Doubling Your Data in Minutes: Ultra-fast Tabular Data Generation via LLM-Induced Dependency Graphs

Anonymous ACL submission

Abstract

Tabular data is critical across diverse domains, yet high-quality datasets remain scarce due to privacy concerns and the cost of collection. Contemporary approaches adopt large language models (LLMs) for tabular augmentation, but exhibit two major limitations: (1) dense dependency modeling among tabular features that can introduce bias, and (2) high computational overhead in sampling. To address these issues, we propose SPADA (for SPArse Dependency-driven Augmentation), a lightweight generative framework that explicitly captures sparse dependencies via an LLM-induced graph. We treat each feature as a node and synthesize values by traversing the graph, conditioning each feature solely on its parent nodes. We explore two synthesis strategies: a non-parametric method using Gaussian kernel density estimation, and a conditional normalizing flow model that learns invertible mappings for conditional density estimation. Experiments on four datasets show that SPADA reduces constraint violations by 4% compared to diffusion-based methods and accelerates generation by nearly 9,500× over LLM-based baselines.[1]

1 Introduction

With the rapid advancement of data science, tabular data has become a fundamental format for storing information across diverse domains, including finance (Sharma et al., 2024), medicine (Ulmer et al., 2020), cybersecurity (Buczak and Guven, 2016), and many more. Systems powered by tabular data, e.g., decision support tools (Borisov et al., 2021) and anomaly detection algorithms (Wang et al., 2024), have demonstrated irreplaceable value in real-world applications. Meanwhile, the high cost of data collection, coupled with privacy concerns, has rendered high-quality tabular datasets extremely scarce in practice (Yang et al., 2024). This scarcity underscores the urgent need for low-cost tabular augmentation methods capable of generating realistic and privacy-preserving tables.

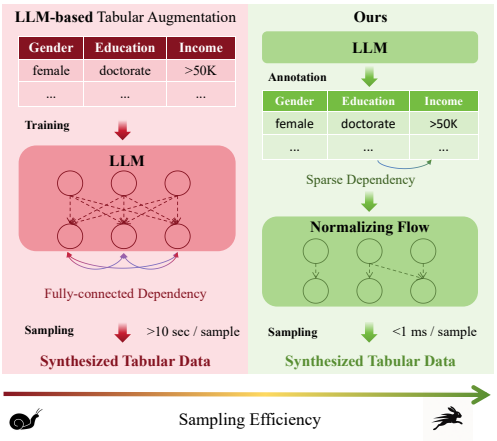


Figure 1: A comparison between SPADA and LLM-based approaches. Our approach leverages LLMs to annotate sparse dependency structures, effectively mitigating the bias introduced by fully connected feature assumptions in traditional methods.

However, the complex tabular feature dependencies pose significant challenges to high-quality generation (Ren et al., 2025). For example, in a population census dataset, it is rather implausible for an individual aged "18" to have the occupation "professor." Methods that rely solely on learning statistical information from data, such as TVAE (Xu et al., 2019) and CopulaGAN (Kamthe et al., 2021), lack external knowledge and therefore often generate samples with logical inconsistencies (Yang et al., 2024). This failure to capture dependencies not only reduces the fidelity of synthetic data with respect to the real data, but also introduces unpredictable biases that increase risks when used in downstream systems (Park and Ko, 2024).

To capture tabular dependencies, LLMs have recently been adopted as implicit knowledge bases. Methods like GReaT (Borisov et al., 2023) pioneered this direction by converting each record into a sequence of <*subject, predicate, object*>-phrases,

[1]Our code is available at https://anonymous.4open.science/r/SPADA-5C7C

i.e., "*feature* is *value*" templates, enabling fine-tuning of LLMs. Building on this, Xu et al. (2025) further optimized the ordering of these phrases to potentially strengthen these dependencies.

Despite these advancements, we argue that LLM-based methods suffer from two critical limitations: **Overly dense dependency modeling.** LLMs theoretically produce fully-connected information fusion among input features in hidden layers, whereas real-world entities are typically structured with sparse and heterogeneous relations (Liu et al., 2023). The fine-tuning leads to unintended associations between independent features, e.g., gender and education, reflecting inherent biases in the training data (More and Bradbury, 2025). Therefore, tables generated by fine-tuned LLMs may fail to achieve the realism compared to real-world data. **High computational cost in sampling.** LLM-based approaches require autoregressive generation of each feature value, causing the model to repeatedly pass through all layers to produce a single sample. As reported by Borisov et al. (2023), GPT-2 (Radford et al., 2019) takes 17 seconds on average to generate one sample on two GPUs. This substantial time overhead (Xia et al., 2024) renders LLM-based methods impractical for realistic scenarios that demand large-scale data augmentation.

To address these limitations, we propose the SPADA, a novel synthesizer that explicitly models feature dependencies while dramatically reducing computational cost, as shown in Figure 1. Specifically, we treat each feature as a node and use an LLM to extract a relational structure among them to build a directed graph (Zhou et al., 2022). We then perform a topological traversal (Zheng et al., 2018) over this graph to sequentially generate feature values. Unlike traditional approaches, SPADA enforces sparsity by conditioning the synthesis of a node's value solely on its parent nodes. After that, we introduce two synthesis strategies instead of LLM-based generation: (1) non-parametric statistics and (2) conditional normalizing flows (NF) (Rezende and Mohamed, 2015a).

Our contributions are summarized as follows:

1. We propose a novel method for tabular augmentation. Our approach leverages LLMs to capture sparse dependencies among features and performs conditional generation using either KDE or NFs. This design decouples dependency modeling from generation, combining the structural insight of LLMs with the efficiency of lightweight models. It enables logical consistency among features while avoiding the costly autoregressive generation.

2. SPADA achieves an unprecedented improvement in sampling efficiency compared to existing LLM-based baselines, reducing generation time by nearly 9,500× while improving the quality of the generated data.

3. We conduct comprehensive experiments against SoTA methods, across four datasets, spanning binary, multi-class classification and regression tasks. The evaluation includes downstream utility, distributional alignment, privacy leakage, visual comparison, and discriminator measure. SPADA demonstrates remarkably reliable, scalable, and high-quality performance across the board.

2 Related Work

Recent advances in tabular data generation have led to a diverse range of deep generative models, including GANs, VAEs, diffusion models and LLMs. These approaches are often evaluated along four key dimensions: utility, realism, statistical fidelity, and privacy preservation (Borisov et al., 2022; Stoian et al., 2025). However, a critical challenge persists in balancing these requirements while maintaining scalability and transparency.

Contemporary methods focus on end-to-end generation, learning the full joint distribution of tabular data using black-box neural architectures (Hollmann et al., 2025). GAN-based models such as CTGAN (Xu et al., 2019), Ganblr (Zhang et al., 2021), and Ctab-gan+ improve sample realism and utility but often suffer from mode collapse and poor interpretability. More recently, diffusion-based methods like TabDDPM (Kotelnikov et al., 2023), Findiff (Sattarov et al., 2023), and TabSyn (Zhang et al., 2024) have shown promise in generating high-quality samples with better coverage. Yet, they entail significant computational overhead due to their iterative sampling procedures.

A parallel line of research investigates alignment by incorporating knowledge. Approaches such as DRL (Stoian and Giunchiglia, 2025) encode structural constraints through Bayesian networks or auxiliary autoencoders, but these typically require manual rule specification and are limited to discrete features. LLM-based models such as GReaT (Borisov et al., 2023), Pred-LLM (Nguyen et al., 2024), and LLM-TabFlow (Long et al., 2025) attempt to model

dependencies through textual representations; however, they treat generation as a monolithic text completion task, leading to excessive sampling latency and entangled dependencies.

Unlike prior work, SPADA decouples dependency modeling from generation. We use LLMs to extract a sparse, interpretable dependency graph, which then guides sampling via either parametric or non-parametric models. This design enhances interpretability, accelerates generation, and improves logical consistency, offering a scalable and transparent framework for structure-aware synthesis.

3 Methodology

3.1 Problem Formulation

Let $T = \{\mathbf{t}_1, \ldots, \mathbf{t}_N\}$ denote a tabular dataset with N samples. Each $\mathbf{t}_i \in \mathbb{R}^M$ is a record consisting of M values, i.e. $\mathbf{t}_i = \{v_{i1}, \ldots, v_{iM}\}$, where v_{ij} denotes the value of feature $f_j \in \mathcal{F}$ in sample $\mathbf{t}_i$. Following the standard taxonomy in tabular toolbox (Patki et al., 2016), we partition the feature set $\mathcal{F} = \{f_1, \ldots, f_M\}$ into two disjoint subsets:

- $\mathcal{F}_{\text{num}}$ contains **numerical features**, and
- $\mathcal{F}_{\text{cat}}$: contains **categorical features** such as boolean and discrete text strings.

A key distinction lies in their value domains. For $f_j \in \mathcal{F}_{\text{num}}$, the domain is continuous and potentially unbounded. Therefore, a generated value $v'_{ij} \in \mathbb{R}$ may not appear in T. On the other hand, for $f_j \in \mathcal{F}_{\text{cat}}$, the domain is a finite unordered set. Hence, any synthesized value must be drawn from the observed support: $v'_{ij} \in \mathcal{V}_j = \{v_{1j}, \ldots, v_{Nj}\}$.

Our objective is to synthesize a new dataset $T' = \{\mathbf{t}'_1, \ldots, \mathbf{t}'_{N'}\}$ with low computational cost, where each $\mathbf{t}'_i \notin T$ follows the same feature structure in T. That is, for all $j \in \{1, \ldots, M\}$, the value v'_{ij} corresponds to the same feature $f_j \in \mathcal{F}$ in T.

To ensure the realism of synthetic data, we impose the following constraints: (1) **Distributional similarity**, that is, the marginal and joint distributions of T' should match those of T; specifically, $\mathcal{D}_{T'} \approx \mathcal{D}_T$, where $\mathcal{D}_T$ denotes the empirical distribution of the original dataset, and (2) **Logical consistency**, i.e., for any subset of feature values $\{v'_{ij}, \ldots, v'_{ik}\}$ within a synthesized sample, there should be no semantic implausibilities or contradictions (e.g, a retired 18-years-old professor).

3.2 Dependency Graph

In traditional LLM-based methods (Borisov et al., 2023), fully-connected modeling typically introduce training bias (Liu et al., 2023). For example, given two logically independent features f_a and f_b, the attention weight $\text{Att}(v_a, v_b)$ between their corresponding values v_a and v_b should ideally approach zero. However, due to the lack of explicit constraints on feature correlations, the $\text{Att}(v_a, v_b)$ will inevitably produce positive values (Vaswani et al., 2017), resulting in a spurious correlation that distorts the feature dependency structure.

