TABDLM: Free-Form Tabular Data Generation via Joint Numerical–Language Diffusion

Donghong Cai[1] **Jiarui Feng**[1] **Yanbo Wang**[2] **Da Zheng**[3]
Yixin Chen[1] **Muhan Zhang**[2]
{cai.d, feng.jiarui, ychen25}@wustl.edu, wangyanbo@stu.pku.edu.cn
zhengda.zheng@antgroup.com, muhan@pku.edu.cn
[1]Washington University in St. Louis [2]Peking University [3]Ant Group

Abstract

Synthetic tabular data generation has attracted growing attention due to its importance for data augmentation, foundation models, and privacy. However, real-world tabular datasets increasingly contain *free-form text* fields (e.g., reviews or clinical notes) alongside structured numerical and categorical attributes. Generating such heterogeneous tables with joint modeling of **different modalities** remains challenging. Existing approaches broadly fall into two categories: diffusion-based methods and LLM-based methods. Diffusion models can capture complex dependencies over numerical and categorical features in continuous or discrete spaces, but extending them to open-ended text is nontrivial and often leads to degraded text quality. In contrast, LLM-based generators naturally produce fluent text, yet their discrete tokenization can distort precise or wide-range numerical values, hindering accurate modeling of both numbers and language. In this work, we propose TABDLM, a unified framework for free-form tabular data generation via a joint numerical–language diffusion model built on masked diffusion language models (MDLMs). TABDLM models textual and categorical features through masked diffusion, while modeling numerical features with a continuous diffusion process through learned specialized numeric tokens embedding; bidirectional attention then captures cross-modality interactions within a single model. Extensive experiments on diverse benchmarks demonstrate the effectiveness of TABDLM compared to strong diffusion- and LLM-based baselines. The code is available at https://github.com/ilikevegetable/TabDLM

1 Introduction

Tabular data is one of the most fundamental data types in modern machine learning and is indispensable across domains such as finance [32], healthcare [15], and social media [22]. In practice, however, deploying machine learning on tabular datasets faces recurring obstacles, including privacy and security constraints [12, 3], limited data availability [10], and missing values [45, 42]. These challenges motivate synthetic tabular data generation, which seeks to produce synthetic samples that match the schema and key statistical properties of the original dataset for replacing the original data.

High-fidelity synthetic tabular data generation remains challenging because real-world tables exhibit complex dependencies across columns [40, 6] and often contain a mixture of numerical, categorical, and free-form textual features [33]. Existing approaches can broadly be categorized into diffusion-based and large language model (LLM)-based methods. Diffusion-based methods employ diffusion processes [35, 13, 34, 4] to learn denoising transformations that approximate the data distribution during training and to generate realistic samples from noise during inference. Such models have been widely applied to tabular data generation, particularly for continuous (numerical) features [18, 17, 45]. More recent work has explored combining continuous and discrete diffusion to jointly model

Preprint.

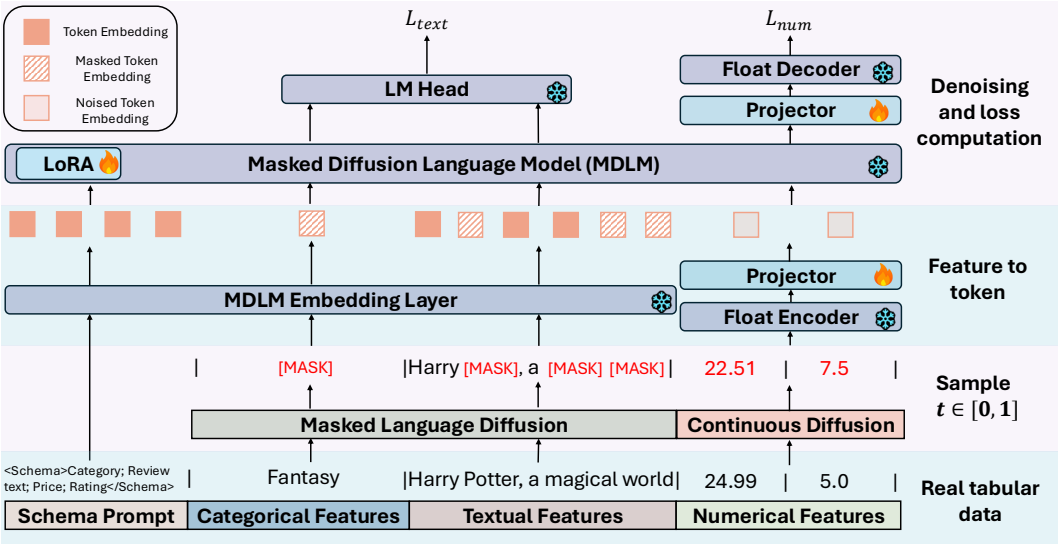

Figure 1: Overview of TABDLM. During training, given an input tabular sample, TABDLM applies masked language diffusion to categorical and free-form textual features, and continuous diffusion to numerical features. Noisy inputs are mapped to token embeddings via a textual embedding layer and a numerical encoder. An MDLM then denoises these embeddings to reconstruct the original sample. Training updates only the projector in the numerical encoder and decoder, as well as the LoRA modules within each MDLM layer.

numerical and categorical attributes [44, 21, 23, 33]. However, extending standard diffusion models to open-ended textual fields remains challenging due to the **exponential size of the text space**. To the best of our knowledge, no existing diffusion-based methods have been successfully applied to tabular data generation involving open-ended text. In contrast, LLM-based methods naturally support free-form text generation [6, 46] owing to their strong language modeling capabilities. However, their token-level representations can be unreliable for modeling high-precision or wide-range numerical values [41], as **numbers are often fragmented into multiple tokens**. Furthermore, **autoregressive generation enforces a left-to-right dependency structure** that is misaligned with the mutually dependent relationships across tabular columns.

To address the above limitations, we propose TABDLM, a unified framework that integrates the complementary strengths of diffusion-based and LLM-based modeling. To preserve the most effective modeling paradigm for each modality, TABDLM employs continuous diffusion for numerical columns and language models for categorical and free-form textual columns. However, rather than relying on autoregressive language models, we adopt Masked Diffusion Language Models (MDLMs) as the backbone architecture. MDLMs offer two key advantages. First, their diffusion-based formulation enables TABDLM to model numerical and language modalities within a unified generative process. Second, unlike autoregressive models, MDLMs employ bidirectional attention, enabling the model to capture mutual dependencies among tabular columns. To enable joint numerical–language diffusion within a single MDLM, we introduce a trainable numerical tokenization module into the MDLM architecture, which allows continuous diffusion to represent each numerical value with a single token. As a result, TABDLM learns to **jointly denoise numerical values and language content** during training and generates synthetic tabular data by simultaneously performing continuous diffusion and masked language diffusion during inference. The overview of the TABDLM is provided in Figure 1. We evaluate TABDLM across a range of scenarios and benchmarks, where it consistently outperforms both diffusion-based and language-based baselines.

2 Methods

2.1 Preliminary

Problem setup and notation. Let $\mathcal{D} = \{\mathbf{x}^{(i)}\}_{i=1}^{N}$ denote a tabular dataset with M columns. Each record is $\mathbf{x}^{(i)} = (x_1^{(i)}, \ldots, x_M^{(i)})$, where columns can be either numerical, categorical, or textual. We use index sets $\mathcal{I}_{\text{num}}$, $\mathcal{I}_{\text{cat}}$, and $\mathcal{I}_{\text{text}}$ to denote numerical, categorical, and text columns, respectively. Our goal is to learn a generative model $p_\theta(\mathbf{x})$ that can sample realistic records while capturing cross-column dependencies.

Continuous diffusion. We model the distribution of a continuous vector $\mathbf{z}_0 \in \mathbb{R}^d$ using a diffusion process defined by the stochastic differential equation (SDE) $\mathrm{d}\mathbf{z} = \mathbf{f}(\mathbf{z}, t)\mathrm{d}t + g(t)\mathrm{d}\mathbf{w}$. In the variance-exploding (VE) formulation [36], we set $\mathbf{f}(\mathbf{z}, t) = \mathbf{0}$ and $g(t) = \sqrt{2\dot{\sigma}(t)\sigma(t)}$, where $\sigma(t) : [0, 1] \to \mathbb{R}_+$ is a strictly increasing function governing the noise level and $\dot{\sigma}(t) = \frac{\mathrm{d}\sigma(t)}{\mathrm{d}t}$ denotes its time derivative. From a probabilistic perspective, this SDE corresponds to a continuous-time limit of a Markov chain where Gaussian noise is progressively added to the data. The transition kernel $q(\mathbf{z}_t \mid \mathbf{z}_s)$ for $s < t$ and the marginal $q(\mathbf{z}_t \mid \mathbf{z}_0)$ are given by:

$$\begin{aligned} q(\mathbf{z}_t \mid \mathbf{z}_s) &= \mathcal{N}\big(\mathbf{z}_s,\, (\sigma^2(t) - \sigma^2(s))\mathbf{I}\big)\,, \\ q(\mathbf{z}_t \mid \mathbf{z}_0) &= \mathcal{N}\big(\mathbf{z}_0,\, \sigma^2(t)\mathbf{I}\big)\,. \end{aligned} \tag{1}$$

The reverse generative process solves the probability flow ODE [36], as described in the below:

$$\mathrm{d}\mathbf{z} = -\frac{1}{2}g(t)^2\nabla_{\mathbf{z}} \log p_t(\mathbf{z})\mathrm{d}t = -\dot{\sigma}(t)\sigma(t)\nabla_{\mathbf{z}} \log p_t(\mathbf{z})\mathrm{d}t. \tag{2}$$

We use a parameterized denoising model $\boldsymbol{\epsilon}_\theta(\mathbf{z}_t, t)$ that estimates the score function via $\nabla_{\mathbf{z}} \log p_t(\mathbf{z}) \approx -\boldsymbol{\epsilon}_\theta(\mathbf{z}_t, t)/\sigma(t)$. Training minimizes the denoising error:

$$\mathcal{L}_{\text{diff}} = \mathbb{E}_{t,\mathbf{z}_0,\boldsymbol{\epsilon}}\left[\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\mathbf{z}_0 + \sigma(t)\boldsymbol{\epsilon}, t)\|_2^2\right], \;\; \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}). \tag{3}$$

Masked diffusion language models (MDLMs). MDLMs can be viewed as a discrete-state diffusion process [4] over token sequences. Let $\mathbf{s}_0 = (s_1, \ldots, s_L)$ be a length-L sequence with tokens in a vocabulary $\mathcal{V}$ augmented with an absorbing mask state $m = [\text{MASK}]$. Using a token-wise Markov chain, the forward noising kernel factorizes as

$$q(\mathbf{s}_t \mid \mathbf{s}_{t-1}) = \prod_{j=1}^{L} q(s_{t,j} \mid s_{t-1,j}), \tag{4}$$

with a masking schedule $\beta_t \in [0, 1]$ and absorbing transitions

$$q(s_{t,j} \mid s_{t-1,j}) = \begin{cases} 1, & s_{t-1,j} = m,\ s_{t,j} = m, \\ \beta_t, & s_{t-1,j} \neq m,\ s_{t,j} = m, \\ 1 - \beta_t, & s_{t-1,j} \neq m,\ s_{t,j} = s_{t-1,j}, \\ 0, & \text{otherwise}, \end{cases} \tag{5}$$

Equivalently, the marginal has the closed form $q(s_{t,j} = s_{0,j} \mid s_{0,j}) = \bar{\alpha}_t$ and $q(s_{t,j} = m \mid s_{0,j}) = 1 - \bar{\alpha}_t$, where $\bar{\alpha}_t = \prod_{\tau=1}^{t}(1 - \beta_\tau)$. The reverse model uses a bidirectional Transformer to predict the original token distribution from the partially masked sequence:

$$p_\theta(\mathbf{s}_0 \mid \mathbf{s}_t) = \prod_{j=1}^{L} p_\theta(s_{0,j} \mid \mathbf{s}_t), \tag{6}$$

and is typically trained by maximizing the log-likelihood of the ground-truth tokens at corrupted (masked) positions:

$$\mathcal{L}_{\text{MDLM}} = \mathbb{E}_{t,\mathbf{s}_0,\mathbf{s}_t}\left[-\sum_{\{j:\, s_{t,j} = [\text{MASK}]\}} \log p_\theta(s_{0,j} \mid \mathbf{s}_t)\right]. \tag{7}$$

At sampling time, MDLMs start from an all-mask sequence and iteratively unmask tokens according to p_θ.

