TABMADO: DUAL-GUIDED TABULAMBA DIFFUSION OVERSAMPLING FOR IMBALANCED TABULAR DATA CLASSIFICATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Tabular data, as a prevalent form of structured data, often suffers from class imbalance that limits classification performance. To address this issue, diffusion models have been employed for oversampling due to their ability to capture complex feature dependencies. However, when minority class samples are extremely scarce, they struggle to learn the genuine distribution of the minority class, consequently yielding generated samples of poor quality and diversity. To mitigate this challenge, this study proposes TabMaDo, a dual-guided diffusion framework tailored for tabular data. The framework consists of two components, a Mamba-based noise predictor named Tabulamba incorporates Class-Aware Mamba Blocks that dynamically highlights minority class information, preventing the generation process from being overwhelmed by the majority class. This enables the diffusion model to learn the overall data distribution, moving beyond the constraints imposed by scarce minority class samples. Complementing this, a lightweight Guider is introduced to provide dual guidance on both class label and distribution. This ensures the reverse diffusion process is steered toward minority class characteristics while preserving sample diversity, effectively overcoming the limitation of minority class sample scarcity. Together, they form a cohesive framework that enables high-quality sample generation under extreme class imbalance. Extensive experiments on 10 binary classification datasets demonstrate that TabMaDo outperforms six SOTA baseline methods across F1-score, G-mean and AUC metrics, achieving maximum improvements of 12% in F1-score, 16.5% in G-mean and 0.45% in AUC. These results confirm that TabMaDo provides an effective solution for imbalanced tabular data generation.

1 INTRODUCTION

Tabular data, as a structured data form, finds widespread applications across diverse domains, such as financial risk control, medical diagnosis, and customer relationship management Wang et al. (2025). A common challenge associated with tabular data is class imbalance, i.e., the size of minority class samples is significantly smaller than that of the majority class samples. In practice, this imbalance biases classifiers toward the majority class during training Sun et al. (2023), thereby affecting their ability to accurately identify samples from the minority class.

Numerous methods have been proposed to tackle the issue of class imbalance Ma et al. (2025), which can be categorized into algorithm-level and data-level methods de Oliveira & Berton (2023). Algorithm-level approaches redesign or modify existing classification algorithms to enhance their robustness to the minority class or increase the cost of misclassifying minority class samples D'souza et al. (2025). Although effective in reducing imbalance impacts, these methods often require dataset-specific adjustments to the loss function and lack generalizability. In contrast, data-level approaches mitigate class imbalance by improving the distribution of raw data Zhu et al. (2023), offering easier implementation, better universality, and independence from downstream tasks. Data-level methods are further divided into undersampling, oversampling, and hybrid sampling techniques Sun et al. (2022). The undersampling method reduces the number of samples in the majority class to rebalance the datasets Chen et al. (2026), but may discard valuable information. Hybrid sampling combines the undersampling and oversampling methods, although it tends to be complex and prone to noise

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

insertion. Oversampling, which increases samples from the minority class, is often considered more practical and advantageous Li et al. (2024b).

Among existing oversampling methods, traditional oversampling methods (e.g., SMOTE) are prone to overfitting or limited to local patterns Kozina & Pikus (2025), deep generative models such as Generative Adversarial Networks (GANs) Gui et al. (2023) and Variational Autoencoders (VAEs) Wijanarko et al. (2024) better capture the genuine data distribution, though they still face notable shortcomings, particularly in training stability and insufficient diversity in generated samples Liu et al. (2023). Recently, diffusion models have emerged as a powerful alternative due to their strong distribution modeling capability Karras et al. (2022) and have been adapted for tabular data generation. Nevertheless, a key limitation persists when minority class samples are extremely scarce, as diffusion models may struggle to accurately capture the minority class distribution. This can lead to the generation of synthetic samples with low diversity and quality, ultimately impairing downstream classification performance.

To address the above challenge, TabMaDo is proposed. The main contributions are summarized as follows. (1) Tabulamba is proposed as a Mamba-based noise predictor for tabular data, which enables diffusion models to better learn the overall data distribution and lays the groundwork for generating more realistic minority class samples; (2) a lightweight Guider is designed to leverage raw data labels and a prior prototype of the minority class, providing dual guidance on class label and distribution to steer the diffusion model toward generating samples aligned with the minority class; and (3) the superiority of TabMaDo over SOTA oversampling methods is demonstrated through experiments across diverse datasets and classifiers.

2 Related Work

2.1 Oversampling Methods

Oversampling remains an active and challenging research area in data mining. Early methods such as random oversampling (ROS) randomly replicate samples from the minority class Amagata (2025), which often leads to a lack of diversity and increased overfitting. SMOTE improves upon ROS by generating synthetic samples via linear interpolation between nearest neighbors Dablain et al. (2023), though it struggles to capture nonlinear feature relationships Zhou et al. (2023). Variants like Borderline-SMOTE focus on generating samples near class boundaries Sun et al. (2025), yet they remain largely constrained by the geometry of the raw sample distribution. These traditional techniques are inherently limited in their ability to model complex data distributions.

With the rapid advancement of deep learning Gorishniy et al. (2021), generative models such as VAE and GAN have been explored for tabular data synthesis Togo et al. (2025). TVAE, for instance, employs latent variable modeling and specialized preprocessing to handle heterogeneous tabular features Guidotti et al. (2024), yet it may produce suboptimal quality and diversity under multi-modal or strongly discrete distributions. TableGAN integrates convolutional neural networks into GANs to enhance the quality of synthesized tabular data and improve prediction accuracy for label columns Park et al. (2018). CWGAN-GP generates minority class samples for imbalanced binary classification Zheng et al. (2020). ACWGAN further adapts the approach to multi-class imbalance by incorporating an auxiliary classifier Kim et al. (2021). CTGAN and its extension CTAB-GAN are designed to model tabular data with mixed continuous and discrete features, supporting more complex data structures Xu et al. (2019). Despite their potential, GAN-based methods are often hampered by training instability and mode collapse.

The recent emergence of diffusion models has introduced a powerful alternative for tabular data generation. These models operate through a forward process that gradually adds noise to data until it becomes pure noise, followed by a learned reverse process that recovers the data distribution Kim et al. (2022). This unique bidirectional process constitutes a Markov process, enabling diffusion models to effectively learn complex distributions and generate high-quality, diverse samples. Notable diffusion models designed for tabular data generation include TabDDPM, which employs a classification-based feature processing architecture to handle heterogeneous data types Kotelnikov et al. (2023). CoDi integrates contrastive learning to preserve feature correlations Lee et al. (2023), while TabDiT enhances the TabDDPM framework with a Transformer backbone Hu et al. (2025). and RelDDPM trains a controller to steer the diffusion process toward generating condition-specific

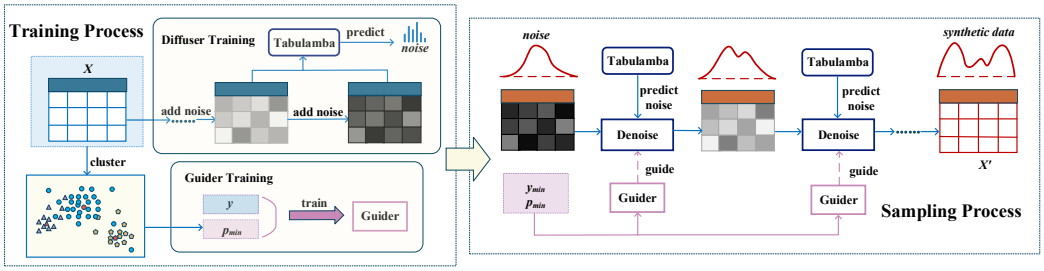


Figure 1: An overview of the proposed TabMaDo. This framework comprises two core components, the Diffuser and the Guider. The Diffuser employs a tailored Mamba architecture named Tabulamba to learn the overall data distribution, while the Guider supplies dual guidance derived from class label and a minority class prototype. During generation, the Diffuser incorporates guidance signals from the Guider at each step to synthesize realistic samples aligned with the target minority class.

tabular data Liu et al. (2024). Another approach, GOIO first maps raw data to a latent feature space via a VAE, then employs a conditional latent diffusion model for data generation within this latent feature space Ren et al. (2025). Although diffusion models generally offer stronger distribution modeling and greater stability compared to earlier generative approaches, they still face significant challenges when minority class samples are extremely scarce, providing insufficient information for the model to learn the minority class distribution.

2.2 State Space Models

State Space Models (SSMs) were initially proposed for sequence modeling and are widely applied in signal processing and time series analysis Gu et al. (2022). In continuous time, an SSM is described by:

$$h(t) = Ah(t-1) + Bx(t), \tag{1}$$
$$y(t) = Ch(t) + Dx(t), \tag{2}$$

in these equations, $h(t)$ represents the hidden state at time t, $h(t\text{-}1)$ is the previous hidden state, $x(t)$ is the current input sequence, and $y(t)$ is the output, A, B, C and D are learnable weight matrices.

Mamba represents a recent breakthrough in SSMs, introducing a selective state mechanism that enables efficient long-sequence modeling with linear time complexity $O(n)$ Guo et al. (2025), while maintaining performance competitive with Transformers, establishing a new paradigm for long sequence tasks Bick et al.. The success of Mamba has spurred its adoption across various domains. In computer vision, Vision Mamba establishes a pure SSM-based visual backbone, addressing limitations such as unidirectional modeling and lack of positional awareness Zhu et al. (2024). In the video domain, VideoMamba overcomes limitations of existing 3D CNNs and video Transformers Li et al. (2024a). Its linearly complex operators enable efficient long-range temporal modeling. In tabular data, MambaTab is the first Mamba-based tabular model, compressing all features into a pseudo-sequence of length 1 to achieve extreme parameter efficiency and native support for feature-incremental learning **?**, whereas Mambular treats each feature as a token and lets the sequence length equal the number of features, enabling genuine sequential modeling Thielmann et al. (2024). While MambaTab and Mambular demonstrate the promise of SSMs for discriminative tabular learning, neither addresses diffusion-based denoising, or the extremely scarce minority-class regime, leaving the design of an SSM noise-prediction backbone for imbalanced tabular diffusion entirely unexplored.

3 Proposed TabMaDo

3.1 Motivation

To address the above challenge, a two-stage strategy is adopted. In the first stage, the overall data distribution is learned to capture the complete feature space. In the second stage, the generation process is steered toward the minority class, avoiding direct modeling of the minority class distribution

from scarce samples. However, learning the overall distribution is non-trivial under severe imbalance. Standard diffusion models using MLPs or Transformers are easily dominated by the majority class, as the sparse minority class samples contribute little to shaping the learned distribution. This undermines the two-stage strategy. Mamba features a selective state space that dynamically adjusts state updates based on input. Building on this property, a dual-guided diffusion framework named TabMaDo is proposed. This framework consists of two integrated components. The first component is Tabulamba, a noise predictor built upon Class-Aware Mamba Blocks. By treating each sample as a sequence and leveraging a selective state space, Tabulamba emphasizes state updates for the minority class and employs bidirectional learning to capture feature importance, achieving robust overall distribution learning even under majority class dominance. The second component is a lightweight Guider, which provides guidance based on class label and a prior minority class prototype, steering generation toward the minority class.

3.2 Overview of TabMaDo

TabMaDo integrates two core components, the Diffuser and the Guider. The overall architecture is illustrated in Fig. 1. Within this framework, The Diffuser, which uses the newly designed Tabulamba as its noise predictor, learns the overall data distribution to generate synthetic samples. The Guider provides dual guidance based on class label and distribution to steer the generation toward the minority class.

The complete workflow is outlined in AppendixA.1.1. During training, Tabulamba learns feature correlations and their relative importance by predicting the added noise. This is achieved through bidirectional learning and state updating focused on the minority class. This enables the Diffuser to capture the overall data distribution. Concurrently, the Guider learns to recognize the minority class label and distinguish between minority and majority class features. In the generation phase, the Diffuser incorporates guidance signals from the Guider at each denoising time step, ensuring the generated samples align with the data distribution and label of the minority class, thereby implicitly steering the generation away from the majority class.

By leveraging the overall data distribution, TabMaDo generates higher-quality and more diverse minority class samples, even under extreme class imbalance, ultimately enhancing performance in downstream tasks. Prior to training, categorical attributes are preprocessed using analog bit encoding Chen et al., which first maps each category to an ordinal integer and then converts the integer into a continuous vector via binary representation.