To introduce the sparse constraints among features, we employ LLMs to explicitly generate the logical dependencies. Specifically, we prompt GPT-4o (Achiam et al., 2023) with the following inputs: (1) P_{Intro}, a brief description of T; (2) $\mathcal{F}$; and (3) P_{Task}, a task-specific instruction describing the goal of dependency identification. For each target feature $f_j \in \mathcal{F}$, the model returns a subset $\hat{F}_{f_j} \subseteq \mathcal{F}$ representing the features on which f_j depends:

$$\begin{aligned} \hat{F}_{f_j} &= \text{LLM}(\mathcal{F}, f_j, \text{P}_{\text{Intro}}, \text{P}_{\text{Task}}), \\ \text{where } f_j &\notin \hat{F}_{f_j} \quad \text{and} \quad \hat{F}_{f_j} \subseteq \mathcal{F}. \end{aligned} \tag{1}$$

Here, the prompt template we used are detailed in Appendix G. In designing prompts, we follow the standard methodology outlined in (Amatriain, 2024). For example, an output may be "`marital status` $\rightarrow$ `age`", indicating that "marital status" constrains "age", i.e., individuals below a certain age are unlikely to be married or divorced.

To structurally represent the generated dependencies, we construct a directed dependency graph $G = (\mathcal{F}, E)$. Here, each node $f_j \in \mathcal{F}$ is a feature, and each edge $(f \rightarrow f_j) \in E$ denotes that feature f_i constrains feature f_j. In practice, for f_j which is not constrained by any other feature, its dependency set $\hat{F}_{f_j}$ can be empty. Such features typically represent inherent properties of an entity, such as "*gender*". To ensure that all features are integrated into a unified graph, we introduce an artificial root node $f_{\text{root}} \notin F$. For any feature f_j with $\hat{F}_{f_j} = \emptyset$, we define a dependency from f_{root} to f_j, resulting in the following extended definition of the edge set:

$$E = \bigcup_{f_j \in F} \begin{cases} \left\{ (f_i \rightarrow f_j) \;\middle|\; f_i \in \hat{F}_{f_j} \right\}, & \text{if } \hat{F}_{f_j} \neq \emptyset \\ \{(f_{\text{root}} \rightarrow f_j)\}, & \text{if } \hat{F}_{f_j} = \emptyset \end{cases} \tag{2}$$

Note that f_{root} does not hold realistic significance. We refer readers to Appendix H for the complete dependencies extracted from the datasets we used.

Following (Xu et al., 2025), we prioritize features that impose constraints on others, and subsequently generate the values of child nodes conditioned on their parent node values to prevent logical inconsistencies. In practice, we traverse the dependency graph G starting from f_{root} and follow the directed edges to generate feature values in a dependency-aware manner:

$$\forall f_j \in F, \quad v_j \sim p(v_j \mid \{v_k : f_k \in \hat{F}_{f_j}\}). \quad (3)$$

Mathematically, given G, we define a topological ordering over the F, denoted as:

$$f^{(1)} \prec f^{(2)} \prec \cdots \prec f^{(M)}, \quad (4)$$

where $f^{(i)} \prec f^{(j)}$ implies that $f^{(i)}$ is a parent node of $f^{(j)}$ in G. During inference, we synthesize feature values following this topological order and consider the constraints from their parent nodes

$$v^{(i)} \sim p(v^{(i)} \mid \{v^{(k)} \mid f^{(k)} \in \hat{F}_{f^{(i)}}\}), \quad \text{for } i = 1, \ldots, M \quad (5)$$

Nevertheless, the generated constraints $\hat{F}_f$ may introduce cycles in the graph G, thereby preventing the derivation of a valid topological ordering. For instance, "latitude" and "longitude" could determine the "country", while the "country" might also constrain the ranges of "latitude" and "longitude". To avoid encountering cycles, we employ an Integer Linear Programming (ILP) algorithm (Nemhauser and Wolsey, 1988) to break the cycles by deleting the fewest edges that participate in any cycle of G. Our objective function O_{ILP} is shown in Eq. (6).

$$O_{\text{ILP}} = \min \sum_{(f_i \to f_j) \in E} e_{(f_i \to f_j)}, \quad (6)$$

where $e_{(f_i \to f_j)} \in \{0, 1\} \quad \forall (f_i \to f_j) \in E$. After obtaining a directed acyclic graph (DAG), we proceed to synthesize feature values for each node sequentially, as defined in Eq. (5). To reduce the training and sampling costs, we propose two LLM-independent strategies: a non-parametric method and a Normalizing Flow (NF) approach.

3.3 Non-parametric Synthesis

We propose a training-free method based on KDE to estimate conditional probabilities of a value v from its similar instances in T. Our motivation is rooted in the fact that probabilistic models are often more effective than the neural models when the training data T is not sufficient (Xu et al., 2021; Grinsztajn et al., 2022). Furthermore, non-parametric synthesis eliminates the need for resource-intensive training associated with LLMs.

Specifically, for a target feature f_j to be synthesized, given its dependency set $\hat{F}_{f_j}$, we filter T to obtain a subset $\hat{T}$ consisting of samples that match the values of all features in $\hat{F}_{f_j}$:

$$\hat{T} = \left\{ t_i \in T \;\middle|\; \forall f_k \in \hat{F}_{f_j},\ v_{ik} = v_k^* \right\}, \quad (7)$$

where v_k^* denotes the generated value for feature $f_k \in \hat{F}_{f_j}$ during the inference process. After that, we estimate the conditional distribution of $p(v_j \mid \hat{T})$ and sample a synthesized value from it.

3.3.1 Fuzzy Matching

However, $\hat{T}$ may be empty when $\{v_k^*\}$ are rarely observed in the original dataset. To avoid this issue, we employ a range query to relax the exact match constraint, instead of requiring all (v_k^*, v_{ik}) pairs to match exactly. For each target feature f_j, we define the fuzzy candidate set as:

$$\tilde{T} = \left\{ t_i \in T \;\middle|\; \text{Dist}\left(\mathbf{v}^*_{\hat{F}_{f_j}}, \mathbf{v}_{i,\hat{F}_{f_j}} \right) \leq \epsilon \right\} \quad (8)$$

Here, $\mathbf{v}^*_{\hat{F}_{f_j}}$ is the vector of previously generated conditioning values, $\mathbf{v}_{i,\hat{F}_{f_j}}$ is the corresponding vector of values in sample t_i, and $\text{Dist}(\cdot, \cdot)$ denotes a Hamming distance (Norouzi et al., 2012) for categorical features in $\mathcal{F}_{\text{cat}}$ and an L1-Norm for numerical features in $\mathcal{F}_{\text{num}}$, and ϵ is the tolerance threshold controlling the fuzziness of the match. This fuzzy matching ensures that we can always obtaine a non-empty $\tilde{T}$, even under sparse conditioning combinations.

3.3.2 BallTree

To accelate the matching process, we construct a BallTree (Omohundro, 1989) on T in advance, reducing the time complexity of nearest neighbor search. Specifically, for each sample $t_i \in T$, we project it onto the subspace spanned by the dependency features $\hat{F}_{f_j}$, and organize these projected vectors into a BallTree structure:

$$\mathcal{B}_{f_j} = \text{BallTree}\left(\left\{ \mathbf{v}_{ij}^{\hat{F}_{f_j}} \mid t_i \in T \right\} \right), \quad (9)$$

where $\mathbf{v}_{ij}^{\hat{F}_{f_j}}$ denotes the feature vector of sample t_i restricted to the dependency set $\hat{F}_{f_j}$.

The BallTrees make our synthesis method significantly more effective, especially when T is large.

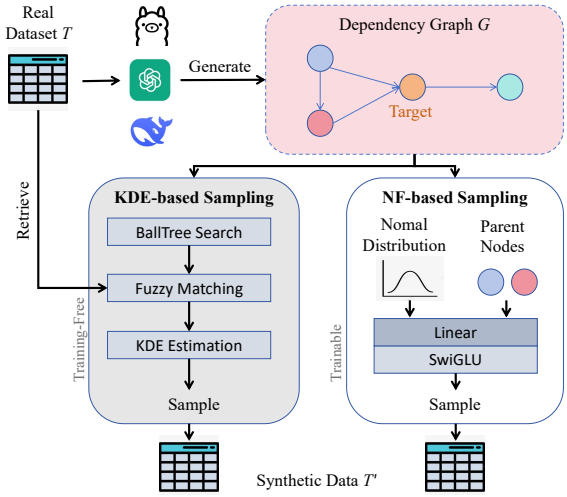


Figure 2: An overview of the proposed method. We leverage an LLM to extract dependencies among tabular features. Based on the resulting dependency graph, we obtain synthetic data using KDE or NF-based sampling.

By pre-building a BallTree for each $f_j \in F_{\text{num}}$ based on the corresponding dependency features, we achieve fast query times while maintaining the fidelity of conditional sampling. We refer readers to Appendix A.3 for the time complexity optimization introduced by the BallTree.

3.3.3 Kernel Density Estimation

For f in F_{num}, we adopt KDE, aiming to model the distribution of continuous variables and can create new, smooth values. For categorical features, we simply draw from the finite set of observed categories represented as $p(v_j \mid \hat{T})$, which is merely applicable to F_{cat} with a finite set of discrete values and can only generate values previously observed in T. To estimate the continuous probability density of a numerical feature $f_j \in F_{\text{num}}$, we apply a Gaussian Kernel to model the distribution (Bishop and Nasrabadi, 2006), as defined below:

$$\hat{p}(v_j \mid \tilde{T}) = \frac{1}{|\tilde{T}|h} \sum_{t_i \in \tilde{T}} K\left(\frac{v_j - v_{ij}}{h}\right), \quad (10)$$

where $K(\cdot)$ is the Gaussian kernel and h is the bandwidth parameter.

Finally, we generate a new sample t' by sequentially assigning values to nodes following the traversal order in Eq. (4). This non-parametric synthesis strategy eliminates the need for model training, while enabling the generation of continuous values beyond the discrete support of the original dataset. As a result, it reduces the synthesis cost and enhances the diversity of the synthesized data.

3.4 Conditional Normalizing Flow

The non-parametric synthesis method achieves a theoretically lower computational cost compared to LLM-based generation. However, when the size of T is small, the number of matching samples in $\hat{T}$ may be insufficient, leading to biased probability estimates under the fuzzy matching strategy. Conversely, when T is large, repeated access to the dataset during sampling increases the sampling overhead. To address this, we introduce a parametric generative method based on conditional NFs, enabling efficient synthesis without data access.

3.4.1 Theoretical Framework

NFs transform a standard Gaussian distribution into the probability density of our target feature value v through a sequence of differentiable mappings. Specifically, let $z \sim p_Z(z)$ be a latent variable sampled from the standard Gaussian distribution, and let f_θ be a learnable transformation parameterized by θ. The target feature value v is then given by:

$$v = f_\theta(z \mid \{v_k \mid f_k \in \hat{F}_f\}), \quad (11)$$

where the transformation is conditioned on the values of features in $\hat{F}_f$. We encode categorical features in F_{cat} by using a label encoder (Pedregosa et al., 2011) into continuous representations. The conditional density is:

$$p(v \mid \{v_k\}) = p_Z(f_\theta^{-1}(v \mid \{v_k\})) \left| \det \left(\frac{\partial f_\theta^{-1}}{\partial v} \right) \right|, \quad (12)$$

where the Jacobian determinant (Rezende and Mohamed, 2015b) captures the local volume change induced by the transformation.

