \oplus con_2 \ldots con_{N_d}$. For example, for any two categorical columns, $D_1, D_2 \in$T), if we let $D_1 = race = \{white, black, others\}$ and $D_2 = income = \{\geq 50K, < 50K\}$, then the condition on D_2 for the first value can be reflected in mask vectors as $con_1 = [0,0,0]$ and $con_2 = [1,0]$, and overall con can be expressed as $con = [0,0,0,1,0]$.

During training, the conditional G can produce one-hot vectors for categorical columns each time. We denote the newly produced columns with $d_i^{'}$, where $i = 1, \ldots, N_d$. Conditions made in the form $(D_{k^*=v^*})$ and encoded in a con vector are enforced during the sample generation for any column $d_i^{'}$. Furthermore, losses are penalized by incorporating the cross-entropy between con_k^* and $d_{k*}^{'}$. After certain epochs, G learns the condition well and correctly reflects conditions in $d_i^{'}$. A critic is used to assess output quality Q_G of conditional G, which takes the difference between conditional distributions in T as well as the learned conditional distributions, which is mathematically expressed in Eq. 14:

$$Q_G = dist(P_G(r_s|con), P(r|con)) \tag{14}$$

where r_s and r are the learned row and the row of T, respectively.

Accurate representation of con and the training data is imperative to help the critic accurately compute the distance between T and T_s. In Fig. 5, we present an overview of the implementation of a conditional G used to explore all possible categorical values in C_i. The whole process can be implemented in six steps. (i) Create $|N_d|$ mask vectors filled with zeros initially, $m_k = m_k^{(v)}$, $k = 1, 2, \ldots, |D_k|$. (ii) Select categorical columns with equal probability (for example, if D_2 is selected, then $k = 2$). (iii) Compute the PMF of each value in a selected column; for example, in the income column, if $v^* = \geq 50K$ and it occurs 9000 times in total, then $PMF_{v^*} = log(9000)$. (iv) Let v^* be a randomly chosen value as per the PMF given in Step (iii), and if it is the first value in D_2, as shown in Fig. 5, then $v^* = 1$. (v) Update the respective components of a mask to 1 under the condition $m_k^{(v^*)} = m_k^{(\geq 50K)} = 1$; and (vi) Calculate the con vector based on the process explained above; for example, $con = [0,0,0,1,0]$, is the condition imposed on the first value of D_2. By using the six-step process, all categorical attributes can be evenly sampled to assist in producing a T_s with representations of all values.

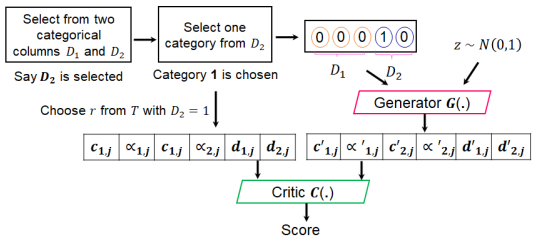


Fig. 5: Overview of a conditional G that explores all possible categorical values in columns for accurate data modeling.

4.2 Structure of the network

In order to generate T_s from T using a CGAN, we used fully connected networks (with two fully connected hid-

den layers) in both G and critic C (from here on, C is an alternate term for discriminator D). In G, the ReLU activation function and batch normalization are employed. The true representation of r_s is obtained from two hidden layers via a mix function (i.e., an activation function). Scaler numerical values, $c_i^{'}$, are obtained using the $tanh$ function. In contrast, α_i (the mode indicator) and categorical values $d_i^{'}$ are produced via $gumbel\ softmax$. In C, dropout and Leaky ReLU are used in both hidden layers. The overall structure of conditional G with z and con can be formally expressed in Eq. 15:

$$\begin{cases} l_0 = z \oplus con \\ l_1 = l_0 \oplus Relu(BN(FC_{|con|+|z|\rightarrow 256}(l_0))) \\ l_2 = l_1 \oplus Relu(BN(FC_{|con|+|z|+256\rightarrow 256}(l_1))) \\ c_i^{'} = tanh(FC_{|con|+|z|+512\rightarrow 1}(l_2)) & 1 \leq i \leq N_c \\ \alpha_i = gumbel_{0.20}(FC_{|con|+|z|+512\rightarrow m_i}(l_2)) & 1 \leq i \leq N_c \\ d_i^{'} = gumbel_{0.20}(FC_{|con|+|z|+512\rightarrow |D_i|}(l_2)) & 1 \leq i \leq N_d \end{cases} \tag{15}$$

In C, we employed PacGAN [38] to avoid model collapse with 10 samples. The overall architecture of C with a pac-size of 10 is given in Eq. 16:

$$\begin{cases} l_0 = r_1 \oplus r_2 \ldots r_{10} \oplus con_1 \oplus con_2 \oplus \cdots \oplus con_{10} \\ l_1 = drop(leaky_{(0.20)}(FC_{10|r|+10|con|\rightarrow 256}(l_0))) \\ l_2 = drop(leaky_{(0.20)}(FC_{256\rightarrow 256}(l_1))) \\ C(.) = FC_{256\rightarrow 1}(l_2) \end{cases} \tag{16}$$

To prevent the overfitting and memorization issues, we used optimized values for most parameters, i.e., the Adam optimizer with a 2×10^{-4} learning rate, the α value in the Leaky ReLU set to 0.2, βs=(0.5, 0.9), and a Dropout (0.5) where the ideal dropout value is between 0.4 and 0.7. With the help of these hyperparameters and their optimized values, the training process was stable, and overfitting issues did not occur. Lastly, due to the optimized network structure, the time overheads were small, and memory runout issues also didn't occur.

4.3 Training process and generation of a T_s using T

In this subsection, the training process is discussed with the loss updating mechanism for G and C. We performed the training using Wasserstein GAN (WGAN) loss with a gradient penalty [40]. We employed the Adam optimizer at the optimal learning rate (i.e., 2×10^{-4}) to produce a high-quality T_s. During training, the loss weights are updated until the convergence of the conditional GAN model. The generic form of C is Eq. 17:

$$L_C = E_{x' \sim P_{T_s}}[C(x^{'})] - E_{x \sim P_T}[C(x)] + \lambda E_{\bar{x} \sim P_{\bar{x}}}[(||\nabla_{\bar{x}} C(\bar{x})||_2 - 1)^2] \tag{17}$$

where P_{T_s} and P_T are the synthetically generated and the original data distributions, respectively. The third term denotes the gradient penalty, and λ is the co-efficient of the gradient penalty. $P_{\bar{x}}$ is the sampling implicitly defined along a straight line between distributions P_T and P_{T_s}. The generic loss function of G during training is expressed as

$$L_G = E_{x' \sim P_{T_s}}[C(x^{'})] \tag{18}$$

The complete procedure applied to produce T_s from T using the conditional GAN is formally expressed in Algorithm 1. In this algorithm, we describe the main tasks performed to transform data from one form to another. Besides the pseudocode in Algorithm 1, we specify the URL of the real data, the labels of categorical columns, the number of samples a data analyst wants to generate, and the epoch strength from the interface program. The initial learning and decay rates, the dimensions of C and G, and the frequency values are specified in the core Python functions. After thorough implementation and compilation of the program, different versions of T_s with varying tuples are obtained for quality, privacy, and utility evaluation.

Algorithm 1 Training the CGAN to convert T into T_s.

Require: T (Original data (a.k.a. training data)), m (batch size), pac (pac size), Φ_G and Φ_C (parameters of conditional G and critic C), and $|T_s|$ (# of tuples to be curated).
Ensure: Updated parameters Φ_C and Φ_G and T_s
1: T quality enhancement to reduce redundant operations
2: Generate masks for d columns,$\{m_1.m_2, \ldots, m_{N_d}\}_j$
3: $con \leftarrow con_j$, for $1 \leq j \leq m$
4: Sample $z_j \sim$ MVN (0, I), for $1 \leq j \leq m$
5: $r_j^{'} \leftarrow G(z_j, con_j)$, for $1 \leq j \leq m$ ▷ Yield synthetic data
6: Sample $r_j \sim Uniform(T|con_j)$, for $1 \leq j \leq m$
7: $con_v^{(pac)} \leftarrow con_{v \times pac+1} \oplus \cdots \oplus con_{v \times pac+pac}$
8: $r_v^{'(pac)} \leftarrow r_{v \times pac+1}^{'} \oplus \cdots \oplus r_{v \times pac+pac}^{'}$
9: $r_v^{(pac)} \leftarrow r_{v \times pac+1} \oplus \cdots \oplus r_{v \times pac+pac}$
10: $L_C \leftarrow \frac{1}{m/pac} \sum_{v=1}^{m/pac} C(r_v^{'(pac)}, con_{(v)}^{pac}) - \frac{1}{m/pac} \sum_{v=1}^{m/pac} C(r_v^{(pac)}, con_{(v)}^{pac})$
11: Sample φ_1 , φ_2, $\ldots$, $\varphi_{m/pac} \sim$ Uniform (0,1)
12: $\bar{r}_v^{(pac)} \leftarrow \varphi_v r_v^{'(pac)} + (1 - \varphi_v) r_v^{(pac)}$
13: $L_{GP} \leftarrow \frac{1}{m/pac} \sum_{v=1}^{m/pac} ((||\nabla_{\bar{r}_v^{(pac)}} C(\bar{r}_v^{(pac)}, con_v^{(pac)})||_2 - 1)^2)$
14: $\Phi_C \leftarrow \Phi_C - 0.0002 \times adam(\nabla_{\Phi_C}(L_C + 10 L_{GP}))$
15: Generate $r_j^{'}$ again using lines 4–10.
16: $L_G \leftarrow -\frac{1}{m/pac} \sum_{v=1}^{m/pac} C(\bar{r}_v^{(pac)}, con_v^{(pac)}) + \frac{1}{m} \sum_{j=1}^{m} CE(d_i^{'}, m_i, j)$
17: $\Phi_G \leftarrow \Phi_G - 0.0002 \times adam(\nabla_{\Phi_G} L_G)$
18: Update the C and G parameters in a feedforward way.
19: Return Φ_G, Φ_C, T_s ▷ Required outputs

4.4 Analysis of various methods used in CGAN model

In the proposed SD generation model, various methods have been used to obtain good-quality SD. The important methods used in the model that have a greater impact on performance are summarized below.

1) The *conditional vector* in the CGAN model is necessary to guide G regarding class labels or values of the particular attributes to be generated. In this work, the conditional vector is used to guide G on labels for discrete columns and to generate synthetic tuples that are conditioned on one of the columns of a discrete type. By imposing the *conditional vector*, all values are evenly sampled from the real data.
2) The *VGM model* is used to address multimode issues of the N_c. For example, there can be multiple types of series in a numerical column, and to ensure balanced learning, they need to be represented in a consistent format. By multiple types of series, we mean a group of values (e.g., series) that can differ

from another group/series based on the # of digits (e.g., single-digit versus two/more digits) as well as the gap (small versus large) in values. To this end, VGM first computes the modes of the N_c. Later, standard deviation and mean values for each of the modes are determined. Afterward, the values of the numerical column are normalized with the associated standard deviation and mean. The resulting normalized value is then concatenated with hot vectors of the categorical columns to prepare input for the CGAN model. Specifically, it assists in modeling non-Gaussian distributions in N_c.

3) The *PacGAN strategy* is used to prevent mode collapse during training of the networks. Most GAN models are vulnerable to mode collapse when working with attributes having skewed and multimodal distributions, and as a result, T_s can have a lot less diversity. To fix this issue, PacGAN modifies the discriminator structure to make final decisions with the help of multiple samples drawn from the same class, either artificially generated or real. This paper makes use of PacGAN to address the issue of mode collapse during training.
4) *WGAN* loss is used to maintain stability during training and to ensure the condition of the Lipschitz constraint. Specifically, in this strategy, gradients are penalized at sample points to adhere to the Lipschitz constraint (e.g., the norm of the gradients is close to 1 everywhere) [40], [41]. This strategy has been widely used in the GAN model because it is superior to the weight-clipping technique.

All the above-cited methods are necessary to correctly transform data from one form to another when T has mixed attributes. The *VGM model* is no longer required if T contains all categorical columns. Similarly, the *conditional vector* can be omitted when all data are numeric. However, *WGAN* loss and *PacGAN* are more important than the others, and have a greater effect on data improvement and model stability.

5 Experimental evaluation

In this section, we discuss and analyze the output of the CGAN with the help of multiple use cases. To prove the feasibility of the proposed concept, we conducted experiments with the Adult dataset [42], which is a benchmark for evaluating most privacy algorithms. In the Adult dataset, there are 32,561 records and 15 distinct attributes (final dimensions of $32,561 \times 15$), with values of N_c and N_d at 6 and 9, respectively. The sensitive information is income, and all other attributes are treated as non-sensitive. Table 3 presents concise details of the dataset used in the experimental evaluation. In Table 3, N and C refer to numerical and categorical, respectively.

The reason to use this dataset is that it contains a reasonable mix of both numerical and categorical attributes. Furthermore, we emulated real-world cases by accessing the dataset through a URL from data owner environments. Since the adult dataset is a benchmark in the privacy community, we used it to test the efficacy of our method. Furthermore, this dataset has skewed distributions for some attributes and is a good candidate to verify the significance of the proposed method. Furthermore, our method is highly flexible, meaning it can produce T_s from any real-world T.

TABLE 3: Details of dataset used in performance evaluation.

Attribute type	Attribute label(Type, Cardinality)
QIDs	Hours-per-week(N , 93), Capital gain (N , 91) Capital loss (N, 118), fnlwgt (N, 21, 648) Age (N, 74), Education no. (N, 16) Country/Citizenship (C, 41), Race (C, 5) Occupation (C, 14), Work class (C, 8) Education (C , 16), Relationship(C , 7) Marital status(C , 6), Gender (C , 2)
SA	Income(C, 2)

5.1 Experimental settings

The implementation of the CGAN model was done on a notebook having Intel Core CPU i5-3320M @2.60GHZ. The RAM size was 8GB. We implement our model using Python 3.9 (64-bit version) with the help of built-in libraries (torch, scikit-learn, torchvision, numpy, pandas, matplotlib, etc.). The relevant functions from each library were used to accomplish the task of data generation. For example, BatchNorm1d, LeakyReLU, Dropout, Module, ReLU, Sequential, etc. were leveraged from the torch.nn library. Similarly, the relevant functions for giving suitable structure to data were used in the form of separate Python scripts. The interface program has various inputs such as real data directory links, labels of columns that are non-numeric, the number of epochs to be used in training, tentative data templates, the number of synthetic records to be curated, and file designation to store SD. The interface program allows users to specify the number of records to be curated, making our implementation more flexible compared to existing generative models. The SD curated with our model can be either saved to a file for downstream tasks or sent to the ML pipelines for training/testing models. In the implementation setup, we first test the model under the default setting of different parameters (e.g., G and D learning rates, decay, dimensions, batch_size, etc.) and then optimize them to improve T_s quality. The activation functions used in the model were tanh, mix, and softmax. We generate T_s in a similar structure to T, and the order of attributes is also the same as T.

The evaluation of T_s was performed from five different perspectives: enhancing ML models performance, query accuracy (involving numeric, discrete, and hybrid attributes), pattern/rule mining, privacy and utility analysis, and general benefits. While comparing the performance of our model from the perspective of ML, we used accuracy (Eq. 19) as the main evaluation metric and compared the results with five SOTA generative models (MedGAN [27], TableGAN [28], CTGAN [29], CW-GAN [30], and GANBLR [31]) and six SOTA augmentation techniques (RUS [32], ROS [32], SMOTE [33], k-means-SMOTE [34], Fair-SMOTE [35], and CTGANSamp [36]). Furthermore, we also trained three different ML classifiers (DT, SVM, RF) to compare accuracy between T_s and T. We analyzed and discussed the impact of T_s on samples and confusion matrices. While comparing the performance of our model from the perspective of query accuracy, we created sixteen different aggregation/count queries by utilizing the information from T and analyzed the performance of query answers from T_s by using $Supp$

metric (Eq. 21). Afterward, we compared the results with T to check the performance of T_s. Specifically, we used T as the benchmark to evaluate and compare the performance of T_s curated with our model. To perform privacy and utility analysis, the anonymized versions of T_s were created with k-anonymity model with different k, and privacy and utility results were compared with relevant baselines and T. To compute privacy leakage, we executed two well-known attacks on anonymized data created from T_s and T, and computed the D_r (Eq. 22). We compared the privacy leakage (i.e., D_r) from the anonymized version of T_s with T and a baseline method [15]. To perform utility analysis, we computed the IL with the help of DM (Eq. 23) metric and compared the results with T. While analyzing the general benefits of T_s, we analyzed the effectiveness of T_s on five different grounds as discussed in detail in Section 5.7.

5.2 Ablation study

Although the proposed CGAN model accomplished the task of T_s generation from T, and proposed implementations are highly optimized, some parameters can degrade its performance in real scenarios. For example, if the condition vector is removed from the CGAN, there is a decrease of about 20.09% in the learning ability of G, and some minor distributions were not learned properly. Therefore, the CGAN can no longer apply to T having skewed distributions. Similarly, if the mode-specific normalization module is replaced with $min-max$ normalization, there is a drop of about 27.10% in the quality of T_s, and the vanishing gradient problem occurs while training neural networks. We removed the PacGAN component and tested the performance of the CGAN model, and mode collapse occurred during training with a 0.26 probability in 10 different tests. When we removed both PacGAN and the conditional vector, there was a 40% drop in performance, and the CGAN was overfitting. We used the data without pre-processing, and there was a 6% drop in T_s quality. Furthermore, when we set loose values for the training hyperparameters (learning rate, α, β, dropout, etc.), the performance dropped by 39.89%. We set the value for the coefficient of the gradient penalty (λ) to 10, which is a feasible value for generating a good-quality T_s [40]. However, the different values of λ can have a different effect on the accuracy of the T_s. To validate the effects of λ on the accuracy of the T_s, we conducted experiments by varying the values of λ. The best accuracy was obtained with the RF model on T_s at 83.78% when λ=10. When λ=2,4,6,8, there was a drop in accuracy by up to 15.66%, 12.04%, 8.41%, and 6.02%, respectively. When λ=12, there was no change in accuracy, and the original value was sustained (e.g., accuracy= 83.78%). This analysis verifies the feasible value of λ for generating a T_s of good quality. With the optimal values of hyperparameters, and using all supportive modules, our CGAN model yielded more competitive performance than previous SOTA generative methods. Next, we assess the performance of T_s curated with the CGAN model from five different perspectives.

5.3 Analyzing the use of T_s from the ML perspective

The performance of any ML model depends highly upon the quality of the data. The better the quality, the better the performance of any ML model, and vice versa. To evaluate the utility of T_s from ML perspectives, we computed and compared accuracy values (using Eq. 19) with T and relevant baselines.

$$A = \frac{T_p + T_n}{|T|} \tag{19}$$

In Eq. 19, T_p is true positive, and T_n is true negative. $|T|$ shows the total records in the data. In the denominator, $|T|$ should be replace with $|T_s|$ while calculating A for T_s.

5.3.1 Comparison with existing SOTA augmentation algorithms and generative models

The T_s produced by generative models can be used in training ML classifiers as well as for statistical analysis. To verify the quality of T_s produced with our model, we fused T_s with T and verified the efficacy of augmented data in terms of classifier accuracy enhancement and sampling quality while training them. As discussed earlier, we used T_s to augment only the problematic portion of T, whereas existing data augmentation techniques simply double the data, leading to marginal improvements in results because the distributions of majority and minority classes are not balanced. To compute and compare accuracy, we used a random forest classifier, which is SOTA, and a popular ML classifier. We divided the data for training (2/3) and testing (1/3). Table 4 presents comparisons of accuracy between our method and existing SOTA data augmentation algorithms.

TABLE 4: Average accuracy: our method versus existing SOTA augmentation algorithms.

Data augmentation algorithm	% accuracy (avg. of 10 tests)
Default (without augmentation)	85.38
RUS algorithm	76.15
ROS algorithm	75.38
SMOTE algorithm	70.89
k-means-SMOTE algorithm	74.23
Fair-SMOTE algorithm	85.96
CTGANSamp algorithm	90.25
Our method	**100.00**

From the results, we can see that our method outperformed all algorithms in terms of accuracy. It is worth noting that these improvements came from adding more records in the minority class only (e.g., income is $>$ 50K). In addition, our augmentation method yielded superior accuracy using fewer trees. The main reasons for the perfect prediction accuracy are the alteration in the importance score of each feature and correlation enhancement with the target class. When data is imbalanced, the classifier does not explore the features of the minority class well and misclassifies them as the majority class. In contrast, when classifiers are trained on balanced/augmented data, some salient but difficult-to-learn features can easily be perceived in the training process, leading to better prediction accuracy [43]. Also, the main focus of data augmentation in ML is to improve the quality, diversity, and equity of data, which can enhance accuracy. Lastly, data augmentation increases the prediction accuracy by increasing the counts of examples which are common in feature space across classes but different in label.

Furthermore, the confusion matrices yielded by our method were more balanced compared to the existing SOTA augmentation algorithms, as shown in Fig. 6.

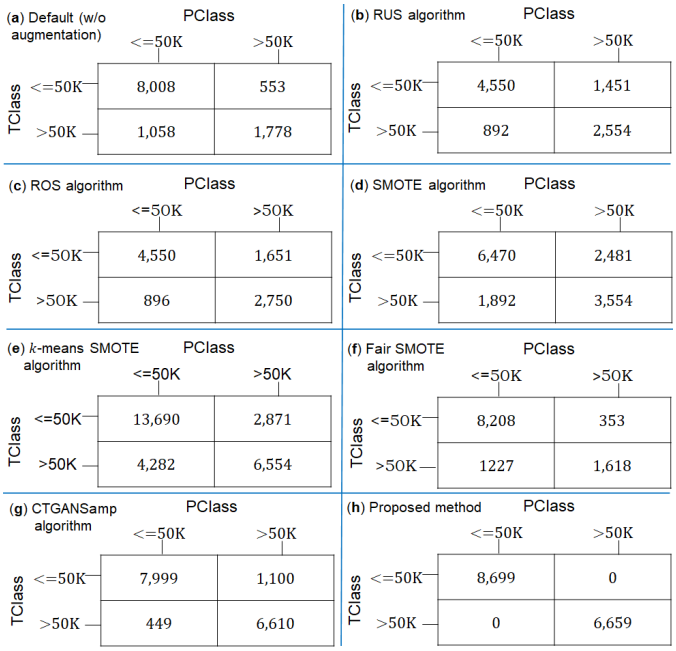


Fig. 6: Comparison of confusion matrices: proposed method vs. existing SOTA data augmentation algorithms.

In the next experiments, we compared the performance of our method in terms of sampling quality. Specifically, we drew samples from the augmented data and compared the differences between the frequency of the target class's values (e.g., ≤ 50K and > 50K). Since our method adds new records only to the minority class, the sampling quality of the generated data was much better, compared to the other data augmentation algorithms. Fig. 7 presents comparisons of sampling quality (e.g., the difference in frequency of target class values). From the results, we can see that our method has better sampling quality compared to existing algorithms, and the drawn samples are more balanced and fair. In contrast, existing algorithms have poor sampling quality, and differences are sufficiently large in some cases. These results fortify the significance of our method in terms of achieving better sampling quality, leading to the development of high-quality ML classifiers.