3.3 Diffuser Training

This study begin by corrupting the preprocessed data with Gaussian noise through a Markov chain, defining the forward process as:

$$q(x_t|x_{t-1}) = \mathcal{N}(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t\mathbf{I}), \tag{3}$$

where $\{\beta_t \in (0,1)\}_{t=1}^{T}$ is the noise schedule controlling the variance at each step t, and x_t denotes the data after t steps of diffusion. As the forward process is a Markov Gaussian chain, the noisy data at time step t can be derived directly.

$$q(x_t|x_0) = \mathcal{N}(x_t; \sqrt{\bar{\alpha}_t}x_0, (1-\bar{\alpha}_t)\mathbf{I}), \tag{4}$$

where $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \prod_{i=0}^{t} \alpha_i$. And x_0 represents the raw data.

To effectively capture feature dependencies in tabular data, Tabulamba is designed as the noise predictor within the Diffuser. Tabulamba treats each sample as a sequence and operates within a selective state space. To dynamically propagate information, it employs state updating focused on the minority class as well as bidirectional learning, thereby learning feature importance and modeling feature interactions effectively. The structure of Tabulamba is depicted in Fig. 2.

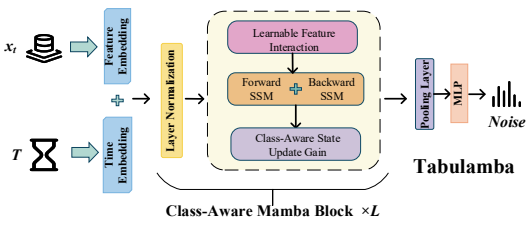


Figure 2: The architecture of Tabulamba.

Given tabular data $X \in \mathbb{R}^{N\times F}$, where N is the number of samples and F is the number of features, the noisy data x_t is obtained according to the forward diffusion process. The timestep $T \in \mathbb{R}^{N\times 1}$ is encoded via sinusoidal embedding, and the resulting timestep embedding is fused with the data embedding to form the initial input sequence T_0. The core of Tabulamba consists of L Class-Aware Mamba Blocks. The input sequence is first normalized and then processed through a learnable feature interaction layer.

$$X = T_{j-1} \times W_{\text{interaction}}, \tag{5}$$

where T_{j-1} is the input sequence, and $W_{\text{interaction}} \in \mathbb{R}^{F\times F}$ is a learnable interaction matrix. The feature vector is then split and processed by forward and backward SSM branches. Within each SSM, a class-aware state update gain is applied.

$$h_t = \Delta A_t \cdot h_{t-1} + \Delta B_t \cdot x_t \cdot g_{class}, \tag{6}$$

where h_t is the hidden state at time step t, ΔA_t and ΔB_t are discretized state space parameters, gain g_{class} is set to a fixed value based on the class label with $1+\alpha$ for minority class samples and 1 for majority class samples, where $\alpha > 0$ is a preset hyperparameter. And x_t is the input at time step t. The outputs from forward and backward SSMs are fused.

$$y = \frac{y_{\text{fwd}} + y_{\text{bwd}}}{2}, \tag{7}$$

where y_{fwd} and y_{bwd} are the outputs from the forward and backward SSM branches, respectively. The fused output is combined with the gating vector.

$$T_j = y \odot \sigma(z) + T_{j-1}, \tag{8}$$

where $\odot$ denotes element-wise multiplication, $\sigma(\cdot)$ is the sigmoid activation function, z is the gating vector obtained from the input features, and the addition implements a residual connection. After L blocks, the final sequence T_L is passed through an MLP to obtain the predicted noise:

$$Noise = \text{MLP}(T_L), \tag{9}$$

The reverse process of the diffusion model uses the noise predicted by Tabulamba to estimate the conditional distribution.

$$p_\theta(x_{t-1}|x_t) = \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t), \sigma_t^2 \mathbf{I}), \tag{10}$$

where $\sigma_t^2 = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t$, and the mean is calculated as:

$$\mu_\theta(x_t, t) = \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(x_t, t)\right), \tag{11}$$

where $\epsilon_\theta(x_t, t)$ denotes the output of the Tabulamba, which is an estimate of the Gaussian noise ϵ_t added during the forward process. By iteratively this reverse step, the raw data distribution can be progressively reconstructed from pure noise.

The training procedure of the Diffuser is outlined in Appendix A.1.2. First, the forward process is executed according to Eq. equation 4, transforming tabular data into pure noise (lines 5–8). then training Tabulamba to approximate the reverse process, i.e., to predict the noise added during the forward process (line 9). The loss function is defined as:

$$L_D = \mathbb{E}_{t,x_t,\epsilon_t} \parallel \epsilon_t - \epsilon_\theta(x_t, t) \parallel^2, \tag{12}$$

where L_D denotes the loss function of the Diffuser. This loss function aims to minimize the mean squared error between the predicted noise $\epsilon_\theta(x_t, t)$ and the actual noise ϵ_t.

3.4 Guider Training

To steer generated data toward the minority class, a lightweight Guider is designed to provide dual guidance concerning both class label

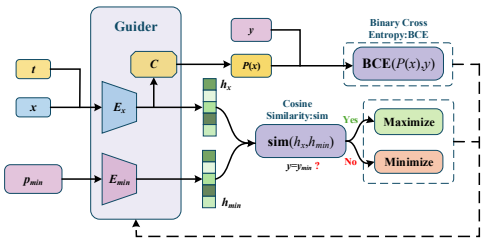


Figure 3: The training flowchart of Guider.

and distribution. Before training, a prior prototype for the minority class is constructed by computing the empirical mean of all minority class samples. The prior prototype is defined as:

$$p_{\min} = \mathbb{E}[x_{\min}], \tag{13}$$

where p_{min} is the computed prior prototype, $\mathbb{E}[\cdot]$ denotes the empirical mean operator, and $x_{\min}$ represents the set of minority class samples.

The training pipeline of the Guider is shown in Fig. 3. It employs two encoders to map noisy samples and the prior prototype into a joint embedding space. Since the Guider must operate at every step of the reverse process, The sample encoder E_x is designed to accept both the noisy sample x_t and the corresponding diffusion time step t as input. And the prior prototype encoder E_{min} takes p_{min} as input. A prediction head C attached to E_x outputs the estimated probability that the input noisy sample belongs to the minority class y_{min}. The classification loss is given by:

$$L_{class} = BCE(P(x_t), y), \tag{14}$$

L_{class} is the binary cross-entropy loss, $P(x)$ is the predicted probability for the minority class, and y is the true class label of the sample (1 for minority class, 0 for majority class).

To further align the distribution, a prototype loss is introduced. The cosine similarity between the noisy sample embedding and that of the prior prototype in the joint space is computed.

$$sim = cos_similarity(h_x, h_{min}), \tag{15}$$

where sim represents the cosine similarity between the two embeddings, $h_x = E_x(x_t, t)$, denotes the embedding of the noisy sample at diffusion step t, and $h_{min} = E_{min}(p_{min})$, denotes the embedding of the minority class prior prototype. Intuitively, a high similarity indicates that the noisy sample is likely to follow the genuine distribution of the minority class.

The prototype loss is then formulated as:

$$L_{proto} = \mathbb{E}_{x_t \sim \mathbb{P}_{\text{min}}} \left[1 - \text{sim}(h_x, h_{\text{min}})\right] + \mathbb{E}_{x_t \sim \mathbb{P}_{\text{maj}}} \left[\text{ReLU}\left(\text{sim}(h_x, h_{\text{min}})\right)\right]. \tag{16}$$

where the first term pulls minority class samples toward the prototype, and the second term pushes majority class samples away from the prototype. This objective effectively enforces distribution guidance during generation.

The total training loss for the Guider is the weighted sum of classification and contrastive losses.

$$L_G = L_{class} + \gamma L_{proto}, \tag{17}$$

L_G is the total loss of the Guider, and $\gamma > 0$ is a hyperparameter that balancees the contribution of the two loss terms.

The training process of the Guider is provided in Appendix A.1.3. The sample encoder E_x is shared between the classification and prototype tasks, ensuring consistent feature representation across both objectives and avoiding potential conflicts when jointly guiding the class label and distribution.

3.5 Data Sampling

During the generation phase, TabMaDo utilizes the trained Guider to steer the trained Diffuser to generate synthetic data that better aligns with the minority class label and distribution. The procedure is detailed in Appendix A.1.4.

In each reverse process, the Diffuser first obtains the mean μ_t and variance σ_t^2 of $p_\theta(x_{t-1}|x_t)$ (Lines 6–8). Then, based on the guidance gradient g_t provided by the Guider, the new mean μ_* is recalculated to adjust the mean for the reverse process (lines 9-10) using the following formula.

$$\mu_* = \mu_t + s \cdot g_t, \tag{18}$$

where *s* is a scalar controlling the guidance strength.

Generate x_{t-1} using the adjusted mean and sampled noise $z \sim \mathcal{N}(0, \mathbf{I})$ (Line 12). By incorporating the guidance gradient from the Guider at each step of the reverse process, the class label and distribution of the generated data are progressively converged toward the minority class.

3.6 Convergence Analysis

Eq. equation 12 defines the loss function of the Diffuser, which is a standard empirical loss for diffusion models. Eq. equation 17 is the composite loss of the Guider, requiring convergence analysis. The Guider includes three learnable components, the gradient for updating the sample encoder E_x combines contributions from both L_{class} and L_{proto}, the prior prototype encoder E_{min} is updated based on L_{proto}, and the prediction head C is updated using L_{class}.

Under standard smoothness and boundedness assumptions, detailed in supplementary materials, the following results are established.

Lemma 1 (Monotonicity of L_{class} with θ_h Update)**.** *At iteration t+1, fix θ_x and θ_{min}, and only update θ_h. Then $L_{class}(t+1) \leq L_{class}(t)$, and L_{proto} remains unchanged.*

Lemma 2 (Monotonicity of L_{proto} with θ_{min} Update)**.** *At iteration t+1, fix θ_x and θ_h, and only update θ_{min}. Then $L_{proto}(t+1) \leq L_{proto}(t)$, and L_{class} remains unchanged.*

Lemma 3 (Monotonicity of L_G with θ_x Update)**.** *At iteration $t+1$, fix θ_h and $\theta_{\min}$, and update θ_x. The total gradient of the composite loss with respect to θ_x is $\nabla_{\theta_x} L_G = \nabla_{\theta_x} L_{class} + \gamma \nabla_{\theta_x} L_{proto}$, and the composite loss satisfies $L_G(t+1) \leq L_G(t)$.*

Proof. Details are provided in Appendix A.2.2. □

Based on the above lemmas, we present the final convergence theorem for the composite loss function.

Theorem 1 (Convergence of the Overall Loss)**.** *Under Assumptions 1–3, set the learning rate $\eta = \frac{1}{\beta_\xi}$ where $\xi = 2(1+\gamma^2)G$. For the composite loss sequence $\{L_G(\theta_t)\}$ generated by gradient descent updates, the loss sequence converges to a constant Loss*, i.e., $\lim_{t\to\infty} L_G(\theta_t) = Loss^*$. Meanwhile, the gradient norm of the loss function converges to 0: $\lim_{t\to\infty} \|\nabla L_G(\theta_t)\| = 0$.*

Formally, for any $\epsilon > 0$, there exists $N > 0$ such that for all $t > N$: $L_G(\theta) - Loss^* < \varepsilon$, $\|\nabla L_G(\theta)\| < \varepsilon$.

Proof. Detailed in Appendix A.2.3. □

Thus, the training of the Guider is guaranteed to converge under the stated assumptions.

4 Experiments And Analysis

4.1 Experimental Settings

Datasets. Extensive experiments on 10 binary classification datasets from diverse domains are conducted to evaluate TabMaDo. The key characteristics of these datasets are summarized in Appendix A.3.1. The Imbalance Ratio (IR) is defined as the ratio of the size of majority class samples (N_{maj}) to the size of minority class samples (N_{min}) Li et al. (2023).

Baseline Methods. TabMaDo is compared against following six SOTA methods. stable_SMOTE Feng et al. (2021), TabDDPM Kotelnikov et al. (2023), TabDiT Hu et al. (2025), CoDi Lee et al. (2023), RelDDPM Liu et al. (2024) and GOIO Ren et al. (2025). Detailed descriptions of these baselines are provided in AppendixA.3.2

Evaluation Method. The performance of each oversampling method is assessed by training classifiers on balanced training sets. Three widely-used classifiers are employed: Random Forest (RF) Rhodes et al. (2023), Gradient Boosting Decision Tree (GBDT) Wu et al. (2023), and TabPFN Hollmann et al. (2025).