3.4.2 Implementation Strategy

In training, we maximize the likelihood of observed feature values under the modeled conditional distribution. The loss funtion is shown in Eq. (13):

$$\mathcal{L}(\theta) = -\frac{1}{N} \sum_{i=1}^{N} \log p(v \mid \{v_k\}; \theta). \quad (13)$$

In practice, we parameterize f_θ using a fully connected neural layer with SwiGLU (Shazeer, 2020). The model takes as input the latent variable z and the conditioning values $\{v_k\}$, and outputs the synthesized value v. This NF enables learning expressive conditional distributions over both continuous and categorical features, facilitating high-fidelity and efficient data synthesis. An overview of our framework is shown in Figure 2.

Dataset		Original		TVAE		CTGAN		GReaT		TabSyn		**Ours (w/KDE)**		**Ours (w/NF)**	
		ACC/MSE	F1	ACC/MSE	F1	ACC/MSE	F1	ACC/MSE	F1	ACC/MSE	F1	ACC/MSE	F1	ACC/MSE	F1
Income (↑)	DT	80.83%	0.73	**79.50%**	0.71	76.27%	0.64	59.85%	0.60	77.58%	0.70	77.30%	0.69	78.95%	**0.71**
	RF	85.40%	0.78	82.43%	0.75	82.50%	0.71	69.42%	0.69	83.67%	**0.77**	**84.18%**	0.75	84.14%	0.75
	LR	80.32%	0.67	78.05%	0.66	78.90%	0.61	69.57%	0.70	80.21%	**0.70**	**80.42%**	0.62	78.61%	0.56
HELOC (↑)	DT	61.67%	0.61	**64.30%**	**0.64**	63.90%	0.64	61.31%	0.61	62.13%	0.62	59.85%	0.60	63.24%	0.63
	RF	71.39%	0.71	68.91%	0.68	62.78%	0.63	70.18%	0.70	**70.73%**	0.71	69.41%	0.69	70.58%	**0.71**
	LR	69.72%	0.70	65.06%	0.65	65.47%	0.65	68.51%	0.68	69.77%	0.70	69.57%	0.70	**70.53%**	**0.70**
Iris (↑)	DT	100%	1.00	55.17%	0.47	62.07%	0.56	41.38%	0.36	96.55%	0.97	89.66%	0.90	**100%**	**1.00**
	RF	100%	1.00	55.17%	0.49	41.38%	0.37	44.83%	0.35	**100%**	**1.00**	**100%**	**1.00**	**100%**	**1.00**
	LR	100%	1.00	62.07%	0.56	51.72%	0.44	41.38%	0.34	**100%**	**1.00**	**100%**	**1.00**	**100%**	**1.00**
Housing (↓)	DT	0.14	N/A	1.28	N/A	5.49	N/A	0.27	N/A	0.29	N/A	0.23	N/A	**0.17**	N/A
	RF	0.08	N/A	0.52	N/A	3.23	N/A	0.13	N/A	0.13	N/A	0.11	N/A	**0.11**	N/A
	LR	0.38	N/A	0.48	N/A	1.62	N/A	**0.40**	N/A	0.40	N/A	0.44	N/A	0.45	N/A

Table 1: Performance of classifiers/regressors trained on synthetic data for downstream tasks. **Bold** indicates the best performance, and underline indicates the second-best. "Original" is the original dataset T. "ACC" stands for accuracy and "MSE" stands for mean squared error. Besides in the original dataset, all classifiers are trained on synthetic data and tested on real ones.

Table 2: Violation rates ↓. We present measurements with 95% confidence interval. **Bold** indicates the best performance, and underline indicates the second-best.

Dataset	TVAE	CTGAN	GReaT	TabSyn	**Ours (w/KDE)**	**Ours (w/NF)**
Income	4.21 ±0.64%	34.31 ±1.18%	**0.00 ±0.00%**	2.32 ±0.49%	3.56 ±0.98%	**0.00** ±0.00%
Housing	15.55 ±0.55%	34.48 ±0.72%	3.61 ±0.28%	10.70 ±0.72%	5.26 ±0.62%	**1.37** ±0.18%
Mean (↓)	9.88%	34.40%	1.81%	6.51%	4.41%	**0.69%**

4 Experiments

4.1 Datasets

Binary Classification. The *Adult Income* dataset (Becker and Kohavi, 1996) contains 16 demographic and occupational features, which are used to predict an individual's annual income level. The *Home Equity Line of Credit* (HELOC) dataset (Oliabev, 2022) includes 24 credit-related features extracted from people's credit reports. The task is to predict whether an individual will repay their HELOC amount within the next two years.

Multi-class Classification. The well-known *Iris* dataset (Fisher, 1936) comprises four numerical features describing the sepal and petal dimensions of iris flowers. The task is to classify the sample into an iris species.

Regression. The *California Housing* dataset (Nugent, 2018) consists of 10 features related to housing and geographic attributes. Our task is to predict the latitude and longitude of a property.

4.2 Evaluation Metrics

Downstream Utility. We generated synthetic datasets of the same size as the original datasets and trained decision tree (DT), random forest (RF), and logistic regression (LR) models as classifiers and regressors, following the respective tasks of the datasets used. For classification, we reported accuracy and F1, while for regression tasks, we reported Mean Squared Errors. The results are presented in Table 1.

Privacy Protection. Following (Zhang et al., 2024), we used the L1 norm to calculate the Distance to Closest Records (DCR) to T. We normalize each column and compute the average. A high DCR suggests minimal overlap between the feature values of synthetic data and those in the original dataset, which contributes to stronger privacy preservation. The results are presented in Figure 3.

Data Fidelity. Following (Xu et al., 2025), we compute the two kinds of violation rates pre-defined on the Income and California Housing datasets. For the Income dataset, the violation rate refers to the proportion of generated samples exhibiting inconsistencies between the "`educational-num`" and "`education`" features. For the Housing dataset, it refers that of falling outside the geographic boundaries of California. The results are presented in Table 2. Additionally, we visualized the geographic coordinates of the synthetic samples generated from the California Housing dataset, as shown in Figure 4.

5 Results

5.1 Computational Cost and Time Cost

Table 3 presents the training time and the average sampling time per sample for both the baseline methods and our proposed approaches.

5.2 Discussion

SPADA consistently outperforms all the baselines in terms of downstream utility, as shown in Table 1.

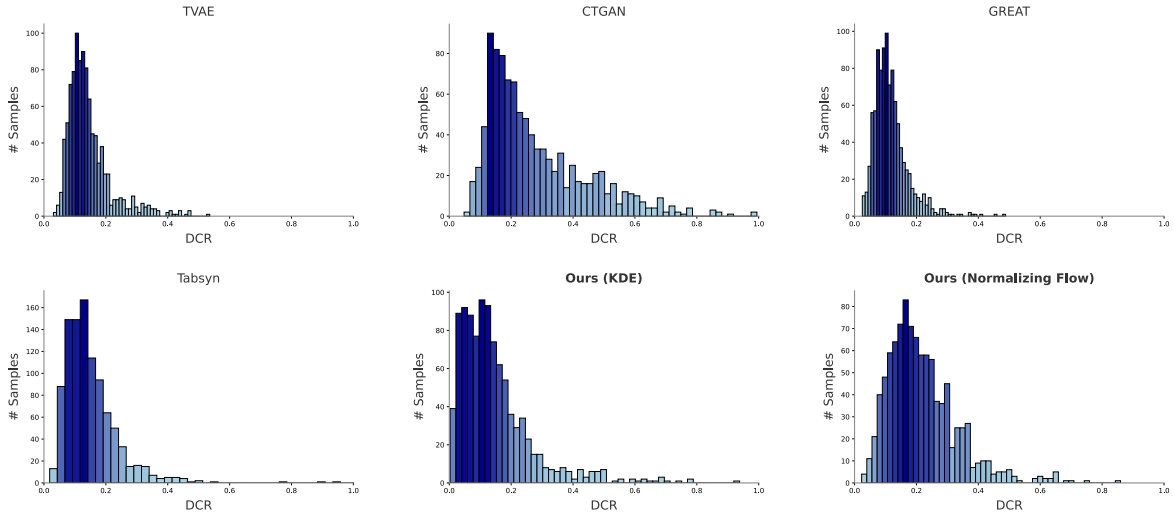


Figure 3: DCR for the California Housing dataset, evaluated with respect to the original training set. A smaller DCR suggests that the model may overfit and copy certain feature values from the original data.

Dataset		TVAE	CTGAN	GReaT	TabSyn	**Ours (w/KDE)**	**Ours (w/NF)**
Income	Training	57 sec	108 sec	6h 10m 34 sec	34 min	**0 sec**	36 sec
	Sampling	**<1 ms**	**<1 ms**	9 sec	5 sec	16 ms	**<1 ms**
HELOC	Training	29 sec	41 sec	1h 47min 38sec	29 min	**0 sec**	45 sec
	Sampling	**<1 ms**	**<1 ms**	45 sec	2 sec	65 ms	2 ms
Iris	Training	<1s	3 sec	17 sec	14 min	**0 sec**	<1s
	Sampling	**<1 ms**	**<1 ms**	4 sec	254 ms	**<1 ms**	**<1 ms**
Housing	Training	33 sec	50 sec	1h 18min 8sec	30 min	**0 sec**	1min 8 sec
	Sampling	**<1 ms**	**<1 ms**	8 sec	2 sec	74 ms	3 ms

Table 3: Average training time and sampling time per instance. The devices we used are shown in Appendix D.1. **Bold** indicates the best performance.

The most striking result is observed on the multi-class Iris dataset, where all three classifiers trained by our NF-based method achieve perfect accuracies. We analyze that the small sample size and low feature dimensionality of the Iris dataset hinder the ability of LLM-based methods such as GReaT to effectively learn the underlying data distribution, thereby resulting in suboptimal performance. Moreover, we observe that the NF-based approach slightly outperforms the KDE-based one on average. This supports our hypothesis mentioned in §3.4: *with small size $\hat{T}$, fuzzy matching can lead to biased density estimation, making it less accurate than trainable neural networks*.

In terms of privacy protection, we observe that the DCR values of synthetic data generated by most baselines are close to zero, as shown in Figure 3. The most severe overlap is observed in our KDE-based method. Since KDE estimates probability densities by retrieving from the original dataset, it inevitably results in a large amount of similar feature values. Similarly, the DCR means for synthetic data produced by TVAE, GReaT, and TabSyn are all below 0.1. In contrast, our NF-based method and CTGAN achieve average DCR values around 0.2, indicating better privacy preservation compared to other methods. Mendelevitch and Lesh (2021) reported that synthetic datasets with higher DCR values exhibit reduced risks of unintended memorization and re-identification. Therefore, increasing the DCR from 0.1 to 0.2 extremely enhances privacy protection by doubling the distinguishability between synthetic and real records, thereby reducing the likelihood of re-identification attacks. This aligns with established research indicating that higher DCR values contribute to stronger privacy safeguards. However, we note that the high DCR of CTGAN may stem from its difficulty in faithfully capturing the underlying distribution of the real data.