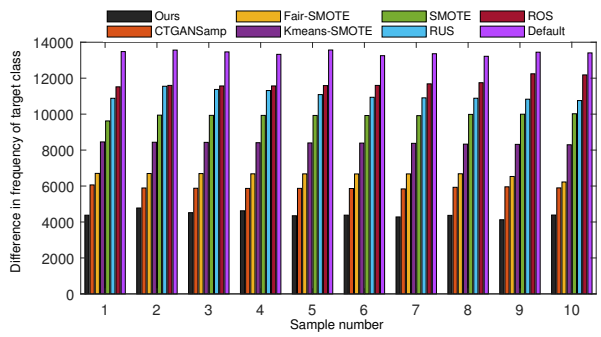


Fig. 7: Comparisons of sampling quality: proposed method vs existing SOTA data augmentation techniques.

To further justify the efficacy of the proposed model, we compared the results with five SOTA generative models in terms of ML utility. Specifically, we built RF classifiers with varying amounts of data and analyzed the accuracy. Table 5 presents a comparison of the results (an average of 10 tests). Although most methods yielded competitive performance in terms of accuracy, the quality of the underlying T_s was better for either categorical columns or numerical columns. Also, a greater contribution in accuracy came from the majority class only, and therefore, the T_s made by the existing generative models cannot be used in safety-critical applications. In contrast, our method generated a high-quality T_s with balanced distributions, leading to better performance in terms of ML utility.

TABLE 5: Accuracy comparison: proposed method versus existing SOTA generative models (avg. of 10 tests).

Generative Model	Accuracy (%)
MedGAN [27]	72.11
TableGAN [28]	78.33
CTGAN [29]	79.29
CW-GAN [30]	79.95
GANBLR [31]	81.21
Proposed CGAN	83.78

5.3.2 *Accuracy comparisons between T, T_s, and $T + T_s$ by varying number of records*

In this section, we evaluate and compare the accuracy between T, T_s, and $T + T_s$ by varying number of records. The obtained results are shown in Fig. 8. From the results in Fig. 8, we can conclude that the accuracy of T_s alone was lower than T, as (seen in Fig. 8 (left)). However, T_s can help augment the performance of AI models when jointly used with T, as shown in Fig. 8(right).

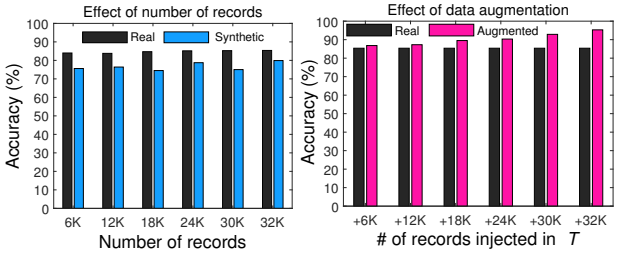


Fig. 8: Accuracy comparison of T_s, T, and $T + T_s$.

From the extensive experiments and comparisons, we found that T_s can enhance the accuracy of ML models by 9.92% if we mix T and T_s (e.g., $|T| + |T_s|$). After careful analysis, we found that mixing T_s into T can increase the strength of minority classes, leading to better prediction accuracy. The generality of the training process was also enhanced. We found that T_s can be a viable alternate (from the ML perspective) when T is available in a limited way.

5.3.3 *Accuracy comparisons between T and T_s using different ML classifiers*

To further justify the universality of results, we compared the results by using three different ML classifiers, as shown in Table 6. From the results, it can be seen that the quality of T is vital for generating the best quality T_s.

TABLE 6: Accuracy comparison of T_s with T and a prior method using three different classifiers.

Dataset/study	DT (%)	SVM (%)	RF (%)	Avg. (%)
T (real data)	78.25	81.15	85.38	79.76
T_s (using raw T)	72.84	75.20	79.90	75.98
T_s (using improved T)	75.04	79.29	83.78	79.37

The main reason for the improvement was the selection of optimized values for hyperparameters. Experimental analysis certified that a good-quality T_s has many benefits in the era of AI. Through these experiments, we found that RF is a good choice for prediction/classification tasks. The results given in Tables 4-6 and Figures 8-6 demonstrate the efficacy of T_s in enhancing the performance of ML models. The comparisons with diverse generative models and augmentation methods prove the technical supremacy of our method. In addition, the accuracy results by varying data characteristics and utilization of different ML models verify the efficacy of T_s generated with our method for ML applications. Lastly, our method contributed to balanced sampling and confusion matrices, ensuring robust ML model development compared to previous methods. In the next subsection, we analyze and compare the quality of T_s from four different aspects, such as query answers, data mining, privacy and utility, and general perspectives.

5.4 Quantifying the significance of T_s from the perspective of query-based systems

In most data-driven applications, analysts usually execute different queries to perform the desired analytics without acquiring the whole T. By doing so, storage space is preserved, and most of the computation can be performed in the data owners' environments. To further evaluate the significance of T_s from the perspective of knowledge-based systems (i.e., query and answer), we executed multiple queries involving one, two, and multiple attributes, and compared the results from T and T_s. We devised these queries in the form of questions such as *What is the common profession of people in the Republic of China? Which profession is good in terms of earnings (i.e., income greater than 50K)? Which race value is most dominant in a dataset? Does gender inequity exist in terms of earnings? What is the best age for earning* $60K$ *a month?* We checked answers from both T and T_s, finding that T_s yielded consistent answers in most cases, and can be helpful in knowledge-based systems (e.g., query-based systems). The statements for queries and the tuples returned (or %) analyzed from both T_s and T are in Table 7.

We analyzed the error rate in terms of differences in the number of tuples, and we present the results for each query (in %) as (1, 64), (2, 39.5), (3,13.2), (4, 64.7), (5, 11.6), (6, 12.5), (7, 64.4), (8, 17.65), (9, 24.6), (10, 7.2), (11, 23.35), (12, 20.9), (13, 32.7), (14, 10.7), (15, 10.9), (16, 11). Through analysis of results, we noticed that queries executed on numerical attributes had relatively more errors than categorical ones. The largest difference in the number of tuples was observed in query # 4, while the least difference was observed in query # 10. In most cases, the differences were within an acceptable range considering other qualitative factors (especially the nature of the queries). These findings shed light on the significance of T_s in query-based systems that are most prevalent in a digitized society. Lastly, this analysis highlights another potential use of T_s in digital systems (e.g., count/aggregation query-answering).

TABLE 7: Descriptions of queries executed on both T and T_s and the corresponding results.

Query syntax/overview	**Focus attribute**	$\|r\|$ **in** T	$\|r\|$ **in** T_s
*SELECT * FROM T/T_s WHERE age < 20;*	Numerical	1,657	4,591
*SELECT * FROM T/T_s WHERE income = '>50K';*	Categorical	7,841	12,971
SELECT Race, COUNT(Race) As Most-Dominance FROM T/T_s GROUP BY Race ORDER BY COUNT (Race) DESC;	Categorical	White: 27816	White: 24139
*SELECT * FROM T/T_s WHERE income = '<50K' AND working-hours > 40;*	Hybrid	5,725	16,222
SELECT Country, COUNT(Country) As Most-Earnings FROM T/T_s WHERE income ='>50K' GROUP BY Country ORDER BY COUNT (Country) DESC;	Categorical	US: 29170	US: 25,765
SELECT age, SUM(income='>50K') AS low, SUM(income='<=50K') As high, COUNT (*) As Overall FROM T/T_s WHERE age BETWEEN 0 AND 90 AND Country='US' GROUP BY age;	Categorical	40^{+} Yrs.	35^{+} Yrs.
*SELECT * FROM T/T_s WHERE income = '>50K' AND working-hours > 40;*	Hybrid	3,856	10,860
SELECT gender, SUM (income='>50K') As low, SUM (income='<=50K') As Total-income FROM T/T_s Group By gender	Categorical	F: >50K → 10.9% & M: >50K → 30.5%	F: >50K → 11% & M: >50K → 40.5%
*SELECT * FROM T/T_s WHERE income = '>50K' AND education NOT IN (' Bachelors', ' Masters',' Doctorate')*	Categorical	Yes: 15.4%	Yes: 21.01%
SELECT profession, COUNT(profession) As Least-Earnings FROM T/T_s WHERE income ='<=50K' GROUP BY profession ORDER BY COUNT (profession) DESC;	Categorical	PHS: 99.32%	TS: 65.59%
SELECT profession, COUNT(profession) As Most-Earnings FROM T/T_s WHERE income ='>50K' GROUP BY profession ORDER BY COUNT (profession) DESC;	Categorical	EM: 48.5%	CR: 43.8%
SELECT COUNT (*) FROM T/T_s WHERE income ='>50K' AND education = 'Doctorate' AND country= 'US'	Categorical	77.9%	35.21%
SELECT gender, COUNT(gender) As Rich-Individuals FROM T/T_s WHERE income ='>50K' AND age ¿ 60 GROUP BY gender ORDER BY COUNT (gender) DESC;	Categorical	F: 11.07% & M: 31.2%	F: 37.9 % & M: 41.9%
SELECT profession, COUNT(profession) As Common-Profession FROM T/T_s WHERE country ='China' GROUP BY profession ORDER BY COUNT (profession) DESC;	Categorical	Professor: 23.3%	Professor: 12.4 %
SELECT COUNT (*) FROM T/T_s WHERE martial-status ='Married'	Categorical	47.3%	58.3%
SELECT COUNT (*) FROM T/T_s WHERE martial-status IN('Divorced', 'Never-married', 'Separated', 'Widowed')	Categorical	52.7%	41.7%

Abbreviations: F: Females, M: Males, EM: Executive-managerial, CR: Craft-repair, PHS: Private house servant, TS: Tech support, DESC=:descending order

5.5 Quantifying the significance of T_s from the perspective of data mining (e.g., pattern mining & analysis)

Analyzing data and drawing pictures from them is one of the most investigated topics, and is a mainstream solution for data-driven applications. Many companies around the globe are generating huge profits by finding insightful patterns from data [44]. To mine patterns from data, we executed different association rules from T_s and T, and compared support values (denoted as $Supp$). Rules can be made based on dominant quasi-identifier (QID) and sensitive attribute (SA) values. We present sample rule R_i in Eq. 20. In this rule, we analyze the correlation between two QIDs (education and country) with income (as an SA).

$$R_i = \{edu, country\} \rightarrow income = \{Ph.D., China\} \rightarrow > 50K \tag{20}$$

The $Supp$ value for any association rule (R_i) can be computed with Eq. 21:

$$Supp_{R_i} = {}^{f(QIDs,SA)}/_{N} \tag{21}$$

where f denotes the frequency of tuples in which the respective QIDs and the SA co-exist.

To evaluate the significance of T_s from the perspective of data mining (i.e., patterns), we executed multiple associations involving one, two, and multiple QIDs, and compared the results for $Supp$ between T and T_s. Since most data-driven applications use pertinent attributes for marketing/recommendation, we used only relevant QIDs in our analysis. Through this analysis, we evaluated the quality of T_s from the perspective of data mining.

TABLE 8: Association rules-based analysis: T versus T_s.

Association rule	$Supp$ in T	$Supp$ in T_s	Diff. (%)
$US \rightarrow > 50K$	0.25	0.41	0.17
$White \rightarrow \leq 50K$	0.74	0.59	0.14
$(Doctorate, China) \rightarrow > 50K$	0.06	0.01	0.04
$(Male, > 40Hrs.perweek) \rightarrow > 50K$	0.15	0.19	0.05
$(Exec-Mng, US, White) \rightarrow > 50K$	0.44	0.37	0.06
$(> 50Yrs., Divorced, Black) \rightarrow \leq 50K$	0.017	0.009	0.007

From the analysis presented in Table 8, we can conclude that the performance of association rules (e.g., $Supp$) computed from T_s was comparable with T. Interestingly, in some cases, the $Supp$ values from T_s were higher than from T. These findings indicate the utility of T_s from the perspective of marketing/recommender systems. Rules-based analysis can assist in validating a research hypothesis as well as in generating a new hypothesis in diverse fields (especially, medical domains).

5.6 Quantifying privacy and utility aspects of T_s and T

Most previous studies assumed that privacy issues from T_s are minor because AI models do not overfit. However, we now hold a different view after analyzing the privacy and utility aspects of T_s w.r.t. T. We experimentally compared the results and found that since T_s is a replication of T, the probability of identity or SA disclosure can be high, owing to the capabilities of adversaries as well as the amount of data available to them. To verify the results, we created various anonymized versions of both T_s and T, comparing privacy and utility. We executed a background knowledge attack and a linking attack to measure privacy. In contrast, we used an information loss metric to evaluate utility.

5.6.1 Evaluation of privacy

We used a disclosure risk (a.k.a. probability of re-identification) metric in two distinct attack scenarios: linking and background knowledge (BK) attacks. In the former, an adversary tries to associate published and auxiliary data to extract the users' SA. In the latter, the adversary already has some information about the target individuals. The disclosure risk, D_r, can be determined via Eq. 22:

$$D_r(u_i, v_i) = \sum_{i=1}^{|b|} v_i / \sum_{m=1}^{k} v_m \tag{22}$$

where v_i denotes an SA value that an attacker wants to find related to the target user, u_i. The denominator shows the aggregate of the frequencies of different SA values in a class. $D_r = 1$, when all users have the same SA in a class.

We considered worst-case scenarios (journalist case) while measuring and comparing D_r values from T and T_s. We found accurate matches using QID values in distinct classes made from both T and T_s, and determined the probability of income being either $\leq$50K or $>$50K by employing multiple BK forms [45]. The overview of attributes and actual values used in the BK attack are given in Table 9.

TABLE 9: Overview of QIDs/SA and their values used in the BK attack executed on T and T_s (adapted from [45]).

Known (or already exposed) QIDs/SA	Target SA/QID value
Race, work-hours, age, country: {white, $\geq$ 40, 20, USA}	P(income $\leq$50K)
Age interval, marital status, profession: {40-50, Married, Prof.}	P(income $>$50K)
Country, gender, income: {China, Male, $>$50K }	P(age $\leq$50 Yrs.)
Sex, qualification, work-class, race : {Female, Ph.D., Private, White }	P(income $>$50K)
Marital status, sex, profession, age: {Widowed, Female, Sales, 50}	P(income $\leq$50K)

In the linkage attack, we computed the correct links based on QID values and analyzed distinct values accordingly. Table 10 presents a comparative analysis of D_r results from both attacks.

TABLE 10: Privacy analysis of T and T_s under two attacks.

k	Avg. D_r values (BK attack)		Exposed records	Avg. D_r values (Linking attack)		Previous method
	Real Data	Synthetic data		Real Data	Synthetic data	
2	$0.99 \pm \theta$	$0.53 \pm \theta$	331	$0.45 \pm \theta$	$0.40 \pm \theta$	0.06
5	$0.79 \pm \theta$	$0.41 \pm \theta$	981	$0.49 \pm \theta$	$0.42 \pm \theta$	0.06
10	$0.88 \pm \theta$	$0.49 \pm \theta$	1,954	$0.51 \pm \theta$	$0.49 \pm \theta$	0.06
15	$0.59 \pm \theta$	$0.49 \pm \theta$	3,263	$0.58 \pm \theta$	$0.49 \pm \theta$	0.06
25	$0.54 \pm \theta$	$0.50 \pm \theta$	4,895	$0.64 \pm \theta$	$0.56 \pm \theta$	0.06

From the results given in Table 10, we can see that D_r depends on the type of attack, and D_r from T_s was much higher than a previously measured generic analysis via demographics [15]. Through experiments, we found that D_r can vary based on the granularity of auxiliary information and the amount of data readily exposed to an adversary. In addition, the D_r is subject to change when a strong privacy protection method is used or fewer records are exposed. Therefore, we use $\pm\ \theta$ in Table 10 to reflect these changes in results in realistic scenarios. In one test, we found that D_r differed by just $\sim$ 2%. See 1,954 records exposed scenario in Table 10. These findings and results further rectify/improve previous studies. Through experimental analysis, it can be concluded that privacy leakage can occur from SD when contemporary attacks are launched on it.

5.6.2 Evaluation of utility

To evaluate utility, we employed an information loss (IL) metric. During anonymization, QID values are replaced with new values, and thereby, some information is always lost. To measure IL from anonymized versions of both T_s and T, we used a distortion measure (DM) metric in order to quantify IL. DM values can be determined by figuring out the level of hierarchy on which values of QIDs are mapped. The DM value can be calculated via Eq. 23:

$$DM = \sum_{u=1}^{|T^{'}|} \sum_{Q=1}^{p} \frac{l_g}{l_t} \tag{23}$$

where l_g represents the level at which the QID value was mapped, and l_t represents the total # of levels in the hierarchy. DM values from every QID and sample were summed for all N subjects in the data. If the value of a particular QID is not changed, the DM value will be 0. The IL results for five different k values are shown in Table 11.

TABLE 11: IL (a.k.a. utility) comparisons between T and T_s.

k	Avg. IL values		
	Real Data	Synthetic data	IL difference in %
4	0.51	0.55	6.77
8	0.59	0.63	7.84
12	0.60	0.66	10.00
16	0.63	0.71	12.69
20	0.65	0.74	13.84

Referring to Table 11, it can be seen that IL increase with k value, and the difference in IL between T and T_s is not very large, which indicate that quality of T_s resulted from our model is better. Although T_s has higher IL than T, it can still offer useful information concerning analytics (e.g., histogram analysis) or general data mining tasks.

5.7 Analysis of T_s from theoretical perspectives

In this section, we report experimental findings concerning T_s from theoretical perspectives.

5.7.1 Effects of data distributions

Besides the other analyses, we analyzed values' distributions in SA and QIDs. We chose four attributes, (profession, income, race, and age) for analysis. The first two belong to the SA category, and the last two are QIDs. The purpose of analyzing the distributions is to certify that T_s correctly mirrors the properties of T. Figure 9 presents the comparative results of distributions in the profession (an SA) and race (a QID). The x-axis in left Figure shows the

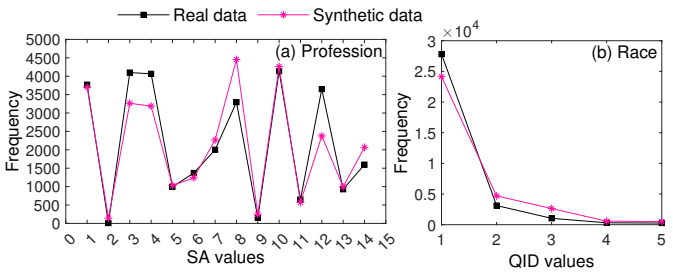


Fig. 9: SA and QID value distributions in T and T_s.

different values of profession (For example, 1=Adm-clerical, and 14=Transport-moving). The x-axis in the right Figure shows the different values of race (For example, 1=White, and 5= Other). From the results, we can see that the CGAN can correctly model all values from T with nearly perfect distributions. These results confirm the validity of T_s for downstream/general-purpose tasks. Similarly, for another SA (income), the difference between values distributions for income: $> 50K$ and $\leq 50K$ was not considerably high.

From the results, observe that all category values from T (i.e., major and minor) were correctly replicated in T_s, and the distributions are mostly alike. These results indicate that T_s has a better distribution for QID and SA. Analysis of the distribution for age is presented in Fig. 10. From the results, observe that the distribution of values in T_s is very close to the distribution in T. In addition, most age values in T_s are also very close. These findings confirm the utility of T_s from multiple aspects (e.g., data analytics and mining).

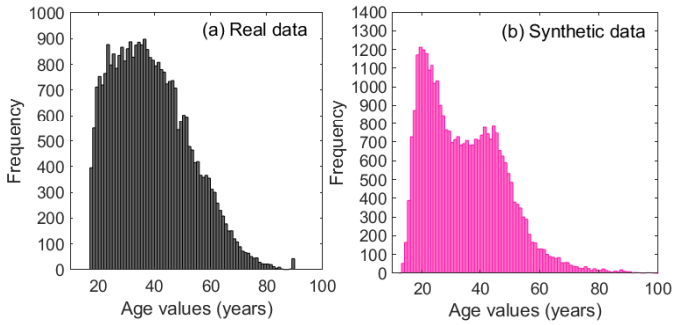


Fig. 10: Distribution analysis of age: T versus T_s.

5.7.2 Missing values

Recently, the GAN has been increasingly used in medical domains for data interpolation and augmentation [46]. To this end, we analyzed T_s regarding missing values, and found that a CGAN can assist in reducing the number of missing values, which can help knowledge discovery from T_s when using advanced data mining tools. There were three attributes with missing values (work class, occupation, and country). The CGAN reduced the number of missing values significantly in the work class, but in the other two attributes, the number of missing values was slightly large. However, the number of missing values can be reduced by optimizing the hyperparameters of the CGAN and enforcing conditions. These findings confirm the validity of T_s in improving various critical aspects of a real-world T.

5.7.3 Fulfilment of legal/regulatory compliance

Since T_s is close to T in most aspects, it can be shared with researchers without hesitation for analytics. SD can stay legally compliant by providing population-level information accurately, rather than at the individual (or sample-level) level. Also, SD can ensure legality by not sharing the real data, and can overcome the challenges of unnecessary processing and anonymization. Lastly, SD can help develop data products where most testing can be done using T_s.

5.7.4 Overcoming barriers to data distribution

There are many situations (e.g., COVID-19 tests/cases, HIV patients, dementia patients) in which access to real data is restricted due to privacy concerns and/or data unavailability on a large scale. In these circumstances, bringing algorithms close to the real data to mirror the properties of T and generate SD can be a handy tool for wide-scale distribution. In our experiments, we found that SD can be created at factors of $1\times$, $1.5\times$, and $2\times$ T, and therefore, its distribution to legitimate information consumers can be helpful in their research or in observing commonalities among differences. T_s can have multiple applications in healthcare, smart cities, natural language processing (NLP), and compliance analysis.

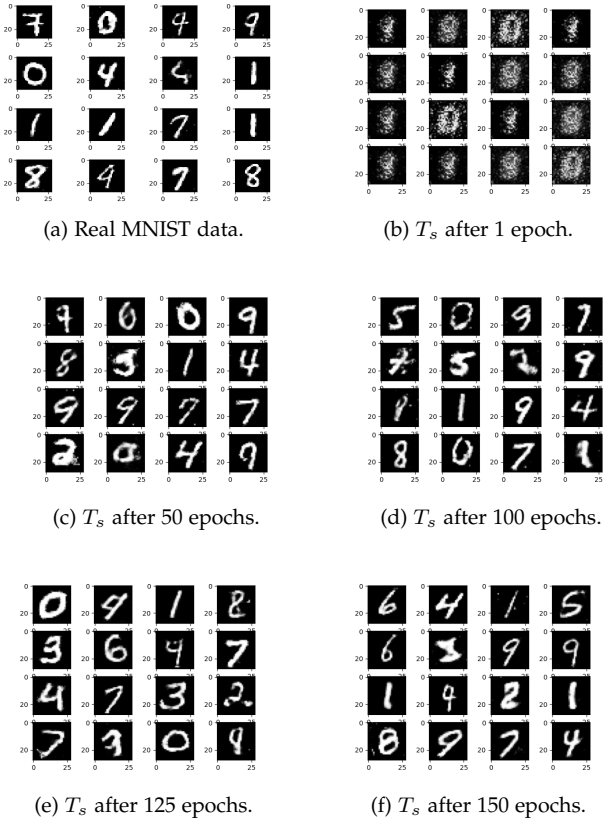
5.7.5 Privacy preservation in AI systems using T_s

As stated earlier, SD has already been used in AI systems as a defense tool against membership inference attacks [1]. In the coming years, SD will be vital in securing training data for AI systems as well as in augmenting AI model

performance. Since performance can be enhanced from huge amounts of data, joint use of T_s and T is inevitable in the near future. Finally, SD can prevent concept drift issues and white box attacks on AI systems.