Ablation Study Settings. To isolate the contribution of each core component, four ablated variants are constructed as baselines. TabMaDo$_{MLP}$ (V_1) substitutes the Tabulamba with a standard MLP. TabMaDo$_{Only-C}$ (V_2) restricts the Guider to use only labels for guidance. TabMaDo$_{NoG}$ (V_3) removes the Guider entirely. TabMaDo$_{MLP-NoG}$ (V_4) combines the changes from V_1 and V_3, replacing Tabulamba with an MLP while also removing the Guider. All models are trained under identical experimental settings.

Implementation Details. All experiments employ stratified 10-fold cross-validation, preserving the original class distribution in each fold and ensuring reliable and generalizable results. The class-aware gain g_{class} is set to $1 + \alpha$ for minority class samples, with $\alpha \in [0.5, 2.0]$ with a default value of 1.0. The balancing hyperparameter $\gamma \in [0.1, 0.5]$. And this framework is implemented using Python 3.10.

4.2 Performance Evaluation Results

Table 1: Average Performance of Seven Methods on F1-score, G-mean, and AUC Metrics for RF Classifier Across Fourteen Benchmark Datasets.

Metric	Dataset	stable_SMOTE	TabDDPM	CoDi	TabDiT	RelDDPM	GOIO	TabMaDo
F1-score	yeast6	0.4745±0.16	0.4342±0.15	0.3043±0.20	0.2138±0.22	0.5139±0.20	0.2795±0.18	**0.5372±0.20**
	wilt	0.8249±0.05	0.8002±0.06	0.7742±0.06	0.7981±0.07	0.8340±0.04	0.2479±0.07	**0.8454±0.03**
	abalone9-18	0.3546±0.18	0.3388±0.14	0.2829±0.24	0.2257±0.15	0.3910±0.21	0.2262±0.13	**0.4015±0.15**
	ecoli4	0.7767±0.31	0.7800±0.33	0.7367±0.20	0.6300±0.27	0.8000±0.31	0.5019±0.30	**0.8100±0.31**
	glass-0-1-4-6-vs-2	0.2238±0.23	0.1833±0.28	0.0667±0.20	0.1167±0.24	0.2733±0.28	0.0667±0.20	**0.3933±0.33**
	vehicle0	0.9023±0.08	0.9014±0.07	0.9063±0.03	0.8977±0.05	0.9135±0.05	0.5425±0.13	**0.9227±0.05**
	haberman	0.4229±0.17	0.3487±0.15	0.3199±0.19	0.3549±0.14	0.4332±0.16	0.3322±0.07	**0.4533±0.11**
	phoneme	0.8227±0.02	0.8219±0.03	0.8063±0.02	0.8116±0.02	0.8208±0.02	0.4533±0.05	**0.8238±0.01**
	german	0.4723±0.09	0.4612±0.06	0.4067±0.09	0.4803±0.07	0.5262±0.06	0.5140±0.11	**0.5279±0.09**
	magic	0.7940±0.01	0.8015±0.01	0.7908±0.01	0.7927±0.01	0.8011±0.01	0.6970±0.03	**0.8016±0.01**
G-mean	yeast6	0.7482±0.16	0.7318±0.16	0.4830±0.27	0.2779±0.28	0.7651±0.17	0.6317±0.27	**0.7664±0.17**
	wilt	0.9075±0.05	0.9452±0.03	0.8462±0.05	0.8420±0.07	0.9319±0.03	0.5533±0.08	**0.9478±0.03**
	abalone9-18	0.6025±0.16	0.4888±0.20	0.3792±0.27	0.3418±0.22	0.5999±0.16	0.5302±0.21	**0.6378±0.14**
	ecoli4	0.8371±0.30	0.8376±0.30	0.8453±0.15	0.7193±0.27	0.8121±0.30	0.6680±0.36	**0.8664±0.30**
	glass-0-1-4-6-vs-2	0.3591±0.37	0.2102±0.32	0.0707±0.21	0.1395±0.28	0.3460±0.35	0.0707±0.21	**0.5241±0.44**
	vehicle0	0.9355±0.06	0.9300±0.05	0.9372±0.03	0.9290±0.04	0.9496±0.04	0.6240±0.11	**0.9561±0.04**
	haberman	0.5776±0.15	0.5063±0.14	0.4583±0.21	0.5079±0.12	0.5819±0.14	0.4972±0.06	**0.6005±0.10**
	phoneme	0.8815±0.01	0.8812±0.02	0.8533±0.02	0.8555±0.02	0.8781±0.01	0.5562±0.04	**0.8831±0.01**
	german	0.5975±0.07	0.5753±0.05	0.5327±0.07	0.5914±0.06	0.6400±0.05	0.6347±0.10	**0.6479±0.07**
	magic	0.8374±0.01	**0.8419±0.01**	0.8272±0.01	0.8277±0.01	0.8408±0.01	0.7485±0.02	0.8405±0.01
AUC	yeast6	0.8687±0.09	0.8443±0.08	0.8120±0.11	0.8383±0.13	0.8969±0.08	0.8936±0.09	**0.9014±0.11**
	wilt	0.9797±0.02	0.9872±0.01	0.9644±0.02	0.9704±0.03	0.9880±0.01	0.7496±0.07	**0.9888±0.02**
	abalone9-18	0.7738±0.14	0.8212±0.11	0.7687±0.17	0.8109±0.10	**0.8746±0.09**	0.7761±0.10	0.8009±0.11
	ecoli4	0.9663±0.08	0.9904±0.02	0.9272±0.11	0.9865±0.02	**0.9944±0.01**	0.9089±0.11	0.9639±0.09
	glass-0-1-4-6-vs-2	0.8027±0.08	0.8109±0.21	0.7041±0.20	0.6446±0.22	0.8284±0.15	0.5565±0.19	**0.8298±0.13**
	vehicle0	0.9867±0.02	0.9822±0.03	0.9839±0.01	0.9802±0.01	0.9855±0.01	0.8856±0.04	**0.9894±0.01**
	haberman	0.6503±0.12	0.6210±0.11	0.5786±0.14	0.6033±0.12	0.6512±0.15	0.5958±0.07	**0.6544±0.09**
	phoneme	0.9405±0.01	0.9436±0.01	0.9402±0.01	0.9447±0.01	0.9430±0.01	0.8044±0.02	**0.9453±0.01**
	german	0.7319±0.06	0.7300±0.06	0.7338±0.06	0.7337±0.04	0.7329±0.06	0.7305±0.08	**0.7344±0.06**
	magic	0.9125±0.01	0.9150±0.01	0.9135±0.01	0.9148±0.00	0.9152±0.01	0.8604±0.02	**0.9156±0.01**

Table 1 compares the performance of different oversampling methods on RF classifier across F1-score, G-mean, and AUC, where the best results are highlighted in bold. Metric results for other classifiers are provided in Appendix A.4.1. As shown, TabMaDo achieves the highest average performance on all three metrics. The most substantial improvements are observed on *glass-0-1-4-6-vs-2*, where TabMaDo raises F1-score and G-mean by 12% and 16.5%, respectively. On *yeast-6*, AUC improved by 0.45%. Notably, on datasets with high IR (e.g., *yeast-6*, *Wilt* and *abalone9-18*), TabMaDo maintains strong performance while other methods degrade.

The average performance and standard deviations across datasets and classifiers are detailed in Appendix A.4.1. The figure clearly demonstrates that TabMaDo significantly outperforms other SOTA baselines in the vast majority of cases.

Beyond the classification performance reported above, we further directly evaluate the quality of the generated minority-class samples in terms of fidelity, diversity, and distribution matching. The results are detailed in Appendix A.4.2. TabMaDo achieves the best average rank on WD, JS, KL, KS, and Coverage, and remains competitive on L2, MMD, and KNN. These results indicate that the synthetic minority-class samples produced by TabMaDo not only improve downstream classification but also closely match the real data distribution.

4.3 Statistical Analysis

To statistically compare the performance of different oversampling methods across multiple datasets, this study employ the Friedman test followed by the Nemenyi post-hoc test, with the detailed experimental setup and complete statistical results provided in AppendixA.4.3. The average ranking scores show that TabMaDo consistently attains the lowest ranking scores across all classifiers and evaluation metrics, indicating superior performance. The results of the Nemenyi post-hoc test show

that TabMaDo significantly outperforms baseline methods in most cases. For example, under the RF classifier, TabMaDo is significantly superior to CoDi, TabDiT and GOIO for F1-score and G-mean, and to TabDiT and GOIO for AUC.

In summary, the Friedman rankings and Nemenyi post-hoc tests confirm that TabMaDo exhibits statistically significant and consistent superiority over SOTA oversampling methods.

4.4 Ablation Study

The complete results of the ablation study are provided in Appendix A.4.4. Based on the results of the ablation study, the following analysis addresses the two research questions.

- **RQ-1**: Does Tabulamba outperform a standard MLP for modeling tabular data distribution?
- **RQ-2**: Does the Guider improve the quality and diversity of generated minority class samples?

For **RQ-1**, we compare variants with and without Tabulamba while keeping the Guider status unchanged. V1 replaces Tabulamba with an MLP. On *glass-0-1-4-6-vs-2*, V1 achieves an F1-score of 0.1833 versus 0.3933 for the full TabMaDo. Similarly, V4 which replaces Tabulamba with an MLP and removes the Guider underperforms V3 which retains Tabulamba but removes the Guider. On *abalone9-18*, V4 achieves a G-mean of 0.3098 versus 0.3572 for V3. These results confirm that Tabulamba outperforms a standard MLP in capturing tabular data dependencies.

For **RQ-2**, the full TabMaDo significantly outperforms V3 which removes the Guider entirely. On *glass-0-1-4-6-vs-2*, V3 achieves a G-mean of 0.0707 versus 0.5241 for the full TabMaDo. V2 which restricts the Guider to use only labels also underperforms, achieving 0.3630 on *yeast6*. This underscores the critical role of the Guider in leveraging class label and distribution information to steer the diffusion process, thereby enhancing the quality and diversity of generated minority class samples.

In summary, the ablation study confirms that all core components of TabMaDo provide substantial contributions. Tabulamba models tabular data distributions more effectively than a standard MLP, while the Guider ensures effective conditional generation for the minority class.

4.5 Additional Feature Order Experiments

To evaluate the sensitivity of Tabulamba to feature order and its ability to preserve minority-class information during diffusion, we conduct two additional analyses and report the full results in Appendix A.4.5. Results show that Tabulamba is largely insensitive to feature order variations and consistently achieves lower total, majority, and minority losses than MLP and Transformer backbones, indicating that it mitigates majority class dominance and enables balanced distribution learning on imbalanced tabular data.

5 Conclusion And Future Work

This study tackles severe class imbalance in tabular data by proposing TabMaDo, a novel oversampling framework built on a dual-guided diffusion architecture with a Tabulamba backbone. The framework introduces two key components. Tabulamba, a Mamba-based noise predictor specifically designed to model overall data distribution for tabular data within the diffusion model, and a lightweight Guider that provides dual guidance from both class label and distribution. Together, they enable high-quality and diversity generation of minority class samples even under extreme scarcity.

The proposed TabMaDo framework was evaluated against six SOTA methods on 10 imbalanced tabular datasets. TabMaDo achieved significant improvements validated by statistical tests. Ablation studies verified the essential role of Tabulamba and the Guider. Additional feature order experiments further reveal that Tabulamba is insensitive to feature order and mitigates majority class dominance via the selective state space of Mamba, a capability lacking in MLPs and Transformers. This confirms that applying Mamba to tabular data is necessary for balanced distribution learning under extreme imbalance.

In future work, this study plans to extend TabMaDo to multi-class imbalanced scenarios and explore its applicability to other data modalities such as image and time series data.

AI USE STATEMENT

In this work, we used generative AI tools for language polishing of the manuscript. We have not used generative AI tools for generating research ideas, designing methods, deriving theoretical results, or producing experimental data, and the rest of the required disclosure tasks are not applicable to this work. Additionally, we used generative AI tools for grammar checking and improving the clarity of the writing. We have reviewed all AI-assisted work. Specifically, all AI-polished text was carefully checked by the authors against the original meaning to ensure that no technical content, claims, or citations were altered or fabricated. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

REFERENCES

Daichi Amagata. Random sampling over spatial range joins. In *41st IEEE International Conference on Data Engineering, ICDE*, pp. 2080–2093, 2025.

Aviv Bick, Kevin Y. Li, Eric P. Xing, J. Zico Kolter, and Albert Gu. Transformers to ssms: Distilling quadratic knowledge to subquadratic models. In *Advances in Neural Information Processing Systems 37: Annual Conference on Neural Information Processing Systems, NeurIPS 2024*.