Figure 4 illustrates that SPADA faithfully captures the real data distributions. Notably, CT-

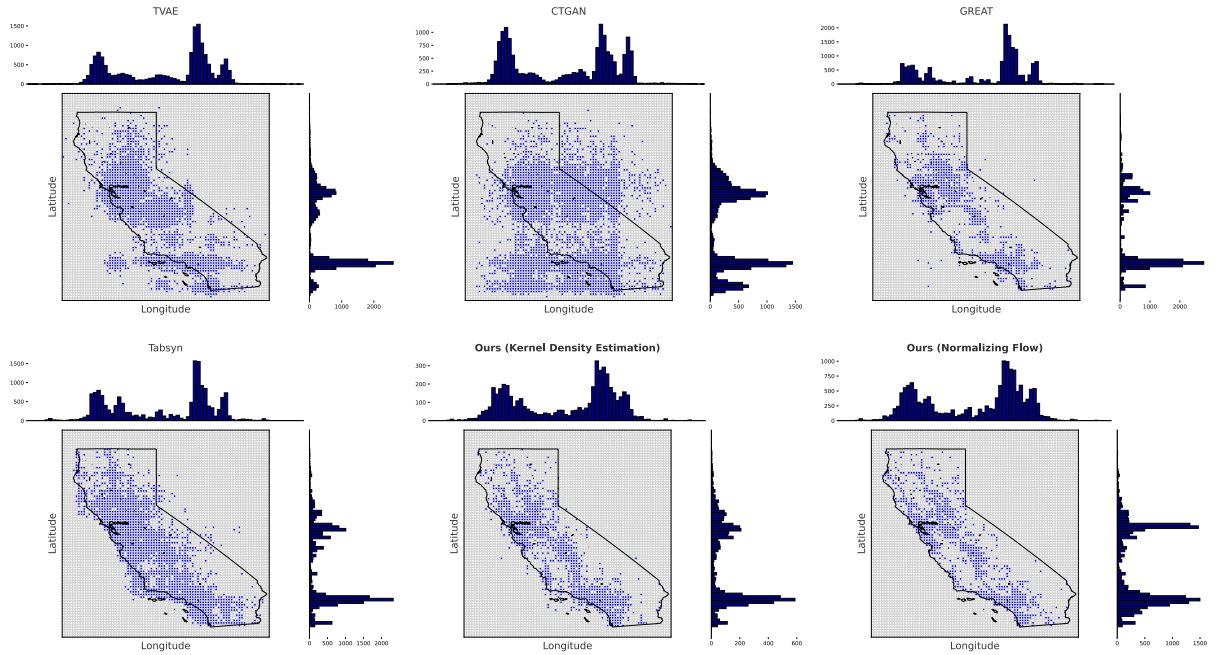


Figure 4: Comparison of the generated samples for the California Housing dataset, which includes characteristic information about various properties in California, USA. Joint histogram plots of the highly correlated variables Latitude and Longitude are shown. The black outline represents the true boundary of the state of California.

GAN exhibits the most severe boundary violations, where the spatial outline of California becomes unrecognizable. Compared to baselines, SPADA produces samples that remain largely within valid geographic boundaries, indicating that they better model the relationship between latitude and longitude. These findings are further supported by Table 2. Our NF-based method achieves a 2% reduction in the violation rate compared to the LLM-based GReaT model on the Housing dataset. This substantial improvement highlights the effectiveness of incorporating sparsity-aware dependency structures during data generation. Unlike fully-connected generative strategies adopted by LLM-based methods, our approach explicitly models and respects the true conditional dependencies among features, thereby ensuring higher logical fidelity in synthesized data.

SPADA demonstrates high efficiency in both training and sampling stages, as shown in Table 3. Compared to parameter-heavy models such as GREAT and TABSYN, our NF-based model reduces training time by over 100× and approximately 30×, respectively. During sampling, the NF-based model synthesizes each sample in under one millisecond, achieving efficiency comparable to TVAE and CTGAN. Remarkably, our model is on average 9,500× faster than the LLM-based GREAT, highlighting its potential for large-scale data augmentation. On the other hand, although our KDE-based method is slightly slower than the NF-based approach, it estimates the need of training and thus advantageous in scenarios with limited computational resources.

6 Conclusion

We proposed SPADA, a novel and lightweight framework for tabular augmentation that disentangles dependency modeling from data generation. By leveraging LLMs to extract sparse feature dependencies and employing lightweight generators, our approach significantly improves the fidelity of synthetic data by 2% while achieving 9,500x speedup in sampling. Extensive experiments across multiple datasets and evaluation metrics demonstrate the effectiveness of SPADA in maintaining downstream utility, enhancing realism, and ensuring privacy preservation. Overall, our findings underscore the potential of leveraging LLM-derived structural priors in conjunction with lightweight generative models for scalable, high-fidelity, and privacy-preserving tabular synthesis. Future work may explore domain-specific adaptations and further integration with interdisciplinary evaluation frameworks to better assess the societal impact of synthetic data technologies.

7 Limitation

While SPADA significantly enhances the effectiveness of LLM-based approaches and greatly reduces the cost of data augmentation, it still has two key limitations:

1) Reliance on the quality of LLM-annotated dependencies. Both of our synthesis strategies rely on the dependency graph produced by an LLM. As such, the accuracy of this graph may theoretically affect downstream model performance. Nevertheless, recent studies have demonstrated the reliability of LLMs in annotation tasks (Gilardi et al., 2023), and in our experiments, the GPT-4o-generated dependency graphs enabled our models to achieve strong performance across multiple datasets. Therefore, although this limitation exists in theory, we have not observed significant evidence that it affects practical performance.

2) Inability to model complex data types. SPADA, following convention, categorizes tabular values into categorical and numerical features. However, for multimodal datasets that include images, videos, or other high-dimensional data types, our current framework is not applicable. Moreover, categorical features are assumed to be finite and text-representable. As a result, SPADA cannot generate open-domain text or novel tokens outside the original dataset; instead, it selects from a fixed set of observed values. We also note that this limitation is shared by existing methods such as TVAE and SynTab, which are similarly restricted to structured tabular data.

Despite these limitations, SPADA remains broadly applicable and offers significant improvements in efficiency and effectiveness over existing approaches. The observed limitations either have minimal empirical impact or are inherent to the general problem setting, rather than specific to our solution. Therefore, we argue that they do not diminish the core contributions of our work.

Acknowledgments

The authors acknowledge the use of ChatGPT solely to refine the text in the final manuscript.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Jonas Adler and Sebastian Lunz. 2018. Banach wasserstein gan. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc.

Xavier Amatriain. 2024. Prompt design and engineering: Introduction and advanced methods. *arXiv preprint arXiv:2401.14423*.

Anthropic. 2024. The claude 3 model family: Opus, sonnet, haiku.

Barry Becker and Ronny Kohavi. 1996. Adult. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C5XW20.

Christopher M. Bishop. 2006. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg.

Christopher M Bishop and Nasser M Nasrabadi. 2006. *Pattern recognition and machine learning*, volume 4. Springer.

Vadim Borisov, Enkelejda Kasneci, and Gjergji Kasneci. 2021. Robust cognitive load detection from wrist-band sensors. *Computers in Human Behavior Reports*, 4:100116.

Vadim Borisov, Tobias Leemann, Kathrin Seßler, Johannes Haug, Martin Pawelczyk, and Gjergji Kasneci. 2022. Deep neural networks and tabular data: A survey. *IEEE transactions on neural networks and learning systems*.

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

Anna L. Buczak and Erhan Guven. 2016. A survey of data mining and machine learning methods for cyber security intrusion detection. *IEEE Communications Surveys & Tutorials*, 18(2):1153–1176.

Nilka Rios Burrows. 2017. Incidence of end-stage renal disease attributed to diabetes among persons with diagnosed diabetes—united states and puerto rico, 2000–2014. *MMWR. Morbidity and mortality weekly report*, 66.

Corinna Cortes and Vladimir Vapnik. 1995. Support-vector networks. *Machine learning*, 20:273–297.

Ronald A Fisher. 1936. The use of multiple measurements in taxonomic problems. *Annals of eugenics*, 7(2):179–188.

Fabrizio Gilardi, Meysam Alizadeh, and Maël Kubli. 2023. Chatgpt outperforms crowd workers for text-annotation tasks. *Proceedings of the National Academy of Sciences*, 120(30):e2305016120.

Leo Grinsztajn, Edouard Oyallon, and Gael Varoquaux. 2022. Why do tree-based models still outperform deep learning on typical tabular data? In *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Noah Hollmann, Samuel Müller, Lennart Purucker, Arjun Krishnakumar, Max Körfer, Shi Bin Hoo, Robin Tibor Schirrmeister, and Frank Hutter. 2025. Accurate predictions on small data with a tabular foundation model. *Nature*.

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

Sanket Kamthe, Samuel A. Assefa, and Marc Peter Deisenroth. 2021. Copula flows for synthetic data generation. *ArXiv*, abs/2101.00598.

Jerome Kelleher, Rob W Ness, and Daniel L Halligan. 2013. Processing genome scale tabular data with wormtable. *BMC bioinformatics*, 14:1–5.

Durk P Kingma and Prafulla Dhariwal. 2018. Glow: Generative flow with invertible 1x1 convolutions. *Advances in neural information processing systems*, 31.

Akim Kotelnikov, Dmitry Baranchuk, Ivan Rubachev, and Artem Babenko. 2023. Tabddpm: modelling tabular data with diffusion models. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org.

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

Yunbo Long, Liming Xu, and Alexandra Brintrup. 2025. Llm-tabflow: Synthetic tabular data generation with inter-column logical relationship preservation. *arXiv preprint arXiv:2503.02161*.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Ofer Mendelevitch and Michael D. Lesh. 2021. Fidelity and privacy of synthetic medical data.

Riddhi More and Jeremy S Bradbury. 2025. Assessing data augmentation-induced bias in training and testing of machine learning models. *arXiv preprint arXiv:2502.01825*.

George Nemhauser and Laurence Wolsey. 1988. *The Scope of Integer and Combinatorial Optimization*, chapter I.1. John Wiley & Sons, Ltd.

Dang Nguyen, Sunil Gupta, Kien Do, Thin Nguyen, and Svetha Venkatesh. 2024. Generating realistic tabular data with large language models. *arXiv preprint arXiv:2410.21717*.

Mohammad Norouzi, Ali Punjani, and David J Fleet. 2012. Fast search in hamming space with multi-index hashing. In *2012 IEEE conference on computer vision and pattern recognition*, pages 3108–3115. IEEE.