2.2 The Model Architecture of TABDLM

We first present the architecture of TABDLM. As illustrated in Figure 1, TABDLM comprises five main components: (1) a numerical encoder module, (2) a text embedding layer, (3) a masked diffusion language model, (4) a numerical decoder module, and (5) an LM head. We describe each component in detail below. Note that in this section we use $\hat{x}_j^{(i)}$ instead of $x_j^{(i)}$ to denote the noise version of the input. We will describe how we obtain the $\hat{x}_j^{(i)}$ from $x_j^{(i)}$ in Section 2.3.

Number encoder module. Given an input numerical value $\hat{x}_j^{(i)} \in \mathbb{R}, j \in \mathcal{I}_{\text{num}}$, the number encoder module transforms the number into a token embedding with the same size as the MDLM. To achieve this, we first use quantile normalization [5] to standardize the numerical values as done in previous works [38, 33]. Then, we use a pretrained Multi-Layer Perceptron (MLP) as the float number encoder to convert the normalized values into an r-dimensional embedding vector. Finally, a trainable projection is used to align the r-dimensional embedding vector into the d-dimensional MDLM embedding space:

$$\hat{\mathbf{z}}_j^{(i)} = \text{ENC}(\hat{x}_j^{(i)}), \;\; \mathbf{z}_j^{(i)} = \text{PROJ}_e(\hat{\mathbf{z}}_j^{(i)}), \;\; j \in \mathcal{I}_{\text{num}}, \tag{8}$$

where ENC and PROJ_e denote the MLP-based encoder and projection module, respectively, with $\hat{\mathbf{z}}_j^{(i)} \in \mathbb{R}^r$ and $\mathbf{z}_j^{(i)} \in \mathbb{R}^d$. In our implementation, we pretrained ENC based on Griffin [38] and keep it fixed throughout all experiments, while PROJ_e is jointly optimized during training.

Text embedding layer. The text embedding layer transforms textual input features $\hat{x}_j^{(i)}, j \in \mathcal{I}_{\text{cat}} \cup \mathcal{I}_{\text{text}}$ into a token embedding sequence. We directly adopt the pretrained embedding table from the underlying MDLM. Formally,

$$\mathbf{z}_j^{(i)} = (\mathbf{z}_{j,1}^{(i)}, \ldots, \mathbf{z}_{j,l_j^i}^{(i)}) = \text{EMB}(\hat{x}_j^{(i)}), j \in \mathcal{I}_{\text{cat}} \cup \mathcal{I}_{\text{text}}, \tag{9}$$

where each token embedding $\mathbf{z}_{j,k}^{(i)} \in \mathbb{R}^d, k \in [l_j^i]$ and l_j^i denotes the token sequence length of column j in sample i, as each textual column may be mapped to multiple tokens in the MDLM embedding space.

Masked diffusion language model. The masked diffusion language model is employed to jointly denoise both numerical and textual tokens. Formally, given an input sequence of token embeddings $\mathbf{z}^{(i)} = (\mathbf{z}_1^{(i)}, \ldots, \mathbf{z}_S^{(i)})$ of length S and a diffusion time step t, the MDLM outputs a denoised sequence $\mathbf{o}^{(i)}$ that estimates the corresponding representations at time $t = 0$. In TABDLM, the input token sequence is constructed by concatenating four components: a schema prompt, categorical features, textual features, and numerical features. The schema prompt provides a textual description of the feature type and the semantic meaning of each column, which is fixed for a given dataset. In detail, the input token sequence for sample i is represented as:

$$\mathbf{z}^{(i)} = \big(\mathbf{z}_p, (\mathbf{z}_j^{(i)} | j \in \mathcal{I}_{\text{cat}} \cup \mathcal{I}_{\text{text}}), (\mathbf{z}_j^{(i)} | j \in \mathcal{I}_{\text{num}})\big),$$

where $\mathbf{z}_p$ is the token embedding sequence for the schema prompt and $\mathbf{z}_j^{(i)} = (\mathbf{z}_{j,1}^{(i)}, \ldots, \mathbf{z}_{j,l_j^i}^{(i)})$ for any $j \in \mathcal{I}_{\text{cat}} \cup \mathcal{I}_{\text{text}}$. The forward process of MDLM can be described as:

$$\mathbf{o}^{(i)} = \text{MDLM}(\mathbf{z}^{(i)}, t). \tag{10}$$

For the MDLM architecture, we adopt a standard transformer architecture with bidirectional attention [37, 8]. For numerical tokens, an additional positional embedding is added to encode the diffusion noise level applied to the input, following the design in DiT [29]. No such embedding is added for textual tokens, as the noise level can be implicitly captured by the proportion of masked tokens.

Number decoder module. The number decoder module is used to decode the output token embedding for the numerical value back to the original number. We use a symmetric version to the number encoder module with one trainable projection, along with a pretrained MLP decoder:

$$\bar{\mathbf{o}}_j^{(i)} = \text{PROJ}_d(\mathbf{o}_j^{(i)}), \;\; \bar{x}_j^{(i)} = \text{DEC}(\bar{\mathbf{o}}_j^{(i)}), \;\; j \in \mathcal{I}_{\text{num}}, \tag{11}$$

where DEC and PROJ_d denote the MLP-based decoder and projection module, respectively, with $\mathbf{o}_j^{(i)} \in \mathbb{R}^d$ representing the output token embedding from the MDLM and $\bar{\mathbf{o}}_j^{(i)} \in \mathbb{R}^r$. Similarly, the DEC is pretrained jointly with the float number encoder and fixed during the experiment.

LM head. Finally, the LM head is used to decode the output token embedding for the textual value back to the original text. We directly use the frozen LM head from MDLM:

$$\bar{x}_j^{(i)} = \text{HEAD}(\mathbf{o}_j^{(i)}), \quad j \in \mathcal{I}_{\text{cat}} \cup \mathcal{I}_{\text{text}}. \tag{12}$$

2.3 The Forward Process of TABDLM

The forward diffusion process typically adds noise to the input. In TABDLM, this process is applied to both numerical and textual modalities; for clarity, we describe the forward process for each modality separately.

Forward process of numerical features. For numerical features, we adopt continuous diffusion to better capture fine-grained distributions. Given a clean input value $x_j^{(i)}, j \in \mathcal{I}_{\text{num}}$, we first sample a time $t \in [0, 1]$. Then, according to Equation 1, the forward process adds noise as follows:

$$\hat{x}_j^{(i)} = x_j^{(i)} + \sigma(t)\epsilon, \;\; \epsilon \sim \mathcal{N}(0, 1), \;\; j \in \mathcal{I}_{\text{num}}. \tag{13}$$

Note that the noise is added to the normalized value.

Forward process of textual features. For textual features, we adopt discrete masked diffusion, as used in MDLMs. Given a text sequence $x_j^{(i)} = (x_{j,1}^{(i)}, \ldots, x_{j,l_j^i}^{(i)}), j \in \mathcal{I}_{\text{cat}} \cup \mathcal{I}_{\text{text}}$, we first sample a time $t \in [0, 1]$. Then, according to Equation 5 and the accompanying discussion, the forward process independently transforms each token into a mask token with probability $1 - \bar{\alpha}_t$, resulting in the masked token sequence $\hat{x}_j^{(i)} = (\hat{x}_{j,1}^{(i)}, \ldots, \hat{x}_{j,l_j^i}^{(i)})$.

Finally, in TABDLM, we use a single t for both continuous diffusion and discrete masked diffusion. This design enables the model to jointly denoise numerical and textual modalities under a unified noise level.

2.4 Training of TABDLM

During training, we optimize the TABDLM to jointly denoise both the numerical modality and textual modality. Specifically, given an output sequence $\mathbf{o}^{(i)} = \big(\mathbf{o}_p, (\mathbf{o}_j^{(i)} | j \in \mathcal{I}_{\text{cat}} \cup \mathcal{I}_{\text{text}}), (\mathbf{o}_j^{(i)} | j \in \mathcal{I}_{\text{num}})\big)$ from MDLM, we first transform the numerical token embedding back to real number using the number decoder module and then the loss is computed separately for continuous diffusion and masked diffusion:

$$\mathcal{L}_{\text{num}} = \frac{1}{N} \sum_{i=1}^{N} \sum_{j \in \mathcal{I}_{\text{num}}} \left\| x_j^{(i)} - \bar{x}_j^{(i)} \right\|_2^2, \quad j \in \mathcal{I}_{\text{num}}, \tag{14}$$

$$\begin{aligned} \mathcal{L}_{\text{text}} = & \frac{1}{N} \sum_{i=1}^{N} \sum_{\hat{x}_{j,k}^{(i)} = \text{[MASK]}} \text{CE} < \bar{x}_{j,k}^{(i)}, x_{j,k}^{(i)} >, \\ & k \in [l_j^i], \quad j \in \mathcal{I}_{\text{cat}} \cup \mathcal{I}_{\text{text}}, \end{aligned} \tag{15}$$

where $\text{CE} < \cdot, \cdot >$ is the cross entropy loss. The final objective is optimized using a step-dependent weighting schedule:

$$\mathcal{L} = \mathcal{L}_{\text{text}} + \lambda(s)\, \mathcal{L}_{\text{num}}, \;\; \lambda(s) = \lambda_{\max} \cdot \min\left(1, \frac{s}{s_{\text{warm}}}\right), \tag{16}$$

where s denotes the global optimization step, and this warm-up schedule stabilizes the alignment of newly initialized numerical projections with the pretrained MDLM backbone. We use default hyperparameters $\lambda_{\max} = 1$ and $s_{\text{warm}} = 2000$. In practice, we freeze the pretrained weights of the MDLM and introduce trainable LoRA modules [14] into both the feed-forward and attention components of each transformer layer.

2.5 The backward sampling in TABDLM

Finally, we describe the sampling process of TABDLM. After training, generation starts from noise and progressively denoises the inputs to produce synthetic tabular samples following the learned data distribution. For numerical modality, the process is initialized with pure Gaussian noise, while for textual modality, it starts from a fully masked token sequence. We then iteratively apply Equation 2 and Equation 6 to the numerical and textual modalities to recover clean data. More details are provided in Appendix B.

3 Related Works

Existing methods in tabular data generation can be classified into three categories based on their underlying frameworks.

