TABREP: Training Tabular Diffusion Models with a Simple and Effective Continuous Representation

Anonymous Authors[1]

Abstract

Diffusion models for tabular data generation face a conundrum between separate and unified data representations. The former struggles with jointly capturing multi-modal distributions, while the latter often relies on sparse, suboptimal encodings and incurs high computational costs. In this work, we address the latter by presenting TABREP, a diffusion architecture trained with a unified, continuous representation tailored for tabular data. Motivated by geometric insights of the data manifold, our representation is dense, separable, and preserves intrinsic relationships. TABREP achieves state-of-the-art performance, synthesizing data that surpasses the original in downstream quality, while maintaining privacy and efficiency.

1. Introduction

The best-performing tabular generative models have been based on diffusion models (Ho et al., 2020b; Song & Ermon, 2020)—both continuous and discrete (Lee et al., 2023; Kotelnikov et al., 2023; Shi et al., 2024b; Hoogeboom et al., 2021)—to model mixed-type tabular data. However, these approaches often require complex multimodal objectives or rely on heuristics like one-hot encodings (Kim et al., 2022) or latent embeddings via β-VAEs (Zhang et al., 2023), which introduce sparsity or additional computational overhead (Krishnan et al., 2017; Poslavskaya & Korolev, 2023; Higgins et al., 2017; Kingma & Welling, 2013).

Since diffusion models rely on continuous transformations of denoising score-matching, or invertible mappings between data and latent spaces, designing an effective data representation is critical (Bengio et al., 2014). In this work, we propose TABREP, a simple and effective continuous representation tailored for tabular diffusion. Our design is motivated by geometric insights on the tabular data manifold, promoting separability for nominal features, supporting ordinal encoding via cyclicality, and maintaining compactness to avoid the curse of dimensionality. These attributes present desirable characteristics that make it easier for diffusion models to extract meaningful information from a unified continuous tabular data representation. Extensive experiments show that TABREP outperforms existing methods in fidelity, privacy, and efficiency across diverse benchmarks. Our architecture is in Figure 3.

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

2. Method

Tabular data consist of heterogeneous features. In our analysis, we represent a dataset with N rows as $\mathcal{D} = \{\mathbf{z}^{(i)}\}_{i=1}^{N} = \{[\mathbf{x}^{(i)}, \mathbf{c}^{(i)}]\}_{i=1}^{N}$, where $\mathbf{x}^{(i)} \in \mathbb{R}^{D_{\text{cont}}}$ are continuous features and $\mathbf{c}^{(i)} \in \prod_{j\in\{1,\dots,D_{\text{cat}}\}}\{1,\dots,K_j\}$ are categorical features, with K_j being the number of unique categories for the j-th categorical feature. Tabular data undergo preprocessing before the diffusion process. Designing an effective, unified continuous representation streamlines and improves model performance. See Appendix D.1 for related works.

2.1. Geometric Implications on the Data Representation

In traditional deep learning (Goodfellow et al., 2016), a sparse representation suffers from the curse of dimensionality (Bellman, 1957), where the feature space grows exponentially with the number of categories, reducing model generalization (Krishnan et al., 2017; Poslavskaya & Korolev, 2023). While maintaining a dense representation is key, representation learning also (Bengio et al., 2013; LeCun et al., 2015) emphasizes the importance of separability, enabling the neural network to learn decision boundaries in the continuous embedding space. In the following excerpt, we find that balancing both *density* and *separability* while incorporating *order* is essential for unified tabular diffusion.

Density. Diffusion models, which learn to generate data through iterative perturbations and reconstructions of noise, encounter geometric challenges when applied to high-dimensional, sparse representations of categorical features. In this discussion, we use the sparse one-hot representation as an exemplar to provide geometric insights into these

challenges.

Let $\{e_1, e_2, \ldots, e_K\} \subset \mathbb{R}^K$ denote the set of one-hot vectors, where

$$e_k = (0, \ldots, 0, \underbrace{1}_{k\text{-th entry}}, 0, \ldots, 0), \tag{1}$$

represents the k-th category. For any point $x \in \mathbb{R}^K$, let

$$d_k(x) = \|x - e_k\|_2, \tag{2}$$

denote the Euclidean distance between x and e_k. On the data manifold, there exist "singular" (Wikipedia contributors, 2025) points where the vector field is hard to learn for diffusion models with Gaussian transitions. We demonstrate this with a uniform K-categorical distribution. Specifically, we define:

Definition 2.1 (n-singular point). *A point $x \in \mathbb{R}^K$ is an n-singular point if there exists a subset $\mathcal{S} \subseteq \{1, \ldots, K\}$ with $|\mathcal{S}| = n$ such that*:

1. $d_k(x) = d_{k'}(x), \quad \forall k, k' \in \mathcal{S}.$
2. $d_k(x) \neq d_m(x), \quad \forall k \in \mathcal{S}, \forall m \notin \mathcal{S}.$

An n-singular point can be extended as a *minimal* n-singular point if it satisfies Definition 2.1, and minimizes the Euclidean distance to all one-hot vectors in $\mathcal{S}$: $\min_x \|x - e_k\|_2$ $\forall k \in \mathcal{S}$. Hence, the minimal n-singular point is given by:

$$x_{\mathcal{S}}^{(n)} = \frac{1}{n} \sum_{k \in \mathcal{S}} e_k, \tag{3}$$

that corresponds to the centroid of the n one-hot vectors.

Definition 2.2 (n-singular hyperplane). *For each minimal n-singular point, there exists an n-singular hyperplane that is comprised of the set of all n-singular points associated with its respective n one-hot vectors in $\mathcal{S}$.* Formally, it is defined as:

$$H_{\mathcal{S}} = \left\{ x \in \mathbb{R}^K \mid d_k(x) = d_{k'}(x), \quad \forall k, k' \in \mathcal{S} \right\}, \tag{4}$$

where $H_{\mathcal{S}}$ is an affine subspace in $\mathbb{R}^K$ of dimension $\dim(H_{\mathcal{S}}) = K - |\mathcal{S}| + 1$. The hyperplane spans the corresponding minimal n-singular point $x_{\mathcal{S}}^{(n)}$ and non-minimal n-singular points. For each $n < K$, there are $\binom{K}{n}$ distinct n-singular hyperplanes, one for each subset $\mathcal{S}$ of size n. Across all $2 \leq n \leq K$, the number of minimal n-singular points on the probability simplex scales combinatorially: $\sum_{n=2}^{K} \binom{K}{n} = 2^K - (K+1)$. Thus, each minimal singular point carries the additional complexity of a continuous singular hyperplane.

Diffusion models rely on gradients derived from a learned vector field to denoise data iteratively (Ho et al., 2020a; Lipman et al., 2022). For regions within proximity of the singular hyperplanes, learning the gradients of diffusion models suffers from high variance due to conflicting directions arising from equidistant one-hot points. We show that the variance of the conditional score function increases asymptotically with the degree of n-singular points.

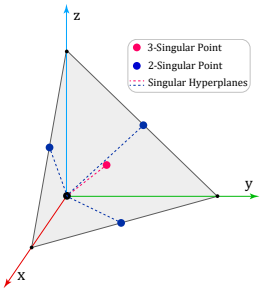


Figure 1. Singular Regions in a 3D One-Hot Setting.

Theorem 2.1 (Variance of Conditional Score Function)**.** *Assume x is a noisy observation from a Gaussian centered at a weighted one-hot vector $\alpha_t e_k \in \mathbb{R}^K$. We can define the forward diffusion process as: $p_t(x|e_k) = \mathcal{N}(x|\alpha_t e_k, \sigma_t^2 I)$. We derive the variance of the conditional score function evaluated at a minimal n-singular point as:*

$$\mathrm{Var}(g|x) = \frac{\alpha_t^2}{\sigma_t^4} \frac{n-1}{n}, \tag{5}$$

where we define the conditional and expected score as g and $\bar{g}$. See Appendix C.1 for proofs. We find that near a minimal n-singular point, x, the posterior-weighted variance of the conditional score function is strictly positive and increases asymptotically with n. In contrast, at a non-singular point, the posterior leans towards e_{k^*}, leading the score variance to approach zero.

In Figure 1, we depict a three-dimensional setting of the one-hot representation. As illustrated, a minimal 3-singular point occurs at the (red) centroid, $\left(\frac{1}{3}, \frac{1}{3}, \frac{1}{3}\right)$. Along with the non-minimal 3-singular points, these points form a (red-dashed) 1-dimensional singular hyperplane $H_{\{1,2,3\}}$ perpendicular to the centroid of the probability simplex formed by e_1, e_2, e_3. Similarly, there are $\binom{3}{2}$ 2-singular (blue) points accompanied by their respective (blue-dashed) hyperplanes $H_{\{1,2\}}$, $H_{\{1,3\}}$, and $H_{\{2,3\}}$, each of which is a 2-dimensional affine subspace (plane). These singular hyperplanes pose more difficulty for diffusion models to learn effectively, especially when K is large.

Separability and Order. A sparse representation naturally accommodates separability for nominal features, enabling one-hot to assign each category to each dimension. However, higher-dimensional spaces introduce higher-order singular hyperplanes that complicate training. This presents a density-separability trade-off. In our experiments (Table 10), we discover that *sparse representations has a signifi-*

REPRESENTATION	DIMENSIONS
ONE-HOT	$\sum_{j=1}^{D_{\text{CAT}}} K_j$
LEARNED EMBEDDING	$d_{\text{EMB}} \cdot D_{\text{CAT}}$
ANALOG BITS	$\sum_{j=1}^{D_{\text{CAT}}} \lceil \log_2(K_j) \rceil$
DICTIONARY	D_{CAT}
CATCONVERTER	$2 \cdot D_{\text{CAT}}$

Table 1. Categorical Representation Dimensions.

cant impact on harming diffusion generative performance. Thus, our initial goal is to reduce the dimensionality when designing our representation.

Methods like Analog Bits (Chen et al., 2022) admit a similar idea of shrinking data dimension. Hence, a potential concern of a dense representation is to retain their ability to represent nominal features. However, the notion of separability (Bengio et al., 2014) demonstrates that nominal features can still be effectively encoded by dense representations, provided they are *sufficiently separated within the embedding space*. This perspective aligns with learned entity embeddings (Guo & Berkhahn, 2016), which demonstrate that nominal categories—despite lacking inherent order—can be effectively represented as low-dimensional embeddings. However, for datasets with a large presence of ordinal features such as "Education" in Adult and "Day" in Beijing, our experiment in Appendix D.3, Table 3 highlights that *order is an important factor for ordinal features*. Therefore, designing a dense representation that is separable and capable of encoding ordinal structure is critical for encoding both nominal and ordinal features.

2.2. TABREP Architecture

CatConverter. Inspired by the discrete Fourier transform (DFT) (Oppenheim, 1999; Bracewell & Kahn, 1966), we draw on the concept of roots of unity to design a continuous representation for diffusion models to generate categorical variables. We refer to our representation as the CatConverter. In harmonic analysis, the DFT maps signals into the frequency domain using complex exponentials, where the K-th roots of unity represent equally spaced points on the unit circle, given by phases:

$$\theta_k = \frac{2\pi k}{K_j^{(i)}} \quad \text{for } k = 0, 1, \ldots, K_j^{(i)} - 1. \tag{6}$$

Analogously, we treat a categorical feature $c_j^{(i)}$ with $K_j^{(i)}$ distinct values as selecting one of these $K_j^{(i)}$ points on the unit circle. Each category is thus mapped to a unique phase, and we represent it using the real and imaginary components of the corresponding complex exponential:

$$\text{CatConverter}(c_j^{(i)}, K_j^{(i)}) = \left[\cos\left(\frac{2\pi c_j^{(i)}}{K_j^{(i)}}\right), \sin\left(\frac{2\pi c_j^{(i)}}{K_j^{(i)}}\right)\right], \tag{7}$$

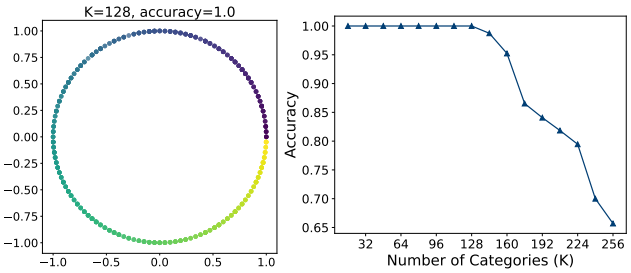


Figure 2. Separability of CatConverter. CatConverter preserves *nominal* representations for up to 128 categories.

where CatConverter() $\in \mathbb{R}^{2 \cdot D_{\text{CAT}}}$. Viewing a categorical entry as a discrete harmonic index enables us to embed the category in a two-dimensional phase space. This viewpoint retains the geometric insight from the DFT—the roots of unity form a symmetric and uniform structure on the unit circle—providing a smooth, dense, and geometry-aware representation for categorical variables.

