Breaking the Quality–Privacy Tradeoff in Tabular Data Generation via In-Context Learning

Xinyan Han[1*] **Yan Lu**[1*] **Xiaoyu Lin**[1] **Yuanyuan Jiang**[1]
Yuanrui Wang[1] **Xuanyue Li**[2] **Wenchao Zou**[3] **Xingxuan Zhang**[1†]
[1]Tsinghua University [2]Southeast University [3]Siemens China
{han-xy25,luyan22,amy_jyy,wang-yr23}@mails.tsinghua.edu.cn
clementinexiaoyu@gmail.com Lixy_ai@163.com
wenchao.zou@siemens.com xingxuanzhang@hotmail.com

Abstract

Tabular data synthesis aims to generate high-quality data while preserving privacy. However, we find that existing tabular generative models exhibit a clear tradeoff in the small-data regime: improving data quality typically comes at the cost of increased memorization of training samples, thereby weakening privacy protection. This tradeoff arises because small training sets make it difficult for dataset-specific generative models to distinguish generalizable structure from sample-specific patterns. To address this, we propose **DiffICL**, which formulates tabular data generation as an in-context learning problem. Instead of fitting each dataset from scratch, DiffICL leverages pretrained structural priors learned from a large collection of datasets, enabling it to infer data distributions from limited context rather than memorizing individual samples. We evaluate DiffICL on 14 real-world datasets. Results show that DiffICL improves both data quality and privacy, and generates synthetic data that provides effective data augmentation. Our findings suggest that the quality–privacy tradeoff can be improved through better training paradigms.

1 Introduction

Tabular data is widely used across high-stakes domains such as healthcare, finance, and public administration, where data sharing is often hindered by privacy concerns and regulatory constraints [8, 6, 31, 7, 27]. This limitation creates a strong demand for privacy-preserving data sharing mechanisms, making tabular data synthesis a promising solution. Tabular data synthesis involves two key objectives: generating high-quality samples that faithfully capture the underlying data distribution to ensure strong downstream utility, and mitigating privacy risks to protect sensitive individual-level information [11, 36]. To achieve these goals, deep generative models have been widely adopted to model the complex and heterogeneous distributions of tabular data. Existing approaches can be broadly categorized into three main lines of work: GAN-based methods [47, 32, 46, 52, 53], VAE-based methods [46, 23, 2], and the recently popular Diffusion-based methods [17, 19, 49, 20, 21, 38].

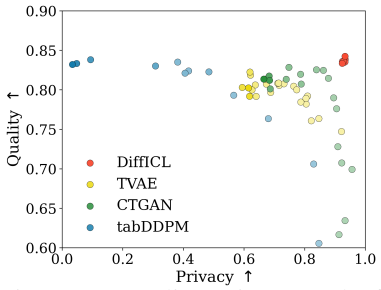


Figure 1: Quality–privacy tradeoff frontiers.

Despite the recent progress in tabular data generation, achieving a favorable tradeoff between data quality and privacy remains particularly challenging in small-data regimes. To characterize this issue,

*Equal contribution.
†Corresponding author.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

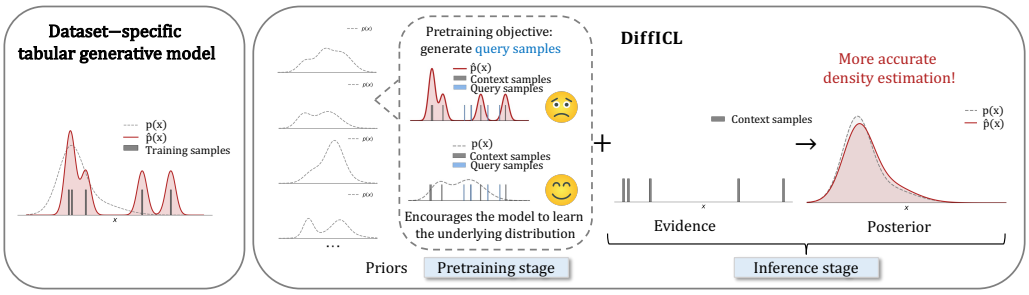


Figure 2: ICL pretraining enables more accurate density estimation from limited data by learning transferable structural priors. **Left**: standard dataset-specific generative model overfits sparse observations and yields a poor estimate of the underlying distribution. **Right**: at the pretraining stage, the model learns to generate query samples D_{qry} conditioned on context samples D_{ctx}, which requires inferring the underlying distribution instead of copying the context; the diverse distributions encountered during pretraining provide the priors. At the inference stage, the model treats the real dataset as context, infers the target distribution from D_{ctx} and generates new synthetic samples.

we evaluate representative methods from different paradigms—tabDDPM [19], TVAE [46], and CTGAN [46]—on the Adult [18] dataset with $N = 200$ training samples. For each baseline, we vary the number of training steps to examine how generation quality and privacy protection change during optimization in Figure 1. Each point represents a model checkpoint obtained after a specific number of training steps. Darker points correspond to later checkpoints. Diffusion-based (blue) models exhibit strong fitting capacity and can achieve high sample quality, yet this comes at the cost of severe overfitting: they generate samples that are overly close to the training records. This memorization can reveal the presence of individual training records or expose record-specific information, directly weakening privacy. In contrast, VAE-based (yellow) and GAN-based (green) methods introduce stronger architectural regularization that mitigates memorization, but their limited expressiveness yields substantially inferior generation quality. Under limited data, existing approaches fail to simultaneously achieve high fidelity and strong privacy protection, instead trading one for the other along a suboptimal Pareto front. This tradeoff arises because limited samples provide only a noisy and incomplete view of the underlying distribution. Due to the curse of dimensionality, small datasets provide only sparse coverage of the data space, making reliable inference of global distribution difficult.

To address this limitation, we turn to in-context learning (ICL) pretraining. In predictive tasks, tabular foundation models [13, 15, 25, 10, 51, 50] have demonstrated that ICL pretraining encodes rich structural priors into the network, enabling strong performance on small context sets without task-specific retraining. We argue that leveraging such pretrained priors is the key to solving the memorization problem of generative models in data scarce scenarios: Standard generative models optimize directly on the empirical data; with small sample sizes, deceptively low loss can be easily achieved through simple memorization. ICL pretraining removes the memorization incentive: each dataset is randomly partitioned into a context set D_{ctx} and a disjoint query set D_{qry}, and the model is trained to generate D_{qry} conditioned on D_{ctx}. Because D_{ctx} and D_{qry} are i.i.d., memorizing D_{ctx} cannot yield a lower pretraining loss than genuinely learning the underlying distribution. The model is thus driven to infer the underlying distribution rather than memorize the context, acquiring a prior for "distributions given a context." At inference, the scarce real dataset is fed as D_{ctx}, enabling generation that is faithful to the context distribution yet distinct from individual records, thereby improving privacy. We implement this idea in DiffICL. To handle the heterogeneity of tabular data—where tables differ in size, column semantics, and types—DiffICL projects raw features into a unified latent space via a tabular foundation model, then trains a dual-axis attention Transformer to perform conditional generation over these latent embeddings. As shown in Figure 1 (red), DiffICL establishes a superior Pareto frontier over existing paradigms. We summarize our main contributions as follows:

1. We propose DiffICL, a generative in-context learning framework that pretrains a context-conditioned latent diffusion model across heterogeneous tabular datasets.(Section 3).
2. We evaluate DiffICL on 14 real-world datasets covering both classification and regression tasks. DiffICL achieves markedly better data quality and privacy protection than representative methods from each major generative paradigm on average (Section 4.2).
3. DiffICL generates synthetic data that serves as effective data augmentation, improving downstream task performance when added to real training data (Section 4.3). Ablation

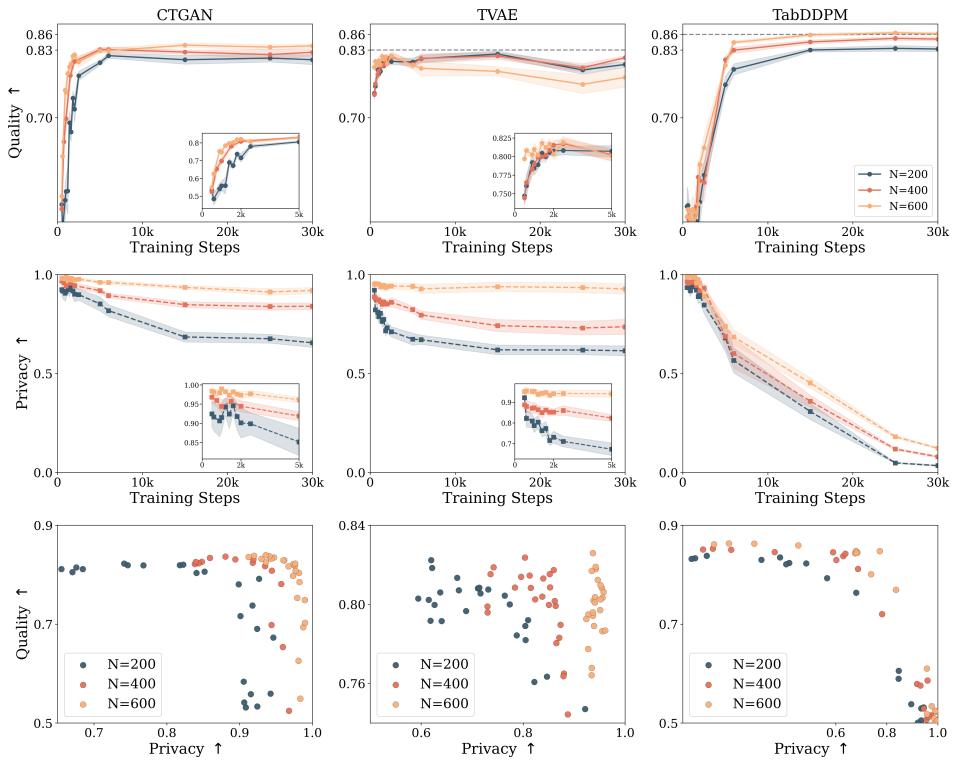


Figure 3: Training dynamics and quality–privacy tradeoffs across dataset sizes ($N = 200, 400, 600$) on the Adult dataset. **Top** and **middle** rows show the evolution of generation quality (AUC) and privacy (DCROverfit) over training steps for CTGAN, TVAE, and tabDDPM. **Bottom** row plots the resulting quality–privacy frontiers, where each point corresponds to a training checkpoint.

results further show that this benefit depends on the structural priors acquired during pretraining, which guide the generation process at inference time and produce more informative augmentation samples (Section 4.4).