Rujia Chen, Yi Xie, Minglang Liao, Jiankun Hu, and Xingcheng Liu. Identifying fraudulent users in e-commerce applications through spatiotemporal fusion and selective aggregation. *ACM Trans. Priv. Secur.*, 29(1):6:1–6:27, 2026.

Ting Chen, Ruixiang Zhang, and Geoffrey E. Hinton. Analog bits: Generating discrete data using diffusion models with self-conditioning. In *The Eleventh International Conference on Learning Representations, ICLR 2023*.

Damien Dablain, Bartosz Krawczyk, and Nitesh V. Chawla. Deepsmote: Fusing deep learning and SMOTE for imbalanced data. *IEEE Trans. Neural Networks Learn. Syst.*, 34(9):6390–6404, 2023.

Willian Dihanster G. de Oliveira and Lilian Berton. A systematic review for class-imbalance in semi-supervised learning. *Artif. Intell. Rev.*, 56(S2):2349–2382, 2023.

Annie D'souza, Swetha M, and Sunita Sarawagi. Synthetic tabular data generation for imbalanced classification: The surprising effectiveness of an overlap class. In *Thirty-Ninth AAAI Conference on Artificial Intelligence, Thirty-Seventh Conference on Innovative Applications of Artificial Intelligence, Fifteenth Symposium on Educational Advances in Artificial Intelligence, AAAI*, pp. 16127–16134, 2025.

Shuo Feng, Jacky Keung, Xiao Yu, Yan Xiao, and Miao Zhang. Investigation on the stability of smote-based oversampling techniques in software defect prediction. *Inf. Softw. Technol.*, 139: 106662, 2021.

Yury Gorishniy, Ivan Rubachev, Valentin Khrulkov, and Artem Babenko. Revisiting deep learning models for tabular data. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems*, pp. 18932–18943, 2021.

Albert Gu, Karan Goel, Ankit Gupta, and Christopher Ré. On the parameterization and initialization of diagonal state space models. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems, NeurIPS*, 2022.

Jie Gui, Zhenan Sun, Yonggang Wen, Dacheng Tao, and Jieping Ye. A review on generative adversarial networks: Algorithms, theory, and applications. *IEEE Trans. Knowl. Data Eng.*, 35(4): 3313–3332, 2023.

Riccardo Guidotti, Anna Monreale, Mattia Setzu, and Giulia Volpi. Generative model for decision trees. In *Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI 2024, Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence, IAAI 2024, Fourteenth Symposium on Educational Advances in Artificial Intelligence, EAAI*, pp. 21116–21124, 2024.

Hang Guo, Yong Guo, Yaohua Zha, Yulun Zhang, Wenbo Li, Tao Dai, Shu-Tao Xia, and Yawei Li. Mambairv2: Attentive state space restoration. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR*, pp. 28124–28133, 2025.

Noah Hollmann, Samuel Müller, Lennart Purucker, Arjun Krishnakumar, Max Körfer, Shi Bin Hoo, Robin Tibor Schirrmeister, and Frank Hutter. Accurate predictions on small data with a tabular foundation model. *Nat.*, 637(8044):319–326, 2025.

Xiaowen Hu, Ming Zheng, Rui Zhu, Xuan Zhang, and Zhi Jin. Fed-olf: Federated oversampling learning framework for imbalanced software defect prediction under privacy protection. *IEEE Trans. Reliab.*, 74(3):3266–3280, 2025.

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems*, 2022.

Hayoung Kim, Jongwon Park, Kyushik Min, and Kunsoo Huh. Anomaly monitoring framework in lane detection with a generative adversarial network. *IEEE Trans. Intell. Transp. Syst.*, 22(3): 1603–1615, 2021.

Jayoung Kim, Chaejeong Lee, Yehjin Shin, Sewon Park, Minjung Kim, Noseong Park, and Jihoon Cho. SOS: score-based oversampling for tabular data. In *KDD '22: The 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 762–772, 2022.

Akim Kotelnikov, Dmitry Baranchuk, Ivan Rubachev, and Artem Babenko. Tabddpm: Modelling tabular data with diffusion models. In *International Conference on Machine Learning, ICML*, volume 202, pp. 17564–17579, 2023.

Agata Kozina and Michal Pikus. A comprehensive review of data augmentation methods for support business processes using machine learning and deep learning. In *Emerging Challenges in Intelligent Management Information Systems - Proceedings of 28th European Conference on Artificial Intelligence ECAI*, volume 1643, pp. 66–79, 2025.

Chaejeong Lee, Jayoung Kim, and Noseong Park. Codi: Co-evolving contrastive diffusion models for mixed-type tabular synthesis. In *International Conference on Machine Learning, ICML*, volume 202, pp. 18940–18956, 2023.

Kunchang Li, Xinhao Li, Yi Wang, Yinan He, Yali Wang, Limin Wang, and Yu Qiao. Videomamba: State space model for efficient video understanding. In *Computer Vision - ECCV - 18th European Conference*, volume 15084, pp. 237–255, 2024a.

Wen-Zhi Li, Chang-Dong Wang, Hui Xiong, and Jian-Huang Lai. Graphsha: Synthesizing harder samples for class-imbalanced node classification. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD*, pp. 1328–1340, 2023.

Zhixun Li, Yushun Dong, Qiang Liu, and Jeffrey Xu Yu. Rethinking fair graph neural networks from re-balancing. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD*, pp. 1736–1745, 2024b.

Haozhe Liu, Bing Li, Haoqian Wu, Hanbang Liang, Yawen Huang, Yuexiang Li, Bernard Ghanem, and Yefeng Zheng. Combating mode collapse via offline manifold entropy estimation. In *Thirty-Seventh AAAI Conference on Artificial Intelligence, AAAI 2023, Thirty-Fifth Conference on Innovative Applications of Artificial Intelligence, IAAI 2023, Thirteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2023*, pp. 8834–8842, 2023.

Tongyu Liu, Ju Fan, Nan Tang, Guoliang Li, and Xiaoyong Du. Controllable tabular data synthesis using diffusion models. *Proc. ACM Manag. Data*, 2(1):28:1–28:29, 2024.

Yihong Ma, Yijun Tian, Nuno Moniz, and Nitesh V. Chawla. Class-imbalanced learning on graphs: A survey. *ACM Comput. Surv.*, 57(8):207:1–207:16, 2025.

Noseong Park, Mahmoud Mohammadi, Kshitij Gorde, Sushil Jajodia, Hongkyu Park, and Youngmin Kim. Data synthesis based on generative adversarial networks. *Proc. VLDB Endow.*, 11(10): 1071–1083, 2018.

Shiqi Ren, Jinliang Ding, Cuie Yang, and Yiu-ming Cheung. GOIO: generative oversampling approach to class imbalance and overlap of tabular data. *IEEE Trans. Knowl. Data Eng.*, 37(11): 6450–6463, 2025.

Jake S. Rhodes, Adele Cutler, and Kevin R. Moon. Geometry- and accuracy-preserving random forest proximities. *IEEE Trans. Pattern Anal. Mach. Intell.*, 45(9):10947–10959, 2023.

Hao Sun, Jianping Li, and Xiaoqian Zhu. A novel expandable borderline smote over-sampling method for class imbalance problem. *IEEE Trans. Knowl. Data Eng.*, 37(5):2183–2199, 2025.

Yi Sun, Lijun Cai, Bo Liao, and Wen Zhu. Minority sub-region estimation-based oversampling for imbalance learning. *IEEE Trans. Knowl. Data Eng.*, 34(5):2324–2334, 2022.

Yi Sun, Lijun Cai, Bo Liao, Wen Zhu, and Junlin Xu. A robust oversampling approach for class imbalance problem with small disjuncts. *IEEE Trans. Knowl. Data Eng.*, 35(6):5550–5562, 2023.

Anton Frederik Thielmann, Manish Kumar, Christoph Weisser, Arik Reuter, Benjamin Säfken, and Soheila Samiee. Mambular: A sequential model for tabular deep learning. *CoRR*, abs/2408.06291, 2024.

Ren Togo, Nao Nakagawa, Takahiro Ogawa, and Miki Haseyama. Concvae: Conceptual representation learning. *IEEE Trans. Neural Networks Learn. Syst.*, 36(4):7529–7541, 2025.

Xiaoguang Wang, Chenxu Wang, Mengqin Wang, Jun Liu, and Xiaohong Guan. B2BGAN: A backbone-to-branches gan-based oversampling approach for class-imbalanced tabular data. *IEEE Trans. Knowl. Data Eng.*, 37(10):5808–5822, 2025.

Hansen Wijanarko, Evelyne Calista, Li-Fen Chen, and Yong-Sheng Chen. Tri-vae: Triplet variational autoencoder for unsupervised anomaly detection in brain tumor MRI. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR*, pp. 3930–3939, 2024.

Zhaomin Wu, Junhui Zhu, Qinbin Li, and Bingsheng He. Deltaboost: Gradient boosting decision trees with efficient machine unlearning. *Proc. ACM Manag. Data*, 1(2):168:1–168:26, 2023.

Lei Xu, Maria Skoularidou, Alfredo Cuesta-Infante, and Kalyan Veeramachaneni. Modeling tabular data using conditional GAN. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems, NeurIPS*, pp. 7333–7343, 2019.

Ming Zheng, Tong Li, Rui Zhu, Yahui Tang, Mingjing Tang, Leilei Lin, and Zifei Ma. Conditional wasserstein generative adversarial network-gradient penalty-based approach to alleviating imbalanced data classification. *Inf. Sci.*, 512:1009–1023, 2020.

Yan Zhou, Murat Kantarcioglu, and Chris Clifton. On improving fairness of AI models with synthetic minority oversampling techniques. In *Proceedings of the 2023 SIAM International Conference on Data Mining, SDM*, pp. 874–882, 2023.

Lianghui Zhu, Bencheng Liao, Qian Zhang, Xinlong Wang, Wenyu Liu, and Xinggang Wang. Vision mamba: Efficient visual representation learning with bidirectional state space model. In *Forty-first International Conference on Machine Learning, ICML*, volume 235, pp. 62429–62442, 2024.

Tuanfei Zhu, Xinwang Liu, and En Zhu. Oversampling with reliably expanding minority class regions for imbalanced data learning. *IEEE Trans. Knowl. Data Eng.*, 35(6):6167–6181, 2023.

A Appendix

A.1 Algorithm Details

We provide the detailed procedures of the proposed framework in Algorithms 1, 2, 3, and 4.

A.1.1 ALGORITHM 1: TRAINING PROCESS OF TABMADO

Algorithm 1: Training Process of TabMaDo

Input : Training set: X; timesteps: T; schedule variables: $\{\beta_t\}_{t=1}^{T}$; Diffuser training iterations: N_D; Guider training iterations: N_G; batch size: m; minority class label: y_{min}; the prior prototype of the minority class: p_{min}; synthetic data size: K; scaling factor: s.

Output: Generated tabular data X_s.

$\{\alpha_t\}_{t=1}^{T} \leftarrow \{1-\beta_t\}_{t=1}^{T}$;

$\{\bar{\alpha}_t\}_{t=1}^{T} \leftarrow \left\{\prod_{i=1}^{t} \alpha_i\right\}_{t=1}^{T}$;

Train models:;

Initialize parameters θ of the Diffuser;

Trained Diffuser $\leftarrow$ **Algorithm 2** $(X, T, \{\bar{\alpha}_t\}_{t=1}^{T}, N_D)$;

Initialize parameters $\{\theta_x, \theta_{min}, \theta_h\}$ of the Guider;

Trained Guider $\leftarrow$ **Algorithm 3** $(X, T, m, \{\bar{\alpha}_t\}_{t=1}^{T}, N_G, p_{min})$;

Generate synthetic data:;

$X_s \leftarrow$ **Algorithm 4** (trained Diffuser, trained Guider, y_{min}, p_{min}, $\{\beta_t\}_{t=1}^{T}$, $\{\alpha_t\}_{t=1}^{T}$, $\{\bar{\alpha}_t\}_{t=1}^{T}$, K, T, s);

return X_s;

A.1.2 ALGORITHM 2: DIFFUSER TRAINING

Algorithm 2: Diffuser Training

Input : Training set: X; timesteps: T; schedule variables: $\{\bar{\alpha}_t\}_{t=1}^{T}$; Diffuser training iterations: N_D.

