TabDiff: a Multi-Modal Diffusion Model for Tabular Data Generation

Juntong Shi[2*†], Minkai Xu[1*‡], Harper Hua[1*], Hengrui Zhang[3*], Stefano Ermon[1], Jure Leskovec[1]

[*]Equal contribution [†]Work done during a visit at Stanford [‡]Corresponding author

[1]Stanford University [2]University of Southern California [3]University of Illinois Chicago

{minkai,shuohua,ermon,jure}@stanford.edu

shisteve@usc.edu hzhan55@uic.edu

Abstract

Synthesizing high-quality tabular data is an important topic in many data science tasks, ranging from dataset augmentation to privacy protection. However, developing expressive generative models for tabular data is challenging due to its inherent heterogeneous data types, complex inter-correlations, and intricate column-wise distributions. In this paper, we introduce TabDiff, a joint diffusion framework that models all multi-modal distributions of tabular data in one model. Our key innovation is the development of a joint continuous-time diffusion process for numerical and categorical data, where we propose feature-wise learnable diffusion processes to counter the high disparity of different feature distributions. TabDiff is parameterized by a transformer handling different input types, and the entire framework can be efficiently optimized in an end-to-end fashion. We further introduce a multi-modal stochastic sampler to automatically correct the accumulated decoding error during sampling, and propose classifier-free guidance for conditional missing column value imputation. Comprehensive experiments on seven datasets demonstrate that TabDiff achieves superior average performance over existing competitive baselines across all eight metrics, with up to 22.5% improvement over the state-of-the-art model on pair-wise column correlation estimations. Code is available at https://github.com/MinkaiXu/TabDiff.

1 Introduction

Tabular data is ubiquitous in various databases, and developing effective generative models for it is a fundamental problem in many data processing and analysis tasks, ranging from training data augmentation (Fonseca & Bacao, 2023), data privacy protection (Assefa et al., 2021; Hernandez et al., 2022), to missing value imputation (You et al., 2020; Zheng & Charoenphakdee, 2022). With versatile synthetic tabular data that share the same format and statistical properties as the existing dataset, we are able to completely replace real data in a workflow or supplement the data to enhance its utility, which makes it easier to share and use. The capability of anonymizing data and enlarging sample size without compromising the overall data quality enables it to revolutionize the field of data science. Unlike image data, which comprises pure continuous pixel values with local spatial correlations, or text data, which comprises tokens that share the same dictionary space, tabular data features have much more complex and varied distributions (Xu et al., 2019; Borisov et al., 2023), making it challenging to learn joint probabilities across multiple columns. More specifically, such inherent heterogeneity leads to obstacles from two aspects: 1) typical tabular data often contains multi-modal data types, *i.e.*, continuous (*e.g.*, numerical features) and discrete (*e.g.*, categorical features) variables; 2) within the same feature type, features do not share the exact same data property because of the

Table Representation Learning Workshop at NeurIPS 2024

different meaning they represent, resulting in different column-wise marginal distributions (even after normalizing them into same value ranges).

In recent years, numerous deep generative models have been proposed for tabular data generation with autoregressive models (Borisov et al., 2023), VAEs (Liu et al., 2023), and GANs (Xu et al., 2019) in the past few years. Though they have notably improved the generation quality compared to traditional machine learning generation techniques such as resampling (Chawla et al., 2002), the generated data quality is still far from satisfactory due to limited model capacity. Recently, with the rapid progress in diffusion models (Song & Ermon, 2019; Ho et al., 2020; Rombach et al., 2022), researchers have been actively exploring extending this powerful framework to tabular data (Kim et al., 2022; Kotelnikov et al., 2023; Zhang et al., 2024). For example, Zheng & Charoenphakdee (2022); Zhang et al. (2024) transform all features into a latent continuous space via various encoding techniques and apply Gaussian diffusion there, while Kotelnikov et al. (2023); Lee et al. (2023) combine discrete-time continuous and discrete diffusion processes (Austin et al., 2021) to deal with numerical and categorical features separately. However, prior methods are trapped in sub-optimal performance due to additional encoding overhead or imperfect discrete-time diffusion modeling, and none of them consider the feature-wise distribution heterogeneity issue in a multi-modal framework.

In this paper, we present TabDiff, a novel and principled multi-modal diffusion framework for tabular data generation. TabDiff perturbs numerical and categorical features with a joint diffusion process, and learns a single model to simultaneously denoising all modalities. Our key innovation is the development of multi-modal feature-wise learnable diffusion processes to counteract the high heterogeneity across different feature distributions. Such feature-specific learnable noise schedules enable the model to optimally allocate the model capacity to different features in the training phase. Besides, it encourages the model to capture the inherent correlations during sampling since the model can denoise different features in a flexible order based on the learned schedule. We parameterize TabDiff with a transformer operating on different input types and optimize the entire framework efficiently in an end-to-end fashion. The framework is trained with a continuous-time limit of evidence lower bound. To reduce the decoding error during denoising sampling, we design a multi-modal stochastic sampler that automatically corrects the accumulated decoding error during sampling. In addition, we highlight that TabDiff can also be applied to conditional generation tasks such as missing column imputation, and we further introduce classifier-free guidance technique to improve the conditional generation quality.

TabDiff enjoys several notable advantages: 1) our model learns the joint distribution in the original data space with an expressive continuous-time diffusion framework; 2) the framework is sensitive to varying feature marginal distribution and can adaptively reason about feature-specific information and pair-wise correlations. We conduct comprehensive experiments to evaluate TabDiff against state-of-the-art methods across seven widely adopted tabular synthesis benchmarks. Results show that TabDiff consistently outperforms previous methods over eight distinct evaluation metrics, with up to 22.5% improvement over the state-of-the-art model on pair-wise column correlation estimations, suggesting our superior generative capacity on mixed-type tabular data.

2 Method

2.1 Overview

Notation. For a given mixed-type tabular dataset $\mathcal{T}$, we denote the number of numerical features and categorical features as M_{num} and M_{cat}, respectively. The dataset is represented as a collection of data entries $\mathcal{T} = \{\mathbf{x}\} = \{[\mathbf{x}^{\text{num}}, \mathbf{x}^{\text{cat}}]\}$, where each entry $\mathbf{x}$ is a concatenated vector consisting of its numerical features $\mathbf{x}^{\text{num}}$ and categorical features $\mathbf{x}^{\text{cat}}$. We represent the numerical features as a M_{num} dimensional vector $\mathbf{x}^{\text{num}} \in \mathbb{R}^{M_{\text{num}}}$ and denote the i-th feature as $(\mathbf{x}^{\text{num}})_i \in \mathbb{R}$. We represent each categorical column with C_j finite categories as a one-hot column vector $(\mathbf{x}^{\text{cat}})_j \in \{0,1\}^{(C_j+1)}$, with an extra dimension dedicated to the [MASK] state. The (C_j+1)-th category corresponds to the special [MASK] state and we use $\mathbf{m} \in \{0,1\}^K$ as the one-hot vector for it. In addition, we define $\text{Cat}(\cdot;\boldsymbol{\pi})$ as the categorical distribution over K classes with probabilities given by $\boldsymbol{\pi} \in \Delta^K$, where Δ^K is the K-simplex.

Different from common data types such as images and text, developing generative models for tabular data is challenging as the distribution is determined by multi-modal data. We therefore propose

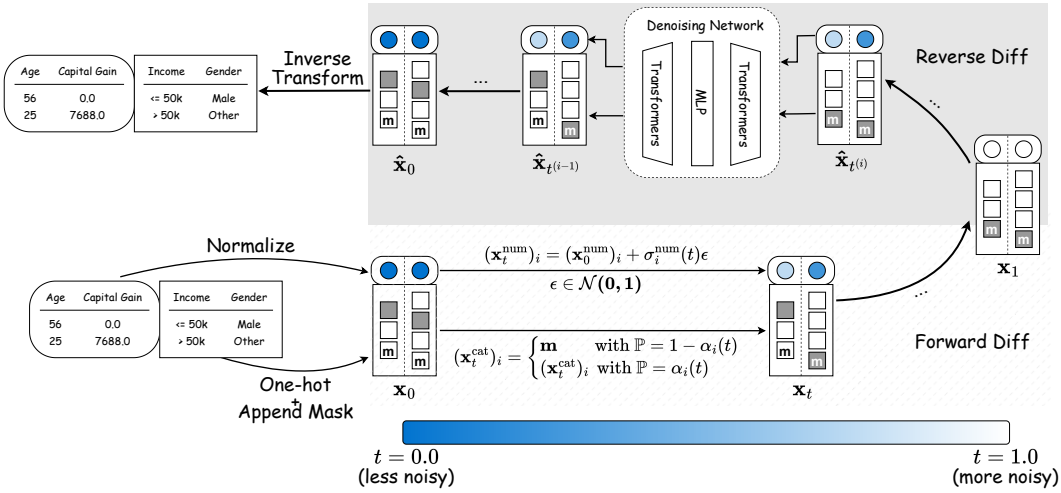

Figure 1: A high-level overview of TabDiff. TabDiff operates by normalizing numerical columns and converting categorical columns into one-hot vectors with an extra [MASK] class. Joint forward diffusion processes are applied to all modalities with each column's noise rate controlled by learnable schedules. New samples are generated via reverse process, with the denoising network gradually denoising $\mathbf{x}_1$ into $\mathbf{x}_0$ and then applying the inverse transform to recover the original format.