2 Quality–Privacy Tradeoff in Tabular Generative Models

In this section, we examine the tradeoff between generation quality and privacy risk on the Adult dataset with $N \in \{200, 400, 600\}$. We evaluate multiple generative models at different stages of training. Generation quality is measured by the AUC of an XGBoost classifier trained on synthetic data and evaluated on real test data. Privacy risk is quantified based on DCR, a widely used privacy metric for tabular synthetic data [52, 19, 49, 38]. For a synthetic sample x_{syn} and a reference set D, we denote its distance to the closest record as $\text{DCR}(x_{\text{syn}}, D)$. Let p denote the proportion of synthetic samples whose DCR to the training set is smaller than DCR to the held-out validation set:

$$p = \frac{1}{|D_{\text{syn}}|} \sum_{x_{\text{syn}} \in D_{\text{syn}}} \mathbb{I}\left[\text{DCR}(x_{\text{syn}}, D_{\text{train}}) < \text{DCR}(x_{\text{syn}}, D_{\text{val}})\right]. \tag{1}$$

Under an ideal i.i.d. setting, p should be close to 0.5, since synthetic samples should not be systematically closer to training set than to held-out validation set. Values substantially above 0.5 suggest overfitting of the training set. The DCROverfit score is then defined as

$$\text{DCROverfit} = \min\left(1,\ 2(1-p)\right) \in [0, 1], \tag{2}$$

where smaller values indicate greater privacy leakage risk.

As illustrated in the first two rows of Figure 3, for a fixed dataset size N, we observe a consistent training dynamic: quality improves steadily and eventually saturates, while privacy degrades over time. Following prior work [3], this behavior can be interpreted through two characteristic timescales: τ_{gen}, when useful generative capability is established, and τ_{mem}, when memorization begins to emerge.

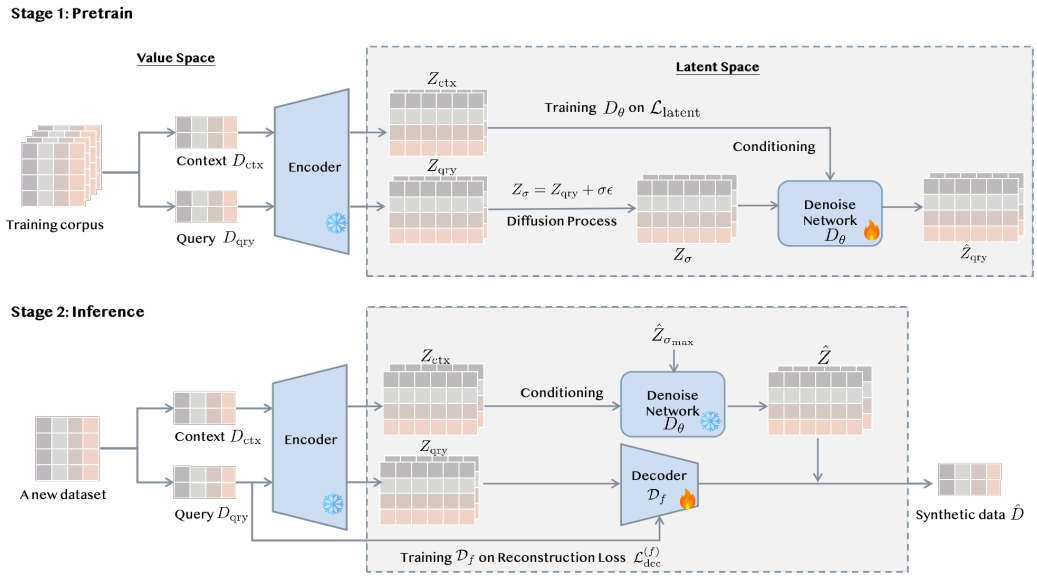


Figure 4: Illustration of DiffICL framework. **Top**: At pretraining stage, tabular data are split into context and query sets and encoded into latent representations. A conditional diffusion model learns to denoise noisy query latents conditioning on context latents, capturing the dataset distribution in latent space. **Bottom**: At inference stage, the pretrained model generates synthetic latent samples from noise conditioned on the context set. These latents are then mapped back to raw features using feature-wise decoders trained on the query data.

Ideally, these two timescales are separated, creating a window in which models achieve high quality without memorization. However, this separation does not occur in the low-data regime ($N = 200$ and $N = 400$). Instead, memorization appears early in training, before sufficient generation quality is achieved, leaving no effective room for early stopping. This limitation is further reflected in the quality–privacy frontiers (last row of Figure 3). Most trajectories exhibit a direct tradeoff: quality improves at the expense of privacy, preventing convergence to the desirable upper-right region. Ideally [3], training would first exhibit a vertical phase (quality improvement with stable privacy), followed by a horizontal phase (privacy degradation after quality saturates). In practice, however, these two effects occur simultaneously. As the dataset size N decreases, the frontiers contract inward, indicating a more severe quality–privacy tradeoff where both objectives cannot be simultaneously achieved. CTGAN and tabDDPM exhibit memorization early in training, suggesting that a dynamical regularization regime does not emerge in the small-data setting. At $N = 600$, TVAE shows a nearly vertical trajectory, where quality improves without noticeable privacy degradation. However, this behavior does not imply the presence of dynamical regularization: if such a regime existed, it would be followed by a horizontal phase corresponding to delayed memorization. Instead, it reflects architectural regularization—TVAE lacks sufficient capacity to fit the empirical distribution and thus does not enter a memorization regime even with prolonged training. This benefit comes at a cost. While architectural regularization mitigates memorization, it also constrains model capacity, leading to lower peak quality compared to tabDDPM (e.g., 0.83 vs. 0.86 at $N = 600$).

Overall, our results show that existing generative models fail to achieve a favorable quality–privacy tradeoff in small-data tabular settings, where gains in quality are accompanied by increased memorization risk. We further find that this limitation cannot be mitigated through implicit training regularization; models must rely on architectural constraints to control memorization, which in turn restricts their achievable generation quality. This suggests that improving the quality–privacy tradeoff requires new mechanisms beyond those provided by current generative models.

3 Methodology

Notations To handle the heterogeneity of diverse tabular data, we map raw features to a continuous latent space with hidden dimension d. Throughout this section, uppercase letters denote sample sets: D_{ctx} and D_{qry} represent raw context and query sets, while Z_{ctx} and Z_{qry} represent their latent

counterparts. M_{ctx} and M_{qry} denote the number of context and query samples, respectively. Synthetic sets are denoted with a hat (e.g., $\hat{D}$, $\hat{Z}$).

Method Overview Our goal is to learn a conditional generative model that produces high-fidelity synthetic tabular samples x based on a small set of context samples D_{ctx} from a new dataset. Formally, we aim to approximate the conditional distribution $p_\theta(x|D_{\text{ctx}})$ for an arbitrary D_{ctx}. To prevent the model from trivially memorizing the training data, we formulate this through an ICL pretraining paradigm: For a given dataset $D \in \mathbb{R}^{N\times F}$ with N samples and F features, we randomly partition it into two disjoint subsets: a context set $D_{\text{ctx}} \in \mathbb{R}^{M_{\text{ctx}}\times F}$ and a query set $D_{\text{qry}} \in \mathbb{R}^{M_{\text{qry}}\times F}$, where $M_{\text{ctx}} + M_{\text{qry}} = N$. By fitting the conditional distribution of D_{qry} given D_{ctx}, i.e. $\hat{p}(x|D_{\text{ctx}}) = \frac{1}{M_{\text{qry}}}\sum_{x^{(i)}\in D_{\text{qry}}} \delta(x - x^{(i)})$, we force the model to infer the underlying data distribution rather than copying the context. Building upon this ICL formulation, we propose a two-stage framework: *Stage 1: Pretraining.* We train a latent diffusion model on a large-scale corpus of over 800 real-world datasets. To promote permutation invariance and augment training diversity, for each dataset we randomly permute its rows and columns to obtain five structural variants, yielding a pretraining corpus five times the size of the original collection. The raw data D_{ctx} and D_{qry} are first projected into latent space through an encoder. The model then learns to generate the latent query set Z_{qry} conditioned on the latent context set Z_{ctx}. *Stage 2: Inference.* For a new dataset, we use the pretrained latent diffusion model to sample a synthetic latent set $\hat{Z}_0$ conditioned on the provided context. These latents are then mapped back to the raw feature space via lightweight, dataset-specific decoders. We illustrate our method framework in Figure 4.

3.1 Stage 1: Conditional Latent Diffusion Pretraining

Latent Encoding. We adopt LimiX [50] as the shared encoder of context D_{ctx} and query D_{qry} because LimiX is among the strongest currently available tabular foundation models and supports missing-value imputation tasks, suggesting that its hidden representations preserve information about the input features rather than only label-predictive signals. This property is particularly important for DiffICL, since DiffICL requires reconstructing raw feature values from the generated latent representations. The LimiX encoder is kept frozen throughout our framework. For any input $(D_{\text{ctx}}, D_{\text{qry}})$, we extract the final-layer hidden representations of LimiX to obtain the latent tensors:

$$(Z_{\text{ctx}}, Z_{\text{qry}}) = \text{Encoder}(D_{\text{ctx}}, D_{\text{qry}}) \tag{3}$$

This operation projects the raw 2D matrices into 3D latent tensors, $Z_{\text{ctx}} \in \mathbb{R}^{M_{\text{ctx}}\times F\times d}$ and $Z_{\text{qry}} \in \mathbb{R}^{M_{\text{qry}}\times F\times d}$ (with $d = 192$).

In-Context Diffusion Process. We formulate generation as a continuous-time diffusion process [43, 16] over the latent query set Z_{qry}. Given a clean latent set Z_{qry}, we construct a noisy latent set Z_σ by adding independent Gaussian noise:

$$Z_\sigma = Z_{\text{qry}} + \sigma\mathcal{E}, \qquad \mathcal{E} \in \mathbb{R}^{M_{\text{qry}}\times F\times d} \text{ with } \mathcal{E}_{i,j,k} \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0,1), \tag{4}$$

where the noise level $\sigma \in \mathbb{R}$ follows a log-normal distribution $\ln\sigma \sim \mathcal{N}(-1.2, 1.2^2)$. The denoising network D_θ is trained to reconstruct query latents Z_{qry} from noisy latents Z_σ, conditioned on the context Z_{ctx}. The training objective is:

$$\mathcal{L}_{\text{latent}}(\theta) = \mathbb{E}_{Z_{\text{qry}},\sigma,\mathcal{E}}\left[\lambda(\sigma)\big(D_\theta(Z_\sigma;\sigma,Z_{\text{ctx}}) - Z_{\text{qry}}\big)^2\right], \tag{5}$$

where $\lambda(\sigma) = \frac{\sigma^2+\sigma_{\text{data}}{}^2}{(\sigma\sigma_{\text{data}})^2}$ and $\sigma_{\text{data}} = 0.5$. We provide more pretraining details and hyperparameter configuration in Appendix A.1.

Denoising Network Architecture. Since different datasets may contain different numbers of samples and features, the latent tensors produced by the frozen LimiX encoder have variable shapes: $Z_{\text{ctx}} \in \mathbb{R}^{M_{\text{ctx}}\times F\times d}$ and $Z_{\text{qry}} \in \mathbb{R}^{M_{\text{qry}}\times F\times d}$. Therefore, the denoising network D_θ must operate on both variable sample size and variable feature dimension. Additionally, the denoising model should allow query latents to leverage the context for guidance while strictly preventing information exchange among individual query latents. To meet these requirements, we design the denoising network as a dual-axis attention Transformer. Within each layer, the model employs a decoupled, dual-axis attention mechanism: *feature-wise attention* is applied over the feature dimension within each sample. This allows the model to capture dependencies among features and supports datasets

with different numbers of features. *Sample-wise attention* is applied over the sample dimension. Context latents attend to each other, while each query latent attends only to the context latents. Query latents are not allowed to attend to other query latents, so the generated samples remain conditionally independent given the context. Further architectural details are provided in Appendix A.2.

