TabUnite: An Efficient Encoding Framework for Tabular Data Generation

Anonymous Author(s)
Affiliation
Address
email

Abstract

Generative models for tabular data face a long-standing challenge in the effective modelling of heterogeneous feature interrelationships, especially for generating tabular data with both continuous and categorical input features. Capturing these interrelationships is crucial as it allows models to understand complex patterns and dependencies that exist in the underlying data. A promising option to address the challenge is to devise suitable encoding/embedding schemes for the input features before the generative modelling process. However, prior methods often rely on either suboptimal heuristics such as one-hot encoding of discrete features and separated modelling of discrete/continuous features, or latent space generative models. Instead, our proposed solution leverages efficient continuous encodings to unify the data space and applies a single generative process across all the encodings jointly, thereby efficiently capturing heterogeneous feature interrelationships. Specifically, it employs encoding schemes such as Analog Bits or Dictionary Encoding that effectively convert discrete features into continuous ones. Extensive experiments on real-world and synthetic tabular datasets comprising of heterogeneous features demonstrate that our encoding schemes, combined with Flow Matching as the generative model, significantly enhances model capabilities. Our models, TabUnite-i2bFlow and TabUnite-dicFlow, are able to address data heterogeneity, achieving superior performances across a broad suite of datasets, baselines, and benchmarks while generating accurate, robust, and diverse tabular data.

1 Introduction

Tabular data is omnipresent in data ecosystems of many sectors such as healthcare, finance, and insurance (Clore et al., 2014; Moro et al., 2012; Datta, 2020). These industries utilise tabular data generation for many practical purposes, including imputing missing values, reducing sparse data, and better handling imbalanced datasets (Jolicoeur-Martineau et al., 2024; Onishi & Meguro, 2023; Sauber-Cole & Khoshgoftaar, 2022). However, generative models face challenges inherent to tabular data including feature heterogeneity (Liu et al., 2023). Unlike homogeneous data modalities such as images or text, tabular data often contain mixed feature types, ranging from (dense) continuous features to (sparse) categorical features. More importantly, these tabular features, regardless of form, are intertwined contextually (Borisov et al., 2023). For example, the numerical salary of a person is correlated to their categorical age and education (Becker & Kohavi, 1996). Therefore capturing the interrelationships between tabular heterogeneous features is crucial, as it allows models to incorporate contextual knowledge for understanding complex patterns and dependencies in the underlying data. Additionally, an increasing demand is observed for larger tabular generative models trained potentially on many different datasets, where the capability to model heterogeneous feature spaces across datasets is of utmost importance (van Breugel & van der Schaar, 2024).

Submitted to 38th Conference on Neural Information Processing Systems (NeurIPS 2024). Do not distribute.

A promising solution for the feature heterogeneity challenge is to devise suitable encoding/embedding schemes for pre-processing the input features before applying the generative model. However, existing methodologies often rely on (1) separate generative processes on discrete & continuous features which do not model their correlations properly, (2) sub-optimal encoding heuristics, or (3) learned latent embedding which is parameter inefficient. For example, the one-hot encoding approach for categorical variables leads to sparse representations in high dimensions, where generative models are susceptible to under-fitting (Krishnan et al., 2017; Poslavskaya & Korolev, 2023). On the other hand, creating a latent embedding space requires training an additional embedding model based on e.g., ResNet (He et al., 2015) or a Transformer-based β-VAE (Higgins et al., 2017; Kingma & Welling, 2013; Zhang et al., 2023) and trained using e.g., self-supervised learning (Chen et al., 2020). Hence, the quality of latent space generative models also depends on the embedding model's capability to capture the underlying dependency structure of the tabular data. Overall, proper pre-processing of heterogeneous features is crucial for high-quality tabular data generation, and poor encoding schemes for the data features can lead to information loss that can not be recovered from the generative model.

The goal of our work is to generate high-quality synthetic tabular data by employing *(1) proficient categorical encoding schemes* to unify the data space. This enables a single generative model to be applied while enforcing a *(2) fast and efficient sampling* procedure. In summary, our contributions are as follows:

1. We devise two categorical encoding schemes using Analog Bits (Chen et al., 2022) and Dictionary Encoding (partially inspired by Mairal et al. (2008, 2009)) that seamlessly convert categorical variables into an efficient and compact continuous representation. By facilitating the model to generate data in a unified continuous space, we can "unite" the mixed features to capture heterogeneous feature interrelationships based on a single generative model on continuous inputs. Empirically, under our encoding schemes, the model learns to accommodate the heterogeneity of tabular features.
2. We employ Flow Matching (Lipman et al., 2022; Liu et al., 2022; Tong et al., 2023) as our generative model. It is a simulation-free framework for training continuous normalizing flow models (Chen et al., 2019) by replacing the stochastic diffusion process with a predefined probability path constructed with theories from optimal transport (McCann, 1997). Our results showcase that combining our categorical encoding schemes with Flow Matching speeds up the sampling speed dramatically, saving time and computation power, while enhancing the generation quality. Consequently, we propose two models: TabUnite-i2bFlow and TabUnite-dicFlow. Both models achieve superior performances across a wide spectrum of tabular data generation baselines, datasets, and benchmarks. The architecture of our models is illustrated in Figure 1.
3. We curate a large-scale heterogeneous tabular dataset from the Census dataset (Meek et al., 2001) with over 80 features of mix-types and over 2.4 million samples. This new benchmark is significantly more challenging for tabular generative models than existing benchmarks from public data repositories (Dua & Graff, 2017; Vanschoren et al., 2013) which often have $< 100k$ datapoints and ≤ 30 features. It reflects better on the scalability of tabular generative models, where our empirical results again reveal the importance of good encoding schemes for heterogeneous features.

2 Related Works

Generative Models in Tabular Data Generation. The latest tabular data generation methods have made considerable progress compared to traditional methods such as Bayesian networks (Rabaey et al., 2024) and SMOTE (Chawla et al., 2002). CTGAN and TVAE (Xu et al., 2019) were two models based on the Generative Adversarial Network (Goodfellow et al., 2014) and Variational Autoencoder (Kingma & Welling, 2013) architectures respectively. These models were applied along with techniques such as conditional generation and mode-specific normalization to further learn column-wise correlation. Other works such as GReaT (Borisov et al., 2023) and GOGGLE (Liu et al., 2023) saw successes with the use of graph neural networks and autoregressive transformer architectures respectively in performing tabular data synthesis. Recently, Diffusion (Ho et al., 2020) and Flow Matching (Lipman et al., 2022) provided new avenues for exploration within the tabular domain. This included STaSy (Kim et al., 2022), which employed a score-matching diffusion model

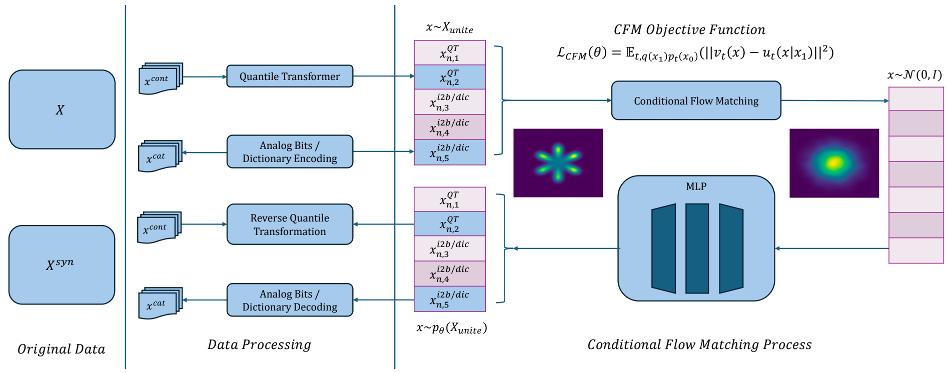


Figure 1: TabUnite-i2bFlow and TabUnite-dicFlow Architecture. Continuous features x^{cont} are encoded via a QuantileTransformer (Pedregosa et al., 2011). Categorical data x^{cat} are encoded using Analog Bits or Dictionary Encoding methods. With an efficient continuous data space, we apply Conditional Flow Matching as our generative model where we ultimately synthesise samples. These samples are then mapped back to their original representation via their respective decoding schemes.

paired with techniques such as self-paced learning and fine-tuning to stabilise the training process, and CoDi (Lee et al., 2023), which utilised separate diffusion schemes for categorical and numerical data along with interconditioning and contrastive learning to improve the synergy among different features. TabDDPM (Kotelnikov et al., 2023) presented a similar diffusion scheme compared to CoDi and showed that the simple concatenation of categorical and numerical data before and after denoising led to improvements in performance. The most recent work in this domain was TabSYN (Zhang et al., 2023), a latent diffusion model which transformed features into a unified embedding via a feature tokenizer before applying EDM diffusion (Karras et al., 2022) to generate synthetic data.

Encoding Schemes. CoDi (Lee et al., 2023) and TabDDPM (Kotelnikov et al., 2023) utilised a separated data space, where Gaussian Diffusion (Ho et al., 2020) was performed on numerical columns and Multinomial Diffusion (Hoogeboom et al., 2021) was performed on categorical columns, with some additional techniques used to bind the two separate diffusion models. However, learning the cross-correlation among various features through separate methods was often less effective than conducting diffusion directly across a unified data space that included all features in the dataset. To achieve this, various encoding schemes were employed to process both categorical and numerical data so they occupy the same data space. One of the most widely used methods was one-hot encoding, which was used in both STaSy (Kim et al., 2022) and TabSYN (Zhang et al., 2023) that encoded categorical columns. One-hot encoding transformed categorical variables into a binary vector, where each category was populated with 0's with the exception of a single 1 that indicated the presence of a particular category. On top of one-hot encoding, TabSYN (Zhang et al., 2023) further used a column-wise feature tokenization technique that together transformed numerical and categorical features all into shared embeddings of the same length.

Flow Methods. Flow methods were introduced to the field of diffusion-based deep generative models as Probability Flow ODEs (Song et al., 2021), which, originally based on the concept of normalizing flows (Rezende & Mohamed, 2016), allowed for deterministic inference and exact likelihood evaluation. Compared to other diffusion-based methods such as score-matching (Song et al., 2021), DDPM (Ho et al., 2020), and DDIM (Song et al., 2022), flow-based models used continuous transformations defined by neural ODEs, to map samples from a simple distribution to samples from a more complex target distribution. This allowed for efficient density estimation and generation of high-dimensional data. In the context of tabular data, Flow Matching was applied to gradient-boosted trees in place of neural networks to learn the vector field (Jolicoeur-Martineau et al., 2024).

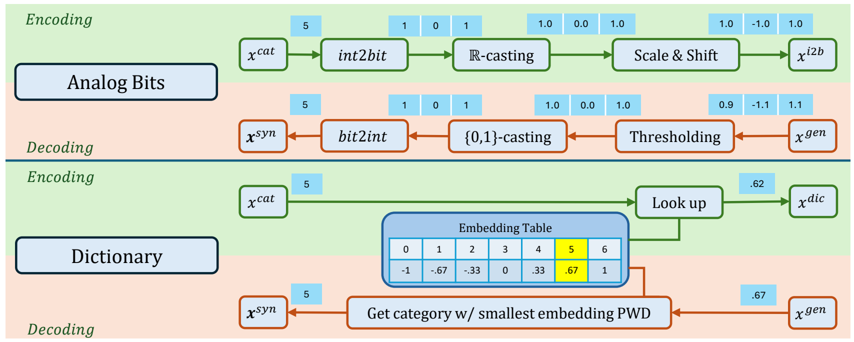


Figure 2: TabUnite Encoding Methods. We leverage Analog Bits & Dictionary encoding to transform categorical features into a compact and efficient continuous representation before applying a single unified generative model to synthesise tabular data.