Cameron Nugent. 2018. California housing prices. `https://www.kaggle.com/datasets/camnugent/california-housing-prices`.

Averkiy Oliabev. 2022. Home equity line of credit (heloc) dataset. `https://www.kaggle.com/datasets/averkiyoliabev/home-equity-line-of-creditheloc`.

Stepphen M. Omohundro. 1989. Five balltree construction algorithms. Technical Report TR-89-063, International Computer Science Institute.

Dae-Young Park and In-Young Ko. 2024. An empirical study of utility and disclosure risk for tabular data synthesis models: In-depth analysis and interesting findings. In *2024 IEEE International Conference on Big Data and Smart Computing (BigComp)*, pages 67–74.

Neha Patki, Roy Wedge, and Kalyan Veeramachaneni. 2016. The synthetic data vault. In *IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, pages 399–410.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. 2011. Scikit-learn: Machine learning in python. *J. Mach. Learn. Res.*, 12(null):2825–2830.

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

Weijieying Ren, Tianxiang Zhao, Yuqing Huang, and Vasant Honavar. 2025. Deep learning within tabular data: Foundations, challenges, advances and future directions.

Danilo Rezende and Shakir Mohamed. 2015a. Variational inference with normalizing flows. In *International conference on machine learning*, pages 1530–1538. PMLR.

Danilo Rezende and Shakir Mohamed. 2015b. Variational inference with normalizing flows. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 1530–1538, Lille, France. PMLR.

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

Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. 2008. The graph neural network model. *IEEE transactions on neural networks*, 20(1):61–80.

David W Scott. 2015. *Multivariate density estimation: theory, practice, and visualization*. John Wiley & Sons.

Kshitij Sharma, Yogesh K. Dwivedi, and Bhimaraya Metri. 2024. Incorporating causality in energy consumption forecasting using deep neural networks. *Annals of Operations Research*, 339(1):537–572.

Noam Shazeer. 2020. Glu variants improve transformer. *arXiv preprint arXiv:2002.05202*.

Bernard W Silverman. 2018. *Density estimation for statistics and data analysis*. Routledge.

Mihaela C. Stoian and Eleonora Giunchiglia. 2025. Beyond the convexity assumption: Realistic tabular data generation under quantifier-free real linear constraints. In *The Thirteenth International Conference on Learning Representations*.

Mihaela Cătălina Stoian, Eleonora Giunchiglia, and Thomas Lukasiewicz. 2025. A survey on tabular data generation: Utility, alignment, fidelity, privacy, and beyond.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, Katie Millican, et al. 2023. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*.

Dennis Ulmer, Lotta Meijerink, and Giovanni Cinà. 2020. Trust issues: Uncertainty estimation does not enable reliable ood detection on medical tabular data. In *Machine Learning for Health*, pages 341–354. PMLR.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 6000–6010, Red Hook, NY, USA. Curran Associates Inc.

Dennis Wagner, Dominik Heider, and Georges Hattab. 2021. Mushroom data creation, curation, and simulation to support classification tasks. *Scientific Reports*, 11.

Siqi Wang, Jiyuan Liu, Guang Yu, Xinwang Liu, Sihang Zhou, En Zhu, Yuexiang Yang, Jianping Yin, and Wenjing Yang. 2024. Multiview deep anomaly detection: A systematic exploration. *IEEE Transactions on Neural Networks and Learning Systems*, 35(2):1651–1665.

Yuchen Xia, Jiho Kim, Yuhan Chen, Haojie Ye, Souvik Kundu, Cong Callie Hao, and Nishil Talati. 2024. Understanding the performance and estimating the cost of llm fine-tuning. In *2024 IEEE International Symposium on Workload Characterization (IISWC)*, pages 210–223. IEEE.

Haoyin Xu, Kaleab A. Kinfu, Will LeVine, Sambit Panda, Jayanta Dey, Michael Ainsworth, Yu-Chung Peng, Madi Kusmanov, Florian Engert, Christopher M. White, Joshua T. Vogelstein, and Carey E. Priebe. 2021. When are deep networks really better than decision forests at small sample sizes, and how?

Lei Xu, Maria Skoularidou, Alfredo Cuesta-Infante, and Kalyan Veeramachaneni. 2019. *Modeling tabular data using conditional GAN*. Curran Associates Inc., Red Hook, NY, USA.

Shengzhe Xu, Cho-Ting Lee, Mandar Sharma, Raquib Bin Yousuf, Nikhil Muralidhar, and Naren Ramakrishnan. 2025. Why llms are bad at synthetic table generation (and what to do about it).

Shuo Yang, Chenchen Yuan, Yao Rong, Felix Steinbauer, and Gjergji Kasneci. 2024. P-TA: Using proximal policy optimization to enhance tabular data augmentation via large language models. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 248–264, Bangkok, Thailand. Association for Computational Linguistics.

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

Yishuo Zhang, Nayyar A Zaidi, Jiahui Zhou, and Gang Li. 2021. Ganblr: a tabular data generation model. In *2021 IEEE International Conference on Data Mining (ICDM)*, pages 181–190. IEEE.

Xun Zheng, Bryon Aragam, Pradeep K Ravikumar, and Eric P Xing. 2018. Dags with no tears: Continuous optimization for structure learning. *Advances in neural information processing systems*, 31.

Kaixiong Zhou, Zirui Liu, Rui Chen, Li Li, Soo-Hyun Choi, and Xia Hu. 2022. Table2graph: Transforming tabular data to unified weighted graph. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*, pages 2420–2426. International Joint Conferences on Artificial Intelligence Organization. Main Track.

A Theory Analysis

A.1 Asymptotic Consistency of the KDE with Gaussian Kernel

Problem Setup Given a target feature $f_j \in \mathcal{F}$ and its parent features $\hat{F}_{f_j} \subseteq \mathcal{F}$. For the dataset $T = \{\mathbf{t}_i\}_{i=1}^n$, we first retrieve a subset $\hat{T}$ via fuzzy matching on $\hat{F}_{f_j}$:

$$\hat{T} = \left\{ \mathbf{t}_i \in T \,\middle|\, \text{Dist}\left(v_{i,\hat{F}_{f_j}}, v^*_{\hat{F}_{f_j}}\right) \leq \epsilon \right\}, \quad (14)$$

where $v^*_{\hat{F}_{f_j}}$ are the parent feature values generated, and $\text{Dist}(\cdot)$ combines Hamming distance for categorical features and L1 distance for numerical features, as we mentioned in §3.3.1.

The conditional density of f_j is then estimated via Gaussian KDE on $\hat{T}$:

$$\hat{f}(v_j|\{v_k\}) = \frac{1}{|\hat{T}|b} \sum_{\mathbf{t}_i \in \hat{T}} \frac{\exp\left(-\frac{(v_j - v_{ij})^2}{2b^2}\right)}{\sqrt{2\pi}}, \quad (15)$$

where $b > 0$ is the bandwidth.

Key Assumptions

A1. The real conditional density $f(v_j|\{v_k\})$ is twice continuously differentiable.

A2. The bandwidth $b \to 0$ and $|\hat{T}|b \to \infty$ as $n \to \infty$.

A3. The fuzzy matching tolerance $\epsilon \to 0$ such that $\hat{T}$ asymptotically covers the real conditional support.

Bias-Variance Decomposition The pointwise MSE is:

$$\begin{aligned} \text{MSE} &= \mathbb{E}\left[\left(\hat{f}(v_j|\{v_k\}) - f(v_j|\{v_k\})\right)^2\right] \quad (16) \\ &= \text{Bias}^2 + \text{Var}. \end{aligned}$$

Bias Analysis Using Taylor expansion (Bishop, 2006), we have:

$$\begin{aligned} \text{Bias} &= \mathbb{E}[\hat{f}(v_j|\{v_k\})] - f(v_j|\{v_k\}) \quad (17) \\ &= \frac{b^2}{2}\frac{\partial^2 f}{\partial v_j^2}(v_j|\{v_k\}) \int u^2 K(u) du + o(b^2) \quad (18) \\ &= \frac{b^2}{2}\frac{\partial^2 f}{\partial v_j^2}(v_j|\{v_k\}) + o(b^2). \quad (19) \end{aligned}$$

Variance Analysis

$$\begin{aligned} \text{Var} &= \mathbb{E}\left[\hat{f}(v_j|\{v_k\})^2\right] - \left(\mathbb{E}[\hat{f}(v_j|\{v_k\})]\right)^2 \quad (20) \\ &= \frac{1}{|\hat{T}|b}\left(\int K(u)^2 du\right) f(v_j|\{v_k\}) \quad (21) \\ &+ o\left(\frac{1}{|\hat{T}|b}\right) \quad (22) \\ &= \frac{1}{|\hat{T}|b\sqrt{4\pi}} f(v_j|\{v_k\}) + o\left(\frac{1}{|\hat{T}|b}\right). \quad (23) \end{aligned}$$

Consistency Proof Under assumptions A1-A3, the Gaussian KDE estimator is asymptotically consistent:

$$\lim_{n\to\infty} \mathbb{E}\left[\left|\hat{f}(v_j|\{v_k\}) - f(v_j|\{v_k\})\right|^2\right] = 0. \quad (24)$$

1. **Bias Convergence:**

$$|\text{Bias}| \leq C_1 b^2 + o(b^2) \to 0 \quad \text{as } b \to 0. \quad (25)$$

2. **Variance Convergence:**

$$\text{Var} \leq \frac{C_2}{|\hat{T}|b} + o\left(\frac{1}{|\hat{T}|b}\right) \to 0 \quad \text{if } |\hat{T}|b \to \infty. \quad (26)$$

3. **MSE Dominance:** Choosing $b \sim |\hat{T}|^{-1/5}$ yields:

$$\text{MSE} = O\left(|\hat{T}|^{-4/5}\right) \to 0 \quad \text{as } |\hat{T}| \to \infty. \quad (27)$$

A.2 Conditional Likelihood Lower Bound Analysis for Conditional Normalizing Flows

Problem Setup We model the conditional distribution $p(v_j|\{v_k : f_k \in \hat{F}_{f_j}\})$ via Conditional NFs. Let $z \sim \mathcal{N}(0, I)$ be the latent variable, and $f_\theta : \mathbb{R} \times \mathbb{R}^{|\hat{F}_{f_j}|} \to \mathbb{R}$ be the invertible transformation conditioned on parent feature values.

Change of Variables Theorem Following Eq. (12), the log-likelihood is given by:

$$\log p(v_j|\{v_k\}) = \log p_Z(z) + \log \left| \det \frac{\partial f_\theta^{-1}}{\partial v_j} \right| \quad (28)$$

where $z = f_\theta^{-1}(v_j|\{v_k\})$.