In Table 1, CatConverter offers a dense 2D representation relative to alternate categorical representations. For CatConverter, there exists one minimal K-singular point. Despite that, and K number of 2-singular points, there are no n singular points or hyperplanes for $2 < n < K$. Therefore, any n-singular point for $n > 2$ coincides with the minimal K-singular point in this 2D representation.

Next, we show that CatConverter's representation has ample separability for handling high-cardinality nominal categorical features commonly found in tabular datasets. As illustrated in Figure 2, our representation could be easily distinguished by a small MLP among 128 categories mapped onto a continuous phase space.

In addition to nominal features, the separability of CatConverter coupled with its circular geometry naturally accommodates both periodic and ordinal features. This facilitates an enhanced preservation of the feature's intrinsic nature and characteristics. Note that prior to CatConverter, we introduced a static one-dimensional embedding, Dictionary (DIC), but we found that it underperformed.

We evaluate the performance of TABREP-DDPM and TABREP-FLOW against baselines. Information regarding implementation, datasets, baselines, benchmarks, as well as the ablation studies we perform can be found in Appendix E.1.

Diffusion Model. We demonstrate that TABREP's representation is effective when modeled by either a DDPM (Ho et al., 2020b) or a Flow Matching (Lipman et al., 2022) unified continuous diffusion process. To unify the dataspace, we represent discrete variables via CatConverter$(\mathbf{c}_0, \mathbf{K})$ and concatenate with continuous features forming our dataset. Our dataset can be denoted as $\{\mathbf{z}_0^{\text{CC}}\} = \{[\mathbf{x}_0, \mathbf{c}_0^{\text{CC}}]\}$.

Table 2. AUC, F1 (classification), and RMSE (regression) scores of Machine Learning Efficiency.

	AUC ↑				F1 ↑	RMSE ↓	
METHODS	ADULT	DEFAULT	SHOPPERS	STROKE	DIABETES	BEIJING	NEWS
REAL	$0.927_{\pm.000}$	$0.770_{\pm.005}$	$0.926_{\pm.001}$	$0.852_{\pm.002}$	$0.384_{\pm.003}$	$0.423_{\pm.003}$	$0.842_{\pm.002}$
STASY	$0.906_{\pm.001}$	$0.752_{\pm.006}$	$0.914_{\pm.005}$	$0.833_{\pm.030}$	$0.374_{\pm.003}$	$0.656_{\pm.014}$	$0.871_{\pm.002}$
CODI	$0.871_{\pm.006}$	$0.525_{\pm.006}$	$0.865_{\pm.006}$	$0.798_{\pm.032}$	$0.288_{\pm.009}$	$0.818_{\pm.021}$	$1.21_{\pm.005}$
TABDDPM	$0.910_{\pm.001}$	$0.761_{\pm.004}$	$0.915_{\pm.004}$	$0.808_{\pm.033}$	$0.376_{\pm.003}$	$0.592_{\pm.012}$	$3.46_{\pm1.25}$
TABSYN	$0.906_{\pm.001}$	$0.755_{\pm.004}$	$0.918_{\pm.004}$	$0.845_{\pm.035}$	$0.361_{\pm.001}$	$0.586_{\pm.013}$	$0.862_{\pm.021}$
TABDIFF	$0.912_{\pm.002}$	$0.763_{\pm.005}$	$0.919_{\pm.005}$	$0.848_{\pm.021}$	$0.353_{\pm.006}$	$0.565_{\pm.011}$	$0.866_{\pm.021}$
TABREP-DDPM	$\mathbf{0.913}_{\pm.002}$	$0.764_{\pm.005}$	$\mathbf{0.926}_{\pm.005}$	$\mathbf{0.869}_{\pm.027}$	$0.373_{\pm.003}$	$\mathbf{0.508}_{\pm.006}$	$0.836_{\pm.001}$
TABREP-FLOW	$0.912_{\pm.002}$	$\mathbf{0.782}_{\pm.005}$	$0.919_{\pm.005}$	$0.854_{\pm.028}$	$\mathbf{0.377}_{\pm.002}$	$0.536_{\pm.006}$	$\mathbf{0.814}_{\pm.002}$

For DDPM (Ho et al., 2020b), we define the forward process by progressively perturbing the data distribution using a Gaussian noise model, where the latent state at time t, $\mathbf{z}_t$, is computed as $\mathbf{z}_t = \alpha_t^{\mathbf{z}} \mathbf{z}_0^{\text{CC}} + \sigma_t^{\mathbf{z}} \boldsymbol{\epsilon}$ with $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I})$. The denoising model $p_\theta(\mathbf{z}_t)$ is trained to predict the posterior gradients $\nabla_{\mathbf{z}_t} \log q(\mathbf{z}_t|\mathbf{z}_0^{\text{CC}})$, minimizing the weighted variance loss $\mathcal{L}_{\text{TABREP-DDPM}}$:

$$\mathbb{E}_{t,\mathbf{z}_0^{\text{CC}},\mathbf{z}_t}\left[\left\|\nabla_{\mathbf{z}_t} \log q(\mathbf{z}_t|\mathbf{z}_0^{\text{CC}}) - \nabla_{\mathbf{z}_t} \log p_\theta(\mathbf{z}_t)\right\|^2\right]. \quad (8)$$

For Flow Matching (Lipman et al., 2022), we instead define the dynamics in terms of a conditional vector field $\mathbf{u}_t(\mathbf{z}_t|\mathbf{z}_0^{\text{CC}}) = \boldsymbol{\epsilon} - \mathbf{z}_0^{\text{CC}}$ with $\mathbf{z}_t := (1-t)\mathbf{z}_0^{\text{CC}} + t\boldsymbol{\epsilon}$. The model learns the target field by minimizing the discrepancy between the predicted vector $\mathbf{v}_\theta(\mathbf{z}_t)$ and the ground truth $\mathbf{u}_t(\mathbf{z}_t|\mathbf{z}_0^{\text{CC}})$ through the flow matching loss $\mathcal{L}_{\text{TABREP-FLOW}}$:

$$\mathbb{E}_{t,\mathbf{z}_0^{\text{CC}},\mathbf{z}_t}\left[\left\|\mathbf{v}_\theta(\mathbf{z}_t) - \mathbf{u}_t(\mathbf{z}_t|\mathbf{z}_0^{\text{CC}})\right\|^2\right]. \quad (9)$$

At sampling time, TABREP-DDPM performs reverse diffusion by iteratively denoising $\mathbf{z}_t$ back to $\mathbf{z}_0^{\text{CC}}$, while TABREP-FLOW solves a deterministic ordinary differential equation (ODE) of the vector field. Our complete training and sampling algorithms can be found in Figure 4.

3. Experiments

3.1. Experimental Setup

We evaluate the performance of TABREP against existing baselines across multiple datasets. Details on experimental setup, datasets, baselines, benchmarks, and ablations can be found in Appendix E.1.

3.2. Experimental Results

Generation Quality. We benchmark TABREP against baselines on a downstream machine learning efficiency task. As observed in Table 2, CatConverter demonstrates effectiveness on both DDPM and Flow Matching, consistently attaining the best MLE performance compared to existing baselines. Our method is also the first to yield performance levels greater than that using the real datasets for Default, Stroke, and News. The results of our ablation studies on representation schemes and encoding schemes also show superior performance, as seen in Table 9 and Table 10 respectively. We also provide additional experiments on Generation Quality using a number of other metrics for both our baseline and ablation experiments in Appendix G.1 and Appendix G.2 respectively.

Privacy Preservation. We perform membership inference attacks to evaluate privacy preservation by assessing the vulnerability of the methods to privacy leakage (Shokri et al., 2017). Table 11 and Appendix G.1 demonstrate that our method effectively preserves privacy with MIAs scoring close to 50% for recall and precision.

Generation Efficacy. Training and Sampling TABREP is the most efficient among all baselines and does not necessitate additional computing power. Our results in Table 12 show that TABREP is the quickest to train and sample in duration. We also conduct experiments on the convergence speed for the training process. As illustrated in Figure 5a, our method converges to a high AUC earliest in the training stages. Lastly, we assess the number of function evaluations (NFEs) the models take to sample. Figure 5b indicates that TABREP-FLOW can attain its best performance as early as 8 NFEs. At 1000 NFEs, we observe that both TABREP models are the best-performing.

4. Conclusion

In this work, we present TABREP, a simple and effective continuous representation for training tabular diffusion models. Motivated by geometric implications on the data manifold, our representation is dense, separable, and captures meaningful information for diffusion models. We conducted extensive experiments to evaluate TABREP. The results showcase TABREP's prowess in generating high-quality, privacy-preserving synthetic data while remaining computationally inexpensive.

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.

Bellman, R. *Dynamic Programming*. Princeton University Press, Princeton, NJ, 1957.

Bengio, Y., Courville, A., and Vincent, P. Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35 (8):1798–1828, 2013. doi: 10.1109/TPAMI.2013.50.

Bengio, Y., Courville, A., and Vincent, P. Representation learning: A review and new perspectives, 2014. URL https://arxiv.org/abs/1206.5538.

Bracewell, R. and Kahn, P. B. The fourier transform and its applications. *American Journal of Physics*, 34(8): 712–712, 1966.

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.

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

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

Ganev, G. and Cristofaro, E. D. The inadequacy of similarity-based privacy metrics: Privacy attacks against "truly anonymous" synthetic datasets, 2024. URL https://arxiv.org/abs/2312.05114.

Goodfellow, I., Bengio, Y., and Courville, A. *Deep Learning*. MIT Press, Cambridge, MA, 2016.

Guo, C. and Berkhahn, F. Entity embeddings of categorical variables, 2016. URL https://arxiv.org/abs/1604.06737.

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. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H. (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 6840–6851. Curran Associates, Inc., 2020a. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/4c5bcfec8584af0d967f1ab10179ca4b-Paper.pdf.

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

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

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 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.

Lautrup, A. D., Hyrup, T., Zimek, A., and Schneider-Kamp, P. Syntheval: a framework for detailed utility and privacy evaluation of tabular synthetic data. *Data Mining and Knowledge Discovery*, 39(1), December 2024. ISSN 1573-756X. doi: 10.1007/s10618-024-01081-4. URL http://dx.doi.org/10.1007/s10618-024-01081-4.

LeCun, Y., Bengio, Y., and Hinton, G. Deep learning. *nature*, 521(7553):436–444, 2015.

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, 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.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. Efficient estimation of word representations in vector space, 2013. URL https://arxiv.org/abs/1301.3781.

Mueller, M., Gruber, K., and Fok, D. Continuous diffusion for mixed-type tabular data, 2025. URL https://arxiv.org/abs/2312.10431.

Oppenheim, A. V. *Discrete-time signal processing*. Pearson Education India, 1999.

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.

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

Sahoo, S. S., Arriola, M., Schiff, Y., Gokaslan, A., Marroquin, E., Chiu, J. T., Rush, A., and Kuleshov, V. Simple and effective masked diffusion language models, 2024. URL https://arxiv.org/abs/2406.07524.

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.

Shi, J., Han, K., Wang, Z., Doucet, A., and Titsias, M. K. Simplified and generalized masked diffusion for discrete data, 2024a. URL https://arxiv.org/abs/2406.04329.

Shi, J., Xu, M., Hua, H., Zhang, H., Ermon, S., and Leskovec, J. Tabdiff: a multi-modal diffusion model for tabular data generation, 2024b. URL https://arxiv.org/abs/2410.20626.

Shokri, R., Stronati, M., Song, C., and Shmatikov, V. Membership inference attacks against machine learning models, 2017. URL https://arxiv.org/abs/1610.05820.

Song, Y. and Ermon, S. Generative modeling by estimating gradients of the data distribution, 2020. URL https://arxiv.org/abs/1907.05600.

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.

Ward, J., Wang, C.-H., and Cheng, G. Data plagiarism index: Characterizing the privacy risk of data-copying in tabular generative models, 2024. URL https://arxiv.org/abs/2406.13012.

Wikipedia contributors. Singularity (mathematics) — Wikipedia, The Free Encyclopedia, 2025. URL https://en.wikipedia.org/wiki/Singularity_(mathematics). [Online; accessed 7-April-2025].