TabDiff, a unified generative model for modeling the joint distribution $p(\mathbf{x})$ using a continuous-time diffusion framework. TabDiff can learn the distribution from finite samples and generate faithful, diverse, and novel samples unconditionally. We provide a high-level overview in Figure 1, which includes a forward *diffusion* process and a reverse *generative* process, both defined in continuous time. The diffusion process gradually adds noise to data, and the generative process learns to recover the data from prior noise distribution with neural networks parameterized by θ. In the following sections, we elaborate on how we develop the unified diffusion framework with learnable noise schedules and perform training and sampling in practice.

2.2 Multi-Modal Diffusion Framework

Diffusion models (Sohl-Dickstein et al., 2015; Song & Ermon, 2019; Ho et al., 2020) are likelihood-based generative models that learn the data distribution via forward and reverse Markov processes. Our goal is to develop a principled diffusion model for generating mixed-type tabular data that faithfully mimics the statistical distribution of the real dataset. Our framework TabDiff is designed to directly operate on the data space and naturally handle each tabular column in its built-in datatype. TabDiff is built on a hybrid forward process that gradually injects noise to numerical and categorical data types separately with different diffusion schedules $\boldsymbol{\sigma}^{\text{num}}$ and $\boldsymbol{\sigma}^{\text{cat}}$. Let $\{\mathbf{x}_t : t \sim [0, 1]\}$ denote a sequence of data in the diffusion process indexed by a continuous time variable $t \in [0, 1]$, where $\mathbf{x}_0 \sim p_0$ are *i.i.d.* samples from real data distribution and $\mathbf{x}_1 \sim p_1$ are pure noise from prior distribution. The hybrid forward diffusion process can be then represented as:

$$q(\mathbf{x}_t \mid \mathbf{x}_0) = q\left(\mathbf{x}_t^{\text{num}} \mid \mathbf{x}_0^{\text{num}}, \boldsymbol{\sigma}^{\text{num}}(t)\right) \cdot q\left(\mathbf{x}_t^{\text{cat}} \mid \mathbf{x}_0^{\text{cat}}, \boldsymbol{\sigma}^{\text{cat}}(t)\right). \tag{1}$$

Then the true reverse process can be represented as the joint posterior:

$$q(\mathbf{x}_s|\mathbf{x}_t, \mathbf{x}_0) = q(\mathbf{x}_s^{\text{num}}|\mathbf{x}_t, \mathbf{x}_0) \cdot q(\mathbf{x}_s^{\text{cat}}|\mathbf{x}_t, \mathbf{x}_0), \tag{2}$$

where s and t are two arbitrary timesteps that $0 < s < t < 1$. We aim to learn a denoising model $p_\theta(\mathbf{x}_s|\mathbf{x}_t)$ to match the true posterior. In the following, we discuss the detailed formulations of diffusion processes for continuous and categorical features in separate manners. To enhance clarity, we omit the superscripts on $\mathbf{x}^{\text{num}}$ and $\mathbf{x}^{\text{cat}}$ when the inclusion is unnecessary for understanding.

Gaussian Diffusion for Numerical Features. In this paper, we model the forward diffusion for continuous features $\mathbf{x}^{\text{num}}$ as a stochastic differential equation (SDE) $\mathrm{d}\mathbf{x} = \mathbf{f}(\mathbf{x}, \mathrm{t})\mathrm{dt} + \mathrm{g}(\mathrm{t})\mathrm{d}\mathbf{w}$, with $\mathbf{f}(\cdot, t) : \mathbb{R}^{M_{\text{num}}} \to \mathbb{R}^{M_{\text{num}}}$ being the drift coefficient, $g(\cdot) : \mathbb{R} \to \mathbb{R}$ being the diffusion coefficient, and $\boldsymbol{w}$ being the standard Wiener process (Song et al., 2021; Karras et al., 2022). The revere generation process solves the probability flow ordinary differential equation (ODE) $\mathrm{d}\mathbf{x} = \left[\mathbf{f}(\mathbf{x}, \mathrm{t}) - \frac{1}{2}\mathrm{g}(\mathrm{t})^2 \nabla_{\mathbf{x}} \log \mathrm{p}_\mathrm{t}(\mathbf{x})\right]\mathrm{dt}$, where $\nabla_{\mathbf{x}} \log p_t(\mathbf{x})$ is the score function of $p_t(\mathbf{x})$. In this paper,

we use the Variance Exploding formulation with $\mathbf{f}(\cdot, t) = \mathbf{0}$ and $g(t) = \sqrt{2[\frac{d}{dt}\boldsymbol{\sigma}^{\text{num}}(t)]\boldsymbol{\sigma}^{\text{num}}(t)}$, which yields the forward process :

$$\mathbf{x}_t^{\text{num}} = \mathbf{x}_0^{\text{num}} + \boldsymbol{\sigma}^{\text{num}}(t)\boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}_{M_{\text{num}}}). \tag{3}$$

And the reversal can then be formulated accordingly as:

$$\mathrm{d}\mathbf{x}^{\text{num}} = -[\frac{\mathrm{d}}{\mathrm{dt}}\boldsymbol{\sigma}^{\text{num}}(\mathrm{t})]\boldsymbol{\sigma}^{\text{num}}(\mathrm{t})\nabla_{\mathbf{x}} \log \mathrm{p_t}(\mathbf{x}^{\text{num}})\mathrm{dt}. \tag{4}$$

In TabDiff, we train the diffusion model $\boldsymbol{\mu}_\theta$ to jointly denoise the numerical and categorical features. We use $\boldsymbol{\mu}_\theta^{\text{num}}$ to denote numerical part of the denoising model output, and train the model via optimizing the denoising loss:

$$\mathcal{L}_{\text{num}}(\theta, \rho) = \mathbb{E}_{\mathbf{x}_0 \sim p(\mathbf{x}_0)} \mathbb{E}_{t \sim U[0,1]} \mathbb{E}_{\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})} \left\| \boldsymbol{\mu}_\theta^{\text{num}}(\mathbf{x}_t, t) - \boldsymbol{\epsilon} \right\|_2^2. \tag{5}$$

Masked Diffusion for Categorical Features, For categorical features, we take inspiration from the recent progress on discrete state-space diffusion for language modeling (Austin et al., 2021; Shi et al., 2024; Sahoo et al., 2024). The forward diffusion process is defined as a masking (absorbing) process that smoothly interpolates between the data distribution $\text{Cat}(\cdot; \mathbf{x})$ and the target distribution $\text{Cat}(\cdot; \mathbf{m})$, where all probability mass are assigned on the [MASK] state:

$$q(\mathbf{x}_t | \mathbf{x}_0) = \text{Cat}(\mathbf{x}_t; \alpha_t \mathbf{x}_0 + (1 - \alpha_t)\mathbf{m}). \tag{6}$$

$\alpha_t \in [0, 1]$ is a strictly decreasing function of t, with $\alpha_0 \approx 1$ and $\alpha_1 \approx 0$. It represents the probability for the real data $\mathbf{x}_0$ to be masked at time step t. By the time $t = 1$, all inputs are masked with probability 1. In practice this schedule is parameterized by $\alpha_t = \exp(-\boldsymbol{\sigma}^{\text{cat}}(t))$, where $\boldsymbol{\sigma}^{\text{cat}}(t) : [0, 1] \to \mathbb{R}^+$ is a strictly increasing function. Such forward process entails the step transition probabilities $q(\mathbf{x}_t | \mathbf{x}_s) = \text{cat}(\mathbf{x}_t; \alpha_{t|s}\mathbf{x}_s + (1 - \alpha_{t|s})\mathbf{m})$, where $\alpha_{t|s} = \alpha_t / \alpha_s$. Under the hood, this transition means that at each diffusion step, the data will be perturbed to the [MASK] state with a probability of $(1 - \alpha_{t|s})$, and remains there until $t = 1$ if perturbed.

Similar to numerical features, in the reverse denoising process for categorical ones, the diffusion model $\boldsymbol{\mu}_\theta$ aims to progressively unmask each column from the 'masked' state. The true posterior distribution conditioned on $\mathbf{x}_0$ has the close form of:

$$q(\mathbf{x}_s | \mathbf{x}_t, \mathbf{x}_0) = \begin{cases} \text{Cat}(\mathbf{x}_s; \mathbf{x}_t) & \mathbf{x}_t \neq \mathbf{m}, \\ \text{Cat}\left(\mathbf{x}_s; \frac{(1-\alpha_s)\mathbf{m} + (\alpha_s - \alpha_t)\mathbf{x}_0}{1-\alpha_t}\right) & \mathbf{x}_t = \mathbf{m}. \end{cases} \tag{7}$$

We introduce the denoising network $\mu_\theta^{\text{cat}}(\mathbf{x}_t, t) : C \times [0, 1] \to \Delta^C$ to estimate $\mathbf{x}_0$, through which we can approximate the unknown true posterior as:

$$p_\theta(\mathbf{x}_s^{\text{cat}} | \mathbf{x}_t^{\text{cat}}) = \begin{cases} \text{Cat}(\mathbf{x}_s^{\text{cat}}; \mathbf{x}_t^{\text{cat}}) & \mathbf{x}_t^{\text{cat}} \neq \mathbf{m}, \\ \text{Cat}\left(\mathbf{x}_s^{\text{cat}}; \frac{(1-\alpha_s)\mathbf{m} + (\alpha_s - \alpha_t)\boldsymbol{\mu}_\theta^{\text{cat}}(\mathbf{x}_t, t)}{1-\alpha_t}\right) & \mathbf{x}_t = \mathbf{m}, \end{cases} \tag{8}$$

which implied that at each reverse step, we have a probability of $(\alpha_s - \alpha_t)/(1 - \alpha_t)$ to recover x_0, and once being recovered, x_t stays fixed for the remainder of the process. Extensive works (Kingma et al., 2021; Shi et al., 2024) have shown that increasing discretization resolution can help approximate tighter evidence lower bound (ELBO). Therefore, we resort to optimizing the likelihood bound $\mathcal{L}_{\text{cat}}$ under continuous time limit:

$$\mathcal{L}_{\text{cat}}(\theta, k) = \mathbb{E}_q \int_{t=0}^{t=1} \frac{\alpha_t'}{1 - \alpha_t} \log \langle \boldsymbol{\mu}_\theta^{\text{cat}}(\mathbf{x}_t, t), \mathbf{x}_0^{\text{cat}} \rangle dt, \tag{9}$$

where α_t' is the first order derivative of α_t.