3 TabUnite Models

Before diving into our methodology, we begin the section with preambles regarding a high-level overview of the tabular setting. Here a tabular dataset is characterized as $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^{N}$ with N samples (rows), where a datapoint $\mathbf{x}_i \in \mathbb{R}^{D_{\text{cont}}} \times \mathbb{N}^{D_{\text{cat}}}$ comprises of D_{cont} continuous features and D_{cat} categorical features. We denote each $\mathbf{x}_i$ as $\mathbf{x}_i := [x_{i,1}^{\text{cont}}, \cdots, x_{i,D_{\text{cont}}}^{\text{cont}}, \cdots, x_{i,1}^{\text{cat}}, \cdots, x_{i,D_{\text{cat}}}^{\text{cat}}]$.

Our goal is to generate synthetic data samples, $\mathbf{x}^{syn}$, that mimic the quality of the real data, $\mathbf{X}$. To do so, we are required to learn a parameterized generative model known as $p_\theta(\mathbf{X})$, from which $\mathbf{x}^{syn}$ can be sampled. Prior to learning, extensive data pre-processing is required where categorical features are encoded into continuous features: $f(x^{cat}) = x^{enc}$, where f denotes the encoder. Poor or sparse feature encoding of categorical features can hinder the model's ability to learn effectively. Therefore, we devise efficient and effective encoding schemes to address this issue.

3.1 Encoding Schemes

We explore Analog Bits (Chen et al., 2022) and Dictionary to encode categorical features. note that continuous features are encoded using the QuantileTransformer (Pedregosa et al., 2011) where we follow TabSYN's and TabDDPM's methodology (Zhang et al., 2023; Kotelnikov et al., 2023).

Analog Bits Encoding. A categorical feature that has K unique categories, $x^{cat} \in \{0, \dots, K-1\}$, can be expressed using $\lceil \log_2(K) \rceil$ binary bits. For example, a categorical feature with $K = 5$ categories is expressed using $\lceil \log_2(5) \rceil = 3$ bits with an embedding function $f(x^{cat}) = x^{enc} \equiv x^{i2b}$ that maps $x^{cat} \in \{0, 1, 2, 3, 4\}$ to $x^{i2b} \in \{000, 001, 010, 100, 101\}$ respectively. Subsequently, each binary bit is cast into a real-valued representation, followed by a shift and scale formula: $x^{i2b} = (x^{i2b} \cdot 2 - 1)$. This transformation shifts and scales the binary values $\{0, 1\}$ to $\{-1, 1\}$. Thus, training and sampling of continuous-feature generative models (e.g., diffusion models) become computationally tractable. For generations, thresholding and rounding are applied to the generated continuous bits from the model to convert them back into binary form, which can be decoded trivially back into the original categorical values.

Dictionary Encoding. A categorical feature that has K unique categories, $x^{cat} \in \{0, \dots, K-1\}$, can be expressed using a look-up embedding table function which encodes the categories to equally spaced real-valued representations within the range $[-1, 1]$. Note that when a categorical feature contains more categories, the embedding requires a larger range to prevent the values from being too close to each other, hindering the model's ability to distinguish between categories. This can be addressed by increasing the range accordingly. The encoding function is defined as follows: $f(x^{cat}) = x^{enc} \equiv x^{dic} = -1 + \frac{2x^{cat}}{K-1}$. For example, a categorical feature with $K = 5$ categories is encoded using the look-up table function, $f(x^{cat})$, that maps $x^{cat} \in \{0, 1, 2, 3, 4\}$ to $x^{dic} \in \{-1, -0.5, 0, 0.5, 1\}$ respectively. Consequently, this also ensures the preservation of the intrinsic order in ordinal data. To perform decoding, the Euclidean pairwise distance between x^{gen} and each of the K categorical embeddings is calculated. The categorical value that corresponds to the nearest embedding vector is chosen. In our experiments, we use a 1-dimensional encoding setup described above. We can also extend Dictionary Encoding to n dimensions when there is a need to capture more nuanced patterns in complex datasets. We create an embedding matrix $M \in \mathbb{R}^{K,n}$ by filling it with randomly sampled values from a standard normal distribution $\mathcal{N}(0, 1)$. We then normalise this embedding matrix by scaling the values of each column linearly to the range $[-1, 1]$, using each column's minimum and maximum values. The resulting matrix is our Dictionary, where we denote the lookup operation as function f.

In Figure 2, we consider an example categorical data point of $x^{cat} = 5$ with $K = 7$ categories where $x^{cat} \in \{0, 1, 2, 3, 4, 5, 6\}$. Analog Bits can encode $x^{cat} = 5$ into $\lceil \log_2(7) \rceil = 3$ bits where we deemed it to be $x^{i2b} = 101$. It is then cast into $\mathbb{R}$ followed by the scale and shift formula. Dictionary creates a look-up embedding table where the different categories are distributed evenly as a real number within the range $[-1, 1]$. In our example, $x^{cat} = 5$ is mapped to $x^{dic} = .67$ by the table. A similar reverse process is applied to both methods for obtaining the decoded representations.

In contrast to traditional one-hot categorical encoding, our encoding methods offer more efficient and dense representations. One-hot encoding can lead to high-dimensional sparse vectors (Poslavskaya & Korolev, 2023) and cause underfitting when learning from it (Krishnan et al., 2017). On the contrary, Analog Bits encoding reduces dimensionality whereas Dictionary encoding transforms the data into a more compact format, preserving the intrinsic relationships between categories. This efficiency can lead to faster training/sampling times, and improved performance in machine learning models by leveraging continuous representations for categorical data. Comparing our two encoding methods, Dictionary encoding is preferred when converting *ordinal* categorical data due to the presence of an intrinsic ordering among the categories that are preserved in the embedding space.

3.2 Conditional Flow Matching

After encoding our continuous and categorical columns, we are presented with a unified and continuous data space, $\mathbf{X}_{\text{i2b}} \in \mathbb{R}^{N \times (D_{cont} + \lceil log_2(D_{cat}) \rceil)}$ and $\mathbf{X}_{\text{dic}} \in \mathbb{R}^{N \times (D_{cont} + D_{cat} \times n)}$. For convenience, we define $\mathbf{X}_{\text{unite}}$ to represent either $\mathbf{X}_{\text{i2b}}$ or $\mathbf{X}_{\text{dic}}$, depending on the encoding method used. Subsequently, we apply Conditional Flow Matching (Lipman et al., 2022) as our generative model to synthesise our tabular data. The Flow matching models built on top of the feature encodings with Analog Bits ("i2b") and Dictionary ("dic") encodings are referred to as TabUnite-i2bFlow and TabUnite-dicFlow, respectively.

Let $\mathbf{x}$ denote a sample from the dataset $\mathbf{X}_{\text{unite}}$, i.e. $\mathbf{x} \sim \mathbf{X}_{\text{unite}}$. We learn a vector field $v_t(\mathbf{x})$ to approximate the true vector field $u_t(\mathbf{x}|\mathbf{x}_1)$, yielding an objective function of the following:

$$L_{CFM}(\theta) = \mathbb{E}_{q(\mathbf{x}_1), p_t(\mathbf{x}|\mathbf{x}_1)} ||v_t(\mathbf{x}) - u_t(\mathbf{x}|\mathbf{x}_1)||^2 \tag{1}$$

This in turn generates a probability density path $p_t(\mathbf{x}|\mathbf{x}_1)$. In order to generate the path $p_t(\mathbf{x}|\mathbf{x}_1)$ via vector field $u_t(\mathbf{x}|\mathbf{x}_1)$, we consider the flow ψ_t:

$$[\psi_t]_* p(\mathbf{x}) = p_t(\mathbf{x}|\mathbf{x}_1) \tag{2}$$

where $\psi_t(\mathbf{x}) = \sigma(\mathbf{x}_1)\mathbf{x} + \mu_t(\mathbf{x}_1)$. This property helps establish a probability path from the noise distribution $p_0(\mathbf{x}|\mathbf{x}_1) = p(\mathbf{x})$ to $p_t(\mathbf{x}|\mathbf{x}_1)$. With the simple affine map property of ψ_t, we use it to solve for vector field u:

$$u_t(\mathbf{x}|\mathbf{x}_1) = \frac{\sigma_t'(\mathbf{x}_1)}{\sigma_t(\mathbf{x}_1)}(\mathbf{x} - \mu_t(\mathbf{x}_1)) + \mu_t'(\mathbf{x}_1) \tag{3}$$

generating Gaussian probability path $p_t(\mathbf{x}|\mathbf{x}_1)$. Lastly, by integrating optimal transport theories, the final objective function is the following:

$$L_{CFM}(\theta) = \mathbb{E}_{t, q(\mathbf{x}_1), p(\mathbf{x}_0)} ||v_t(\psi_t(\mathbf{x}_0)) - (\mathbf{x}_1 - (1 - \sigma_{min})\mathbf{x}_0)||^2 \tag{4}$$

Relative to other generative models, particularly Diffusion, Conditional Flow Matching synthesises tabular data with a much higher sampling speed while also attaining a better generalization.

4 Experiments

We evaluate the performance of TabUnite-i2bFlow (Analog Bits + Flow Matching) and TabUnite-dicFlow (Dictionary encoding + Flow Matching) on a wide range of real-world and synthetic datasets, benchmarks, and compare the proposed models with a comprehensive number of baselines.

Table 1: AUC (classification) and RMSE (regression) scores of Machine Learning Efficiency. ↑ indicates that the higher the score, the better the performance, vice versa. Values bolded in **red** and **blue** are the best and second best-performing models respectively. Details are found in Appendix C.

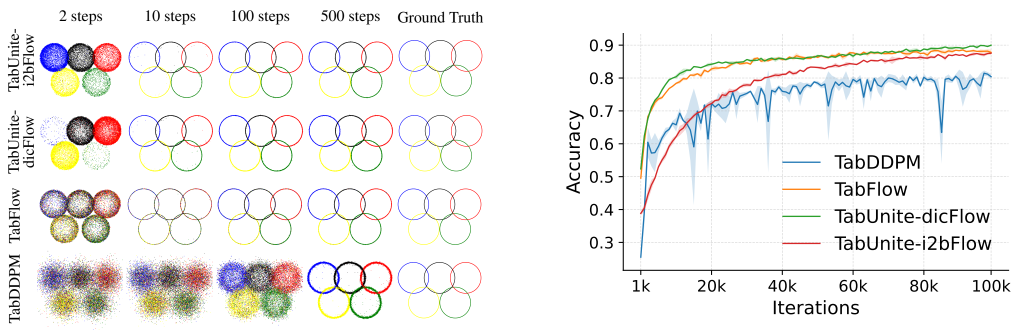
Methods	Adult	Default	Shoppers	Magic	Beijing	News	Overall Rank
	AUC ↑	AUC ↑	AUC ↑	AUC ↑	RMSE ↓	RMSE ↓	
Real	0.927±0.000	0.770±0.005	0.926±0.001	0.946±0.001	0.423±0.003	0.842±0.002	N/A
SMOTE	0.899±0.007	0.741±0.009	0.911±0.012	0.934±0.008	0.593±0.011	0.897±0.036	5
CTGAN	0.886±0.002	0.696±0.005	0.875±0.009	0.855±0.006	0.902±0.019	0.880±0.016	8
TVAE	0.878±0.004	0.724±0.005	0.871±0.006	0.887±0.003	0.770±0.011	1.01±0.016	8
GOGGLE	0.778±0.012	0.584±0.005	0.658±0.052	0.654±0.024	1.09±0.025	0.877±0.002	11
GReaT	0.844±0.005	0.755±0.006	0.902±0.005	0.888±0.008	0.653±0.013	OOM	7
STaSy	0.906±0.001	0.752±0.006	0.914±0.005	0.934±0.003	0.656±0.014	0.871±0.002	4
CoDi	0.871±0.006	0.525±0.006	0.865±0.006	0.932±0.003	0.818±0.021	1.21±0.005	10
TabDDPM	**0.910±0.001**	**0.761±0.004**	**0.915±0.004**	0.932±0.003	1.91±0.680	3.46±1.25	6
TabSYN	0.906±0.001	0.755±0.004	**0.918±0.004**	0.935±0.003	0.586±0.013	0.862±0.021	3
TabUnite-i2bFlow	**0.911±0.001**	**0.763±0.004**	**0.918±0.005**	**0.941±0.003**	**0.543±0.007**	**0.847±0.014**	**1**
TabUnite-dicFlow	**0.911±0.002**	0.758±0.006	0.908±0.006	**0.943±0.003**	**0.555±0.006**	**0.848±0.013**	**2**