3.2 Stage 2: Inference and Feature Reconstruction

At inference time, we are given a new dataset D, which may not have appeared during pretraining. We split it into two disjoint subsets, D_{ctx} and D_{qry}. Following the same encoding procedure used in pretraining, we feed the pair $(D_{\mathrm{ctx}}, D_{\mathrm{qry}})$ into the frozen LimiX encoder and obtain their latent representations $(Z_{\mathrm{ctx}}, Z_{\mathrm{qry}})$. Here, Z_{ctx} is used to condition the pretrained diffusion model, while $(Z_{\mathrm{qry}}, D_{\mathrm{qry}})$ is used to train feature decoders for the current dataset.

Latent Sampling. Given the context latent tensor Z_{ctx}, let K be the number of synthetic samples to generate. We initialize a noisy latent tensor $Z_{\sigma_{\max}} \in \mathbb{R}^{K\times F\times d}$, where each entry is sampled independently from $\mathcal{N}(0, \sigma_{\max}^2)$ with $\sigma_{\max} = 80$. Starting from this noise tensor, we run the reverse diffusion process for 50 steps using the EDM second-order Euler–Heun solver [16]. At each step, the denoising network D_θ removes noise from the latent tensor while conditioning on Z_{ctx}. This process iteratively produces clean synthetic latents $\hat{Z}_0 \in \mathbb{R}^{K\times F\times d}$.

Feature Reconstruction. The generated latents $\hat{Z}_0$ need to be mapped back to the raw feature space of the current dataset. Because different datasets may have different feature types and value spaces, we train a separate lightweight decoder for each feature at inference time. For feature f, we define an MLP decoder $\mathcal{D}_f : \mathbb{R}^d \to \mathcal{X}_f$, where $\mathcal{X}_f$ denotes the value space of feature f. The decoder for feature f is trained on the query set of the current dataset. Let $Z_{\mathrm{qry},f} \in \mathbb{R}^{M_{\mathrm{qry}}\times d}$ be the latent representations of feature f for all query samples, and let $D_{\mathrm{qry},f}$ be the corresponding raw feature values. We optimize $\mathcal{D}_f$ by minimizing

$$\mathcal{L}_{\mathrm{dec}}^{(f)} = \frac{1}{M_{\mathrm{qry}}} \sum_{i=1}^{M_{\mathrm{qry}}} l_f\big(\mathcal{D}_f(Z_{\mathrm{qry},f}^{(i)}), D_{\mathrm{qry},f}^{(i)}\big), \tag{6}$$

where l_f is the mean squared error for numerical features and the cross-entropy loss for categorical features. The superscript (i) indexes query samples. After training the feature decoders, we apply them to the generated latents $\hat{Z}_0$. For each feature f, we take the corresponding latent slice $\hat{Z}_{0,f} \in \mathbb{R}^{K\times d}$ and reconstruct the feature values by $\hat{D}_f = \mathcal{D}_f(\hat{Z}_{0,f})$. Finally, we concatenate the reconstructed feature columns $\{\hat{D}_f\}_{f=1}^{F}$ to obtain the synthetic dataset $\hat{D} \in \mathbb{R}^{K\times F}$. Although the decoders are trained at inference time, they are small MLPs and add only a limited computational cost. The overall inference time is comparable to single-stage generative models such as TVAE and tabDDPM, and is much lower than the two-stage method TabSYN. We report the running time of different methods in Table 6. More details about the decoder architecture and training configuration are provided in Appendix A.3. Any privacy risk introduced by the decoder training step is included in our end-to-end empirical privacy evaluation in Section 4.2.

4 Experiment

Baselines. We compare our method with representative tabular data generation approaches across different modeling paradigms, including VAE-based (TVAE [46]), GAN-based (CTGAN [46] and CTABGAN+ [53]), diffusion-based (tabDDPM [19], TabDiff [38] and TabSYN [49]), and LLM-based (GReaT [4]) methods. We provide a detailed introduction of these methods in Appendix C.2.

Datasets. We evaluate DiffICL and the baseline methods on 7 classification datasets (kc2 [26], wdbc [45], ilpd [34], diabetes [40], vehicle [39], mfeat-zernike [14], kr-vs-kp [37]) and 7 regression datasets (red wine [5], auction verification [30], abalone [28], white wine [5], wind [22], cpu activity [35], health insurance [29]). For each dataset, we randomly split the samples into equal halves for training and testing. The training set sizes range from 259 to 10,535. The datasets cover a diverse range of feature types, including fully numerical, fully categorical, and mixed-type tabular data. Detailed dataset metadata is provided in Table 5.

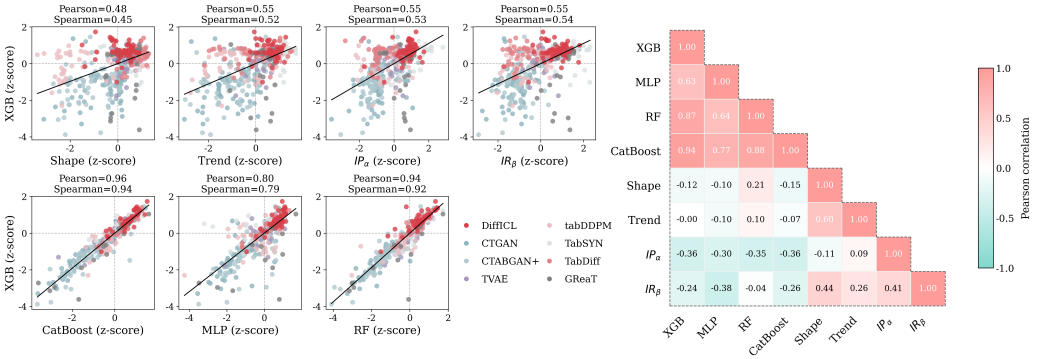


Figure 5: Alignment between different quality metrics. **Left**: pairwise relationships between downstream performance (XGBoost) and representative distribution-based metrics (Shape, Trend, IP_α, IR_β), as well as other downstream learners, after z-score normalization and aggregation across all datasets and runs. Each point corresponds to a (dataset, method, seed) instance. **Right**: Pearson correlation matrix on the *diabetes* dataset. Correlation matrices for additional datasets are provided in Figure 7.

4.1 On the Reliability of Quality Metrics

Existing quality metrics often yield inconsistent rankings of generative models, raising the question of which metrics truly reflect data quality. Since the primary goal of tabular data generation is to replace real data in downstream tasks, a reliable metric should measure how well models trained on synthetic data generalize to real data [1, 38]. We consider a set of widely used quality metrics and assess their alignment with downstream utility.

Quality Metrics. We consider two categories of quality metrics: downstream performance metrics and distributional similarity metrics. For downstream performance metrics, we assess performance using four widely adopted learners: (1) *XGBoost*, (2) *CatBoost*, (3) *MLP*, and (4) *Random Forest*. We report AUC for classification tasks and R^2 for regression tasks. For distributional similarity metrics, we include the following metrics: (1) *Shape* [33] measures column-wise similarity, with the final score averaged across all columns. (2) *Trend* [33] evaluates pairwise correlation between columns, with the final score averaged across all column pairs. (3) IP_α [1] summarizes the fidelity of synthetic data by measuring the extent to which synthetic samples fall within high-density regions of the real distribution. (4) IR_β [1] captures the diversity of synthetic data by quantifying how well the synthetic distribution covers the support of the real data. Shape, Trend, IP_α and IR_β metrics take values in $[0, 1]$. For all quality metrics, higher values correspond to better generation quality.

Details. For each real dataset d, we train each generative method with 5 random seeds and obtain multiple synthetic datasets. Each synthetic dataset is then evaluated by all quality metrics. For downstream utility, we train downstream learners (XGBoost, MLP, Random Forest, and CatBoost), on the synthetic data and evaluate it on the held-out real test data. For distribution-based metrics, including Shape, Trend, IP_α, and IR_β, we compare the synthetic data with the corresponding real training data using the standard metric definitions.

Distribution-Based Metrics Fail to Reflect Downstream Utility. We first examine metric agreement within each dataset. For a fixed dataset d, each synthetic dataset generated by one method and one random seed gives one evaluation point, and we compute pairwise Pearson correlations between metrics over these points. The right panel of Fig. 5 shows the correlation matrix on the *diabetes* dataset as a representative example, with additional datasets shown in Fig. 7. Downstream performance metrics evaluated by different learners are strongly correlated, indicating that they provide consistent assessments of synthetic data utility. In contrast, distribution-based metrics, including Shape, Trend, IP_α, and IR_β, show weak correlations with downstream performance, suggesting that distributional similarity does not necessarily translate into downstream utility.

We further aggregate results across all datasets. Because different datasets may have different metric scales and variances, we normalize each metric within each dataset:

$$z_i^{(d,m)} = \frac{x_i^{(d,m)} - \mu^{(d,m)}}{\sigma^{(d,m)}}, \tag{7}$$

Table 1: Quality and Privacy comparison across 9 and 14 datasets. The results represent the mean ± standard deviation (%) over 5 independent runs. Data quality is measured by downstream task performance (using XGBoost, CatBoost, RandomForest, and MLP as learners), and privacy is quantified by DCROverfit. GReaT is included in the 9 Datasets group where it ran successfully, but excluded from the 14 Datasets group as it failed to generate valid samples across all subsets. Best results are highlighted in red.

Method	9 Datasets					14 Datasets				
	XGB	CatB	RF	MLP	DCR	XGB	CatB	RF	MLP	DCR
CTGAN	32.4± 1.6	36.8± 1.5	34.5± 1.8	36.7± 1.6	69.0± 2.0	42.8± 1.1	46.1± 1.0	44.6± 1.0	45.6± 1.1	70.3± 1.7
CTABGAN+	23.6± 9.2	36.4± 6.1	29.6± 7.3	39.1± 4.3	68.4± 0.8	36.5± 6.0	46.4± 3.6	41.4± 4.8	48.3± 2.0	70.8± 0.5
GReaT	61.9± 3.7	65.4± 1.8	63.6± 1.9	64.2± 1.3	73.4± 1.1	-	-	-	-	-
TVAE	62.9± 1.1	64.8± 0.9	63.6± 1.1	61.5± 1.5	71.2± 0.7	68.6± 0.9	70.2± 0.9	69.5± 0.8	68.1± 1.0	72.7± 0.5
tabDDPM	62.7± 1.0	65.7± 0.7	62.8± 0.8	60.8± 2.0	66.4± 0.9	68.1± 0.7	71.0± 0.6	69.3± 0.5	68.9± 1.2	61.9± 1.9
TabSYN	66.3± 1.2	68.2± 1.3	66.8± 1.2	62.4± 5.6	66.8± 1.0	71.4± 0.8	72.8± 0.9	72.0± 0.8	67.8± 3.5	66.3± 0.7
TabDiff	56.6± 1.9	60.4± 1.6	56.5± 2.3	58.8± 2.0	72.8± 0.4	64.8± 1.2	67.7± 1.1	64.9± 1.5	66.5± 1.3	73.2± 0.5
DiffICL	70.2± 0.7	71.0± 0.5	69.6± 0.8	69.7± 1.1	74.4± 0.5	74.0± 0.7	74.6± 0.4	73.4± 0.6	73.5± 0.9	76.1± 0.4

where $x_i^{(d,m)}$ denotes a valid metric value for metric m on dataset d, and $\mu^{(d,m)}$ and $\sigma^{(d,m)}$ are computed over all methods and seeds within that dataset. We then concatenate the normalized values across datasets and compute pairwise Pearson and Spearman correlations. Consistent with the within-dataset results, distribution-based metrics show weak correlations with ML efficiency metrics, with Pearson correlations below 0.6 in the left panel of Fig. 5. Shape, which mainly captures low-order marginal distributions, has the weakest correlation with ML efficiency, with Pearson correlation 0.48. These results indicate that distribution-based metrics are poorly aligned with downstream utility and cannot faithfully reflect the practical quality of synthetic tabular data.