2.3 Training with Adaptively Learnable Noise Schedules

Tabular data is inherently highly heterogeneous of mixed numerical and categorical data types, and mixed feature distributions within each data type. Therefore, unlike pixels that share a similar distribution across three RGB channels and word tokens that share the exact same vocabulary space, each column (feature) of the table has its own specific marginal distributions, which requires the model to amortize its capacity adaptively across different features. We propose to adaptively learn a more fine-grained noise schedule for each feature respectively. To balance the trade-off between the learnable noise schedule's flexibility and robustness, we design two function families: the power mean numerical schedule and the log-linear categorical schedule.

Power-mean schedule for numerical features. For the numerical noise schedule $\boldsymbol{\sigma}^{\text{num}}(t)$ in Eq. (3), we define $\boldsymbol{\sigma}^{\text{num}}(t) = [\sigma^{\text{num}}_{\rho_i}(t)]$, with ρ_i being a learnable parameter for individual numerical features. For $\forall i \in \{1, \cdots, M_{\text{num}}\}$, we have $\sigma^{\text{num}}_{\rho_i}(t)$ as:

$$\sigma^{\text{num}}_{\rho_i}(t) = \left(\sigma_{\min}^{\frac{1}{\rho_i}} + t(\sigma_{\max}^{\frac{1}{\rho_i}} - \sigma_{\min}^{\frac{1}{\rho_i}})\right)^{\rho_i}. \tag{10}$$

Log-linear schedule for categorical features. Similarly, for the categorical noise schedule $\boldsymbol{\sigma}^{\text{cat}}(t)$ in Section 2.2, we define $\boldsymbol{\sigma}^{\text{cat}}(t) = [\sigma^{\text{cat}}_{k_j}(t)]$, with k_i being a learnable parameter for individual categorical features. For $\forall j \in \{1, \cdots, M_{\text{cat}}\}$, we have $\sigma^{\text{cat}}_{k_j}(t)$ as:

$$\sigma^{\text{cat}}_{k_j}(t) = -\log(1 - t^{k_j}). \tag{11}$$

Algorithm 1 Training

1: **repeat**
2: Sample $\mathbf{x}_0 \sim p_0(\mathbf{x})$
3: Sample $t \sim U(0, 1)$
4: Sample $\boldsymbol{\epsilon}_{\text{num}} \sim \mathcal{N}(0, \mathbf{I}_{M\,\text{num}})$
5: $\mathbf{x}^{\text{num}}_t = \mathbf{x}^{\text{num}}_0 + \boldsymbol{\sigma}^{\text{num}}(t)\boldsymbol{\epsilon}_{\text{num}}$
6: $\boldsymbol{\alpha}_t = \exp(-\boldsymbol{\sigma}^{\text{cat}}(t))$
7: Sample $\mathbf{x}^{\text{cat}}_t \sim \text{q}(\mathbf{x}_\text{t}|\mathbf{x}_0, \boldsymbol{\alpha}_\text{t})$ from Eq. (6)
8: $\mathbf{x}_t \leftarrow \text{CONCAT}(\mathbf{x}^{\text{num}}_t, \mathbf{x}^{\text{cat}}_t)$
9: Take gradient descent step on $\nabla_{\theta,\rho,k}\mathcal{L}_{\text{TabDiff}}$
10: **until** converged

In practice, we fix the same initial and final noise levels across all numerical features so that $\sigma^{\text{num}}_i(0) = \sigma_{\min}$ and $\sigma^{\text{num}}_i(1) = \sigma_{\max}$ for $\forall i \in \{1, \cdots, M_{\text{num}}\}$. We similarly bound the initial and final noise levels for the categorical features, as detailed in Appendix A.3.1. This enables us to constrain the freedom of schedules and thus stabilize the training.

Joint objective function. We update $M_{\text{num}} + M_{\text{cat}}$ parameters $\rho_1, \cdots, \rho_{M_{\text{num}}}$ and $k_1, \cdots, k_{M_{\text{cat}}}$ via backpropagation without the need of modifying the loss function. Consolidating $\mathcal{L}_{\text{num}}$ and $\mathcal{L}_{\text{cat}}$, we have the total loss $\mathcal{L}$ with two weight terms λ_{num} and λ_{cat} as:

$$\begin{aligned}
&\mathcal{L}_{\text{TabDiff}}(\theta, \rho, k) = \lambda_{\text{num}}\mathcal{L}_{\text{num}}(\theta, \rho) + \lambda_{\text{cat}}\mathcal{L}_{\text{cat}}(\theta, k) \\
&= \mathbb{E}_{t\sim U(0,1)}\mathbb{E}_{(\mathbf{x}_t,\mathbf{x}_0)\sim q(\mathbf{x}_t,\mathbf{x}_0)}\left(\lambda_{\text{num}}\left\|\boldsymbol{\mu}^{\text{num}}_\theta(\mathbf{x}_t, t) - \boldsymbol{\epsilon}\right\|_2^2 + \frac{\lambda_{\text{cat}}\,\alpha'_t}{1-\alpha_t}\log\langle\boldsymbol{\mu}^{\text{cat}}_\theta(\mathbf{x}_t, t), \mathbf{x}^{\text{cat}}_0\rangle\right).
\end{aligned} \tag{12}$$

With the forward process defined in Eq. (3) and Eq. (6), we present the detailed training procedure in Algorithm 1. Here, we sample a continuous time step t from a uniform distribution $U(0, 1)$ and then perturb numerical and categorical features with their respective noise schedules based on this same time index. Then, we input the concatenated $\mathbf{x}^{\text{num}}_t$ and $\mathbf{x}^{\text{cat}}_t$ into the model and take gradient on the joint loss function defined in Eq. (12).

2.4 Sampling with Backward Stochastic Sampler

One notable property of the joint sampling process is that the intermediate decoded categorical feature will not be updated anymore during sampling (see Eq. (8)). However, as tabular data are highly structured with complicated inter-column correlations, we expect the model to correct the error during sampling. To this end, we introduce a novel stochastic sampler by restarting the backward process with an additional forward process at each denoising step. Related work on continuous diffusions Karras et al. (2022); Xu et al. (2023) has shown that incorporating such stochasticity can yield better generation quality. We extend such intuition to both numerical and categorical features in tabular generation. At each sampling step t, we first add a small time increment to the current time step t to $t^+ = t + \gamma_t t$ according to a factor γ_t, and then perform the intermediate forward sampling between t^+ and t by joint diffusion process Equations (3) and (6). From the increased-noise sample $\mathbf{x}_{t^+}$, we solve the ODE backward for $\mathbf{x}^{\text{num}}$ and $\mathbf{x}^{\text{cat}}$ from t^+ to $t - 1$, respectively, with a single update. This framework enables self-correction by randomly perturbing decoded features in the forward step. We summarize the sampling framework in Algorithm 2, and provide the ablation study for the stochastic sampler in Section 4.3.