Datasets. The datasets in our experiments are from the UCI Machine Learning Repository (Dua & Graff, 2017), synthetic toy datasets (Chen et al., 2018), and our own self-curated dataset, "Census Synthetic". The real-world UCI tabular datasets are chosen because they were previously utilised to evaluate the existing baselines. Next, we leverage synthetic toy datasets to prove the faithfulness of our model. Lastly, we curate a dataset that is much larger than existing datasets in the number of samples (approx. 2.5 million samples) and comes with a large set of mixed features (approx. 40 and 41 categorical and continuous features each). The training/validation/testing sets are split into 80/10/10% apart from the Adult dataset which we adhere to its original documented splits. Full details of the datasets can be found in Appendix C.1.

Baselines: Existing modeling approaches. We compare our model against eight other existing methods for tabular generation. This includes CTGAN (Xu et al., 2019), TVAE (Xu et al., 2019), GOGGLE (Liu et al., 2023), GReaT (Borisov et al., 2023), TabDDPM (Kotelnikov et al., 2023), STaSy (Kim et al., 2022), CoDi (Lee et al., 2023), and, TabSYN (Zhang et al., 2023). SMOTE (Chawla et al., 2002), an interpolation-based method, is also included as a base reference model. The results from CTGAN, TVAE, GOGGLE, GReaT, STaSy, and CoDi are taken from the TabSYN paper (Zhang et al., 2023). The main competitors to our model are TabSYN and TabDDPM since they are the best-performing models to date. Hence, we reproduce the results of TabSYN and TabDDPM per the recommended hyperparameters mentioned by the authors of their respective papers. More details regarding these baselines can be found in Appendix C.2.

Ablations: Encoding schemes and generative models (Flow/Diffusion). We conduct our ablation studies with respect to various encoding schemes and generative models. This assists us in proving the effectiveness of our encoding schemes (Analog Bits and Dictionary) as well as Flow Matching (Lipman et al., 2022) as the generative model. The detailed implementations of these ablations are introduced in Appendix C.3.

Benchmarks & metrics. We evaluate the generative performance on a broad suite of benchmarks from TabSYN (Zhang et al., 2023). We analyse the capabilities in *downstream tasks* such as machine learning efficiency, where we determine the AUC score for classification tasks and RMSE for regression tasks of a tabular data classifier (XGBoost (Chen & Guestrin, 2016)) on the generated synthetic datasets. Next, we conduct experiments on *low-order statistics* where we perform column-wise density estimation (CDE) and pair-wise column correlation (PCC). Lastly, we examine the models' quality on *high-order metrics* such as α-*precision* and β-*recall* scores (Alaa et al., 2022). We add two extra benchmarks (part of Appendix C.4) including a detection test metric, Classifier Two Sample Tests (C2ST) (SDMetrics, 2024) and a privacy preservation metric, Distance to Closest Record (DCR) (Minieri, 2022). Further details regarding this section can be found in Appendix C.4.



(a) Qualitative Synthetic Toy Dataset. (b) Quantitative Synthetic Toy Dataset.

Figure 3: (a) The x-axis illustrates the sampling steps and the "Ground Truth" of the dataset whereas the y-axis depicts the methods. TabUnite methods are faithful in generating high-quality samples that match the ground truth in a short period of sampling duration. (b) The x-axis illustrates the training iterations whereas the y-axis depicts the accuracy of the generated categorical columns. Training TabUnite methods are stable and converge at a higher accuracy when compared to TabDDPM.

4.1 Model Comparisons on Predefined Baselines

We benchmark TabUnite-i2bFlow and TabUnite-dicFlow across 6 datasets, against a wide range of baselines, in terms of a downstream task (machine learning efficiency) – XGBoost's classification/regression performance (Chen & Guestrin, 2016) trained on generated synthetic data (AUC/RMSE). Following the setting in TabDDPM and TabSYN (Kotelnikov et al., 2023; Zhang et al., 2023), we split the datasets into training and testing sets where the generative models are trained on the training set. Synthetic samples of equivalent size are then generated based on the trained generative models. The generated data is subsequently evaluated against the mentioned benchmarks, using the testing set—unseen during training and generation phases—to assess the models' performance and generalization.

As observed in Table 1, both results of TabUnite-i2bFlow and TabUnite-dicFlow achieve the best performance compared to existing baselines. We also identify that TabUnite-i2bFlow is superior to TabUnite-dicFlow as most datasets contain more non-ordinal categorical features than ordinal ones. To further justify the faithfulness of our model, we use synthetic toy examples, allowing us to assess our model's integrity against the known ground truth.

4.2 Ground Truth Assessment with Synthetic Toy Examples

Qualitative Results. We further demonstrate the effectiveness of our method in identifying ground truth relevance for data synthesis. We created a synthetic "Olympic" tabular dataset and visualised it qualitatively in terms of its structure (shape and sharpness of Olympic rings) and colour. Details regarding the dataset can be found in Appendix C.1. Our goal is to illustrate the integrity of our encoding method and sampling speed by mimicking the qualitative ground truth attributes of the real dataset. Our primary predefined model for comparison is TabDDPM. We also introduce TabFlow, a replica of TabDDPM except that we replace DDPM/Multinomial Diffusion with Flow Matching/Discrete Flow Models (Campbell et al., 2024).

Figure 3a displays the synthesised samples for TabUnite-i2bFlow, TabUnite-dicFlow, TabFlow, and TabDDPM across various sampling steps. As early as 10 steps, both TabUnite methods converge, achieving high-quality structure and colour in relation to the ideal "Ground Truth" visualisation. However, there is no apparent "Olympic" structure for TabDDPM. Although TabFlow presents an "Olympic" structure, it is difficult to identify the colours. TabFlow requires approximately 100 steps to differentiate between the colours clearly. Even at 500 steps, TabDDPM is still lacking in terms of its structure where the rings are visually less precise when compared to the "Ground Truth". Therefore, the experiment highlights both TabUnite-i2bFlow and TabUnite-dicFlow's faithfulness and integrity in generating high-quality samples that match the ground truth in a short period of sampling duration.

Quantitative Results. In addition to our qualitative results, we further demonstrate quantitatively that our methods are faithful to the model's decision-making process by creating an additional synthetic toy dataset. In this dataset, categorical columns are created through an injective mapping

Table 2: RMSE (regression), Column-Wise Density Estimation (CDE), Pair-Wise Column Correlation (PCC), α-Precision, and β-Recall scores for our Census Synthetic and Beijing datasets. ↑ indicates that the higher the score, the better the performance, vice versa. Values bolded in **red** and **blue** are the best and second best-performing models respectively. Details are found in Appendix C.

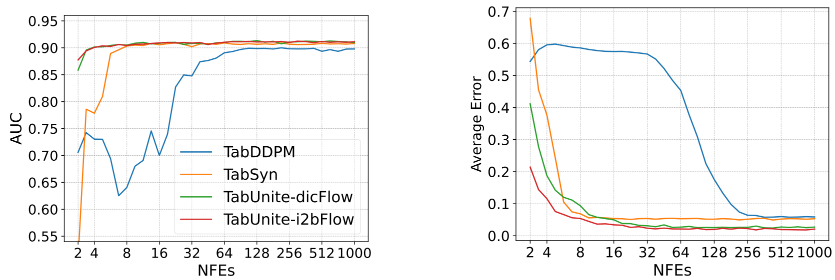
Methods	Census Synthetic					Overall Rank
	RMSE ↓	CDE ↑	PCC ↑	α ↑	β ↑	
TabDDPM	$0.194_{\pm 0.012}$	$\mathbf{86.44}_{\pm 0.011}$	$90.29_{\pm 0.109}$	$86.60_{\pm 0.104}$	$34.37_{\pm 0.050}$	5
oheDDPM	$1.171_{\pm 0.024}$	$55.34_{\pm 0.023}$	$50.66_{\pm 0.014}$	$0.600_{\pm 0.001}$	$0.000_{\pm 0.000}$	8
i2bDDPM	$0.156_{\pm 0.004}$	$76.52_{\pm 0.006}$	$77.38_{\pm 0.584}$	$77.54_{\pm 0.098}$	$1.25_{\pm 0.008}$	6
dicDDPM	$0.168_{\pm 0.005}$	$\mathbf{86.55}_{\pm 0.023}$	$90.36_{\pm 0.109}$	$91.86_{\pm 0.019}$	$34.11_{\pm 0.080}$	4
TabFlow	$\mathbf{0.131}_{\pm 0.005}$	$86.12_{\pm 0.007}$	$90.07_{\pm 0.704}$	$\mathbf{95.31}_{\pm 0.038}$	$\mathbf{39.17}_{\pm 0.098}$	3
oheFlow	$0.332_{\pm 0.003}$	$75.57_{\pm 0.011}$	$79.58_{\pm 0.189}$	$69.59_{\pm 0.080}$	$0.241_{\pm 0.015}$	7
TabUnite-i2bFlow	$\mathbf{0.125}_{\pm 0.003}$	$86.41_{\pm 0.016}$	$\mathbf{90.95}_{\pm 0.106}$	$91.65_{\pm 0.067}$	$\mathbf{39.30}_{\pm 0.074}$	**1**
TabUnite-dicFlow	$0.140_{\pm 0.003}$	$86.13_{\pm 0.022}$	$\mathbf{90.49}_{\pm 0.101}$	$\mathbf{98.15}_{\pm 0.060}$	$36.16_{\pm 0.047}$	**2**
	Beijing					**Overall Rank**
	RMSE ↓	CDE ↑	PCC ↑	α ↑	β ↑	
TabDDPM	$1.91_{\pm 0.680}$	$66.98_{\pm 22.6}$	$61.63_{\pm 24.3}$	$33.99_{\pm 46.1}$	$19.89_{\pm 24.9}$	7
oheDDPM	$2.07_{\pm 0.697}$	$48.88_{\pm 2.26}$	$44.70_{\pm 3.61}$	$2.74_{\pm 0.78}$	$3.43_{\pm 2.05}$	8
i2bDDPM	$0.662_{\pm 0.017}$	$82.17_{\pm 0.27}$	$69.95_{\pm 0.60}$	$57.78_{\pm 0.83}$	$27.15_{\pm 3.56}$	5
dicDDPM	$0.960_{\pm 0.100}$	$84.23_{\pm 1.46}$	$69.07_{\pm 2.26}$	$74.73_{\pm 12.5}$	$12.74_{\pm 3.89}$	6
TabFlow	$0.583_{\pm 0.018}$	$96.57_{\pm 0.07}$	$94.10_{\pm 0.16}$	$\mathbf{96.16}_{\pm 0.95}$	$58.43_{\pm 1.22}$	3
oheFlow	$0.741_{\pm 0.017}$	$85.45_{\pm 0.98}$	$75.39_{\pm 1.96}$	$84.98_{\pm 6.39}$	$20.45_{\pm 1.71}$	4
TabUnite-i2bFlow	$\mathbf{0.538}_{\pm 0.007}$	$\mathbf{97.47}_{\pm 0.33}$	$\mathbf{96.23}_{\pm 0.39}$	$96.08_{\pm 1.45}$	$\mathbf{61.02}_{\pm 0.59}$	**2**
TabUnite-dicFlow	$\mathbf{0.559}_{\pm 0.009}$	$\mathbf{98.15}_{\pm 0.17}$	$\mathbf{96.27}_{\pm 0.31}$	$\mathbf{97.64}_{\pm 0.55}$	$\mathbf{60.69}_{\pm 0.40}$	**1**

from the numerical columns. We evaluate the synthesis of these categorical variables by taking the absolute value of the difference between the real value and the synthesised value. More details can be found in Appendix C.1. Our result in Figure 3b depicts the accuracy of the generated categorical columns over the number of training iterations. It illustrates that training both TabUnite models is stable and converges at a higher accuracy when compared to TabDDPM while remaining competitive with TabFlow.