4.2 Bridging Quality and Privacy: Can DiffICL Reshape the Trade-off Frontier?

While all generative models often exhibit a tradeoff between data utility and privacy risk, the attainable tradeoff frontier can vary significantly across different methods. In this experiment, we investigate a central question: can DiffICL push this frontier outward by simultaneously improving both data quality and privacy protection in tabular data generation? We use downstream task performance as the quality metric, measured using four representative learners: XGBoost, MLP, CatBoost, and Random Forest. This choice is motivated by the observation that distribution-based metrics often fail to faithfully reflect the practical utility of synthetic data (Section 4.1). For privacy evaluation, we primarily use DCROverfit as a distance-based indicator of potential memorization. We additionally conduct membership inference attacks (MIAs), which assess whether an adversary can infer if a record was included in the training data (see Appendix E.1).

Details. Since data quality and privacy risk both vary during training, we select checkpoints for each baseline using the same balanced criterion. For each method, we evaluate multiple checkpoints and report the one that maximizes $\frac{1}{2} \cdot \texttt{XGBoost performance} + \frac{1}{2} \cdot \texttt{DCROverfit}$, as detailed in Appendix C.2. Due to sample-generation failures of GReaT on several datasets, we report two evaluation settings: comparison with all methods on the datasets where GReaT successfully generates valid samples, and comparison of the consistently successful methods across all 14 datasets.

DiffICL Improves Both Data Quality and Privacy Simultaneously. DiffICL achieves the best performance in terms of both data quality and privacy, as shown in Table 1. This result suggests that for a given dataset, the quality-privacy tradeoff is not a fixed limitation, but can be improved through more effective model architectures and training paradigms.

4.3 Data Augmentation for Downstream Tasks

While the experiment in Section 4.2 demonstrates the utility of using synthetic data to completely replace real data for privacy protection purposes, another application of generative models is data augmentation. In this section, we evaluate whether our generated data can improve the performance of downstream tasks when combined with real data. We construct augmented training sets by combining the original real data with 2500 synthetic samples. We then evaluate downstream models trained on these augmented sets against the models trained on real data, assessing whether the generated samples effectively improve model generalization.

Table 2: Data augmentation performance comparison across 9 and 14 datasets. The results represent the mean ± standard deviation (%) over 5 independent runs. We compare the downstream task performance of models trained solely on real data versus those trained on an augmented dataset (real + synthetic) generated by various methods. Best results in each column are highlighted in red, and the second-best in blue.

Method	9 Datasets				14 Datasets			
	XGB	CatB	RF	MLP	XGB	CatB	RF	MLP
Real Data	74.2± 0.0	74.2± 0.1	73.5± 0.1	72.2± 0.4	76.9± 0.0	76.9± 0.1	76.8± 0.1	75.6± 0.3
CTGAN	70.0± 0.8	69.0± 0.8	66.9± 0.9	66.5± 0.7	73.5± 0.6	72.8± 0.8	71.9± 0.7	70.7± 0.9
CTABGAN+	73.0± 0.3	71.8± 0.4	72.4± 0.2	69.6± 0.3	75.9± 0.3	75.2± 0.4	75.8± 0.2	73.5± 0.3
GReaT	65.8± 1.1	66.2± 0.8	64.3± 1.4	65.0± 1.2	-	-	-	-
TVAE	72.5± 0.5	72.0± 0.5	70.9± 0.5	71.0± 0.3	75.8± 0.4	75.7± 0.4	75.2± 0.4	74.7± 0.2
tabDDPM	72.6± 0.4	72.0± 0.4	71.1± 0.4	69.2± 0.3	75.7± 0.2	75.3± 0.2	75.3± 0.2	73.8± 0.2
TabSYN	72.8± 0.2	72.4± 0.2	71.7± 0.4	70.7± 0.6	76.0± 0.3	75.7± 0.2	75.7± 0.3	73.4± 0.6
TabDiff	71.6± 0.3	70.4± 0.3	69.2± 0.3	66.7± 0.2	75.3± 0.1	74.7± 0.1	74.0± 0.2	72.1± 0.2
DiffICL	74.6± 0.2	75.0± 0.2	73.8± 0.2	73.0± 0.8	77.0± 0.2	77.1± 0.2	76.9± 0.1	75.8± 0.5

Table 3: Ablation study on the source of performance gains. We compare the full DiffICL model against two variants: **DiffICL(S)**, which restricts pre-training to the single target dataset (removing cross-dataset priors), and **DiffICL(N)**, which omits permutation-based corpus expansion during pre-training. Downstream performance is evaluated under two application scenarios: using purely synthetic data as a replacement for real data (**Syn.**), and using a mixture of real and synthetic data for data augmentation (**Aug.**). The results report the mean ± standard deviation (%). The best results in each column are highlighted in red.

Method	XGB		CatB		RF		MLP	
	Syn.	Aug.	Syn.	Aug.	Syn.	Aug.	Syn.	Aug.
DiffICL	74.0 ± 0.7	77.0 ± 0.2	74.6 ± 0.4	77.1 ± 0.2	73.4 ± 0.6	76.9 ± 0.1	73.5 ± 0.9	75.8 ± 0.5
DiffICL(S)	57.0 ± 1.2	76.3 ± 0.3	59.1 ± 1.1	75.6 ± 0.4	58.3 ± 1.2	76.0 ± 0.3	56.2 ± 2.6	73.6 ± 0.5
DiffICL(N)	72.8 ± 0.3	75.2 ± 0.1	73.6 ± 0.2	74.5 ± 0.1	72.4 ± 0.3	75.1 ± 0.2	72.6 ± 0.7	74.2 ± 0.4

DiffICL Enables Effective Data Augmentation. As demonstrated in Table 2, integrating synthetic samples generated by DiffICL with the real training set yields performance improvements for all downstream learners. Among all our evaluated deep generative models, DiffICL is the only evaluated method that consistently improves augmentation performance in our experiments. Conversely, augmenting the training set with samples from other approaches degrades the predictive performance compared to the real-data baseline. We attribute the augmentation capability of DiffICL to the ICL pretraining paradigm. Standard tabular generative models learn the target distribution from scratch. Consequently, their learning is constrained by the empirical distribution of the limited training samples. In contrast, DiffICL is pretrained across a vast, diverse corpus of tabular datasets, allowing it to acquire strong structural priors. By leveraging its extensive prior knowledge to complement the limited empirical data, DiffICL effectively mitigates the information bottleneck in finite training sets.

4.4 Disentangling the Sources of DiffICL's Performance Gain

The preceding experiments show that DiffICL improves the quality–privacy frontier (Section 4.2) and enables effective data augmentation (Section 4.3). We next conduct ablation studies to identify the sources of these gains. This section focuses on the construction of the pretraining corpus, studying the contribution of multi-dataset pretraining and permutation-based corpus expansion. We defer inference-stage analyses to Appendix E, including the effect of the context ratio and the number of training samples.

The Role of Multi-Dataset Priors. We first evaluate DiffICL(S), a variant that uses the same encoder and latent diffusion architecture as DiffICL, but is pretrained only on the target training dataset. This variant isolates the contribution of multi-dataset ICL pretraining from that of the encoded tabular representations. As shown in Table 3, DiffICL substantially outperforms DiffICL(S), indicating that the performance gains cannot be explained by the LimiX representations alone. Instead, they rely on structural priors acquired through multi-dataset pretraining. Without these priors, DiffICL(S) remains constrained by the empirical distribution of a single dataset and generalizes less effectively beyond the observed samples. **The Impact of Corpus Expansion.** We then evaluate DiffICL(N), which is

pretrained on the same datasets as DiffICL but without row and column permutations. Table 3 shows that removing permutation-based corpus expansion consistently degrades performance, indicating that this expansion strategy is an important component of DiffICL's pretraining. **The Effect of Pretraining Corpus Size.** We vary the number of pretraining datasets from 50 to 826 while keeping the diffusion architecture and training budget fixed (Table 8). Average generation quality increases from 58.4 to 74.0, while average DCROverfit varies non-monotonically between 74.8 and 76.8. Thus, increasing corpus size improves generation quality without a systematic decline in the measured privacy score.

5 Conclusion

We study the quality–privacy tradeoff in small-data tabular generation and propose DiffICL, an ICL pretraining framework that uses multi-dataset structural priors to generate synthetic data from limited context. Through experiments, we then show that commonly used distribution-based quality metrics are weakly aligned with downstream utility, motivating the use of predictive performance as the main quality measure. DiffICL achieves better utility and privacy than representative baselines. The data augmentation experiment further shows that DiffICL improves downstream performance when synthetic data are combined with real data. Finally, ablation studies confirm that these gains mainly come from multi-dataset pretraining.

A promising direction for future work is attribute-aware privacy control. DiffICL's feature-wise latent generation and independent feature decoders offer a basis for selectively applying privacy constraints to sensitive attributes. For example, perturbing the latent representations of sensitive features before decoding could help mitigate attribute-specific memorization. Investigating how such interventions affect privacy, cross-feature dependencies, and downstream utility would extend DiffICL toward more flexible privacy control.

Acknowledgments and Disclosure of Funding

This work was supported by the project "Research on the Construction of a Cognitive Reasoning Engine for Water Systems in the Beijing–Tianjin–Hebei Region" (No. 2026ZD1212206) and the Tsinghua University (Department of Computer Science and Technology)-Siemens Ltd., China Joint Research Center for Industrial Intelligence and Internet of Things (JCIIOT). All opinions expressed in this paper are those of the authors and do not necessarily reflect the views of the funding agencies. The authors declare no additional competing interests beyond the funding disclosed above.

References

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

[2] Patricia A Apellániz, Juan Parras, and Santiago Zazo. An improved tabular data generator with vae-gmm integration. In *2024 32nd European Signal Processing Conference (EUSIPCO)*, pages 1886–1890. IEEE, 2024.

[3] Tony Bonnaire, Raphaël Urfin, Giulio Biroli, and Marc Mézard. Why diffusion models don't memorize: The role of implicit dynamical regularization in training. *arXiv preprint arXiv:2505.17638*, 2025.

[4] Vadim Borisov, Kathrin Seßler, Tobias Leemann, Martin Pawelczyk, and Gjergji Kasneci. Language models are realistic tabular data generators. *arXiv preprint arXiv:2210.06280*, 2022.

[5] Paulo Cortez, António Cerdeira, Fernando Almeida, Telmo Matos, and José Reis. Modeling wine preferences by data mining from physicochemical properties. *Decision support systems*, 47(4):547–553, 2009.