Output: A trained Diffuser with the optimal parameters θ.

for $n = 1$ **to** N_D **do**

  Randomly sample $x_0 \in X$;

  Randomly sample t from Uniform($\{1, 2, \ldots, T\}$);

  Sample noise $\epsilon_t \sim \mathcal{N}(0, \mathbf{I})$;

  $x_t \leftarrow \sqrt{\bar{\alpha}_t} x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon_t$;

  $\epsilon_\theta(x_t, t) \leftarrow$ Diffuser(x_t, t);

  $L_D \leftarrow \|\epsilon_t - \epsilon_\theta(x_t, t)\|^2$;

  Compute the gradient: $g \leftarrow \nabla_\theta L_D$;

  Update θ using stochastic gradient descent;

end

return *Trained Diffuser*;

A.1.3 ALGORITHM 3: GUIDER TRAINING

Algorithm 3: Guider Training

Input : Training set: X; timesteps: T; schedule variables: $\{\bar{\alpha}_t\}_{t=1}^{T}$; batch size: m; Guider training iterations: N_G; prototype weight: γ.

Output: A trained Guider with optimal parameters θ_x, θ_{min}, θ_h.

for $n = 1$ **to** N_G **do**
- Randomly sample m training data $\{(x^i, y^i)\}_{i=1}^{m} \in X$;
- Randomly sample t from Uniform($\{1, 2, \ldots, T\}$);
- Compute minority class prototype: $p_{\text{min}} = \mathbb{E}[x_{\text{min}}]$;
- **for** $i = 1$ **to** m **do**
  - Sample $x_t^i \sim q(x_t|x_0) = \mathcal{N}(x_t; \sqrt{\bar{\alpha}_t}x_0, (1 - \bar{\alpha}_t)\mathbf{I})$;
  - $h_x^i \leftarrow E_x(x_t^i, t)$;
  - $P(x_t^i) \leftarrow C(h_x^i)$;
- **end**
- $h_{\text{min}} \leftarrow E_{\text{min}}(p_{\text{min}})$;
- Compute cosine similarity: $\text{sim}^i = \cos(h_x^i, h_{\text{min}})$ for $i = 1, \ldots, m$;
- $L_{\text{class}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} \text{BCE}(P(x_t^i), y^i)$;
- $L_{\text{minority}} \leftarrow \frac{1}{|\mathcal{M}_{\text{min}}|}\sum_{i \in \mathcal{M}_{\text{min}}}(1 - \text{sim}^i)$;
- $L_{\text{majority}} \leftarrow \frac{1}{|\mathcal{M}_{\text{maj}}|}\sum_{i \in \mathcal{M}_{\text{maj}}} \text{ReLU}(\text{sim}^i)$;
- $L_{\text{proto}} \leftarrow L_{\text{minority}} + L_{\text{majority}}$;
- $L_G \leftarrow L_{\text{class}} + \gamma \cdot L_{\text{proto}}$;
- Compute gradients $\nabla_{\theta_x} L_G$, $\nabla_{\theta_{\text{min}}} L_G$ and $\nabla_{\theta_h} L_G$;
- Update θ_x, θ_{min} and θ_h using stochastic gradient descent;

end

return *Trained Guider*;

A.1.4 ALGORITHM 4: DATA SAMPLING

Algorithm 4: Data Sampling

Input : Trained Diffuser; trained Guider; the prior prototype of the minority class: p_{min}; timesteps: T; minority class label: y_{min}; schedule variables: $\{\beta_t\}_{t=1}^{T}$, $\{\alpha_t\}_{t=1}^{T}$, $\{\bar{\alpha}_t\}_{t=1}^{T}$; the size of synthetic data: K; scaling factor: s.

Output: Generated tabular data X_s.

Initialize $X_s \leftarrow \emptyset$;

for $k = 1$ **to** K **do**
- Randomly sample $x_t \sim \mathcal{N}(0, \mathbf{I})$;
- **for** $t = T$ **to** 1 **do**
  - $\epsilon_\theta(x_t, t) \leftarrow \text{Diffuser}(x_t, t)$;
  - $\mu_t \leftarrow \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(x_t, t)\right)$;
  - $g_t \leftarrow \nabla_{x_t}\text{Guider}(p_{min}, y_{min}, x_t, t)$;
  - $\mu_t^* \leftarrow \mu_t + s \cdot g_t$;
  - $\sigma_t^2 \leftarrow \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t$;
  - Randomly sample noise $z \sim \mathcal{N}(0, \mathbf{I})$;
  - $x'_{t-1} \leftarrow \mu_t^* + \sigma_t^2 \cdot z$;
- **end**
- $X_s \leftarrow X_s \cup \{x'_0\}$;

end

return X_s;

A.2 ROOF OF CONVERGENCE ANALYSIS

A.2.1 ASSUMPTIONS OF CONVERGENCE ANALYSIS

Assumption 1 (Gradient Lipschitz Smoothness)**.** *The gradients of L_{class} and L_{proto} are Lipschitz continuous. For any two parameter vectors θ_1, θ_2, there exist positive constants $\beta_1 > 0$, $\beta_2 > 0$ such that:*

$$\|\nabla L_{class}(\theta_2) - \nabla L_{class}(\theta_1)\| \leq \beta_1 \|\theta_2 - \theta_1\|,$$
$$\|\nabla L_{proto}(\theta_2) - \nabla L_{proto}(\theta_1)\| \leq \beta_2 \|\theta_2 - \theta_1\|.$$

The composite loss L_G is Lipschitz smoothness with $\beta = \max(\beta_1, \gamma\beta_2)$.

Assumption 2 (Gradient Boundedness)**.** *The expected squared norm of stochastic gradients for all parameters is uniformly bounded. Let $\theta \in \{\theta_x, \theta_{\min}, \theta_h\}$ denote the parameter sets of E_x, $E_{\min}$ and C, respectively. There exists $G > 0$ such that:*

$$\mathbb{E}\left[\|\nabla_\theta L_{class}\|^2\right] \leq G, \quad \mathbb{E}\left[\|\nabla_\theta L_{proto}\|^2\right] \leq G.$$

Assumption 3 (Encoder Lipschitz Continuity)**.** *Encoders E_x and $E_{\min}$ are L_E-Lipschitz continuous. For any inputs a, b:*

$$\|E_x(a) - E_x(b)\| \leq L_E \|a - b\|,$$
$$\|E_{\min}(a) - E_{\min}(b)\| \leq L_E \|a - b\|.$$

A.2.2 PROOF OF LEMMAS 1-3

Lemma 1 (Monotonicity of L_{class} with θ_h Update)**.** *At iteration t+1, fix θ_x and θ_{min}, and only update θ_h. Then $L_{class}(t+1) \leq L_{class}(t)$, and L_{proto} remains unchanged.*

Proof. L_{class} is a binary cross-entropy loss with respect to θ_h, satisfying non-negativity: $L_{class} \geq 0$. By Assumption 2, its gradient norm is bounded: $\| \nabla_{\theta_h} L_{class} \| \leq \sqrt{G}$. The update rule for θ_h at iteration *t* is:

$$\theta_h(t+1) = \theta_h(t) - \eta(t) \cdot \nabla_{\theta_h} L_{class}(\theta_h(t)),$$

where $\eta(t) > 0$ denotes the learning rate at iteration *t*. Under Assumption 1 (Lipschitz smoothness of L_{class}), for any two parameter vectors $\theta_h(t)$ and $\theta_h(t+1)$, the following inequality holds:

$$L_{class}(\theta_h(t+1)) \leq L_{class}(\theta_h(t)) + \langle \nabla L_{class}(\theta_h(t)), \Delta\theta_h \rangle + \frac{\beta_1}{2}\|\Delta\theta_h\|^2,$$

where $\Delta\theta_h = \theta_h(t+1) - \theta_h(t) = -\eta(t)\nabla_{\theta_h} L_{class}(\theta_h(t))$. Substitute $\Delta\theta_h$ into the above inequality:

$$L_{class}(t+1) \leq L_{class}(t) - \eta(t)\|\nabla_{\theta_h} L_{class}\|^2 + \frac{\beta_1 \eta(t)^2}{2}\|\nabla_{\theta_h} L_{class}\|^2.$$

Set the learning rate constraint $\eta(t) \leq \frac{1}{\beta_1}$, and the inequality can be further simplified as:

$$L_{class}(t+1) \leq L_{class}(t) - \frac{\eta(t)}{2}\|\nabla_{\theta_h} L_{class}\|^2 \leq L_{class}(t).$$

Since L_{proto} is independent of θ_h, it remains unchanged:

$$L_{proto}(t+1) = L_{proto}(t).$$

Thus, the overall loss satisfies:

$$L_G(t+1) = L_{class}(t+1) + \gamma L_{proto}(t+1) < L_G(t).$$

Hence, Lemma 1 is proved. □

Lemma 2 (Monotonicity of L_{proto} with θ_{min} Update)**.** *At iteration t+1, fix θ_x and θ_h, and only update the prototype encoder parameter θ_{min}. Then $L_{proto}(t+1) \leq L_{proto}(t)$, and L_{class} remains unchanged.*

Proof. The prototype loss L_{proto} depends on the cosine similarity $sim = cos(h_x, h_{\min})$, where $h_{\min} = E_{min}(p_{\min})$ Updating $\theta_{\min}$ modifies $h_{\min}$, thereby affecting the value of L_{proto}. By Assumption 2, the gradient norm of L_{proto} with respect to θ_{min} satisfies $\|\nabla_{\theta_{\min}} L_{proto}\| \leq \sqrt{G}$. The gradient descent update rule for $\theta_{\min}$ is:

$$\theta_{\min}(t+1) = \theta_{\min}(t) - \eta(t)\nabla_{\theta_{\min}} L_{proto}(\theta_{\min}(t)).$$

Note that L_{proto} is a sum of two convex-like terms: (i) $1 - sim$ for minority samples, which is convex in sim; (ii) $ReLU(sim)$ for majority samples, which is convex and non-decreasing. Since $sim = \cos(h_x, h_{\min})$ is Lipschitz in $h_{\min}$ (by Assumption 3), and the overall composition preserves Lipschitz smoothness, the following descent property holds with learning rate constraint $\eta(t) \leq \frac{1}{\beta_2}$:

$$L_{proto}(t+1) \leq L_{proto}(t) - \frac{\eta(t)}{2}\|\nabla_{\theta_{\min}} L_{proto}\|^2 \leq L_{proto}(t).$$

Since L_{class} is independent of θ_{min}, it remains unchanged. Thus, the composite loss satisfies:

$$L_G(t+1) = L_{class}(t+1) + \gamma L_{proto}(t+1) < L_G(t).$$

Hence, Lemma 2 is proved. □

Lemma 3 (Monotonicity of Composite Loss with θ_x Update)**.** *At iteration $t+1$, fix θ_h and $\theta_{\min}$, and update the sample encoder parameter θ_x. The total gradient of the composite loss with respect to θ_x is $\nabla_{\theta_x} L_G = \nabla_{\theta_x} L_{class} + \gamma \nabla_{\theta_x} L_{proto}$, and the composite loss satisfies $L_G(t+1) \leq L_G(t)$.*

Proof. By the Cauchy-Schwarz inequality and Assumption 2, the squared norm of the total gradient can be bounded as:

$$\begin{aligned}\|\nabla_{\theta_z} L_G\|^2 &= \|\nabla_{\theta_z} L_{\text{class}} + \gamma \nabla_{\theta_z} L_{\text{proto}}\|^2 \\ &\leq 2\left(\|\nabla_{\theta_z} L_{\text{class}}\|^2 + \gamma^2 \|\nabla_{\theta_z} L_{\text{proto}}\|^2\right) \\ &\leq 2(1+\gamma^2)G.\end{aligned}$$

Performing gradient descent with step size $\eta(t)$:

$$\theta_x(t+1) = \theta_x(t) - \eta(t) \cdot \nabla_{\theta_x} L_G(\theta_x(t)).$$

Based on Assumption 1 (Lipschitz smoothness of L_{class} and L_{proto}), with the learning rate constraint $\eta(t) \leq \frac{1}{\beta}$, the following inequalities for the two loss components can be derived:

$$L_{class}(t+1) \leq L_{class}(t) - \frac{\eta(t)}{2}\|\nabla_{\theta_x} L_{class}\|^2,$$

$$L_{proto}(t+1) \leq L_{proto}(t) - \frac{\eta(t)}{2}\|\nabla_{\theta_x} L_{proto}\|^2.$$

Multiply the second inequality by γ and sum the two inequalities:

$$\begin{aligned}L_G(t+1) &\leq L_G(t) - \frac{\eta(t)}{2}\left(\|\nabla_{\theta_x} L_{class}\|^2 + \gamma^2 \|\nabla_{\theta_z} L_{proto}\|^2\right) \\ &\leq L_G(t).\end{aligned}$$