Algorithm 2 Sampling

1: Sample $\mathbf{x}^{\text{num}}_T \sim \mathcal{N}(0, \mathbf{I}_{M\,\text{num}})$, $\mathbf{x}^{\text{cat}}_T = \boldsymbol{m}$
2: **for** $t = T$ to 1 **do**
3: Sample $\boldsymbol{\epsilon}^{\text{num}} \sim \mathcal{N}(0, \mathbf{I}_{M\,\text{num}})$
4: $\gamma_t = 1/T$, $t^+ \leftarrow t + \gamma_t t$
5: $\mathbf{x}^{\text{num}}_{t^+} \leftarrow \mathbf{x}^{\text{num}}_t + \sqrt{\boldsymbol{\sigma}^{\text{num}}(t^+)^2 - \boldsymbol{\sigma}^{\text{num}}(t)^2}\boldsymbol{\epsilon}$
6: Sample $\mathbf{x}^{\text{cat}}_{t^+} \sim q\left(\mathbf{x}^{\text{cat}}_{t^+}|\mathbf{x}^{\text{cat}}_t,\right)$ from Eq. (6)
7: $\mathbf{x}_{t^+} = [\mathbf{x}^{\text{num}}_{t^+}, \mathbf{x}^{\text{cat}}_{t^+}]$
8: $d\mathbf{x}^{\text{num}} = (\mathbf{x}^{\text{num}}_{t^+} - \mu_\theta(\mathbf{x}_{t^+}, t^+))/\boldsymbol{\sigma}^{\text{num}}(t^+)$
9: $\mathbf{x}^{\text{num}}_{t-1} \leftarrow \mathbf{x}^{\text{num}}_{t^+} + (\boldsymbol{\sigma}^{\text{num}}(t-1) - \boldsymbol{\sigma}^{\text{num}}(t^+))d\mathbf{x}^{\text{num}}$
10: Sample $\mathbf{x}^{\text{cat}}_{t-1} \sim p_\theta(\mathbf{x}^{\text{cat}}_{t-1}|\mathbf{x}^{\text{cat}}_{t^+})$ from Eq. (8)
11: **end for**
12: **return** $\mathbf{x}^{\text{num}}_0, \mathbf{x}^{\text{cat}}_0$

2.5 Classifier-free Guidance Conditional Generation

TabDiff can also be extended as a conditional generative model, which is important in many tasks such as missing value imputation. Let $\mathbf{y} = \{[\mathbf{y}^{\text{num}}, \mathbf{y}^{\text{cat}}]\}$ be the collection of provided properties in tabular data, containing both categorical and numerical features, and let $\mathbf{x}$ denote the missing interest features in this section. Imputation means we want to predict $\mathbf{x} = \{[\mathbf{x}^{\text{num}}, \mathbf{x}^{\text{cat}}]\}$ conditioned on $\mathbf{y}$. TabDiff can be freely extended to conditional generation by only conducting denoising sampling for $\mathbf{x}_t$, while keeping other given features $\mathbf{y}_t$ fixed as $\mathbf{y}$.

Previous works on diffusion models (Dhariwal & Nichol, 2021) show that conditional generation quality can be further improved with a guidance classifier/regressor $p(\mathbf{y} \mid \mathbf{x})$. However, training the guidance classifier becomes challenging when $\mathbf{x}$ is a high-dimensional discrete object, and existing methods typically handle this by relaxing $\mathbf{x}$ as continuous (Vignac et al., 2023). Inspired by the classifier-free guidance (CFG) framework (Ho & Salimans, 2022) developed for continuous diffusion, we propose a unified CFG framework that eliminates the need for a classifier and handles multi-modal $\mathbf{x}$ and $\mathbf{y}$ effectively. The guided conditional sample distribution is given by $\tilde{p}_\theta(\mathbf{x}_t|\mathbf{y}) \propto p_\theta(\mathbf{x}_t|\mathbf{y})p_\theta(\mathbf{y}|\mathbf{x}_t)^\omega$, where $\omega > 0$ controls strength of the guidance. Applying Bayes' Rule, we get

$$\tilde{p}_\theta(\mathbf{x}_t|\mathbf{y}) \propto p_\theta(\mathbf{x}_t|\mathbf{y})p_\theta(\mathbf{y}|\mathbf{x}_t)^\omega = p_\theta(\mathbf{x}_t|\mathbf{y}) \left(\frac{p_\theta(\mathbf{x}_t|\mathbf{y})p(\mathbf{y})}{p_\theta(\mathbf{x}_t)} \right)^\omega = \frac{p_\theta(\mathbf{x}_t|\mathbf{y})^{\omega+1}}{p_\theta(\mathbf{x}_t)^\omega} p(\mathbf{y})^\omega. \tag{13}$$

We drop $p(\mathbf{y})$ for it does no depend on θ. Taking the logarithm of the probabilities, we obtain,

$$\log \tilde{p}_\theta(\mathbf{x}_t|\mathbf{y}) = (1+\omega) \log p_\theta(\mathbf{x}_t|\mathbf{y}) - \omega \log p_\theta(\mathbf{x}_t), \tag{14}$$

which implies the following changes in the sampling steps. For the numerical features, $\boldsymbol{\mu}_\theta^{\text{num}}(\mathbf{x}_t, t)$ is replaced by the interpolation of the conditional and unconditional estimates (Ho & Salimans, 2022):

$$\tilde{\boldsymbol{\mu}}_\theta^{\text{num}}(\mathbf{x}_t, y, t) = (1+\omega)\boldsymbol{\mu}_\theta^{\text{num}}(\mathbf{x}_t, y, t) - \omega \boldsymbol{\mu}_\theta^{\text{num}}(\mathbf{x}_t, t). \tag{15}$$

And for the categorical features, we instead predict x_0 with $\tilde{p}_\theta(\mathbf{x}_s^{\text{cat}}|\mathbf{x}_t^{\text{cat}}, \mathbf{y})$, satisfying

$$\log \tilde{p}_\theta(\mathbf{x}_s^{\text{cat}}|\mathbf{x}_t^{\text{cat}}, \mathbf{y}) = (1+\omega) \log p_\theta(\mathbf{x}_s^{\text{cat}}|\mathbf{x}_t^{\text{cat}}, \mathbf{y}) - \omega \log p_\theta(\mathbf{x}_s^{\text{cat}}|\mathbf{x}_t^{\text{cat}}). \tag{16}$$

The guided probability utilizes the original unconditional model and requires only an additional small model for modeling the missing value distribution, *i.e.*, $\boldsymbol{\mu}_\theta^{\text{num}}(\mathbf{x}_t, t)$ and $p_\theta(\mathbf{x}_s^{\text{cat}}|\mathbf{x}_t^{\text{cat}})$. We provide empirical results for CFG sampling in Appendix B.2.

3 Related Work

Recent studies have developed different generative models for tabular data, including VAE-based methods, TVAE (Xu et al., 2019), and GAN (Generative Adversarial Networks)-based methods, CTGAN (Xu et al., 2019). These methods usually lack sufficient model expressivity for complicated data distribution. Recently, diffusion models have shown powerful generative ability for diverse data types and thus have been adopted by many tabular generation methods. Kotelnikov et al. (2023); Lee et al. (2023) designed separate discrete-time diffusion processes (Austin et al., 2021) for numerical and categorical features separately. However, they built their diffusion processes on discrete time steps, which have been proven to yield a looser ELBO estimation and thus lead to sub-optimal generation quality (Song et al., 2021; Kingma et al., 2021). To tackle such a problem caused by limited discretization of diffusion processes and push it to a continuous time framework, Zheng & Charoenphakdee (2022); Zhang et al. (2024) transform features into a latent continuous space via various encoding techniques, since advanced diffusion models are mainly designed for continuous random variables with Gaussian perturbation and thus cannot directly handle tabular data. However, it has shown that these solutions either are trapped with sub-optimal performance due to encoding overhead or cannot capture complex co-occurrence patterns of different modalities because of the indirect modeling and low model capacity. In summary, none of existing methods have explored the powerful multi-modal diffusion framework in the continuous-time limit and explicitly tackle the feature-wise heterogeneity issue in the multi-modal diffusion process.

4 Experiments

4.1 Experimental Setups

Datasets. We conduct experiments on seven real-world tabular datasets – Adult, Default, Shoppers, Magic, Faults, Beijing, News, and Diabetes – each containing both numerical and categorical

Table 1: Performance comparison of TabDiff's against baselines across key metrics. Each column represents the mean performance with std. on each metric across seven datasets. **Bold Face** highlights the best score for each metric.

Methods	**Shape**↓	**Trend**↓	α-**Precision**↑	β-**Recall**↑	**Detection**↑	**MLE div**↓
CTGAN	15.99±4.72	16.36±15.72	82.40±13.19	23.11±10.45	64.44±10.72	23.73±39.80
TVAE	15.97±16.26	16.43±16.82	75.85±28.99	25.32±10.00	52.50±31.13	20.15±27.89
GOGGLE	17.91±18.07	28.18±25.33	70.82±26.24	9.78±6.62	33.79±34.33	42.06±51.94
GReaT	14.20±14.71	40.52±46.25	80.87±8.12	42.86±4.42	51.18±12.41	13.31±23.03
STaSy	7.72±7.01	7.77±6.43	88.91±2.98	42.32±8.66	60.83±10.98	10.95±21.64
CoDi	21.56±21.59	23.23±23.35	84.29±11.75	27.12±	34.35±32.21	30.18±32.01
TabDDPM	16.93±19.47	11.95±13.44	72.48±43.18	35.44±26.17	70.44±44.19	11.95±16.88
TabSyn	1.35±1.44	2.33±2.39	97.86±1.58	46.77±8.30	91.56±15.27	6.78±10.54
TabDiff	**1.17±1.26**	**1.80±1.85**	**98.22±1.35**	**49.40±6.62**	**97.87±2.34**	**5.71±12.27**
Improv.	13.32%	22.64%	0.37%	5.62%	6.89%	15.8%

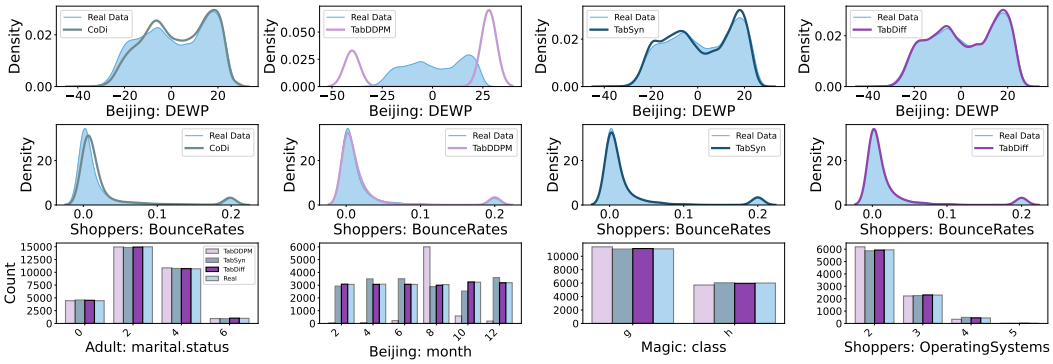

Figure 2: Visualization of the marginal densities of the generated data in comparison to the real data. Top and Middle: individual numerical column; Bottom: individual categorical column.

attributes. In addition, each dataset has an inherent machine-learning task, either classification or regression. Detailed profiles of the datasets are presented in Appendix A.1.