4.3 Ablation Study: Encoding Scheme and Model Choice

To further validate the effectiveness of Analog Bits and Dictionary encoding schemes, as well as Flow Matching as our generative model, we conduct an ablation study to isolate the generative model while varying the encoding methods among Analog Bits, Dictionary, separate modelling, and one-hot encoding. We also perform the reverse, isolating the encoding schemes while varying the generative models between Flow Matching and DDPM. The real-world dataset we select for comparison is "Beijing" since it has a good amount of samples ($43,824$) as well as a balanced set of continuous (7) and categorical (5) features. However, an issue is that a vast majority of these publicly available datasets from the UCI machine learning repository (Dua & Graff, 2017) as well as other databases (Vanschoren et al., 2013) lack datasets with a large number of samples ($>$ 100k) and mixed features ($>$ 15 continuous and categorical features). Furthermore, accessing high-dimensional real-world datasets with heterogeneous features can be challenging. For instance, the PLCO dataset (Gohagan et al., 2000) requires 1-4 weeks for access approval due to ethical considerations and patient privacy protocols, and the MAGGIC dataset (Pocock et al., 2013) involves stringent access requests. Therefore, the need for curating publicly available large datasets with mixed features remains crucial for determining the effectiveness of our categorical encoding schemes.

Curation of a Large-Scaled Mixed Synthetic Dataset. A considerably larger dataset is the US Census Data (1990) (Meek et al., 2001) which contains $2,458,285$ samples and 61 features. However, these samples consist of only categorical variables. To incorporate continuous features, we begin by converting ordinal categorical features into continuous features. With the remaining non-ordinal categorical features, we select a subset and convert them to continuous using Frequency Encoding. Lastly, we leverage a synthetic data generation model (Chen et al., 2018; Si et al., 2023) to create continuous composite indicators (OECD et al., 2008) that can help capture interactions between



(a) AUC vs. Sampling Speed (NFEs) (b) Avg. Error vs. Sampling Speed (NFEs)

Figure 4: Synthetic Data Quality vs. Sampling Speed of TabUnite (i2bFlow/dicFlow), TabSYN and TabDDPM on the Adult dataset. TabUnite converges to its best AUC/Average Error in much fewer NFEs when compared to TabSyn and TabDDPM.

different aspects of the data. The synthetic continuous data are then generated per the following two polynomials: $\text{Syn1} = \exp(x_i x_j)$ and $\text{Syn2} = \exp(\sum_{i=1}^{3}(x_i^2 - 4))$ before applying a logistic function $\frac{1}{1+logit(\mathbf{X})}$. Finally, we concatenate our synthesised continuous features with the categorical. We have now constructed a Census Synthetic dataset comprised of 41 continuous features, 40 categorical features and $2,458,285$ samples. For a regression task, the label is "dIncome1" which is the annual income of an individual. Further details can be found in Appendix C.1.

Analysis. As observed in Table 2, both TabUnite methods achieve the highest ranking performances in both datasets across all the benchmarks. Solely comparing the performance of our encoding methods, we observed that our "i2b{ }" ({ } refers to either Flow or DDPM) and "dic{ }" encoding schemes outperform separated modelling (Tab{ }) and one-hot encoding (ohe{ }) in almost all metrics. Focusing on the "Beijing" dataset, TabUnite-dicFlow outperforms TabUnite-i2bFlow in 3/5 metrics. We hypothesise that since "Beijing" contains "combined wind direction" as an ordinal categorical feature, TabUnite-dicFlow should be able to outperform TabUnite-i2bFlow in several metrics depending on the feature's importance. Within our "Census Synthetic" dataset, we observe that TabUnite-i2bFlow dominates the performance when compared to TabUnite-dicFlow. This is because "Census Synthetic" contains no ordinal categorical features after converting them to continuous ones hence, it is rational for Analog Bits to have a better performance. On the other hand, comparing the performance of the generative models (Flow Matching vs. Diffusion) i.e. []Flow methods vs []DDPM methods ([] refers to either i2b, dic, Tab or ohe), Flow Matching achieves a superior performance. Additionally, we also investigate the sampling speed of our flow-based methods against TabSyn and TabDDPM. As shown in Figure 4, we observe that TabUnite converges to its best AUC/Average Error in much fewer NFEs when compared to TabSyn and TabDDPM. Therefore, the TabUnite methods have the best sampling efficiency, followed by TabSYN and TabDDPM.

5 Conclusion and Limitation

We propose an efficient encoding framework for tabular data generation that leverages effective categorical encoding schemes to unify the data space. This enables us to apply a single generative model that captures heterogeneous feature interrelationships, improving generation quality. Our models are curated by employing Analog Bits and Dictionary encoding that efficiently convert categorical variables into a dense and compact continuous representation, before applying Conditional Flow Matching to generate the data. To further strengthen our findings on our categorical embedding schemes, we curate a large-scale heterogeneous tabular dataset. Relative to the baselines, our TabUnite models outperform them across a wide range of datasets whilst evaluated on a broad suite of benchmarks. Additionally, leveraging Flow Matching greatly bolsters our sampling efficiency, saving computational cost and time. Overall, we justify our claim of applying efficient encoding methods to enable the application of a single/unified generative model on a coherent data space. A limitation of our methodology is that we have not extensively explored a continuous embedding scheme where we perform the reverse and unify the generative space into a categorical one. Inspired by (Ansari et al., 2024), we conduct initial explorations of time series tokenization to embed continuous features yet, our results are still inconclusive and left to future work.

References

Alaa, A. M., van Breugel, B., Saveliev, E., and van der Schaar, M. How faithful is your synthetic data? sample-level metrics for evaluating and auditing generative models, 2022.

Ansari, A. F., Stella, L., Turkmen, C., Zhang, X., Mercado, P., Shen, H., Shchur, O., Rangapuram, S. S., Arango, S. P., Kapoor, S., Zschiegner, J., Maddix, D. C., Wang, H., Mahoney, M. W., Torkkola, K., Wilson, A. G., Bohlke-Schneider, M., and Wang, Y. Chronos: Learning the language of time series, 2024.

Becker, B. and Kohavi, R. Adult. UCI Machine Learning Repository, 1996. DOI: https://doi.org/10.24432/C5XW20.

Borisov, V., Seßler, K., Leemann, T., Pawelczyk, M., and Kasneci, G. Language models are realistic tabular data generators, 2023.

Campbell, A., Yim, J., Barzilay, R., Rainforth, T., and Jaakkola, T. Generative flows on discrete state-spaces: Enabling multimodal flows with applications to protein co-design, 2024.

Chawla, N. V., Bowyer, K. W., Hall, L. O., and Kegelmeyer, W. P. Smote: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16:321–357, June 2002. ISSN 1076-9757. doi: 10.1613/jair.953. URL http://dx.doi.org/10.1613/jair.953.

Chen, J., Song, L., Wainwright, M. J., and Jordan, M. I. Learning to explain: An information-theoretic perspective on model interpretation, 2018.

Chen, R. T. Q., Rubanova, Y., Bettencourt, J., and Duvenaud, D. Neural ordinary differential equations, 2019.

Chen, T. and Guestrin, C. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16. ACM, August 2016. doi: 10.1145/2939672.2939785. URL http://dx.doi.org/10.1145/2939672.2939785.

Chen, T., Kornblith, S., Norouzi, M., and Hinton, G. A simple framework for contrastive learning of visual representations, 2020.

Chen, T., Zhang, R., and Hinton, G. Analog bits: Generating discrete data using diffusion models with self-conditioning. *arXiv preprint arXiv:2208.04202*, 2022.

Clore, J., Cios, K., DeShazo, J., and Strack, B. Diabetes 130-US hospitals for years 1999-2008. UCI Machine Learning Repository, 2014. DOI: https://doi.org/10.24432/C5230J.

Datta, A. Us health insurance dataset, Feb 2020. URL https://www.kaggle.com/datasets/teertha/ushealthinsurancedataset.

Dua, D. and Graff, C. Uci machine learning repository, 2017. URL http://archive.ics.uci.edu/ml.

Gohagan, J. K., Prorok, P. C., Hayes, R. B., Kramer, B. S., Prostate, Lung, C., and Team, O. C. S. T. P. The prostate, lung, colorectal and ovarian (plco) cancer screening trial of the national cancer institute: history, organization, and status. *Control Clin Trials*, 21(6 Suppl):251S–272S, Dec 2000. doi: 10.1016/s0197-2456(00)00097-0.

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial networks, 2014.

Gorishniy, Y., Rubachev, I., Khrulkov, V., and Babenko, A. Revisiting deep learning models for tabular data, 2023.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition, 2015.

Higgins, I., Matthey, L., Pal, A., Burgess, C., Glorot, X., Botvinick, M., Mohamed, S., and Lerchner, A. beta-VAE: Learning basic visual concepts with a constrained variational framework. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=Sy2fzU9gl.

Ho, J., Jain, A., and Abbeel, P. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Hoogeboom, E., Nielsen, D., Jaini, P., Forré, P., and Welling, M. Argmax flows and multinomial diffusion: Learning categorical distributions, 2021.

Jolicoeur-Martineau, A., Fatras, K., and Kachman, T. Generating and imputing tabular data via diffusion and flow-based gradient-boosted trees, 2024.

Karras, T., Aittala, M., Aila, T., and Laine, S. Elucidating the design space of diffusion-based generative models, 2022.

Kim, J., Lee, C., and Park, N. Stasy: Score-based tabular data synthesis. *arXiv preprint arXiv:2210.04018*, 2022.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

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

Kotelnikov, A., Baranchuk, D., Rubachev, I., and Babenko, A. Tabddpm: Modelling tabular data with diffusion models. In *International Conference on Machine Learning*, pp. 17564–17579. PMLR, 2023.

Krishnan, R. G., Liang, D., and Hoffman, M. On the challenges of learning with inference networks on sparse, high-dimensional data, 2017.