Lemma 3 is proved. □

A.2.3 Proof of Theorem 1

Theorem 1 (Convergence of the Overall Loss)**.** *Under Assumptions 1–3, set the learning rate $\eta = \frac{1}{\beta_\xi}$ where $\xi = 2(1+\gamma^2)G$. For the composite loss sequence $\{L_G(\theta_t)\}$ generated by gradient descent updates, the loss sequence converges to a constant Loss*, i.e., $\lim_{t\to\infty} L_G(\theta_t) = Loss^*$. Meanwhile, the gradient norm of the loss function converges to 0: $\lim_{t\to\infty} \|\nabla L_G(\theta_t)\| = 0$.*

Proof. Lemmas 1–3 prove that the composite loss is non-increasing under updates of any parameter component. Since L_{class} and L_{proto} are non-negative cross-entropy losses, $L_G(\theta_t) \geq 0$ for all t. By the Monotone Convergence Theorem, the bounded non-increasing sequence $\{L_G(\theta_t)\}$ converges to a limit Loss*. From the descent inequality $L_G(\theta_t) - L_G(\theta_{t+1}) \geq \frac{\eta(t)}{2}\|\nabla L_G(\theta_t)\|^2$, summing from $t = 0$ to $t = T - 1$ gives:

$$L_G(\theta_0) - L_G(\theta_T) \geq \frac{\eta}{2} \sum_{t=0}^{T-1} \|\nabla L_G(\theta_t)\|^2.$$

Since the left-hand side is bounded by $L_G(\theta_0) - \text{Loss}^*$, the series $\sum_{t=0}^{\infty} \|\nabla L_G(\theta_t)\|^2$ converges, implying $\lim_{t\to\infty} \|\nabla L_G(\theta_t)\| = 0$. □

A.3 Details of experiments

A.3.1 Key characteristics of environment datasets

Table 2: A Summary of The Datasets.

Datasets	**Samples**	**Features**	N_{min}	**IR**
yeast6	1484	9	35	41.40
Wilt	4839	5	261	17.54
abalone9-18	731	8	42	16.4
ecoli4	336	6	20	15.8
glass-0-1-4-6-vs-2	205	10	17	11.06
vehicle0	846	17	199	3.25
haberman	306	3	81	2.78
phoneme	5404	4	1586	2.41
german	1000	19	300	2.33
magic	19020	9	6688	1.84

A.3.2 Baseline Descriptions

- stable_SMOTE Feng et al. (2021): A stable oversampling method that reduces randomness in traditional SMOTE by sequentially selecting samples, choosing neighbors based on distance, and performing uniform interpolation.
- TabDDPM Kotelnikov et al. (2023): Employs diffusion models to generate tabular data with arbitrary data types.
- TabDiT Hu et al. (2025): Integrates a Transformer architecture into the diffusion-based generation process for tabular data.
- CoDi Lee et al. (2023): A diffusion model designed to synthesize mixed-type tabular data.
- RelDDPM Liu et al. (2024): Utilizes a controller to steer the diffusion model for generating tabular data under specified conditions.
- GOIO Ren et al. (2025): Maps raw tabular data into a latent space via a VAE, then generates data using a conditional latent diffusion model.

A.4 Additional experiments results

A.4.1 Performance of Additional Classifiers

This part provides detailed performance metrics for GBDT and TabPFN, provided in Tables 3 and 4. Each table reports the mean and standard deviation of the corresponding metric across the 10 benchmark datasets. The best result for each dataset is highlighted in bold. And Fig. 4 visualizes the average performance and standard deviations across datasets and classifiers.

Table 3: Performance Comparison of Seven Oversampling Methods on GBDT.

Metric	Dataset	ssmote	tabddpm	codi	tabdit	relddpm	GOIO	TabMaDo
F1-score	yeast6	0.4402±0.17	0.3829±0.20	0.3317±0.27	0.3783±0.33	0.5548±0.15	0.4187±0.21	**0.5743±0.20**
	wilt	0.8498±0.04	0.8131±0.04	0.7756±0.06	0.8416±0.03	0.8654±0.03	0.1533±0.06	**0.8658±0.03**
	abalone9-18	0.3794±0.14	0.4659±0.19	0.3688±0.19	0.3276±0.20	0.4285±0.18	0.3023±0.18	**0.5002±0.20**
	ecoli4	0.7733±0.29	0.7433±0.30	0.7033±0.32	0.7800±0.30	**0.7933±0.30**	0.5267±0.30	0.7433±0.30
	glass-0-1-4-6-vs-2	0.3271±0.28	0.3467±0.30	0.2500±0.31	0.2167±0.35	0.3619±0.28	0.0500±0.15	**0.3700±0.29**
	vehicl0	**0.9464±0.03**	0.9424±0.03	0.9272±0.03	0.9351±0.03	0.9429±0.04	0.6101±0.18	0.9400±0.03
	haberman	0.4448±0.11	0.3378±0.09	0.3563±0.15	0.3010±0.10	0.3906±0.09	0.3214±0.12	**0.4465±0.08**
	phoneme	0.7629±0.02	0.7661±0.02	0.7552±0.02	0.7685±0.02	0.7714±0.01	0.2356±0.04	**0.7716±0.02**
	german	0.5581±0.06	0.5490±0.06	0.5225±0.06	0.5328±0.05	0.5586±0.06	**0.5732±0.07**	0.5598±0.08
	magic	0.8156±0.01	0.8163±0.01	0.8022±0.01	0.8020±0.01	0.8162±0.01	0.6148±0.21	**0.8176±0.01**
G-mean	yeast6	0.7658±0.14	0.6670±0.19	0.4553±0.33	0.4817±0.35	0.7768±0.14	0.7617±0.15	**0.7850±0.16**
	wilt	0.9609±0.02	0.9607±0.02	0.8398±0.05	0.8958±0.03	0.9547±0.03	0.4541±0.08	**0.9658±0.02**
	abalone9-18	0.6446±0.11	0.6294±0.16	0.4787±0.18	0.4289±0.23	0.6602±0.15	0.5904±0.24	**0.7288±0.13**
	ecoli4	0.8627±0.30	0.8078±0.30	0.8300±0.30	0.8366±0.30	**0.8642±0.30**	0.6980±0.37	0.8078±0.30
	glass-0-1-4-6-vs-2	0.4624±0.39	0.4670±0.39	0.2809±0.34	0.2395±0.37	**0.5069±0.34**	0.0687±0.21	0.5067±0.35
	vehicle0	0.9722±0.02	0.9646±0.02	0.9543±0.03	0.9604±0.02	0.9680±0.03	0.6723±0.14	**0.9735±0.02**
	haberman	0.5917±0.10	0.5012±0.08	0.4887±0.18	0.4590±0.10	0.5464±0.08	0.4829±0.11	**0.5929±0.07**
	phoneme	0.8532±0.02	0.8510±0.02	0.8211±0.02	0.8334±0.02	0.8521±0.01	0.3790±0.04	**0.8550±0.01**
	german	**0.7079±0.05**	0.6577±0.05	0.6378±0.05	0.6426±0.03	0.6728±0.05	0.6870±0.06	0.6774±0.07
	magic	0.8522±0.01	0.8522±0.01	0.8355±0.01	0.8351±0.01	0.8520±0.01	0.6637±0.22	**0.8523±0.01**
AUC	yeast6	0.9176±0.05	0.8791±0.07	0.8993±0.07	0.8936±0.07	0.9271±0.05	0.8991±0.08	**0.9354±0.06**
	wilt	0.9933±0.01	0.9915±0.01	0.9886±0.01	0.9908±0.01	0.9946±0.01	0.7426±0.03	**0.9957±0.00**
	abalone9-18	0.8334±0.07	0.8437±0.11	0.8047±0.12	0.8188±0.09	0.8438±0.08	0.7806±0.14	**0.8514±0.08**
	ecoli4	0.9452±0.16	**0.9920±0.02**	0.9921±0.01	0.9841±0.03	0.9871±0.03	0.8991±0.13	0.9839±0.04
	glass-0-1-4-6-vs-2	0.8941±0.07	0.8911±0.11	0.6601±0.25	0.7956±0.18	**0.9175±0.09**	0.5854±0.17	0.8984±0.08
	vehicle0	**0.9960±0.00**	0.9962±0.00	0.9948±0.00	0.9945±0.00	0.9943±0.01	0.9122±0.05	0.9956±0.00
	haberman	0.6267±0.13	0.6207±0.07	0.6339±0.10	0.6247±0.08	0.6351±0.09	0.5837±0.07	**0.6297±0.09**
	phoneme	0.9226±0.01	0.9249±0.01	0.9213±0.01	0.9257±0.01	0.9271±0.01	0.7131±0.04	**0.9264±0.01**
	german	**0.7810±0.05**	0.7800±0.05	0.7775±0.04	0.7799±0.05	0.7771±0.04	0.7776±0.07	0.7628±0.07
	magic	**0.9301±0.01**	0.9295±0.01	0.9263±0.01	0.9273±0.01	0.9296±0.01	0.7764±0.26	0.9291±0.01

Table 4: Performance Comparison of Seven Oversampling Methods on TabPFN.

Metric	Dataset	ssmote	tabddpm	codi	tabdit	relddpm	GOIO	TabMaDo
F1-score	yeast6	0.1400±0.23	0.3973±0.22	0.4857±0.20	0.3686±0.30	0.5342±0.20	0.3389±0.27	**0.5349±0.13**
	wilt	0.9872±0.02	0.9825±0.02	0.9924±0.01	0.9797±0.02	0.9828±0.02	0.9499±0.05	**0.9927±0.02**
	abalone9-18	0.0800±0.16	0.5392±0.11	0.5394±0.24	0.5037±0.13	**0.5585±0.14**	0.3862±0.22	0.5479±0.10
	ecoli4	0.5467±0.38	0.8123±0.30	0.8133±0.30	0.8167±0.30	0.7567±0.30	0.5471±0.32	**0.8173±0.30**
	glass-0-1-4-6-vs-2	0.0500±0.15	0.1833±0.32	0.1567±0.25	0.3167±0.32	0.0667±0.20	0.0400±0.12	**0.3633±0.34**
	vehicle0	0.9872±0.02	0.9825±0.02	0.9924±0.01	0.9797±0.02	0.9828±0.02	0.9499±0.05	**0.9927±0.02**
	haberman	0.5007±0.14	0.4721±0.07	0.3052±0.19	0.2952±0.15	0.4887±0.15	0.2774±0.11	**0.5356±0.11**
	phoneme	0.7843±0.26	0.8730±0.01	0.8782±0.02	**0.8799±0.01**	0.8709±0.01	0.6744±0.03	0.8725±0.01
	german	0.4888±0.17	0.5722±0.07	0.5406±0.07	0.5729±0.06	0.5823±0.06	0.5737±0.07	**0.5838±0.06**
	magic	0.8405±0.01	**0.8578±0.01**	0.8549±0.01	0.8553±0.01	0.8553±0.00	0.7849±0.03	0.8560±0.01
G-mean	yeast6	0.1705±0.27	0.5866±0.23	0.6239±0.16	0.4410±0.31	0.7983±0.15	0.6307±0.27	**0.8362±0.13**
	wilt	0.9890±0.01	0.9892±0.01	0.9958±0.01	0.9865±0.02	0.9911±0.01	0.9614±0.04	**0.9960±0.01**
	abalone9-18	0.1000±0.20	0.6882±0.11	0.6452±0.26	0.6218±0.12	0.7582±0.13	0.6163±0.24	**0.8169±0.08**
	ecoli4	0.5813±0.40	0.8398±0.30	0.8398±0.30	0.8591±0.30	0.8355±0.30	0.6991±0.37	**0.8659±0.30**
	glass-0-1-4-6-vs-2	0.0688±0.21	0.2336±0.37	0.2064±0.32	0.3517±0.35	0.0707±0.21	0.0669±0.20	**0.4686±0.40**
	vehicle0	0.9890±0.01	0.9892±0.01	0.9958±0.01	0.9865±0.02	0.9911±0.01	0.9614±0.04	**0.9960±0.01**
	haberman	0.6470±0.12	0.6176±0.06	0.4334±0.20	0.4186±0.17	0.6324±0.13	0.4360±0.10	**0.6745±0.09**
	phoneme	0.8140±0.27	0.9181±0.01	0.9101±0.01	0.9122±0.01	0.9160±0.01	0.7300±0.02	**0.9201±0.01**
	german	0.5831±0.20	0.6714±0.05	0.6454±0.06	0.6704±0.05	0.6912±0.05	0.6821±0.06	**0.6958±0.05**
	magic	0.8733±0.01	0.8829±0.01	0.8784±0.00	0.8786±0.00	0.8815±0.00	0.8176±0.03	**0.8830±0.00**
AUC	yeast6	0.4051±0.41	0.7839±0.28	0.9284±0.06	0.9129±0.07	0.9343±0.06	0.8958±0.08	**0.9404±0.07**
	wilt	0.9997±0.00	0.9996±0.00	0.9997±0.00	0.9995±0.00	0.9995±0.00	0.9964±0.01	**0.9997±0.00**
	abalone9-18	0.8934±0.10	0.9227±0.06	0.9461±0.05	0.9336±0.06	0.9362±0.05	0.8792±0.08	**0.9477±0.04**
	ecoli4	0.9694±0.09	0.9912±0.01	0.9920±0.02	0.9932±0.01	0.9920±0.02	0.9607±0.05	**0.9936±0.01**
	glass-0-1-4-6-vs-2	0.8841±0.11	**0.9278±0.04**	0.8620±0.14	0.9072±0.07	0.8692±0.12	0.7101±0.11	0.9098±0.09
	vehicle0	0.9997±0.00	0.9996±0.00	0.9997±0.00	0.9995±0.00	0.9995±0.00	0.9964±0.01	**0.9997±0.00**
	haberman	0.6840±0.12	0.6900±0.10	0.5914±0.24	0.6797±0.10	0.7042±0.13	0.5987±0.08	**0.7266±0.10**
	phoneme	0.8794±0.29	0.9766±0.00	0.9766±0.00	0.9764±0.00	0.9768±0.00	0.9258±0.02	**0.9774±0.00**
	german	0.7066±0.24	0.8046±0.06	0.8060±0.06	**0.8082±0.06**	0.7833±0.05	0.7882±0.06	0.7878±0.06
	magic	0.9446±0.00	0.9540±0.00	**0.9542±0.00**	0.9541±0.00	0.9536±0.00	0.9239±0.01	0.9535±0.00