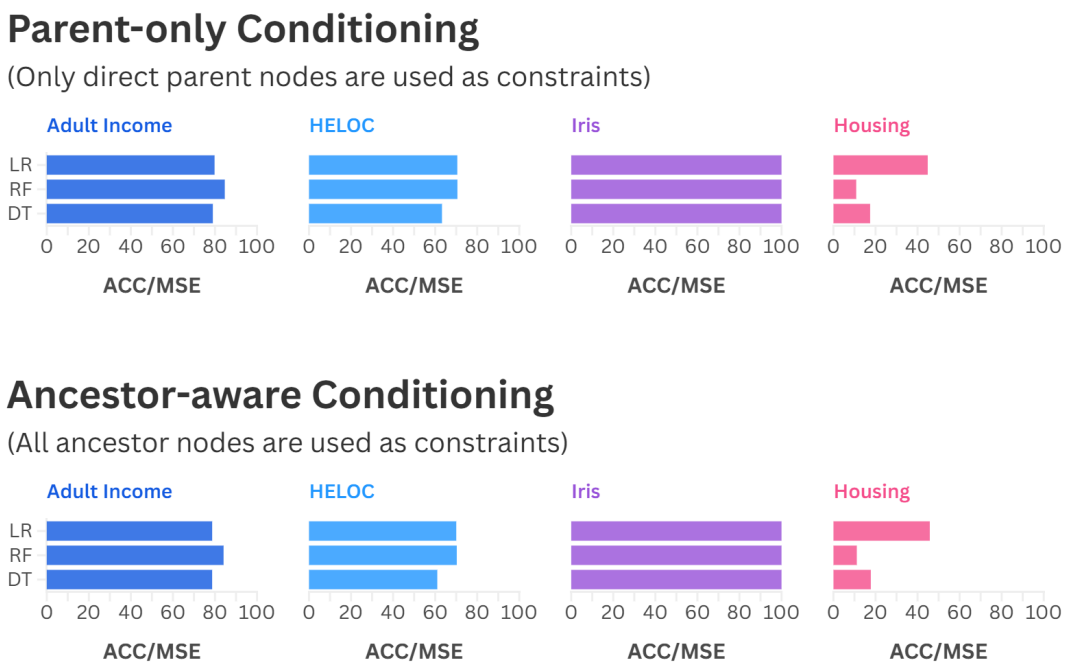


Figure 5: Comparison of synthesis performance using parent-only conditioning vs. ancestor-aware conditioning strategies. To facilitate comparison, we multiplied the MSE values by a factor of 10.

Lipschitz-Constrained Transformation To ensure a stable gradient propagation, we constrain each layer $f_\theta^{(i)}$ in our flow to be L-Lipschitz continuous (Adler and Lunz, 2018):

$$\begin{aligned}\|f_\theta^{(i)}(z^{(i-1)}|\{v_k\}) - f_\theta^{(i)}(z'^{(i-1)}|\{v_k\})\| \\ \leq L\|z^{(i-1)} - z'^{(i-1)}\|\end{aligned} \tag{29}$$

Lower Bound Derivation 1. **Layer-wise Jacobian Bound:** For each layer, the Lipschitz continuity (Kingma and Dhariwal, 2018) implies:

$$\left|\det \frac{\partial f_\theta^{(i)}}{\partial z^{(i-1)}}\right| \geq L_i^{-d} \tag{30}$$

where d is the dimension of $z^{(i-1)}$.

2. **Recursive Likelihood Decomposition:**

$$\log p(v_j|\{v_k\}) = \log p_Z(z_0) + \tag{31}$$

$$\sum_{i=1}^{K} \log \left|\det \frac{\partial f_\theta^{(i)-1}}{\partial z^{(i)}}\right| \tag{32}$$

$$\geq \log p_Z(z_0) - \sum_{i=1}^{K} d \log L_i \tag{33}$$

3. **Equality Condition:** Here, the bound becomes tight when each layer achieves exact Lipschitz constant $L_i = 1$.

Stability Analysis The gradient of the loss $\mathcal{L}(\theta) = -\log p(v_j|\{v_k\})$ satisfies:

$$\|\nabla_\theta \mathcal{L}\| \leq \sqrt{K} \cdot \max_{1\leq i\leq K} \left(\frac{\|J_i\|_F}{L_i}\right) \tag{34}$$

where J_i is the Jacobian of layer $f_\theta^{(i)}$.

Using the Lipschitz property and the chain rule:

$$\begin{aligned}\|\nabla_\theta \mathcal{L}\| &\leq \sum_{i=1}^{K} \left\|\frac{\partial \mathcal{L}}{\partial f_\theta^{(i)}}\right\| \cdot \|\nabla_\theta f_\theta^{(i)}\| \\ &\leq \sqrt{K} \cdot \max_i \left(\frac{\|J_i\|_F}{L_i}\right)\end{aligned} \tag{35}$$

Implementation Consistency

- **SwiGLU Parameterization:** The SwiGLU activation $\sigma(xW) \odot xV$ naturally satisfies L-Lipschitz continuity with $L = \|W\|\|V\|$

A.3 Time Complexity Optimization by Using BallTrees

Compared to the original brute-force search, the BallTree significantly improves efficiency. Given a dependency set $\hat{F}_{f_j}$ of size $d = |\hat{F}_{f_j}|$, the time complexity of brute-force search is:

$$\mathcal{O}(N \cdot d). \tag{36}$$

	TabSyn		Gemini 2.0 Flash		Deepseek-R1		Claude 3.7 Sonnet		GPT-4o	
	ACC	F1	ACC	F1	ACC	F1	ACC	F1	ACC	F1
DT (↑)	77.58%	0.70	78.37%	0.71	77.14%	0.69	**80.13%**	**0.73**	78.95	0.71
RF (↑)	83.67%	0.77	84.99%	0.77	85.05%	0.77	**85.13%**	**0.78**	84.14	0.75
LR (↑)	80.21%	**0.70**	80.05%	0.65	**80.43%**	0.65	80.31%	0.67	78.61	0.56

Table 4: Performance of classifiers trained on synthetic data for income-level classificaton. **Bold** indicates the best performance, and underline indicates the second-best. "ACC" stands for accuracy.

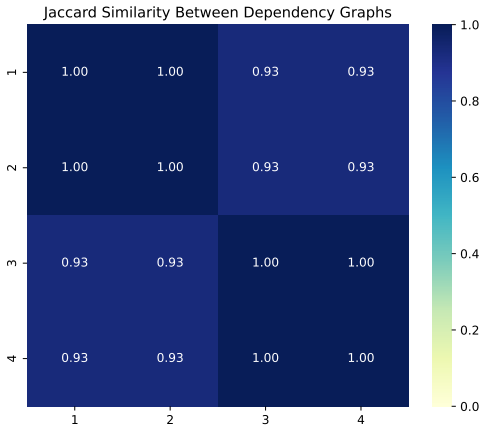


Figure 6: Heatmap of Jaccard similarities between the dependency graphs constructed from four annotation runs of the GPT-4o model. The high similarity values indicate strong consistency.

In contrast, BallTrees reduce the average-case complexity of approximate nearest neighbor search to:

$$\mathcal{O}(\log N + d), \tag{37}$$

under the assumption that the number of dependencies d is much smaller than dataset size N.

B Ablation Study

To further demonstrate the effectiveness of our proposed methods, we conduct three ablation studies to answer the following questions:

1. Are the LLM-based annotations robust?
2. Does the bandwidth significantly affect the performance of KDE-based synthesis?
3. Is the dependency relation transitive in the generated dependency graph?

Cross-LLM Annotation. As mentioned in §7, the accuracy of annotations produced by LLMs could theoretically affect the effectiveness of our approach. To assess this, we evaluated the robustness of the NL-based method by testing the downstream performance of synthetic data generated using annotations derived from different LLMs, while keeping the training and sampling settings exactly the same. We selected the Adult dataset—on which all baselines perform well according to Table 1 for this experiment. We employed Deepseek-R1 (Guo et al., 2025), Gemini 2.0 Flash (Team et al., 2023), and Claude 3.7 Sonnet (Anthropic, 2024), and compared them with the best-performing baseline, TabSyn. The results are presented in Table 4. SPADA remains effective regardless of which LLM annotation is used. Notably, the best performance was achieved using annotations from Claude 3.7 Sonnet, which outperformed other baselines across nearly all evaluation metrics. This suggests that for a given dataset, there may not be a single unique set of complete dependency relations, further confirming the robustness of SPADA to variations in LLM-generated annotations.

Repeated Annotation. We conducted four repeated annotations using GPT-4o on the 16 mixed-type features in the Adult Income dataset. We then computed the pairwise Jaccard similarity between the resulting dependency graphs, as shown in Figure 6. We observe a high degree of consistency across multiple annotations, with the first and second annotations, also the third and fourth, producing identical dependency graphs. This highlights the reliability of using LLMs for dependency annotation.

Scott's Rule vs Silverman's Rule. For the KDE-based method, the bandwidth h in Eq. (10) is selected using two widely adopted empirical rules: **Scott's rule** (Scott, 2015) and **Silverman's rule** (Silverman, 2018):

$$h_{\text{Scott}} = n^{-1/(d+4)}, \tag{38}$$

$$h_{\text{Silverman}} = \left(\frac{4}{d+2}\right)^{1/(d+4)} n^{-1/(d+4)}, \tag{39}$$

where n is the number of samples and d is the dimensionality of the feature space.

The comparison results are shown in Figure 7. We observe that the downstream performance is

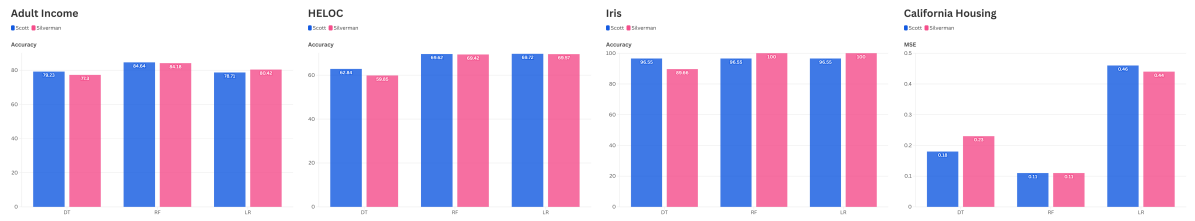


Figure 7: Impact of Bandwidth Selection on KDE-based Synthesis: Scott's rule vs. Silverman's rule.

Dataset	TVAE	CTGAN	GReaT	TabSyn	**Ours (w/KDE)**	**Ours (w/NF)**
Income (↓)	77.62 ± 0.63%	63.88 ± 0.49%	100%	**52.74 ± 0.58%**	75.96 ± 0.65%	67.05 ± 0.70%
HELOC (↓)	74.33 ± 1.20%	70.49 ± 6.81%	67.91 ± 1.13%	**52.59 ± 1.22%**	58.30 ± 0.72%	66.38 ± 0.83%
Iris (↓)	83.33 ± 7.97%	82.92 ± 10.72%	68.33 ± 8.06%	**52.08 ± 3.17%**	65.42 ± 7.03%	65.52 ± 7.68%
Housing (↓)	59.65 ± 1.60%	62.70 ± 1.07%	58.14 ± 1.35%	**50.48 ± 0.48%**	76.75 ± 1.15%	68.33 ± 0.95%
Mean (↓)	73.73	69.99	73.59	**51.97**	69.05	68.92

Table 5: Discriminator measure with a 5-fold cross-validation. Lower accuracy values indicate that the discriminator struggles to distinguish synthetic records from real data.

similar regardless of the choice of bandwidth rule. This indicates that the effectiveness of KDE-based synthesis is not sensitive to hyperparameter selection, further supporting the robustness of SPADA.