5.7.6 Efficacy in image recognition and NLP-related tasks

Recently, SD has gained momentum in the research community and has been widely used in real-world applications for augmenting classifier performance, privacy preservation, software/product testing, and data governance. SD can be generated in multiple forms such as tables, images, and time series. GAN-based models are handy in generating synthetic images that can be used for downstream tasks. To generate SD in image form, we used an open-source PyTorch-based implementation of a deep convolution GAN (DCGAN) [47] and produced synthetic characters using the MNIST database of handwritten digits. In the experiments, we divided data into a training set that constituted 60,000 images, and a testing set that encompassed 10,000 images. Fig. 11 presents the results of SD for image recognition tasks.



(a) Real MNIST data. (b) T_s after 1 epoch.

(c) T_s after 50 epochs. (d) T_s after 100 epochs.

(e) T_s after 125 epochs. (f) T_s after 150 epochs.

Fig. 11: GAN performance in an image recognition task.

From the results, we can see that GAN models can assist in generating images of good quality that are close to the real data. It is important to note that image quality was not ideal at the start, and characters were not easily readable. However, the quality of characters improved with an increase in the number of epochs. We believe that by training the GAN model for a long time, good-quality images can be curated. Similarly, a GAN can also be used to generate SD that can be used in NLP tasks [48]. For instance, SD can be used for sentiment analysis, automatic correction in machine translation, knowledge distillation, next-word prediction, and text generation, especially when the labeled data are limited [49]. However, the quality and size of the SD to be used in NLP tasks are very challenging and require further investigation from the research community.

It is important to note that T_s has many general benefits as discussed in Section 5.7. For example, it can be helpful to analyze the distribution of values in some specific attribute of data as shown in Figure 9. It can also be used to perform histogram/frequency analysis of some attributes as shown in Figure 10. It can be used to impute missing values to complete data as discussed in subsection 5.7.2. It can be used to fulfill legal/regulatory requirements as it is regarded as a coarse form of T, as discussed in subsection 5.7.3. It can foster secondary uses of data by enabling data sharing at a large scale as discussed in subsection 5.7.4. It can be used to address privacy issues of different types in AI environments as discussed in subsection 5.7.5. It can also be used in diverse tasks such as image recognition (as shown in Figure 11) and NLP tasks, as discussed in subsection 5.7.6. With the advancements in AI, T_s is becoming one of the mainstream technologies, particularly when access to real data is limited due to either privacy concerns or regulatory measures.

6 Discussion

Tabular data is one of the ubiquitous and most widely used forms of data in AI applications. Due to its simplified nature and easier interpretation, it has been used in most sectors. Due to privacy concerns and a lack of expertise in handling personal data, tabular data cannot be easily outsourced from data owner environments. In contrast, most AI applications require a mammoth volume of data for training classifiers to solve real-world problems [50]. In some cases, data owners outsource the data in an anonymized form to conduct analytics. However, anonymized data cannot be directly fed into classifiers, and extensive post-processing is needed to make it AI-ready. Furthermore, anonymized data yield limited performance (e.g., lower accuracy) in AI models, as shown in Fig. 12. In contrast, data generated with generative AI tools can be synergized with different forms of data to compensate for the availability of good data. Augmented data (real data combined with SD) can enhance the accuracy of AI models, leading to an effective solution for classification/prediction.

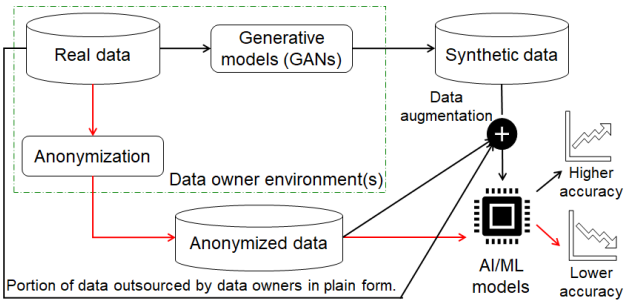
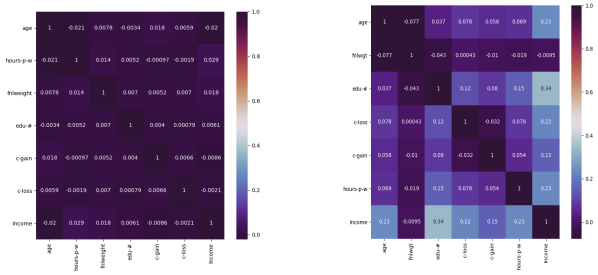


Fig. 12: Potential usage of T_s in the date-driven era.

It is important to note that T_s generation is a non-trivial task, especially when T has mixed attribute types

and many vulnerabilities (e.g., imbalanced distributions, fewer records, outliers, missing values, etc.). Thus far, many generative models have been proposed to generate T_s from T, but when the quality of underlying T is poor, most models cannot generate good T_s. Furthermore, most of the prior generative models are less flexible, meaning AI practitioners cannot curate data based on their needs. Also, the poor design of previous generative models and hyperparameters can also lead to overfitting issues, and the training process can become unstable. To address these practical challenges, this paper implemented a CGAN-based model to yield superior-quality SD. Our model is more flexible than previous SOTA models. The resulting SD were evaluated from both privacy and utility perspectives, whereas existing methods only analyzed the closeness between T and T_s, and ignored privacy issues that can emerge from T_s. Lastly, the T_s produced with our model has many applications in the AI domain such as data augmentation, balanced learning in AI models, higher accuracy, and fair decision-making. We believe our analysis and findings from a real dataset can effectively address the performance bottlenecks currently faced by numerous GAN-based models.

To prove the importance of the proposed CGAN in realistic scenarios, more intuitive results are given in Figs. 13 and 14. From Fig. 13, we can see that a T_s produced by our method has better quality than the SOTA model in terms of correlations. Through detailed analysis, we found only one attribute negatively correlated with income in T. However, the T_s produced with the CTGAN had more attributes negatively correlated with income. Therefore, the T_s produced with that model can lead to wrong analyses or conclusions in realistic scenarios. In contrast, our method correctly retained this property of T, and T_s had only a few attributes negatively correlated with income. Hence, the possibility of wrong analyses or conclusions in realistic scenarios is the least.

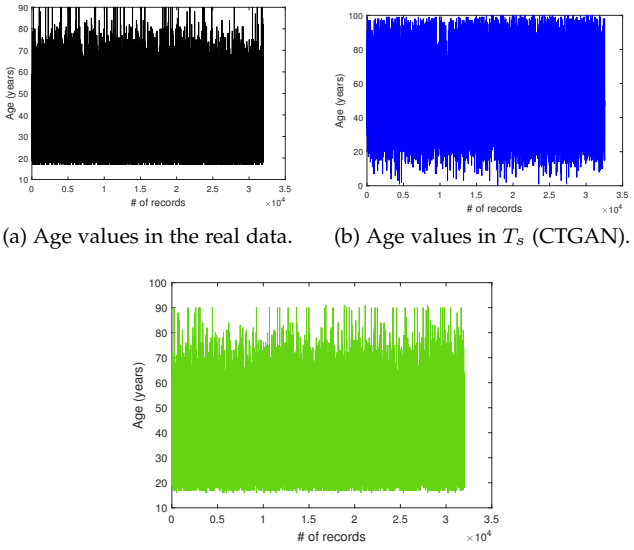


(a) Correlation matrix for CTGAN model.

(b) Correlation matrix for proposed method.

Fig. 13: Comparisons of correlation matrices for T_s.

In another set of experiments, we compared the closeness between attribute values in T and T_s, and the corresponding results are in Fig. 14. As stated earlier, the CGAN model can produce inconsistent values for some columns, and the offset between real and synthetic values is very large, as shown in Fig. 14(b). In contrast, our method did not yield any inconsistent values, and the offset between real and synthetic values was very small, as shown in Figure 14(c). These results fortify the significance of our method in terms of correctly mimicking the properties of T.



(a) Age values in the real data.

(b) Age values in T_s (CTGAN).

(c) Age values in T_s (our method).

Fig. 14: Comparison of closeness of values in T and T_s.

Based on the analysis above, good-quality SD can be used as a substitute for personal big data (i.e., private data stemming from people's daily lives and/or work). However, creating a knowledge base/store of SD to evaluate its effectiveness from multiple perspectives with advanced data mining and AI tools needs further investigation.

6.1 Broader impact of this research

Due to the rapid proliferation of data generation tools (e.g., wearable devices, sensors, etc.), data of diverse modalities is curated, which can be utilized to generate actionable insights with AI models. Unfortunately, most of the data gets locked in the data owner's environment due to privacy issues, and cannot be shared with relevant data owners/analysts, impacting the data-driven innovation and financial opportunities. How to share the data at a large scale while limiting privacy concerns is a key challenge in the modern AI-driven era. To alleviate privacy concerns, governments in Europe, the U.S., and other countries have been proposing and passing regulations to ensure privacy protection and data traceability across data-driven products/frameworks. Also, recent developments like federated learning are assisting in accomplishing the crucial task of user privacy preservation while allowing AI model development with fragmented/scattered datasets. However, addressing all types of privacy problems and ensuring the responsible use of data is still very challenging. To this end, SD is a novel solution that can effectively address privacy issues while enabling data sharing on a large scale. Recently, it has become a mainstream solution for AI developments to improve training data quality, contributing to AI model development of higher generalizability and robustness.

SD can be very useful in solving data island problem, the realization of a data-centric AI concept, democratizing AI developments, addressing privacy concerns, and developing data-driven products. Our work targets two crucial

aspects in the SD field: high-quality SD generation and its experimental evaluation. Our model results in the best quality SD generation via an improved CGAN which can be generically applied, and can improve the performance of ML techniques in terms of accuracy, sampling quality, and confusion matrix's balancedness. Our evaluation of SD in terms of privacy and utility underscores the need for proper assessment of SD w.r.t. privacy leakage and data/information availability. We delve into the downsides of SD in terms of privacy leakage, and we find that SD is less resistant to contemporary privacy attacks. On the other hand, we test the efficacy of SD in terms of various data mining tasks, which can be useful to harness the potential of SD in relevant scenarios. Lastly, our findings can be vital to ML and the database community to properly govern and use SD in real-world applications. Finally, we suggest that careful analysis of scenarios/situations when one can/cannot fully rely on the results derived from T_s needs more research from the research communities.

7 Conclusion and Future Work

In this paper, we implemented a flexible and generic solution for best-quality synthetic tabular data generation with mixed attribute types (i.e., categorical and numerical). Specifically, we solved the following implementation challenges in data generation: numerical attribute modeling and transformation when attributes cannot be represented in the normalized form of [-1,1], which can lead to the vanishing gradient problem in GAN, and categorical attribute modeling when distributions of some attributes can be highly imbalanced, thus preventing model collapse during simultaneous training of neural networks. We also enforced conditions in data generation to reflect all categories from T in T_s (whether major or minor), relaxing constraints on the amount of T_s to be generated. We fortified the effectiveness of the data generated by the CGAN with the help of multiple use cases that have not been experimentally tested in the literature. To the best of our knowledge, this is the maiden attempt at providing a deeper analysis of T_s in both original and anonymized forms. Our work contributes a major privacy-enhancing technology (PET) to reach the goals of responsible data science (e.g., analytics of personal data without misuse) and data augmentation (e.g., improving the performance of AI models by amalgamating T and T_s). In the future, we plan to further optimize the performance of the CGAN model and compare it with advanced generative models. We intend to extend our method to generate synthetic image data that can be helpful in medical domains for COVID-19 diagnosis or mobile-physician apps. Lastly, we intend to generate a T_s of high quality that can be applied to various NLP-related tasks.

Acknowledgments

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (Ministry of Science and ICT) (RS-2024-00340882).

References

[1] L. Hu, J. Li, G. Lin, S. Peng, Z. Zhang, Y. Zhang, and C. Dong, "Defending against membership inference attacks with high utility by gan," *IEEE Transactions on Dependable and Secure Computing*, 2022.

[2] M. Hernandez, G. Epelde, A. Beristain, R. Álvarez, C. Molina, X. Larrea, A. Alberdi, M. Timoleon, P. Bamidis, and E. Konstantinidis, "Incorporation of synthetic data generation techniques within a controlled data processing workflow in the health and wellbeing domain," *Electronics*, vol. 11, no. 5, p. 812, 2022.

[3] M. Hernandez, G. Epelde, A. Alberdi, R. Cilla, and D. Rankin, "Synthetic data generation for tabular health records: A systematic review," *Neurocomputing*, 2022.

[4] S. James, C. Harbron, J. Branson, and M. Sundler, "Synthetic data use: exploring use cases to optimise data utility," *Discover Artificial Intelligence*, vol. 1, no. 1, pp. 1–13, 2021.

[5] L. Li, Y. Fan, M. Tse, and K.-Y. Lin, "A review of applications in federated learning," *Computers & Industrial Engineering*, vol. 149, p. 106854, 2020.

[6] Z. Azizi, C. Zheng, L. Mosquera, L. Pilote, and K. El Emam, "Can synthetic data be a proxy for real clinical trial data? a validation study," *BMJ open*, vol. 11, no. 4, p. e043497, 2021.

[7] E. Strickland, "Andrew ng, ai minimalist: The machine-learning pioneer says small is the new big," *IEEE Spectrum*, vol. 59, no. 4, pp. 22–50, 2022.

[8] C. Hegde, "Anomaly detection in time series data using data-centric ai," in *2022 IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT)*. IEEE, 2022, pp. 1–6.

[9] M. Motamedi, N. Sakharnykh, and T. Kaldewey, "A data-centric approach for training deep neural networks with less data," *arXiv preprint arXiv:2110.03613*, 2021.

[10] A. Zeiser, B. Özcan, B. van Stein, and T. Bäck, "Evaluation of deep unsupervised anomaly detection methods with a data-centric approach for on-line inspection," *Computers in Industry*, vol. 146, p. 103852, 2023.

[11] A. Yale, S. Dash, R. Dutta, I. Guyon, A. Pavao, and K. P. Bennett, "Generation and evaluation of privacy preserving synthetic health data," *Neurocomputing*, vol. 416, pp. 244–255, 2020.

[12] D. Rankin, M. Black, R. Bond, J. Wallace, M. Mulvenna, G. Epelde *et al.*, "Reliability of supervised machine learning using synthetic data in health care: Model to preserve privacy for data sharing," *JMIR Medical Informatics*, vol. 8, no. 7, p. e18910, 2020.

[13] G. Deng, C. Han, and D. S. Matteson, "Extended missing data imputation via gans for ranking applications," *Data Mining and Knowledge Discovery*, pp. 1–23, 2022.

[14] K. El Emam, "Seven ways to evaluate the utility of synthetic data," *IEEE Security & Privacy*, vol. 18, no. 4, pp. 56–59, 2020.

[15] K. El Emam, L. Mosquera, E. Jonker, and H. Sood, "Evaluating the utility of synthetic covid-19 case data," *JAMIA open*, vol. 4, no. 1, p. ooab012, 2021.

[16] N. C. Abay, Y. Zhou, M. Kantarcioglu, B. Thuraisingham, and L. Sweeney, "Privacy preserving synthetic data release using deep learning," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2019, pp. 510–526.

[17] S.-C. Li, B.-C. Tai, and Y. Huang, "Evaluating variational autoencoder as a private data release mechanism for tabular data," in *2019 IEEE 24th Pacific Rim International Symposium on Dependable Computing (PRDC)*. IEEE, 2019, pp. 198–1988.

[18] S. Wang, C. Rudolph, S. Nepal, M. Grobler, and S. Chen, "Part-gan: privacy-preserving time-series sharing," in *International Conference on Artificial Neural Networks*. Springer, 2020, pp. 578–593.

[19] J. Martinsson, E. L. Zec, D. Gillblad, and O. Mogren, "Adversarial representation learning for synthetic replacement of private attributes," in *2021 IEEE International Conference on Big Data (Big Data)*. IEEE, 2021, pp. 1291–1299.

[20] A. Torfi, E. A. Fox, and C. K. Reddy, "Differentially private synthetic medical data generation using convolutional gans," *Information Sciences*, vol. 586, pp. 485–500, 2022.

[21] A. Appenzeller, M. Leitner, P. Philipp, E. Krempel, and J. Beyerer, "Privacy and utility of private synthetic data for medical data analyses," *Applied Sciences*, vol. 12, no. 23, p. 12320, 2022.

[22] N. Elaraby, S. Barakat, and A. Rezk, "A conditional gan-based approach for enhancing transfer learning performance in few-shot hcr tasks," *Scientific Reports*, vol. 12, no. 1, p. 16271, 2022.

[23] H. Lu, M. Du, K. Qian, X. He, and K. Wang, "Gan-based data augmentation strategy for sensor anomaly detection in industrial robots," *IEEE Sensors Journal*, vol. 22, no. 18, pp. 17 464–17 474, 2021.

[24] M. Hammami, D. Friboulet, and R. Kéchichian, "Cycle gan-based data augmentation for multi-organ detection in ct images via

yolo," in *2020 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2020, pp. 390–393.

[25] R. Zhang, W. Lu, J. Gao, Y. Tian, X. Wei, C. Wang, X. Li, and M. Yu, "Rfi-gan: A reference-guided fuzzy integral network for ultrasound image augmentation," *Information Sciences*, vol. 623, pp. 709–728, 2023.

[26] H. Chen, J. Chen, and J. Ding, "Data evaluation and enhancement for quality improvement of machine learning," *IEEE Transactions on Reliability*, vol. 70, no. 2, pp. 831–847, 2021.

[27] E. Choi, S. Biswal, B. Malin, J. Duke, W. F. Stewart, and J. Sun, "Generating multi-label discrete patient records using generative adversarial networks," in *Machine learning for healthcare conference*. PMLR, 2017, pp. 286–305.

[28] N. Park, M. Mohammadi, K. Gorde, S. Jajodia, H. Park, and Y. Kim, "Data synthesis based on generative adversarial networks," *arXiv preprint arXiv:1806.03384*, 2018.

[29] 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.

[30] J. Engelmann and S. Lessmann, "Conditional wasserstein gan-based oversampling of tabular data for imbalanced learning," *Expert Systems with Applications*, vol. 174, p. 114582, 2021.

[31] Y. Zhang, N. Zaidi, J. Zhou, and G. Li, "Interpretable tabular data generation," *Knowledge and Information Systems*, pp. 1–29, 2023.

[32] H. Shamsudin, U. K. Yusof, A. Jayalakshmi, and M. N. A. Khalid, "Combining oversampling and undersampling techniques for imbalanced classification: A comparative study using credit card fraudulent transaction dataset," in *2020 IEEE 16th international conference on control & automation (ICCA)*. IEEE, 2020, pp. 803–808.

[33] A. A. El-Sayed, M. A. M. Mahmood, N. A. Meguid, and H. A. Hefny, "Handling autism imbalanced data using synthetic minority over-sampling technique (smote)," in *2015 third world conference on complex systems (WCCS)*. IEEE, 2015, pp. 1–5.

[34] T. Tang, D. Jiao, T. Chen, and G. Gui, "Medium-and long-term precipitation forecasting method based on data augmentation and machine learning algorithms," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 15, pp. 1000–1011, 2022.

[35] J. Chakraborty, S. Majumder, and T. Menzies, "Bias in machine learning software: Why? how? what to do?" in *Proceedings of the 29th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, 2021, pp. 429–440.

[36] A. S. Dina, A. Siddique, and D. Manivannan, "Effect of balancing data using synthetic data on the performance of machine learning classifiers for intrusion detection in computer networks," *IEEE Access*, vol. 10, pp. 96 731–96 747, 2022.

[37] C. M. Bishop and N. M. Nasrabadi, *Pattern recognition and machine learning*. Springer, 2006, vol. 4, no. 4.

[38] Z. Lin, A. Khetan, G. Fanti, and S. Oh, "Pacgan: The power of two samples in generative adversarial networks," *Advances in neural information processing systems*, vol. 31, 2018.

[39] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[40] 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.

[41] K. Liu and G. Qiu, "Lipschitz constrained gans via boundedness and continuity," *Neural Computing and Applications*, vol. 32, pp. 18 271–18 283, 2020.

[42] P. Murphy, "Uci repository of machine learning databases. department of information and computer science, university of california," *http://www. ics. uci. edu/AI/ML/MLDBRepository. html*, 1992.

[43] R. Shen, S. Bubeck, and S. Gunasekar, "Data augmentation as feature manipulation," in *International conference on machine learning*. PMLR, 2022, pp. 19 773–19 808.

[44] I. K. Nti, J. A. Quarcoo, J. Aning, and G. K. Fosu, "A mini-review of machine learning in big data analytics: Applications, challenges, and prospects," *Big Data Mining and Analytics*, vol. 5, no. 2, pp. 81–97, 2022.

[45] A. Majeed and S. O. Hwang, "When ai meets information privacy: The adversarial role of ai in data sharing scenario," *IEEE Access*, 2023.

[46] A. Tsourtis, G. Papoutsoglou, and Y. Pantazis, "Gan-based training of semi-interpretable generators for biological data interpolation and augmentation," *Applied Sciences*, vol. 12, no. 11, p. 5434, 2022.

[47] Z. Liu, M. Tong, X. Liu, Z. Du, and W. Chen, "Research on extended image data set based on deep convolution generative adversarial network," in *2020 IEEE 4th Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)*, vol. 1. IEEE, 2020, pp. 47–50.

[48] L. F. A. O. Pellicer, T. M. Ferreira, and A. H. R. Costa, "Data augmentation techniques in natural language processing," *Applied Soft Computing*, vol. 132, p. 109803, 2023.

[49] X. He, I. Nassar, J. Kiros, G. Haffari, and M. Norouzi, "Generate, annotate, and learn: Nlp with synthetic text," *Transactions of the Association for Computational Linguistics*, vol. 10, pp. 826–842, 2022.

[50] A. Kiran and S. S. Kumar, "Synthetic data and its evaluation metrics for machine learning," in *Information Systems for Intelligent Systems: Proceedings of ISBM 2022*. Springer, 2023, pp. 485–494.