VAE/GAN-based tabular generators. This line of work formulates tabular data generation using Variational Autoencoders (VAEs) [20] or Generative Adversarial Networks (GANs) [11]. Representative methods include CTGAN and TVAE [40]. GOGGLE [24] further enhances this paradigm by explicitly modeling column dependencies through a Graph Neural Network–augmented VAE architecture. However, these approaches often lack sufficient expressivity when confronted with complex tabular distributions involving intricate feature interactions.

Diffusion-based tabular generators. Motivated by the strong generative capacity of diffusion models [13], a growing body of work adapts diffusion processes for tabular data generation. Early methods model numerical and categorical features using separate discrete-time diffusion [4], as in Kotelnikov et al. [21], Lee et al. [23], but discretization can lead to looser ELBO bounds and suboptimal generation quality [36, 19]. More recent approaches encode tabular features into continuous latent spaces and apply Gaussian diffusion [45, 44]. However, such latent modeling introduces additional encoding overhead and may only indirectly capture heterogeneous feature interactions. Recently, CDTD [25] explores feature-wise noise schedules under continuous diffusion, and TabDiff [33] further extends it to mixed-type feature-level diffusion. More recently, TabNAT [43] combines continuous diffusion for numerical features with a bidirectional masked Transformer over column-specific categorical indices, but its discrete generation is restricted to fixed category sets and does not support free-form text. In contrast, TABDLM leverages a pretrained MDLM with a large semantic vocabulary to jointly generate numerical, categorical, and free-form textual fields within a single model.

Language model-based tabular generators. Recent advances in large language models have also inspired language model–based approaches for tabular data generation. These methods serialize each row into a text sequence and fine-tune an autoregressive language model to capture row-level distributions, as exemplified by GReaT [6] with a GPT-2 [30] backbone. DiffLM [46] is the closest to our work, as it integrates VAEs and latent diffusion within a language modeling framework. However, DiffLM applies continuous diffusion in a latent space derived from textual representations, which may lose fine-grained token-level information and limit its applicability to free-form text fields. To the best of our knowledge, TABDLM is the first approach to apply masked language diffusion to explicitly model textual features at the token level, enabling faithful generation of free-form text in tabular data.

4 Experiments

In this section, we conduct extensive experiments to evaluate the proposed TABDLM. Specifically, we aim to answer the following questions. **Q1:** Can TABDLM effectively model the mutual dependencies among numerical, categorical, and free-form text columns? **Q2:** How well does TABDLM perform on real-world tabular data generation tasks involving free-form text columns? **Q3:** Can TABDLM achieve performance on par with existing methods on tasks that do not include textual columns? We implement TABDLM based on LLaDA-8B [26] for all experiments. Other implementation details can be found in Appendix B.

4.1 Results on synthetic tabular datasets

In this section, we answer the question **Q1** through two carefully designed synthetic tabular datasets.

Datasets. Existing real-world tabular generation datasets predominantly focus on numerical and categorical features and lack free-form text fields. Thus, we construct two synthetic tabular datasets containing numerical, categorical, and free-form text columns: **MathExpr** and **ProfileBio**. MathExpr focuses on mathematical expressions. Each sample includes two floating-point variables, categorical features indicating the unary and binary operators applied to them, and a textual column containing the corresponding LaTeX expression. ProfileBio contains numerical and categorical attributes describing a person's demographic and educational background, along with a textual biography generated from the other columns. For both datasets, accurate generation requires models to **capture correlations between numerical, categorical, and textual columns**. We show an example of both datasets in Table 5 and Table 6, respectively, and leave the detailed dataset generation process in Appendix A.1.

Baselines. Given these two datasets contain free-form textual features, we compare the proposed TABDLM with the autoregressive LLM and Masked Diffusion LLM. Specifically, we include Qwen2.5 (7B/14B) with 3-shot in-context learning (ICL), and Qwen2.5-7B and LLaDA-8B with LoRA-based supervised fine-tuning (SFT).

Metrics. We evaluate generated data along two groups. **1) Fidelity:** Shape and Trend measure marginal column-wise similarity and pairwise dependencies among numerical and categorical columns, both reported as error rates; **2) Cross-field consistency:** for MathExpr, we report Operation Match Rate (Op-MR) and Expression Match Rate (Exp-MR), measuring whether operators and numerical literals in the generated LaTeX align with the structured fields. For ProfileBio, we report Biography Match Rate (Bio-MR), checking consistency between the generated biography and structured attributes. Detailed definitions are provided in Appendix A.2.

Table 1: Evaluation results on the **MathExpr** and **ProfileBio** datasets (%).

Method	MathExpr				ProfileBio		
	Shape ↓	Trend ↓	Op-MR ↑	Exp-MR ↑	Shape ↓	Trend ↓	Bio-MR ↑
Qwen2.5-7B$_{\text{ICL}}$	39.04	56.65	53.93	53.92	41.29	64.52	7.97
Qwen2.5-14B$_{\text{ICL}}$	25.61	36.07	76.01	75.12	43.97	65.00	13.00
Qwen2.5-7B$_{\text{SFT}}$	5.07	50.81	99.68	99.34	10.51	36.26	98.51
LLaDA-8B$_{\text{SFT}}$	5.41	42.54	98.36	97.90	5.81	44.14	97.63
TABDLM	**3.69**	**6.07**	**99.99**	**99.80**	**4.84**	**7.81**	**98.60**

Results. As shown in Table 1, TABDLM achieves the best overall performance on both MathExpr and ProfileBio. On MathExpr, TABDLM obtains the lowest Shape and Trend errors, outperforming the strongest baseline by 27.1% and 83.2%, respectively, while also achieving the best results for both types of Match Rate. Since MathExpr requires the generated LaTeX expression to be consistent with both numerical values and categorical operation descriptors, these results indicate that TABDLM can preserve explicit dependencies across numerical, categorical, and textual fields rather than merely matching marginal distributions. On ProfileBio, TABDLM also achieves the lowest Shape and Trend errors and the highest Match Rate, demonstrating its ability to generate biographies that remain semantically aligned with the corresponding structured attributes. Together, the strong fidelity and cross-field consistency results provide a direct answer to **Q1**: TABDLM can effectively model mutual dependencies among heterogeneous tabular fields through joint numerical-language denoising.

4.2 Results on real-world tabular datasets with free-form text features

In this section, we evaluate TABDLM on real-world tabular datasets with free-text to answer **Q2**.

Dataset and Baselines. For this section, we use two real-world datasets, Amazon and Arxiv, both derived from the RelBench [31] relational benchmark. Amazon is constructed from `rel-amazon` by converting multiple relational tables into a single heterogeneous table containing numerical attributes, categorical attributes, and multiple long-text fields (e.g., title, description, and review). Arxiv is constructed from `rel-arxiv` as a paper-level table containing both numerical and categorical

attributes, and free-form text fields (e.g., title, arxiv code, and abstract). We provide more details on dataset generation in Appendix A.1. We use the same baseline as in Section 4.1.

Table 2: Results on the **Amazon** and **Arxiv** datasets. (* indicates *(w/o nums)*)

Method	Amazon				Arxiv			
	Shape (%)↓	Trend (%)↓	MLE* ↑	MLE ↑	Shape (%)↓	Trend (%)↓	MLE* ↑	MLE ↑
Real	0.0	0.0	.905	.906	0.0	0.0	.968	.968
Qwen2.5-7B$_{\text{ICL}}$	37.93	68.27	.636	.629	68.04	42.34	.521	.516
Qwen2.5-14B$_{\text{ICL}}$	39.19	78.22	.684	.679	40.79	34.35	.556	.527
Qwen2.5-7B$_{\text{SFT}}$	11.41	46.31	.824	.834	11.19	10.02	.701	.727
LLaDA-8B$_{\text{SFT}}$	7.76	37.13	.891	.892	14.10	43.81	.931	.930
TABDLM	**4.67**	**5.33**	**.893**	**.895**	**6.30**	**6.30**	**.946**	**.948**

Metrics. Similar to Section 4.1, we evaluate **1) Fidelity:** Shape and Trend. Additionally, we include **2) Downstream utility:** Machine Learning Efficiency (MLE), which measures how well predictive models trained on synthetic data generalize to real test data. Finally, to evaluate cross-field consistency between free-form text and the remaining numerical and categorical features, we follow an MLE-like protocol: we first serialize all non-numerical fields into structured text and encode them with the sentence embedding model Nomic [27], then train an XGBoost Classifier [7] using either (i) text embeddings only or (ii) text embeddings concatenated with numerical features, which isolates the contribution of generated text/categorical fields. For the Amazon and Arxiv dataset, we report AUC for MLE task.

Results. The results on Amazon and Arxiv are presented in Table 2. TABDLM consistently achieves the best performance across Shape, Trend, and MLE on both datasets. Specifically, compared to the strongest baseline, TABDLM reduces Shape Error by 39.8% and 43.7%, and Trend Error by 85.6% and 37.1%, on Amazon and Arxiv, respectively. These results show that TABDLM effectively preserves structured-field marginal distributions and pairwise structured dependencies in real-world datasets that also contain free-form textual fields. Regarding downstream utility, TABDLM achieves the best MLE on both datasets, closely approaching the upper bound set by real data. This indicates that the generated text, categorical, and numerical fields jointly preserve label-relevant information. Notably, when numerical attributes are excluded, TABDLM's performance on *MLE (w/o nums)* exhibits a decline consistent with the trend observed in real data. This suggests that the synthetic numerical features are not merely reproducing marginal statistics, but are meaningfully coupled with target labels and other modalities, thereby providing substantive value for downstream predictive tasks. Together, these results provide a direct answer to **Q2**: TABDLM performs strongly on real-world tabular generation tasks involving free-form text, preserving both structured-field fidelity and downstream task-relevant information across modalities.

4.3 Results on real-world tabular datasets without free-form text features

Finally, we evaluate the performance of TABDLM on standard tabular data generation benchmarks without free-form text features, enabling direct comparison with existing methods. This evaluation addresses **Q3** by examining whether TABDLM retains strong modeling capacity on *traditional* tabular data, despite being designed for a significantly broader heterogeneous generation setting.

Datasets, baselines, and metrics. We conduct experiments on five real-world tabular datasets: Adult, Default, Shoppers, Magic, and Beijing. Detailed dataset profiles are presented in Appendix A.1.5. For baselines, we compare TABDLM with some widely-used synthetic tabular data generation methods from four categories: 1) GAN-based: CTGAN [40]; 2) VAE-based: TVAE [40] and GOGGLE [24]; 3) Autoregressive Tabular Language Model: GReaT [6] and DiffLM [46]; 4) Diffusion-based: STaSy [17], CoDi [23], TabDDPM [21], TabSyn [44], TabDiff [33], and TabNAT [43]. We evaluate along three groups of metrics: **1) Fidelity:** Shape, Trend, α-Precision, and C2ST assess how faithfully synthetic data recovers the ground-truth distribution; **2) Downstream utility:** Machine Learning Efficiency (MLE) measures the usefulness of synthetic data for predictive tasks; **3) Privacy:** Distance to Closest Record (DCR) evaluates privacy risk by measuring the distance between each synthetic sample and its nearest training sample, where overly small distances may indicate memorization. We report Shape, Trend in the main paper and defer α-Precision, C2ST, MLE, and DCR results to Appendix C.