Baselines. We compare the proposed TabDiff with eight popular synthetic tabular data generation methods that are categorized into four groups: 1) GAN-based method: CTGAN (Xu et al., 2019); 2) VAE-based methods: TVAE (Xu et al., 2019) and GOGGLE (Liu et al., 2023); 3) Autoregressive Language Model: GReaT (Borisov et al., 2023); 4) Diffusion-based methods: STaSy (Kim et al., 2023), CoDi (Lee et al., 2023), TabDDPM (Kotelnikov et al., 2023) and TabSyn (Zhang et al., 2024).

Evaluation Methods. Following the previous methods (Zhang et al., 2024), we evaluate the quality of the synthetic data using eight distinct metrics categorized into three groups – 1) **Fidelity**: Shape, Trend, α-Precision, β-Recall, and Detection assess how well the synthetic data can faithfully recover the ground-truth data distribution; 2) **Downstream tasks**: Machine learning efficiency (MLE) and missing value imputation reveal the models' potential to power downstream tasks; 3) **Privacy**: The Distance to Closest Records (DCR) score evaluates the level of privacy protection by measuring how closely the synthetic data resembles the training data.

Implementation Details. All reported experiment results are the average of 20 random sampled synthetic data generated by the best-validated models. Additional implementation details, such as the hardware/software information as well as hyperparameter settings, are in Appendix C.

4.2 Results

In Table 1, we compare TabDiff to the baselines across the six chosen metrics. For each metric, we report the average score with standard deviation across the seven datasets. As shown in the table, TabDiff yields significant improvement over the competitive baselines on all the metrics. Notably, even on Shape and Trend, where the state-of-the-art (SOTA) performance is already extremely high with little room for improvement, TabDiff still achieved over 10% performance boost. Such superior

Method	Shape	Trend
TabSyn	1.35	2.33
TabDiff-Fix.+Det.	1.39	2.29
TabDiff-Fix.+Sto.	1.20	1.93
TabDiff-Learn.+Det.	1.24	1.92
TabDiff-Learn.+Sto.	**1.17**	**1.80**

Table 2: Ablation Studies on the stochastic sampler and learnable noise schedules.

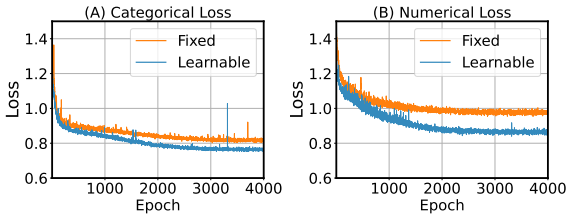

Figure 4: The adaptively learnable noise schedules reduce training loss.

ability to capture the shapes and trends of the original data distribution can also be visualized in Figures 2 and 3, where we see TabDiff's samples match most closely with the real data's density curves and correlation matrices. In addition, TabDiff improves upon the SOTA on MLE, demonstrating its ability to generate synthetic data that can power learning of downstream tasks. These results thoroughly demonstrate TabDiff's strong capacity in modeling multi-modal joint distributions. Due to space limits, we defer the discussion of the privacy metric (DCR score), missing value imputation, and detailed per-dataset results to Appendix B.

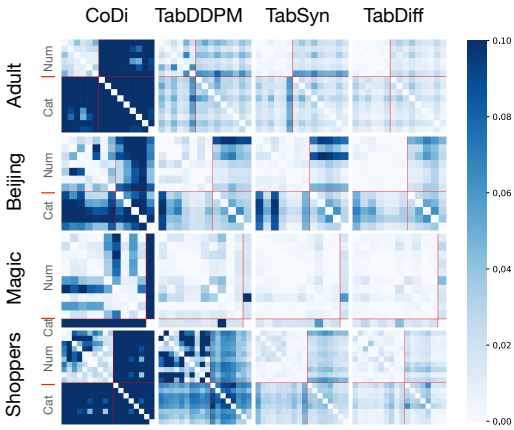

Figure 3: Pair-wise correlation heatmaps. Values represent the error rate (the lighter, the better).

4.3 Ablation Studies

Stochastic Sampler. We conduct ablation studies to assess the effectiveness of the stochastic sampler, discussed in Section 2.4. The results are presented in Table 2. We use 'Det.' and 'Sto.' as abbreviations for deterministic and stochastic samplers. The deterministic sampler refers to the conventional diffusion backward process described in Song et al. (2021); Karras et al. (2022), consisting of a series of deterministic ODE steps. According to Table 2, under both fixed and learnable noise schedules, TabDiff with the stochastic sampler consistently outperforms the deterministic sampler on the fidelity metrics Shape and Trend, regardless of whether learnable noise schedules are enabled. These confirm the efficacy of additional stochasticity in reducing decoding errors during backward diffusion sampling.

Adaptively Learnable Noise Schedule. Next, we perform ablation studies to evaluate the effectiveness of our adaptively learnable noise schedules, discussed in Section 2.3. We compare the model with learnable schedules against the model with non-learnable noise schedules, where the noise parameters for numerical features are fixed to $\rho_i \equiv 7, \forall i$ in Eq. (10) and, for numerical features, fixed to $k_j \equiv 1, \forall j$ in Eq. (11). We refer to these models as 'Learn.' and 'Fix.', respectively. According to the results in Table 2, the learnable noise schedules substantially improve performance, particularly in Trend and regardless of whether the stochastic sampler is enabled. Furthermore, we closely examine the training process of both models on the Adult dataset by plotting their changes of training loss in Figure 4. According to the figure, the learnable schedules (orange curves) significantly reduce both numerical and categorical losses in Eq. (12).

5 Conclusion

In this paper, we have introduced TabDiff, a multi-modal diffusion framework for generating high-quality synthetic data. TabDiff combines a hybrid diffusion process to handle numerical and categorical features in their native formats. To address the disparate distributions of features and their interrelationships, we further introduced several key innovations, including learnable column-wise noise schedules and the stochastic sampler. We conducted extensive experiments using a diverse set of datasets and metrics, comprehensively comparing TabDiff with existing approaches. The results demonstrate TabDiff's superior capacity in learning the original data distribution and generating faithful and diverse synthetic data to power downstream tasks.

References

Ahmed Alaa, Boris Van Breugel, Evgeny S Saveliev, and Mihaela van der Schaar. How faithful is your synthetic data? sample-level metrics for evaluating and auditing generative models. In *International Conference on Machine Learning*, pp. 290–306. PMLR, 2022.

Samuel A. Assefa, Danial Dervovic, Mahmoud Mahfouz, Robert E. Tillman, Prashant Reddy, and Manuela Veloso. Generating synthetic data in finance: Opportunities, challenges and pitfalls. In *Proceedings of the First ACM International Conference on AI in Finance*, ICAIF '20. Association for Computing Machinery, 2021. ISBN 9781450375849.

Jacob Austin, Daniel D Johnson, Jonathan Ho, Daniel Tarlow, and Rianne Van Den Berg. Structured denoising diffusion models in discrete state-spaces. *Advances in Neural Information Processing Systems*, 34:17981–17993, 2021.

Vadim Borisov, Kathrin Sessler, Tobias Leemann, Martin Pawelczyk, and Gjergji Kasneci. Language models are realistic tabular data generators. In *The Eleventh International Conference on Learning Representations*, 2023.

Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357, 2002.

Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 785–794, 2016.

Prafulla Dhariwal and Alexander Quinn Nichol. Diffusion models beat GANs on image synthesis. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, 2021.

Joao Fonseca and Fernando Bacao. Tabular and latent space synthetic data generation: a literature review. *Journal of Big Data*, 10(1):115, 2023.

Mikel Hernandez, Gorka Epelde, Ane Alberdi, Rodrigo Cilla, and Debbie Rankin. Synthetic data generation for tabular health records: A systematic review. *Neurocomputing*, 493:28–45, 2022.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, pp. 6840–6851, 2020.

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, pp. 26565–26577, 2022.

Jayoung Kim, Chaejeong Lee, Yehjin Shin, Sewon Park, Minjung Kim, Noseong Park, and Jihoon Cho. Sos: Score-based oversampling for tabular data. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 762–772, 2022.

Jayoung Kim, Chaejeong Lee, and Noseong Park. Stasy: Score-based tabular data synthesis. In *The Eleventh International Conference on Learning Representations*, 2023.

Diederik Kingma, Tim Salimans, Ben Poole, and Jonathan Ho. Variational diffusion models. *Advances in neural information processing systems*, 34:21696–21707, 2021.

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

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

Tennison Liu, Zhaozhi Qian, Jeroen Berrevoets, and Mihaela van der Schaar. Goggle: Generative modelling for tabular data by learning relational structure. In *The Eleventh International Conference on Learning Representations*, 2023.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10684–10695, 2022.