Lee, C., Kim, J., and Park, N. Codi: Co-evolving contrastive diffusion models for mixed-type tabular synthesis. In *International Conference on Machine Learning*, pp. 18940–18956. PMLR, 2023.

Lipman, Y., Chen, R. T., Ben-Hamu, H., Nickel, M., and Le, M. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

Liu, T., Qian, Z., Berrevoets, J., and van der Schaar, M. GOGGLE: Generative modelling for tabular data by learning relational structure. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=fPVRcJqspu.

Liu, X., Gong, C., and Liu, Q. Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003*, 2022.

Mairal, J., Ponce, J., Sapiro, G., Zisserman, A., and Bach, F. Supervised dictionary learning. In Koller, D., Schuurmans, D., Bengio, Y., and Bottou, L. (eds.), *Advances in Neural Information Processing Systems*, volume 21. Curran Associates, Inc., 2008. URL https://proceedings.neurips.cc/paper_files/paper/2008/file/c0f168ce8900fa56e57789e2a2f2c9d0-Paper.pdf.

Mairal, J., Bach, F., Ponce, J., and Sapiro, G. Online dictionary learning for sparse coding. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pp. 689–696, New York, NY, USA, 2009. Association for Computing Machinery. ISBN 9781605585161. doi: 10.1145/1553374.1553463. URL https://doi.org/10.1145/1553374.1553463.

McCann, R. J. A convexity principle for interacting gases. *Advances in Mathematics*, 128(1): 153–179, 1997. ISSN 0001-8708. doi: https://doi.org/10.1006/aima.1997.1634. URL https://www.sciencedirect.com/science/article/pii/S0001870897916340.

Meek, C., Thiesson, B., and Heckerman, D. US Census Data (1990). UCI Machine Learning Repository, 2001. DOI: https://doi.org/10.24432/C5VP42.

Minieri, A. Synthetic data for privacy preservation - part 2. https://www.clearbox.ai/blog/2022-06-07-synthetic-data-for-privacy-preservation-part-2, 2022. Accessed: 2024-05-20.

Moro, S., Rita, P., and Cortez, P. Bank Marketing. UCI Machine Learning Repository, 2012. DOI: https://doi.org/10.24432/C5K306.

OECD, Union, E., and European Commission, J. R. C. *Handbook on Constructing Composite Indicators: Methodology and User Guide*. OECD Publishing, 2008. doi: https://doi.org/10.1787/9789264043466-en. URL https://www.oecd-ilibrary.org/content/publication/9789264043466-en.

Onishi, S. and Meguro, S. Rethinking data augmentation for tabular data in deep learning. *arXiv preprint arXiv:2305.10308*, 2023.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Pocock, S. J., Ariti, C. A., McMurray, J. J. V., Maggioni, A., Køber, L., Squire, I. B., Swedberg, K., Dobson, J., Poppe, K. K., Whalley, G. A., Doughty, R. N., and in Chronic Heart Failure, M.-A. G. G. Predicting survival in heart failure: a risk score based on 39 372 patients from 30 studies. *European Heart Journal*, 34(19):1404–1413, May 2013. doi: 10.1093/eurheartj/ehs337.

Poslavskaya, E. and Korolev, A. Encoding categorical data: Is there yet anything 'hotter' than one-hot encoding?, 2023.

Rabaey, P., Deleu, J., Heytens, S., and Demeester, T. Clinical reasoning over tabular data and text with bayesian networks. *arXiv preprint arXiv:2403.09481*, 2024.

Rezende, D. J. and Mohamed, S. Variational inference with normalizing flows, 2016.

Sauber-Cole, R. and Khoshgoftaar, T. M. The use of generative adversarial networks to alleviate class imbalance in tabular data: a survey. *Journal of Big Data*, 9(1):98, 2022.

SDMetrics. Detection metrics (single table) - sdmetrics documentation, 2024. URL https://docs.sdv.dev/sdmetrics/metrics/metrics-in-beta/detection-single-table. Accessed: 2024-05-20.

Sennrich, R., Haddow, B., and Birch, A. Neural machine translation of rare words with subword units, 2016.

Si, J. Y. H., Cooper, M., Cheng, W. Y., and Krishnan, R. Interpretabnet: Enhancing interpretability of tabular data using deep generative models and large language models. In *NeurIPS 2023 Second Table Representation Learning Workshop*, 2023. URL https://openreview.net/forum?id=kzR5Cj5blw.

Song, J., Meng, C., and Ermon, S. Denoising diffusion implicit models, 2022.

Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., and Poole, B. Score-based generative modeling through stochastic differential equations, 2021.

Tong, A., Malkin, N., Huguet, G., Zhang, Y., Rector-Brooks, J., Fatras, K., Wolf, G., and Bengio, Y. Improving and generalizing flow-based generative models with minibatch optimal transport. In *ICML Workshop on New Frontiers in Learning, Control, and Dynamical Systems*, 2023.

van Breugel, B. and van der Schaar, M. Why tabular foundation models should be a research priority, 2024.

Vanschoren, J., van Rijn, J. N., Bischl, B., and Torgo, L. Openml: networked science in machine learning. *SIGKDD Explorations*, 15(2):49–60, 2013. doi: 10.1145/2641190.2641198. URL http://doi.acm.org/10.1145/2641190.264119.

Xu, L., Skoularidou, M., Cuesta-Infante, A., and Veeramachaneni, K. Modeling tabular data using conditional gan, 2019.

Yoon, J., Jordon, J., and van der Schaar, M. INVASE: Instance-wise variable selection using neural networks. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=BJg_roAcK7.

Zhang, H., Zhang, J., Srinivasan, B., Shen, Z., Qin, X., Faloutsos, C., Rangwala, H., and Karypis, G. Mixed-type tabular data synthesis with score-based diffusion in latent space. *arXiv preprint arXiv:2310.09656*, 2023.

Appendix

Contents

A Algorithms

Algorithms 1 and 2 describe the training and sampling Flow Matching process of TabUnite. For more information regarding Flow Matching, please refer to “Flow Matching for Generative Modeling" (Lipman et al., 2022) or “Improving and Generalizing Flow-Based Generative Models with Minibatch Optimal Transport" (Tong et al., 2023).

Algorithm 1 TabUnite: Training Flow Matching using CFM

```
Sample initial data points x_1 ~ q(x_1)
Initialize vector field v_t(x) and parameters θ
while not converged do
    Sample time step t ~ U([0, 1])
    Sample x ~ p_t(x|x_1)
    Calculate true vector field u_t(x|x_1) as per Eq. 3
    Compute loss L_CFM(θ) = E|v_t(x) − u_t(x|x_1)|^2
    Update θ using gradient descent to minimize L_CFM(θ)
end while
```

Algorithm 2 TabUnite: Sampling Flow Matching using CFM

```
Sample x ~ N(0, I) (start with the noise distribution)
Set t_max = T and initialize x_T = x
for i = T, . . . , 1 do
    Use ψ_t to map x_T to x_{t_{i−1}} using the learned vector field u_t
    Compute x_{t_{i−1}} with ψ_{t_i}(x_T) = σ_{t_i}(x_1)x_T + μ_{t_i}(x_1)
    Update x_T = x_{t_{i−1}}
end for
x_0 is a synthetic sample generated by CFM
```

B Architecture

B.1 Flow Matching MLP

Figure 5 illustrates the MLP architecture used as part of our Flow Matching network, also used in TabDDPM (Kotelnikov et al., 2023) and TabSYN (Zhang et al., 2023), which is based on Gorishniy et al. (2023).

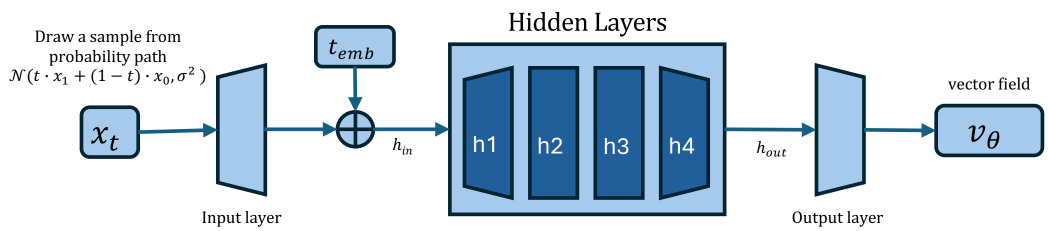


Figure 5: The MLP architecture used in the Flow Matching process. The neural network takes in a batch of samples drawn from the probability path at time t's sampled from $\mathcal{U}(0,1)$ to create a vector field v_θ that represents a continuous normalizing flow from pure noise to our data distribution $p_1(x_1)$.

The input layer projects the batch of tabular data input samples x_t, each with dimension d_{in}, to the dimensionality d_t of our time step embeddings t_{emb} through a fully connected layer. This is so that we may leverage temporal information, which is appended to the result of the projection in the form of sinusoidal time step embeddings.

$$h_{in} = FC_{d_t}(x_t) + t_{emb} \tag{5}$$

The hidden layers $h1$, $h2$, $h3$, and $h4$ are fully connected networks used to learn and create the vector field. The output dimension of each layer is chosen as d_t, $2d_t$, $2d_t$, and d_t respectively. On top of the FC networks, each layer also consists of an activation function followed by dropout, as seen in the formulas below. This formulation is repeated for each hidden layer, at the end of which we obtain h_{out}. The exact activations, dropout, and other hyperparameters chosen are shown in Table 3.

$$h_1 = Dropout(Activation(FC(h_{in}))) \tag{6}$$

At last, the output layer transforms h_{out}, of dimension t_{emb} back to dimension d_{in} through a fully connected network, which now represents the vector field v_θ.

$$v_\theta = FC_{d_{in}}(h_{out}) \tag{7}$$

B.2 Hyperparameters

We generally utilise the same hyperparameters as TabSYN (Zhang et al., 2023) and TabDDPM (Kotelnikov et al., 2023) for comparability. The exact hyperparameters selected for our models are shown below in Table 3.

Table 3: TabUnite Hyperparameters.

General		**Flow Matching MLP**	
Hyperparameter	**Value**	**Hyperparameter**	**Value**
Training Iterations	100,000	Timestep embedding dimension d_t	1024
Flow Matching Timesteps	1,000	Activation	ReLU
Learning Rate	1e−4	Dropout	0.0
Weight Decay	5e−4	Hidden layer dimension $[h1, h2, h3, h4]$	[1024, 2048, 2048, 1024]
Batch Size	4096		

C Experimental Details

The following delineates the foundation of our experiments:

- Codebase: Python & PyTorch
- GPU: Nvidia RTX 3090, 24GB VRAM
- Optimizer: Adam (Kingma & Ba, 2014)

Experiment Table Details

For Tables 1 and 2, the Overall Rank is calculated by first ranking them individually within each benchmark (row-wise), then averaging their ranks for each method across the benchmarks (column-wise), before rounding the ranks to the nearest integer.

In Table 1 and Appendix Tables, all reported results of baselines in our experiments are taken from Zhang et al. (2023), except for TabSYN and TabDDPM, whose results are reproduced utilising the public repository: `https://github.com/amazon-science/tabsyn`. Additionally, for Table 1, we decided to rerun GReaT in the same original setting (1 Train, 20 Samples) for the Adult dataset as TabSYN's reported results (0.913 ± 0.003) were unusually high. All reported results follow TabSYN's 1 Training and 20 Sampling trial setting. Note that TabDDPM collapses on the News dataset for all the benchmarks.