Parent-only Conditioning vs Ancestor-aware Conditioning. For the NF-based method, we compare two strategies for conditioning on feature dependencies: one that conditions only on direct parents, and another that conditions on all ancestors. As shown in Figure 5, both approaches yield nearly identical downstream performance. This suggests that the constraints are not transitive, validating the precision of our dependency annotation. It further confirms the reliability of using LLMs to construct accurate dependency graphs.

C More Experiment

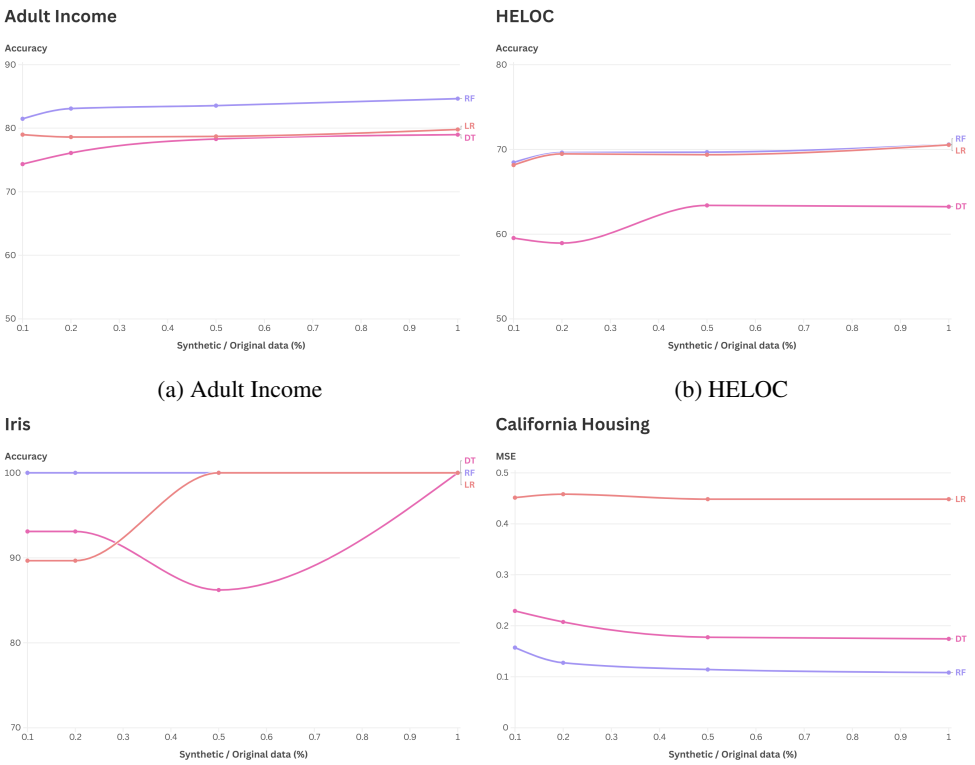
Realism. We trained a support vector machine (Cortes and Vapnik, 1995) using 5-fold cross-validation to distinguish between the original dataset T and the synthetic dataset T'. The accuracy serves as the discriminator measure. High-quality synthetic data should be difficult for the discriminator to distinguish from real data. The results are shown in Table 5.

Based on Table 5, we observe that SPADA produces synthetic data that substantially confuses classifiers trained on real data, indicating a high degree of realism. Although our approach underperforms TabSyn in terms of the average discriminator accuracy, it still demonstrates an advantage over the other baselines. Theoretically, for perfectly realistic data, the discriminator accuracy should approach 50%. Among all methods, only TabSyn, which is based on diffusion models, achieves this ideal. We attribute this to TabSyn's use of a score function in the latent space, which guides the generation process toward high-probability regions and thus produces samples closely aligned with the original data distribution. However, this close resemblance may also result in considerable feature overlap between real and synthetic samples, potentially raising privacy concerns.

Synthetic Data Size. To demonstrate that the classifier/regressors used in our evaluation have been sufficiently trained on our augmented data, we assessed the impact of varying augmentation sizes on model performance.

As shown in Figure 8, we observe that the performance of nearly all classifier/regressors converges to their respective optima when the size of synthetic data reaches approximately 0.5 times that of the original dataset. This finding supports that all evaluation models in our experiments have been adequately trained, further validating the soundness and reproducibility of our experimental setup.

Visualization. To compare the distributional differences between our synthetic data and the original data, we visualize the density distributions of the four numerical features used to determine the iris species in the Iris dataset—namely, the lengths and widths of the Sepal and Petal. As shown in Figure 9, we observe that both of our methods are able to accurately capture the distributional patterns of the original data.



(a) Adult Income

(b) HELOC

(c) Iris

(d) Housing

Figure 8: Model performance with increasing synthetic data (ratio to the original dataset size).

Dataset	Domain	#Samples	#Features	Task	#Classes
Income (Becker and Kohavi, 1996)	Social	48,842	15	Classification	2
HELOC (Oliabev, 2022)	Finance	10,459	24	Classification	2
Iris (Fisher, 1936)	Biology	150	5	Classification	3
Housing (Nugent, 2018)	Real Estate	20,640	10	Regression	N/A
CDC (Burrows, 2017)	Healthcare	253,680	20	Classification	3
Mushroom (Wagner et al., 2021)	Biology	61,068	19	Classification	2

Table 6: The statistics of the datasets employed in our experiments.

D Reproducibility

D.1 Hardware Environment

The experimental hardware environment we used is shown in Table 7.

Memory	1012G
CPU	AMD EPYC 7763 2.45G Hz
GPU	4 x NVIDIA A100 80G
Operating System	Ubuntu 20.04.6 LTS

Table 7: Experimental hardware environment.

D.2 Training Details

In our experiments, the conditional batch normalization module is implemented using a single linear layer, a single SwiGLU activation layer, and a layer normalization module. The linear layer contains 128 neurons, with a dropout rate of 0.1. During training, we employ batch gradient descent with the AdamW optimizer (Loshchilov and Hutter, 2017), and the learning rate is set to 5×10^{-4}. The initial input distribution for the normalizing flow NF is a standard normal distribution, and the kernel density estimation is based on a standard Gaussian kernel. The statistics of the datasets used is shown in Table 6.

D.3 Hyperparameter Search

We adopt a temperature-based sampling strategy to adjust the sharpness of the predicted distribution during synthesis. Specifically, given the estimated density function $p(v)$, we sample from the tem-

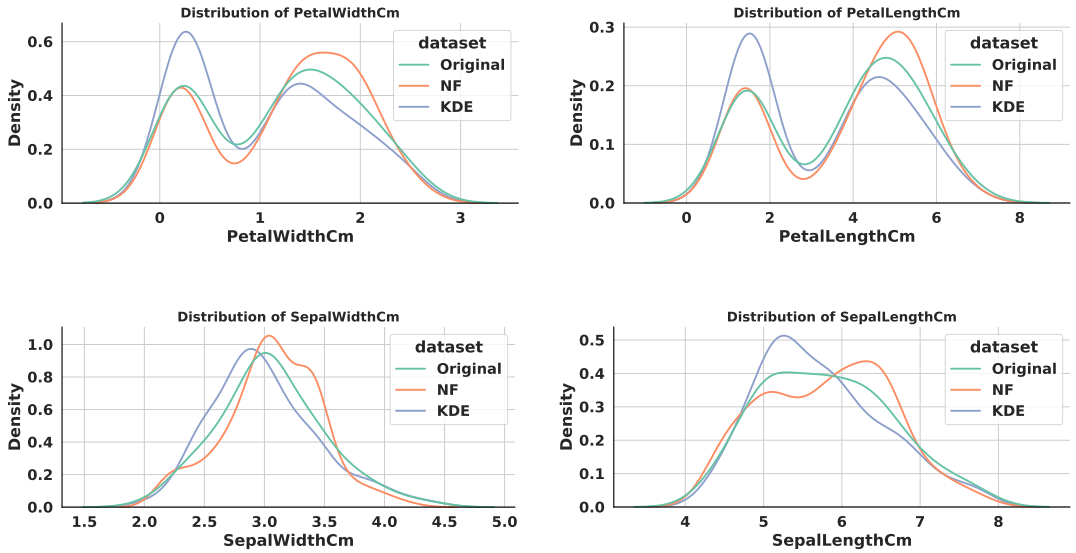


Figure 9: A visualization of the density distributions of Sepal and Petal lengths and widths on the Iris dataset, comparing the original and synthetic data.

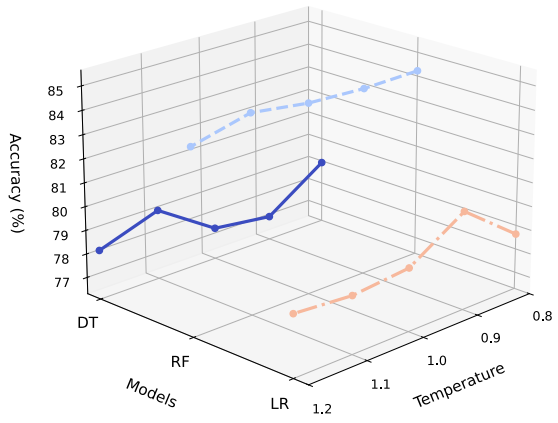


Figure 10: Grid Search for sampling temperature with our KDE-based method on the Adult Income dataset. The numbers stand for accuracy on the downsteam task.

pered distribution defined as:

$$p_\tau(v) = \frac{p(v)^{1/\tau}}{\int p(u)^{1/\tau}\, du}, \tag{40}$$

where $\tau > 0$ denotes the temperature parameter. A lower τ results in a more peaked distribution (i.e., less diverse samples), while a higher T smooths the distribution and encourages greater diversity.

In Figure 10, we take the *Adult Income* dataset as an example to illustrate how we tune the temperature hyperparameter via grid search. We report the downstream utility of the KDE-based method under different temperature settings, demonstrating the impact of temperature on sample quality and task performance.

E Scalability Analysis

The scalability of tabular augmentation methods may become crucial when dealing with high-dimensional datasets. In domains such as genomics (Kelleher et al., 2013), the feature counts are extremely high in theory. Our framework inherently addresses this challenge through two key designs: **sparse dependency modeling** and **modular generation pipelines**.

Traditional LLM-based methods incur quadratic memory overhead in self-attention layers, making them impractical for datasets with excessive features. In contrast, our sparse dependency extraction reduces pairwise interactions to $O(M \cdot k)$, where $k \ll M$ is the average number of parent nodes per feature (empirically $k \leq 6$ across our four datasets). Topological traversal ensures linear time complexity $O(M)$ during sampling.