Subham Sekhar Sahoo, Marianne Arriola, Yair Schiff, Aaron Gokaslan, Edgar Marroquin, Justin T Chiu, Alexander Rush, and Volodymyr Kuleshov. Simple and effective masked diffusion language models. *arXiv preprint arXiv:2406.07524*, 2024.

Jiaxin Shi, Kehang Han, Zhe Wang, Arnaud Doucet, and Michalis K Titsias. Simplified and generalized masked diffusion for discrete data. *arXiv preprint arXiv:2406.04329*, 2024.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pp. 2256–2265. PMLR, 2015.

Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in neural information processing systems*, 32, 2019.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *The Ninth International Conference on Learning Representations*, 2021.

Clement Vignac, Igor Krawczuk, Antoine Siraudin, Bohan Wang, Volkan Cevher, and Pascal Frossard. Digress: Discrete denoising diffusion for graph generation. In *The Eleventh International Conference on Learning Representations*, 2023.

Lei Xu, Maria Skoularidou, Alfredo Cuesta-Infante, and Kalyan Veeramachaneni. Modeling tabular data using conditional gan. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pp. 7335–7345, 2019.

Yilun Xu, Mingyang Deng, Xiang Cheng, Yonglong Tian, Ziming Liu, and Tommi Jaakkola. Restart sampling for improving generative processes. *Advances in Neural Information Processing Systems*, 36:76806–76838, 2023.

Jiaxuan You, Xiaobai Ma, Yi Ding, Mykel J Kochenderfer, and Jure Leskovec. Handling missing data with graph representation learning. *Advances in Neural Information Processing Systems*, 33: 19075–19087, 2020.

Hengrui Zhang, Jiani Zhang, Zhengyuan Shen, Balasubramaniam Srinivasan, Xiao Qin, Christos Faloutsos, Huzefa Rangwala, and George Karypis. Mixed-type tabular data synthesis with score-based diffusion in latent space. In *The Twelfth International Conference on Learning Representations*, 2024.

Shuhan Zheng and Nontawat Charoenphakdee. Diffusion models for missing value imputation in tabular data. *arXiv preprint arXiv:2210.17128*, 2022.

A Detailed Experiment Setups

A.1 Datasets

We use seven tabular datasets from UCI Machine Learning Repository[1]: Adult, Default, Shoppers, Magic, Beijing, News, and Diabetes, where each tabular dataset is associated with a machine-learning task. Classification: Adult, Default, Magic, Shoppers, and Diabetes. Regression: Beijing and News. The statistics of the datasets are presented in Table 3.

Table 3: Statistics of datasets. # Num stands for the number of numerical columns, and # Cat stands for the number of categorical columns. # Max Cat stands for the number of categories of the categorical column with the most categories.

Dataset	# Rows	# Num	# Cat	# Max Cat	# Train	# Validation	# Test	Task
Adult	$48,842$	6	9	42	$28,943$	$3,618$	$16,281$	Classification
Default	$30,000$	14	11	11	$24,000$	$3,000$	$3,000$	Classification
Shoppers	$12,330$	10	8	20	$9,864$	$1,233$	$1,233$	Classification
Magic	$19,019$	10	1	2	$15,215$	$1,902$	$1,902$	Classification
Beijing	$43,824$	7	5	31	$35,058$	$4,383$	$4,383$	Regression
News	$39,644$	46	2	7	$31,714$	$3,965$	$3,965$	Regression
Diabetes	$101,766$	9	27	716	$61,059$	$2,0353$	$20,354$	Classification

A.2 Metrics

A.2.1 Shape and Trend

Shape and Trend are proposed by SDMetrics[2]. They are used to measure the column-wise density estimation performance and pair-wise column correlation estimation performance, respectively. Shape uses Kolmogorov-Sirnov Test (KST) for numerical columns and the Total Variation Distance (TVD) for categorical columns to quantify column-wise density estimation. Trend uses Pearson correlation for numerical columns and contingency similarity for categorical columns to quantify pair-wise correlation.

Shape. *Kolmogorov-Sirnov Test (KST)*: Given two (continuous) distributions $p_r(x)$ and $p_s(x)$ (r denotes real and s denotes synthetic), KST quantifies the distance between the two distributions using the upper bound of the discrepancy between two corresponding Cumulative Distribution Functions (CDFs):

$$\text{KST} = \sup_x |F_r(x) - F_s(x)|, \tag{17}$$

where $F_r(x)$ and $F_s(x)$ are the CDFs of $p_r(x)$ and $p_s(x)$, respectively:

$$F(x) = \int_{-\infty}^{x} p(x)\mathrm{d}x. \tag{18}$$

Total Variation Distance: TVD computes the frequency of each category value and expresses it as a probability. Then, the TVD score is the average difference between the probabilities of the categories:

$$\text{TVD} = \frac{1}{2}\sum_{\omega\in\Omega} |R(\omega) - S(\omega)|, \tag{19}$$

where ω describes all possible categories in a column Ω. $R(\cdot)$ and $S(\cdot)$ denotes the real and synthetic frequencies of these categories.

Trend. *Pearson Correlation Coefficient*: The Pearson correlation coefficient measures whether two continuous distributions are linearly correlated and is computed as:

$$\rho_{x,y} = \frac{\text{Cov}(x,y)}{\sigma_x\sigma_y}, \tag{20}$$

[1] https://archive.ics.uci.edu/datasets

[2] https://docs.sdv.dev/sdmetrics

where x and y are two continuous columns. Cov is the covariance, and σ is the standard deviation.

Then, the performance of correlation estimation is measured by the average differences between the real data's correlations and the synthetic data's corrections:

$$\text{Pearson Score} = \frac{1}{2}\mathbb{E}_{x,y}|\rho^R(x,y) - \rho^S(x,y)|, \tag{21}$$

where $\rho^R(x,y)$ and $\rho^S(x,y))$ denotes the Pearson correlation coefficient between column x and column y of the real data and synthetic data, respectively. As $\rho \in [-1,1]$, the average score is divided by 2 to ensure that it falls in the range of $[0,1]$, then the smaller the score, the better the estimation.

Contingency similarity: For a pair of categorical columns A and B, the contingency similarity score computes the difference between the contingency tables using the Total Variation Distance. The process is summarized by the formula below:

$$\text{Contingency Score} = \frac{1}{2}\sum_{\alpha\in A}\sum_{\beta\in B}|R_{\alpha,\beta} - S_{\alpha,\beta}|, \tag{22}$$

where α and β describe all the possible categories in column A and column B, respectively. $R_{\alpha,\beta}$ and $S_{\alpha,\beta}$ are the joint frequency of α and β in the real data and synthetic data, respectively.

A.2.2 α-Precision and β-Recall

Following Liu et al. (2023) and Alaa et al. (2022), we adopt the α-Precision and β-Recall proposed in Alaa et al. (2022), two sample-level metric quantifying how faithful the synthetic data is. In general, α-Precision evaluates the fidelity of synthetic data – whether each synthetic example comes from the real-data distribution, β-Recall evaluates the coverage of the synthetic data, e.g., whether the synthetic data can cover the entire distribution of the real data (In other words, whether a real data sample is close to the synthetic data.)

A.2.3 Detection

The detection measures the difficulty of detecting the synthetic data from the real data when they are mixed. We use the classifer-two-sample-test (C2ST) implemented by SDMetrics, where a logistic regression model plays the role of a detector.

A.2.4 Machine Learning Efficiency

In MLE, each dataset is first split into the real training and testing set. The generative models are learned on the real training set. After training, a synthetic set of equivalent size is sampled.

The performance of synthetic data on MLE tasks is evaluated based on the divergence of test scores using the real and synthetic training data. Therefore, we first train the machine learning model on the real training set, split into training and validation sets with a $8:1$ ratio. The classifier/regressor is trained on the training set, and the optimal hyperparameter setting is selected according to the performance on the validation set. After the optimal hyperparameter setting is obtained, the corresponding classifier/regressor is retrained on the training set and evaluated on the real testing set. The performance of synthetic data is obtained in the same way.

A.3 Method

A.3.1 Adaptively Learnable Noise Schedule

For numerical stability, we need to bound σ_{min} and σ_{max} within $(0,1)$. As shown in Eq. (10), our formulation of the power-mean noise schedule boundaries the noise level in between σ_{min} and σ_{max}. To make sure that the noise level for numerical features is also bounded, we linearly map t to the interval $[\delta, 1-\delta]$, thus recasting Eq. (11) into

$$\sigma^{\text{cat}}_{k_j}(t) = -\log\big(1 - \big((1-\delta)\cdot t^{k_j} + \delta\big)\big). \tag{23}$$

B Detailed Experiments Results

In the following sections, we discuss the results of all eight metrics across all seven datasets in detail.

Table 4: Performance comparison on the error rates (%) of **Shape**.