In Table 2, we limit ourselves to only one real-world dataset + our curated "Census Synthetic" dataset. Additionally, we computed 1 Training and 3 Sampling trials for our error bars. Lastly, Pair-Wise Column Correlation for the "Census Synthetic" dataset is evaluated on a 10% subsample. These reasons are due to the fact that it is computationally costly to compute results for the diffusion-based models.

C.1 Datasets

Real World Datasets

Experiments were conducted with a total of 6 tabular datasets from the UCI Machine Learning Repository (Dua & Graff, 2017) with a (CC-BY 4.0) license. Classification tasks were performed on the Adult, Default, Magic, and Shoppers datasets, while regression tasks were performed on the Beijing and News datasets. Each dataset was split into training, validation, and testing sets with a ratio of 8:1:1, except for the Adult dataset, whose official testing set was used and the remainder split into training and validation sets with an 8:1 ratio. The resulting statistics of each dataset are shown below in Table 4. Note that the target column indicates the specific operation applied to each dataset: binary classification for a categorical target with two classes, multiclass classification for a categorical target with more than two classes, and regression for a numerical target feature. Some detailed information as well as the statistics of the datasets are shown in Tables 4 and 5 respectively.

Table 4: Statistics of datasets. "# Num" stands for the number of numerical columns, and "# Cat" stands for the number of categorical columns.

Dataset	# Rows	# Num	# Cat	# Train	# Validation	# Test	Task Type
Adult	48,842	6	9	28,943	3,618	16,281	Binary Classification
Default	30,000	14	11	24,000	3,000	3,000	Binary Classification
Shoppers	12,330	10	8	9,864	1,233	1,233	Binary Classification
Magic	19,019	10	1	15,215	1,902	1,902	Binary Classification
Beijing	43,824	7	5	35,058	4,383	4,383	Regression
News	39,644	46	2	31,714	3,965	3,965	Regression
Census Synthetic	2,458,285	41	40	1,966,621	245,827	245,829	Regression

Synthetic Toy Datasets

Qualitative Toy Dataset: The dataset consists of four columns, with the first two columns representing numerical data point coordinates. Subsequently, the third column categorizes the data points into five circles whereas the last column indicates the 5 colours each data point can be classified into.

Table 5: Details of datasets. The "Feature Information" column details the contents of the dataset and how it is curated. The "Prediction Task" column describes the model's objective on that dataset.

Dataset	Feature Information	Prediction Task
Adult	Demographic and occupational variables from census data	Whether an individual's income exceeds $50,000
Default	Demographic and account-specific data collected from credit card clients	Whether an individual will default on their credit card payments next month
Shoppers	Internet users' browser session information	Whether the user will engage in online shopping
Magic	Generated events simulating the imaging of gamma-ray air showers	Predict the type of high-energy gamma particles in the atmosphere
Beijing	Hourly atmospheric PM2.5 and meteorological data readings at the U.S. Embassy in Beijing	Predict future PM2.5 readings
News	Various features from the news site Mashable's published articles	The number of "shares" articles will have on social media
Census Synthetic	1990 Census Demographics of the US Population	Annual Income of an individual

Therefore, each row in the dataset contains 2 numerical features and 2 categorical features. A total of $10,000$ samples are generated for this dataset.

Quantitative Toy Dataset: To quantify our model's ability to generate high-quality data, we generate a synthetic toy dataset with 11 numerical features, all drawn from a unit Gaussian distribution, to represent a complex underlying data distribution. From these numerical features, we derive six categorical variables by applying a variety of transformations, the details of which are described by the equations below.

$$
\begin{aligned}
x_1^{cat} &= x_0^{num} \cdot x_1^{num} \\
x_2^{cat} &= (x_2^{num})^2 + (x_3^{num})^2 + (x_4^{num})^2 + (x_5^{num})^2 - 4 \\
x_3^{cat} &= -10 \cdot \sin(2 \cdot x_6^{num}) + 2 \cdot |x_7^{num}| + x_8^{num} - e^{-x_9^{num}} \\
x_4^{cat} &= (x_9^{num} < 0) \cdot x_1^{cat} + (1 - (x_9^{num} < 0)) \cdot x_2^{cat} \\
x_5^{cat} &= (x_9^{num} < 0) \cdot x_1^{cat} + (1 - (x_9^{num} < 0)) \cdot x_3^{cat} \\
x_6^{cat} &= (x_9^{num} < 0) \cdot x_2^{cat} + (1 - (x_9^{num} < 0)) \cdot x_3^{cat}
\end{aligned}
\tag{8}
$$

Following the transformations, tanh activation functions are applied followed by digitization to 10 separate bins. A total of $10,000$ samples are generated for this dataset, resulting in our discrete categorical variables. We quantify the performance of our models by examining the fidelity of generating these categorical variables. The scoring is determined by taking the absolute value of the difference between the real and synthesized values.

We perform three trial experiments for each method and report their mean and standard deviation. Note that in the quantitative experiments, we use a DDIM sampler for TabDDPM thus, the results are slightly worse than those we reported in our previous tables.

Census Synthetic Dataset

The US Census Data (1990) (Meek et al., 2001) ((CC-BY 4.0) license) contains $2,458,285$ samples and 61 features (excluding "dIncome2" to "dIncome8" since they are redundant). However, these samples consist of only categorical variables. To incorporate continuous features, we begin by converting the following ordinal categorical features into continuous features:

- Annual income: dIncome1
- Earnings from employment: dRearning
- Age: dAge

- English proficiency: iEnglish
- Hours worked in 1989: dHour89
- Hours worked per week: dHours
- Travel time to work: dTravtime
- Years spent schooling: iYearsch
- Years spent working: iYearwrk

A total of 9 ordinal categorical features are converted. With the remaining non-ordinal categorical features, we select 12 additional categorical features and convert them to continuous using Frequency Encoding yielding us 21 continuous features in total. We consider features that are likely to have a variety of categories and could benefit from a frequency-based transformation. For instance, occupation covers a wide range of jobs and ancestry covers many different backgrounds. The features are as follows:

- Primary ancestry: dAncstry1
- Secondary ancestry: dAncstry2
- Citizenship status: iCitizen
- Marital status: iMarital
- Hispanic origin: dHispanic
- Class of worker: iClass
- Place of birth: dPOB
- Occupation: dOccup
- Industry: dIndustry
- Mobility status: iMobility
- Relationship to head of household: iRelat1
- Sex: iSex

Lastly, to balance out the remaining categorical features 40 with the 21 continuous ones, we leverage a synthetic data generation model (Chen et al., 2018; Yoon et al., 2019; Si et al., 2023) to generate 20 more continuous features based on the converted continuous features. We create continuous composite indicators (OECD et al., 2008) by combining our curated continuous features in sets of 2 or 3 that can help capture interactions and relationships between different aspects of the data. An example is a gender and earnings indicator that shows income disparities. Here are the composite indicators:

- Work hours (Hours worked per week and Hours worked in 1989): dHours, dHour89
- Educational attainment with age (Age and Years of schooling): dAge, iYearsch
- Language skills based on birthplace (English proficiency and Place of birth): iEnglish, dPOB
- Demographic relationships (Citizenship status and Hispanic origin): iCitizen, dHispanic
- Commuting patterns (Travel time to work and Years worked): dTravtime, iYearwrk
- Family structure (Marital status and Relationship to household head): iMarital, iRelat1
- Employment characteristics (Industry and Occupation): dIndustry, dOccup
- Income disparities (Gender and Earnings): iSex, dRearning
- Migration patterns (Mobility status and Citizenship): iMobility, iCitizen
- Heritage (Primary and Secondary Ancestry): dAncstry1, dAncstry2
- Career dedication (Hours worked per week, Hours worked in 1989, and Travel time to work): dHours, dHour89, dTravtime
- Career progression (Age, years of schooling, and years worked): dAge, iYearsch, iYearwrk
- Cultural integration (English proficiency, place of birth, and citizenship): iEnglish, dPOB, iCitizen

- Household dynamics (Marital status, relationship to household head, and mobility status): iMarital, iRelat1, iMobility
- Job characteristics (Industry, Occupation, and Earnings): dIndustry, dOccup, dRearning
- Income trends (Gender, Earnings, and Age): iSex, dRearning, dAge
- Heritage and immigration status (Primary and Secondary heritage, and Citizenship): dAncstry1, dAncstry2, iCitizen
- Demographic patterns (Hispanic origin, Relationship to household head, and Age): dHispanic, iRelat1, dAge
- Job location and stability (Travel time, Years worked, and Occupation): dTravtime, iYearwrk, dOccup
- Education's impact on earnings (Years of schooling, Years worked, and Earnings): iYearsch, iYearwrk, dRearning

Before generating these composite indicators, we first apply a Standard scaler to the converted continuous features since the input features are "generated from a Gaussian distribution ($X \sim N(0, I)$)" (per (Chen et al., 2018)). The synthetic continuous data are then generated according to the following two polynomials:

- Syn1 $= \exp(\mathbf{X}_i \mathbf{X}_j)$
- Syn2 $= \exp(\sum_{i=1}^{3}(\mathbf{X}_i^2 - 4))$

where the first set consists of 10 indicators derived from pairs of variables following Syn1 and the second set consists of 10 indicators derived from triples of variables following Syn2. These composite indicators are then transformed using the logistic function $\frac{1}{1+\exp(\boldsymbol{X})}$. Finally, we merge our continuous features with the categorical features to create a comprehensive "Census Synthetic" dataset. The "Census Synthetic" dataset we construct comprises of 41 continuous features, 40 categorical features and $2,458,285$ samples. For a regression task, the label is "dIncome1" which is the Annual income of an individual. Note that the dataset will be released with a CC-BY 4.0 license.

C.2 Additional Details on Baselines: Predefined Models.

TabUnite's performance is evaluated in comparison to previous works in mixed-type tabular data generation. This includes CTGAN and TVAE (Xu et al., 2019), GOGGLE (Liu et al., 2023), GReaT (Borisov et al., 2023), STaSy (Kim et al., 2022), CoDi (Lee et al., 2023), TabDDPM (Kotelnikov et al., 2023), and TabSYN (Zhang et al., 2023). The underlying architectures and implementation details of these models are presented below in Table 7.

C.3 Additional Details on Ablations: Encoding schemes and generative models (Flow/Diffusion).

On top of the models developed by previous related works in mixed-type tabular data synthesis, we developed baselines that would provide a direct and analogous comparison to justify flow-matching and our particular encoding methods. This includes the flow-matching-based one-hotFlow (oheFlow), TabFlow, and the DDPM-based i2bDDPM, dicDDPM, and one-hotDDPM (oheDDPM).

Our DDPM-based baseline methods (i2bDDPM, dicDDPM, and oheDDPM) primarily inherit the design and implementation of TabDDPM (Kotelnikov et al., 2023). Whereas TabDDPM leverages two separate diffusion models, namely Gaussian diffusion and Multinomial diffusion, we devise i2bDDPM, dicDDPM, and oheDDPM to rely solely on Gaussian Diffusion. This is because their corresponding methods of Analog Bits, Dictionary Encoding, and One-Hot Encoding allow us to perform diffusion in a unified data space. Implementation of these methods is done by simply altering the data processing stage of the model. The DDPM architecture is largely kept the same.

Our Flow-based baseline methods (oheFlow, TabFlow) are extended from the TabUnite architecture, which consists of i2bFlow and dicFlow. oheFlow, as the name suggests, utilizes One-Hot Encoding in its data processing stage. Tabflow, on the other hand, mirrors the idea of TabDDPM in that two separate models are used: one for learning categorical features and the other for learning numerical

Table 7: Comparison of previous methods in Tabular Data Synthesis.