E.1 Memory and Storage Optimization

We recommend the following optimizations:

1. **Sparse Graph Representation**: Use adjacency lists for dependency edges, reducing memory from $O(M^2)$ to $O(M \cdot k)$.

2. **Parallelized NF Training**: Deploy feature-specific normalizing flows on GPUs, sharing parameters for categorical features with similar dependencies.

Dataset		Original		TVAE		CTGAN		**Ours (w/NF)**	
		ACC	F1	ACC	F1	ACC	F1	ACC	F1
CDC (↑)	DT	73.65%	0.40	**71.96%**	**0.46**	71.51%	0.36	71.19%	0.36
	RF	83.15%	0.38	82.44%	**0.43**	82.67%	0.35	**82.96%**	0.31
	LR	82.70%	0.40	81.62%	**0.48**	81.98%	0.36	**82.77%**	0.31
Mushroom (↑)	DT	52.49%	0.47	51.30%	**0.51**	51.38%	0.50	**51.79%**	0.45
	RF	59.22%	0.44	63.96%	**0.58**	60.57%	0.53	**64.06%**	0.54
	LR	55.97%	0.36	55.96%	0.35	55.88%	0.35	**55.97%**	**0.36**

Table 8: Performance of classifiers trained on synthetic data for downstream tasks.

3. **On-Demand KDE Sampling**: Cache Ball-Tree indices in distributed key-value stores for large datasets.

E.2 Practical Limitations and Mitigations

While theoretically scalable, two bottlenecks emerge in practice:

- **LLM Annotation Overhead**: Prompting LLMs for a large number of features incurs prohibitive costs. Future work may use lightweight predictors, e.g., GNN (Scarselli et al., 2008), to bootstrap the process.
- **Feature Interaction Sparsity**: Assumes local dependencies dominate, which holds empirically in real-world tabular data (Liu et al., 2023). For systems with global interactions, hybrid architectures combining SPADA with low-rank attention layers (Hu et al., 2022) could be explored.

F Experimental Evaluations on Real-world and Large-scale Datasets

Based on the scalability discussion above, we conducted experiments on two additional large-scale datasets and reported the downstream utility under synthetic data, as shown in Table 8. Due to the large scale of the datasets, it was impractical to compare with LLM- and diffusion-based methods. Therefore, we compared against TVAE and CTGAN.

CDC Diabetes Health Indicators. The *CDC* dataset (Burrows, 2017) comprises over 250,000 samples, containing healthcare statistics and lifestyle survey responses, along with diabetes diagnoses. It includes 35 features covering demographics and laboratory test results. The task is to predict whether a patient is healthy, Diabetes mellitus Typ 1, or Diabetes mellitus Typ 2.

Mushroom. The *Mushroom* dataset (Wagner et al., 2021) comprises over 61,000 samples, containing biological features of mushrooms for binary classification into edible and poisonous.

The experiments conducted on larger-scale datasets further support the conclusions drawn from Table 1, demonstrating the scalability of SPADA.

G Prompt Used

The prompt template we used is shown in Table 9.

LLM Prompt

Given a tabular dataset with the following description:
"`{description}`"

The dataset holds the following features, represented in numbers or text strings:
`{numerical_feature + categorical_feature}`

Please list the constraints for each feature based on the others. Return the results in the following format: for each feature, first output the feature name followed by a colon, and then a set of constraints represented by square brackets. The '->' symbol indicates that the former is the cause and the latter is the effect. Different constraints should be separated by commas.

Here is an example:
`Feature A: [Feature B->Feature A, Feature C->Feature A]`
This means that both Feature B and Feature C determine the range of Feature A.

Please leave it blank if there is no relation between a feature and others.

Table 9: Prompt used for dependency annotation.

H Dependencies Extracted

The follows are the complete dependencies extracted from the datasets we used, as shown in Table 10, 11, 12, 13, 14 and 15.

Feature	Dependencies
Sepal-Length-Cm	—
Sepal-Width-Cm	—
Petal-Length-Cm	—
Petal-Width-Cm	—
Species	SepalLengthCm, SepalWidthCm, PetalLengthCm, PetalWidthCm → Species

Table 10: Extracted Feature Dependencies for `Iris` Dataset.

Feature	Dependencies
age	—
fnlwgt	—
educational-num	education → educational-num
capital-gain	occupation → capital-gain
capital-loss	occupation → capital-loss
hours-per-week	occupation → hours-per-week
workclass	occupation, education → workclass
education	educational-num → education
marital-status	age → marital-status
occupation	education, workclass → occupation
relationship	marital-status, gender → relationship
race	—
gender	—
native-country	—
income	education, workclass, occupation, capital-gain, capital-loss, hours-per-week → income

Table 11: Extracted Feature Dependencies for `Income` Dataset.

Feature	Dependencies
cap-diameter	—
stem-height	—
stem-width	—
class	cap-shape, cap-surface, cap-color, does-bruise-or-bleed, gill-attachment, gill-spacing, gill-color, stem-root, stem-surface, stem-color, veil-type, veil-color, has-ring, ring-type, spore-print-color, habitat, season → class
cap-shape	—
cap-surface	—
cap-color	—
does-bruise-or-bleed	—
gill-attachment	—
gill-spacing	—
gill-color	—
stem-root	—
stem-surface	—
stem-color	—
veil-type	—
veil-color	—
has-ring	—
ring-type	—
spore-print-color	—
habitat	—
season	—

Table 12: Extracted Feature Dependencies for `Mushroom` Dataset.

Feature	Dependencies
longitude	—
latitude	longitude → latitude
housing_median_age	longitude, latitude, ocean_proximity → housing_median_age
total_rooms	population, households, median_income → total_rooms
total_bedrooms	total_rooms, population, households → total_bedrooms
population	households, total_rooms, total_bedrooms → population
households	population, total_rooms, total_bedrooms → households
median_income	longitude, latitude, ocean_proximity → median_income
median_house_value	median_income, housing_median_age, ocean_proximity, latitude, longitude → median_house_value
ocean_proximity	longitude, latitude → ocean_proximity

Table 13: Extracted Feature Dependencies for `Housing` Dataset.

Feature	Dependencies
ExternalRiskEstimate	NumSatisfactoryTrades, PercentTradesNeverDelq, NumTotalTrades, NumTrades90Ever2DerogPubRec → ExternalRiskEstimate
MSinceOldestTradeOpen	—
MSince-Most-Recent-Trade-Open	MSinceOldestTradeOpen → MSinceMostRecentTradeOpen
AverageMInFile	MSinceOldestTradeOpen, MSinceMostRecentTradeOpen → AverageMInFile
NumSatisfactoryTrades	NumTotalTrades, PercentTradesNeverDelq → NumSatisfactoryTrades
Num-Trades-60-Ever-2-Derog-Pub-Rec	NumTotalTrades → NumTrades60Ever2DerogPubRec
Num-Trades-90-Ever-2-Derog-Pub-Rec	NumTotalTrades → NumTrades90Ever2DerogPubRec
PercentTradesNeverDelq	NumSatisfactoryTrades, NumTotalTrades → PercentTradesNeverDelq
MSinceMostRecentDelq	MSinceMostRecentTradeOpen → MSinceMostRecentDelq
Max-Delq-2-Public-Rec-Last-12M	NumTrades60Ever2DerogPubRec, NumTrades90Ever2DerogPubRec → MaxDelq2PublicRecLast12M
MaxDelqEver	MaxDelq2PublicRecLast12M → MaxDelqEver
NumTotalTrades	NumSatisfactoryTrades, NumTrades60Ever2DerogPubRec, NumTrades90Ever2DerogPubRec → NumTotalTrades
Num-Trades-Open-in-Last-12M	MSinceMostRecentTradeOpen, NumTotalTrades → NumTradesOpeninLast12M
PercentInstallTrades	NumInstallTradesWBalance, NumTotalTrades → PercentInstallTrades
M-Since-Most-Recent-Inqexcl7days	—
NumInqLast6M	MSinceMostRecentInqexcl7days → NumInqLast6M
NumInqLast6Mexcl7days	NumInqLast6M → NumInqLast6Mexcl7days
Net-Fraction-Revolving-Burden	NumRevolvingTradesWBalance → NetFractionRevolvingBurden
NetFractionInstallBurden	NumInstallTradesWBalance → NetFractionInstallBurden
Num-Revolving-Trades-W-Balance	NumTotalTrades → NumRevolvingTradesWBalance
Num-Install-Trades-W-Balance	NumTotalTrades → NumInstallTradesWBalance
NumBank-2-Natl-Trades-W-High-Utilization	NumRevolvingTradesWBalance → NumBank2NatlTradesWHighUtilization
PercentTradesWBalance	NumRevolvingTradesWBalance, NumInstallTradesWBalance, NumTotalTrades → PercentTradesWBalance
RiskPerformance	ExternalRiskEstimate, PercentTradesNeverDelq, NumTrades90Ever2DerogPubRec, MaxDelqEver, NetFractionRevolvingBurden, NumInqLast6M → RiskPerformance

Table 14: Extracted Feature Dependencies for `HELOC` Dataset

Feature	Dependencies
Diabetes_012	HighBP, HighChol, BMI, Smoker, Stroke, HeartDiseaseorAttack, PhysActivity, Fruits, Veggies, HvyAlcoholConsump, GenHlth, MentHlth, PhysHlth, DiffWalk, Age, Sex, Income, Education → Diabetes_012
HighBP	BMI, Age, Sex, PhysActivity, HeartDiseaseorAttack, GenHlth, Income → HighBP
HighChol	BMI, Age, HighBP, PhysActivity, GenHlth, Income → HighChol
CholCheck	AnyHealthcare, Income, Education → CholCheck
BMI	PhysActivity, Fruits, Veggies, Sex, Age → BMI
Smoker	Age, Sex, Education, Income → Smoker
Stroke	HighBP, HeartDiseaseorAttack, Age, GenHlth, BMI → Stroke
HeartDiseaseorAttack	HighBP, HighChol, Stroke, Age, BMI → HeartDiseaseorAttack
PhysActivity	Age, Sex, Income, Education → PhysActivity
Fruits	Income, Education, Sex → Fruits
Veggies	Income, Education, Sex → Veggies
HvyAlcoholConsump	Sex, Age, Income → HvyAlcoholConsump
AnyHealthcare	Income, Education → AnyHealthcare
NoDocbcCost	Income, AnyHealthcare → NoDocbcCost
GenHlth	PhysHlth, MentHlth, DiffWalk → GenHlth
MentHlth	Income, Age, Sex → MentHlth
PhysHlth	Age, Income, Sex → PhysHlth
DiffWalk	Age, BMI, HeartDiseaseorAttack → DiffWalk
Sex	—
Age	—
Education	—
Income	—

Table 15: Extracted Feature Dependencies for `CDC Diabetes` Dataset.