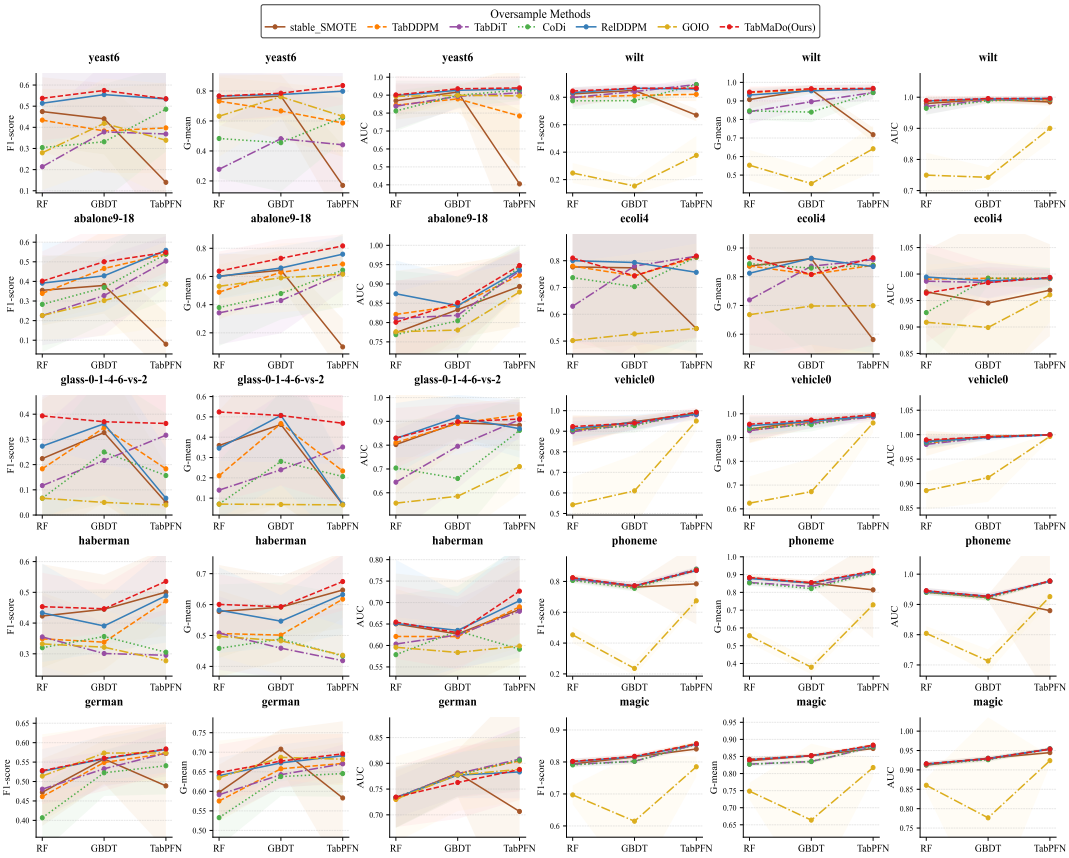


Figure 4: A line chart displaying experimental results that illustrates the average metric and standard deviation confidence intervals.

A.4.2 Evaluation of Synthetic Sample Fidelity, Diversity, and Distribution Matching

This part supplement eight quantitative metrics to evaluate the generated minority class samples, including WD, JS, L2, KL, MMD, KS, KNN, and Coverage. Average ranks (lower = better) across datasets are summarized in Table 5.

Table 5: Average ranks of synthetic sample fidelity, diversity, and distribution matching metrics across datasets.

Method	WD↓	JS↓	L2↓	KL↓	MMD↓	KS↓	KNN↑	Coverage↓
TabMaDo (Ours)	1.73	1.60	2.78	2.82	1.92	1.55	3.69	1.70
GOIO	3.36	2.20	2.22	2.27	2.77	3.00	4.00	3.30
TabDDPM	2.55	3.40	3.67	3.64	2.85	3.45	5.00	1.70
RelDDPM	3.18	4.40	3.33	4.00	3.00	3.55	3.38	2.80
TabDiT	4.64	5.00	4.33	4.45	5.31	5.18	3.85	4.20
CoDi	5.55	4.40	4.67	3.82	5.15	4.27	1.08	4.40

- **WD (Wasserstein Distance):** Measures the overall distribution difference in numerical features between synthetic and real data.
- **JS (Jensen–Shannon Divergence):** Measures the probability distribution difference in categorical features between synthetic and real data.
- **L2 (L2 Distance of Correlation Matrix):** Measures the difference in feature correlation structures between synthetic and real data.

- **KL (Kullback–Leibler Divergence):** Measures the information loss when approximating the real data distribution with the synthetic data distribution.
- **MMD (Maximum Mean Discrepancy):** Measures the distribution difference between synthetic and real data in high-dimensional feature space.
- **KS (Kolmogorov–Smirnov Statistic):** Measures the maximum difference between the cumulative distribution functions of synthetic and real data.
- **KNN (K-Nearest Neighbors Distance):** Measures the diversity within synthetic data.
- **Coverage (Coverage Density):** Measures the proportion of the real data distribution space covered by synthetic data.

A.4.3 Statistical Analysis

First, each method is ranked based on its evaluation metric scores on each test set, and the mean rank across all classifiers, metrics, and datasets is then computed. The average ranking scores are detailed in Table 6. With the significance level α set to 0.05, the critical value F_α is 2.2172. All computed $\mathcal{T}_F$ values exceed F_α, confirming that significant performance differences exist among the seven oversampling methods.

The Nemenyi post-hoc test is then conducted using TabMaDo as the control method. The Critical Difference (CD) is calculated as 2.4078. If the difference between the average ranks of two methods exceeds CD, their performance is considered statistically different. The detailed results of the Nemenyi post-hoc test using TabMaDo as the control method in Table 7.

Table 6: Mean Ranking Scores of Seven Oversampling Methods.

Classifier	Metric	stable_SMOTE	TabDDPM	CoDi	TabDiT	RelDDPM	GOIO	TabMaDo (Ours)
RF	F1-score	3.400	4.000	5.400	5.650	2.300	6.250	**1.000**
	G-mean	3.200	3.800	5.600	5.450	2.800	5.950	**1.200**
	AUC	4.300	3.700	4.300	5.500	2.300	6.200	**1.700**
GBDT	F1-score	3.300	3.850	5.100	5.700	2.300	6.000	**1.821**
	G-mean	2.350	4.000	5.600	5.700	2.700	5.900	**1.750**
	AUC	3.200	3.600	4.600	4.600	2.600	6.500	**2.900**
TabPFN	F1-score	5.400	3.800	4.050	3.500	3.450	6.300	**1.500**
	G-mean	5.800	3.450	4.700	4.150	3.100	5.800	**1.000**
	AUC	5.100	3.750	3.700	3.200	3.850	6.300	**2.100**

Table 7: Statistical Significance Comparison of Different Methods Across Three Classifiers and Three Metrics

Classifier	Metric	Superior but not significantly superior	Superior and significantly superior
RF	F1-score	stable_SMOTE and RelDDPM	TabDDPM, CoDi, TabDiT and GOIO
	G-mean	stable_SMOTE, TabDDPM, and RelDDPM	CoDi, TabDiT and GOIO
	AUC	stable_SMOTE, TabDDPM, CoDi and RelDDPM	TabDiT and GOIO
GBDT	F1-score	stable_SMOTE, TabDDPM, and RelDDPM	CoDi, TabDiT and GOIO
	G-mean	stable_SMOTE, TabDDPM, and RelDDPM	CoDi, TabDiT and GOIO
	AUC	stable_SMOTE, TabDDPM, CoDi, TabDiT and RelDDPM	GOIO
TabPFN	F1-score	TabDDPM, TabDiT, CoDi and RelDDPM	stable_SMOTE and GOIO
	G-mean	TabDDPM and RelDDPM	stable_SMOTE, CoDi, TabDiT and GOIO
	AUC	TabDDPM, TabDiT, CoDi and RelDDPM	stable_SMOTE and GOIO

A.4.4 Ablation Study Results on Additional Classifiers

This part provides the detailed results of the ablation study for the three classifiers RF, GBDT and TabPFN. Tables 8, 9 and 10, report the mean F1-score, G-mean, and AUC of the full TabMaDo framework compared to its four ablated variants across twelve datasets. The best result for each dataset and metric is highlighted in bold.

Table 8: Ablation Study Results of TabMaDo Across Ten Benchmark Datasets on RF

Metric	Dataset	V1	V2	V3	V4	TabMaDo (Ours)
F1-score	yeast6	0.4919±0.21	0.3067±0.30	0.2567±0.22	0.3167±0.28	**0.5372±0.20**
	wilt	0.8437±0.04	0.8097±0.05	0.8203±0.06	0.8302±0.06	**0.8454±0.03**
	abalone9-18	0.3549±0.10	0.1967±0.20	0.2790±0.20	0.2486±0.22	**0.4015±0.15**
	ecoli4	**0.8300±0.32**	0.6333±0.35	0.5500±0.39	0.7667±0.30	0.8100±0.31
	glass-0-1-4-6-vs-2	0.1833±0.28	0.0667±0.20	0.0667±0.20	0.0500±0.15	**0.3933±0.33**
	vehicle0	0.9110±0.04	0.8900±0.07	0.9148±0.04	0.9102±0.04	**0.9227±0.05**
	haberman	0.4461±0.12	0.3098±0.19	0.3710±0.09	0.2733±0.17	**0.4533±0.11**
	phoneme	0.8187±0.02	0.7972±0.02	0.7992±0.01	0.8125±0.02	**0.8238±0.01**
	german	**0.5289±0.09**	0.3787±0.08	0.3909±0.08	0.3500±0.11	0.5279±0.09
	magic	0.7997±0.01	0.7855±0.01	0.7868±0.01	0.7936±0.01	**0.8016±0.01**
G-mean	yeast6	0.6483±0.26	0.3630±0.32	0.3229±0.27	0.3676±0.31	**0.7664±0.17**
	wilt	0.9449±0.04	0.8528±0.04	0.8631±0.06	0.8703±0.05	**0.9478±0.03**
	abalone9-18	0.6049±0.10	0.2640±0.27	0.3572±0.24	0.3098±0.26	**0.6378±0.14**
	ecoli4	**0.8680±0.30**	0.6536±0.35	0.5817±0.40	0.7828±0.30	0.8664±0.30
	glass-0-1-4-6-vs-2	0.2102±0.32	0.0707±0.21	0.0707±0.21	0.0688±0.21	**0.5241±0.44**
	vehicle0	0.9552±0.03	0.9195±0.05	0.9416±0.03	0.9366±0.03	**0.9561±0.04**
	haberman	0.6002±0.10	0.4443±0.20	0.5255±0.08	0.3992±0.22	**0.6005±0.10**
	phoneme	0.8766±0.01	0.8400±0.02	0.8437±0.01	0.8546±0.01	**0.8831±0.01**
	german	0.6432±0.07	0.5063±0.06	0.5214±0.06	0.4792±0.10	**0.6479±0.07**
	magic	0.8386±0.01	0.8205±0.01	0.8224±0.01	0.8279±0.00	**0.8405±0.01**
AUC	yeast6	0.8573±0.09	0.8193±0.13	0.8373±0.10	0.8199±0.09	**0.9014±0.11**
	wilt	**0.9891±0.01**	0.9701±0.02	0.9731±0.03	0.9675±0.03	0.9888±0.02
	abalone9-18	0.8044±0.11	0.7727±0.15	0.8017±0.12	0.7688±0.14	**0.8009±0.11**
	ecoli4	0.9629±0.01	0.9389±0.11	**0.9703±0.08**	0.9694±0.09	0.9639±0.09
	glass-0-1-4-6-vs-2	0.8222±0.13	0.8115±0.17	0.8273±0.13	0.6685±0.25	**0.8298±0.13**
	vehicle0	0.9868±0.02	0.9866±0.02	**0.9920±0.01**	0.9909±0.01	0.9894±0.01
	haberman	0.6451±0.10	0.6449±0.11	0.6312±0.09	0.6330±0.12	**0.6544±0.09**
	phoneme	0.9451±0.01	0.9452±0.01	0.9403±0.01	0.9416±0.01	**0.9453±0.01**
	german	0.7318±0.07	0.7025±0.05	0.6951±0.05	0.7249±0.05	**0.7344±0.06**
	magic	0.9129±0.00	0.9123±0.00	0.9132±0.01	0.9145±0.01	**0.9156±0.01**