Method	Model[1]	Type[2]	Categorical Encoding	Numerical Encoding	Additional Techniques
CTGAN	GAN	U	One-Hot Encoding	Scaled Bayesian Gaussian Mixture	Mode-specific normalization to represent complex distributions & conditional generation to address data imbalances
TVAE	VAE	U	One-Hot Encoding	Scaled Bayesian Gaussian Mixture	Mode-specific normalization & conditional generation
GOGGLE	VAE + GNN	U	One-Hot Encoding	-	Learning relational structures among features graphically through an adjacency matrix
GReaT	Autoregressive GPT	U	Byte-Pair Encoding[3]	Byte-Pair Encoding[3]	Textual Encoder which converts data into natural language, followed by Feature Order Permutation and Fine-tuning
STaSy	Score-based Diffusion	U	One-Hot Encoding	Min-max scaler	Self-paced learning and fine-tuning
CoDi	DDPM/ Multinomial Diffusion	S	One-Hot Encoding	Min-max scaler	Model Inter-conditioning and Contrastive learning to learn dependencies between categorical and numerical data
TabDDPM	DDPM/ Multinomial Diffusion	S	One-Hot Encoding	Quantile Transformer	Concatenation of numerical and categorical features
TabSYN	VAE + EDM	U	One-Hot	Quantile Transformer	Feature Tokenizer and Transformer encoder to learn cross-feature relationships with adaptive loss weighing to increase reconstruction performance
TabUnite-i2BFlow	Flow Matching	U	Analog Bits	Quantile Transformer	Concatenation of numerical and categorical features encoded with TabUnite's embedding scheme
TabUnite-dicFlow	Flow Matching	U	Dictionary	Quantile Transformer	Concatenation of numerical and categorical features encoded with TabUnite's embedding scheme

[1] The 'Model' Column indicates the underlying architecture used for the model. Options include Generative Adversarial Networks or GANs (Goodfellow et al., 2014), Variational Autoencoders or VAEs (Kingma & Welling, 2013), Denoising Diffusion Probabilistic Models or DDPMs (Ho et al., 2020), Multinomial Diffusion (Hoogeboom et al., 2021), EDM, as introduced in Karras et al. (2022).

[2] The 'Type' column indicates the data integration approach used in the model. 'U' denotes a unified data space where numerical and categorical data are combined after initial processing and fed collectively into the model. 'S' represents a separated data space, where numerical and categorical data are processed and fed into distinct models.

[3] Byte-Pair Encoding (Sennrich et al., 2016) is a tokenization method that iteratively merges the most frequent adjacent characters or character pairs into single tokens, creating a vocabulary of subwords that efficiently handles rare and unknown words in text processing.

features. Here, the implementation combines ordinary Flow Matching (Lipman et al., 2022) with Discrete Flow Matching (Campbell et al., 2024). The respective results of these two models are concatenated afterward to allow for the synthesis of mixed-type tabular data.

These methods all utilize the QuantileTransformer (Pedregosa et al., 2011) to process numerical data, which normalizes features to follow a uniform or normal distribution. This is done through sorting and ranking data points, and then mapping them to fit to the target distribution.

C.4 Benchmarks

In this section, we expand on the concrete formulations behind our benchmarks including machine learning efficiency, low-order statistics, and high-order metrics. We also provide an overview on the detection and privacy metrics used in our experiments. These comprehensive benchmarks as well as their implementations are identical to those established by TabSYN (Zhang et al., 2023), ensuring a direct and accurate comparison.

Machine Learning Efficiency

AUC (Area Under Curve) is used to evaluate the efficiency of our model in binary classification tasks. It measures the area under the Receiver Operating Characteristic (or ROC) curve, which plots the True Positive Rate against the False Positive Rate. AUC may take values in the range [0,1]. A higher AUC value suggests that our model achieves a better performance in binary classification tasks and vice versa.

$$\mathrm{AUC} = \int_0^1 \mathrm{TPR}(\mathrm{FPR})\, d(\mathrm{FPR}) \tag{9}$$

RMSE (Root Mean Square Error) is used to evaluate the efficiency of our model in regression tasks. It measures the average magnitude of the deviations between predicted values ($\hat{y}_i$) and actual values (y_i). A smaller RMSE model indicates a better fit of the model to the data.

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{10}$$

Low-Order Statistics.

Column-wise Density Estimation between numerical features is achieved with the Kolmogorov-Smirnov Test (KST). The Kolmogorov-Smirnov statistic is used to evaluate how much two underlying one-dimensional probability distributions differ, and is characterized by the below equation:

$$\mathrm{KST} = \sup_x |F_1(x) - F_2(x)|, \tag{11}$$

where $F_n(x)$, the empirical distribution function of sample n is calculated by

$$\mathrm{F_n(x)} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{1}_{(-\infty,x]}(X_i) \tag{12}$$

Column-wise Density Estimation between two categorical features is determined by calculating the Total Variation Distance (TVD). This statistic captures the largest possible difference in the probability of any event under two different probability distributions. It is expressed as

$$\mathrm{TVD} = \frac{1}{2}\sum_{x\in X} |P_1(x) - P_2(x)|, \tag{13}$$

where $P_1(x)$ and $P_2(x)$ are the probabilities (PMF) assigned to data point x by the two sample distributions respectively.

Pair-wise Column Correlation between two numerical features is computed using the Pearson Correlation Coefficient (PCC). It assigns a numerical value to represent the linear relationship between two columns, ranging from -1 (perfect negative linear correlation) to +1 (perfect positive linear correlation), with 0 indicating no linear correlation. It is computed as:

$$\rho(x, y) = \frac{\mathrm{cov}(x, y)}{\sigma_x \sigma_y}, \tag{14}$$

To compare the Pearson Coefficients of our real and synthetic datasets, we quantify the dissimilarity in pair-wise column correlation between two samples

$$\text{Pearson Score} = \frac{1}{2}\mathbb{E}_{x,y}|\rho^1(x, y) - \rho^2(x, y)| \tag{15}$$

Pair-wise Column Correlation between two categorical features in a sample is characterized by a Contingency Table. This table is constructed by tabulating the frequencies at which specific combinations of the levels of two categorical variables work and recording them in a matrix format.

To Quantify the dissimilarity of contingency matrices between two different samples, we use the notion of the Contingency Score.

$$\text{Contingency Score} = \frac{1}{2}\sum_{\alpha \in A}\sum_{\beta \in B}|P_{1,(\alpha,\beta)} - P_{2,(\alpha,\beta)}|, \tag{16}$$

where α and β describe possible categorical values that can be taken in features A and B. $P_{1,(\alpha,\beta)}$ and $P_{2,(\alpha,\beta)}$ refer to the contingency tables representing the features α and β in our two samples, which in this case corresponds to the real and synthetic datasets.

In order to obtain the column-wise density estimation and pair-wise correlation between a categorical and a numerical feature, we bin the numerical data into discrete categories before applying TVD and Contingency score respectively to obtain our low-order statistics.

We utilize the implementation of these experiments as provided by the SDMetrics library[1].

High-Order Statistics

We utilize the implementations of High-Order Statistics as provided by the synthcity[2] library.

α-precision measures the overall fidelity of the generated data and is an extension of the classical machine learning quality metric of "precision". This formulation is based on the assumption that α fraction of our real samples are characteristic of the original data distribution and the rest are outliers. α-precision therefore quantifies the percentage of generated synthetic samples that match α fraction of real samples (Alaa et al., 2022).

β-recall characterizes the diversity of our synthetic data and is similarly based on the quality metric of "recall". β-recall shares a similar assumption as α-precision, except that we now assume that β fraction of our synthetic samples are characteristic of the distribution. Therefore, this measure obtains the fraction of the original data distribution that is represented by the β fraction of our generated samples (Alaa et al., 2022).

Detection Metric: Classifier Two-Sample Test (C2ST)

The Classifier Two-Sample Test, a detection metric, assesses the ability to distinguish real data from synthetic data. This is done through a machine learning model that attempts to label whether a data point is synthetic or real. The score ranges from 0 to 1 where a score closer to 1 is superior, as it indicates that the machine learning model cannot concretely identify whether the data point in question is real or generated. We select logistic regression as our machine learning model in this case, using the implementation provided by SDMetric (SDMetrics, 2024).

Privacy Metric: Distance to Closest Record (DCR)

The Distance to Closest Record metric quantifies the distance between each generated sample to our training set. The score is calculated as the proportion of synthetic data points that have a closer match

[1] https://github.com/sdv-dev/SDMetrics

[2] https://github.com/vanderschaarlab/synthcity

to the real data set compared to the holdout set. A score close to 50% is ideal, as it indicates that our generated sample represents the underlying distribution of our training samples without revealing specific points present in the dataset.

D Further Experimental Results

We run all experiments outlined in this section on at least 4 main models: TabUnite-i2bFlow, TabUnite-dicFlow, TabSYN(Zhang et al., 2023), and TabDDPM(Kotelnikov et al., 2023) due to their competitive performance on our MLE experiments as seen in Table 1 as well as prior literature (Zhang et al., 2023). Unless otherwise stated, we use experimental results collected by TabSYN's author for all other model benchmarks. The metrics and error bars shown in the tables in this section are derived from the mean and standard deviation of experiments performed on 20 randomly sampled sets of synthetic data.

D.1 Training and Sampling Time

We showcase the training and sampling durations for TabUnite and other competitive diffusion-based baseline models obtained from our experiments in this section. Experiments for all datasets outlined in table Table 9 are performed in the computing environment described in section Appendix C. For the two TabUnite methods (i2bFlow and dicFlow) and the flow-matching-based baseline TabFlow, we use the hyperparameters as specified in Table 3. For all non-TabUnite methods, we follow the recommended parameters set forth by their respective authors, see (Kim et al., 2022), (Lee et al., 2023), (Kotelnikov et al., 2023), and (Zhang et al., 2023).

Table 9: Training and Sampling Times of TabUnite and baselines on the Beijing Dataset. The hyperparameters used to run these experiments are included in Table 3.

Model	Training Time (s)	Training Steps	Training Time/step (s)	Sampling Time (s)
STaSy	8029.92	10,000	0.803	17.39
CoDi	30342.05	20,000	1.517	11.15
TabDDPM	4188.56	100,000	0.042	73.82
TabSYN	3671.48	4,000+625	0.509	5.97
TabFlow	6772.25	100,000	0.068	3.87
TabUnite-i2bFlow	5182.89	100,000	0.052	3.80
TabUnite-dicFlow	4380.02	100,000	0.044	3.40

Note that for TabSYN, the VAE is trained for 4000 steps, taking 3352.70 seconds to complete. Early stopping when training the EDM model is reached at 625/10001 epochs, finishing in an additional 318.78 seconds. The training times presented in the figure are the sum of the times required to complete training on both the VAE and diffusion models.

D.2 Low-order statistics: Column-wise density estimation and Pair-wise column correlation

The results for our Low-Order metrics tests can be found in Table 10 and Table 11.

Table 10: Error rate (%) of column-wise density estimation.Values bolded in **red** and **blue** are the best and second best-performing models respectively for each dataset.