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

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. Architecture

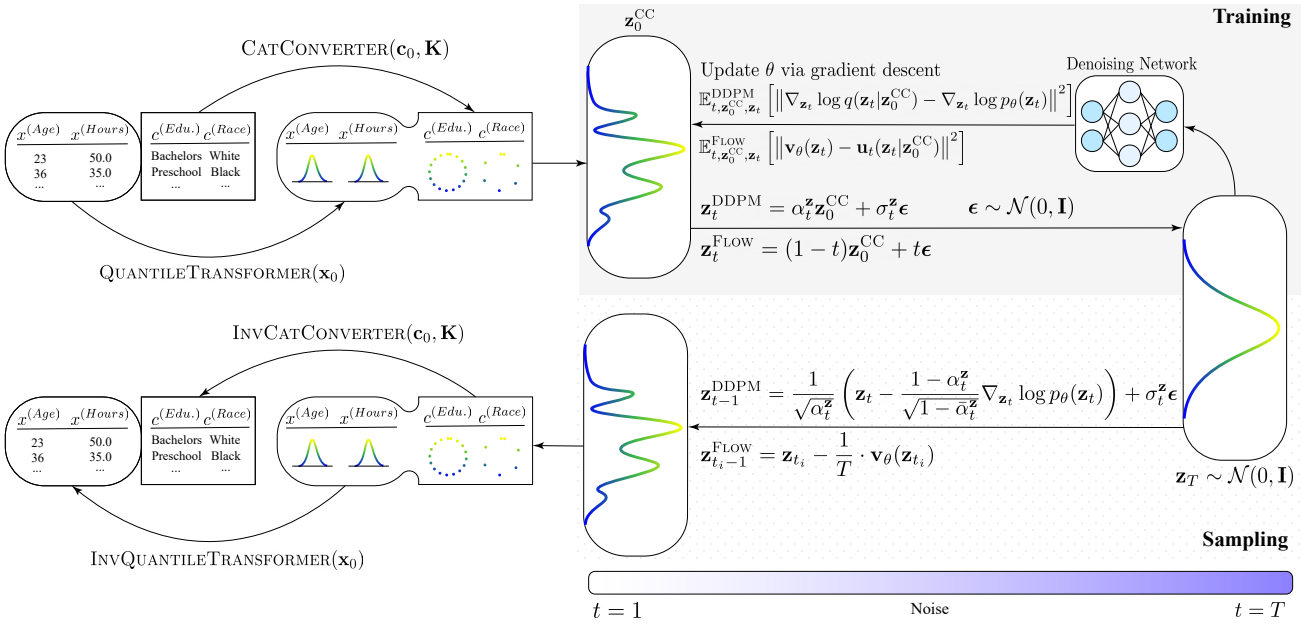


Figure 3. The TABREP Architecture. TABREP transforms and unifies the data space under a continuous regime via the our representation. A diffusion or flow matching process is trained to optimize the denoising network. Once training is completed, samples can be generated through a reverse denoising process before inverse transforming back into their original data representation.

B. Algorithm

Algorithm 1 Training TABREP-DDPM/FLOW

DDPM:

while not converged **do**

Sample $\mathbf{z}_0 = [\mathbf{x}_0, \mathbf{c}_0] \sim p(\mathbf{z})$

Encode $\mathbf{c}_0^{\text{CC}} \leftarrow \text{CatConverter}(\mathbf{c}_0, \mathbf{K})$

Encode $\mathbf{x}_0 \leftarrow \text{QUANTILETRANSFORMER}(\mathbf{x}_0)$

$\mathbf{z}_0^{\text{CC}} \leftarrow \text{CONCAT}(\mathbf{x}_0, \mathbf{c}_0^{\text{CC}})$

Sample $t \sim \text{Uniform}(\{1, \dots, T\})$

Sample noise $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I})$

Compute $\mathbf{z}_t = \alpha_t^{\mathbf{z}} \mathbf{z}_0^{\text{CC}} + \sigma_t^{\mathbf{z}} \boldsymbol{\epsilon}$

Compute $\mathcal{L}_{\text{TABREP-DDPM}} = \mathbb{E}_{t, \mathbf{z}_0^{\text{CC}}, \mathbf{z}_t} \left[\left\| \nabla_{\mathbf{z}_t} \log q(\mathbf{z}_t | \mathbf{z}_0^{\text{CC}}) - \nabla_{\mathbf{z}_t} \log p_\theta(\mathbf{z}_t) \right\|^2 \right]$

Update θ via gradient descent for $\nabla_\theta \mathcal{L}_{\text{TABREP-DDPM}}$

FLOW MATCHING:

Compute $\mathbf{z}_t = (1-t)\mathbf{z}_0^{\text{CC}} + t\boldsymbol{\epsilon}$

Define $\mathbf{u}_t(\mathbf{z}_t | \mathbf{z}_0^{\text{CC}}) = \boldsymbol{\epsilon} - \mathbf{z}_0^{\text{CC}}$

Compute $\mathcal{L}_{\text{TABREP-FLOW}} = \mathbb{E}_{t, \mathbf{z}_0^{\text{CC}}, \mathbf{z}_t} \left[\left\| \mathbf{v}_\theta(\mathbf{z}_t) - \mathbf{u}_t(\mathbf{z}_t | \mathbf{z}_0^{\text{CC}}) \right\|^2 \right]$

Update θ via gradient descent for $\nabla_\theta \mathcal{L}_{\text{TABREP-FLOW}}$

end while

Algorithm 2 Sampling TABREP-DDPM/FLOW

DDPM:

Sample $\mathbf{z}_T^{\text{CC}} \sim \mathcal{N}(0, \mathbf{I})$

for $t = T, \dots, 1$ **do**

Sample $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I})$ if $t > 1$, else $\boldsymbol{\epsilon} = 0$

$\mathbf{z}_{t-1} = \frac{1}{\sqrt{\alpha_t^{\mathbf{z}}}} \left(\mathbf{z}_t - \frac{1-\alpha_t^{\mathbf{z}}}{\sqrt{1-\bar{\alpha}_t^{\mathbf{z}}}} \nabla_{\mathbf{z}_t} \log p_\theta(\mathbf{z}_t) \right) + \sigma_t^{\mathbf{z}} \boldsymbol{\epsilon}$

end for

Output $\mathbf{z}_0^{\text{CC}} = [\mathbf{x}_0, \mathbf{c}_0^{\text{CC}}]$

Decode $\mathbf{c}_0 \leftarrow \text{InvCatConverter}(\mathbf{c}_0^{\text{CC}}, \mathbf{K})$

Decode $\mathbf{x}_0 \leftarrow \text{INVQUANTILETRANSFORMER}(\mathbf{x}_0)$

$\mathbf{z}_0 \leftarrow \text{CONCAT}(\mathbf{x}_0, \mathbf{c}_0)$

return $\mathbf{z}_0^{\text{CC}}$

FLOW MATCHING:

Discretize time $t_i = i/T$, for $i = T, T-1, \dots, 1$

$\mathbf{z}_{t_i - 1} = \mathbf{z}_{t_i} - \frac{1}{T} \cdot \mathbf{v}_\theta(\mathbf{z}_{t_i})$, via Euler ODE Solver

Figure 4. Training and sampling algorithms of TABREP-DDPM/FLOW.

C. Proofs

C.1. Variance of Learning Gradients in Diffusion Models

Proof. **Uniform Prior**. We show that the variance of the score function at a minimal n-singular point increases asymptotically with respect to n dimensions. Assume x is a noisy observation from a Gaussian centered at a weighted one-hot vector $\alpha_t e_k \in \mathbb{R}^K$. We can define the forward diffusion process as:

$$p_t(x|e_k) = \mathcal{N}(x|\alpha_t e_k, \sigma_t^2 I) \tag{10}$$

Hence, the score function of the conditional distribution $p_t(x|e_k)$ is given by:

$$\nabla_x \log p_t(x|e_k) = \nabla_x \log \left[\frac{1}{(2\pi\sigma_t^2)^{K/2}} \exp\left(-\frac{1}{2\sigma_t^2} ||x - \alpha_t e_k||^2 \right) \right] \tag{11}$$

$$= \nabla_x \left[-\frac{K}{2} \log(2\pi\sigma_t^2) - \frac{1}{2\sigma_t^2} ||x - \alpha_t e_k||^2 \right] \tag{12}$$

$$= -\frac{1}{\sigma_t^2}(x - \alpha_t e_k) \tag{13}$$

In which we define it as the gradient:

$$g_k(x) := \nabla_x \log p_t(x|e_k) = -\frac{1}{\sigma_t^2}(x - \alpha_t e_k) \tag{14}$$

Assume we have a uniform prior over categories:

$$p(e_k) = \frac{1}{K} \quad \forall k \in \mathcal{S} = \{1, \ldots, K\} \tag{15}$$

Per Bayes' Rule, we compute the posterior over e_k:

$$q(e_k|x) = \frac{p(x|e_k)p(e_k)}{\sum_{m=1}^K p(x|e_m)p(e_m)} \tag{16}$$

$$= \frac{p(x|e_k)}{\sum_{m=1}^K p(x|e_m)} \tag{17}$$

$$= \frac{p(x|e_k)}{\sum_{k\in S} p(x|e_k) + \sum_{m\notin S} p(x|e_m)} \tag{18}$$

Since the forward process is modeled as: $p_t(x|e_k) = \mathcal{N}(x|\alpha_t e_k, \sigma_t^2 I)$, we can infer that:

$$p(x|e_k) \propto \exp\left(-\frac{1}{2\sigma_t^2} ||x - \alpha_t e_k||^2 \right) \tag{19}$$

Then, $\forall k \in \mathcal{S}$, the likelihood terms are identical:

$$p_t(x|e_k) = \exp\left(-\frac{d_k^2}{2\sigma_t^2} \right) := A \tag{20}$$

where d is defined in Equation 2. $\forall m \notin S$, $d_m(x)^2 > d_k(x)^2$, thus:

$$p_t(x|e_m) = \exp\left(-\frac{d_m^2}{2\sigma_t^2} \right) \ll A \tag{21}$$

Therefore, the posterior simplifies to:

$$q(e_k|x) = \frac{p(x|e_k)}{\sum_{k\in S} p(x|e_k)} \tag{22}$$

$$\approx \frac{A}{nA} \tag{23}$$

$$= \frac{1}{n}, \quad \forall k \in \mathcal{S}. \tag{24}$$

We now compute the expected score:

$$\bar{g}(x) = \sum_{k=1}^{K} q(e_k|x) \cdot g_k(x) \tag{25}$$

$$\approx \sum_{k\in\mathcal{S}} \frac{1}{n} \cdot g_k(x) \tag{26}$$

$$= -\frac{1}{n\sigma_t^2} \sum_{k\in\mathcal{S}} (x - \alpha_t e_k) \tag{27}$$

$$= -\frac{1}{\sigma_t^2} \left(x - \alpha_t \bar{e}\right), \tag{28}$$

where $\bar{e} := \frac{1}{n}\sum_{k\in\mathcal{S}} e_k$ is the centroid of the vectors in $\mathcal{S}$. Next, compute the variance:

$$\mathrm{Var}(g|x) = \sum_{k\in\mathcal{S}} \frac{1}{n} \left\|g_k(x) - \bar{g}(x)\right\|^2 \tag{29}$$

$$= \sum_{k\in\mathcal{S}} \frac{1}{n} \left\| -\frac{1}{\sigma_t^2}(x - \alpha_t e_k) + \frac{1}{\sigma_t^2}(x - \alpha_t \bar{e}) \right\|^2 \tag{30}$$

$$= \sum_{k\in\mathcal{S}} \frac{1}{n} \cdot \frac{1}{\sigma_t^4} \left\|\alpha_t(\bar{e} - e_k)\right\|^2 \tag{31}$$

$$= \frac{\alpha_t^2}{\sigma_t^4} \sum_{k\in\mathcal{S}} \frac{1}{n} \left\|\bar{e} - e_k\right\|^2 \tag{32}$$

To complete the variance computation, we compute the squared distance between each one-hot vector and the centroid:

$$\left\|\bar{e} - e_k\right\|^2 = \sum_{i=1}^{K} (\bar{e}_i - e_k(i))^2 \tag{33}$$

$$= \left(\frac{n-1}{n}\right)^2 + (n-1) \cdot \left(\frac{1}{n}\right)^2 \tag{34}$$

$$= \frac{n-1}{n} \tag{35}$$

Note that for $i = k \in S$, $\bar{e}_k = \frac{1}{n}, e_k(k) = 1$ and for $i \neq k \in S$, $e_k(k) = 0$. Substituting into the variance:

$$\mathrm{Var}(g|x) = \frac{\alpha_t^2}{\sigma_t^4} \cdot \sum_{k\in\mathcal{S}} \frac{1}{n} \cdot \frac{n-1}{n} \tag{36}$$

$$= \frac{\alpha_t^2}{\sigma_t^4} \cdot \frac{n-1}{n} \tag{37}$$

We find that at a minimal n-singular point x, the posterior-weighted variance of the score function is strictly positive and increases asymptotically with n. In contrast, at a non-singular point, the posterior leans towards e_{k^*}:

$$q(e_{k^*}|x) \approx 1, \quad q(e_k|x) \approx 0 \quad \forall k \neq k^* \tag{38}$$