Table 3: Performance comparison on the error rates (%) of **Shape** ↓ / **Trend** ↓. Each cell reports Shape / Trend. Full per-metric results are provided in Appendix C.1. (**B**: best overall; **B**: best in language-based model)

Method	Adult	Default	Shoppers	Magic	Beijing	Average
CTGAN	16.84 / 20.23	16.83 / 26.95	21.15 / 13.08	9.81 / 7.00	21.39 / 22.95	17.20 / 18.04
TVAE	14.22 / 14.15	10.17 / 19.50	24.51 / 18.67	8.25 / 5.82	19.16 / 18.01	15.26 / 15.23
GOGGLE	16.97 / 45.29	17.02 / 21.94	22.33 / 23.90	1.90 / 9.47	16.93 / 45.94	15.03 / 29.31
STaSy	11.29 / 14.51	5.77 / 5.96	9.37 / 8.49	6.29 / 6.61	6.71 / 8.00	7.89 / 8.71
CoDi	21.38 / 22.49	15.77 / 68.41	31.84 / 17.78	11.56 / 6.53	16.94 / 7.07	19.50 / 24.46
TabDDPM	1.75 / 3.01	1.57 / 4.89	2.72 / 6.61	1.01 / 1.70	1.30 / 2.71	1.67 / 3.78
TabSyn	0.81 / 1.93	1.01 / 2.81	1.44 / 2.13	1.03 / 0.88	1.26 / 3.13	1.11 / 2.18
TabDiff	**0.63** / **1.49**	1.24 / 2.55	1.28 / 1.74	0.78 / **0.76**	1.03 / 2.59	0.99 / **1.83**
TabNAT	0.73 / 1.61	**0.80** / **2.21**	**1.11** / **1.67**	**0.62** / 1.37	**0.79** / **2.48**	**0.81** / 1.87
GReaT	12.12 / 17.59	19.94 / 70.02	14.51 / 45.16	16.16 / 10.23	8.25 / 59.60	14.20 / 40.52
DiffLM	9.74 / –	9.06 / –	10.07 / –	7.53 / –	6.35 / –	8.55 / –
Qwen2.5-7B$_{\text{ICL}}$	27.80 / 37.61	22.39 / 31.02	34.37 / 34.77	13.41 / 20.62	31.83 / 40.83	25.96 / 32.97
Qwen2.5-14B$_{\text{ICL}}$	25.17 / 42.67	21.52 / 28.72	51.21 / 42.32	17.51 / 18.11	29.76 / 38.20	29.03 / 34.00
Qwen2.5-7B$_{\text{SFT}}$	5.11 / 17.03	2.58 / 16.99	8.76 / 10.28	7.53 / 14.45	7.49 / 28.68	6.29 / 17.49
LLaDA-8B$_{\text{SFT}}$	1.69 / 26.15	2.00 / 26.30	13.56 / 16.42	5.60 / 13.71	3.64 / 27.90	5.30 / 22.10
TabDLM	**1.46** / **2.74**	**1.18** / **2.33**	**2.05** / **2.40**	**2.93** / **2.85**	**2.42** / **2.93**	**2.01** / **2.65**

Results. As reported in Table 3, TabDLM performs competitively with state-of-the-art tabular diffusion models on Shape and Trend. **This is notable because such tabular-specific synthetic methods operate in a relatively restricted feature space** (e.g., categorical columns are represented via one-hot vectors or limited discrete states), whereas TabDLM is built to jointly model heterogeneous fields and ultimately support open-ended text generation, yet it still remains strong on purely numerical and categorical datasets. Beyond this, TabDLM consistently and substantially outperforms all language-based baselines, with average gains of 62.1% in Shape and 84.8% in Trend. Notably, **language-based generators often achieve reasonable Shape but markedly worse Trend**. A likely reason is that row serialization simplifies matching column-wise marginals, as each field can be generated locally to fit its token-level distribution. However, Trend depends on modeling **joint** dependencies among numerical and categorical columns, which is particularly difficult to preserve under autoregressive language backbones. In practice, numerical values are represented as subword token sequences with precision-sensitive semantics, whereas categorical fields act as discrete identifiers, making consistent numerical–categorical correlations hard to preserve under autoregressive generation. In contrast, TabDLM jointly generates numerical and categorical features within a unified diffusion process and leverages bidirectional interactions across columns, enabling more direct modeling of tabular dependencies than left-to-right row generation and yielding greater fidelity and downstream utility on structured datasets.

4.4 Ablation Study

We conduct an ablation study to assess the contribution of each component in TabDLM. We consider three variants: (i) TabDLM-noFloatAE, which replaces the pretrained float encoder/decoder with randomly initialized counterparts; (ii) TabDLM-onlyContDiff, which removes MDLM-based modeling for categorical fields; and (iii) TabDLM-onlyMDLM, which removes the continuous diffusion branch and processes numerical features as serialized tokens. Detailed variant descriptions are provided in Appendix C.6.

Table 4: Ablation study on the average performance across Adult, Default, and Shoppers

Method	Shape ↓	Trend ↓	MLE ↑	C2ST ↑	α-Precision ↑
TabDLM-noFloatAE	2.15	4.34	.870	0.9256	98.50
TabDLM-onlyContDiff	4.19	7.28	.850	0.8399	91.70
TabDLM-onlyMDLM	5.75	22.96	.852	0.7892	88.58
TabDLM	**1.56**	**2.49**	**.874**	**.9571**	**98.56**

The full TabDLM consistently outperforms all three variants. The largest drop occurs in TabDLM-onlyMDLM, supporting our motivation in Section 1 that subword tokenization fragments precision-sensitive numerical values and hampers fine-grained numerical–categorical correlation modeling.

TABDLM-noFloatAE and TABDLM-onlyContDiff also degrade noticeably, validating the necessity of pretrained numerical encoding and MDLM-based modeling for categorical fields, respectively.

5 Conclusion

In this work, we propose TABDLM, the first unified framework that can generate high-fidelity synthetic tabular data with both numeric, categorical, and free-form text features. TABDLM leverages MDLM with special numerical tokenization to allow a single model to perform joint numerical-language diffusion. Extensive experiments validate the effectiveness of TABDLM over baseline models. We discuss limitations and future directions, including sampling efficiency and modality-specific noise schedules, in Appendix D.

References

[1] Ahmed Alaa, Boris Van Breugel, Evgeny S Saveliev, and Mihaela Van Der Schaar. 2022. How faithful is your synthetic data? sample-level metrics for evaluating and auditing generative models. In *International conference on machine learning*. PMLR, 290–306.

[2] Marianne Arriola, Aaron Gokaslan, Justin T Chiu, Zhihan Yang, Zhixuan Qi, Jiaqi Han, Subham Sekhar Sahoo, and Volodymyr Kuleshov. 2025. Block diffusion: Interpolating between autoregressive and diffusion language models. *arXiv preprint arXiv:2503.09573* (2025).

[3] Samuel A Assefa, Danial Dervovic, Mahmoud Mahfouz, Robert E Tillman, Prashant Reddy, and Manuela Veloso. 2020. Generating synthetic data in finance: opportunities, challenges and pitfalls. In *Proceedings of the First ACM International Conference on AI in Finance*. 1–8.

[4] Jacob Austin, Daniel D Johnson, Jonathan Ho, Daniel Tarlow, and Rianne Van Den Berg. 2021. Structured denoising diffusion models in discrete state-spaces. *Advances in neural information processing systems* 34 (2021), 17981–17993.

[5] Benjamin M Bolstad, Rafael A Irizarry, Magnus Åstrand, and Terence P. Speed. 2003. A comparison of normalization methods for high density oligonucleotide array data based on variance and bias. *Bioinformatics* 19, 2 (2003), 185–193.

[6] Vadim Borisov, Kathrin Sessler, Tobias Leemann, Martin Pawelczyk, and Gjergji Kasneci. 2023. Language Models are Realistic Tabular Data Generators. In *The Eleventh International Conference on Learning Representations*.

[7] Tianqi Chen and Carlos Guestrin. 2016. XGBoost: A Scalable Tree Boosting System. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2016).

[8] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*. 4171–4186.

[9] Stefan Elfwing, Eiji Uchibe, and Kenji Doya. 2018. Sigmoid-weighted linear units for neural network function approximation in reinforcement learning. *Neural networks* 107 (2018), 3–11.

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

[11] Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. *Advances in neural information processing systems* 27 (2014).

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

[13] Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising diffusion probabilistic models. *Advances in neural information processing systems* 33 (2020), 6840–6851.

[14] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. 2022. Lora: Low-rank adaptation of large language models. *ICLR* 1, 2 (2022), 3.

[15] Alistair EW Johnson, Lucas Bulgarelli, Lu Shen, Alvin Gayles, Ayad Shammout, Steven Horng, Tom J Pollard, Sicheng Hao, Benjamin Moody, Brian Gow, et al. 2023. MIMIC-IV, a freely accessible electronic health record dataset. *Scientific data* 10, 1 (2023), 1.

[16] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. 2022. Elucidating the design space of diffusion-based generative models. *Advances in neural information processing systems* 35 (2022), 26565–26577.

[17] Jayoung Kim, Chaejeong Lee, and Noseong Park. 2023. STaSy: Score-based Tabular data Synthesis. In *The Eleventh International Conference on Learning Representations*.

[18] Jayoung Kim, Chaejeong Lee, Yehjin Shin, Sewon Park, Minjung Kim, Noseong Park, and Jihoon Cho. 2022. Sos: Score-based oversampling for tabular data. In *Proceedings of the 28th ACM SIGKDD conference on knowledge discovery and data mining*. 762–772.

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

[20] Diederik P Kingma and Max Welling. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114* (2013).

[21] Akim Kotelnikov, Dmitry Baranchuk, Ivan Rubachev, and Artem Babenko. 2023. Tabddpm: Modelling tabular data with diffusion models. In *International conference on machine learning*. PMLR, 17564–17579.

[22] Himabindu Lakkaraju, Julian McAuley, and Jure Leskovec. 2013. What's in a name? understanding the interplay between titles, content, and communities in social media. In *Proceedings of the international AAAI conference on web and social media*, Vol. 7. 311–320.

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

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

[25] Markus Mueller, Kathrin Gruber, and Dennis Fok. 2023. Continuous diffusion for mixed-type tabular data. *arXiv preprint arXiv:2312.10431* (2023).

[26] Shen Nie, Fengqi Zhu, Zebin You, Xiaolu Zhang, Jingyang Ou, Jun Hu, Jun Zhou, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. 2025. Large language diffusion models. *arXiv preprint arXiv:2502.09992* (2025).

[27] Zach Nussbaum, John Xavier Morris, Andriy Mulyar, and Brandon Duderstadt. 2025. Nomic Embed: Training a Reproducible Long Context Text Embedder. *Transactions on Machine Learning Research* (2025). Reproducibility Certification.

[28] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems* 32 (2019).

[29] William Peebles and Saining Xie. 2023. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*. 4195–4205.

[30] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog* 1, 8 (2019), 9.

[31] Joshua Robinson, Rishabh Ranjan, Weihua Hu, Kexin Huang, Jiaqi Han, Alejandro Dobles, Matthias Fey, Jan Eric Lenssen, Yiwen Yuan, Zecheng Zhang, et al. 2024. Relbench: A benchmark for deep learning on relational databases. *Advances in Neural Information Processing Systems* 37 (2024), 21330–21341.

[32] Timur Sattarov, Marco Schreyer, and Damian Borth. 2023. Findiff: Diffusion models for financial tabular data generation. In *Proceedings of the Fourth ACM International Conference on AI in Finance*. 64–72.

[33] Juntong Shi, Minkai Xu, Harper Hua, Hengrui Zhang, Stefano Ermon, and Jure Leskovec. 2025. TabDiff: a Mixed-type Diffusion Model for Tabular Data Generation. In *The Thirteenth International Conference on Learning Representations*.