Method	Adult	Default	Shoppers	Magic	Beijing	News	Overall Rank
SMOTE	1.60±0.23	1.48±0.15	2.68±0.19	0.91±0.05	1.85±0.21	5.31±0.46	N/A
CTGAN	16.84± 0.03	16.83±0.04	21.15±0.10	9.81±0.08	21.39±0.05	16.09±0.02	8
TVAE	14.22±0.08	10.17±0.05	24.51±0.06	8.25±0.06	19.16±0.06	16.62±0.03	7
GOGGLE	16.97	17.02	22.33	1.90	16.93	25.32	6
GReaT	12.12±0.04	19.94±0.06	14.51±0.12	16.16±0.09	8.25±0.12	OOM	9
STaSy	11.29±0.06	5.77±0.06	9.37±0.09	6.29±0.13	6.71±0.03	6.89±0.03	4
CoDi	21.38±0.06	15.77± 0.07	31.84±0.05	11.56±0.26	16.94±0.02	32.27±0.04	10
TabDDPM	**1.37±0.05**	**2.06±0.06**	4.49±0.09	**2.64±0.19**	49.25±0.13	75.11±0.03	4
TabSYN	3.96±0.08	2.90± 0.04	**2.56±0.07**	2.65±0.12	**2.24±0.04**	5.74±0.05	3
TabUnite-i2bFlow	**1.19±0.05**	**2.17±0.09**	3.19±0.10	**2.54±0.20**	2.49±0.04	**2.81±0.03**	**1**
TabUnite-dicFlow	1.64±0.06	2.70± 0.07	**3.14±0.07**	3.09±0.19	**2.10±0.06**	**3.31±0.04**	**2**

Table 11: Error rate (%) of pair-wise column correlation score. Values bolded in **red** and **blue** are the best and second best-performing models respectively for each dataset.

Method	**Adult**	**Default**	**Shoppers**	**Magic**	**Beijing**	**News**	**Overall Rank**
SMOTE	3.28±0.29	8.41±0.38	3.56±0.22	3.16±0.41	2.39±0.35	5.38±0.76	N/A
CTGAN	20.23±1.20	26.95±0.93	13.08±0.16	7.00±0.19	22.95±0.08	5.37±0.05	7
TVAE	14.15±0.88	19.50±0.95	18.67±0.38	5.82±0.49	18.01±0.08	6.17±0.09	6
GOGGLE	45.29	21.94	23.90	9.47	45.94	23.19	9
GReaT	17.59±0.22	70.02±0.12	45.16±0.18	10.23±0.40	59.60±0.55	OOM	10
STaSy	14.51±0.25	**5.96±0.26**	8.49±0.15	6.61±0.53	8.00±0.10	3.07±0.04	4
CoDi	22.49±0.08	68.41±0.05	17.78±0.11	6.53±0.25	7.07±0.15	11.10±0.01	7
TabDDPM	**2.67±0.05**	13.56±0.16	11.89±0.09	**2.27±0.09**	50.76±0.08	15.65±0.23	5
TabSYN	6.64±0.15	12.44±1.02	**6.45±0.08**	3.19±0.12	5.80±0.13	4.16±0.03	3
TabUnite-i2bFlow	**2.95±0.37**	11.69±1.19	**6.04±0.55**	**3.18±0.46**	**5.71±0.10**	**2.48±0.03**	**1**
TabUnite-dicFlow	3.63±0.35	**11.46±1.78**	7.28±0.33	3.28±0.45	**5.65±0.13**	**2.74±0.09**	**2**

D.3 High-order metrics: α-precision and β-recall

The results for our High-Order metrics tests can be found in Table 12 and Table 13.

Note that similar to the results obtained in TabSYN's paper, TabDDPM also collapses on the News dataset in our experiments.

Table 12: Comparison of α-Precision scores. Higher values indicate superior results. Values bolded in **red** and **blue** are the best and second best-performing models respectively for each dataset.

Methods	**Adult**	**Default**	**Shoppers**	**Magic**	**Beijing**	**News**	**Overall Rank**
CTGAN	77.74±0.15	62.08±0.08	76.97±0.39	86.90±0.22	96.27±0.14	96.96±0.17	8
TVAE	98.17±0.17	85.57±0.34	58.19±0.26	86.19±0.48	97.20±0.10	86.41±0.17	7
GOGGLE	50.68	68.89	86.95	90.88	88.81	86.41	10
GReaT	55.79±0.03	85.90±0.17	78.88±0.13	85.46±0.54	**98.32±0.22**	-	8
STaSy	82.87±0.26	90.48±0.11	89.65±0.25	86.56±0.19	89.16±0.12	94.76±0.33	5
CoDi	77.58±0.45	82.38±0.15	94.95±0.35	85.01±0.36	98.13±0.38	87.15±0.12	6
TabDDPM	94.79±0.27	**98.27±0.34**	98.33±0.40	93.35±0.53	0.01±0.73	0.00±0.00	4
TabSYN	98.51±0.31	**98.73±0.20**	**98.80±0.36**	98.01±0.30	97.30±0.30	97.98±0.08	3
TabUnite-i2bFlow	**99.42±0.13**	97.08±0.33	**98.78±0.47**	**99.10±0.20**	97.60±0.27	**98.77±0.39**	**1**
TabUnite-dicFlow	**99.27±0.2**	96.16±0.34	97.34±0.55	**99.27±0.19**	**98.90±0.22**	**98.47±0.29**	**2**

Table 13: Comparison of β-Recall scores. Higher values indicate superior results. Values bolded in **red** and **blue** are the best and second best-performing models respectively for each dataset.

Methods	**Adult**	**Default**	**Shoppers**	**Magic**	**Beijing**	**News**	**Overall Rank**
CTGAN	30.80±0.20	18.22±0.17	31.80±0.350	11.75±0.20	34.80±0.10	24.97±0.29	9
TVAE	38.87±0.31	23.13±0.11	19.78±0.10	32.44±0.35	28.45±0.08	29.66±0.21	8
GOGGLE	8.80	14.38	9.79	9.88	19.87	2.03	10
GReaT	**49.12±0.18**	42.04±0.19	44.90±0.17	34.91±0.28	43.34±0.31	-	6
STaSy	29.21±0.34	39.31±0.39	37.24±0.45	53.97±0.57	54.79±0.18	39.42±0.32	4
CoDi	9.20±0.15	19.94±0.22	20.82±0.23	50.56±0.31	52.19±0.12	34.40±0.31	7
TabDDPM	50.74±0.37	46.90±0.35	**53.32±0.52**	46.26±0.35	0.02±0.68	0.00±0.00	5
TabSYN	45.13±0.23	44.30±0.29	48.68±0.57	45.28±0.40	55.50±0.21	35.70±0.18	3
TabUnite-i2bFlow	48.49±0.17	**47.43±0.33**	**54.47±0.57**	**67.60±0.28**	**60.34±0.20**	**50.89±0.27**	**2**
TabUnite-dicFlow	**51.34±0.25**	**50.75±0.34**	52.24±0.59	**66.93±0.19**	**60.66±0.21**	**50.07±0.29**	**1**

D.4 Detection metric: Classifier Two-Sample Test (C2ST)

The results for our C2ST tests can be found in Table 14. We are generally competitive with TabSYN and TabDDPM.

Table 14: Comparison of C2ST scores. Higher values indicate superior results. Values bolded in **red** and **blue** are the best and second best-performing models respectively for each dataset.

Methods	**Adult**	**Default**	**Shoppers**	**Magic**	**Beijing**	**News**	**Overall Rank**
CTGAN	0.5949	0.4875	0.7488	0.6728	0.7531	0.6947	7
TVAE	0.6315	0.6547	0.2962	0.7706	0.8659	0.4076	5
GOGGLE	0.1114	0.5163	0.1418	0.9526	0.4779	0.0745	8
GReaT	0.5376	0.4710	0.4285	0.4326	0.6893	-	9
STaSy	0.4054	0.6814	0.5482	0.6939	0.7922	0.5287	6
CoDi	0.2077	0.4595	0.2784	0.7206	0.7177	0.0201	10
TabDDPM	0.1263	**0.9844**	0.8545	**0.9951**	0.0380	0.0000	4
TabSYN	**0.9235**	**0.9664**	**0.9516**	**0.9526**	0.8937	0.7934	**1**
TabUnite-i2bFlow	0.7180	0.9407	0.8538	0.9304	**0.9304**	**0.9005**	3
TabUnite-dicFlow	**0.9004**	0.9275	**0.9176**	0.9514	**0.9477**	**0.8784**	**2**

D.5 Privacy metric: Distance to Closest Record

The results for our DCR tests can be found in Table 15. As observed, we remain competitive but do not outperform TabSYN as the best method under this metric. This aligns with our hypothesis where TabSYN leverages a latent space thus, resulting in a lossy compression, improving their DCR scores.

Table 15: Comparison of DCR. Results closer to 50% indicate better performance on the test. Values bolded in **red** and **blue** are the best and second best-performing models respectively for each dataset.

Methods	**Adult**	**Default**	**Shoppers**	**Magic**	**Beijing**	**News**	**Overall Rank**
TabDDPM	81.92$\pm$0.13	64.05$\pm$0.18	91.49$\pm$0.07	63.51$\pm$0.47	82.44$\pm$0.09	59.09$\pm$0.16	0.00
TabSYN	51.67$\pm$0.35	50.87$\pm$0.17	52.05$\pm$0.88	52.10$\pm$0.39	51.55$\pm$0.38	50.72$\pm$0.25	0.0
TabUnite-i2bFlow	53.87$\pm$0.27	52.96$\pm$0.44	59.66$\pm$0.54	83.71$\pm$0.28	54.33$\pm$0.65	55.81$\pm$0.11	0.00
TabUnite-dicFlow	65.35$\pm$0.04	57.79$\pm$0.26	72.16$\pm$0.65	82.90$\pm$0.46	60.97$\pm$0.25	55.76$\pm$0.51	0.00

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We elaborate the model architecture as well as encoding methods introduced in the abstract in depth in Section 3, with visual diagrams presented in Figure 1 and Figure 2. The claims on our model's performance are backed by Table 1, Table 7 where we highlighted the highest-performing models for each dataset, as well as various other results in the appendix.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: A limitation of our methodology is that we have not extensively explored a continuous embedding scheme where we perform the reverse and unify the generative space into a categorical one. Inspired by (Ansari et al., 2024), we conduct initial explorations of time series tokenization to embed continuous features yet, our results are still inconclusive and left to future work.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best

judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

   Justification: We do not include theoretical results in our paper.

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Detailed experimental setup, methodologies, and chosen parameters are shown in Appendix. We evaluate our models on a variety of metrics and tests. A, B, and C.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The code is anonymised and zipped along with our submission.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: Data splits can be found in Appendix C.1. We provide a detailed list of our hyperparameters in Table 3. We explicitly state in the section that we utilize the same parameters as two prior models to ensure that experimental results are commensurable.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We provide standard deviation error bars for our experimental results when permissible. Specifically, this is shown in our tables.

   Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We introduce in Appendix C the computer resources used in our experiments. The compute required for experimental runs are detailed in Table 9.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: With have adhered to the NeurIPS Code of Ethics when conducting our research on this paper.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

Justification: Our work on tabular data generation does not include potential malicious or unintended uses or impact specific groups. It does not violate privacy and security concerns either.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: One of our main contributions, a large mixed-type tabular dataset, is curated from augmenting a public-domain dataset based on anonymised US census data in 1990 (Dua & Graff, 2017), which poses a very low risk for potential misuse.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We properly acknowledge and cite all assets and resources used in the paper. The license of the datasets is also explicitly mentioned as the CC-BY 4.0 license in our Appendix.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, paperswithcode.com/datasets has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification: We introduce a new dataset, Census Synthetic, with proper documentation on how the dataset is curated in the Appendix. License is also based on the existing Census dataset where it is CC-BY 4.0.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: The paper does not involve crowdsourcing nor research with human subjects

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: The paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.