Then, the expected score becomes:

$$\bar{g}(x) = \sum_{k=1}^{K} q(e_k|x) \cdot g_k(x) \approx g_{k^*}(x) \tag{39}$$

And the variance reduces to:

$$\mathrm{Var}(g|x) = \sum_{k=1}^{K} q(e_k|x) \cdot \left\|g_k(x) - \bar{g}(x)\right\|^2 \tag{40}$$

$$\approx 1 \cdot \left\|g_{k^*}(x) - g_{k^*}(x)\right\|^2 + \sum_{k\neq k^*} 0 \cdot \left\|g_k(x) - g_{k^*}(x)\right\|^2 \tag{41}$$

$$= 0 \tag{42}$$

Categorical Prior. We now generalize the above result to an arbitrary categorical prior $\Pi = \{\pi_k\}_{k\in\mathcal{S}}$ over categories $\{e_k\}_{k\in\mathcal{S}}$, where $x_0 \sim \Pi$ and the forward diffusion process is defined as:

$$p_t(x|x_0) = \mathcal{N}(x|\alpha_t x_0, \sigma_t^2 I). \tag{43}$$

The marginal likelihood is then given by:

$$p_t(x) = \sum_{k\in\mathcal{S}} p_t(x|x_0 = e_k)\pi_k = \sum_{k\in\mathcal{S}} \mathcal{N}(x|\alpha_t e_k, \sigma_t^2 I)\pi_k. \tag{44}$$

Suppose we observe a noised sample $x \in \mathbb{R}^k$ at time t. Then the posterior probability that x was generated by adding noise to e_k is:

$$q_t(x_0 = e_k|x) = \frac{p_t(x|x_0 = e_k)\pi_k}{\sum_{m\in\mathcal{S}} p_t(x|x_0 = e_j)\pi_m}. \tag{45}$$

We compute the expected score at time t as follows:

$$\mathbb{E}_{p_t(x)}[\nabla_x \log p_t(x)] = \mathbb{E}_{x_0\sim\Pi}\left[\mathbb{E}_{p_t(x|x_0)}\left[\nabla_x \log p_t(x|x_0)\right]\right] \tag{46}$$

$$= \sum_{k\in\mathcal{S}} \pi_k \mathcal{N}(x|\alpha_t e_k, \sigma_t^2 I)\nabla_x \log \mathcal{N}(x|\alpha_t e_k, \sigma_t^2 I). \tag{47}$$

The score of the Gaussian is:

$$\nabla_x \log \mathcal{N}(x|\alpha_t e_k, \sigma_t^2 I) = -\frac{1}{\sigma_t^2}(x - \alpha_t e_k) = \frac{\alpha_t e_k - x}{\sigma_t^2}. \tag{48}$$

Therefore, the expected score becomes:

$$\bar{g} := \mathbb{E}_{p_t(x)}[\nabla_x \log p_t(x)] = \sum_{k\in\mathcal{S}} \pi_k \mathcal{N}(x|\alpha_t e_k, \sigma_t^2 I) \cdot \left(\frac{\alpha_t e_k - x}{\sigma_t^2}\right) \tag{49}$$

$$= C \sum_{k\in\mathcal{S}} \pi_k \exp\left(-\frac{\|x - \alpha_t e_k\|^2}{2\sigma_t^2}\right)\left(\frac{\alpha_t e_k - x}{\sigma_t^2}\right), \tag{50}$$

where $C := \frac{1}{(2\pi)^{k/2}\sigma_t^k}$.

Next, we compute the variance of the conditional score around its expectation:

$$\mathbb{E}_{x_0\sim\Pi}\left[\mathbb{E}_{p_t(x|x_0)}\left[\|\nabla_x \log p_t(x|x_0) - \bar{g}\|^2\right]\right] = \sum_{k\in\mathcal{S}} \pi_k \mathcal{N}(x|\alpha_t e_k, \sigma_t^2 I)\left\|\frac{\alpha_t e_k - x}{\sigma_t^2} - \bar{g}\right\|^2 \tag{51}$$

$$= C \sum_{k\in\mathcal{S}} \pi_k \exp\left(-\frac{\|x - \alpha_t e_k\|^2}{2\sigma_t^2}\right)\left\|\frac{\alpha_t e_k - x}{\sigma_t^2} - \bar{g}\right\|^2. \tag{52}$$

□

This formulation reveals that score variance is low when the posterior is sharply peaked (i.e., x is close to a single one-hot vector), and increases when the posterior mass is spread over multiple categories. The minimal n-singular point case is recovered when $\pi_k = \frac{1}{n}$ for $k \in \mathcal{S}$ and $x = \frac{1}{n}\sum_{k\in\mathcal{S}} e_k$, confirming the consistency of both analyses.

D. Implementation

The following delineates the foundation of our experiments:

- Codebase: Python & PyTorch
- CPU: AMD EPYC-Rome 7002
- GPU: NVIDIA A100 80GB PCIe

D.1. Related Works

The latest tabular diffusion models have made considerable progress compared to previous generative models such as VAEs (Xu et al., 2019) and GANs (Xu et al., 2019). This included STaSy (Kim et al., 2022), which employed a score-matching diffusion model paired with self-paced learning and fine-tuning to stabilize the training process, and CoDi (Lee et al., 2023), which used separate diffusion schemes for categorical and numerical data along with interconditioning and contrastive learning to improve synergy among 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. TabSYN (Zhang et al., 2023) is a latent diffusion model that transformed features into a unified embedding via a β-VAE (Kingma & Welling, 2013) before applying EDM diffusion (Karras et al., 2022) to generate synthetic data. CDTD (Mueller et al., 2025) combines score matching and score interpolation to enforce a unified continuous noise distribution for both continuous and categorical features but different benchmarks are used to perform evaluation. TabDiff, couples DDPM (Ho et al., 2020b) and Discrete Masked Diffusion (Shi et al., 2024a; Sahoo et al., 2024) to synthesize tabular data.

D.2. Categorical Representations

One-Hot Encoding. The one-hot encoding (ONEHOT) representation of $\mathbf{c}^{(i)}$ is constructed by concatenating the one-hot encoded vectors of each individual feature $c_j^{(i)}$. Specifically, the one-hot encoding for $c_j^{(i)}$ is a vector $\mathbf{e}(c_j^{(i)}) \in \{0,1\}^{K_j}$, where the k-th entry is defined as:

$$\mathbf{e}(c_j^{(i)})_k = \begin{cases} 1 & \text{if } k = c_j^{(i)}, \\ 0 & \text{otherwise,} \end{cases} \tag{53}$$

for $k \in \{1, 2, \ldots, K_j\}$. The one-hot encoded representation of $\mathbf{c}^{(i)}$ is then:

$$\text{ONEHOT}(\mathbf{c}^{(i)}) = [\mathbf{e}(c_1^{(i)}), \mathbf{e}(c_2^{(i)}), \ldots, \mathbf{e}(c_{D_{\text{cat}}}^{(i)})]. \tag{54}$$

The resulting vector has a total length of $\sum_{j=1}^{D_{\text{cat}}} K_j$, which corresponds to the sum of the unique categories across all categorical features. A softmax or a logarithm is then applied to the one-hot representation to yield a continuous probability distribution.

Learned Embeddings. Learned Embeddings (LEARNED) (Mikolov et al., 2013) encode the categorical component $\mathbf{c}^{(i)}$ using a representation trained directly within the model. Specifically, each categorical feature $c_j^{(i)}$ (the j-th element of $\mathbf{c}^{(i)}$) is assigned a trainable embedding vector of fixed dimensionality d_{EMB}. Hence, for each $j \in \{1, \ldots, D_{\text{cat}}\}$, we have an embedding matrix

$$E_j \in \mathbb{R}^{K_j \times d_{\text{EMB}}}.$$

The embedding lookup operation retrieves the embedding vector for each $c_j^{(i)}$, and these vectors are then concatenated to form the full embedding for the categorical features:

$$\text{LEARNED}(\mathbf{c}^{(i)}) = \text{concat}\big(E_1[c_1^{(i)}],\ E_2[c_2^{(i)}],\ \ldots,\ E_{D_{\text{cat}}}[c_{D_{\text{cat}}}^{(i)}]\big) \in \mathbb{R}^{D_{\text{cat}} \cdot d_{\text{EMB}}}. \tag{55}$$

To decode the learned embeddings back into categorical values, a nearest-neighbor approach is applied. For each embedding segment corresponding to a categorical feature, the pairwise distance between the embedding and the learned weights is computed, and the category with the minimum distance is selected:

$$\hat{c}_j^{(i)} = \underset{k \in \{1,\ldots,K_j\}}{\arg\min} \left\| \tilde{\mathbf{c}}_j^{(i)} - E_j[k] \right\|, \tag{56}$$

where $\tilde{\mathbf{c}}_j^{(i)} \in \mathbb{R}^{d_{\text{EMB}}}$ is the segment of the concatenated embedding corresponding to the j-th categorical feature, and $E_j \in \mathbb{R}^{K_j \times d_{\text{EMB}}}$ is the embedding matrix for that feature.

Analog Bits. Analog Bits (I2B) (Chen et al., 2022) encode categorical features using a binary-based continuous representation. The encoding process involves two steps. We convert the categorical value to a real-valued binary representation where each category can be expressed using $\lceil \log_2(K) \rceil$ binary bits based on the number of categories:

$$\text{I2B}(c_j^{(i)}, K_j^{(i)}) \in \mathbb{R}^{\lceil \log_2 K_j^{(i)} \rceil} \tag{57}$$

followed by a shift and scale formula:

$$\text{I2B}(c_j^{(i)}, K_j^{(i)}) \leftarrow (\text{I2B}(c_j^{(i)}, K_j^{(i)}) \cdot 2 - 1). \tag{58}$$

Thus, training and sampling of continuous-feature generative models (e.g., diffusion models) become computationally tractable. To decode, 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. Dictionary encoding (DIC) represents categorical features using a continuous look-up embedding function. The encoding assigns equally spaced real-valued representations within a tunable specified range, $[-1, 1]$, to balance sparsity and separability. For categorical features with more categories, a wider range may be necessary to ensure proper distinction between values. The encoding process is defined as:

$$\text{DIC}(c_j^{(i)}, K_j^{(i)}) = -1 + \frac{2c_j^{(i)}}{K_j^{(i)} - 1}, \quad \text{DIC}(c_j^{(i)}, K_j^{(i)}) \in [-1, 1] \tag{59}$$

To decode, the nearest embedding is determined by comparing the encoded continuous value with all $K_j^{(i)}$ possible embeddings, selecting the category with the smallest Euclidean distance.

D.3. Implementation Specifics

Ordering of Categorical Feature. Our CATCONVERTER representation induces an order on the categorical features. We assess how the order influences the results. By default, we assign a lexicographic ordering to the categorical features for simplicity purposes. We conducted an experiment to display the performance of lexicographic ordering vs. random ordering on the Adult and Beijing datasets. These datasets are chosen since they contain variables with a natural ordering such as "Education" (ordinal) and "Day" (periodic). As observed by the AUC disparity in Table 3, the order is an important factor in our CATCONVERTER representation. It also implicitly highlights that CATCONVERTER's geometry aids in preserving the inherent semantics of ordinal and cyclical categorical features.

Table 3. AUC (classification) and RMSE (regression) scores of Machine Learning Efficiency. Higher scores indicate better performance.

	AUC ↑	RMSE ↓
METHODS	ADULT	BEIJING
TABREP-DDPM (RANDOM)	0.776±.002	1.050±.006
TABREP-DDPM (LEXICOGRAPHIC)	0.913±.002	0.508±.006
TABREP-FLOW (RANDOM)	0.807±.003	1.041±.005
TABREP-FLOW (LEXICOGRAPHIC)	0.912±.002	0.536±.006

Out-of-index (OOI). Out-of-index (OOI) can potentially occur due to the generative nature of DDPM and FM. For CATCONVERTER, OOI values are cast to the value of the 0-th index. Although casting ensures that all generated categorical values fall within the valid range, it introduces a bias. We conducted additional experiments highlighting the casting rate that occurs when using our method. As observed in Table 4, casting rate is relatively low for most datasets, ranging between 5% to 20% apart from the Stroke dataset at around 30%. Nonetheless, we still achieve exceptional results across all datasets and benchmarks validating the effectiveness of our method.