[6] Yves-Alexandre De Montjoye, César A Hidalgo, Michel Verleysen, and Vincent D Blondel. Unique in the crowd: The privacy bounds of human mobility. *Scientific reports*, 3(1):1376, 2013.

[7] Yves-Alexandre De Montjoye, Laura Radaelli, Vivek Kumar Singh, and Alex "Sandy" Pentland. Unique in the shopping mall: On the reidentifiability of credit card metadata. *Science*, 347(6221):536–539, 2015.

[8] Khaled El Emam and Luk Arbuckle. *Anonymizing health data: case studies and methods to get you started*. " O'Reilly Media, Inc.", 2013.

[9] Liancheng Fang, Aiwei Liu, Hengrui Zhang, Henry Peng Zou, Weizhi Zhang, and Philip S Yu. Tabgen-icl: Residual-aware in-context example selection for tabular data generation. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 20027–20041, 2025.

[10] Anurag Garg, Muhammad Ali, Noah Hollmann, Lennart Purucker, Samuel Müller, and Frank Hutter. Real-tabpfn: Improving tabular foundation models via continued pre-training with real-world data. *arXiv preprint arXiv:2507.03971*, 2025.

[11] Mikel Hernandez, Pablo A Osorio-Marulanda, Mikel Catalina, Lorea Loinaz, Gorka Epelde, and Naiara Aginako. Comprehensive evaluation framework for synthetic tabular data in health: fidelity, utility and privacy analysis of generative models with and without privacy guarantees. *Frontiers in Digital Health*, 7:1576290, 2025.

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

[13] 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. *Nature*, 637(8045):319–326, 2025.

[14] Anil K Jain, Robert P. W. Duin, and Jianchang Mao. Statistical pattern recognition: A review. *IEEE Transactions on pattern analysis and machine intelligence*, 22(1):4–37, 2000.

[15] Jingang Qu and David Holzmüller and Gaël Varoquaux and Marine Le Morvan. Tabicl: A tabular foundation model for in-context learning on large data. *arXiv preprint arXiv:2502.05564*, 2025.

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

[17] Jayoung Kim, Chaejeong Lee, and Noseong Park. Stasy: Score-based tabular data synthesis. *arXiv preprint arXiv:2210.04018*, 2022.

[18] Ron Kohavi et al. Scaling up the accuracy of naive-bayes classifiers: A decision-tree hybrid. In *Kdd*, volume 96, pages 202–207, 1996.

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

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

[21] Xiaofeng Lin, Chenheng Xu, Matthew Yang, and Guang Cheng. Ctsyn: A foundation model for cross tabular data generation. *arXiv preprint arXiv:2406.04619*, 2024.

[22] Si-Yang Liu, Hao-Run Cai, Qi-Le Zhou, Huai-Hong Yin, Tao Zhou, Jun-Peng Jiang, and Han-Jia Ye. Talent: A tabular analytics and learning toolbox. *Journal of Machine Learning Research*, 26(226):1–16, 2025.

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

[24] Junwei Ma, Apoorv Dankar, George Stein, Guangwei Yu, and Anthony Caterini. Tabpfgen–tabular data generation with tabpfn. *arXiv preprint arXiv:2406.05216*, 2024.

[25] Junwei Ma, Valentin Thomas, Rasa Hosseinzadeh, Alex Labach, Hamidreza Kamkari, Jesse C Cresswell, Keyvan Golestan, Guangwei Yu, Anthony L Caterini, and Maksims Volkovs. Tabdpt: Scaling tabular foundation models on real data. *arXiv preprint arXiv:2410.18164*, 2024.

[26] T. Menzies and J.S. Di Stefano. How good is your blind spot sampling policy. In *High Assurance Systems Engineering, 2004. Proceedings. Eighth IEEE International Symposium on*, pages 129–138, March 2004.

[27] Arvind Narayanan and Vitaly Shmatikov. Robust de-anonymization of large sparse datasets. In *2008 IEEE Symposium on Security and Privacy (sp 2008)*, pages 111–125. IEEE, 2008.

[28] Warwick J Nash, Tracy L Sellers, Simon R Talbot, Andrew J Cawthorn, and Wes B Ford. The population biology of abalone (haliotis species) in tasmania. i. blacklip abalone (h. rubra) from the north coast and islands of bass strait. *Sea Fisheries Division, Technical Report*, 48:p411, 1994.

[29] Craig A Olson. A comparison of parametric and semiparametric estimates of the effect of spousal health insurance coverage on weekly hours worked by wives. *Journal of Applied Econometrics*, 13(5):543–565, 1998.

[30] Elaheh Ordoni, Jakob Bach, and Ann-Katrin Fleck. Analyzing and predicting verification of data-aware process models–a case study with spectrum auctions. *IEEE Access*, 10:31699–31713, 2022.

[31] Adedoyin Tolulope Oyewole, Bisola Beatrice Oguejiofor, Nkechi Emmanuella Eneh, Chidiogo Uzoamaka Akpuokwe, and Seun Solomon Bakare. Data privacy laws and their impact on financial technology companies: a review. *Computer science & IT research journal*, 5(3):628–650, 2024.

[32] Noseong Park, Mahmoud Mohammadi, Kshitij Gorde, Sushil Jajodia, Hongkyu Park, and Youngmin Kim. Data synthesis based on generative adversarial networks. *arXiv preprint arXiv:1806.03384*, 2018.

[33] Neha Patki, Roy Wedge, and Kalyan Veeramachaneni. The synthetic data vault. In *2016 IEEE international conference on data science and advanced analytics (DSAA)*, pages 399–410. IEEE, 2016.

[34] Bendi Venkata Ramana, M Surendra Prasad Babu, and NB Venkateswarlu. A critical comparative study of liver patients from usa and india: an exploratory analysis. *International Journal of Computer Science Issues (IJCSI)*, 9(3):506, 2012.

[35] CE Rasmussen, RM Neal, G Hinton, D Van Camp, M Revow, Z Ghahramani, R Kustra, and R Tibshirani. Delve data for evaluating learning in valid experiments, 1995–1996. *URL http://www. cs. toronto. edu/ delve*, 2003.

[36] Fatima Jahan Sarmin, Atiquer Rahman Sarkar, Yang Wang, and Noman Mohammed. Synthetic data: revisiting the privacy-utility trade-off: F. jahan sarmin et al. *International Journal of Information Security*, 24(4):156, 2025.

[37] Alen D Shapiro. *Structured induction in expert systems*. Addison-Wesley Longman Publishing Co., Inc., 1987.

[38] Juntong Shi, Minkai Xu, Harper Hua, Hengrui Zhang, Stefano Ermon, and Jure Leskovec. Tabdiff: a mixed-type diffusion model for tabular data generation. *arXiv preprint arXiv:2410.20626*, 2024.

[39] Jan Paul Siebert. Vehicle recognition using rule based methods. 1987.

[40] Jack W Smith, James E Everhart, William C Dickson, William C Knowler, and Robert Scott Johannes. Using the adap learning algorithm to forecast the onset of diabetes mellitus. In *Proceedings of the annual symposium on computer application in medical care*, page 261, 1988.

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

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

[43] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

[44] Theresa Stadler, Bristena Oprisanu, and Carmela Troncoso. Synthetic data–anonymisation groundhog day. In *31st USENIX Security Symposium (USENIX Security 22)*, pages 1451–1468, 2022.

[45] W Nick Street, William H Wolberg, and Olvi L Mangasarian. Nuclear feature extraction for breast tumor diagnosis. In *Biomedical image processing and biomedical visualization*, volume 1905, pages 861–870. SPIE, 1993.

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

[47] Lei Xu and Kalyan Veeramachaneni. Synthesizing tabular data using generative adversarial networks. *arXiv preprint arXiv:1811.11264*, 2018.

[48] Jinsung Yoon, Lydia N Drumright, and Mihaela Van Der Schaar. Anonymization through data synthesis using generative adversarial networks (ads-gan). *IEEE journal of biomedical and health informatics*, 24(8):2378–2388, 2020.

[49] Hengrui Zhang, Jiani Zhang, Balasubramaniam Srinivasan, Zhengyuan Shen, Xiao Qin, Christos Faloutsos, Huzefa Rangwala, and George Karypis. Mixed-type tabular data synthesis with score-based diffusion in latent space. *arXiv preprint arXiv:2310.09656*, 2023.

[50] Xingxuan Zhang, Gang Ren, Han Yu, Hao Yuan, Hui Wang, Jiansheng Li, Jiayun Wu, Lang Mo, Li Mao, Mingchao Hao, et al. Limix: Unleashing structured-data modeling capability for generalist intelligence. *arXiv preprint arXiv:2509.03505*, 2025.

[51] Xiyuan Zhang, Danielle C Maddix, Junming Yin, Nick Erickson, Abdul Fatir Ansari, Boran Han, Shuai Zhang, Leman Akoglu, Christos Faloutsos, Michael W Mahoney, et al. Mitra: Mixed synthetic priors for enhancing tabular foundation models. *arXiv preprint arXiv:2510.21204*, 2025.

[52] Zilong Zhao, Aditya Kunar, Robert Birke, and Lydia Y Chen. Ctab-gan: Effective table data synthesizing. In *Asian conference on machine learning*, pages 97–112. PMLR, 2021.

[53] Zilong Zhao, Aditya Kunar, Robert Birke, Hiek Van der Scheer, and Lydia Y Chen. Ctab-gan+: Enhancing tabular data synthesis. *Frontiers in big Data*, 6:1296508, 2024.

A Implementation Details

A.1 Pretraining Details

We construct a pretraining corpus from real-world tabular datasets collected from Kaggle and the UCI Machine Learning Repository. We exclude datasets with fewer than 50 samples or more than 50 features, and further remove regression datasets with negative R^2 scores, which typically indicate poor data quality. This results in 826 datasets. To increase diversity and encourage permutation invariance over rows and features, we perform data augmentation by randomly permuting the order of samples and features (while keeping the target column unchanged). For each dataset, we generate five augmented variants, resulting in a total of 4,130 datasets. For each dataset, we randomly split samples into a context set D_{ctx} and a query set D_{qry}, where the context ratio r is uniformly sampled from $[0.2, 0.5]$. Both subsets are encoded into latent representations, yielding training pairs $(Z_{\text{ctx}}, Z_{\text{qry}})$.

We pretrain the diffusion model for 20,000 epochs using a batch size of 128. When $|Z_{\text{qry}}| > 128$, we randomly subsample 128 query instances. Training is conducted on 32 NVIDIA RTX 4090 GPUs with a cosine learning rate schedule and linear warmup. The initial learning rate is set to 2×10^{-4} with a warmup ratio of 0.05, and no weight decay is applied. The choice of hyperparameters is guided by a systematic validation protocol designed to assess *cross-dataset generalization*. Specifically, from the collected dataset pool, we reserve a subset as a held-out validation set, which remains entirely unseen during pretraining. For each hyperparameter configuration, we evaluate model checkpoints by measuring the Fréchet Inception Distance (FID) between generated query latents $\hat{Z}_{\text{qry}}$ and real query latents Z_{qry} on these validation datasets. Since the validation datasets are not observed during pretraining, this FID score serves as a proxy for dataset-level generalization: it quantifies whether a conditional diffusion model trained on a collection of datasets can generate high-quality latent representations for previously unseen datasets under the in-context conditioning framework. We select hyperparameter configurations that achieve the best performance on this validation criterion. We take *epoch* as a representative example: we observe that increasing the number of pretraining epochs initially improves generalization performance, but beyond a certain point leads to degradation. This behavior suggests that excessive training causes the model to overfit to the datasets seen during pretraining, thereby reducing its ability to generalize to new datasets. Similar selection principles are applied to other hyperparameters.