Method	Adult	Default	Shoppers	Magic	Beijing	News	Diabetes	Average
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	9.82±0.08	15.99
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	18.86±0.13	15.97
GOGGLE	16.97	17.02	22.33	1.90	16.93	25.32	24.92	17.91
GReaT	12.12±0.04	19.94±0.06	14.51±0.12	16.16±0.09	8.25±0.12	OOM	OOM	14.20
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	OOM	7.72
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	21.13±0.25	21.55
TabDDPM	1.75±0.03	1.57± 0.08	2.72±0.13	1.01±0.09	1.30±0.03	78.75±0.01	31.44±0.05	16.93
TabSyn [1]	0.81±0.05	**1.01±0.08**	1.44±0.07	1.03±0.14	1.26±0.05	**2.06±0.04**	1.85±0.02	1.35
TabDiff	**0.63±0.05**	1.24±0.07	**1.28±0.09**	**0.78±0.08**	**1.03±0.05**	2.35±0.03	**0.89±0.23**	**1.17**
Improv.	22.2% ↓	0.0% ↓	11.11% ↓	14.29% ↓	18.25% ↓	0% ↓	46.39% ↓	13.3% ↓

[1] TabSyn's performance is obtained via our reproduction. The results of other baselines except on Diabetes, are taken from Zhang et al. (2024). The OOM entries are explained in Appendix C.

Table 5: Performance comparison on the error rates (%) of **Trend**.

Method	Adult	Default	Shoppers	Magic	Beijing	News	Diabetes	Average
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	18.95±0.34	16.36
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	32.74±0.26	16.44
GOGGLE	45.29	21.94	23.90	9.47	45.94	23.19	27.56	28.18
GReaT	17.59±0.22	70.02±0.12	45.16±0.18	10.23±0.40	59.60±0.55	OOM	OOM	44.24
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	OOM	7.77
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	29.21±0.12	23.21
TabDDPM	3.01±0.25	4.89±0.10	6.61±0.16	1.70±0.22	2.71±0.09	13.16±0.11	51.54±0.05	11.95
TabSyn	1.93±0.07	2.81±0.48	2.13±0.10	0.88±0.18	3.13±0.34	1.52±0.03	3.90±0.04	2.33
TabDiff	**1.49±0.16**	**2.55±0.75**	**1.74±0.08**	**0.76±0.12**	**2.59±0.15**	**1.28±0.04**	**2.20±0.16**	**1.80**
Improve.	22.8% ↓	9.3% ↓	18.3% ↓	13.6% ↓	4.4% ↓	15.8% ↓	37.3% ↓	22.6% ↓

B.1 Data Fidelity

Shape and Trend. We first evaluate the fidelity of synthetic data using the Shape and Trend metrics. Shape measures the synthetic data's ability to capture each single column's marginal density, while Trend assesses its capacity to replicate the correlation between different columns in the real data.

The detailed results for Shape and Trend metrics, measured across each dataset separately, are presented in Tables 4 and 5, respectively. On the Shape metric, TabDiff outperforms all baselines on five out of seven datasets and surpasses the current state-of-the-art method TabSyn by an average of 13.3%. This demonstrates TabDiff's superior performance in maintaining the marginal distribution of individual attributes across various datasets. Regarding the Trend metric, TabDiff consistently outperforms all baselines and surpasses TabSyn by 22.6%. This significant improvement suggests that TabDiff is substantially better at capturing column-column relationships than previous methods. We also highlight TabDiff's exceptional performance on the larger, more categorical-heavy dataset Diabetes, demonstrating our model's capacity to model datasets with higher dimensionality and discrete features.

α-precision. We first evaluate TabDiff on α-Precision score, a metric that measures the quality of synthetic data. Higher scores indicate the synthetic data is more faithful to the real. We present the results across all seven datasets in Table 6. TabDiff achieves the best or second-best performance on all datasets. Specifically, TabDiff ranks first with an average α-Precision score of 98.22, surpassing all other baseline methods.

β-recall. Next, we compare TabDiff to the baselines on the β-Recall scores, which evaluates the extent to which synthetic data covers the real data distribution. The results are presented in Table 7, with a higher score reflecting a more comprehensive coverage of the real data's feature space. TabDiff consistently outperforms or matches the top-performing baselines, achieving the highest average β-Recall score of 49.40. This indicates that the generated data spans a broad range of the real distribution. Though some baseline methods attained higher scores on specific datasets, they fail to demonstrate competitive performance on α-Precision, as models have to trade off fine-grained details in order to capture a broader range of features.

Overall, TabDiff maintains a balance between broad data coverage and preserving fine-grained details. This balance highlights TabDiff 's capability in generating synthetic data that faithfully captures both the breadth and depth of the original data distribution.

Detection Score (C2ST). Lastly, we assess the fidelity of synthetic data by using the C2ST test, which evaluates how difficult it is to distinguish the synthetic data from the real data. The results are shown in Table 8, where a higher score indicates better fidelity. TabDiff achieves the best performance on five of seven datasets, outperforming the most competitive baseline model by 6.89% on average. Notably, TabDiff excels on Diabetes, which contains many numerous high-cardinality categorical features (as indicated by *# Max Cat* in Table 3), showcasing its ability to generate high-quality categorical data. These results, therefore, demonstrate TabDiff's capacity to generate synthetic data that closely resembles the real data.

Table 6: Comparison of α-Precision scores. **Bold Face** highlights the best score for each dataset. Higher scores reflect better performance.

Methods	Adult	Default	Shoppers	Magic	Beijing	News	Diabetes	Average	Ranking
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	79.89±0.10	82.40	5
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	19.24±0.15	75.85	7
GOGGLE	50.68	68.89	86.95	90.88	88.81	86.41	23.09	70.81	9
GReaT	55.79±0.03	85.90±0.17	78.88±0.13	85.46±0.54	**98.32±0.22**	OOM	OOM	80.87	6
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	OOM	88.91	3
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	64.80±0.53	84.29	4
TabDDPM	96.36±0.20	97.59±0.36	88.55±0.68	98.59±0.17	97.93±0.30	0.00±0.00	28.35±0.11	72.48	8
TabSyn	**99.39±0.18**	**98.65±0.23**	98.36±0.52	99.42±0.28	97.51±0.24	95.05±0.30	**96.61±0.24**	97.86	2
TabDiff	99.02±0.20	98.49±0.28	**99.11±0.34**	**99.47±0.21**	98.06±0.24	**97.36±0.17**	95.69±0.19	**98.22**	1

Table 7: Comparison of β-Recall scores. **Bold Face** highlights the best score for each dataset. Higher scores reflects better results.

Methods	Adult	Default	Shoppers	Magic	Beijing	News	Diabetes	Average	Ranking
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	9.42±0.26	23.11	8
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	4.92±0.13	25.32	7
GOGGLE	8.80	14.38	9.79	9.88	19.87	2.03	3.74	9.78	9
GReaT	49.12±0.18	42.04±0.19	44.90±0.17	34.91±0.28	43.34±0.31	OOM	OOM	43.34	3
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	OOM	42.32	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	2.70±0.06	27.12	6
TabDDPM	47.05±0.25	47.83±0.35	47.79±0.25	**48.46±0.42**	56.92±0.13	0.00±0.00	0.03±0.01	35.44	5
TabSyn	47.92±0.23	46.45±0.35	49.10±0.60	48.03±0.50	59.15±0.22	**43.01±0.28**	33.72±0.16	46.77	2
TabDiff	**51.64±0.20**	**51.09±0.25**	**49.75±0.64**	48.01±0.31	**59.63±0.23**	42.10±0.32	**41.74±0.17**	**49.40**	1

Table 8: Detection score (C2ST) using logistic regression classifier. Higher scores reflect superior performance.

Method	Adult	Default	Shoppers	Magic	Beijing	News	Diabetes	Average
CTGAN	0.5949	0.4875	0.7488	0.6728	0.7531	0.6947	0.5593	0.6444
TVAE	0.6315	0.6547	0.2962	0.7706	0.8659	0.4076	0.0487	0.5250
GOGGLE	0.1114	0.5163	0.1418	0.9526	0.4779	0.0745	0.0912	0.3380
GReaT	0.5376	0.4710	0.4285	0.4326	0.6893	OOM	OOM	0.5118
STaSy	0.4054	0.6814	0.5482	0.6939	0.7922	0.5287	OOM	0.6083
CoDi	0.2077	0.4595	0.2784	0.7206	0.7177	0.0201	0.0008	0.3435
TabDDPM	0.9755	0.9712	0.8349	**0.9998**	0.9513	0.0002	0.1980	0.7044
TabSyn	0.9910	**0.9826**	0.9662	0.9960	0.9528	0.9255	0.5953	0.9156
TabDiff	**0.9950**	0.9774	**0.9843**	0.9989	**0.9781**	**0.9308**	**0.9865**	**0.9787**
Improv.	0.40% ↓	0.0% ↓	1.87% ↓	0.0% ↓	2.66% ↓	0.57% ↓	65.71% ↓	6.89% ↓

B.2 Performance on Downstream Tasks

Machine Learning Efficiency. A key advantage of high-quality synthetic data is its ability to serve as an anonymized proxy for real datasets and power effective learning on downstream tasks such as classification and regression. We measure the synthetic table's capacity to support downstream task learning via Machine Learning Efficiency (MLE). Following established protocols (Kim et al., 2023; Lee et al., 2023; Xu et al., 2019), we first split the real dataset into training and test sets, then train the

Table 9: Evaluation of **MLE** (Machine Learning Efficiency): AUC and RMSE are used for classification and regression tasks, respectively.