Table 4. Out-of-Index Casting Rate

METHODS	ADULT	DEFAULT	SHOPPERS	STROKE	DIABETES	BEIJING	NEWS
TABREP-DDPM	$0.100_{\pm.001}$	$0.094_{\pm.001}$	$0.112_{\pm.005}$	$0.272_{\pm.003}$	$0.183_{\pm.002}$	$0.072_{\pm.002}$	$0.092_{\pm.003}$
TABREP-FLOW	$0.108_{\pm.001}$	$0.069_{\pm.001}$	$0.138_{\pm.001}$	$0.348_{\pm.002}$	$0.149_{\pm.001}$	$0.054_{\pm.001}$	$0.093_{\pm.001}$

Flow Matching/DDPM Denoising Network. The input layer projects the batch of tabular data input samples $\mathbf{z}_t$, 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.

$$h_{in} = \texttt{FC}_{d_t}(\mathbf{z}_t) + t_{emb} \tag{60}$$

Subsequently, the output is passed through hidden layers h_1, h_2, h_3, and h_4 which are fully connected networks used to learn the denoising direction or 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 5.

$$h_1 = \texttt{Dropout}(\texttt{Activation}(\texttt{FC}(h_{in}))) \tag{61}$$

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 score function $\nabla_{\mathbf{z}_t} \log p_\theta(\mathbf{z}_t)$.

$$\nabla_{\mathbf{z}_t} \log p_\theta(\mathbf{z}_t) = \texttt{FC}_{d_{in}}(h_{out}) \tag{62}$$

D.4. Hyperparameters

The hyperparameters selected for our model are shown in Table 5. The remaining hyperparameters of the baselines are tuned accordingly.

Table 5. TABREP Hyperparameters.

General		**Flow Matching/DDPM Denoising MLP**	
Hyperparameter	**Value**	**Hyperparameter**	**Value**
Training Iterations	$100,000$	Timestep embedding dimension d_t	1024
Flow Matching/DDPM Sampling Steps	$50/1000$	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		
Optimizer	Adam		

E. Experiments

E.1. Overview

Implementation Information. Experimental results are obtained over an average of 20 sampling seeds using the best-validated model. Continuous features are encoded using a QuantileTransformer (Pedregosa et al., 2011). The order in which the categories of a feature are assigned in our encoding schemes is based on lexicographic ordering for simplicity. Further details are in Appendix D.

Datasets. We select seven datasets from the UCI Machine Learning Repository to conduct our experiments. This includes Adult, Default, Shoppers, Stroke, Diabetes, Beijing, and News which contain a mix of continuous and discrete features. Further details are in Appendix E.2.

Baselines. We compare our model against existing diffusion baselines for tabular generation since they are the best-performing. This includes STASY (Kim et al., 2022), CODI (Lee et al., 2023), TABDDPM (Kotelnikov et al., 2023), TABSYN (Zhang et al., 2023), and TABDIFF (Shi et al., 2024b). Further details are in Appendix E.3.

Benchmarks. We observe that downstream task performance measured by machine learning efficiency (MLE) typically translates to most other benchmarks. Thus, the primary quality benchmark in our main paper will be MLE for brevity, and a privacy benchmark, membership inference attacks (MIA). The remaining fidelity benchmarks include: column-wise density (CWD), pairwise-column correlation (PCC), α-precision, β-recall, and classifier-two-sample test (C2ST). Further details regarding benchmark information and additional results are in Appendix E.4 and G.

Ablation: Unified and Separate Data Representations. We analyze the impact of training tabular diffusion models between our unified data representation and a separate data representation. We compare TABREP-DDPM and TABREP-FLOW to TABDDPM (Kotelnikov et al., 2023) and TABFLOW. Note that we introduce TABFLOW, by modeling continuous features using Flow Matching (Lipman et al., 2022; Liu et al., 2022) and categorical features using Discrete Flow Matching (Campbell et al., 2024) under the same separate continuous-discrete data representation as TABDDPM.

Ablation: Categorical Representations. We conduct ablation studies with respect to categorical data representations used in existing diffusion baselines to assess the performance of diffusion models under a unified data representation. This includes one-hot encoding used in (Kim et al., 2022; Lee et al., 2023; Shi et al., 2024b), learned one-dimensional and two-dimensional embeddings (Mikolov et al., 2013; Guo & Berkhahn, 2016), and Analog Bits (I2B) (Chen et al., 2022) from the discrete image diffusion domain. Additionally, we introduce an intuitive static one-dimensional embedding, Dictionary (DIC).

E.2. Datasets

Experiments were conducted with a total of 7 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, Shoppers, Stroke, and Diabetes 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, and the Diabetes dataset, which was split into a ratio of 6:2:2. The resulting statistics of each dataset are shown in Table 6.

Table 6. Dataset Statistics. "# Num" and "# Cat" refer to the number of numerical and categorical columns.

Dataset	# Samples	# Num	# Cat	# Max Cat	# Train	# Validation	# Test	Task Type
Adult	48,842	6	9	42	28,943	3,618	16,281	Binary Classification
Default	30,000	14	11	11	24,000	3,000	3,000	Binary Classification
Shoppers	12,330	10	8	20	9,864	1,233	1,233	Binary Classification
Beijing	41,757	7	5	31	33,405	4,175	4,175	Regression
News	39,644	46	2	7	31,714	3,965	3,965	Regression
Stroke	4,909	3	8	5	3,927	490	490	Binary Classification
Diabetes	99,473	8	21	10	59,683	19,895	19,895	Multiclass Classification

E.3. Baselines

TabRep's performance is evaluated in comparison to previous works in diffusion-based mixed-type tabular data generation. This includes STaSy (Kim et al., 2022), CoDi (Lee et al., 2023), TabDDPM (Kotelnikov et al., 2023), TabSYN (Zhang et al., 2023), and TabDiff (Shi et al., 2024b). The underlying architectures and implementation details of these models are presented in Table 7. Note that different benchmarks were used for CDTD (Mueller et al., 2025) thus, we decided to not include it in our baselines.

Table 7. Comparison of tabular data synthesis baselines.

Method	Model[1]	Type[2]	Categorical Encoding	Numerical Encoding	Additional Techniques
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	VAE-Learned	Quantile Transformer	Feature Tokenizer and Transformer encoder to learn cross-feature relationships with adaptive loss weighing to increase reconstruction performance.
TabDiff	EDM/Masked Diffusion	S	One-Hot Encoding	Quantile Transformer	Joint continuous-time diffusion process of numerical and categorical variables under learnable noise schedules, with a stochastic sampler to correct sampling errors.
TABREP-DDPM	DDPM	U	CAT-CONVERTER	Quantile Transformer	Plug-and-play for Diffusion Models.
TABREP-Flow	Flow Matching	U	CAT-CONVERTER	Quantile Transformer	Plug-and-play for Diffusion Models.

[1] The "Model" Column indicates the underlying architecture used for the model. Options include Denoising Diffusion Probabilistic Models or DDPMs (Ho et al., 2020b), 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.

E.4. Benchmarks

We evaluate the generative performance on a broad suite of benchmarks. We analyze the capabilities in *downstream tasks* such as machine learning efficiency (MLE), where we determine the AUC score for classification tasks and RMSE for regression tasks of 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 three additional benchmarks including a detection test metric, Classifier Two Sample Tests (C2ST) (SDMetrics, 2024), and privacy preservation metrics: the precision and recall of a membership inference attack (MIA) (Shokri et al., 2017). 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.

Machine Learning Efficiency. To evaluate the quality of our generated synthetic data, we use the data to train a classification/regression model, using XGBoost (Chen & Guestrin, 2016). This model is applied to the real test set. *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.

$$\text{AUC} = \int_0^1 \text{TPR}(\text{FPR})\, d(\text{FPR}) \tag{63}$$

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.

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{64}$$

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:

$$\text{KST} = \sup_x |F_1(x) - F_2(x)|, \tag{65}$$

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

$$\text{F}_\text{n}(\text{x}) = \frac{1}{n}\sum_{i=1}^{n} \mathbf{1}_{(-\infty,x]}(X_i) \tag{66}$$

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

$$\text{TVD} = \frac{1}{2}\sum_{x\in X} |P_1(x) - P_2(x)|, \tag{67}$$

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:

$$\rho(x, y) = \frac{\text{cov}(x, y)}{\sigma_x \sigma_y}, \tag{68}$$

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

$$\text{Pearson Score} = \frac{1}{2}\mathbb{E}_{x,y}|\rho^1(x,y) - \rho^2(x,y)| \tag{69}$$

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 Contingency Score.

$$\text{Contingency Score} = \frac{1}{2}\sum_{\alpha\in A}\sum_{\beta\in B}|P_{1,(\alpha,\beta)} - P_{2,(\alpha,\beta)}|, \tag{70}$$

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. 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 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, indicating that the machine learning model cannot concretely identify whether the data point is real or generated. We select logistic regression as our machine learning model, using the implementation provided by SDMetric (SDMetrics, 2024).

Privacy Metric: Membership Inference Attacks (MIA). Membership inference attacks evaluate the vulnerability of machine learning models to privacy leakage by determining whether a given instance was included in the training dataset (Shokri et al., 2017). The attacker often constructs a shadow model to mimic the target model's behavior and trains a binary classifier to distinguish membership status based on observed patterns. We replaced DCR with Membership Inference Attacks since existing privacy ML literature (Ganev & Cristofaro, 2024; Ward et al., 2024) conducted research highlighting the "Inadequacy of Similarity-based Privacy Metrics" such as DCR.

These attacks are evaluated using precision, the fraction of inferred members that are true members, and recall, the fraction of true members correctly identified. When records are equally balanced between members and non-members, the ideal precision and recall are 0.5, indicating that the attack is no better than random guessing. Higher values suggest privacy leakage and reveal vulnerabilities in the model. Implementation of this metric is provided by SynthEval (Lautrup et al., 2024).

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

F. Experimental Results

Table 9. Ablation study on TABREP's unified representation versus a separate representation.

	AUC ↑				F1 ↑	RMSE ↓	
METHODS	ADULT	DEFAULT	SHOPPERS	STROKE	DIABETES	BEIJING	NEWS
TABDDPM	0.910±.001	0.761±.004	0.915±.004	0.808±.033	**0.376±.003**	0.592±.012	3.46±1.25
TABREP-DDPM	**0.913±.002**	**0.764±.005**	**0.926±.005**	**0.869±.027**	0.373±.003	**0.508±.006**	**0.836±.001**
TABFLOW	0.908±.002	0.742±.008	0.914±.005	0.821±.082	**0.377±.002**	0.574±.010	0.850±.017
TABREP-FLOW	**0.912±.002**	**0.782±.005**	**0.919±.005**	**0.830±.028**	**0.377±.002**	**0.536±.006**	**0.814±.002**

Table 10. Ablation study on categorical representations under a unified continuous data space.

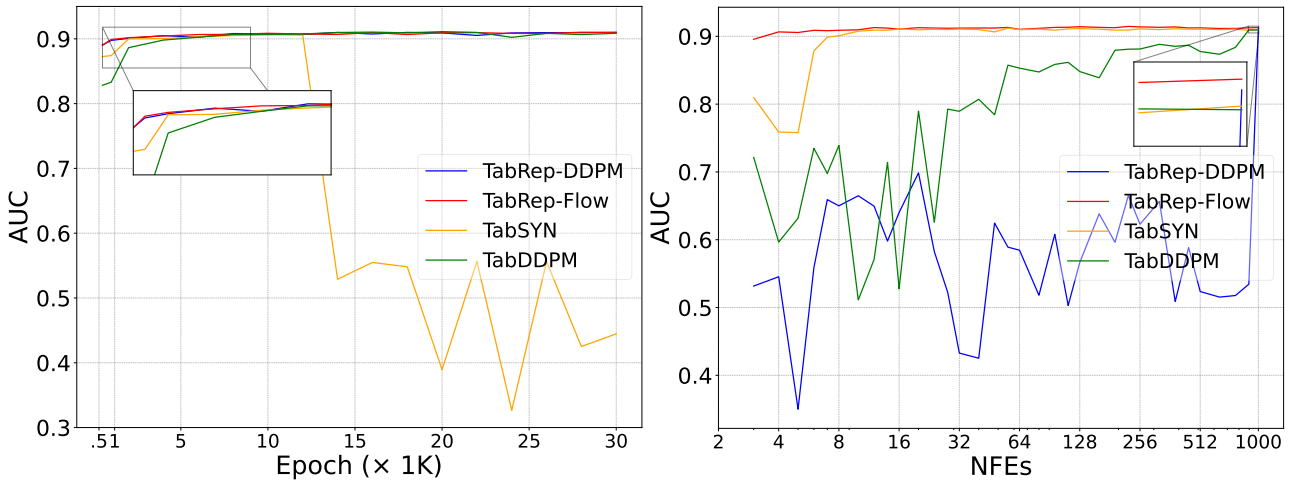
	AUC ↑				F1 ↑	RMSE ↓	
METHODS	ADULT	DEFAULT	SHOPPERS	STROKE	DIABETES	BEIJING	NEWS
ONEHOT-DDPM	0.476±.057	0.557±.052	0.799±.126	0.797±.133	0.363±.008	2.143±.339	0.840±.020
LEARNED1D-DDPM	0.611±.008	0.575±.012	0.876±.028	0.743±.032	0.179±.010	0.921±.006	0.858±.010
LEARNED2D-DDPM	0.793±.003	0.290±.009	0.103±.011	0.850±.035	0.205±.007	0.969±.005	0.857±.023
I2B-DDPM	0.911±.001	0.762±.003	0.919±.004	0.852±.029	0.370±.008	0.542±.008	0.844±.013
DIC-DDPM	0.912±.002	0.763±.003	0.910±.004	0.824±.027	**0.375±.006**	0.547±.008	0.851±.013
TABREP-DDPM	**0.913±.002**	**0.764±.005**	**0.926±.005**	**0.869±.027**	0.373±.003	**0.508±.006**	**0.836±.001**
ONEHOT-FLOW	0.895±.003	0.759±.005	0.910±.006	0.812±.129	0.372±.005	0.765±.016	0.850±.017
LEARNED1D-FLOW	0.260±.014	0.438±.009	0.134±.015	0.142±.033	0.184±.007	0.806±.009	0.873±.007
LEARNED2D-FLOW	0.126±.012	0.709±.008	0.868±.007	0.180±.030	0.177±.008	0.787±.007	0.866±.005
I2B-FLOW	0.911±.001	0.763±.004	0.910±.005	0.797±.027	0.372±.003	0.543±.007	0.847±.014
DIC-FLOW	0.912±.002	0.763±.004	0.903±.005	0.807±.028	0.376±.007	0.561±.007	0.853±.014
TABREP-FLOW	**0.912±.002**	**0.782±.005**	**0.919±.005**	**0.830±.028**	**0.377±.002**	**0.536±.006**	**0.814±.002**

Table 11. Recall Scores of MIAs. A score closer to 50% is better for privacy-preservation.