A.2 Denoising Network Architecture

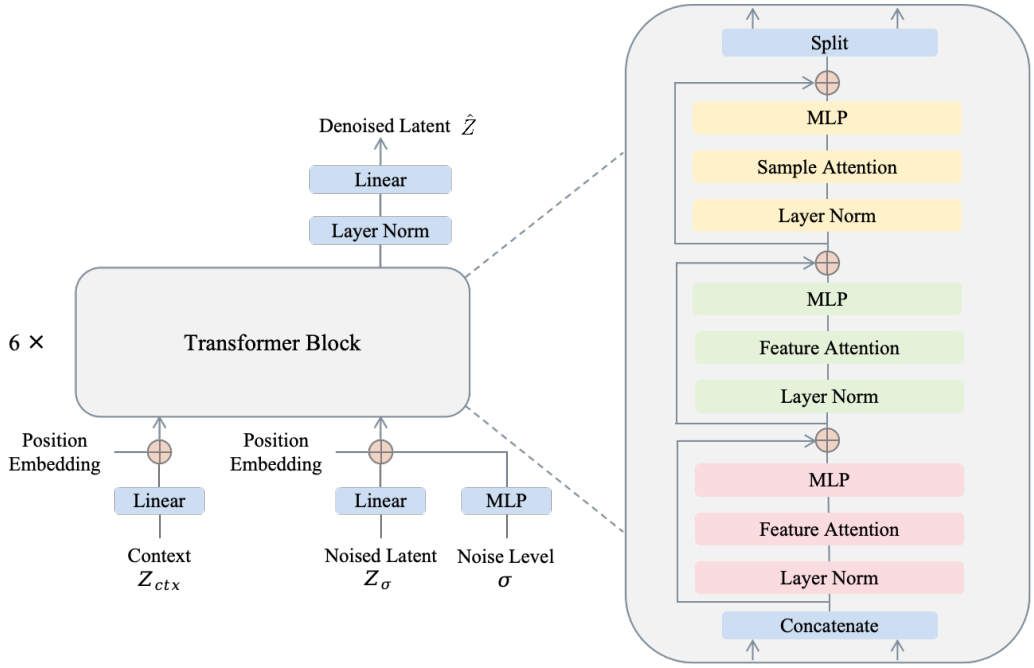


Figure 6: Denoise network architecture.

Given a noised latent input $Z_\sigma \in \mathbb{R}^{M_{\text{qry}} \times F \times d}$, we first project it into the model dimension (512) via a linear layer. Context latents $Z_{\text{ctx}} \in \mathbb{R}^{M_{\text{ctx}} \times F \times d}$ are processed in parallel using a separate projection. To encode structural information, we add positional embeddings to both inputs: for context representations, we use 2D sinusoidal positional encodings over the sample and feature dimensions (M_{ctx}, F), while for noised query representations, we apply 1D sinusoidal positional encodings along the feature dimension. In addition, diffusion timestep (noise level σ) embeddings are computed using a sinusoidal mapping followed by an MLP, and added to the query representations. The core of our denoising network consists of a stack of six transformer layers that jointly process context and query representations by concatenating them along the sample dimension, resulting in a unified representation $h \in \mathbb{R}^{(M_{\text{ctx}}+M_{\text{qry}}) \times F \times 512}$ and enabling in-context learning-style conditioning. Each transformer layer comprises three attention sublayers: two feature-wise attention sublayers and one sample-wise attention sublayer. The feature-wise attention operates across feature dimensions within

each sample, while the sample-wise attention captures interactions across different samples, allowing query tokens to attend to context tokens. All sublayers share the same architecture: Layer Normalization is first applied, followed by an 8-head attention module (implemented with FlashAttention for efficiency), a 2-layer MLP as feed-forward network, and a residual connection. Formally, for an input representation h, each sublayer computes

$$h \leftarrow h + \mathrm{FFN}(\mathrm{Attn}(\mathrm{LayerNorm}(h))).$$

After passing through all layers, the joint representation $h \in \mathbb{R}^{(M_{\text{ctx}}+M_{\text{qry}})\times F\times 512}$ is split back into context and query parts. We retain only the query representations corresponding to M_{qry} samples, apply a final Layer Normalization, and project them back to the latent dimension d to obtain the denoised output $\hat{Z} \in \mathbb{R}^{M_{\text{qry}}\times F\times d}$. To stabilize training, we adopt an EDM-style preconditioning scheme in which the network predicts a scaled residual and the final output is formed via a noise-dependent linear combination of the input and network prediction , and the model is trained with the corresponding EDM loss across a range of noise levels.

A.3 Decoder Architecture and Training

To reconstruct raw tabular data from latent representations, we train a lightweight embedding decoder for each evaluation dataset. The decoder takes per-feature latent embeddings as input and predicts the corresponding raw feature values and target variable. It assigns an independent MLP head to each predicted variable (features and target). Each MLP head contains 2 hidden layers with hidden dimension 768, where each hidden layer is followed by ReLU activation and dropout with rate 0.1. A final linear layer maps the hidden representation to the required output dimension: feature heads output the corresponding reconstructed feature values, while the target head outputs a scalar for regression tasks or C logits for classification tasks, where C is the number of classes.

The decoder is trained on the query set using latent embeddings and corresponding raw values. We use mean squared error for numerical features and regression targets, and cross-entropy loss for categorical features and classification targets. We optimize the decoder using Adam with a learning rate of 2×10^{-5}.

B Related Works

Diffusion Model Building on the early diffusion probabilistic framework of Sohl-Dickstein et al. [41], DDPM [12] reformulates reverse diffusion as noise prediction and trains the model with a simplified denoising objective, which substantially improves the practicality and sample quality of diffusion models. DDIM [42] accelerates DDPM sampling by replacing the original Markovian diffusion process with a non-Markovian process that keeps the same training objective but allows deterministic, shorter-step sampling with much less loss in sample quality. Song et al. [43] formulate score-based generative modeling in a continuous-time SDE framework, enabling flexible sampling through reverse-time SDEs, probability-flow ODEs, and predictor-corrector samplers. EDM [16] improves diffusion-based generative models by preconditioning the denoising network, which rescales its inputs, outputs, skip connection, and loss across noise levels to make training more stable. It also introduces a more efficient sampler that combines improved noise schedules, second-order Heun integration, and controlled stochasticity.

Tabular generative model TableGAN [32] uses a GAN-based architecture with an additional classifier, information loss, and classification loss to synthesize relational tables whose records are statistically similar to the original data and maintain semantic consistency among attributes. CTGAN [46] models continuous columns by decomposing each value into a mode indicator and a normalized value within that mode, while modeling discrete columns with one-hot representations and a conditional generator trained with sampling strategies that give rare categories more training opportunities. TVAE adapts the variational autoencoder framework to mixed-type tabular data by modeling continuous values with Gaussian distributions and discrete values with categorical distributions. CTAB-GAN [52] extends CTGAN to more complex industrial tabular data by using a mixed-type encoder for variables that combine categorical and continuous values, and applying logarithm transformation to better handle long-tail continuous distributions. CTAB-GAN+[53] improves CTAB-GAN by combining more adaptive feature encoding and more stable Wasserstein training with downstream supervision, and further extends the framework to differentially private

Table 4: Component-wise comparison of tabular generation methods.

Paradigm	Method	Generative architecture	Use existing pretrained model	Pretrain a generative model	Dataset-specific training	In-context conditioning
Dataset-specific generators	TVAE	VAE	✗	✗	✓	✗
	CTGAN	GAN	✗	✗	✓	✗
	CTABGAN+	GAN	✗	✗	✓	✗
	TabDDPM	Diffusion	✗	✗	✓	✗
	TabDiff	Diffusion	✗	✗	✓	✗
	TabSYN	Latent diffusion	✗	✗	✓	✗
Generation using existing pretrained models	GReaT	LLM generation	✓ (LLM)	✗	✓ (LLM fine-tuning)	✓
	TabPFGen	Energy-based generation	✓ (TabPFN)	✗	✗	✓
	TabGen-ICL	LLM generation	✓ (LLM)	✗	✗	✓
Generative-model pretraining	DiffICL	ICL-conditioned latent diffusion	✓ (LimiX encoder only)	✓	✓ (decoder)	✓

tabular data synthesis. GOGGLE [23] is a VAE-based tabular generative model that replaces the standard MLP decoder with a message-passing neural network operating on a learned relational structure between variables. As GAN-based tabular generators remain difficult to train and can suffer from mode collapse, recent studies have increasingly explored diffusion models for tabular synthesis. STaSy [17] adopts score-based generative modeling for tabular synthesis and introduces self-paced learning plus log-probability-based fine-tuning to improve the stability of denoising score matching. TabDDPM [19] proposes a DDPM framework that uses Gaussian diffusion for numerical features, multinomial diffusion for categorical features, and an MLP-based denoising network to generate mixed-type synthetic tables. CoDi [20] uses two coupled diffusion models, one for continuous variables and one for discrete variables, and trains them to condition on each other with an additional contrastive loss. TabSyn [49] first maps numerical and categorical columns into a unified VAE latent space with Transformer-based encoders and decoders, and then training a score-based diffusion model in this continuous latent space. TabDiff [38] directly models numerical columns with Gaussian diffusion and categorical columns with masked discrete diffusion, and learns a separate noise schedule for each column to handle the heterogeneous distributions of mixed-type tabular data. Beyond dataset-specific generators, recent work leverages pretrained models and in-context examples for tabular synthesis. TabPFGen [24] converts the pretrained TabPFN predictor into an energy-based generative model, enabling synthesis without additional model training. TabGen-ICL [9] prompts a frozen language model with in-context examples selected through a residual-aware iterative procedure. Unlike TabPFGen, which repurposes a pretrained predictor, and TabGen-ICL, which prompts a frozen language model, DiffICL explicitly pretrains a context-conditioned latent diffusion model for tabular generation. DiffICL's objective requires generating held-out query samples conditioned on a disjoint context set, encouraging the model to learn distributional structure that generalizes beyond the observed records.

We provide a component-wise comparison to distinguish dataset-specific generators, pretrained-model-based generators, and generative ICL models in Table 4. The comparison highlights three different paradigms. Traditional tabular generative models (e.g., TVAE, CTGAN, CTABGAN+, TabDDPM, TabDiff, and TabSYN) learn a generator from each target dataset independently. GReaT, TabPFGen, and TabGen-ICL instead leverage existing pretrained models (LLMs or TabPFN) and use their pretrained capabilities for generation, without pretraining a dedicated tabular generative model. In contrast, DiffICL pretrains a tabular generative model with an in-context learning objective. The pretrained LimiX model in DiffICL is only used as a frozen encoder, while the generative capability is learned through DiffICL's cross-dataset generative pretraining.