Abdul Majeed received the B.S. degree in Information Technology from the UIIT, PMAS-UAAR, Rawalpindi, Pakistan, in 2013, the M.S. degree in Information Security from the COMSATS University, Islamabad, Pakistan, in 2016, and the Ph.D. degree in Computer Information Systems & Networks from the Korea Aerospace University, Korea, in 2021. He worked as a Security Analyst with Trillium Information Security Systems (TISS), Rawalpindi, Pakistan, from 2015 to 2016. He is currently working as an Assistant Professor with the Department of Computer Engineering, Gachon University, Korea. His research interests include data-centric artificial intelligence, privacy-preserving data publishing, and machine learning.

Seong Oun Hwang received the B.S. degree in mathematics from Seoul National University, in 1993, the M.S.degree in information and communications engineering from the Pohang University of Science and Technology, in 1998, and the Ph.D. degree in computer science from the Korea Advanced Institute of Science and Technology, in 2004, South Korea. He worked as a Software Engineer with LG-CNS Systems,Inc., from 1994 to 1996. He also worked as a Senior Researcher with the Electronics and Telecommunications Research Institute (ETRI), from 1998 to 2007. He worked as a Professor with the Department of Software and Communications Engineering, Hongik University, from 2008 to 2019. He is currently working as a full Professor with the Department of Computer Engineering, Gachon University, Korea. His research interests include cryptography, cybersecurity, and data-centric artificial intelligence.

Manuscript ID: TBD-2023-03-0133.R1

Article Title: "Moving Conditional GAN Close to Data: Synthetic Data Generation and its Experimental Evaluation"

Submission Date: 2023-03-29

To: IEEE Transactions on Big Data, Editors

Re: Response to reviewers

Dear Editor-in-chief, Prof. Dr. Jie Tang,

We are pleased to submit a revised version of our paper, with an opportunity to address the associate editor's minor concerns. We would like to express our sincere appreciation to the Editor-in-chief, Associate Editor, and Expert Reviewers for providing us with valuable comments that have greatly improved our paper compared to former versions. We have carefully modified the paper in response to the constructive and helpful comments provided by the distinguished associate editor. We added additional explanations and improvised the contents, references, in-depth analysis of results, and artwork quality to address the raised concerns. We sincerely thank the editors and evaluators again for their very helpful feedback on our paper.

We are uploading (a) our point-by-point response to the comments (below) (**Summary of Changes**), (b) an updated manuscript with highlighting indicating changes (**Main Manuscript-Changes Highlighted**), and (c) a clean updated manuscript without highlights (**Main Manuscript***).*

Best regards,

Prof. Dr. Seong Oun Hwang,
Department of Computer Engineering,
Gachon University, South Korea
Email: sohwang@gachon.ac.kr

Associate Editor Comments and Authors' Responses

Note: The changes suggested by the **Distinguished Associate Editor** are marked in **Blue** in the revised paper.

In this paper, the authors propose and implement a synthetic data generation method from a real dataset containing numerical and categorical attributes using a conditional generative adversarial network. The reviews are generally positive in the appreciation of the proposed method and experimental study. The authors should further address the comments.

First of all, we sincerely thank the distinguished associate editor for handling our paper and providing very constructive feedback. Based on constructive guidance and advice, we thoroughly reworked our paper and improved its quality as advised by the associate editor. A detailed overview of our changes regarding the Associate Editor's comments is listed in the point-by-point response below.

Associate Editor, Concern # 1: There are too many defined variables

Author response:

We agree with the assessment of the reviewer regarding the too many defined variables in the earlier version of our paper. We thank the reviewer for the careful observation and for spotting this mistake that has allowed us to remove some redundant variables and equations. Based on the associate editor's constructive comment, we have re-examined all variables and excluded some variables. We also improved the description of frequently used variables in the revised manuscript to address the comment.

Author action: We updated the manuscript by improving the description of variables and by removing some redundant variables as advised by the editor in the revised work.

The suggested changes are highlighted/removed in the following places in our revised manuscript.
Page #: 8, Section: #: 4.2, Colum Side: Left, Equation #: 17-18, and corresponding variables [Removed]
Page #: 13, Section: #: 5.5.2, Colum Side: Left, Variable H [Removed]
Page #: 7, Section: #: 4.1.2, Colum Side: Left, Variable y changed to con for consistency [Modified]
Page #: 5, Section: #: 4, Colum Side: Right, Table #: 02, two variables defined [Defined]
Page #: 7, Colum Side: Right, Line #: 11, Redundant brackets [Removed]
Page #: 10, Section: #: 5.2, Colum Side: Right, Line #: 05, defined a variable used in Eq. 19 [Defined]

Associate Editor, Concern # 2: Clarify the definitions and explain the methods/terms in the manuscript per the review comments.

Author response:

We agree with the reviewer's assessment regarding the lack of clarification of definitions and the missing explanation of the methods/terms in the earlier version of our paper. We thank the reviewer for spotting these mistakes which has allowed us to rectify them in the revision. According to the constructive directions, we have clarified the definitions and explained the methods/terms as advised by the editor to address the raised concern. We kindly state that methods are adequately described based on the reviewer's comments in the previous round, and therefore, we only highlight the place of description for the editor's assessment in the revision.

Author action: We updated the manuscript by clarifying definitions and explaining the methods/terms as advised by the associate editor in the revised work.

The suggested changes are highlighted in the following places in our revised manuscript.
Page #: 7, Section: #: 4, Colum Side: Right, Paragraph #: 02, Lines #: 25-onward, Point (I, II, III)
[Methods general components description]
Page #: 8 and 9, Section: #: 4.4, Colum Side: Right & Left, Entire subsection
[Methods supportive components description]

Associate Editor, Concern # 3: Provide a comprehensive description of the experimental settings.

Author response:

We agree with the reviewer's assessment regarding the lack of a comprehensive description of the experimental settings in the earlier versions of our paper. We thank the associate editor for the careful observation and valuable suggestions that have allowed us to correct these deficits in our paper. Based on the editor's direction, we have provided a comprehensive description of the experimental settings in the revised work from both aspects: data generation, and generated data evaluation.

Author action: We updated the manuscript by adding a comprehensive description of the experimental settings (data generation part and its experimental evaluation part) as advised by the associate editor.

The suggested changes are highlighted in the following places in our revised manuscript.
Page #: 9 and 10, SubSection: #: 5.2, Paragraph 1 and 2, Colum Side: Right & Left, Entire Subsection.
Paragraph #: 01, [Experimental settings for data generation part]
Paragraph #: 02, [Experimental settings for generated data evaluation part]

Associate Editor, Concern # 4: Conduct in-depth analysis of experimental results.

Author response:

We sincerely thank the associate editor for advising us to conduct an in-depth analysis of experimental results that was lacking in the earlier versions of our paper. We sincerely thank the editor for the constructive feedback that allowed us to explain the results with more clarity. Based on the reviewer's direction, we have arranged the results into five main parts and provided an in-depth analysis of experimental results discussed in each part. Lastly, we provided the broader impact of this research to further highlight the concept as well results discussed in our paper.

Author action: We updated the manuscript by arranging the result, and providing an in-depth analysis of the experimental results as well as the broader impact of this research as advised by the associate editor.

The suggested changes are highlighted in the following places in our revised manuscript.
Page #: 10 & 11, Section: #: 5.3 [subsection 5.3.1, 5.3.2, and 5.3.3]
[Better organization of the results and comparisons].
.....
Page #: 12, Section: #: 5.3., Paragraph #: 02, Column Side: Left, Lines #: 7-17
[Analysis of results reported in subsection 5.3]
.....
Page #: 12, Section: #: 5.4, Paragraph #: 02, Column Side: Left & Right, Lines #: 5-10 & 1-6
[Analysis of results reported in subsection 5.4]
.....
Page #: 13, Section: #: 5.5, Paragraph #: 02, Column Side: Left, Lines #: 1-9
[Analysis of results reported in subsection 5.5]
.....
Page #: 13, Section: #: 5.6.1, Paragraph #: 04, Column Side: Right, Lines #: 1-13
[Analysis of results reported in subsection 5.6.1]
Page #: 14, Section: #: 5.6.2, Paragraph #: 01, Column Side: Left, Lines #: 1-6
[Analysis of results reported in subsection 5.6.2]
.....
Page #: 15, Section: #: 5.7.6, Paragraph #: 02, Column Side: Right, Lines #: 1-18
[Analysis of results reported in subsection 5.7 including subsections 5.7.1 ~ 5.7.6]
.....^^^^^.....
Page #: 16 & 17, Section: #: 6.1, Paragraph #: 01 & 02, Column Side: Right &left, Entire subsection
[Broad impact (e.g., analysis of results as well as the concept) of this research]

<Authors sincerely thank the distinguished Editor, Esteemed Associate Editor, and three Expert Evaluators for their constructive comments and very helpful feedback on our manuscript. The suggested comments and changes have vastly improved our paper compared to the previous versions.>

Moving Conditional GAN Close to Data: Synthetic Tabular Data Generation and its Experimental Evaluation

Abdul Majeed, *Member, IEEE*, and Seong Oun Hwang, *Senior Member, IEEE*

Abstract—Recently, data has ousted oil as the most economical resource in the world, but most companies are reluctant to share customer/user data in pure form and on a large scale due to privacy concerns. Many innovative technologies (e.g., federated learning, split learning) are employed to meet the growing demand for privacy preservation. Despite these technologies, acquiring personal data in order to optimize utility, and then sharing it on a large scale, is still very challenging. Thanks to the rapid development of artificial intelligence (AI), a relatively new and promising solution to resolve these challenges is to generate synthetic data (SD) by mirroring the original dataset's properties. SD is a promising solution to address growing privacy demands as well as the utility/analytics requirements of many industry stakeholders. In this paper, we propose and implement an SD generation method from a real dataset containing both numerical and categorical attributes by using an improved conditional generative adversarial network (CGAN), and we quantify the feasibility of SD on technical and theoretical grounds. We provide a detailed analysis of SD in original and anonymized forms with the help of multiple use cases, whereas prior research simply assumed that privacy issues in SD are small because AI models do not overfit or SD has a poor connection with real data. We provide insights into the characteristics of SD (distributions, value frequencies, correlations, etc.) produced by the CGAN in relation to the real data. To the best of our knowledge, this is the pioneering work that provides an experiment-based analysis of the quality, privacy, and utility of SD in relation to a real benchmark dataset.

Index Terms—synthetic data, generative adversarial networks, privacy, utility, federated learning, privacy-enhancing technology.

1 Introduction

Synthetic data (SD) have become a mainstream privacy enhancing technology (PET) of modern times, that can offer reliable analytical results, just like real data [1]. SD can be produced with machine/deep learning methods or mathematical models by simulating the distributions and structure of real data. SD is a main pillar in health and well-being fields for validating hypotheses and creating new hypotheses related to biomedical research [2]. In the modern era, SD offers many ways to advance science and influence societies, particularly in the era of big data and AI [3]. We highlight the importance of SD in real-world applications in six different ways. (i) SD can be published widely, which may not be the case with real data owing to many privacy issues and legal measures (e.g., the GDPR in Europe) [4]. (ii) SD can enhance the results of AI models by increasing the size, diversity, and quality of training data. (iii) SD has abilities to solve the data island problem (a deficient result from AI models owing to greater disparities in distributions and sizes of data at each site) [5]. This problem was experienced by most hospitals amid the COVID-19 pandemic. (See Fig. 1.) (iv) SD can enable early access when the actual data cannot be retrieved due to unexpected circumstances (taking COVID-19 as an example), or there are too few [6]. (v) SD can be produced in multiple formats, such as electronic health records, medical images, biomedical signals, a time series, or activity data, which can be imperative for analytical tasks, research, or policy-making. (vi) It can become a mainstream PET (like FL) that does not access the real data but still allows analytics of distinct types.

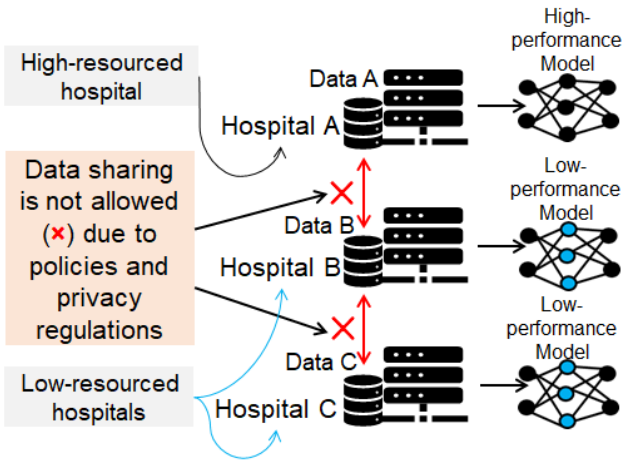


Fig. 1: Illustration of the data island problem (DIP).

1.1 Motivation and Novelty

Andrew Ng coined the term data-centric artificial intelligence (DC-AI) to develop high-quality and reliable AI systems [7]. DC-AI gives a higher preference to data quality rather than to the code of the AI models. Since the inception of the DC-AI concept, many breakthroughs have been achieved in multiple domains. Hedge achieved 100% accuracy in anomaly detection problems using the DC-AI [8]. Motamedi et al. [9] developed a DC-AI approach to

Abdul Majeed and Seong Oun Hwang are with the Department of Computer Engineering, Gachon University, Seongnam 13120, South Korea.
Manuscript received April 19, 2005; revised August 26, 2015.

training a deep neural network with as little data as possible without degrading accuracy much. Recently, DC-AI-based approaches have contributed significantly to solving longstanding industrial problems [10]. However, in some domains, good-quality data can be absent (or cannot be curated owing to small budgets). To compensate for a deficiency of good data, SD of good quality is vital and has become an integral component for data quality enhancement in AI applications. However, when the quality of real data is poor (incomplete records, skewed distributions, outliers, etc.), generating SD is challenging and can lead to many performance bottlenecks in the generative models. To the best of our knowledge, no framework deals with such problems (e.g., generating SD when the quality of real data is poor), which is the main motivation behind this research. Although many generative methods have been proposed that can create synthetic data from real data, certain challenges remain unaddressed: (i) accurate modeling of both numerical and categorical attributes, (ii) preventing model collapse during simultaneous training of two neural networks, (iii) quantifying the level of privacy versus utility from the SD, (iv) accurately reflecting in the SD all values of attributes from real data, regardless of status (i.e., minor, super-minor, major, and super-major), and (v) preventing imbalanced learning while generating SD.

Despite promising applications, research on SD generation is still in its infancy. Some studies only advocate the use of SD for research purposes [6]. Few studies have evaluated the credibility of SD generated with different AI tools [11]. Some studies have explored cases where SD can act as a replacement for real data in the healthcare sector [12] or in ranking applications [13]. Some studies described different ways to quantify the utility of SD [14], [15]. However, most studies have ignored privacy issues that can emerge from SD when an AI model overfits or when the similarity between the SD and the real data is high. Furthermore, deeper insights on utility in terms of query results, association rules, and value distributions have not been reported. Also, generating high-quality SD (i.e., data that preserves most characteristics of the original data, and that encompasses all attributes, values, and distributions to the fullest extent possible) in the presence of mixed-type attributes in the real data has not been thoroughly investigated. The main contributions of this research are as follows.

- We analyze the performance bottlenecks caused by poor-quality data in generative models, and we identify opportunities to develop a conditional generative adversarial network (CGAN)-based model for generating high-quality tabular SD to fulfill the growing need for privacy, as well as data utility/analytics.
- Our proposed method does not require data conversion from one form to another, and can correctly mirror the properties of the real data, regardless of the nature of the attributes (e.g., categorical or numerical) and the number of columns, whereas existing methods often require data conversion (numerical $\leftrightarrow$ discrete) and yield a higher offset from the real data.
- We introduce various new methods in the CGAN model to address major performance bottlenecks, such as data transformation issues, neural network model collapse, imbalanced learning, and reflecting all values and distributions in the synthetic data.
- We highlight the significance of SD in original and anonymized forms through experimental analysis that remains unexplored in the recent literature (most methods ignore the possibility of privacy breaches from SD based on the assumption that AI models do not overfit, or that similarities among SD and real data are not significantly high) [14].

Extensive experiments were conducted on a real-life benchmark dataset with the help of different use cases to verify the feasibility of the proposed method from both technical and theoretical points of view. The results and comparisons indicate better performance from the proposed method in SD generation, compared to existing SOTA methods.

The remainder of this paper is organized as follows. Section 2 presents related work, and Section 3 presents some preliminaries (e.g., data representations, the GAN architecture, the system model) and the problem formulation. Section 4 demonstrates the proposed method employed to generate high-quality data by leveraging real data. Section 5 evaluates the results obtained from experimentation on a real-world benchmark dataset. Section 6 summarizes the important findings of this research and discusses the implications. We wrap up the paper in Section 7.

2 Related work

SD has been increasingly used in modern times to fulfill deficiencies in good data, and many techniques have been developed to curate SD. Abay et al. [16] evaluated the utility of SD (T_s), obtained from real data (T), by using various techniques. The authors also developed a DL-based technique to generate T_s that can provide considerable privacy. Li et al. [17] employed a variational autoencoder (VAE) to produce T_s that can maintain an equilibrium between privacy and utility. The VAE proposed in their study outperformed the plain AE in terms of accuracy. However, this method cannot yield desirable performance when data are highly skewed (i.e., the distributions of values in some columns are not uniform). Wang et al. [18] developed a low-cost method named PART-GAN for time series data sharing and augmentation. PART-GAN has abilities to provide considerable privacy and can generate unlimited amounts of data. The authors evaluated PART-GAN on medical datasets for both privacy and utility. Martinsson et al. [19] developed a privacy-preserving method by adding new random information in data. By injecting random information, it offers robust privacy guarantees against strong adversaries. Torfi et al. [20] developed differential privacy combined with a convolutional AE (CAE) and convolutional GAN (CGAN) method to generate T_s. The CAE-CGAN can capture salient features and temporal interactions from T. Appenzeller et al. [21] evaluated three different SD generation methods, and explored various promising use cases of SD in real-life scenarios by amalgamating DP with the CGAN.

Recently, SD has contributed to the medical domain by helping doctors to devise new treatment modalities and to understand the dynamics and clinical properties of different diseases. Elaraby et al. [22] developed a CGAN-based data augmentation approach to enhance performance from

transfer learning in handwritten character recognition tasks. The proposed approach helped reduce the # of epochs and the generalization error when used with three distinct DL models (GoogLeNet, AlexNet, and VGG-16). Lu et al. [23] devised an improved version of the GAN for enhancing accuracy in anomaly detection by enhancing the sample quality as well as stability when training under realistic scenarios. Hammami et al. [24] developed a CycleGAN and YOLO-based technique to detect multiple organs in CT images, which enables robust detection of organs, and has many applications in medical imaging. Zhang et al. [25] developed the RFI-GAN model for generating medical images of good quality. The authors addressed the issues of variable scaling ratios and multiple sections in ultrasound images to prevent illogical image generation.

In recent years, many generative models have been proposed to curate tabular data to be used with ML models for classification and regression tasks [26]. MedGAN [27] is the first state-of-the-art (SOTA) generative model in this line of work. MedGAN can generate SD encompassing high-dimensional discrete variables for medical purposes. TableGAN [28] is another notable development for SD generation in tabular form. TableGAN uses an extra classifier in addition to D and G. It has higher complexity from using the classifier and an additional loss function. CTGAN [29] was devised to process tabular data without modifying the structure, and has yielded promising results in SD generation. However, this model cannot produce SD of good quality when the quality of real data is poor. CW-GAN [30] was proposed as an enhancement of TableGAN to generate SD of good quality and enhance the predictive performance of classifiers. Recently, GANBLR [31], which combines Logistic Regression and Naive Bayes with a GAN-based model, was developed to enhance the interpretation of data. However, the GANBLR model has clear limitations in that it only processes categorical columns. Table 1 highlights the status and comparisons of SOTA generative models. Our proposed method can address the limitations in the existing models and can produce SD in a designated format. Our method can be directly applied to any real-world T, and format conversions are not needed. In addition, it can be applied to poor-quality data and still yields high-quality SD. Lastly, we provide privacy analysis of T_s that remains unexplored.

In the absence of sufficient data, data augmentation techniques are mostly used to complement minor parts of the data by curating more samples. Fig. 2 presents the workflow of the proposed technique and the most commonly used data augmentation techniques. Random undersampling (RUS) downsamples the majority class whereas the minority class is upsampled in random oversampling (ROS) [32]. The synthetic minority oversampling technique (SMOTE) curates more samples for the minority class by computing the distance between the samples of the minority class and by using the k-nearest neighbor strategy [33]. The k-means SMOTE generates new points in each cluster to compensate for a deficiency of samples in the minority class [34]. Fair SMOTE was developed to address the imbalanced learning problem in classifiers by curating more samples for some skewed attributes and target classes [35]. Recently, a new augmentation technique that increases the size of all classes by adding synthetically generated samples from

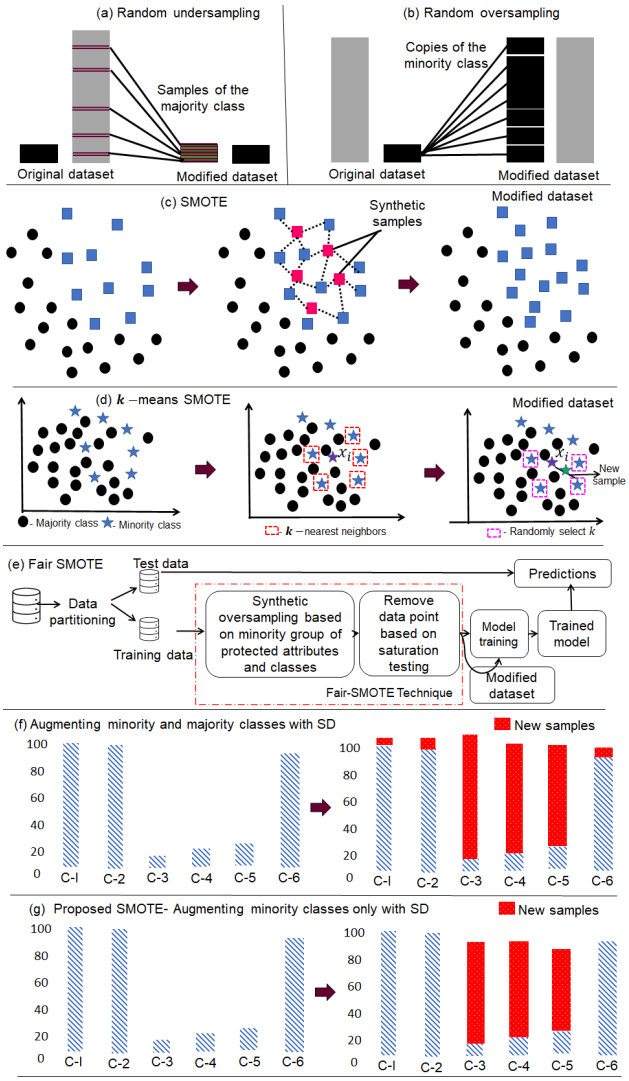


Fig. 2: Overview of the SOTA data augmentation techniques.

the GAN model was proposed [36]. That technique can balance the data, leading to better results than unbalanced data. In this paper, we devise a novel data augmentation technique that adds only required samples in the minority class, leading to better performance than existing SOTA techniques.