[34] Jiaming Song, Chenlin Meng, and Stefano Ermon. 2020. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502* (2020).

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

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

[37] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).

[38] Yanbo Wang, Xiyuan Wang, Quan Gan, Minjie Wang, Qibin Yang, David Wipf, and Muhan Zhang. 2025. Griffin: Towards a Graph-Centric Relational Database Foundation Model. *arXiv preprint arXiv:2505.05568* (2025).

[39] Chengyue Wu, Hao Zhang, Shuchen Xue, Zhijian Liu, Shizhe Diao, Ligeng Zhu, Ping Luo, Song Han, and Enze Xie. 2025. Fast-dllm: Training-free acceleration of diffusion llm by enabling kv cache and parallel decoding. *arXiv preprint arXiv:2505.22618* (2025).

[40] Lei Xu, Maria Skoularidou, Alfredo Cuesta-Infante, and Kalyan Veeramachaneni. 2019. Modeling tabular data using conditional gan. *Advances in neural information processing systems* 32 (2019).

[41] Haotong Yang, Yi Hu, Shijia Kang, Zhouchen Lin, and Muhan Zhang. 2024. Number cookbook: Number understanding of language models and how to improve it. *arXiv preprint arXiv:2411.03766* (2024).

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

[43] Hengrui Zhang, Liancheng Fang, Qitian Wu, and Philip S. Yu. 2025. TabNAT: A Continuous-Discrete Joint Generative Framework for Tabular Data. In *International Conference on Machine Learning*.

[44] Hengrui Zhang, Jiani Zhang, Zhengyuan Shen, Balasubramaniam Srinivasan, Xiao Qin, Christos Faloutsos, Huzefa Rangwala, and George Karypis. 2024. Mixed-Type Tabular Data Synthesis with Score-based Diffusion in Latent Space. In *The Twelfth International Conference on Learning Representations*.

[45] S Zheng and N Charoenphakdee. 2022. Diffusion models for missing value imputation in tabular data. arXiv. *arXiv preprint arXiv:2210.17128* (2022).

[46] Ying Zhou, Xinyao Wang, Yulei Niu, Yaojie Shen, Lexin Tang, Fan Chen, Ben He, Le Sun, and Longyin Wen. 2025. Difflm: Controllable synthetic data generation via diffusion language models. In *Findings of the Association for Computational Linguistics: ACL 2025*. 20638–20658.

A Detailed Experiment Setups

A.1 Datasets

A.1.1 MathExpr

MathExpr is a synthetic dataset designed to evaluate the joint generation of heterogeneous tabular records involving numerical values, categorical operators, and free-form LaTeX expressions grounded in the structured columns. Each record contains two numerical columns, three categorical operator columns, and one text column:

$$\mathbf{r} = (x_1, x_2, o_1, o_2, o_3, e_{\texttt{latex}}).$$

Here, o_1 and o_2 are unary operators applied to x_1 and x_2, respectively, while o_3 is a binary operator that combines the two transformed terms. The text column $e_{\texttt{latex}}$ is a LaTeX expression deterministically constructed from the preceding structured columns.

We generate 10,000 records and split them into a training/real set and a validation set with a ratio of 9:1. During sampling, each model generates 9,000 synthetic samples, and these synthetic samples are then used to compute the distribution fidelity metrics and expression consistency metrics described in Appendix A.2.6.

Numerical value sampling. We sample x_1 and x_2 from discrete supports with step size 0.1:

$$x_1 \in \{0.1, 0.2, \ldots, 6.0\}, \qquad x_2 \in \{3.0, 3.1, \ldots, 9.9\}.$$

To induce a non-uniform yet diverse distribution, we sample x_1 and x_2 from an equal-weight mixture of a discrete Gaussian distribution and a uniform distribution over the corresponding support. The discrete Gaussian components are centered at $\mu_1 = 3$ and $\mu_2 = 6.5$, respectively, with shared standard deviation $\sigma = 1.5$.

Categorical operator sampling. The unary operators o_1 and o_2 are sampled independently from

$$\mathcal{O}_{\text{unary}} = \{\texttt{none}, \texttt{log}, \texttt{exp}, \texttt{sqrt}, \texttt{sin}, \texttt{cos}, \texttt{tan}, \texttt{square}, \texttt{cube}\}.$$

Their categorical priors are:

$$p(o_1) = \{\texttt{none} : 0.18,\ \texttt{log} : 0.16,\ \texttt{sqrt} : 0.13,\ \texttt{square} : 0.12,\ \texttt{sin} : 0.10,\ \texttt{cos} : 0.10,\ \texttt{tan} : 0.07,\ \texttt{exp} : 0.07,\ \texttt{cube} : 0.07\},$$

$$p(o_2) = \{\texttt{none} : 0.22,\ \texttt{sin} : 0.14,\ \texttt{cos} : 0.14,\ \texttt{sqrt} : 0.12,\ \texttt{log} : 0.10,\ \texttt{square} : 0.09,\ \texttt{tan} : 0.07,\ \texttt{exp} : 0.06,\ \texttt{cube} : 0.06\}.$$

The binary operator o_3 is sampled from

$$\mathcal{O}_{\text{binary}} = \{\texttt{add}, \texttt{sub}, \texttt{mul}, \texttt{div}\},$$

with categorical prior

$$p(o_3) = \{\texttt{add} : 0.35,\ \texttt{mul} : 0.30,\ \texttt{sub} : 0.20,\ \texttt{div} : 0.15\}.$$

Free-form LaTeX expression construction. The expression string $e_{\texttt{latex}}$ is deterministically generated using a fixed grammar: unary operators are rendered as standard LaTeX commands (e.g., $\texttt{log} \mapsto \backslash\texttt{log}(\cdot)$, $\texttt{sqrt} \mapsto \backslash\texttt{sqrt}\{\cdot\}$, $\texttt{square} \mapsto (\cdot)^2$). Binary operators are rendered as $+$, $-$, $\backslash\texttt{times}$, or $\backslash\texttt{frac}\{\cdot\}\{\cdot\}$.

A.1.2 ProfileBio

ProfileBio is a synthetic dataset for evaluating the joint generation of mixed-type personal profiles with a free-form biography text grounded in structured columns. Each record contains two numerical columns, five categorical columns, and one text column:

$$\mathbf{r} = (\texttt{age}, \texttt{salary}, \texttt{sex}, \texttt{birth\_state}, \texttt{college}, \texttt{degree}, \texttt{occupation}, \texttt{biography}).$$

The `biography` field is a natural-language paragraph deterministically constructed from the structured attributes.

Following the setup of MathExpr, we generate 10,000 records and split them into a training/real set and a validation set with a ratio of 9:1. During sampling, each model generates a synthetic dataset with the same cardinality as the real training set. We evaluate distribution fidelity and biography consistency using the metrics described in Appendix A.2.6.

Table 5: An example sample from the **MathExpr** dataset.

Column	Example Value
x_1	2.75
x_2	6.40
operation_x1	`sin`
operation_x2	`log`
operation_between	`mul`
latex_expression	`\sin(2.75) \times \log(6.40)`

Table 6: An example sample from the **ProfileBio** dataset.

Column	Example Value
age	38
salary	135
sex	`female`
birth_state	`California`
college	`Harvard University`
degree	`master`
occupation	`software developer`
biography	`This female individual is in the career-building stage. She was born in California and completed higher education at Harvard University, earning a master degree. She works as a software developer. She earns a strong professional income.`

Categorical value sampling. We sample `sex` uniformly from {`male`, `female`}. The attributes `birth_state` and `college` are sampled from fixed categorical priors over 10 states and 9 colleges, respectively. The `degree` attribute is sampled conditionally on `college`: for `stanford university` and `harvard university`, we use

$$p(\texttt{degree}) = (0.01, 0.29, 0.40, 0.30),$$

and for all other colleges, we use

$$p(\texttt{degree}) = (0.30, 0.50, 0.15, 0.05),$$

where the entries correspond to (`associate`, `bachelor`, `master`, `doctoral`). Finally, `occupation` is sampled from a degree-dependent categorical distribution. All occupation weights are initialized to 1. For `doctoral` degrees, the weights of `research specialist` and `education professional` are set to 6 and 4, respectively; for `associate` degrees, the weights of `customer services professional` and `construction professional` are set to 5 and 5, respectively. The weights are then normalized into a categorical distribution.

Numerical value sampling. We sample `age` uniformly from integers in $[21, 70]$, and draw `salary` from a Gaussian whose mean depends on `degree`, `occupation`, and `age`. The degree-dependent base mean is

$$\mu_0(\texttt{degree}) = \begin{cases} 82, & \texttt{associate}, \\ 125, & \texttt{bachelor or master}, \\ 178, & \texttt{doctoral}, \end{cases}$$

and, letting $\mathcal{O}^\star = \{\texttt{software developer}, \texttt{healthcare practitioner}\}$, the full mean is

$$\mu \;=\; \mu_0(\texttt{degree}) + 4 \cdot \mathbb{1}\{\texttt{occupation} \in \mathcal{O}^\star\} + 0.3\,(\texttt{age} - 45).$$

We then sample $\texttt{salary} \sim \mathcal{N}(\mu,\, 5^2)$, round it to the nearest integer, and clip it to $[75, 200]$.

Biography construction. The `biography` field is deterministically constructed from the structured attributes using a fixed template. The `age` and `salary` values are first mapped to coarse-grained textual descriptors, and Table 7 summarizes both the descriptor mappings and the biography template.

Additional clarification. ProfileBio is a fully synthetic dataset constructed from predefined sampling rules and deterministic templates. It does not rely on, copy, or imitate any real individuals. Sensitive personal attributes such as race, religion, health status, or immigration background are

Table 7: **ProfileBio:** Age/salary mapping rules and the biography template.

Component	Rule / Template
Age descriptor	[21, 30]: `in the early career stage` [31, 40]: `in the career-building stage` [41, 50]: `in the established career stage` [51, 60]: `in the advanced career stage` [61, 70]: `in the late career stage`
Salary descriptor	[75, 100]: `a comfortable income` [101, 150]: `a strong professional income` [151, 200]: `a high-level income`
Biography template	`This {sex} individual is {age_desc}. {He/She} was born in {birth_state} and completed higher education at {college}, earning a {degree} degree. {He/She} works as a {occupation}. {He/She} earns {salary_desc}.`

Table 8: An example sample from the **Amazon** dataset.

Column	Example Value
price	5.98
review_time	1970 (days since earliest review date)
rating	5.0
verified	true
category	Science Fiction & Fantasy > Fantasy
brand	Visit Amazon's J. R. R. Tolkien Page
title	Hobbit
description	The enchanting prelude to "The Lord of the Rings"
review_text	My 13 yr. old grandson was very happy with this book. He likes to know the background of things and this helped him to understand this story.
summary	Grandson happy

intentionally excluded. While attributes such as `sex`, `birth_state`, and `occupation` are included to evaluate cross-field consistency, the salary generation process is explicitly designed to depend only on `age`, `degree`, and `occupation`, and does not condition on `sex` or `birth_state`. ProfileBio is intended solely as a controlled benchmark for evaluating joint generation fidelity and attribute–text consistency, rather than as a model of real-world socioeconomic distributions.