METHODS	ADULT	DEFAULT	SHOPPERS	STROKE	DIABETES	BEIJING	NEWS
STASY	24.51±0.44	30.37±0.99	17.54±0.19	34.63±1.39	29.75±0.16	34.06±0.33	23.49±0.67
CODI	0.05±0.01	3.41±0.53	1.36±0.45	27.32±3.50	0.00±0.00	0.40±0.09	0.04±0.02
TABDDPM	56.90±0.29	44.96±0.59	46.08±1.57	55.12±0.81	52.06±0.11	48.35±0.79	9.88±0.52
TABSYN	42.91±0.31	43.71±0.89	42.14±0.76	47.97±1.27	44.42±0.28	46.53±0.52	34.42±0.90
TABDIFF	52.00±0.35	46.67±0.55	46.86±1.20	47.15±1.30	31.01±0.22	48.20±0.65	16.03±0.66
TABREP-DDPM	52.78±0.21	**48.96±0.41**	48.16±0.90	**50.89±0.93**	**51.43±0.13**	**49.50±0.52**	**40.10±0.55**
TABREP-FLOW	**50.51±0.21**	47.07±0.40	**49.19±0.86**	49.43±1.16	50.33±0.13	49.14±0.81	35.48±0.78

Table 12. Training and Sampling Duration.

METHODS	TRAINING (S)	SAMPLING (S)	TOTAL (S)
STASY	6608.84	12.93	6621.77
CODI	24039.96	9.41	24049.37
TABDDPM	3112.07	66.82	3178.89
TABSYN	2373.98 + 1084.82	10.54	3469.30
TABDIFF	5640	15.2	5655.2
TABREP-DDPM	2070.59	59.00	2130.59
TABREP-FLOW	2028.33	3.07	**2031.40**



(a) AUC vs. Train Epochs

(b) AUC vs. Sample NFEs

Figure 5. Training and Sampling Efficacy.

G. Further Experimental Results

We perform experiments using several diffusion-based tabular generative model baselines, including STaSy (Kim et al., 2022), CoDi (Lee et al., 2023), TabDDPM (Kotelnikov et al., 2023), TabSYN (Zhang et al., 2023). We include TabDiff's reported results as the authors have not released code (Shi et al., 2024b).

We also incorporate several ablation experiments. We introduce TabFlow, an adaptation of TabDDPM (Kotelnikov et al., 2023) that models numerical and categorical tabular data with continuous flow matching (Lipman et al., 2022; Liu et al., 2022) and discrete flow matching (Tong et al., 2023), to examine the effect of unifying the data space. Experiments are also performed on various categorical representations on both diffusion and flow models. This includes one-hot, 1D-Learned Embedding, 2D-Learned Embedding, and Analog Bits (Chen et al., 2022), to demonstrate TABREP's effectiveness. We show that our proposed TABREP framework achieves superior performance on the vast majority of metrics in Appendix G.1.

We evaluate AUC (classification), RMSE (regression), Column-Wise Density Estimation (CDE), Pair-Wise Column Correlation (PCC), α-Precision, β-Recall scores, Classifier-Two Sample Test scores (C2ST), and Membership Inference Attacks Precision (MIA P.) and Recall (MIA R.) scores for our 7 datasets. $\uparrow$ indicates that the higher the score, the better the performance; $\downarrow$ indicates that the lower the score, the better the performance; $\updownarrow$ indicates that an optimal score should be as close to 50% as possible.

The metrics and error bars shown in the tables in this section are derived from the mean and standard deviation of the experiments performed on 20 sampling iterations on the best-validated model.

G.1. Additional Baseline Results

	ADULT							
METHODS	AUC ↑	CDE ↑	PCC ↑	α ↑	β ↑	C2ST ↑	MIA P. ↕	MIA R. ↕
STASY	0.906±.001	92.26±0.04	89.15±0.10	77.05±0.29	33.54±0.36	55.37	49.52±0.50	24.51±0.44
CODI	0.871±0.006	74.28±0.08	77.38±0.21	74.45±0.35	8.74±0.16	15.80	55.00±7.26	0.05±0.01
TABDDPM	0.910±0.001	98.37±0.08	96.69±0.09	90.99±0.35	**62.19±0.48**	97.55	51.30±0.07	56.90±0.29
TABSYN	0.906±0.001	98.89±0.03	97.56±0.06	98.97±0.26	47.68±0.27	95.49	50.91±0.16	42.91±0.31
TABDIFF	0.912±0.002	99.37±0.05	98.51±0.16	99.02±0.20	51.64±0.20	**99.50**	51.03±0.12	52.00±0.35
TABREP-DDPM	**0.913±0.002**	**99.39±0.04**	**98.63±0.04**	**99.11±0.25**	52.04±0.12	**99.50**	**50.44±0.83**	52.78±0.21
TABREP-FLOW	0.912±0.002	98.63±0.02	97.55±0.23	98.21±0.34	49.91±0.28	95.48	50.65±0.20	**50.51±0.21**

	DEFAULT							
METHODS	AUC ↑	CDE ↑	PCC ↑	α ↑	β ↑	C2ST ↑	MIA P. ↕	MIA R. ↕
STASY	0.752±0.006	89.41±0.03	92.64±0.03	94.83±0.15	40.23±0.22	62.82	51.84±0.69	30.37±0.99
CODI	0.525±0.006	81.07±0.08	86.25±0.73	81.21±0.11	19.75±0.30	42.28	46.66±2.54	3.41±0.53
TABDDPM	0.761±0.004	98.20±0.05	97.16±0.19	96.78±0.30	**53.73±0.28**	97.12	51.34±0.49	44.96±0.59
TABSYN	0.755±0.005	98.61±0.08	**98.33±0.67**	98.49±0.20	46.06±0.37	95.83	50.80±0.56	43.71±0.89
TABDIFF	0.763±0.005	98.76±0.07	97.45±0.75	98.49±0.28	51.09±0.25	97.74	51.15±0.62	46.67±0.55
TABREP-DDPM	0.764±0.005	**98.97±0.19**	96.74±0.62	**98.66±0.24**	48.22±0.48	**98.90**	50.07±0.41	**48.96±0.41**
TABREP-FLOW	**0.782±0.005**	97.45±0.06	92.86±1.75	96.50±0.44	49.99±0.23	89.36	**50.04±0.95**	47.07±0.40

	SHOPPERS							
METHODS	AUC ↑	CDE ↑	PCC ↑	α ↑	β ↑	C2ST ↑	MIA P. ↕	MIA R. ↕
STASY	0.914±0.005	82.53±0.19	81.40±0.27	68.18±0.29	26.24±0.60	25.82	51.23±2.38	17.54±0.19
CODI	0.865±0.006	67.27±0.03	80.52±0.12	90.52±0.37	19.22±0.27	19.04	59.67±11.74	1.36±0.45
TABDDPM	0.915±0.004	97.58±0.18	96.72±0.22	90.85±0.60	72.46±0.46	83.49	51.24±1.48	46.08±1.57
TABSYN	0.918±0.004	96.00±0.12	95.18±0.11	96.28±0.24	45.79±0.31	83.77	51.00±1.08	42.14±0.76
TABDIFF	0.919±0.005	98.72±0.09	**98.26±0.08**	**99.11±0.34**	49.75±0.64	**98.43**	51.11±1.11	46.86±1.20
TABREP-DDPM	**0.926±0.005**	**98.97±0.10**	97.62±0.02	96.14±0.19	53.68±0.73	96.37	**49.86±0.98**	48.16±0.90
TABREP-FLOW	0.919±0.005	97.74±0.03	97.08±0.07	95.85±0.46	**55.92±0.37**	94.20	51.38±1.66	**49.19±0.86**

	STROKE							
METHODS	AUC ↑	CDE ↑	PCC ↑	α ↑	β ↑	C2ST ↑	MIA P. ↕	MIA R. ↕
STASY	0.833±0.03	89.36±0.13	84.99±0.09	91.49±0.33	39.92±0.76	40.64	54.22±1.14	34.63±1.39
CODI	0.798±0.032	87.42±0.17	80.65±1.81	86.46±0.53	28.59±0.47	24.47	54.54±1.60	27.32±3.50
TABDDPM	0.808±0.033	99.10±0.05	97.09±1.17	98.05±0.14	**71.42±0.30**	**100.00**	53.45±2.89	55.12±0.81
TABSYN	0.845±0.035	96.79±0.08	95.18±0.22	95.49±0.41	48.85±0.26	89.93	51.11±1.54	47.97±1.27
TABDIFF	0.848±0.021	99.09±0.12	**97.91±0.22**	**98.95±0.53**	49.91±0.86	99.87	52.20±1.61	47.15±1.30
TABREP-DDPM	**0.869±0.027**	**99.14±0.20**	97.11±0.60	98.32±0.82	57.17±0.77	**100.00**	51.74±1.85	**50.89±0.93**
TABREP-FLOW	0.854±0.028	98.42±0.31	97.37±2.12	96.40±0.71	63.91±0.87	95.96	**50.66±1.77**	49.43±1.16

	DIABETES							
METHODS	F1 ↑	CDE ↑	PCC ↑	α ↑	β ↑	C2ST ↑	MIA P. ↕	MIA R. ↕
STASY	0.374±0.003	95.25±0.03	93.41±0.08	90.00±0.17	39.62±0.23	54.71	49.47±0.22	29.75±0.16
CODI	0.288±0.009	76.42±0.02	78.07±0.18	38.96±0.16	6.39±0.16	3.95	0.00±0.00	0.00±0.00
TABDDPM	0.376±0.003	99.26±0.01	98.71±0.01	95.36±0.32	**52.65±0.03**	92.18	50.40±0.65	52.06±0.11
TABSYN	0.361±0.001	99.04±0.01	98.32±0.03	98.08±0.14	45.08±0.04	**93.38**	50.10±0.34	44.42±0.28
TABDIFF	0.353±0.006	98.72±0.02	97.80±0.03	96.84±0.14	36.96±0.18	91.60	49.27±0.41	31.01±0.22
TABREP-DDPM	0.373±0.003	**99.36±0.02**	**98.75±0.03**	97.19±0.22	45.75±0.49	92.65	**50.07±0.37**	**51.43±0.13**
TABREP-FLOW	**0.377±0.002**	99.00±0.02	98.46±0.05	**99.08±0.13**	48.58±0.17	90.41	50.24±0.37	50.33±0.13