3 Preliminaries and Problem Formulation

In this section, we discuss tabular data representation and provide an overview of the GAN model. Later, we provide the system model considered in this work. Lastly, we formulate the problem to be solved with our proposed method.

3.1 Data representation

In this work, we focus on good-quality synthetic data generation to obtain T_s from real data T with the help of a powerful AI-based CGAN model. Let T be a real data table of dimensions $r \times c$, where c refers to the number of columns, and r denotes the number of rows. T consist of both N_c numerical columns, $\{c_1, c_2, \ldots, c_{N_c}\}$, and N_d categorical columns, $\{d_1, d_2, \ldots, d_{N_d}\}$. Each row in T contains full data

TABLE 1: Status and comparisons of SOTA generative models used to curate T_s.

Generative Model	Training method	Limitations
MedGAN [27]	GAN + Auto-encoder	Works well on discrete data only, and results are not generalizable.
TableGAN [28]	Classifier + GAN	No direct application, because data conversion (discrete → numerical) is needed.
CTGAN [29]	WGAN + Condition	Limited application when data size is small, and real data quality is poor.
CW-GAN [30]	Classifier+ WGAN + Condition	Higher offsets in numerical attribute values, and the training process is unstable.
GANBLR [31]	NB + LR	Good for discrete data only, and minor values are not properly reflected in the SD.

about a person, whereas a column contains attribute items (e.g., age/sex). A typical structure of T by using a sample of $20,000$ records is in Eq. 1:

$$T_{users,attributes} = \begin{pmatrix} u_i & a_1 & a_2 & a_3 \cdots & a_p \\ u_1 & v^1_{a_1} & v^1_{a_2} & v^1_{a_3} \cdots & v^1_{a_p} \\ u_2 & v^2_{a_1} & v^2_{a_2} & v^2_{a_3} \cdots & v^2_{a_p} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ u_N & v^N_{a_1} & v^N_{a_2} & v^N_{a_3} \cdots & v^N_{a_p} \end{pmatrix} = \begin{pmatrix} u_i & a_1 = age & a_2 = sex & a_3 = race & a_p = disease \\ 1 & 57 & F & White \cdots & Leukaemia \\ 2 & 34 & M & Black \cdots & Rhinitis \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 20,000 & 35 & F & Black \cdots & Cancer \end{pmatrix} \quad (1)$$

In Eq. 1, $a_1 \in N_c$, whereas $a_2, a_3, a_p \in N_d$. The position of N_c and N_d can be distinct in each real-world T. Each c in T refers to a random variable, and row r_k where $r_k = \{c_{1,k}, c_{2,k}, \ldots, c_{N_c,k}, d_{1,k}, d_{2,k}, \ldots, c_{N_d,k}\}, k \in \{1, 2, \ldots, n\}$ is one complete observation. N_c and N_d can be encoded into different formats when used in a CGAN. Furthermore, T is partitioned into the test set T_{test} and training set T_{train}.

3.2 The generative adversarial network

A generative adversarial network contains two neural network modules: generator G and discriminator D. Training of both these modules is performed in an adversarial manner. Random noise (a.k.a. latent code), denoted as z, serves as input to G in order to produce samples with the approximate distribution of T_{train}. In contrast, D takes $G(z)$ produced by G and T_{train} as input, optimizing it to distinguish $G(z)$ from T_{train} as shown in Eq. 2.

$$\begin{aligned} G &: z \rightarrow G \rightarrow G(z) \\ D &: (G(z), T_{train}) \rightarrow D \rightarrow F/R \end{aligned} \quad (2)$$

In Eq. 2, z is the latent code that is fed into the G, and $G(z)$ denotes the synthetic data produced by the G against z. $G(z)$ (synthetic data) and T_{train} (real data) are both fed into the D for distinguishing real samples from fake ones. F and R denote the fake and real samples, respectively.

During training, both G and D try to outperform each other. The objective function of the GAN is given in Eq. 3:

$$\begin{aligned} \min_G \max_D V(D,G) = E_{x \sim h_T(x)}[logD(x)] \\ + E_{z \sim h_z(z)}[log(1 - D(G(z)))] \end{aligned} \quad (3)$$

where h_T denotes the distributions of real data, and h_z is the distribution of latent code. During the update phase, the first item of the objective function forces D to yield higher scores with x, while the second item forces D to yield lower scores for z. Specifically, D wants to make the $G(z) \approx 0$, by doing so D can accurately distinguish between real and fake samples. In contrast, when updating G, the goal is to yield higher scores for D on z. With the help of joint training, T_s can be obtained for downstream tasks.

The two famous enhancements of the GAN are styleGAN and CGAN. The former is used for multimedia data, and the latter is used for tabular data. In this work, we use a CGAN to produce high-quality SD from T. The CGAN concept is the same as the original GAN, but condition y is attached to both modules. The y in CGAN carries additional information, such as labels for N_d, T modality, class labels, and/or network parameters. Due to the addition of y, the objective function of a CGAN will be slightly modified, as shown in Eq. 4:

$$\begin{aligned} \min_G \max_D V(D,G) = E_{x \sim h_T(x)}[logD(x|y)] \\ + E_{z \sim h_z(z)}[log(1 - D(G(z|y)))] \end{aligned} \quad (4)$$

where y denotes the condition in the CGAN. The CGAN converges when both G and D reach a Nash equilibrium.

3.3 The system model

In this paper, we consider a data-driven scenario/application with multiple actors (record owners, data owners, data analysts, and an adversary), as shown in Fig. 3. For analytical/data-mining purposes, analysts are interested in a fine-grained T from data owners who gather a huge amount of data from individuals. The data owners (hospitals, banks, insurance companies, etc.) are reluctant to share individual data in pure form due to privacy risks, even if analysts are honest. See "a" in Fig. 3. Another solution to this deadlock between data owners and analysts is to share data in an anonymized form. In this case, a copy of the data can still be obtained by an adversary who might be able to compromise the privacy of a user (or a set of users) by linking the data to auxiliary sources. See "b" in Fig. 3. This method cannot contribute to responsible data science, and data owners may hesitate to share anonymized data if, time and again, explicit privacy issues occur from the release of anonymized data. A relatively new solution to this problem is to mirror the properties of the original data and generate a copy similar to the real data by using AI methods ($AI\ algorithms \rightarrow data$) as illustrated with "c" in Fig. 3. By protecting privacy, we can explore opportunities to make use of AI-based methods to address the issues of data sharing on a wide scale. Our goal is to implement a practical method that can generate synthetic data of very good quality for analysts by learning the attributes and their distributions from the original data. We keep the design of the proposed method as flexible as possible to allow analysts to choose the amount of data they wish to generate to meet the growing demand for analytics (or data sharing) while respecting privacy issues. We assume the real dataset has already been collected from relevant users.

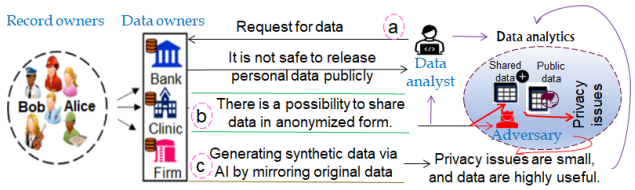


Fig. 3: Overview of system model considered in our study.

3.4 Problem formulation

Consider real-world dataset T where $T = r \times c$. In T, label c_i for the ith column can be either categorical or numerical, denoting a piece of information about the subjects. Let us say a_i is N_c, and another attribute, a_j, is N_d. The format of any row r is as follows:

$$r_x = \{a_i \in N_c, a_j \in N_d, a_k \in N_c, \ldots, a_p \in N_d\} \quad (5)$$

Considering the mixed data types in T, generating a mix of continuous and categorical columns in T_s is challenging. For example, one needs to use both $tanh$ and $softmax$ functions of a GAN on the output to convert $T \rightarrow T_s$. Secondly, numerical values in c can have different cardinalities, which cannot be denoted with $[-1, 1]$ in most cases, and if denoted can lead to the vanishing gradient problem in AI methods. Therefore, transformations are inevitable while using a numerical c of T, which can be tricky in most cases. Third, modeling all modes of continuous columns while transforming $T \rightarrow T_s$ is very challenging. Lastly, some columns in T can have very skewed distributions (e.g., one value can be 80%/90% of T). In such cases, minority classes can be overlooked by G while converting $T \rightarrow T_s$. Consider a T that contains numerical, categorical, and/or mixed attributes. It poses distinct challenges in its conversion to replicated form because of a large distribution of values, mixed-attribute types, multimodal distributions, and non-Gaussian-like distributions. Let T_s be the SD made from T for the downstream task. With T_s, the privacy breach issue can be reduced/resolved to a greater extent compared to T; ML models can be trained, and/or T_s can be distributed on a large scale. However, proving the efficacy of T_s in real-world decision-making scenarios may be subject to bias and inappropriate conclusions. The chosen problem to be formulated is: *how to convert T with mixed attributes (e.g., where categorical attributes can have skewed distributions, and where numerical attributes can have multimodal distributions) into T_s in an automated way by rigorously replicating joint distributions from T such that the conclusions or analysis results drawn from T_s are aligned with, or are approximately similar to, T while the structural similarity between T and T_s remains high. Furthermore, any legitimate information-consumers or data-analysts who see T and T_s should not be able to distinguish between the real data and the synthetic data without prior knowledge*. In most data-driven applications, analysts are mostly concerned with results (extracting pictures from the data). Hence, as long as T_s fulfills this requirement, it can be used as a proxy for T in most data-driven applications.

4 Synthetic Data Generation by Leveraging a Conditional GAN Model

In this section, we discuss how to generate a high-quality T_s by mirroring T using a CGAN (a powerful AI tool). The major steps in this process are data representation/transformation of numerical and categorical columns, designing a fully connected network structure, model training, loss function updating, and T_s generation by combining hot vectors and a ReLU. Fig. 4 presents a schematic of the proposed CGAN model used to generate SD of high quality. The key reason to use the CGAN model is to enable SD generation even when the real data have highly skewed distributions. In many real-world datasets, a single value in discrete columns might account for 80-90% of the data, with the rest having many diverse values. In these circumstances, traditional GAN models cannot generate SD of good quality, and likely face an imbalanced learning situation where only the majority value will be properly learned. The proposed model prevents such situations by imposing a condition on the values in discrete columns, and therefore, the quality of the SD is better. Furthermore, our model applies to any dataset, skewed or balanced. Table 2 shows the key notations in our T_s generation method.

TABLE 2: Main notations used in the proposed method.

Symbols	Description
T, T_s, x	Real data, synthetic data, sample of real data
N_c, N_d, c_i	Numerical columns, categorical columns, ith column
G, D, con	Generator, discriminator, condition of the CGAN
z, $T \rightarrow T_s$	Latent code or prior noise, T's conversion to T_s
$\text{gumbel}_\tau(x)$	Gumbel softmax on vector x with parameter τ
$x_1 \oplus x_2, \ldots$	Concatenation of vectors x_1 and x_2
$\text{FC}_{u \rightarrow v}(x)$	Linear transformation to get v from u
$\text{leaky}_\gamma(x)$	Leaky ReLU activation with ratio γ on x

From Fig. 4, there are three key parts in the proposed CGAN model. A concise description of each is given below.

I. *Column representation and transformation:* In this part, pre-processing is applied to the data, and numerical and categorical columns are converted to their respective representations. Later, the categorical columns are represented with hot vectors, whereas numerical attributes are transformed into modes and normalized values using the variational gaussian mixture (VGM) model. In addition, data are cleaned of vulnerabilities (e.g., missing values, outliers, inconsistent values) to prevent unnecessary processing and to generate a T_s that correctly mimics the properties of T. Technical details on this part are given in Section 4.1.

II. *Structure of the network:* In this part, the structure for both G and D of the network is established. Specifically, we use two fully connected networks with a PacGAN strategy in D to maintain stable training and prevent imbalanced learning. In addition, a ReLU and a Leaky ReLU are used as activation functions. Some other components, such as batch normalization, dropouts, data extractors (i.e., Gumbel, softmax), and tanh also contribute to the network structure. Technical details are given in Section 4.2.

III. *Training process and generation of a T_s using T:* Here, a condition is established concerning discrete columns that give useful hints to G and D that prevent imbalanced learning. The two models are trained in parallel by respecting

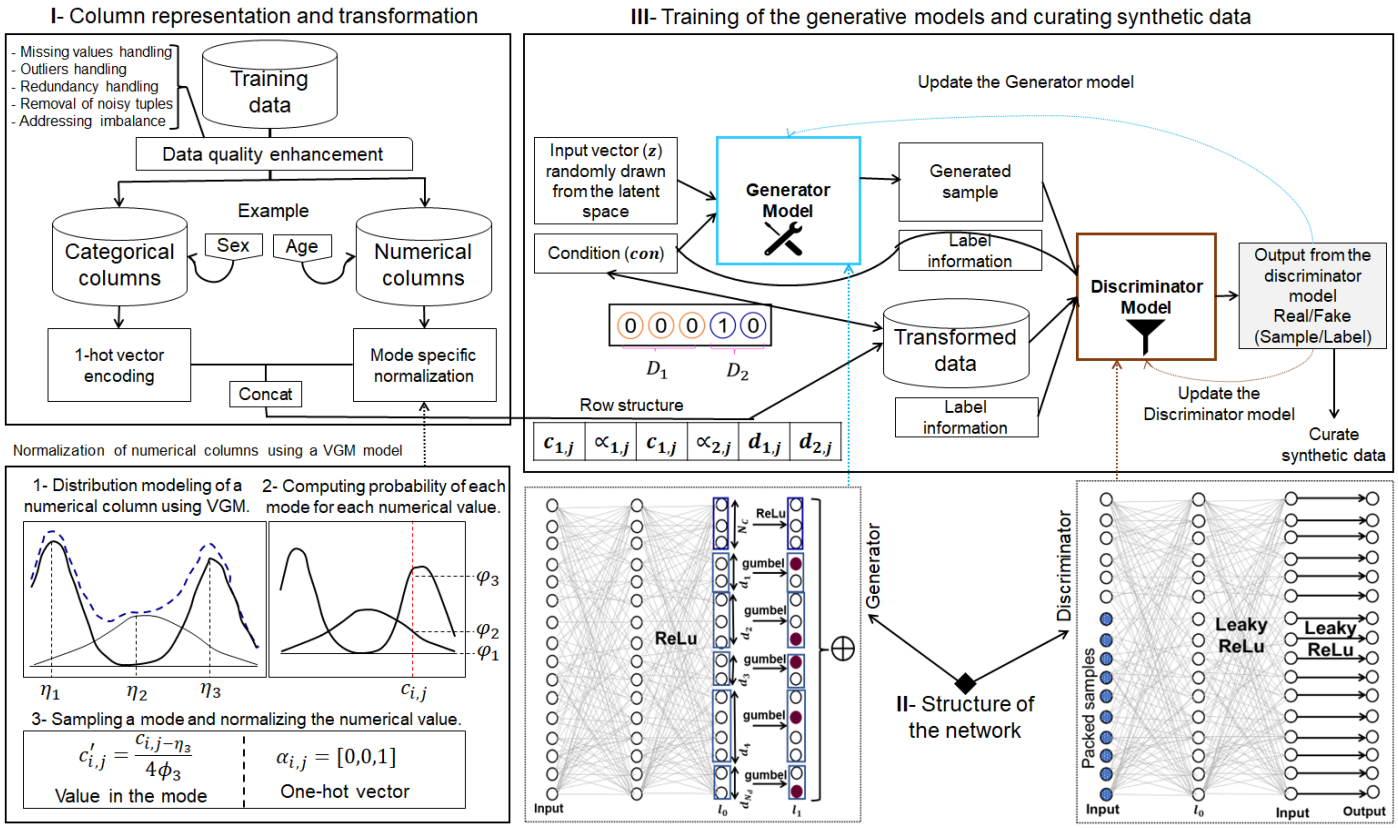


Fig. 4: Schematic of the proposed CGAN model for generating high-quality synthetic data.

the condition established. After training, a T_s that closely represents T is obtained, which is used for downstream tasks. Technical details of this part are in Section 4.3.

4.1 Column representation and transformation

In this section, we first discuss the modeling and transformation of the columns. Subsequently, we discuss the mechanism used to establish the condition that improves the learning abilities of the generator.

4.1.1 *Numerical column modeling and transformation*

The structure of any row, r_x, in T can be expressed as

$$T : r_x = \{a_i \in N_c, a_j \in N_d, a_k \in N_c, \ldots, a_p \in N_d\} \quad (6)$$

As shown in Eq. 6, any column can be either N_c or N_d. Both types pose distinct challenges for representation in GAN environments. Previous research used one of several methods (e.g., min-max normalization, quantile transformation, normalization) in order to transform and represent N_c. In this work, we use mode-specific normalization to address multimodal distribution issues in the CGAN. In contrast, we transform and represent N_d with hot vectors to effectively resolve imbalance issues. As stated earlier, N_c cannot be bounded by [-1,1], and distributions can be highly complicated. To address this challenge, we used a VGM model [37] to transform N_c and accurately represent their values. The three-step process used to model each numerical column is shown in the bottom left portion of Fig. 4. A concise description of each step is given below.

- *Distribution modeling of a column*: Given numerical column C_i, the VGM model fits a Gaussian mixture and estimates the number of modes, m_i, in C_i. As shown in Part 1 in the bottom half of Fig. 4's Section I, the VGM finds three modes denoted with η_i, where $i = 1, 2, 3$. The learned GM is given in Eq. 7:

$$P_{C_i}(c_{i,j}) = \sum_{l=1}^{3} \mu_l N(c_{i,j}; \eta_l, \phi_l) \quad (7)$$

  where ϕ_l denotes the standard deviation of a mode, and μ_l denotes the weight of the respective mode.
- *Computing probabilities of the modes*: After finding the modes in C_i, the probability of each value (say, $c_{i,j} \in C_i$) concerning each mode is computed. As shown in Part 2 in the bottom half of Fig. 4 Part I, three probability densities are φ_1, φ_2, and φ_3. Each φ is computed using Eq. 8:

$$\varphi_l = \mu_l N(c_{i,j}; \eta_l, \phi_l) \quad (8)$$

- *Sampling a mode and normalizing the value*: Pick one mode from the given φ_l values, and normalize value $c_{i,j}$. As shown in Part 3 in the bottom half of Fig. 4 Part I, if φ_3 is chosen as the sample mode, its value can be expressed in the form of a one-hot vector, $\alpha_{i,j} = [0, 0, 1]$. The scaler value, $c'_{i,j}$, in the normalized form in m_3 can be obtained using the formula in Part 3 in the bottom half of Fig. 4 Part I. The output of the three-step process is given in Eq. 9:

$$C_i : c_{i,j} = c^{'}_{i,j} \oplus \alpha_{i,j} \tag{9}$$

where $c^{'}_{i,j}$ is the normalized value in a mode, and $\alpha_{i,j}$ is the hot vector of the mode.

Although mode-specific normalization proposed in [29] yielded 25.70% better performance than the $min-max$ normalization technique, the resulting T_s had two key problems: (i) some values in the numerical columns were negative, even if the entire column in real data had all positive values, and (ii) the offset in some numerical values was significantly higher than in the real data. Furthermore, if the data size is very small, many values in the numerical columns of SD tend to lie outside the desirable range, leading to a higher loss of statistical information. In this paper, we address the above-cited limitations in the CTGAN model by improving the mode-specific normalization process, as well as choosing the optimal values for cluster size and weight threshold. Also, the decision process regarding the distribution selection and value representation is bounded by a new valid-component indicator in the numerical columns transformer to better approximate the real data. The final representation of row r after transformation of numerical columns is expressed in Eq. 10:

$$r_e = c^{'}_{1,j} \oplus \alpha_{1,j} \oplus c^{'}_{2,j} \oplus \alpha_{2,j}, \ldots, c^{'}_{N_c,j} \oplus \alpha_{N_c,j} \oplus d_{1,j} \oplus d_{2,j} \ldots d_{N_d,j} \tag{10}$$

where $d_{i,j}$ is the hot vector form of a categorical value.

4.1.2 Solution to class imbalance in categorical columns

Setting up a suitable condition in N_d can assist G to address class-imbalance problems to generate a T_s that is very close to T. Without conditions, G can ignore minor categories in N_d, and the obtained T_s offers poor utility. Let v^* be the value from the k^*th column and row r_i in a categorical column (i.e., D_{k^*}) to be matched to a new generated sample denoted r_s. In this case, G can be perceived as a conditional distribution of r_i in the k^*th column, which can be formally expressed in Eq. 11:

$$r_s \sim P_G(r_i | D_{k^*=v^*}) = P(r_i | D_{k^*=v^*}) \tag{11}$$

The integration of such a condition forces G to produce high-quality data. By enforcing the condition, original distributions can be reconstructed as formalized in Eq. 12:

$$P(r_i) \sum_{v \in D_{k^*}} P_G(r_i | D_{k^*=v^*}) P(D_{k^*=v}) \tag{12}$$

The conditions (denoted with con) are incorporated in the CGAN implementation via hot vectors. As shown in Eq. 10, all N_d are transformed as hot vectors. Therefore, a conditional vector con is created for each categorical column. Thus, the overall transformation of categorical columns $D_1 \in T$ can be expressed as $(D_1 \rightarrow d_1 \rightarrow [d_1^{(v)}]$, where $v = 1, \ldots, |D_1|)$. For any kth categorical column, $con_k = [con_k^{(v)}]$ can be the masked vectors encompassing conditions for a particular value. Therefore, the condition can be formalized in Eq. 13:

$$con_k^{(v)} = \begin{cases} 1, & \text{if } k = k^* \text{ and } v = v^* \\ 0, & \text{otherwise} \end{cases} \tag{13}$$

Finally, conditional vectors, con, can be made: $con = con_1 \oplus con_2 \ldots con_{N_d}$. For example, for any two categorical columns, $D_1, D_2 \in$T), if we let $D_1 = race = \{white, black, others\}$ and $D_2 = income = \{\geq 50K, < 50K\}$, then the condition on D_2 for the first value can be reflected in mask vectors as $con_1 = [0,0,0]$ and $con_2 = [1,0]$, and overall con can be expressed as $con = [0,0,0,1,0]$.

During training, the conditional G can produce one-hot vectors for categorical columns each time. We denote the newly produced columns with $d_i^{'}$, where $i = 1, \ldots, N_d$. Conditions made in the form $(D_{k^*=v^*})$ and encoded in a con vector are enforced during the sample generation for any column $d_i^{'}$. Furthermore, losses are penalized by incorporating the cross-entropy between con_k^* and $d_{k*}^{'}$. After certain epochs, G learns the condition well and correctly reflects conditions in $d_i^{'}$. A critic is used to assess output quality Q_G of conditional G, which takes the difference between conditional distributions in T as well as the learned conditional distributions, which is mathematically expressed in Eq. 14:

$$Q_G = dist(P_G(r_s|con), P(r|con)) \tag{14}$$

where r_s and r are the learned row and the row of T, respectively.