A.1.3 Amazon

Amazon is a real-world free-text tabular dataset constructed from the RelBench [31] `rel-amazon` relational benchmark, which contains linked product metadata and user reviews. We join the `review` and `product` tables on `product_id` and form a single mixed-type table with two numerical columns, two categorical columns, and six free-form text columns:

$$(\texttt{price}, \texttt{review\_time}, \texttt{rating}, \texttt{verified}, \texttt{category}, \texttt{brand}, \texttt{title}, \texttt{description}, \texttt{review\_text}, \texttt{summary}).$$

We sample $5{,}000$ examples in total and split them into a training/real set and a validation set with a ratio of 9:1. In addition, we sample another test set of $2{,}250$ examples for downstream utility evaluation. During sampling, each model generates $4{,}500$ synthetic examples, matching the cardinality of the real training set. We report distribution fidelity and downstream utility metrics as defined in Appendix A.2.6. Table 8 shows an example record.

A.1.4 Arxiv

Arxiv is another real-world free-text tabular dataset constructed from the RelBench `rel-arxiv` relational benchmark, which contains metadata and abstracts of arXiv papers. We construct a single mixed-type table at the paper level with four numerical columns, one categorical column, and three

Table 9: An example record from the **Arxiv** dataset.

Column	Example Value
submission_time	1672 (days since earliest review date)
submission_year	2022
title_length	7
abstract_length	90
category	Category_3
title	IRC-safe Graph Autoencoder for unsupervised anomaly detection
arxiv_code	arXiv:2204.12231
abstract	Anomaly detection through employing machine learning techniques has emerged as a novel powerful tool in the search for new physics beyond the Standard Model. Historically similar to the development of jet observables, theoretical consistency has not always assumed a central role in the fast development of algorithms and neural network architectures. In this work, we construct an infrared and collinear safe autoencoder based on graph neural networks by employing energy-weighted message passing. We demonstrate that whilst this approach has theoretically favourable properties, it also exhibits formidable sensitivity to non-QCD structures.

Table 10: Statistics of real-world tabular datasets. #Num denotes the number of numerical columns, and #Cat denotes the number of categorical columns. #Max Cat is the maximum number of categories among all categorical columns.

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

free-form text columns:

$$(\texttt{submission\_time}, \texttt{submission\_year}, \texttt{title\_length}, \texttt{abstract\_length}, \texttt{category}, \texttt{title}, \texttt{arxiv\_code}, \texttt{abstract}).$$

We sample $4{,}500$ examples for training and validation and split them into a real training set and a validation set with a ratio of 8:1. In addition, we sample another test set of $2{,}000$ examples for downstream utility evaluation. During sampling, each model generates a synthetic dataset with the same cardinality as the real training set. We report distribution fidelity and downstream utility metrics as defined in Appendix A.2.6. Table 9 shows an example record.

A.1.5 Real-world Tabular Datasets

We evaluate on five widely-used real-world tabular datasets from the UCI Machine Learning Repository.[1] These datasets cover both classification and regression tasks, providing a diverse testbed for regular tabular generation with only numerical and categorical features. Specifically, Adult, Default, Shoppers, and Magic are used for classification, while Beijing is used for regression. Dataset statistics and the corresponding train/validation/test splits are summarized in Table 10.

A.2 Metrics

A.2.1 Shape and Trend

We evaluate distribution fidelity on all datasets using two general-purpose metrics: **Shape** and **Trend**. Shape and Trend are adopted from SDMetrics[2], which quantify the marginal column-wise similarity and the pairwise dependency preservation between real and synthetic data, respectively.

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

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

Shape. *Kolmogorov-Smirnov Test (KST):* This metric quantifies the alignment between the real distribution $p_r(x)$ and the synthetic distribution $p_s(x)$ by calculating the maximum divergence between their respective Cumulative Distribution Functions (CDFs):

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

where $F_r(x)$ and $F_s(x)$ denote the CDFs derived from the probability densities:

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

Total Variation Distance (TVD): For categorical variables, we evaluate the discrepancy in probability mass using the TVD. It is defined as half the sum of the absolute differences between the category frequencies observed in the real data, $R(\omega)$, and the synthetic data, $S(\omega)$:

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

where Ω represents the set of all possible categories within a given column.

Trend. *Pearson Correlation Score:* We examine the preservation of linear dependencies between continuous columns using the Pearson correlation coefficient, $\rho_{x,y}$, defined as:

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

where Cov represents covariance and σ denotes the standard deviation. To evaluate the overall preservation of these trends, we compute the Pearson Score as the normalized average absolute error between the correlation matrices of the real (ρ^R) and synthetic (ρ^S) datasets:

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

Since $\rho \in [-1, 1]$, the factor of $1/2$ normalizes the score to the range $[0, 1]$, where a lower score indicates superior correlation preservation.

Contingency Similarity: To measure the consistency of pairwise associations between categorical columns A and B, we utilize a metric based on the Total Variation Distance applied to contingency tables. The Contingency Score is calculated as:

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

where $R_{\alpha,\beta}$ and $S_{\alpha,\beta}$ correspond to the joint frequencies of category pair (α, β) in the real and synthetic datasets, respectively.

A.2.2 α-Precision

Following Alaa et al. [1], α-Precision is a sample-level fidelity metric that measures whether each synthetic sample falls within the α-support of the real data distribution, i.e., the smallest region containing an α fraction of the real probability mass. Intuitively, it quantifies how typical synthetic samples are with respect to the real distribution rather than lying in low-density or out-of-distribution regions. A higher α-Precision indicates higher fidelity of the generated samples.

A.2.3 Detection Score

The Detection Score is computed via the Classifier Two-Sample Test (C2ST) implemented in SDMetrics, where a logistic regression classifier is trained to distinguish real samples from synthetic ones. The score is derived from the classifier's misclassification rate: when the synthetic distribution closely matches the real one, the classifier performs near chance level, yielding a higher Detection Score. We report the Detection Score in $[0, 1]$, with higher values indicating that synthetic samples are statistically indistinguishable from real samples.

A.2.4 Machine Learning Efficiency

For datasets with an associated downstream prediction task, we additionally report Machine Learning Efficiency (MLE) to assess utility via the test-performance gap between models trained on real versus synthetic samples. In particular, we apply MLE to the regular real-world tabular datasets. During evaluation, we train the XGBoost classifier on the real training set (further split with an 8:1 ratio for validation and hyperparameter tuning) and evaluate it on a held-out real test set. We then train an identical classifier on the synthetic dataset and evaluate it on the same real test set. The MLE score is defined by the divergence between the two test performances, reflecting how well synthetic data can serve as a substitute for real data in downstream predictive modeling.

A.2.5 Distance Closest Record (DCR)

DCR evaluates whether the generative model memorizes training samples rather than learning the underlying distribution, serving as a data privacy metric. For each synthetic sample, we compute its distance to the nearest record in both the training set and a held-out test set, and report the proportion of synthetic samples whose nearest neighbor lies in the training set. A score close to 50% is ideal: it indicates that synthetic samples are equally close to training and test data, suggesting the model has captured the distribution rather than reproducing training instances.

A.2.6 Dataset-specific Metrics

MathExpr. Beyond standard distribution metrics (Shape and Trend), we evaluate whether the generated free-form LaTeX expression is structurally and numerically consistent with the structured number and operation columns. We report: (1) Operation Match Rate (Op-MR), which verifies if the unary/binary operator tokens implied by the expression $e_{\texttt{latex}}$ strictly align with the structured operations (o_1, o_2, o_3); and (2) Expression match rate (Exp-MR), which evaluates the joint validity of structure and values. Specifically, a generated expression is considered a match only if it (i) satisfies operation correctness (as in Op-MR) and (ii) contains two numeric literals whose values align with the structured fields (x_1, x_2) up to a small relative tolerance. Concretely, letting $\hat{x}_1, \hat{x}_2$ be the literals extracted from the generated LaTeX string, we require $|\hat{x}_i - x_i|/x_i \leq \delta$ for $i \in \{1, 2\}$. We fix $\delta = 0.07$ to make the metric robust to minor numeric drift that can arise from stochastic generation and continuous-value approximation (e.g., 0.29 vs. 0.30), while still penalizing outputs whose numeric content is meaningfully inconsistent with the structured inputs.

ProfileBio. For ProfileBio, we assess cross-modality consistency between structured attributes and the generated biography text. We compute a rule-based Match Rate by checking whether the biography instantiates the required template slots with values consistent with the corresponding structured fields (see the example template in Table 7). Concretely, for each record we verify that the text reflects core attributes (e.g., `sex`, `birth_state`, `college`, `degree`, `occupation`) as well as the derived descriptors for continuous variables (`Age` and `Salary`). Since mapping continuous values into discrete natural-language descriptors introduces semantic fuzziness near bin boundaries, we apply a small boundary relaxation tolerance $\delta = 0.06$ when validating the age/salary descriptors. This avoids rigid thresholding artifacts where values close to a boundary may legitimately share descriptions from neighboring categories; for instance, age 30.5 can reasonably be described as either "in the early career stage" or "in the career-building stage."

Amazon. For the Amazon dataset, we measure downstream utility via an MLE-like protocol, with `rating` as the target label in $\{1, 2, 3, 4, 5\}$ (treated as a multi-class classification task). We first serialize all non-numerical fields into structured text and encode them using the sentence embedding model Nomic. We then train an XGBoost classifier using either (i) text embeddings only or (ii) text embeddings concatenated with numerical features, which helps isolate the contribution of generated text/categorical fields. We report Macro-AUC on the held-out test set as the utility metric.

Arxiv. Similar to the Amazon dataset, we measure downstream utility via an MLE-like protocol, with `Category` as the target label (treated as a multi-class classification task). The original `rel-arxiv` benchmark contains more than 50 paper categories, which would induce severe class imbalance and weaken the signal of cross-field consistency; we therefore restrict our benchmark to samples from the 15 most frequent categories, yielding a relatively balanced 15-way classification task. We serialize all non-numerical fields into structured text and encode them using the sentence embedding model Nomic. We then train an XGBoost classifier using either (i) text embeddings only or (ii)

text embeddings concatenated with numerical features. For each configuration, Macro-AUC on the held-out test is reported as the utility metric.

B Implementation Details

We implement TABDLM based on PyTorch [28], the code is provided in an anonymous link `https://github.com/ilikevegetable/TabDLM`. All experiments are run on an NVIDIA A100 GPU with 80GB of memory.

Data preprocessing. We follow the same preprocessing method as prior diffusion-based tabular synthetic model [33]. Missing numerical values are imputed with the column mean, and missing categorical values are treated as an additional category. To stabilize optimization across heterogeneous numerical scales, we apply a quantile-based transformation to numerical columns during training and invert the transform after sampling to recover values in the original space.

Data splits. We adopt the same data split protocol as the TabDiff setting: each dataset is partitioned into a real set and a test set. Models are trained on the real set. For downstream utility evaluation, we further split the real set into training and validation subsets and reserve the test set strictly for evaluation.

Architecture. We build TABDLM on top of the LLaDA-8B base model for all experiments. To incorporate numerical channels, each scalar value is first mapped into a d-dimensional latent using a lightweight pretrained float encoder, implemented as a 3-layer MLP with hidden width $\lfloor\sqrt{d}\rfloor$ and SiLU activations [9]. A float decoder (LayerNorm + linear projection) maps the latent back to a scalar. We set the numerical latent dimension to $d = 512$ by default and keep the pretrained float encoder/decoder frozen during joint training. To interface numerical latents with the MDLM embedding space, we further apply a two-stage projection: an *input projector* that maps per-feature latents from d to the MDLM hidden size D using a 2-layer MLP ($d \to 1024 \to D$) with SiLU and dropout (with LayerNorm), and an *output projector* that maps MDLM hidden states back to the numerical latent space via a symmetric 2-layer MLP ($D \to 1024 \to d$). The dimension D in LLaDA-8B is 4096.