Methods	Adult	Default	Shoppers	Magic	Beijing	News[1]	Diabetes	Average Gap
	AUC ↑	AUC ↑	AUC ↑	AUC ↑	RMSE ↓	RMSE ↓	AUC ↑	%
Real	.927±.000	.770±.005	.926±.001	.946±.001	.423±.003	.842±.002	.704±.002	0.0
CTGAN	.886±.002	.696±.005	.875±.009	.855±.006	.902±.019	.880±.016	.569±.004	23.7
TVAE	.878±.004	.724±.005	.871±.006	.887±.003	.770±.011	1.01±.016	.594±.009	20.2
GOGGLE	.778±.012	.584±.005	.658±.052	.654±.024	1.09±.025	.877±.002	.475±.008	42.1
GReaT	.913±.003	.755±.006	.902±.005	.888±.008	.653±.013	OOM	OOM	13.3
STaSy	.906±.001	.752±.006	.914±.005	.934±.003	.656±.014	.871±.002	OOM	10.9
CoDi	.871±.006	.525±.006	.865±.006	.932±.003	.818±.021	1.21±.005	.505±.004	30.2
TabDDPM	.907±.001	.758±.004	.918±.005	.935±.003	.592±.011	4.86±3.04	.521±.008	11.95
TabSyn	.909±.001	**.763±.002**	.914±.004	**.937±.002**	.580±.009	**.862±.024**	.684±.002	6.78
TabDiff	**.912±.002**	**.763±.005**	**.921±.004**	.936±.003	**.555±.013**	.866±.021	**.689±.016**	**5.76**

Table 10: Performance of TabDiff in the Missing Value Imputation task. We draw a direct comparison to the generative approach employed by TabSyn, with the performance of XGBoost classifiers/regressors included as a reference.

Methods	Adult	Default	Shoppers	Magic	Beijing	News	Diabetes	Avg. Improv.
	AUC ↑	AUC ↑	AUC ↑	AUC ↑	RMSE ↓	RMSE ↓	AUC ↑	%
Predicted by XGBoost	92.7	77.0	92.6	94.6	0.423	0.842	70.4	0.0
Impute with TabSyn	93.1	86.7	**96.5**	91.3	**0.386**	0.818	66.6	4.99
Impute with TabDiff + CFG ($\omega = 0.0$)	92.5	91.6	95.7	92.5	0.424	0.828	66.0	3.76
Impute with TabDiff + CFG ($\omega = 0.6$)	**93.2**	**91.7**	96.4	**93.0**	0.414	**0.815**	**66.9**	**5.60**

given generative model on the real training set. Subsequently, we sample a synthetic dataset of equal size to the real training set from the models and use it to train an XGBoost Classifier or XGBoost Regressor (Chen & Guestrin, 2016). Finally, we evaluate these machine learning models against the real test set to calculate the AUC score and RMSE for classification and regression tasks, respectively.

According to the MLE results presented in Table 9, TabDiff consistently achieves the best or second-best performance across all datasets, with the highest average performance outperforming the most competitive baseline TabSyn by 15.0%. This demonstrates our method's competitive capacity to capture and replicate key features of the real data that are most relevant to learning downstream machine learning tasks. However, while TabDiff shows strong performance on MLE, we observe that methods with varying performance on data fidelity metrics might have very close MLE scores. This suggests that the MLE score evaluated under the current setting may not be a reliable indicator of data quality. Therefore, we complement MLE with additional data quality metrics beyond Shape and Trend.

Missing Value Imputation. We further evaluate TabDiff's conditional generation capacity through the Missing Value Imputation task. Following the approach in Zhang et al. (2024), we treat the inherent classification/regression task of each dataset as an imputation task. Specifically, for each table, we train generative models on the training set to generate the target column while conditioning on the remaining columns. The imputation performance is measured by the model's accuracy in recovering the target column of the test set. Implementing classifier-free guidance (CFG) for this task is straightforward. We approximate the conditional model using the unconditioned TabDiff trained on all columns from the previous unconditional generation tasks. For the unconditional model, we train TabDiff on the target column with a significantly smaller denoising network. Detailed implementation is provided in Appendix C, and results are presented in Table 10.

As demonstrated, TabDiff achieves higher imputation accuracy than TabSyn on five out of seven datasets, with an average improvement of 5.60% over the non-generative XGBoost classifier. This indicates TabDiff's superior capacity for conditional tabular data generation. Moreover, we empirically demonstrate the efficacy of our CFG framework by showing that the model consistently performs better with $\omega = 0.6$ compared to $\omega = 0.0$ (which is equivalent to TabDiff without CFG).

Table 11: DCR score, which represents the probability that a generated data sample is more similar to the training set than to the test set. A score closer to 50% is more preferable. **Bold Face** highlights the best score for each dataset.

Method	Adult	Default	Shoppers	Beijing	News	Diabetes
STaSy	50.33% ±0.19	50.23% ±0.09	51.53% ±0.16	50.59% ±0.29	**50.59% ±0.14**	OOM
CoDi	49.92% ±0.18	51.82% ±0.26	51.06% ±0.18	50.87% ±0.11	50.79% ±0.23	51.12% ±0.19
TabDDPM	51.14% ±0.18	52.15% ±0.20	63.23% ±0.25	80.11% ±2.68	79.31% ±0.29	37.76% ±0.23
TabSyn	50.94% ±0.17	51.20% ±0.18	52.90% ±0.22	**50.37% ±0.13**	50.85% ±0.33	50.62% ±0.28
TabDiff	**50.10 % ±0.32**	**51.11 % ±0.36**	**50.24 % ±0.62**	50.50 % ±0.36	51.04 % ±.32	**50.43 % ±0.18**

B.3 Data Privacy

The ability to protect privacy is another important factor when evaluating synthetic data since we wish the synthetic data to be uniformly sampled from the data distribution manifold rather than being copied (or slightly modified) from each individual real data example. In this section, we use the Distance to Closest Records (DCR) score metric (Zhang et al., 2024), which measures the probability that a synthetic example's nearest neighbor is from a holdout v.s. the training set.

Table 11 shows the DCR scores across all datasets. The DCR score represents the probability that a generated data sample is more similar to the training set than to the test set, with a score closer to 50% being ideal, as it indicates a balance between the similarity to training and test distributions. Across the datasets, TabDiff consistently achieves DCR scores near 50%, highlighting its ability to generalize well while maintaining fidelity to the original data distribution.

C Implementation Details.

We perform our experiment on an Nvidia RTX A4000 GPU with 16G memory and implement TabDiff with PyTorch.

Data preprocessing. The raw tabular datasets usually contain missing entries. Thus in the first phase of preprocessing, we make up these missing values in the same way as existing works (Kotelnikov et al., 2023; Zhang et al., 2024), with numerical missing values being replaced by the column average and categorical missing values being treated as a new category. Moreover, the diverse range of numerical features typically leads to more difficult and unstable training. To counter this, we then transform the numerical values with the QuantileTransformer[3], and recover the original values using its inverse during sampling.

Data splits. For datasets other than Diabetes, we follow the exact same split as Zhang et al. (2024). Each dataset is split into the "real" and "test" sets. For the unconditional generation task on which data fidelity and the imputation task, the models are trained on the "real" set and evaluated on the "real" set. For the MLE task, the "real" set is further split into a training and validation set, and the "test" set is used for testing. Finally, for the data privacy measure DCR, the original dataset is equally split into two halves, with one being treated as the training set and the other as the holdout set.

For Diabetes, we split it into train, validation, and test sets with a ratio of 0.6/0.2/0.2. For the MLE task. The training and test sets are regarded as the "real" and "test" sets for the unconditional generation and imputation tasks. For DCR, we apply an equal split.

Architecture of the denoising network. In our implementation, we project each column individually to a d dimensional vector using a linear layer, ensuring that all columns are treated with the same importance. We set the embedding size d as 4, matching the size used in Zhang et al. (2024). We then process these projected vectors with a two-layer transformer, appending positional encodings at the end. The transformed vectors are then concatenated and passed through a five-layer MLP conditioned on the time embedding. Finally, the output is obtained by sequentially applying another transformer followed by a projection layer that recovers the original dimensions. Our denoising network has a comparable number of parameters as those experimented in TabDDPM (Kotelnikov et al., 2023) and TabSyn (Zhang et al., 2024), as our shared MLP model accounts for most of the parameters.

[3] https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.QuantileTransformer.html

Hyperparameters Setting. TabDiff employs the same hyperparameter setting for all datasets. We train our models for 8000 epochs with the Adam optimizer. The training and sampling batch sizes are set to 4,096 and 10,000, respectively. Regarding the hyperparameters in TabDiff, the values σ_{min} and σ_{max} are set to 0.002 and 80.0, referencing the optimal setting in Karras et al. (2022), and δ are set to $1\text{e}{-3}$. For the loss weightings, we fix λ_{cat} to 1.0 and linear decay λ_{num} from 1.0 to 0.0 as training proceeds.

During inference, we select the checkpoint with the lowest training loss. We observe that our model achieves the superior performance reported paper with as few as 50 discretization steps ($T = 50$).

Details on OOMs in experiment result tables:

1. GOOGLE set fixed random seed during sampling in the official codes, and we follow it for consistency.
2. GReaT cannot be applied on News for maximum length limit.
3. STaSy runs out of memory on Diabetes that has hight cardinality categorical columns
4. TabDDPM cannot produce meaningful content on the News dataset.

Imputation. As mentioned in Appendix B.2, we obtain the unconditional model of the target column by training TabDiff on it with a smaller denoising network. For this network, we keep the same architecture but reduce the number of MLP layers to one.