Accurate representation of con and the training data is imperative to help the critic accurately compute the distance between T and T_s. In Fig. 5, we present an overview of the implementation of a conditional G used to explore all possible categorical values in C_i. The whole process can be implemented in six steps. (i) Create $|N_d|$ mask vectors filled with zeros initially, $m_k = m_k^{(v)}$, $k = 1, 2, \ldots, |D_k|$. (ii) Select categorical columns with equal probability (for example, if D_2 is selected, then $k = 2$). (iii) Compute the PMF of each value in a selected column; for example, in the income column, if $v^* = \geq 50K$ and it occurs 9000 times in total, then $PMF_{v^*} = log(9000)$. (iv) Let v^* be a randomly chosen value as per the PMF given in Step (iii), and if it is the first value in D_2, as shown in Fig. 5, then $v^* = 1$. (v) Update the respective components of a mask to 1 under the condition $m_k^{(v^*)} = m_k^{(\geq 50K)} = 1$; and (vi) Calculate the con vector based on the process explained above; for example, $con = [0,0,0,1,0]$, is the condition imposed on the first value of D_2. By using the six-step process, all categorical attributes can be evenly sampled to assist in producing a T_s with representations of all values.

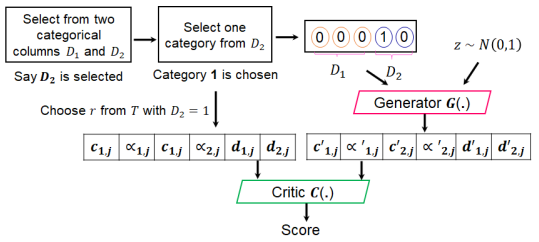


Fig. 5: Overview of a conditional G that explores all possible categorical values in columns for accurate data modeling.

4.2 Structure of the network

In order to generate T_s from T using a CGAN, we used fully connected networks (with two fully connected hid-

den layers) in both G and critic C (from here on, C is an alternate term for discriminator D). In G, the ReLU activation function and batch normalization are employed. The true representation of r_s is obtained from two hidden layers via a mix function (i.e., an activation function). Scaler numerical values, $c_i^{'}$, are obtained using the $tanh$ function. In contrast, α_i (the mode indicator) and categorical values $d_i^{'}$ are produced via $gumbel\ softmax$. In C, dropout and Leaky ReLU are used in both hidden layers. The overall structure of conditional G with z and con can be formally expressed in Eq. 15:

$$\begin{cases} l_0 = z \oplus con \\ l_1 = l_0 \oplus Relu(BN(FC_{|con|+|z|\rightarrow 256}(l_0))) \\ l_2 = l_1 \oplus Relu(BN(FC_{|con|+|z|+256\rightarrow 256}(l_1))) \\ c_i^{'} = tanh(FC_{|con|+|z|+512\rightarrow 1}(l_2)) & 1 \leq i \leq N_c \\ \alpha_i = gumbel_{0.20}(FC_{|con|+|z|+512\rightarrow m_i}(l_2)) & 1 \leq i \leq N_c \\ d_i^{'} = gumbel_{0.20}(FC_{|con|+|z|+512\rightarrow |D_i|}(l_2)) & 1 \leq i \leq N_d \end{cases} \tag{15}$$

In C, we employed PacGAN [38] to avoid model collapse with 10 samples. The overall architecture of C with a pac-size of 10 is given in Eq. 16:

$$\begin{cases} l_0 = r_1 \oplus r_2 \ldots r_{10} \oplus con_1 \oplus con_2 \oplus \cdots \oplus con_{10} \\ l_1 = drop(leaky_{(0.20)}(FC_{10|r|+10|con|\rightarrow 256}(l_0))) \\ l_2 = drop(leaky_{(0.20)}(FC_{256\rightarrow 256}(l_1))) \\ C(.) = FC_{256\rightarrow 1}(l_2) \end{cases} \tag{16}$$

To prevent the overfitting and memorization issues, we used optimized values for most parameters, i.e., the Adam optimizer with a 2×10^{-4} learning rate, the α value in the Leaky ReLU set to 0.2, βs=(0.5, 0.9), and a Dropout (0.5) where the ideal dropout value is between 0.4 and 0.7. With the help of these hyperparameters and their optimized values, the training process was stable, and overfitting issues did not occur. Lastly, due to the optimized network structure, the time overheads were small, and memory runout issues also didn't occur.

4.3 Training process and generation of a T_s using T

In this subsection, the training process is discussed with the loss updating mechanism for G and C. We performed the training using Wasserstein GAN (WGAN) loss with a gradient penalty [39]. We employed the Adam optimizer at the optimal learning rate (i.e., 2×10^{-4}) to produce a high-quality T_s. During training, the loss weights are updated until the convergence of the conditional GAN model. The generic form of C is Eq. 17:

$$L_C = E_{x^{'} \sim P_{T_s}}[C(x^{'})] - E_{x \sim P_T}[C(x)] + \lambda E_{\bar{x} \sim P_{\bar{x}}}[(||\nabla_{\bar{x}} C(\bar{x})||_2 - 1)^2] \tag{17}$$

where P_{T_s} and P_T are the synthetically generated and the original data distributions, respectively. The third term denotes the gradient penalty, and λ is the co-efficient of the gradient penalty. $P_{\bar{x}}$ is the sampling implicitly defined along a straight line between distributions P_T and P_{T_s}. The generic loss function of G during training is expressed as

$$L_G = E_{x^{'} \sim P_{T_s}}[C(x^{'})] \tag{18}$$

The complete procedure applied to produce T_s from T using the conditional GAN is formally expressed in Algorithm 1. In this algorithm, we describe the main tasks performed to transform data from one form to another. Besides the pseudocode in Algorithm 1, we specify the URL of the real data, the labels of categorical columns, the number of samples a data analyst wants to generate, and the epoch strength from the interface program. The initial learning and decay rates, the dimensions of C and G, and the frequency values are specified in the core Python functions. After thorough implementation and compilation of the program, different versions of T_s with varying tuples are obtained for quality, privacy, and utility evaluation.

Algorithm 1 Training the CGAN to convert T into T_s.

Require: T (Original data (a.k.a. training data)), m (batch size), pac (pac size), Φ_G and Φ_C (parameters of conditional G and critic C), and $|T_s|$ (# of tuples to be curated).
Ensure: Updated parameters Φ_C and Φ_G and T_s

1: T quality enhancement to reduce redundant operations
2: Generate masks for d columns,$\{m_1.m_2, \ldots, m_{N_d}\}_j$
3: $con \leftarrow con_j$, for $1 \leq j \leq m$
4: Sample $z_j \sim$ MVN (0, I), for $1 \leq j \leq m$
5: $r_j^{'} \leftarrow G(z_j, con_j)$, for $1 \leq j \leq m$ ▷ Yield synthetic data
6: Sample $r_j \sim Uniform(T|con_j)$, for $1 \leq j \leq m$
7: $con_v^{(pac)} \leftarrow con_{v \times pac+1} \oplus \cdots \oplus con_{v \times pac+pac}$
8: $r_v^{'(pac)} \leftarrow r_{v \times pac+1}^{'} \oplus \cdots \oplus r_{v \times pac+pac}^{'}$
9: $r_v^{(pac)} \leftarrow r_{v \times pac+1} \oplus \cdots \oplus r_{v \times pac+pac}$
10: $L_C \leftarrow \frac{1}{m/pac} \sum_{v=1}^{m/pac} C(r_v^{'(pac)}, con_{(v)}^{pac}) - \frac{1}{m/pac} \sum_{v=1}^{m/pac} C(r_v^{(pac)}, con_{(v)}^{pac})$
11: Sample φ_1 , $\varphi_2, \ldots, \varphi_{m/pac} \sim$ Uniform (0,1)
12: $\bar{r}_v^{(pac)} \leftarrow \varphi_v r_v^{'(pac)} + (1 - \varphi_v) r_v^{(pac)}$
13: $L_{GP} \leftarrow \frac{1}{m/pac} \sum_{v=1}^{m/pac} ((||\nabla_{\bar{r}_v^{(pac)}} C(\bar{r}_v^{(pac)}, con_v^{(pac)})||_2 - 1)^2)$
14: $\Phi_C \leftarrow \Phi_C - 0.0002 \times adam(\nabla_{\Phi_C}(L_C + 10 L_{GP}))$
15: Generate $r_j^{'}$ again using lines 4–10.
16: $L_G \leftarrow -\frac{1}{m/pac} \sum_{v=1}^{m/pac} C(\bar{r}_v^{(pac)}, con_v^{(pac)}) + \frac{1}{m} \sum_{j=1}^{m} CE(d_i^{'}, m_i, j)$
17: $\Phi_G \leftarrow \Phi_G - 0.0002 \times adam(\nabla_{\Phi_G} L_G)$
18: Update the C and G parameters in a feedforward way.
19: Return Φ_G, Φ_C, T_s ▷ Required outputs

4.4 Analysis of various methods used in CGAN model

In the proposed SD generation model, various methods have been used to obtain good-quality SD. The important methods used in the model that have a greater impact on performance are summarized below.

1) The *conditional vector* in the CGAN model is necessary to guide G regarding class labels or values of the particular attributes to be generated. In this work, the conditional vector is used to guide G on labels for discrete columns and to generate synthetic tuples that are conditioned on one of the columns of a discrete type. By imposing the *conditional vector*, all values are evenly sampled from the real data.
2) The *VGM model* is used to address multimode issues of the N_c. For example, there can be multiple types of series in a numerical column, and to ensure balanced learning, they need to be represented in a consistent format. By multiple types of series, we mean a group of values (e.g., series) that can differ

from another group/series based on the # of digits (e.g., single-digit versus two/more digits) as well as the gap (small versus large) in values. To this end, VGM first computes the modes of the N_c. Later, standard deviation and mean values for each of the modes are determined. Afterward, the values of the numerical column are normalized with the associated standard deviation and mean. The resulting normalized value is then concatenated with hot vectors of the categorical columns to prepare input for the CGAN model. Specifically, it assists in modeling non-Gaussian distributions in N_c.

3) The *PacGAN strategy* is used to prevent mode collapse during training of the networks. Most GAN models are vulnerable to mode collapse when working with attributes having skewed and multimodal distributions, and as a result, T_s can have a lot less diversity. To fix this issue, PacGAN modifies the discriminator structure to make final decisions with the help of multiple samples drawn from the same class, either artificially generated or real. This paper makes use of PacGAN to address the issue of mode collapse during training.
4) *WGAN* loss is used to maintain stability during training and to ensure the condition of the Lipschitz constraint. Specifically, in this strategy, gradients are penalized at sample points to adhere to the Lipschitz constraint (e.g., the norm of the gradients is close to 1 everywhere) [39], [40]. This strategy has been widely used in the GAN model because it is superior to the weight-clipping technique.

All the above-cited methods are necessary to correctly transform data from one form to another when T has mixed attributes. The *VGM model* is no longer required if T contains all categorical columns. Similarly, the *conditional vector* can be omitted when all data are numeric. However, *WGAN* loss and *PacGAN* are more important than the others, and have a greater effect on data improvement and model stability.

5 Experimental Evaluation

In this section, we discuss and analyze the output of the CGAN with the help of multiple use cases. To prove the feasibility of the proposed concept, we conducted experiments with the Adult dataset [41], which is a benchmark for evaluating most privacy algorithms. In the Adult dataset, there are 32,561 records and 15 distinct attributes (final dimensions of $32,561 \times 15$), with values of N_c and N_d at 6 and 9, respectively. The sensitive information is income, and all other attributes are treated as non-sensitive. Table 3 presents concise details of the dataset used in the experimental evaluation. In Table 3, N and C refer to numerical and categorical, respectively.

The reason to use this dataset is that it contains a reasonable mix of both numerical and categorical attributes. Furthermore, we emulated real-world cases by accessing the dataset through a URL from data owner environments. Since the adult dataset is a benchmark in the privacy community, we used it to test the efficacy of our method. Furthermore, this dataset has skewed distributions for some attributes and is a good candidate to verify the significance of the proposed method. Furthermore, our method is highly flexible, meaning it can produce T_s from any real-world T.

TABLE 3: Details of dataset used in performance evaluation.

Attribute type	Attribute label(Type, Cardinality)
QIDs	Hours-per-week(N , 93), Capital gain (N , 91) Capital loss (N, 118), fnlwgt (N, 21, 648) Age (N, 74), Education no. (N, 16) Country/Citizenship (C, 41), Race (C, 5) Occupation (C, 14), Work class (C, 8) Education (C , 16), Relationship(C , 7) Marital status(C , 6), Gender (C , 2)
SA	Income(C, 2)

5.1 Experimental settings

The implementation of the CGAN model was done on a notebook having Intel Core CPU i5-3320M @2.60GHZ. The RAM size was 8GB. We implement our model using Python 3.9 (64-bit version) with the help of built-in libraries (torch, scikit-learn, torchvision, numpy, pandas, matplotlib, etc.). The relevant functions from each library were used to accomplish the task of data generation. For example, BatchNorm1d, LeakyReLU, Dropout, Module, ReLU, Sequential, etc. were leveraged from the torch.nn library. Similarly, the relevant functions for giving suitable structure to data were used in the form of separate Python scripts. The interface program has various inputs such as real data directory links, labels of columns that are non-numeric, the number of epochs to be used in training, tentative data templates, the number of synthetic records to be curated, and file designation to store SD. The interface program allows users to specify the number of records to be curated, making our implementation more flexible compared to existing generative models. The SD curated with our model can be either saved to a file for downstream tasks or sent to the ML pipelines for training/testing models. In the implementation setup, we first test the model under the default setting of different parameters (e.g., G and D learning rates, decay, dimensions, batch_size, etc.) and then optimize them to improve T_s quality. The activation functions used in the model were tanh, mix, and softmax. We generate T_s in a similar structure to T, and the order of attributes is also the same as T.

The evaluation of T_s was performed from five different perspectives: enhancing ML models performance, query accuracy (involving numeric, discrete, and hybrid attributes), pattern/rule mining, privacy and utility analysis, and general benefits. While comparing the performance of our model from the perspective of ML, we used accuracy (Eq. 19) as the main evaluation metric and compared the results with five SOTA generative models (MedGAN [27], TableGAN [28], CTGAN [29], CW-GAN [30], and GANBLR [31]) and six SOTA augmentation techniques (RUS [32], ROS [32], SMOTE [33], k-means-SMOTE [34], Fair-SMOTE [35], and CTGANSamp [36]). Furthermore, we also trained three different ML classifiers (DT, SVM, RF) to compare accuracy between T_s and T. We analyzed and discussed the impact of T_s on samples and confusion matrices. While comparing the performance of our model from the perspective of query accuracy, we created sixteen different aggregation/count queries by utilizing the information from T and analyzed

the performance of query answers from T_s by using $Supp$ metric (Eq. 21). Afterward, we compared the results with T to check the performance of T_s. Specifically, we used T as the benchmark to evaluate and compare the performance of T_s curated with our model. To perform privacy and utility analysis, the anonymized versions of T_s were created with k-anonymity model with different k, and privacy and utility results were compared with relevant baselines and T. To compute privacy leakage, we executed two well-known attacks on anonymized data created from T_s and T, and computed the D_r (Eq. 22). We compared the privacy leakage (i.e., D_r) from the anonymized version of T_s with T and a baseline method [15]. To perform utility analysis, we computed the IL with the help of DM (Eq. 23) metric and compared the results with T. While analyzing the general benefits of T_s, we analyzed the effectiveness of T_s on five different grounds as discussed in detail in Section 5.7.

5.2 Ablation study

Although the proposed CGAN model accomplished the task of T_s generation from T, and proposed implementations are highly optimized, some parameters can degrade its performance in real scenarios. For example, if the condition vector is removed from the CGAN, there is a decrease of about 20.09% in the learning ability of G, and some minor distributions were not learned properly. Therefore, the CGAN can no longer apply to T having skewed distributions. Similarly, if the mode-specific normalization module is replaced with $min-max$ normalization, there is a drop of about 27.10% in the quality of T_s, and the vanishing gradient problem occurs while training neural networks. We removed the PacGAN component and tested the performance of the CGAN model, and mode collapse occurred during training with a 0.26 probability in 10 different tests. When we removed both PacGAN and the conditional vector, there was a 40% drop in performance, and the CGAN was overfitting. We used the data without pre-processing, and there was a 6% drop in T_s quality. Furthermore, when we set loose values for the training hyperparameters (learning rate, α, β, dropout, etc.), the performance dropped by 39.89%. We set the value for the coefficient of the gradient penalty (λ) to 10, which is a feasible value for generating a good-quality T_s [39]. However, the different values of λ can have a different effect on the accuracy of the T_s. To validate the effects of λ on the accuracy of the T_s, we conducted experiments by varying the values of λ. The best accuracy was obtained with the RF model on T_s at 83.78% when λ=10. When λ=2,4,6,8, there was a drop in accuracy by up to 15.66%, 12.04%, 8.41%, and 6.02%, respectively. When λ=12, there was no change in accuracy, and the original value was sustained (e.g., accuracy= 83.78%). This analysis verifies the feasible value of λ for generating a T_s of good quality. With the optimal values of hyperparameters, and using all supportive modules, our CGAN model yielded more competitive performance than previous SOTA generative methods. Next, we assess the performance of T_s curated with the CGAN model from five different perspectives.

5.3 Analyzing the use of T_s from the ML perspective

The performance of any ML model depends highly upon the quality of the data. The better the quality, the better the performance of any ML model, and vice versa. To evaluate the utility of T_s from ML perspectives, we computed and compared accuracy values (using Eq. 19) with T and relevant baselines.

$$A = \frac{T_p + T_n}{|T|} \tag{19}$$

In Eq. 19, T_p is true positive, and T_n is true negative. $|T|$ shows the total records in the data. In the denominator, $|T|$ should be replace with $|T_s|$ while calculating A for T_s.

5.3.1 Comparison with existing SOTA augmentation algorithms and generative models

The T_s produced by generative models can be used in training ML classifiers as well as for statistical analysis. To verify the quality of T_s produced with our model, we fused T_s with T and verified the efficacy of augmented data in terms of classifier accuracy enhancement and sampling quality while training them. As discussed earlier, we used T_s to augment only the problematic portion of T, whereas existing data augmentation techniques simply double the data, leading to marginal improvements in results because the distributions of majority and minority classes are not balanced. To compute and compare accuracy, we used a random forest classifier, which is SOTA, and a popular ML classifier. We divided the data for training (2/3) and testing (1/3). Table 4 presents comparisons of accuracy between our method and existing SOTA data augmentation algorithms.

TABLE 4: Average accuracy: our method versus existing SOTA augmentation algorithms.

Data augmentation algorithm	% accuracy (avg. of 10 tests)
Default (without augmentation)	85.38
RUS algorithm	76.15
ROS algorithm	75.38
SMOTE algorithm	70.89
k-means-SMOTE algorithm	74.23
Fair-SMOTE algorithm	85.96
CTGANSamp algorithm	90.25
Our method	**100.00**

From the results, we can see that our method outperformed all algorithms in terms of accuracy. It is worth noting that these improvements came from adding more records in the minority class only (e.g., income is $>$ 50K). In addition, our augmentation method yielded superior accuracy using fewer trees. The main reasons for the perfect prediction accuracy are the alteration in the importance score of each feature and correlation enhancement with the target class. When data is imbalanced, the classifier does not explore the features of the minority class well and misclassifies them as the majority class. In contrast, when classifiers are trained on balanced/augmented data, some salient but difficult-to-learn features can easily be perceived in the training process, leading to better prediction accuracy [42]. Also, the main focus of data augmentation in ML is to improve the quality, diversity, and equity of data, which can enhance accuracy. Lastly, data augmentation increases the prediction accuracy by increasing the counts of examples which are common in feature space across classes but different in label.

Furthermore, the confusion matrices yielded by our method were more balanced compared to the existing SOTA augmentation algorithms, as shown in Fig. 6.

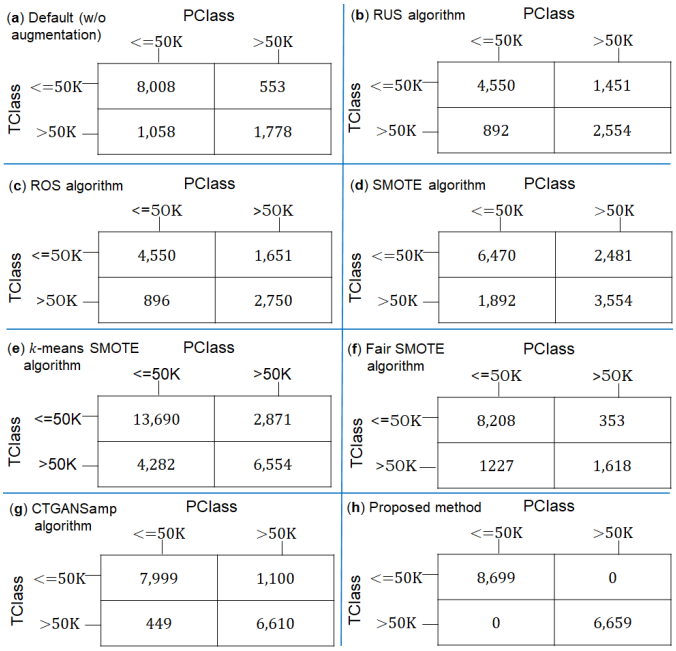


Fig. 6: Comparison of confusion matrices: proposed method vs. existing SOTA data augmentation algorithms.

In the next experiments, we compared the performance of our method in terms of sampling quality. Specifically, we drew samples from the augmented data and compared the differences between the frequency of the target class's values (e.g., ≤ 50K and > 50K). Since our method adds new records only to the minority class, the sampling quality of the generated data was much better, compared to the other data augmentation algorithms. Fig. 7 presents comparisons of sampling quality (e.g., the difference in frequency of target class values). From the results, we can see that our method has better sampling quality compared to existing algorithms, and the drawn samples are more balanced and fair. In contrast, existing algorithms have poor sampling quality, and differences are sufficiently large in some cases. These results fortify the significance of our method in terms of achieving better sampling quality, leading to the development of high-quality ML classifiers.

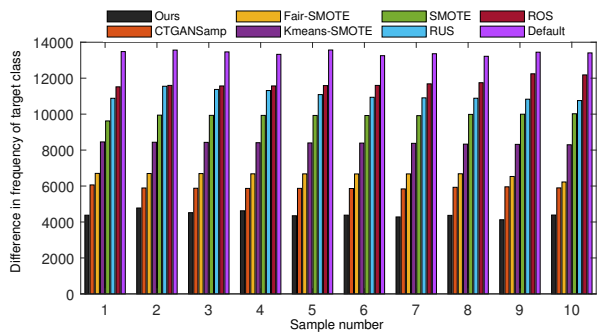


Fig. 7: Comparisons of sampling quality: proposed method vs existing SOTA data augmentation techniques.