Hyperparameter settings. We fine-tune models with LoRA using the same configuration for TABDLM, LLaDA-8B$_{\text{SFT}}$, and Qwen2.5-7B$_{\text{SFT}}$ to ensure fair comparison. Unless otherwise stated, we use LoRA rank $r = 16$, scaling factor $\alpha = 32$, and dropout 0.05, and apply LORA to the attention and MLP components of Transformer blocks. Across methods, we keep the optimizer and training recipe identical; the only dataset-dependent choice is the number of training epochs due to varying dataset sizes. Specifically, we train for 10 epochs on Adult and Beijing, 30 epochs on Shoppers, 15 epochs on Magic and Default, and 75 epochs on MathExpr, Amazon, Arxiv, and ProfileBio. We use AdamW with learning rate 2×10^{-4}, warmup ratio 0.1, $(\beta_1, \beta_2) = (0.9, 0.98)$, weight decay 10^{-4}, and $\epsilon = 10^{-8}$. All experiments use the same fixed random seed and bf16 training when enabled.

Noise schedule. For the continuous numerical diffusion, we adopt the per-feature power-mean noise schedule following TabDiff [33], which extends the EDM parameterization [16] with a learnable shape parameter ρ_i for each numerical column. The schedule smoothly interpolates between $\sigma_{\min}$ and $\sigma_{\max}$ over normalized time $t \in [0, 1]$; we set $\sigma_{\min} = 0.002$ and $\sigma_{\max} = 80.0$ following EDM defaults. Formally, for each numerical feature $i \in \{1, \ldots, M_{\text{num}}\}$:

$$\sigma^{\text{num}}_{\rho_i}(t) = \left(\sigma_{\min}^{1/\rho_i} + t\left(\sigma_{\max}^{1/\rho_i} - \sigma_{\min}^{1/\rho_i}\right)\right)^{\rho_i}. \tag{23}$$

Learning ρ_i per column accommodates heterogeneous numerical distributions and yields a better-conditioned diffusion process than a single global ρ.

Sampling. TABDLM performs joint sampling over discrete tokens (categorical/text) and continuous numerical features through a coupled reverse process discretized into T steps; the full procedure is summarized in Algorithm 1. At each step, we (i) inject EDM-style churn noise [16] into the numerical state, (ii) embed both modalities into a shared MDLM input by mapping numerical values through the number encoder and tokens through the embedding table, (iii) run a single bidirectional MDLM forward pass that jointly conditions text denoising on noisy numericals and vice versa, and (iv) apply modality-specific reverse updates: progressive unmasking on the discrete side and an EDM–Euler step on the continuous side.

Algorithm 1 Sampling

Require: Schema prompt $\mathbf{p}$; generation length G; number of reverse steps T; noise schedules $\{\sigma_t\}_{t=0}^{T}$ and churned schedule $\{\hat{\sigma}_t\}_{t=0}^{T}$; remasking policy $\pi \in \{\textsc{HighConf}, \textsc{Rand}\}$.
Ensure: Synthetic record $\mathbf{x} = \big(x_j\big)_{j \in \mathcal{I}_{\text{num}} \cup \mathcal{I}_{\text{cat}} \cup \mathcal{I}_{\text{text}}}$.

// Initialization
$\mathbf{z}_{\mathbf{p}} \leftarrow \text{EMB}(\mathbf{p})$
$\mathbf{x}_T^{\text{tok}} \leftarrow \texttt{[MASK]}^G$ ▷ discrete state for $\mathcal{I}_{\text{cat}} \cup \mathcal{I}_{\text{text}}$
$\mathbf{x}_T^{\text{num}} \sim \mathcal{N}\big(\mathbf{0},\, \sigma_{\max}^2 \mathbf{I}\big)$ ▷ initial state from a Gaussian prior
for $t = T, T-1, \ldots, 1$ **do**
  // (i) Numerical perturbation (EDM churn)
  $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$; $\quad \hat{\mathbf{x}}_t^{\text{num}} \leftarrow \mathbf{x}_t^{\text{num}} + \sqrt{\hat{\sigma}_t^2 - \sigma_t^2}\, \boldsymbol{\epsilon}$
  // (ii) Map both modalities to MDLM token embeddings
  $\mathbf{z}_t^{\text{num}} \leftarrow \text{PROJ}_e\big(\text{ENC}(\hat{\mathbf{x}}_t^{\text{num}})\big)$ ▷ Eq. (8)
  $\mathbf{z}_t^{\text{tok}} \leftarrow \text{EMB}(\mathbf{x}_t^{\text{tok}})$ ▷ Eq. (9)
  $\mathbf{z}_t \leftarrow \big[\mathbf{z}_{\mathbf{p}};\, \mathbf{z}_t^{\text{tok}};\, \mathbf{z}_t^{\text{num}}\big]$
  // (iii) Joint bidirectional denoising
  $\mathbf{o}_t = \big[\mathbf{o}_t^{\text{p}};\, \mathbf{o}_t^{\text{tok}};\, \mathbf{o}_t^{\text{num}}\big] \leftarrow \text{MDLM}(\mathbf{z}_t,\, t)$ ▷ Eq. (10)
  // (iv) Discrete reverse step: progressive unmasking
  $p_\theta(\cdot \mid \mathbf{x}_t^{\text{tok}}) \leftarrow \text{HEAD}(\mathbf{o}_t^{\text{tok}})$ ▷ Eq. (12)
  $\tilde{\mathbf{x}}_0^{\text{tok}} \sim p_\theta(\cdot \mid \mathbf{x}_t^{\text{tok}})$ ▷ sample candidate tokens
  $\mathbf{x}_{t-1}^{\text{tok}} \leftarrow \textsc{Unmask}_\pi\big(\mathbf{x}_t^{\text{tok}},\, \tilde{\mathbf{x}}_0^{\text{tok}},\, p_\theta,\, t\big)$ ▷ reveal G/T positions per step
  // (v) Continuous reverse step: EDM–Euler update
  $\tilde{\mathbf{x}}_t^{\text{num}} \leftarrow \text{DEC}\big(\text{PROJ}_d(\mathbf{o}_t^{\text{num}})\big)$ ▷ Eq. (11)
  $\mathbf{d}_t \leftarrow \big(\hat{\mathbf{x}}_t^{\text{num}} - \tilde{\mathbf{x}}_t^{\text{num}}\big)/\hat{\sigma}_t$
  $\mathbf{x}_{t-1}^{\text{num}} \leftarrow \hat{\mathbf{x}}_t^{\text{num}} + \big(\sigma_{t-1} - \hat{\sigma}_t\big)\, \mathbf{d}_t$ ▷ discretization of Eq. (2)
end for
// Finalization
$\big\{x_j\big\}_{j \in \mathcal{I}_{\text{cat}} \cup \mathcal{I}_{\text{text}}} \leftarrow \text{Detokenize}(\mathbf{x}_0^{\text{tok}})$
$\big\{x_j\big\}_{j \in \mathcal{I}_{\text{num}}} \leftarrow \text{Denorm}(\mathbf{x}_0^{\text{num}})$ ▷ numerical denormalization
return $\mathbf{x}$

For discrete unmasking we consider two policies: (*i*) **high-confidence** unmasking, which reveals positions with the highest predicted token probability, and (*ii*) **random** unmasking, which reveals a uniformly random subset. In both cases we reveal G/T tokens per step, ensuring a smooth transition from a fully-masked sequence to a fully-specified one. After the final step, we decode tokens through the LM head and denormalize the numerical values to obtain the final mixed-type sample.

C Additional Experimental Results

C.1 Full results of Shape and Trend Metrics

Shape and Trend are fidelity metrics measuring column-wise marginal distributions and pair-wise column dependencies, respectively. Tables 11 and 12 report the complete results on real-world tabular datasets without free-form text features.

C.2 Evaluation results of MLE for real-world tabular dataset

In this section, we provide the evaluation results of MLE for real-world tabular datasets without free-form text features in Table 13. As shown in Table 13, TABDLM achieves competitive MLE performance on Adult, Default, Shoppers, and Magic, demonstrating its ability to faithfully capture the underlying data distribution and support high-quality synthetic data for downstream training. Notably, TABDLM attains the best overall score on Shoppers, and on Default even surpasses models trained on real data, suggesting that TABDLM can produce samples that are both distributionally consistent and beneficial for improving generalization. We observe a noticeable performance drop on

Table 11: Performance comparison on the error rates (%) of **Shape** ↓. (**B**: best overall; **B**: best in language-based model)

Method	Adult	Default	Shoppers	Magic	Beijing	Average
CTGAN	16.84 ± 0.03	16.83 ± 0.04	21.15 ± 0.10	9.81 ± 0.08	21.39 ± 0.05	17.20
TVAE	14.22 ± 0.08	10.17 ± 0.05	24.51 ± 0.06	8.25 ± 0.06	19.16 ± 0.06	15.26
GOGGLE	16.97	17.02	22.33	1.90	16.93	15.03
STaSy	11.29 ± 0.06	5.77 ± 0.06	9.37 ± 0.09	6.29 ± 0.13	6.71 ± 0.03	7.89
CoDi	21.38 ± 0.06	15.77 ± 0.07	31.84 ± 0.05	11.56 ± 0.26	16.94 ± 0.02	19.50
TabDDPM	1.75 ± 0.03	1.57 ± 0.08	2.72 ± 0.13	1.01 ± 0.09	1.30 ± 0.03	1.67
TabSyn	0.81 ± 0.05	1.01 ± 0.08	1.44 ± 0.07	1.03 ± 0.14	1.26 ± 0.05	1.11
TabDiff	**0.63 ± 0.05**	1.24 ± 0.07	1.28 ± 0.09	0.78 ± 0.08	1.03 ± 0.05	0.99
TabNAT	0.73	**0.80**	**1.11**	**0.62**	**0.79**	**0.81**
GReaT	12.12 ± 0.04	19.94 ± 0.06	14.51 ± 0.12	16.16 ± 0.09	8.25 ± 0.12	14.20
DiffLM	9.74	9.06	10.07	7.53	6.35	8.55
Qwen2.5-7B$_{\text{ICL}}$	27.80	22.39	34.37	13.41	31.83	25.96
Qwen2.5-14B$_{\text{ICL}}$	25.17	21.52	51.21	17.51	29.76	29.03
Qwen2.5-7B$_{\text{SFT}}$	5.11	2.58	8.76	7.53	7.49	6.29
LLaDA-8B$_{\text{SFT}}$	1.69	2.00	13.56	5.60	3.64	5.30
TabDLM	**1.46**	**1.18**	**2.05**	**2.93**	**2.42**	**2.01**

Table 12: Performance comparison on the error rates (%) of **Trend** ↓.(**B**: best overall; **B**: best in language-based model)

Method	Adult	Default	Shoppers	Magic	Beijing	Average
CTGAN	20.23 ± 1.20	26.95 ± 0.93	13.08 ± 0.16	7.00 ± 0.19	22.95 ± 0.08	18.04
TVAE	14.15 ± 0.88	19.50 ± 0.95	18.67 ± 0.38	5.82 ± 0.49	18.01 ± 0.08	15.23
GOGGLE	45.29	21.94	23.90	9.47	45.94	29.31
STaSy	14.51 ± 0.25	5.96 ± 0.26	8.49 ± 0.15	6.61 ± 0.53	8.00 ± 0.10	8.71
CoDi	22.49 ± 0.08	68.41 ± 0.05	17.78 ± 0.11	6.53 ± 0.25	7.07 ± 0.15	24.46
TabDDPM	3.01 ± 0.25	4.89 ± 0.10	6.61 ± 0.16	1.70 ± 0.22	2.71 ± 0.09	3.78
TabSyn	1.93 ± 0.07	2.81 ± 0.48	2.13 ± 0.10	0.88 ± 0.18	3.13 ± 0.34	2.18
TabDiff	**1.49 ± 0.16**	2.55 ± 0.75	1.74 ± 0.08	**0.76 ± 0.12**	2.59 ± 0.15	**1.83**
TabNAT	1.61	**2.21**	**1.67**	1.37	**2.48**	1.87
GReaT	17.59 ± 0.22	70.02 ± 0.12	45.16 ± 0.18	10.23 ± 0.40	59.60 ± 0.55	40.52
Qwen2.5-7B$_{\text{ICL}}$	37.61	31.02	34.77	20.62	40.83	32.97
Qwen2.5-14B$_{\text{ICL}}$	42.67	28.72	42.32	18.11	38.20	34.00
Qwen2.5-7B$_{\text{SFT}}$	17.03	16.99	10.28	14.45	28.68	17.49
LLaDA-8B$_{\text{SFT}}$	26.15	26.30	16.42	13.71	27.90	22.10
TabDLM	**2.74**	**2.33**	**2.40**	**2.85**	**2.93**	**2.65**

Beijing, which may be caused by the large fraction of missing values in the target column, where our mean-imputation preprocessing could distort the true target distribution and hurt likelihood-based metrics such as MLE.