	BEIJING							
METHODS	RMSE ↓	CDE ↑	PCC ↑	α ↑	β ↑	C2ST ↑	MIA P. ↕	MIA R. ↕
STASY	0.656±0.014	93.14±0.07	90.63±0.11	96.41±0.10	51.35±0.16	77.80	48.98±0.78	34.06±0.33
CODI	0.818±0.021	83.54±0.04	90.35±0.21	96.89±0.14	53.16±0.12	80.27	39.19±6.35	0.40±0.09
TABDDPM	0.592±0.012	99.09±0.02	96.71±0.18	97.74±0.06	**73.13±0.26**	95.13	50.43±0.58	48.35±0.79
TABSYN	0.555±0.013	98.34±0.01	96.85±0.24	98.08±0.27	55.68±0.16	92.92	51.31±0.42	46.53±0.52
TABDIFF	0.555±0.013	98.97±0.05	**97.41±0.15**	98.06±0.24	59.63±0.23	97.81	50.39±0.46	48.20±0.65
TABREP-DDPM	**0.508±0.006**	**99.11±0.03**	96.97±0.20	**98.98±0.16**	64.08±0.18	**98.16**	51.02±0.42	**49.50±0.52**
TABREP-FLOW	0.536±0.006	98.28±0.07	96.92±0.21	98.16±0.13	62.65±0.12	92.26	**50.35±0.60**	49.14±0.81
	NEWS							
METHODS	RMSE ↓	CDE ↑	PCC ↑	α ↑	β ↑	C2ST ↑	MIA P. ↕	MIA R. ↕
STASY	0.871±0.002	90.50±0.10	96.59±0.02	**97.95±0.14**	38.68±0.30	51.72	49.68±0.70	23.49±0.67
CODI	1.21±0.005	70.82±0.01	95.44±0.05	86.03±0.12	35.01±0.13	9.35	40.00±24.49	0.04±0.02
TABDDPM	3.46±1.25	94.79±0.03	89.52±0.10	90.94±0.31	40.82±0.42	0.02	50.72±0.97	9.88±0.52
TABSYN	0.866±0.021	98.22±0.04	98.53±0.02	95.83±0.33	43.97±0.27	95.85	49.86±0.75	34.42±0.90
TABDIFF	0.866±0.021	97.65±0.03	98.72±0.04	97.36±0.17	42.10±0.32	93.08	52.46±0.75	16.03±0.66
TABREP-DDPM	0.836±0.001	**98.46±0.01**	**99.09±0.05**	95.35±0.11	48.49±0.12	**96.70**	**49.87±0.99**	**40.10±0.55**
TABREP-FLOW	**0.814±0.002**	96.89±0.03	98.34±0.29	90.91±0.25	**51.75±0.16**	88.13	50.90±0.93	35.48±0.78

G.2. Additional Ablation Results

METHODS	AUC ↑	CDE ↑	PCC ↑	α ↑	β ↑	C2ST ↑	MIA P. ↕	MIA R. ↕
ADULT								
ONEHOT-DDPM	0.476±0.057	48.94±0.12	38.05±0.08	17.44±0.21	0.70±0.01	1.89	38.67±19.02	0.03±0.02
LEARNED1D-DDPM	0.611±0.008	53.44±3.39	27.97±3.71	6.64±0.05	0.00±0.01	0.00	10.00±10.00	0.00±0.00
LEARNED2D-DDPM	0.793±0.003	62.69±1.16	38.53±1.54	7.11±0.21	0.02±0.03	0.00	38.00±5.61	0.03±0.01
I2B-DDPM	0.911±0.001	99.10±0.07	97.55±0.26	98.21±0.18	47.44±0.08	98.01	50.56±0.15	50.68±0.87
DIC-DDPM	0.912±0.002	98.95±0.03	97.65±0.12	97.99±0.54	51.09±0.24	93.61	51.34±0.20	51.62±0.33
ONEHOT-FLOW	0.895±0.003	90.66±0.07	84.32±0.10	88.35±0.15	30.64±0.13	38.88	51.08±0.60	13.49±0.36
LEARNED1D-FLOW	0.260±0.014	60.00±3.13	36.20±3.61	6.82±0.08	0.02±0.03	0.00	24.67±10.41	0.02±0.01
LEARNED2D-FLOW	0.126±0.012	62.37±3.03	38.94±4.20	6.64±3.49	0.05±0.04	0.00	28.19±9.68	0.04±0.01
I2B-FLOW	0.911±0.001	98.23±0.09	97.14±0.32	**99.54**±0.31	48.87±0.16	92.18	50.70±0.31	**49.68±0.79**
DIC-FLOW	0.910±0.002	98.10±0.03	96.63±0.03	99.64±0.10	50.29±0.12	90.70	50.55±0.16	42.03±0.42
TABFLOW	0.908±0.002	96.21±0.05	93.59±0.05	86.76±0.28	**53.15±0.14**	77.48	50.99±0.38	43.90±0.18
TABREP-DDPM	**0.913±0.002**	**99.39±0.04**	**98.63±0.04**	99.11±0.25	52.04±0.12	**99.50**	**50.44±0.83**	52.78±0.21
TABREP-FLOW	0.912±0.002	98.63±0.02	97.55±0.23	98.21±0.34	49.91±0.28	95.48	50.65±0.20	50.51±0.21
DEFAULT								
METHODS	AUC ↓	CDE ↑	PCC ↑	α ↑	β ↑	C2ST ↑	MIA P. ↕	MIA R. ↕
ONEHOT-DDPM	0.557±0.052	50.88±0.10	50.88±0.07	4.13±0.05	0.15±0.01	0.11	40.00±24.49	0.08±0.05
LEARNED1D-DDPM	0.575±0.012	72.53±1.92	51.95±2.30	10.96±2.03	0.00±0.01	0.13	0.00±0.00	0.00±0.00
LEARNED2D-DDPM	0.290±0.009	71.11±0.32	50.79±0.36	11.99±2.54	0.01±0.01	0.06	15.33±11.62	0.11±0.08
I2B-DDPM	0.762±0.003	98.76±0.06	**98.36±0.12**	98.20±0.17	47.46±0.37	98.34	50.95±0.45	45.33±0.77
DIC-DDPM	0.763±0.007	98.33±0.11	92.76±0.43	97.20±0.35	49.62±0.37	96.88	51.60±0.70	48.13±0.75
ONEHOT-FLOW	0.759±0.005	91.95±0.05	88.04±1.51	93.12±0.31	30.50±0.19	69.14	51.40±1.23	12.32±0.60
LEARNED1D-FLOW	0.438±0.009	73.52±0.82	53.01±0.21	20.03±5.43	0.00±0.01	0.53	10.00±10.00	0.05±0.05
LEARNED2D-FLOW	0.709±0.008	66.33±4.86	44.11±6.75	2.33±12.11	0.01±0.02	0.01	10.00±10.00	0.03±0.03
I2B-FLOW	0.763±0.004	97.67±0.13	94.65±1.28	97.42±0.57	49.15±0.48	90.04	50.66±0.33	43.09±1.25
DIC-FLOW	0.759±0.007	97.27±0.03	92.26±1.77	95.97±0.27	51.29±0.18	90.58	51.36±0.62	45.95±0.76
TABFLOW	0.742±0.008	97.38±0.03	95.01±1.44	96.92±0.12	**53.12±0.29**	85.89	**50.05±0.53**	44.11±0.65
TABREP-DDPM	0.764±0.005	**98.97±0.19**	96.74±0.62	**98.66±0.24**	48.22±0.48	**98.90**	50.07±0.41	**48.96±0.41**
TABREP-FLOW	**0.782±0.005**	97.45±0.06	92.86±1.75	96.50±0.44	49.99±0.23	89.36	51.02±0.95	47.07±0.40
SHOPPERS								
METHODS	AUC ↑	CDE ↑	PCC ↑	α ↑	β ↑	C2ST ↑	MIA P. ↕	MIA R. ↕
ONEHOT-DDPM	0.799±0.126	90.37±0.14	84.61±0.10	90.67±0.26	37.76±0.57	54.59	51.00±0.97	28.54±1.49
LEARNED1D-DDPM	0.876±0.028	78.94±4.13	62.67±6.40	21.94±7.75	1.17±0.81	1.36	59.09±18.85	0.84±0.26
LEARNED2D-DDPM	0.103±0.011	70.97±3.06	51.21±4.37	8.43±5.81	0.05±0.09	0.09	0.00±0.00	0.00±0.00
I2B-DDPM	0.919±0.005	98.67±0.05	**98.00±0.03**	**97.69±0.63**	50.86±0.11	96.28	51.77±0.98	46.80±1.91
DIC-DDPM	0.910±0.005	97.83±0.15	96.19±0.09	95.33±0.80	56.14±0.85	91.09	50.68±0.52	46.02±1.20
ONEHOT-FLOW	0.910±0.006	92.84±0.08	91.50±0.14	87.57±0.44	48.77±0.69	65.08	49.64±1.69	34.56±0.80
LEARNED1D-FLOW	0.134±0.015	76.25±2.16	60.05±3.32	10.78±7.79	0.10±1.00	0.01	40.00±24.49	0.13±0.08
LEARNED2D-FLOW	0.868±0.007	71.93±0.94	53.29±1.34	16.58±1.72	0.28±0.23	0.07	57.33±20.50	0.65±0.34
I2B-FLOW	0.910±0.005	97.61±0.11	97.20±0.11	97.24±0.66	54.88±0.26	91.83	51.56±1.01	45.11±1.20
DIC-FLOW	0.903±0.006	96.89±0.14	95.78±0.24	95.84±0.38	52.39±0.26	88.74	50.86±0.71	52.53±0.44
TABFLOW	0.914±0.002	95.03±0.04	92.87±0.04	77.55±0.19	**61.94±0.53**	73.74	51.62±0.82	42.59±0.74
TABREP-DDPM	**0.926±0.005**	**98.97±0.10**	97.62±0.02	96.14±0.19	53.68±0.73	**96.37**	**49.86±0.98**	48.16±0.90
TABREP-FLOW	0.919±0.005	97.74±0.03	97.08±0.07	95.85±0.46	55.92±0.37	94.20	51.38±1.66	**49.19±0.86**
STROKE								
METHODS	AUC ↑	CDE ↑	PCC ↑	α ↑	β ↑	C2ST ↑	MIA P. ↕	MIA R. ↕
ONEHOT-DDPM	0.797±0.033	98.74±0.11	97.65±0.10	96.78±1.08	56.27±0.22	98.62	53.08±2.50	**49.76±1.24**
LEARNED1D-DDPM	0.743±0.032	60.20±11.13	35.86±15.52	6.97±42.82	0.26±3.51	0.01	15.00±0.20	0.33±10.00
LEARNED2D-DDPM	0.850±0.035	59.11±14.59	33.77±19.97	6.67±36.39	0.87±2.42	0.02	18.33±0.33	0.49±13.02
I2B-DDPM	0.852±0.029	99.02±0.18	95.12±1.47	98.14±0.16	64.11±0.75	99.77	50.72±2.05	49.11±1.59
DIC-DDPM	0.824±0.021	98.98±0.09	**98.00±2.13**	98.25±0.62	65.00±0.88	99.39	52.97±1.19	52.03±1.76
ONEHOT-FLOW	0.812±0.029	94.17±0.08	90.45±0.08	81.87±0.55	49.23±0.56	64.82	49.32±1.47	39.51±1.25
LEARNED1D-FLOW	0.142±0.033	75.48±6.55	56.60±9.05	71.63±35.72	0.45±0.13	0.16	47.00±0.55	1.14±20.22
LEARNED2D-FLOW	0.180±0.030	54.02±13.09	28.95±17.18	5.67±24.97	0.21±0.67	0.00	20.00±0.16	0.16±20.00
I2B-FLOW	0.797±0.027	98.72±0.11	94.65±1.33	98.26±0.27	65.23±0.26	97.19	52.72±2.26	52.03±1.18
DIC-FLOW	0.807±0.019	98.50±0.23	92.71±2.35	97.76±0.64	65.77±1.95	97.33	51.45±1.29	48.78±1.65
TABFLOW	0.868±0.035	97.72±0.01	96.00±0.03	89.21±0.82	**68.30±0.60**	89.19	51.80±1.67	47.32±1.95
TABREP-DDPM	**0.869±0.027**	**99.14±0.20**	97.11±0.60	**98.32±0.82**	57.17±0.77	**100.00**	51.74±1.85	50.89±0.93
TABREP-FLOW	0.854±0.028	98.42±0.31	97.37±2.12	96.40±0.71	63.91±0.87	95.96	**50.66±1.77**	49.43±1.16