To further justify the efficacy of the proposed model, we compared the results with five SOTA generative models in terms of ML utility. Specifically, we built RF classifiers with varying amounts of data and analyzed the accuracy. Table 5 presents a comparison of the results (an average of 10 tests). Although most methods yielded competitive performance in terms of accuracy, the quality of the underlying T_s was better for either categorical columns or numerical columns. Also, a greater contribution in accuracy came from the majority class only, and therefore, the T_s made by the existing generative models cannot be used in safety-critical applications. In contrast, our method generated a high-quality T_s with balanced distributions, leading to better performance in terms of ML utility.

TABLE 5: Accuracy comparison: proposed method versus existing SOTA generative models (avg. of 10 tests).

Generative Model	Accuracy (%)
MedGAN [27]	72.11
TableGAN [28]	78.33
CTGAN [29]	79.29
CW-GAN [30]	79.95
GANBLR [31]	81.21
Proposed CGAN	83.78

5.3.2 Accuracy comparisons between T, T_s, and $T + T_s$ by varying number of records

In this section, we evaluate and compare the accuracy between T, T_s, and $T + T_s$ by varying number of records. The obtained results are shown in Fig. 8. From the results in Fig. 8, we can conclude that the accuracy of T_s alone was lower than T, as (seen in Fig. 8 (left)). However, T_s can help augment the performance of AI models when jointly used with T, as shown in Fig. 8(right).

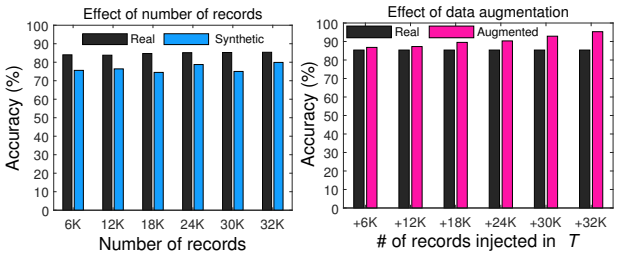


Fig. 8: Accuracy comparison of T_s, T, and $T + T_s$.

From the extensive experiments and comparisons, we found that T_s can enhance the accuracy of ML models by 9.92% if we mix T and T_s (e.g., $|T| + |T_s|$). After careful analysis, we found that mixing T_s into T can increase the strength of minority classes, leading to better prediction accuracy. The generality of the training process was also enhanced. We found that T_s can be a viable alternate (from the ML perspective) when T is available in a limited way.

5.3.3 Accuracy comparisons between T and T_s using different ML classifiers

To further justify the universality of results, we compared the results by using three different ML classifiers, as

shown in Table 6. From the results, it can be seen that the quality of T is vital for generating the best quality T_s.

TABLE 6: Accuracy comparison of T_s with T and a prior method using three different classifiers.

Dataset/study	DT (%)	SVM (%)	RF (%)	Avg. (%)
T (real data)	78.25	81.15	85.38	79.76
T_s (using raw T)	72.84	75.20	79.90	75.98
T_s (using improved T)	75.04	79.29	83.78	79.37

The main reason for the improvement was the selection of optimized values for hyperparameters. Experimental analysis certified that a good-quality T_s has many benefits in the era of AI. Through these experiments, we found that RF is a good choice for prediction/classification tasks. The results given in Tables 4-6 and Figures 8-6 demonstrate the efficacy of T_s in enhancing the performance of ML models. The comparisons with diverse generative models and augmentation methods prove the technical supremacy of our method. In addition, the accuracy results by varying data characteristics and utilization of different ML models verify the efficacy of T_s generated with our method for ML applications. Lastly, our method contributed to balanced sampling and confusion matrices, ensuring robust ML model development compared to previous methods. In the next subsection, we analyze and compare the quality of T_s from four different aspects, such as query answers, data mining, privacy and utility, and general perspectives.

5.4 Quantifying the significance of T_s from the perspective of query-based systems

In most data-driven applications, analysts usually execute different queries to perform the desired analytics without acquiring the whole T. By doing so, storage space is preserved, and most of the computation can be performed in the data owners' environments. To further evaluate the significance of T_s from the perspective of knowledge-based systems (i.e., query and answer), we executed multiple queries involving one, two, and multiple attributes, and compared the results from T and T_s. We devised these queries in the form of questions such as *What is the common profession of people in the Republic of China? Which profession is good in terms of earnings (i.e., income greater than 50K)? Which race value is most dominant in a dataset? Does gender inequity exist in terms of earnings? What is the best age for earning* $60K$ *a month?* We checked answers from both T and T_s, finding that T_s yielded consistent answers in most cases, and can be helpful in knowledge-based systems (e.g., query-based systems). The statements for queries and the tuples returned (or %) analyzed from both T_s and T are in Table 7.

We analyzed the error rate in terms of differences in the number of tuples, and we present the results for each query (in %) as (1, 64), (2, 39.5), (3,13.2), (4, 64.7), (5, 11.6), (6, 12.5), (7, 64.4), (8, 17.65), (9, 24.6), (10, 7.2), (11, 23.35), (12, 20.9), (13, 32.7), (14, 10.7), (15, 10.9), (16, 11). Through analysis of results, we noticed that queries executed on numerical attributes had relatively more errors than categorical ones. The largest difference in the number of tuples was observed in query # 4, while the least difference was observed in query # 10. In most cases, the differences were within an acceptable range considering other qualitative factors (especially the nature of the queries). These findings shed light on the significance of T_s in query-based systems that are most prevalent in a digitized society. Lastly, this analysis highlights another potential use of T_s in digital systems (e.g., count/aggregation query-answering).

TABLE 7: Descriptions of queries executed on both T and T_s and the corresponding results.

Query syntax/overview	Focus attribute	$\|r\|$ in T	$\|r\|$ in T_s
*SELECT * FROM T/T_s WHERE age < 20;*	Numerical	1,657	4,591
*SELECT * FROM T/T_s WHERE income = '>50K';*	Categorical	7,841	12,971
SELECT Race, COUNT(Race) As Most-Dominance FROM T/T_s GROUP BY Race ORDER BY COUNT (Race) DESC;	Categorical	White: 27816	White: 24139
*SELECT * FROM T/T_s WHERE income = '<50K' AND working-hours > 40;*	Hybrid	5,725	16,222
SELECT Country, COUNT(Country) As Most-Earnings FROM T/T_s WHERE income ='>50K' GROUP BY Country ORDER BY COUNT (Country) DESC;	Categorical	US: 29170	US: 25,765
SELECT age, SUM(income='>50K') AS low, SUM(income='<=50K') As high, COUNT (*) As Overall FROM T/T_s WHERE age BETWEEN 0 AND 90 AND Country='US' GROUP BY age;	Categorical	40^{+} Yrs.	35^{+} Yrs.
*SELECT * FROM T/T_s WHERE income = '>50K' AND working-hours > 40;*	Hybrid	3,856	10,860
SELECT gender, SUM (income='>50K') As low, SUM (income='<=50K') As Total-income FROM T/T_s Group By gender	Categorical	F: >50K → 10.9% & M: >50K → 30.5%	F: >50K → 11% & M: >50K → 40.5%
*SELECT * FROM T/T_s WHERE income = '>50K' AND education NOT IN (' Bachelors', ' Masters',' Doctorate')*	Categorical	Yes: 15.4%	Yes: 21.01%
SELECT profession, COUNT(profession) As Least-Earnings FROM T/T_s WHERE income ='<=50K' GROUP BY profession ORDER BY COUNT (profession) DESC;	Categorical	PHS: 99.32%	TS: 65.59%
SELECT profession, COUNT(profession) As Most-Earnings FROM T/T_s WHERE income ='>50K' GROUP BY profession ORDER BY COUNT (profession) DESC;	Categorical	EM: 48.5%	CR: 43.8%
SELECT COUNT (*) FROM T/T_s WHERE income ='>50K' AND education = 'Doctorate' AND country= 'US'	Categorical	77.9%	35.21%
SELECT gender, COUNT(gender) As Rich-Individuals FROM T/T_s WHERE income ='>50K' AND age ¿ 60 GROUP BY gender ORDER BY COUNT (gender) DESC;	Categorical	F: 11.07% & M: 31.2%	F: 37.9 % & M: 41.9%
SELECT profession, COUNT(profession) As Common-Profession FROM T/T_s WHERE country ='China' GROUP BY profession ORDER BY COUNT (profession) DESC;	Categorical	Professor: 23.3%	Professor: 12.4 %
SELECT COUNT (*) FROM T/T_s WHERE martial-status ='Married'	Categorical	47.3%	58.3%
SELECT COUNT (*) FROM T/T_s WHERE martial-status IN('Divorced', 'Never-married', 'Separated', 'Widowed')	Categorical	52.7%	41.7%

Abbreviations: F: Females, M: Males, EM: Executive-managerial, CR: Craft-repair, PHS: Private house servant, TS: Tech support, DESC=:descending order

5.5 Quantifying the significance of T_s from the perspective of data mining (e.g., pattern mining & analysis)

Analyzing data and drawing pictures from them is one of the most investigated topics, and is a mainstream solution for data-driven applications. Many companies around the globe are generating huge profits by finding insightful patterns from data [43]. To mine patterns from data, we executed different association rules from T_s and T, and compared support values (denoted as $Supp$). Rules can be made based on dominant quasi-identifier (QID) and sensitive attribute (SA) values. We present sample rule R_i in Eq. 20. In this rule, we analyze the correlation between two QIDs (education and country) with income (as an SA).

$$R_i = \{edu, country\} \rightarrow income = \{Ph.D., China\} \rightarrow > 50K \quad (20)$$

The $Supp$ value for any association rule (R_i) can be computed with Eq. 21:

$$Supp_{R_i} = f(QIDs,SA)/N \quad (21)$$

where f denotes the frequency of tuples in which the respective QIDs and the SA co-exist.

To evaluate the significance of T_s from the perspective of data mining (i.e., patterns), we executed multiple associations involving one, two, and multiple QIDs, and compared the results for $Supp$ between T and T_s. Since most data-driven applications use pertinent attributes for marketing/recommendation, we used only relevant QIDs in our analysis. Through this analysis, we evaluated the quality of T_s from the perspective of data mining.

TABLE 8: Association rules-based analysis: T versus T_s.

Association rule	$Supp$ in T	$Supp$ in T_s	Diff. (%)
$US \rightarrow > 50K$	0.25	0.41	0.17
$White \rightarrow \leq 50K$	0.74	0.59	0.14
$(Doctorate, China) \rightarrow > 50K$	0.06	0.01	0.04
$(Male, > 40Hrs.perweek) \rightarrow > 50K$	0.15	0.19	0.05
$(Exec-Mng, US, White) \rightarrow > 50K$	0.44	0.37	0.06
$(> 50Yrs., Divorced, Black) \rightarrow \leq 50K$	0.017	0.009	0.007

From the analysis presented in Table 8, we can conclude that the performance of association rules (e.g., $Supp$) computed from T_s was comparable with T. Interestingly, in some cases, the $Supp$ values from T_s were higher than from T. These findings indicate the utility of T_s from the perspective of marketing/recommender systems. Rules-based analysis can assist in validating a research hypothesis as well as in generating a new hypothesis in diverse fields (especially, medical domains).

5.6 Quantifying privacy and utility aspects of T_s and T

Most previous studies assumed that privacy issues from T_s are minor because AI models do not overfit. However, we now hold a different view after analyzing the privacy and utility aspects of T_s w.r.t. T. We experimentally compared the results and found that since T_s is a replication of T, the probability of identity or SA disclosure can be high, owing to the capabilities of adversaries as well as the amount of data available to them. To verify the results, we created various anonymized versions of both T_s and T, comparing privacy and utility. We executed a background knowledge attack and a linking attack to measure privacy. In contrast, we used an information loss metric to evaluate utility.

5.6.1 Evaluation of privacy

We used a disclosure risk (a.k.a. probability of re-identification) metric in two distinct attack scenarios: linking and background knowledge (BK) attacks. In the former, an adversary tries to associate published and auxiliary data to extract the users' SA. In the latter, the adversary already has some information about the target individuals. The disclosure risk, D_r, can be determined via Eq. 22:

$$D_r(u_i, v_i) = \sum_{i=1}^{|b|} v_i / \sum_{m=1}^{k} v_m \tag{22}$$

where v_i denotes an SA value that an attacker wants to find related to the target user, u_i. The denominator shows the aggregate of the frequencies of different SA values in a class. $D_r = 1$, when all users have the same SA in a class.

We considered worst-case scenarios (journalist case) while measuring and comparing D_r values from T and T_s. We found accurate matches using QID values in distinct classes made from both T and T_s, and determined the probability of income being either $\leq$50K or $>$50K by employing multiple BK forms [44]. The overview of attributes and actual values used in the BK attack are given in Table 9.

TABLE 9: Overview of QIDs/SA and their values used in the BK attack executed on T and T_s (adapted from [44]).

Known (or already exposed) QIDs/SA	Target SA/QID value
Race, work-hours, age, country: {white, $\geq$ 40, 20, USA}	P(income $\leq$50K)
Age interval, marital status, profession: {40-50, Married, Prof.}	P(income $>$50K)
Country, gender, income: {China, Male, $>$50K }	P(age $\leq$50 Yrs.)
Sex, qualification, work-class, race : {Female, Ph.D., Private, White }	P(income $>$50K)
Marital status, sex, profession, age: {Widowed, Female, Sales, 50}	P(income $\leq$50K)

In the linkage attack, we computed the correct links based on QID values and analyzed distinct values accordingly. Table 10 presents a comparative analysis of D_r results from both attacks.

TABLE 10: Privacy analysis of T and T_s under two attacks.

k	Avg. D_r values (BK attack)		Exposed records	Avg. D_r values (Linking attack)		Previous method
	Real Data	Synthetic data		Real Data	Synthetic data	
2	$0.99 \pm \theta$	$0.53 \pm \theta$	331	$0.45 \pm \theta$	$0.40 \pm \theta$	0.06
5	$0.79 \pm \theta$	$0.41 \pm \theta$	981	$0.49 \pm \theta$	$0.42 \pm \theta$	0.06
10	$0.88 \pm \theta$	$0.49 \pm \theta$	1,954	$0.51 \pm \theta$	$0.49 \pm \theta$	0.06
15	$0.59 \pm \theta$	$0.49 \pm \theta$	3,263	$0.58 \pm \theta$	$0.49 \pm \theta$	0.06
25	$0.54 \pm \theta$	$0.50 \pm \theta$	4,895	$0.64 \pm \theta$	$0.56 \pm \theta$	0.06

From the results given in Table 10, we can see that D_r depends on the type of attack, and D_r from T_s was much higher than a previously measured generic analysis via demographics [15]. Through experiments, we found that D_r can vary based on the granularity of auxiliary information and the amount of data readily exposed to an adversary. In addition, the D_r is subject to change when a strong privacy protection method is used or fewer records are exposed. Therefore, we use $\pm$ θ in Table 10 to reflect these changes in results in realistic scenarios. In one test, we found that D_r differed by just $\sim$ 2%. See 1,954 records exposed scenario in Table 10. These findings and results further rectify/improve previous studies. Through experimental analysis, it can be concluded that privacy leakage can occur from SD when contemporary attacks are launched on it.

5.6.2 Evaluation of utility

To evaluate utility, we employed an information loss (IL) metric. During anonymization, QID values are replaced with new values, and thereby, some information is always lost. To measure IL from anonymized versions of both T_s and T, we used a distortion measure (DM) metric in order to quantify IL. DM values can be determined by figuring out the level of hierarchy on which values of QIDs are mapped. The DM value can be calculated via Eq. 23:

$$DM = \sum_{u=1}^{|T^{'}|} \sum_{Q=1}^{p} \frac{l_g}{l_t} \tag{23}$$

where l_g represents the level at which the QID value was mapped, and l_t represents the total # of levels in the hierarchy. DM values from every QID and sample were summed for all N subjects in the data. If the value of a particular QID is not changed, the DM value will be 0. The IL results for five different k values are shown in Table 11.

TABLE 11: IL (a.k.a. utility) comparisons between T and T_s.

k	Avg. IL values		
	Real Data	Synthetic data	IL difference in %
4	0.51	0.55	6.77
8	0.59	0.63	7.84
12	0.60	0.66	10.00
16	0.63	0.71	12.69
20	0.65	0.74	13.84

Referring to Table 11, it can be seen that IL increase with k value, and the difference in IL between T and T_s is not very large, which indicate that quality of T_s resulted from our model is better. Although T_s has higher IL than T, it can still offer useful information concerning analytics (e.g., histogram analysis) or general data mining tasks.

5.7 Analysis of T_s from theoretical perspectives

In this section, we report experimental findings concerning T_s from theoretical perspectives.

5.7.1 Effects of data distributions

Besides the other analyses, we analyzed values' distributions in SA and QIDs. We chose four attributes, (profession, income, race, and age) for analysis. The first two belong to the SA category, and the last two are QIDs. The purpose of analyzing the distributions is to certify that T_s correctly mirrors the properties of T. Figure 9 presents the comparative results of distributions in the profession (an SA) and race (a QID). The x-axis in left Figure shows the

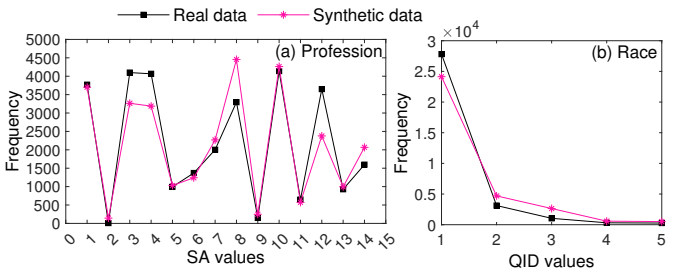


Fig. 9: SA and QID value distributions in T and T_s.

different values of profession (For example, 1=Adm-clerical, and 14=Transport-moving). The x-axis in the right Figure shows the different values of race (For example, 1=White, and 5= Other). From the results, we can see that the CGAN can correctly model all values from T with nearly perfect distributions. These results confirm the validity of T_s for downstream/general-purpose tasks. Similarly, for another SA (income), the difference between values distributions for income: $> 50K$ and $\leq 50K$ was not considerably high.

From the results, observe that all category values from T (i.e., major and minor) were correctly replicated in T_s, and the distributions are mostly alike. These results indicate that T_s has a better distribution for QID and SA. Analysis of the distribution for age is presented in Fig. 10. From the results, observe that the distribution of values in T_s is very close to the distribution in T. In addition, most age values in T_s are also very close. These findings confirm the utility of T_s from multiple aspects (e.g., data analytics and mining).

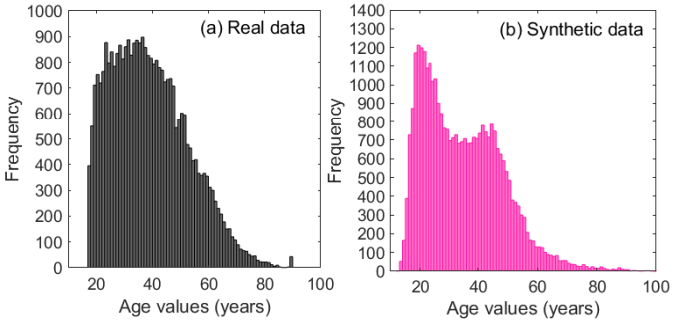


Fig. 10: Distribution analysis of age: T versus T_s.

5.7.2 Missing values

Recently, the GAN has been increasingly used in medical domains for data interpolation and augmentation [45]. To this end, we analyzed T_s regarding missing values, and found that a CGAN can assist in reducing the number of missing values, which can help knowledge discovery from T_s when using advanced data mining tools. There were three attributes with missing values (work class, occupation, and country). The CGAN reduced the number of missing values significantly in the work class, but in the other two attributes, the number of missing values was slightly large. However, the number of missing values can be reduced by optimizing the hyperparameters of the CGAN and enforcing conditions. These findings confirm the validity of T_s in improving various critical aspects of a real-world T.

5.7.3 Fulfilment of legal/regulatory compliance

Since T_s is close to T in most aspects, it can be shared with researchers without hesitation for analytics. SD can stay legally compliant by providing population-level information accurately, rather than at the individual (or sample-level) level. Also, SD can ensure legality by not sharing the real data, and can overcome the challenges of unnecessary processing and anonymization. Lastly, SD can help develop data products where most testing can be done using T_s.

5.7.4 Overcoming barriers to data distribution

There are many situations (e.g., COVID-19 tests/cases, HIV patients, dementia patients) in which access to real data is restricted due to privacy concerns and/or data unavailability on a large scale. In these circumstances, bringing algorithms close to the real data to mirror the properties of T and generate SD can be a handy tool for wide-scale distribution. In our experiments, we found that SD can be created at factors of $1\times$, $1.5\times$, and $2\times$ T, and therefore, its distribution to legitimate information consumers can be helpful in their research or in observing commonalities among differences. T_s can have multiple applications in healthcare, smart cities, natural language processing (NLP), and compliance analysis.

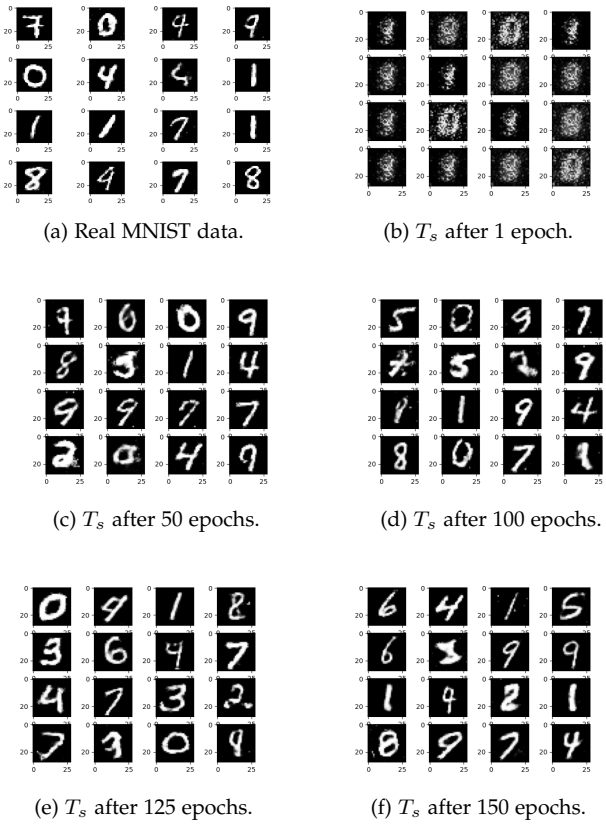
5.7.5 Privacy preservation in AI systems using T_s

As stated earlier, SD has already been used in AI systems as a defense tool against membership inference attacks [1]. In the coming years, SD will be vital in securing training data for AI systems as well as in augmenting AI model

performance. Since performance can be enhanced from huge amounts of data, joint use of T_s and T is inevitable in the near future. Finally, SD can prevent concept drift issues and white box attacks on AI systems.