C.3 Evaluation results of α-Precision

We report α-Precision results in Table 14. TabDLM achieves the best overall score on Default and the best average score among language-based models. Compared with tabular-specific synthetic models that are designed exclusively for numerical and categorical features, TabDLM remains competitive while uniquely supporting free-form text generation.

C.4 Evaluation results of C2ST

Table 15 reports the C2ST detection score, where higher values indicate that synthetic samples are statistically harder to distinguish from real ones. TabDLM achieves the best result among all language-based models, improving the average C2ST score by 13.2% compared with the strongest baseline. Compared with tabular-specific diffusion models designed exclusively for numerical and categorical features, TabDLM still achieves competitive performance.

Table 13: Evaluation of MLE. AUC is used for classification tasks and RMSE for regression tasks. (**B**: best overall; **B**: best in language-based model)

Methods	Adult	Default	Shoppers	Magic	Beijing	Average Gap
	AUC↑	AUC↑	AUC↑	AUC↑	RMSE↓	%
Real	.927 ± .000	.770 ± .005	.926 ± .001	.946 ± .001	.423 ± .003	0.0
CTGAN	.886 ± .002	.696 ± .005	.875 ± .009	.855 ± .006	.902 ± .019	28.48
TVAE	.878 ± .004	.724 ± .005	.871 ± .006	.887 ± .003	.770 ± .011	21.09
GOGGLE	.778 ± .012	.584 ± .005	.658 ± .052	.654 ± .024	1.09 ± .025	51.54
STaSy	.906 ± .001	.752 ± .006	.914 ± .005	.934 ± .003	.656 ± .014	12.45
CoDi	.871 ± .006	.525 ± .006	.865 ± .006	.932 ± .003	.818 ± .021	27.86
TabDDPM	.907 ± .001	.758 ± .004	.918 ± .005	.935 ± .003	.592 ± .011	9.14
TabSyn	.909 ± .001	.763 ± .002	.914 ± .004	**.937 ± .002**	.580 ± .009	8.44
TabDiff	.912 ± .002	.763 ± .005	.921 ± .004	.936 ± .003	**.555 ± .013**	**7.07**
TabNAT	.904	.764	.916	.935	.579	8.48
GReaT	.913 ± .003	.755 ± .006	.902 ± .005	.888 ± .008	**.653 ± .013**	13.31
DiffLM	.906	**.794**	.915	.917	.696	13.59
Qwen2.5-7B$_{\text{ICL}}$	.853	.390	.811	.829	.991	43.28
Qwen2.5-14B$_{\text{ICL}}$	.852	.639	.640	.844	1.03	42.05
Qwen2.5-7B$_{\text{SFT}}$	**.915**	.769	.895	.918	.674	**13.41**
LLaDA-8B$_{\text{SFT}}$	.909	.774	.872	**.919**	.679	14.13
TabDLM	.907	.791	**.923**	.905	.696	13.73

Table 14: Evaluation on α-Precision scores, (**B**: best overall; **B**: best in language-based model)

Method	Adult	Default	Shoppers	Magic	Beijing	Average
CTGAN	77.74 ± 0.15	62.08 ± 0.08	76.97 ± 0.39	86.90 ± 0.22	96.27 ± 0.14	79.99
TVAE	98.17 ± 0.17	85.57 ± 0.34	58.19 ± 0.26	86.19 ± 0.48	97.20 ± 0.10	85.06
GOGGLE	50.68	68.89	86.95	90.88	88.81	77.24
STaSy	82.87 ± 0.26	90.48 ± 0.11	89.65 ± 0.25	86.56 ± 0.19	89.16 ± 0.12	87.74
CoDi	77.58 ± 0.45	82.38 ± 0.15	94.95 ± 0.35	85.01 ± 0.36	98.13 ± 0.38	87.61
TabDDPM	96.36 ± 0.20	97.59 ± 0.36	88.55 ± 0.68	98.59 ± 0.17	97.93 ± 0.30	95.80
TabSyn	**99.39 ± 0.18**	98.65 ± 0.23	98.36 ± 0.52	99.42 ± 0.28	97.51 ± 0.24	98.67
TabDiff	99.02 ± 0.20	98.49 ± 0.28	**99.11 ± 0.34**	99.47 ± 0.21	98.06 ± 0.24	98.83
TabNAT	98.67	99.27	97.67	**99.50**	**99.27**	**98.88**
GReaT	55.79 ± 0.03	85.90 ± 0.17	78.88 ± 0.13	85.46 ± 0.54	**98.32 ± 0.22**	80.87
Qwen2.5-7B$_{\text{ICL}}$	85.51	88.56	27.96	83.39	63.76	69.84
Qwen2.5-14B$_{\text{ICL}}$	68.53	83.17	8.93	89.99	84.53	67.03
Qwen2.5-7B$_{\text{SFT}}$	84.70	95.25	95.46	82.45	94.35	90.44
LLaDA-8B$_{\text{SFT}}$	**99.32**	97.68	68.73	85.09	95.13	89.19
TabDLM	97.62	**99.56**	**98.51**	**97.90**	97.87	**98.29**

Table 15: Evaluation on C2ST, (**B**: best overall; **B**: best in language-based model)

Method	Adult	Default	Shoppers	Magic	Beijing	Average
CTGAN	0.5949	0.4875	0.7488	0.6728	0.7531	0.6514
TVAE	0.6315	0.6547	0.2962	0.7706	0.8659	0.6438
GOGGLE	0.1114	0.5163	0.1418	0.9526	0.4779	0.4400
STaSy	0.4054	0.6814	0.5482	0.6939	0.7922	0.6242
CoDi	0.2077	0.4595	0.2784	0.7206	0.7177	0.4768
TabDDPM	0.9755	0.9712	0.8349	**0.9998**	0.9513	0.9465
TabSyn	0.9910	**0.9826**	0.9662	0.9960	0.9528	0.9777
TabDiff	**0.9950**	0.9774	**0.9843**	0.9989	0.9781	**0.9867**
TabNAT	0.9870	0.9657	0.9626	0.9989	**0.9845**	0.9797
GReaT	0.5376	0.4710	0.4285	0.4326	0.6893	0.5118
Qwen2.5-7B$_{\text{ICL}}$	0.0925	0.3089	0.0591	0.5119	0.1787	0.2302
Qwen2.5-14B$_{\text{ICL}}$	0.1564	0.4866	0.0186	0.5903	0.2702	0.3044
Qwen2.5-7B$_{\text{SFT}}$	0.8037	0.9131	0.5722	0.7845	0.7315	0.7610
LLaDA-8B$_{\text{SFT}}$	0.9366	0.9140	0.5171	0.8489	**0.9799**	0.8393
TabDLM	**0.9386**	**0.9734**	**0.9594**	**0.9254**	0.9528	**0.9499**

C.5 Evaluation results of DCR

Table 16 reports the DCR score, which serves as a privacy metric: a value closer to 50% indicates that synthetic samples are equally close to training and held-out test records, suggesting the model has learned the underlying distribution rather than memorized training instances. TabDLM achieves the best overall average deviation across all baselines and obtains the best per-dataset DCR on Default and Beijing. This indicates that, despite being built on a large pretrained MDLM backbone, TabDLM does not exhibit memorization behavior and generates samples that genuinely reflect the data distribution.

Table 16: Evaluation on DCR score, where a score closer to 50% is more preferable. We also report the average absolute deviation of the DCR score from 50% across all datasets, where lower values are better (**B**: best overall).

Method	Adult	Default	Shoppers	Beijing	Average ↓
STaSy	$50.33\% \pm 0.19$	$50.23\% \pm 0.09$	$51.53\% \pm 0.16$	$50.59\% \pm 0.29$	0.67%
CoDi	$\mathbf{49.92\% \pm 0.18}$	$51.82\% \pm 0.26$	$51.06\% \pm 0.18$	$50.87\% \pm 0.11$	0.96%
TabDDPM	$51.14\% \pm 0.18$	$52.15\% \pm 0.20$	$63.23\% \pm 0.25$	$80.11\% \pm 2.68$	11.66%
TabSyn	$50.94\% \pm 0.17$	$51.20\% \pm 0.18$	$52.90\% \pm 0.22$	$50.37\% \pm 0.13$	1.35%
TabDiff	$50.10\% \pm 0.32$	$51.11\% \pm 0.36$	$\mathbf{50.24\% \pm 0.62}$	$50.50\% \pm 0.36$	0.39%
TabDLM	50.73%	$\mathbf{49.95\%}$	50.87%	$\mathbf{50.13\%}$	$\mathbf{0.36\%}$

C.6 Description of Ablation Variants

The variants in ablation study are defined as follows. (i) TabDLM-noFloatAE replaces the pretrained and frozen float encoder/decoder with randomly initialized modules of the same architecture, trained jointly with the rest of the model. (ii) TabDLM-onlyContDiff converts categorical features into one-hot vectors and feeds them through the continuous diffusion branch, removing MDLM-based modeling for categorical fields. (iii) TabDLM-onlyMDLM serializes numerical features and processes them in the same way as categorical and textual features, removing the continuous diffusion branch entirely. The aggregated results are reported in Table 4.

D Limitations

First, the sampling efficiency of TabDLM is lower than that of existing tabular data generation methods such as TabDiff [33] and TabSyn [44]. This limitation can be primarily attributed to the large model size of the MDLM backbone. Nevertheless, TabDLM is designed for broader application scenarios, as it supports the generation of numerical, categorical, and free-form text fields, capabilities that exceed those of existing methods. Moreover, common methods for accelerating MDLM inference, like block diffusion [2] and KV caching [39], could be incorporated to improve sampling efficiency. However, it is out of the scope of this work. Second, TabDLM models numerical and language diffusion using a shared noise schedule that couples the denoising dynamics across modalities and enforces a common step size for denoising both numerical and textual features. While effective in practice, this design may be suboptimal in certain scenarios. Introducing modality-specific noise schedules to partially decouple the denoising processes can be promising, which we leave for future investigation.