Table 9: Ablation Study Results of TabMaDo Across Fourteen Benchmark Datasets on GBDT.

Metric	Dataset	V1	V2	V3	V4	TabMaDo
F1-score	yeast6	0.4383±0.22	0.3550±0.26	0.3467±0.28	0.3355±0.27	**0.5743±0.20**
	wilt	0.8627±0.03	0.8572±0.03	0.8442±0.05	0.8580±0.04	**0.8658±0.03**
	abalone9-18	0.4162±0.15	0.3565±0.17	0.3762±0.10	0.4002±0.18	**0.5002±0.20**
	ecoli4	0.7100±0.29	0.7133±0.28	0.7767±0.31	**0.7933±0.30**	0.7433±0.30
	glass-0-1-4-6-vs-2	0.3533±0.31	0.1167±0.24	0.2133±0.33	0.1300±0.27	**0.3700±0.29**
	vehicle0	0.9394±0.02	**0.9488±0.04**	0.9370±0.05	0.9315±0.04	0.9400±0.03
	haberman	0.4674±0.09	0.2956±0.12	0.3057±0.14	0.3119±0.16	**0.4465±0.08**
	phoneme	0.7780±0.02	0.7641±0.02	0.7541±0.03	0.7702±0.03	**0.7716±0.02**
	german	**0.5835±0.07**	0.5222±0.09	0.5171±0.05	0.5365±0.07	0.5598±0.08
	magic	0.8194±0.01	0.8052±0.01	0.8061±0.01	0.8080±0.01	**0.8176±0.01**
G-mean	yeast6	0.6286±0.26	0.4683±0.33	0.4405±0.31	0.4477±0.31	**0.7850±0.16**
	wilt	0.9602±0.02	0.9103±0.03	0.9015±0.05	0.9082±0.04	**0.9658±0.02**
	abalone9-18	0.6690±0.14	0.4779±0.18	0.5441±0.09	0.5595±0.13	**0.7288±0.13**
	ecoli4	0.8045±0.30	0.7519±0.29	0.8371±0.30	0.8382±0.30	0.8078±0.30
	glass-0-1-4-6-vs-2	0.4697±0.40	0.1394±0.28	0.2388±0.37	0.1662±0.34	**0.5067±0.35**
	vehicle0	0.9718±0.02	0.9645±0.03	0.9609±0.03	0.9541±0.03	**0.9735±0.02**
	haberman	0.6107±0.08	0.4480±0.11	0.4606±0.13	0.4454±0.19	**0.5929±0.07**
	phoneme	0.8581±0.01	0.8255±0.02	0.8182±0.03	0.8292±0.02	**0.8550±0.01**
	german	**0.6936±0.06**	0.6273±0.08	0.6270±0.04	0.6435±0.06	0.6774±0.07
	magic	0.8547±0.01	0.8366±0.01	0.8374±0.01	0.8391±0.01	**0.8523±0.01**
AUC	yeast6	0.9290±0.05	0.9107±0.06	0.9166±0.06	0.9045±0.07	**0.9354±0.06**
	wilt	0.9935±0.01	0.9939±0.00	0.9934±0.00	0.9867±0.02	**0.9957±0.00**
	abalone9-18	0.8450±0.09	0.8103±0.13	0.8502±0.08	0.8169±0.08	**0.8514±0.08**
	ecoli4	**0.9936±0.01**	0.9872±0.03	0.9792±0.05	0.9808±0.04	0.9839±0.04
	glass-0-1-4-6-vs-2	0.8819±0.10	0.8484±0.16	**0.9038±0.05**	0.8671±0.08	0.8984±0.08
	vehicle0	0.9946±0.01	0.9955±0.00	0.9957±0.00	0.9948±0.00	**0.9956±0.00**
	haberman	**0.6584±0.09**	0.6188±0.08	0.6319±0.10	0.6120±0.09	0.6297±0.09
	phoneme	0.9274±0.01	0.9305±0.01	0.9279±0.01	**0.9328±0.01**	0.9264±0.01
	german	0.7819±0.06	0.7860±0.06	**0.7916±0.05**	0.7846±0.05	0.7628±0.07
	magic	0.9298±0.01	0.9280±0.01	0.9278±0.01	**0.9300±0.01**	0.9291±0.01

Table 10: Ablation Study Results of TabMaDo Across Fourteen Benchmark Datasets on TabPFN

Metric	Dataset	V1	V2	V3	V4	TabMaDo
F1-score	yeast6	0.5277±0.19	0.4671±0.22	0.4533±0.12	0.4433±0.30	**0.5349±0.13**
	wilt	0.8699±0.04	0.9021±0.03	0.9059±0.03	0.9074±0.03	**0.9927±0.02**
	abalone9-18	0.4332±0.16	0.5432±0.22	**0.6071±0.15**	0.5062±0.17	0.5479±0.10
	ecoli4	0.8133±0.30	0.7800±0.30	0.8133±0.30	0.8133±0.30	**0.8173±0.30**
	glass-0-1-4-6-vs-2	0.1733±0.27	0.2067±0.26	0.1567±0.25	0.1467±0.29	**0.3633±0.34**
	vehicle0	0.9850±0.02	0.9800±0.02	0.9850±0.03	0.9806±0.03	**0.9927±0.02**
	haberman	0.5011±0.13	0.2998±0.18	0.2543±0.14	0.3955±0.16	**0.5356±0.11**
	phoneme	0.8802±0.02	0.8728±0.01	0.8638±0.01	0.8753±0.01	**0.8725±0.01**
	german	**0.5891±0.06**	0.5138±0.06	0.5110±0.06	0.5409±0.09	0.5838±0.06
	magic	0.8553±0.01	0.8481±0.01	0.8461±0.01	0.8521±0.01	**0.8560±0.01**
G-mean	yeast6	0.7370±0.16	0.5490±0.22	0.5620±0.09	0.5173±0.30	**0.8362±0.13**
	wilt	0.9698±0.02	0.9479±0.03	0.9574±0.03	0.9516±0.03	**0.9960±0.01**
	abalone9-18	0.6782±0.14	0.6345±0.25	0.7175±0.15	0.6344±0.15	**0.8169±0.08**
	ecoli4	0.8659±0.30	0.8105±0.30	0.8398±0.30	0.8398±0.30	**0.8659±0.30**
	glass-0-1-4-6-vs-2	0.2083±0.32	0.2751±0.34	0.2064±0.32	0.1680±0.34	**0.4686±0.40**
	vehicle0	0.9900±0.01	0.9867±0.02	0.9900±0.02	0.9904±0.02	**0.9960±0.01**
	haberman	0.6401±0.11	0.4206±0.19	0.3816±0.16	0.5261±0.13	**0.6745±0.09**
	phoneme	0.9220±0.01	0.9031±0.01	0.8979±0.01	0.9062±0.01	**0.9201±0.01**
	german	**0.6966±0.05**	0.6191±0.05	0.6171±0.05	0.6405±0.07	0.6958±0.05
	magic	0.8816±0.00	0.8716±0.01	0.8702±0.00	0.8749±0.01	**0.8830±0.00**
AUC	yeast6	0.9241±0.07	0.9396±0.06	0.9311±0.06	0.9352±0.06	**0.9404±0.07**
	wilt	0.9944±0.01	0.9962±0.01	0.9943±0.01	0.9953±0.01	**0.9997±0.00**
	abalone9-18	0.9164±0.05	0.9462±0.05	**0.9510±0.06**	0.9508±0.04	0.9477±0.04
	ecoli4	0.9920±0.02	0.9904±0.02	0.9872±0.02	0.9888±0.03	**0.9936±0.01**
	glass-0-1-4-6-vs-2	0.8791±0.13	0.9020±0.08	0.8830±0.08	0.9016±0.06	**0.9098±0.09**
	vehicle0	0.9996±0.00	0.9995±0.00	0.9996±0.00	0.9996±0.00	**0.9997±0.00**
	haberman	0.7020±0.12	0.7211±0.08	0.7213±0.09	0.7101±0.11	**0.7266±0.10**
	phoneme	0.9773±0.00	0.9771±0.00	0.9763±0.00	0.9770±0.00	**0.9774±0.00**
	german	0.7851±0.05	0.7993±0.05	0.7914±0.06	**0.8015±0.05**	0.7878±0.06
	magic	0.9539±0.00	0.9512±0.00	0.9511±0.00	0.9526±0.00	**0.9535±0.00**

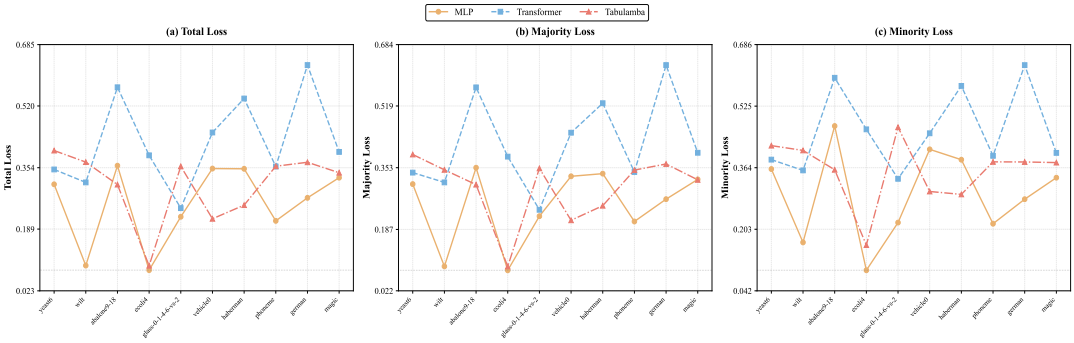


Figure 5: A line chart displaying experimental results that illustrates the average metric and standard deviation confidence intervals.

A.4.5 Additional Feature Order Experiments

This part evaluates Tabulamba's sensitivity to feature order. We randomly shuffled the feature sequence of the *yeast6* dataset and used the diffusion loss (MSE between predicted and true noise) as the metric. Results in Table 11 show that Tabulamba is largely insensitive to feature order variations.

And this part examines whether Tabulamba preserves minority-class information during diffusion. We compare it with MLP and Transformer backbones, with results in Fig. 5. Tabulamba achieves the lowest total, majority, and minority losses in most cases, indicating that it mitigates majority class dominance and enables balanced distribution learning.

Table 11: Loss comparison across different shuffled datasets

Dataset	Total Loss	Majority Loss	Minority Loss
yeast6	0.3086±0.0031	0.3075±0.0030	0.3586±0.0168
disorder1	0.3095±0.0023	0.3083±0.0024	0.3585±0.0118
disorder2	0.3105±0.0030	0.3093±0.0031	0.3459±0.0138