5.7.6 Efficacy in image recognition and NLP-related tasks

Recently, SD has gained momentum in the research community and has been widely used in real-world applications for augmenting classifier performance, privacy preservation, software/product testing, and data governance. SD can be generated in multiple forms such as tables, images, and time series. GAN-based models are handy in generating synthetic images that can be used for downstream tasks. To generate SD in image form, we used an open-source PyTorch-based implementation of a deep convolution GAN (DCGAN) [46] and produced synthetic characters using the MNIST database of handwritten digits. In the experiments, we divided data into a training set that constituted 60,000 images, and a testing set that encompassed 10,000 images. Fig. 11 presents the results of SD for image recognition tasks.



(a) Real MNIST data. (b) T_s after 1 epoch.

(c) T_s after 50 epochs. (d) T_s after 100 epochs.

(e) T_s after 125 epochs. (f) T_s after 150 epochs.

Fig. 11: GAN performance in an image recognition task.

From the results, we can see that GAN models can assist in generating images of good quality that are close to the real data. It is important to note that image quality was not ideal at the start, and characters were not easily readable. However, the quality of characters improved with an increase in the number of epochs. We believe that by training the GAN model for a long time, good-quality images can be curated. Similarly, a GAN can also be used to generate SD that can be used in NLP tasks [47]. For instance, SD can be used for sentiment analysis, automatic correction in machine translation, knowledge distillation, next-word prediction, and text generation, especially when the labeled data are limited [48]. However, the quality and size of the SD to be used in NLP tasks are very challenging and require further investigation from the research community.

It is important to note that T_s has many general benefits as discussed in Section 5.7. For example, it can be helpful to analyze the distribution of values in some specific attribute of data as shown in Figure 9. It can also be used to perform histogram/frequency analysis of some attributes as shown in Figure 10. It can be used to impute missing values to complete data as discussed in subsection 5.7.2. It can be used to fulfill legal/regulatory requirements as it is regarded as a coarse form of T, as discussed in subsection 5.7.3. It can foster secondary uses of data by enabling data sharing at a large scale as discussed in subsection 5.7.4. It can be used to address privacy issues of different types in AI environments as discussed in subsection 5.7.5. It can also be used in diverse tasks such as image recognition (as shown in Figure 11) and NLP tasks, as discussed in subsection 5.7.6. With the advancements in AI, T_s is becoming one of the mainstream technologies, particularly when access to real data is limited due to either privacy concerns or regulatory measures.

6 Discussion

Tabular data is one of the ubiquitous and most widely used forms of data in AI applications. Due to its simplified nature and easier interpretation, it has been used in most sectors. Due to privacy concerns and a lack of expertise in handling personal data, tabular data cannot be easily outsourced from data owner environments. In contrast, most AI applications require a mammoth volume of data for training classifiers to solve real-world problems [49]. In some cases, data owners outsource the data in an anonymized form to conduct analytics. However, anonymized data cannot be directly fed into classifiers, and extensive post-processing is needed to make it AI-ready. Furthermore, anonymized data yield limited performance (e.g., lower accuracy) in AI models, as shown in Fig. 12. In contrast, data generated with generative AI tools can be synergized with different forms of data to compensate for the availability of good data. Augmented data (real data combined with SD) can enhance the accuracy of AI models, leading to an effective solution for classification/prediction.

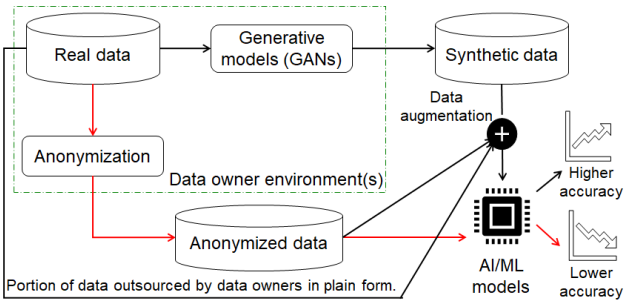
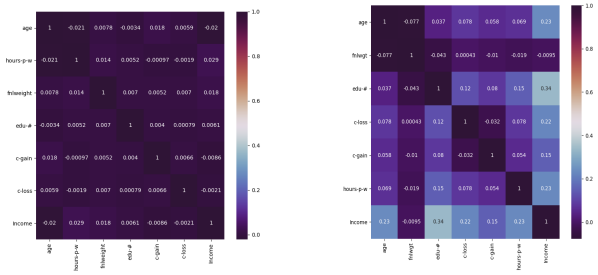


Fig. 12: Potential usage of T_s in the date-driven era.

It is important to note that T_s generation is a non-trivial task, especially when T has mixed attribute types

and many vulnerabilities (e.g., imbalanced distributions, fewer records, outliers, missing values, etc.). Thus far, many generative models have been proposed to generate T_s from T, but when the quality of underlying T is poor, most models cannot generate good T_s. Furthermore, most of the prior generative models are less flexible, meaning AI practitioners cannot curate data based on their needs. Also, the poor design of previous generative models and hyperparameters can also lead to overfitting issues, and the training process can become unstable. To address these practical challenges, this paper implemented a CGAN-based model to yield superior-quality SD. Our model is more flexible than previous SOTA models. The resulting SD were evaluated from both privacy and utility perspectives, whereas existing methods only analyzed the closeness between T and T_s, and ignored privacy issues that can emerge from T_s. Lastly, the T_s produced with our model has many applications in the AI domain such as data augmentation, balanced learning in AI models, higher accuracy, and fair decision-making. We believe our analysis and findings from a real dataset can effectively address the performance bottlenecks currently faced by numerous GAN-based models.

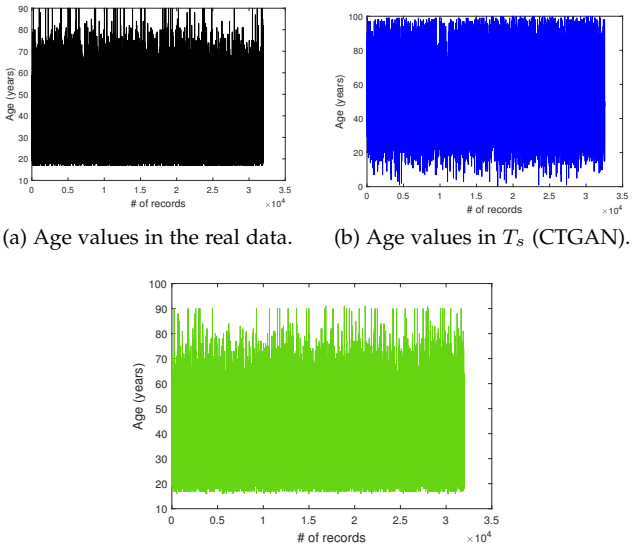
To prove the importance of the proposed CGAN in realistic scenarios, more intuitive results are given in Figs. 13 and 14. From Fig. 13, we can see that a T_s produced by our method has better quality than the SOTA model in terms of correlations. Through detailed analysis, we found only one attribute negatively correlated with income in T. However, the T_s produced with the CTGAN had more attributes negatively correlated with income. Therefore, the T_s produced with that model can lead to wrong analyses or conclusions in realistic scenarios. In contrast, our method correctly retained this property of T, and T_s had only a few attributes negatively correlated with income. Hence, the possibility of wrong analyses or conclusions in realistic scenarios is the least.



(a) Correlation matrix for CTGAN model. (b) Correlation matrix for proposed method.

Fig. 13: Comparisons of correlation matrices for T_s.

In another set of experiments, we compared the closeness between attribute values in T and T_s, and the corresponding results are in Fig. 14. As stated earlier, the CGAN model can produce inconsistent values for some columns, and the offset between real and synthetic values is very large, as shown in Fig. 14(b). In contrast, our method did not yield any inconsistent values, and the offset between real and synthetic values was very small, as shown in Figure 14(c). These results fortify the significance of our method in terms of correctly mimicking the properties of T.



(a) Age values in the real data. (b) Age values in T_s (CTGAN).

(c) Age values in T_s (our method).

Fig. 14: Comparison of closeness of values in T and T_s.

Based on the analysis above, good-quality SD can be used as a substitute for personal big data (i.e., private data stemming from people's daily lives and/or work). However, creating a knowledge base/store of SD to evaluate its effectiveness from multiple perspectives with advanced data mining and AI tools needs further investigation.

6.1 Broader impact of this research

Due to the rapid proliferation of data generation tools (e.g., wearable devices, sensors, etc.), data of diverse modalities is curated, which can be utilized to generate actionable insights with AI models. Unfortunately, most of the data gets locked in the data owner's environment due to privacy issues, and cannot be shared with relevant data owners/analysts, impacting the data-driven innovation and financial opportunities. How to share the data at a large scale while limiting privacy concerns is a key challenge in the modern AI-driven era. To alleviate privacy concerns, governments in Europe, the U.S., and other countries have been proposing and passing regulations to ensure privacy protection and data traceability across data-driven products/frameworks. Also, recent developments like federated learning are assisting in accomplishing the crucial task of user privacy preservation while allowing AI model development with fragmented/scattered datasets. However, addressing all types of privacy problems and ensuring the responsible use of data is still very challenging. To this end, SD is a novel solution that can effectively address privacy issues while enabling data sharing on a large scale. Recently, it has become a mainstream solution for AI developments to improve training data quality, contributing to AI model development of higher generalizability and robustness.

SD can be very useful in solving data island problem, the realization of a data-centric AI concept, democratizing

AI developments, addressing privacy concerns, and developing data-driven products. Our work targets two crucial aspects in the SD field: high-quality SD generation and its experimental evaluation. Our model results in the best quality SD generation via an improved CGAN which can be generically applied, and can improve the performance of ML techniques in terms of accuracy, sampling quality, and confusion matrix's balancedness. Our evaluation of SD in terms of privacy and utility underscores the need for proper assessment of SD w.r.t. privacy leakage and data/information availability. We delve into the downsides of SD in terms of privacy leakage, and we find that SD is less resistant to contemporary privacy attacks. On the other hand, we test the efficacy of SD in terms of various data mining tasks, which can be useful to harness the potential of SD in relevant scenarios. Lastly, our findings can be vital to ML and the database community to properly govern and use SD in real-world applications. Finally, we suggest that careful analysis of scenarios/situations when one can/cannot fully rely on the results derived from T_s needs more research from the research communities.

7 Conclusion and Future Work

In this paper, we implemented a flexible and generic solution for best-quality synthetic tabular data generation with mixed attribute types (i.e., categorical and numerical). Specifically, we solved the following implementation challenges in data generation: numerical attribute modeling and transformation when attributes cannot be represented in the normalized form of [-1,1], which can lead to the vanishing gradient problem in GAN, and categorical attribute modeling when distributions of some attributes can be highly imbalanced, thus preventing model collapse during simultaneous training of neural networks. We also enforced conditions in data generation to reflect all categories from T in T_s (whether major or minor), relaxing constraints on the amount of T_s to be generated. We fortified the effectiveness of the data generated by the CGAN with the help of multiple use cases that have not been experimentally tested in the literature. To the best of our knowledge, this is the maiden attempt at providing a deeper analysis of T_s in both original and anonymized forms. Our work contributes a major privacy-enhancing technology (PET) to reach the goals of responsible data science (e.g., analytics of personal data without misuse) and data augmentation (e.g., improving the performance of AI models by amalgamating T and T_s). In the future, we plan to further optimize the performance of the CGAN model and compare it with advanced generative models. We intend to extend our method to generate synthetic image data that can be helpful in medical domains for COVID-19 diagnosis or mobile-physician apps. Lastly, we intend to generate a T_s of high quality that can be applied to various NLP-related tasks.

Acknowledgments

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (Ministry of Science and ICT) (RS-2024-00340882).

References

[1] L. Hu, J. Li, G. Lin, S. Peng, Z. Zhang, Y. Zhang, and C. Dong, "Defending against membership inference attacks with high utility by gan," *IEEE Transactions on Dependable and Secure Computing*, 2022.

[2] M. Hernandez, G. Epelde, A. Beristain, R. Álvarez, C. Molina, X. Larrea, A. Alberdi, M. Timoleon, P. Bamidis, and E. Konstantinidis, "Incorporation of synthetic data generation techniques within a controlled data processing workflow in the health and wellbeing domain," *Electronics*, vol. 11, no. 5, p. 812, 2022.

[3] M. Hernandez, G. Epelde, A. Alberdi, R. Cilla, and D. Rankin, "Synthetic data generation for tabular health records: A systematic review," *Neurocomputing*, 2022.

[4] S. James, C. Harbron, J. Branson, and M. Sundler, "Synthetic data use: exploring use cases to optimise data utility," *Discover Artificial Intelligence*, vol. 1, no. 1, pp. 1–13, 2021.

[5] L. Li, Y. Fan, M. Tse, and K.-Y. Lin, "A review of applications in federated learning," *Computers & Industrial Engineering*, vol. 149, p. 106854, 2020.

[6] Z. Azizi, C. Zheng, L. Mosquera, L. Pilote, and K. El Emam, "Can synthetic data be a proxy for real clinical trial data? a validation study," *BMJ open*, vol. 11, no. 4, p. e043497, 2021.

[7] E. Strickland, "Andrew ng, ai minimalist: The machine-learning pioneer says small is the new big," *IEEE Spectrum*, vol. 59, no. 4, pp. 22–50, 2022.

[8] C. Hegde, "Anomaly detection in time series data using data-centric ai," in *2022 IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT)*. IEEE, 2022, pp. 1–6.

[9] M. Motamedi, N. Sakharnykh, and T. Kaldewey, "A data-centric approach for training deep neural networks with less data," *arXiv preprint arXiv:2110.03613*, 2021.

[10] A. Zeiser, B. Özcan, B. van Stein, and T. Bäck, "Evaluation of deep unsupervised anomaly detection methods with a data-centric approach for on-line inspection," *Computers in Industry*, vol. 146, p. 103852, 2023.

[11] A. Yale, S. Dash, R. Dutta, I. Guyon, A. Pavao, and K. P. Bennett, "Generation and evaluation of privacy preserving synthetic health data," *Neurocomputing*, vol. 416, pp. 244–255, 2020.

[12] D. Rankin, M. Black, R. Bond, J. Wallace, M. Mulvenna, G. Epelde *et al.*, "Reliability of supervised machine learning using synthetic data in health care: Model to preserve privacy for data sharing," *JMIR Medical Informatics*, vol. 8, no. 7, p. e18910, 2020.

[13] G. Deng, C. Han, and D. S. Matteson, "Extended missing data imputation via gans for ranking applications," *Data Mining and Knowledge Discovery*, pp. 1–23, 2022.

[14] K. El Emam, "Seven ways to evaluate the utility of synthetic data," *IEEE Security & Privacy*, vol. 18, no. 4, pp. 56–59, 2020.

[15] K. El Emam, L. Mosquera, E. Jonker, and H. Sood, "Evaluating the utility of synthetic covid-19 case data," *JAMIA open*, vol. 4, no. 1, p. ooab012, 2021.

[16] N. C. Abay, Y. Zhou, M. Kantarcioglu, B. Thuraisingham, and L. Sweeney, "Privacy preserving synthetic data release using deep learning," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2019, pp. 510–526.

[17] S.-C. Li, B.-C. Tai, and Y. Huang, "Evaluating variational autoencoder as a private data release mechanism for tabular data," in *2019 IEEE 24th Pacific Rim International Symposium on Dependable Computing (PRDC)*. IEEE, 2019, pp. 198–1988.

[18] S. Wang, C. Rudolph, S. Nepal, M. Grobler, and S. Chen, "Part-gan: privacy-preserving time-series sharing," in *International Conference on Artificial Neural Networks*. Springer, 2020, pp. 578–593.

[19] J. Martinsson, E. L. Zec, D. Gillblad, and O. Mogren, "Adversarial representation learning for synthetic replacement of private attributes," in *2021 IEEE International Conference on Big Data (Big Data)*. IEEE, 2021, pp. 1291–1299.

[20] A. Torfi, E. A. Fox, and C. K. Reddy, "Differentially private synthetic medical data generation using convolutional gans," *Information Sciences*, vol. 586, pp. 485–500, 2022.

[21] A. Appenzeller, M. Leitner, P. Philipp, E. Krempel, and J. Beyerer, "Privacy and utility of private synthetic data for medical data analyses," *Applied Sciences*, vol. 12, no. 23, p. 12320, 2022.

[22] N. Elaraby, S. Barakat, and A. Rezk, "A conditional gan-based approach for enhancing transfer learning performance in few-shot hcr tasks," *Scientific Reports*, vol. 12, no. 1, p. 16271, 2022.

[23] H. Lu, M. Du, K. Qian, X. He, and K. Wang, "Gan-based data augmentation strategy for sensor anomaly detection in industrial robots," *IEEE Sensors Journal*, vol. 22, no. 18, pp. 17 464–17 474, 2021.

[24] M. Hammami, D. Friboulet, and R. Kéchichian, "Cycle gan-based data augmentation for multi-organ detection in ct images via yolo," in *2020 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2020, pp. 390–393.

[25] R. Zhang, W. Lu, J. Gao, Y. Tian, X. Wei, C. Wang, X. Li, and M. Yu, "Rfi-gan: A reference-guided fuzzy integral network for ultrasound image augmentation," *Information Sciences*, vol. 623, pp. 709–728, 2023.

[26] H. Chen, J. Chen, and J. Ding, "Data evaluation and enhancement for quality improvement of machine learning," *IEEE Transactions on Reliability*, vol. 70, no. 2, pp. 831–847, 2021.

[27] E. Choi, S. Biswal, B. Malin, J. Duke, W. F. Stewart, and J. Sun, "Generating multi-label discrete patient records using generative adversarial networks," in *Machine learning for healthcare conference*. PMLR, 2017, pp. 286–305.

[28] N. Park, M. Mohammadi, K. Gorde, S. Jajodia, H. Park, and Y. Kim, "Data synthesis based on generative adversarial networks," *arXiv preprint arXiv:1806.03384*, 2018.

[29] 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.

[30] J. Engelmann and S. Lessmann, "Conditional wasserstein gan-based oversampling of tabular data for imbalanced learning," *Expert Systems with Applications*, vol. 174, p. 114582, 2021.

[31] Y. Zhang, N. Zaidi, J. Zhou, and G. Li, "Interpretable tabular data generation," *Knowledge and Information Systems*, pp. 1–29, 2023.

[32] H. Shamsudin, U. K. Yusof, A. Jayalakshmi, and M. N. A. Khalid, "Combining oversampling and undersampling techniques for imbalanced classification: A comparative study using credit card fraudulent transaction dataset," in *2020 IEEE 16th international conference on control & automation (ICCA)*. IEEE, 2020, pp. 803–808.

[33] A. A. El-Sayed, M. A. M. Mahmood, N. A. Meguid, and H. A. Hefny, "Handling autism imbalanced data using synthetic minority over-sampling technique (smote)," in *2015 third world conference on complex systems (WCCS)*. IEEE, 2015, pp. 1–5.

[34] T. Tang, D. Jiao, T. Chen, and G. Gui, "Medium-and long-term precipitation forecasting method based on data augmentation and machine learning algorithms," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 15, pp. 1000–1011, 2022.

[35] J. Chakraborty, S. Majumder, and T. Menzies, "Bias in machine learning software: Why? how? what to do?" in *Proceedings of the 29th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, 2021, pp. 429–440.

[36] A. S. Dina, A. Siddique, and D. Manivannan, "Effect of balancing data using synthetic data on the performance of machine learning classifiers for intrusion detection in computer networks," *IEEE Access*, vol. 10, pp. 96 731–96 747, 2022.

[37] C. M. Bishop and N. M. Nasrabadi, *Pattern recognition and machine learning*. Springer, 2006, vol. 4, no. 4.

[38] Z. Lin, A. Khetan, G. Fanti, and S. Oh, "Pacgan: The power of two samples in generative adversarial networks," *Advances in neural information processing systems*, vol. 31, 2018.

[39] 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.

[40] K. Liu and G. Qiu, "Lipschitz constrained gans via boundedness and continuity," *Neural Computing and Applications*, vol. 32, pp. 18 271–18 283, 2020.

[41] P. Murphy, "Uci repository of machine learning databases. department of information and computer science, university of california," *http://www. ics. uci. edu/AI/ML/MLDBRepository. html*, 1992.

[42] R. Shen, S. Bubeck, and S. Gunasekar, "Data augmentation as feature manipulation," in *International conference on machine learning*. PMLR, 2022, pp. 19 773–19 808.

[43] I. K. Nti, J. A. Quarcoo, J. Aning, and G. K. Fosu, "A mini-review of machine learning in big data analytics: Applications, challenges, and prospects," *Big Data Mining and Analytics*, vol. 5, no. 2, pp. 81–97, 2022.

[44] A. Majeed and S. O. Hwang, "When ai meets information privacy: The adversarial role of ai in data sharing scenario," *IEEE Access*, 2023.

[45] A. Tsourtis, G. Papoutsoglou, and Y. Pantazis, "Gan-based training of semi-interpretable generators for biological data interpolation and augmentation," *Applied Sciences*, vol. 12, no. 11, p. 5434, 2022.

[46] Z. Liu, M. Tong, X. Liu, Z. Du, and W. Chen, "Research on extended image data set based on deep convolution generative adversarial network," in *2020 IEEE 4th Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)*, vol. 1. IEEE, 2020, pp. 47–50.

[47] L. F. A. O. Pellicer, T. M. Ferreira, and A. H. R. Costa, "Data augmentation techniques in natural language processing," *Applied Soft Computing*, vol. 132, p. 109803, 2023.

[48] X. He, I. Nassar, J. Kiros, G. Haffari, and M. Norouzi, "Generate, annotate, and learn: Nlp with synthetic text," *Transactions of the Association for Computational Linguistics*, vol. 10, pp. 826–842, 2022.

[49] A. Kiran and S. S. Kumar, "Synthetic data and its evaluation metrics for machine learning," in *Information Systems for Intelligent Systems: Proceedings of ISBM 2022*. Springer, 2023, pp. 485–494.

Abdul Majeed received the B.S. degree in Information Technology from the UIIT, PMAS-UAAR, Rawalpindi, Pakistan, in 2013, the M.S. degree in Information Security from the COMSATS University, Islamabad, Pakistan, in 2016, and the Ph.D. degree in Computer Information Systems & Networks from the Korea Aerospace University, Korea, in 2021. He worked as a Security Analyst with Trillium Information Security Systems (TISS), Rawalpindi, Pakistan, from 2015 to 2016. He is currently working as an Assistant Professor with the Department of Computer Engineering, Gachon University, Korea. His research interests include data-centric artificial intelligence, privacy-preserving data publishing, and machine learning.

Seong Oun Hwang received the B.S. degree in mathematics from Seoul National University, in 1993, the M.S.degree in information and communications engineering from the Pohang University of Science and Technology, in 1998, and the Ph.D. degree in computer science from the Korea Advanced Institute of Science and Technology, in 2004, South Korea. He worked as a Software Engineer with LG-CNS Systems,Inc., from 1994 to 1996. He also worked as a Senior Researcher with the Electronics and Telecommunications Research Institute (ETRI), from 1998 to 2007. He worked as a Professor with the Department of Software and Communications Engineering, Hongik University, from 2008 to 2019. He is currently working as a full Professor with the Department of Computer Engineering, Gachon University, Korea. His research interests include cryptography, cybersecurity, and data-centric artificial intelligence.