Tabular Foundation Model TabPFN [13] is the first tabular foundation model pretrained on a large collection of synthetic tabular tasks, enabling it to perform classification and regression by conditioning on a labeled context set without task-specific training or hyperparameter tuning. TabICL [15] scales tabular in-context learning to larger classification datasets by compressing each row into a fixed-dimensional embedding and then applying a Transformer over these row embeddings for efficient prediction from a labeled context set. TabDPT [25] trains tabular foundation model on real datasets with self-supervised learning and retrieval. Real-TabPFN [10] improves TabPFN by continuing pretraining on a small, curated set of large real-world tabular datasets, reducing the

Table 5: Summary of the datasets used in the experiments.

Task Type	Dataset Name	Training Size	Categorical	Numerical
Classification	kc2	259	0	21
	wdbc	282	0	30
	ilpd	289	1	9
	diabetes	382	0	8
	vehicle	419	0	18
	mfeat-zernike	990	0	47
	kr-vs-kp	1596	36	0
Regression	red wine	799	0	11
	auction verification	1021	1	7
	abalone	2088	1	7
	white wine	2449	0	11
	wind	3287	0	14
	cpu activity	4096	0	21
	health insurance	10535	7	5

synthetic-to-real gap. Mitra [51] argues that the key to improving synthetic-data-pretrained tabular foundation models lies in the design of synthetic priors, and trains a model on a carefully selected mixture of priors. Unlike tabular foundation models designed mainly for prediction, LimiX [50] pretrains one Transformer to handle partially observed tables, allowing the same model to perform prediction, regression and missing value imputation through a unified framework.

C Experimental Details

C.1 Dataset metadata

We report the metadata of the datasets used in the experiments in Table 5. The number of features excludes the target variable. *We manually verify that there is no overlap between the evaluation datasets and the pretraining corpus.* The `wind` dataset is obtained from Hugging Face[3]. All other datasets are obtained from OpenML[4].

C.2 Baseline Methods

We implement all baseline methods based on their respective open-source repositories, as provided in the footnotes.

- **TVAE.** TVAE models tabular data using a variational autoencoder, where each feature is encoded into a latent space and reconstructed via a decoder. It models numerical features using Gaussian distributions and categorical features using discrete distributions.[5]
- **CTGAN.** CTGAN employs a conditional generative adversarial network designed for tabular data, using conditional vectors to handle imbalanced discrete columns and mode-specific normalization to better model continuous variables.[6]
- **CTABGAN+.** CTABGAN+ extends CTGAN by introducing improved data preprocessing and training strategies, including better handling of mixed-type columns (continuous, categorical, and mixed distributions).[7]
- **tabDDPM.** To the best of our knowledge, tabDDPM is the first work to apply diffusion models to tabular data generation. It adopts a discrete-time diffusion formulation, modeling the data distribution via a forward noising process and a learned reverse denoising process.[8]

[3] `https://huggingface.co/datasets/LAMDA-Tabular/TALENT`
[4] `https://www.openml.org`
[5] `https://github.com/sdv-dev/CTGAN`
[6] `https://github.com/sdv-dev/CTGAN`
[7] `https://github.com/Team-TUD/CTAB-GAN-Plus`
[8] `https://github.com/yandex-research/tab-ddpm`

- **TabDiff.** TabDiff proposes a unified mixed-type diffusion framework. It adopts a continuous-time diffusion formulation and introduces a stochastic sampler to reduce accumulated decoding errors during sampling.[9]
- **TabSYN.** TabSYN proposes a latent diffusion framework for tabular data generation. It first trains a variational autoencoder to project mixed-type tabular data into a continuous latent space, and then learns a diffusion model over the latent representations.[10]
- **GReaT.** GReaT formulates tabular data generation as a conditional language modeling problem, where each row is serialized into a sequence and generated autoregressively using a pre-trained large language model.[11]

Baseline Checkpoint Selection. During training, generative models often improve data quality while also increasing privacy risk, making checkpoint selection important for fair comparison. For each baseline, we therefore evaluate multiple checkpoints and select the checkpoint that maximizes a balanced objective:

$$\frac{1}{2} \cdot \text{XGBoost performance} + \frac{1}{2} \cdot \text{DCROverfit}.$$

Here, XGBoost performance measures downstream utility, while DCROverfit measures privacy protection. For TVAE, CTGAN, CTABGAN+, TabSYN, and TabDiff, we evaluate checkpoints at training steps $\{500, 1000, 2000, 3000, 4000, 5000\}$. For tabDDPM, which generally requires longer training to converge, we use $\{1000, 2000, 3000, 4000, 5000, 8000, 10000\}$. For GReaT, we consider checkpoints at $\{1000, 2000, 5000, 10000\}$. For all baseline methods other than GReaT, the learning rate is set to 2×10^{-4} with a training batch size of 500. For GReaT, we retain its default learning rate and reduce the batch size to 8 due to memory constraints.

Handling GReaT Generation Failures. We observe that GReaT fails to generate valid samples on 9 datasets, including red wine, mfeat-zernike, vehicle, wdbc, kc2, kr-vs-kp, white wine, wind, and cpu activity, even after 10,000 training steps. Since training GReaT for 10,000 steps already requires substantial computational cost, approximately 10 hours in our setting, we do not further increase its training budget. Instead, following the official recommendation, we enable guided sampling, which allows GReaT to successfully generate valid samples on 4 of these 9 datasets: kr-vs-kp, white wine, wind, and cpu activity. Based on these observations, we report results under two settings. First, we compare all methods on the 9 datasets where GReaT successfully produces valid samples. Second, we compare the 7 consistently successful methods across all 14 datasets.

Table 6: Running Time Comparison of Different Methods(Mean $\pm$ Std)

Method	diabetes	cpu activity	kr-vs-kp	health insurance
CTABGAN+	52.12 ± 3.04	21.48 ± 0.63	21.88 ± 0.81	215.16 ± 19.16
CTGAN	40.37 ± 0.25	57.10 ± 1.75	55.65 ± 0.70	266.32 ± 10.60
tabDDPM	13.50 ± 0.21	12.05 ± 0.18	101.47 ± 4.11	43.27 ± 0.94
TabDiff	686.19 ± 26.97	113.96 ± 17.37	385.78 ± 20.38	101.30 ± 12.35
TabSYN	628.47 ± 11.53	104.40 ± 11.49	272.33 ± 28.64	82.56 ± 12.45
TVAE	21.12 ± 0.31	46.61 ± 1.29	21.55 ± 0.99	32.63 ± 1.14
GReaT	169.86 ± 0.57	16537.65 ± 677.19	26759.59 ± 917.63	119.11 ± 4.35
DiffICL	18.17 ± 2.08	41.37 ± 9.99	40.80 ± 2.91	69.16 ± 19.29

C.3 Running Time

Table 6 presents the running times of the evaluated generation methods across the Diabetes, Health Insurance, kr-vs-kp, and CPU Activity datasets. For the baseline models, the execution time accounts for training model and generating 2,500 synthetic samples. The training phase consists of 5,000 steps for tabDDPM and 2,000 steps for the remaining baselines. For DiffICL, the running time accounts for training the dataset-specific decoders and sampling 2500 samples. DiffICL's pre-training time is excluded from this analysis. In practice, generating new samples requires only inference with the

[9] https://github.com/MinkaiXu/TabDiff

[10] https://github.com/amazon-science/tabsyn

[11] https://github.com/tabularis-ai/be_great

pre-trained model weights. Notably, DiffICL demonstrates top-tier computational efficiency, ranking among the most efficient methods alongside tabDDPM and TVAE. We also observe that GReaT incurs a substantially higher computational cost when the guided-samples=True setting is required to generate valid samples (as seen in the kr-vs-kp and CPU Activity datasets).

C.4 Correlation between quality metrics on each evaluation dataset

The right panel of Figure 5 shows the correlation matrix of quality metrics on the diabetes dataset. Here, we further report the corresponding correlation matrices for all 14 datasets in Figure 7. These results show that the disagreement between distribution-based quality metrics and downstream task performance is common across datasets.

D Limitations

A current limitation of DiffICL is that its inference stage requires training decoders for each new dataset. We split the available data into a context set and a query set: the context set is used to condition latent generation, while the query set is used to train feature-wise decoders that map generated latent representations back to raw tabular values. This design improves reconstruction accuracy, but it also introduces an additional data requirement, since part of the available data must be reserved for decoder training.

A more data-efficient alternative would be to learn a universal decoder during pretraining, so that the same decoder could map query embeddings to raw feature values across different datasets and columns. We explored this direction, but found that a simple shared MLP decoder, which maps an arbitrary 192-dimensional feature embedding to a scalar value, leads to noticeable reconstruction errors across heterogeneous datasets and feature types. This suggests that universal decoding for tabular foundation models is non-trivial, because raw feature spaces differ substantially across datasets. Therefore, we adopt dataset-specific lightweight decoders in the current implementation. Although this choice sacrifices a small amount of sample efficiency, we find empirically that the decoder can be trained accurately with as few as several dozen samples. This allows DiffICL to achieve more reliable feature reconstruction while keeping the inference procedure practical.

E Additional Experiments

E.1 Membership Inference Attack Evaluation

We complement DCROverfit with membership inference attacks (MIAs), which assess whether an adversary can infer if a particular record was included in a generator's training set. We adopt a black-box shadow-model attack framework following [48, 44]. For each dataset D, we divide it into disjoint shadow pool D_{sh} and target pool D_{tar} with a 1:1 ratio. The shadow pool D_{sh} is used to train the attacker A, while the target pool is used to evaluate it. Specifically, we randomly split D_{sh} into a training subset $D_{\mathrm{sh,train}}$ and a held-out subset $D_{\mathrm{sh,holdout}}$. We train a shadow generator on $D_{\mathrm{sh,train}}$ and use it to produce synthetic data S. For each real record x in either subset, we extract attack features $\phi(x, S)$ and assign label 1 if $x \in D_{\mathrm{sh,train}}$, and 0 otherwise. We repeat this procedure with 100 independent random splits and pool the resulting feature–label pairs to train the binary attack classifier A. For evaluation, we split D_{tar} into $D_{\mathrm{tar,train}}$ and $D_{\mathrm{tar,holdout}}$. We train a generator on $D_{\mathrm{tar,train}}$ and generate synthetic data S. The trained attack classifier then assigns each record $x \in D_{\mathrm{tar}}$ a membership score $A(\phi(x, S))$. We compare these scores with the true membership labels using AUC. An AUC close to 0.5 indicates that the evaluated attack distinguishes members from non-members no better than random ranking, whereas a higher AUC indicates greater membership leakage. Small deviations below 0.5 can occur due to finite-sample variability. Table 7 reports the mean and standard deviation of attack AUC across five independent repetitions. DiffICL achieves the lowest mean attack AUC among the evaluated methods on diabetes, kc2, and wdbc. On ilpd, its attack AUC is also close to random ranking.

E.2 Effect of Pretraining Corpus Size

We investigate the effect of pretraining corpus size by training DiffICL on 50, 100, 200, and 826 datasets, while keeping the diffusion architecture, optimization procedure, and number of training



Figure 7: Correlation between quality metrics on each evaluation dataset.

Table 7: Membership inference attack AUC, multiplied by 100. Results are reported as mean $\pm$ standard deviation over five independent repetitions. A value of 50 corresponds to random ranking.