	DIABETES							
METHODS	F1 ↑	CDE ↑	PCC ↑	α ↑	β ↑	C2ST ↑	MIA P. ↕	MIA R. ↕
ONEHOT-DDPM	0.363±0.008	69.22±0.04	50.14±0.06	9.93±0.16	2.34±0.11	1.74	45.56±0.03	0.36±2.52
LEARNED1D-DDPM	0.179±0.010	63.40±4.59	39.94±5.34	0.00±0.00	0.00±0.00	0.00	0.00±0.00	0.00±0.00
LEARNED2D-DDPM	0.205±0.007	64.30±1.31	41.49±1.74	0.01±0.01	0.00±0.00	0.00	8.00±0.00	0.00±8.00
I2B-DDPM	0.370±0.008	99.38±0.01	98.84±0.03	97.70±0.13	46.83±0.39	93.98	50.08±0.39	51.63±0.12
DIC-DDPM	0.375±0.006	**99.50±0.02**	**99.12±0.01**	98.92±0.13	48.34±0.18	**95.03**	50.37±0.25	54.48±0.74
ONEHOT-FLOW	0.372±0.005	96.65±0.03	94.61±1.99	97.43±0.06	41.64±0.16	55.44	49.49±0.21	22.78±0.26
LEARNED1D-FLOW	0.184±0.007	53.27±7.48	28.31±8.51	0.00±0.00	0.00±0.00	0.00	0.00±0.00	0.00±0.00
LEARNED2D-FLOW	0.177±0.008	68.39±9.23	46.71±10.95	0.00±0.00	0.00±0.00	0.00	0.00±0.00	0.00±0.00
I2B-FLOW	0.372±0.003	98.92±0.03	98.34±0.05	98.45±0.14	48.97±0.29	89.28	49.86±0.26	48.25±0.13
DIC-FLOW	0.376±0.007	99.01±0.03	98.54±0.03	99.65±0.10	48.98±0.10	89.06	50.10±0.12	48.01±0.18
TABFLOW	0.376±0.006	98.04±0.02	96.82±0.01	79.60±0.09	**51.58±0.04**	73.35	50.19±0.39	44.14±0.15
TABREP-DDPM	0.373±0.003	99.36±0.02	98.75±0.03	97.19±0.22	46.19±0.49	92.65	**50.07±0.37**	51.43±0.13
TABREP-FLOW	**0.377±0.002**	99.00±0.02	98.46±0.05	**99.08±0.13**	48.58±0.17	90.41	50.24±0.37	**50.33±0.13**

	BEIJING							
METHODS	RMSE ↓	CDE ↑	PCC ↑	α ↑	β ↑	C2ST ↑	MIA P. ↕	MIA R. ↕
ONEHOT-DDPM	2.143±0.339	74.21±0.07	63.68±0.05	48.88±0.03	19.07±0.13	19.68	50.28±2.95	5.44±0.50
LEARNED1D-DDPM	0.921±0.006	79.49±3.50	62.23±5.61	26.94±27.91	8.49±5.95	13.51	45.45±2.82	1.97±0.13
LEARNED2D-DDPM	0.969±0.005	81.89±1.24	66.02±2.37	79.49±16.01	7.73±1.53	55.83	49.20±1.89	2.05±0.10
I2B-DDPM	0.542±0.008	98.66±0.04	96.95±0.21	97.92±0.15	59.27±0.14	94.28	50.64±0.47	48.43±0.89
DIC-DDPM	0.547±0.007	98.83±0.04	97.21±0.16	98.97±0.30	61.90±0.12	**98.83**	51.32±0.35	50.61±0.66
ONEHOT-FLOW	0.765±0.016	84.61±0.02	67.28±4.64	84.38±0.61	20.32±0.19	35.76	51.44±1.49	8.12±0.43
LEARNED1D-FLOW	0.806±0.009	80.14±1.60	64.19±2.33	81.15±3.68	13.11±0.38	20.40	44.02±3.58	1.13±0.09
LEARNED2D-FLOW	0.787±0.007	79.50±0.85	63.04±1.01	43.00±4.90	7.55±4.05	14.58	54.23±1.72	2.01±0.09
I2B-FLOW	0.543±0.007	98.08±0.04	96.87±0.43	96.83±0.12	60.58±0.19	91.66	49.64±0.47	45.77±1.07
DIC-FLOW	0.561±0.013	98.09±0.03	96.37±0.08	97.04±0.22	60.78±0.10	93.96	50.86±0.71	52.53±0.44
TABFLOW	0.574±0.01	96.44±0.06	93.71±0.07	94.81±0.42	59.47±0.28	87.23	50.54±0.35	42.20±0.61
TABREP-DDPM	**0.508±0.006**	**99.11±0.03**	**96.97±0.20**	**98.98±0.16**	**64.08±0.18**	98.16	51.02±0.42	**49.50±0.52**
TABREP-FLOW	0.536±0.006	98.28±0.07	96.92±0.21	98.16±0.13	62.65±0.12	92.26	**50.35±0.60**	49.14±0.81

	NEWS							
METHODS	RMSE ↓	CDE ↑	PCC ↑	α ↑	β ↑	C2ST ↑	MIA P. ↕	MIA R. ↕
ONEHOT-DDPM	0.840±0.02	98.11±0.06	92.78±0.08	96.33±0.24	46.87±0.28	95.80	50.25±0.46	32.42±0.80
LEARNED1D-DDPM	0.858±0.01	96.45±0.14	95.00±0.32	90.48±9.11	19.54±4.84	21.58	51.33±0.96	17.60±1.02
LEARNED2D-DDPM	0.857±0.023	96.06±0.49	94.49±0.84	88.83±4.52	5.11±5.24	20.41	50.85±2.97	3.83±0.39
I2B-DDPM	0.844±0.013	98.41±0.02	98.57±0.18	95.02±0.10	48.22±0.18	96.07	50.88±0.57	39.82±0.93
DIC-DDPM	0.866±0.019	98.22±0.06	98.50±0.36	97.08±0.25	47.88±0.10	95.75	49.54±0.57	39.80±0.35
ONEHOT-FLOW	0.850±0.017	96.27±0.05	98.11±0.02	**97.78±0.13**	43.06±0.62	84.56	50.45±2.02	14.84±0.65
LEARNED1D-FLOW	0.873±0.007	95.07±0.14	95.07±0.12	89.16±5.89	18.17±6.18	79.17	49.85±1.69	10.28±0.47
LEARNED2D-FLOW	0.866±0.005	94.85±0.38	94.99±0.89	80.66±15.23	14.63±4.14	78.96	49.59±1.54	14.19±0.69
I2B-FLOW	0.847±0.014	96.64±0.05	98.38±0.34	88.39±0.11	**51.85±0.24**	89.47	51.12±0.71	36.35±0.53
DIC-FLOW	0.853±0.014	96.58±0.04	97.49±0.34	92.28±0.25	50.79±0.29	88.30	50.43±0.63	37.56±0.77
TABFLOW	0.850±0.017	96.51±0.08	97.93±0.02	92.68±0.31	50.03±0.26	87.33	51.01±0.24	28.00±0.89
TABREP-DDPM	0.836±0.001	**98.46±0.01**	**99.09±0.05**	95.35±0.11	48.49±0.12	**96.70**	**49.87±0.99**	**40.10±0.55**
TABREP-FLOW	**0.814±0.002**	96.89±0.03	98.34±0.29	90.91±0.25	51.75±0.16	88.13	50.90±0.93	35.48±0.78

G.3. Additional Training and Sampling Duration Results

We conducted an additional Training and Sampling Duration experiment on the largest dataset among our dataset suite (Diabetes dataset) with 99,473 samples, 8 numerical features, and 21 categorical features. As observed in Table 14, we save around 1700 seconds compared to TabDDPM and around 4900 seconds compared to TabSYN during training and sampling.

Table 14. Training and Sampling Duration in Seconds.

METHODS	TRAINING	SAMPLING	TOTAL
TABDDPM	3455	268	3723
TABSYN	6003 + 882	18	6903
TABREP-DDPM	1980	114	2094
TABREP-FLOW	**2002**	**7**	**2009**

G.4. Additional Results on High Cardinality and Imbalanced Toy Datasets

We curate synthetic toy datasets of high cardinality categorical variables and imbalanced datasets to reinforce our generalizability claims. Our high cardinality toy dataset is a regression task with two categorical features. The first categorical feature is of high cardinality, with 1000 unique categories where each category is assigned a base effect drawn from a normal distribution. The other categorical feature may take 3 values, each having fixed effects of 1.0, -1.0, and 0.5 respectively. The target label is computed by summing the base effect from the high-cardinality category, the fixed effect from the other categorical feature, and an independent numerical feature drawn from a standard normal distribution. Additional Gaussian noise is added to perturb the data.

Table 15. Performance on High Cardinality Setting.

	RMSE ↓	CDE ↑	PWC ↑	C2ST ↑	α-PRECISION ↑	β-RECALL ↑
TABDDPM	0.8253$\pm$0.1419	42.93$\pm$0.17	11.20$\pm$0.09	0.41$\pm$0.02	0.46$\pm$0.01	0.02$\pm$0.01
TABSYN	0.4775$\pm$0.0129	94.23$\pm$0.56	**78.37$\pm$2.00**	**100.00$\pm$0.00**	99.13$\pm$0.31	35.35$\pm$0.47
TABREP-DDPM	**0.4662$\pm$0.0071**	80.88$\pm$0.15	59.96$\pm$1.08	25.45$\pm$0.10	99.31$\pm$0.21	16.47$\pm$0.32
TABREP-FLOW	0.4812$\pm$0.0104	**94.38$\pm$0.28**	76.84$\pm$1.08	98.59$\pm$0.94	**99.32$\pm$0.22**	**36.00$\pm$0.24**

As shown in Table 15, it is worth noting that DDPM models including TabRep-DDPM and TabDDPM perform poorly in CDE, PWC, and C2ST tasks, yet are able to model RMSE and α-precision well. This indicates that with high cardinality, TabRep-DDPM is less capable of learning conditional distributions across features. In contrast, our proposed TabRep-Flow performs on par with TabSYN, as flow-matching models' smooth differentiable transformation allows them to capture subtle conditional variations, and TabSYN's latent space allows for the learning of a simpler latent distribution.

The imbalanced toy dataset is a regression task with one binary categorical feature (distributed 95% class A, 5% class B). Each row also contains a numeric feature drawn from a standard normal distribution. The target is constructed by applying a category-specific linear function before addition of some Gaussian noise to generate variations in the data.

Table 16. Performance on Imbalanced Setting.

	RMSE ↓	CDE ↑	PWC ↑	C2ST ↑	α-PRECISION ↑	β-RECALL ↑
TABDDPM	0.1694$\pm$0.0016	98.93$\pm$0.54	95.87$\pm$5.43	98.98$\pm$1.57	98.96$\pm$0.68	50.29$\pm$0.49
TABSYN	0.1708$\pm$0.0018	94.98$\pm$0.04	96.36$\pm$0.18	90.27$\pm$0.65	95.84$\pm$1.13	48.89$\pm$0.23
TABREP-DDPM	**0.1688$\pm$0.0010**	**99.44$\pm$0.13**	**96.82$\pm$3.97**	**99.42$\pm$0.54**	**99.46$\pm$0.08**	50.96$\pm$0.77
TABREP-FLOW	0.1689$\pm$0.0025	98.52$\pm$0.12	90.49$\pm$5.36	99.06$\pm$0.47	96.69$\pm$0.28	**51.30$\pm$0.31**

In Table 16, we see that for imbalanced data, our proposed methods achieve results that are better compared to existing models like TabDDPM and TabSYN, showing that our methods are generalizable to cases where training data may be highly imbalanced.

G.5. TabSYN's Latent Representation Dimension

By default, TabSYN has a latent dimensionality of 4. To address the concern regarding TabSYN's dimensionality, we run TabSYN using the same dimensions (2D latent space) as our TabRep representation. As observed in Table 17, TabSYN with a 2D latent dimension performs much worse than TabSYN with a 4D latent dimension.

Table 17. AUC (classification) and RMSE (regression) scores of Machine Learning Efficiency. Higher scores indicate better performance.

	AUC ↑				F1 ↑	RMSE ↓	
METHODS	ADULT	DEFAULT	SHOPPERS	STROKE	DIABETES	BEIJING	NEWS
TABSYN	0.906±.001	0.755±.004	0.918±.004	0.845±.035	0.361±.001	0.586±.013	0.862±.021
TABSYN (2D LATENT SPACE)	0.892±.002	0.752±.005	0.916±.002	0.811±.032	0.368±.002	0.720±.015	0.868±.003
TABREP-DDPM	**0.913±.002**	0.764±.005	**0.926±.005**	**0.869±.027**	0.373±.003	**0.508±.006**	0.836±.001
TABREP-FLOW	0.912±.002	**0.782±.005**	0.919±.005	0.830±.028	**0.377±.002**	0.536±.006	**0.814±.002**

In terms of computational costs on the Adult dataset, Table 18 highlights that TabSYN in a 2D Latent Space saves close to 500 seconds while compromising on accuracy when compared to vanilla TabSYN (4D Latent Space). However, it still consumes around 1000 seconds extra when compared to TabRep methods.

Table 18. Training and Sampling Duration in Seconds.

METHODS	TRAINING	SAMPLING	TOTAL
TABSYN	2374 + 1085	11	3470
TABSYN (2D LATENT SPACE)	2333 + 670	6	3009
TABREP-DDPM	2071	59	2130
TABREP-FLOW	**2028**	**3**	**2031**