Method	Diabetes	ILPD	KC2	WDBC
CTGAN	51.2 ± 3.6	49.7 ± 5.1	66.9 ± 4.6	56.0 ± 4.9
CTABGAN+	55.0 ± 2.6	53.3 ± 3.4	67.3 ± 3.2	59.0 ± 3.8
TVAE	53.7 ± 4.5	48.9 ± 3.8	59.8 ± 4.0	59.1 ± 2.2
tabDDPM	55.4 ± 3.2	50.3 ± 3.6	69.2 ± 4.2	61.4 ± 2.9
TabSYN	54.5 ± 3.3	52.7 ± 3.5	60.7 ± 3.7	60.9 ± 3.6
TabDiff	51.6 ± 2.8	50.5 ± 2.9	57.2 ± 3.5	57.6 ± 3.7
DiffICL	50.7 ± 5.1	50.9 ± 2.9	54.0 ± 2.8	55.6 ± 3.6

steps unchanged. We evaluate generation quality using the performance of an XGBoost model trained on synthetic data and tested on held-out real data, and assess distance-based overfitting using DCROverfit. As shown in Table 8, average generation quality improves consistently with corpus size, increasing from 58.4 with 50 datasets to 74.0 with the full corpus. This supports the importance of pretraining diversity for learning transferable distributional structure. Table 8 also shows a non-monotonic pattern in DCROverfit: the average score decreases from 76.8 to 74.8 and then returns to 76.1. These results indicate that larger-scale pretraining substantially improves generation quality without introducing additional privacy risks.

Table 8: Effect of pretraining corpus size on generation quality and privacy. Generation quality is measured by downstream XGBoost performance (AUC for classification and R^2 for regression, both multiplied by 100). Privacy is evaluated using DCROverfit (%). Column headings indicate the number of pretraining datasets.

	Generation quality				DCROverfit			
Dataset	50	100	200	826	50	100	200	826
Abalone	41.4	42.9	47.8	56.0	63.4	63.0	62.8	62.6
Auction verification	48.2	55.9	78.0	83.2	76.3	76.9	77.1	76.7
CPU activity	41.6	78.8	91.5	96.8	71.3	69.1	68.3	69.9
Diabetes	79.8	81.5	82.8	82.0	65.5	65.4	64.7	65.7
Health insurance	18.7	21.7	22.3	38.5	74.7	74.5	73.6	73.3
ILPD	66.8	63.9	67.0	69.4	76.3	73.6	72.1	73.9
KC2	79.4	81.4	80.3	81.0	45.6	45.9	45.3	46.1
KR-vs-KP	86.5	95.0	97.8	99.7	100.0	100.0	100.0	100.0
Red wine	15.7	12.0	17.5	34.0	87.6	85.4	84.2	86.9
Vehicle	80.5	86.8	89.4	93.2	91.2	89.0	86.0	89.7
WDBC	98.5	99.2	99.3	98.8	93.7	91.2	88.1	92.5
White wine	1.2	3.9	13.4	30.6	69.0	68.4	68.2	67.8
Wind	64.9	68.0	68.6	75.7	80.4	75.4	79.4	79.7
Mfeat-zernike	95.0	94.1	94.6	96.4	79.5	79.1	76.5	80.4
Avg.	58.4	63.2	67.9	74.0	76.8	75.5	74.8	76.1

E.3 The Impact of Context Ratio

Context ratio is the ratio we split a new dataset D into D_{ctx} and D_{qry}. In all experiments in Section 4, we set the context ratio of DiffICL to 0.3. We further study the sensitivity of DiffICL to this hyperparameter. As shown in Figure 8, we evaluate DiffICL on each dataset with context ratios ranging from 0.2 to 0.8. Each context ratio is a red point. The results show that DiffICL is robust to the choice of context ratio: the performance variation caused by this hyperparameter is small compared with the gap between DiffICL and the baseline methods.

E.4 Effect of the number of training samples.

We further study how different methods behave when the target dataset contains fewer training samples. For each dataset, we randomly subsample the original training set without replacement using ratios in $\{0.2, 0.4, 0.6, 0.8, 1.0\}$. Each generative model is then trained on the subsampled

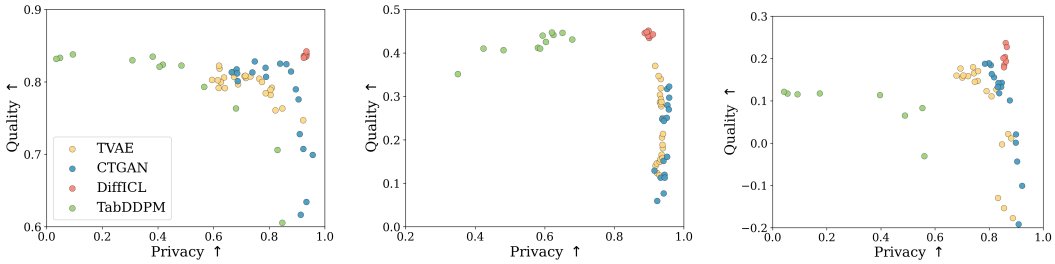


Figure 8: Quality–privacy tradeoffs under different training configurations.Each point corresponds to a model checkpoint under different training configurations. For tabDDPM, TVAE, and CTGAN, the points are obtained by varying the number of training steps from 500 to 30000. For DiffICL, the points are obtained by varying the context ratio from 0.2 to 0.8. The three subplots correspond to the Adult, Abalone, and Health Insurance datasets, respectively, all with $N = 200$ training samples.

training set and used to generate 2,500 synthetic samples. The ratio 1.0 corresponds to the full training set used in the main experiments.

The first row of Figure 9 reports the performance of an XGBoost model trained on synthetic data and evaluated on the held-out test set. DiffICL remains effective even when only a small fraction (e.g. 0.2) of the training data is available. For example, on Abalone, DiffICL achieves an R^2 of 0.49 using only 417 training samples. On Healthcare, with 2,107 training samples, DiffICL reaches an R^2 of 0.358, outperforming the second-best method, TabDiff, which obtains 0.257. These results suggest that DiffICL is less sensitive to limited target-dataset size, consistent with its use of transferable priors learned during cross-dataset pretraining.

The second row of Figure 9 shows the effect of using synthetic data for augmentation, where the learner is trained on the mixture of real and synthetic samples. We find that adding low-quality synthetic data often does not severely degrade the learner, even when the synthetic-only performance is poor or even negative in terms of R^2. In contrast, adding high-quality samples generated by DiffICL improves downstream prediction: the red dashed curve, which represents XGBoost trained with real and DiffICL-generated data, lies above the gray dashed curve, which represents XGBoost trained only on real data.

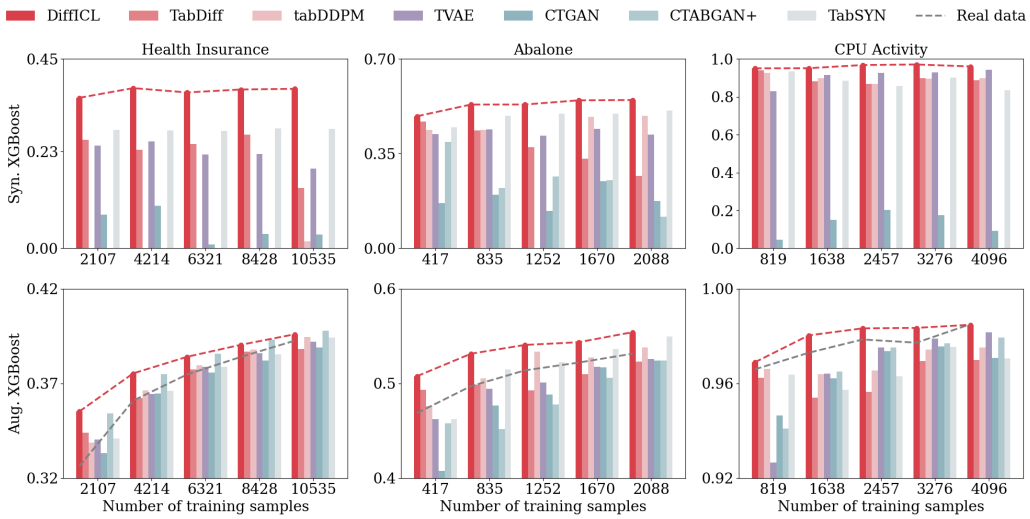


Figure 9: Effect of the number of training samples on synthetic-data quality and data augmentation performance. **Top**: XGBoost is trained on synthetic data generated by each method and evaluated on the held-out test set. **Bottom**: XGBoost is trained on the mixture of real training data and synthetic data; the gray dashed curve reports the performance of XGBoost trained only on real data, and the red dashed curve highlights the augmentation performance of DiffICL.

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: The main claims are stated in the Abstract and Introduction.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: We discuss the limitation in the Appendix.

   Guidelines:

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

3. **Theory assumptions and proofs**

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

   Answer: [N/A]

Justification: This paper provides empirical contributions and does not include theoretical results.

Guidelines:

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

4. **Experimental result reproducibility**

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

   Answer: [Yes]

   Justification: We provide detailed pretraining and inference procedures in the Appendix, including the model architecture, training objective, hyperparameters, data preprocessing protocol, and evaluation setup. The pretraining datasets are collected from public sources.

   Guidelines:

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

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

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

Answer: [No]

Justification: The code and model checkpoints are not yet publicly available. We plan to release them to support reproducibility.

Guidelines:

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

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: We provide training and test details in the Appendix.

   Guidelines:

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

7. **Experiment statistical significance**

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

   Answer: [Yes]

   Justification: The tables of our main results follow a standard formulation of mean $\pm$ std.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)

- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: We provide the type and number of compute workers and the running time in the Appendix.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: The research conducted in the paper conforms with the NeurIPS Code of Ethics.

   Guidelines:

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

10. **Broader impacts**

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

    Answer: [Yes]

    Justification: DiffICL may have positive societal impacts by improving the utility of synthetic tabular data while reducing empirical memorization risks. However, synthetic data generation can also be misused to create misleading datasets, and empirical privacy metrics do not provide formal privacy guarantees. In addition, generated data may inherit biases or artifacts from the original datasets and pretraining corpus. We discuss these risks and emphasize that DiffICL should not be treated as a substitute for formal privacy mechanisms such as differential privacy in high-stakes deployments.

    Guidelines:

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

11. **Safeguards**

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

    Answer: [N/A]

    Justification: The paper does not release data or models that have a high risk for misuse. We plan to release code for reproducibility, but it does not include scraped datasets, sensitive data, or models intended for unrestricted generation of high-risk content.

    Guidelines:

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

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: We properly credit the creators and original owners of the assets used in this paper. We cite the original papers and sources for the datasets, baselines, and pretrained models used in this work.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.

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

13. **New assets**

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

    Answer: [No]

    Justification: The code and model checkpoints are not yet publicly available. We plan to release them to support reproducibility.

    Guidelines:

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

14. **Crowdsourcing and research with human subjects**

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

    Answer: [N/A]

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

    Guidelines:

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

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

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

    Answer: [N/A]

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

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.

- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: LLMs were used only for writing polishing and editing. They were not used as an important, original, or non-standard component of the core methodology. The research problem, main findings, and core algorithmic design were developed by the authors.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.