FAITH: TYPED CELL-LEVEL TABULAR GENERATION WITH A FROZEN LANGUAGE BACKBONE

Anonymous authors
Paper under double-blind review

ABSTRACT

Synthetic tabular data must reproduce heterogeneous column domains, their dependencies, and the predictive structure carried by complete rows. Pretrained causal language models offer a reusable mechanism for composing contextual dependencies, but existing language-model generators usually retain a textual interface: a cell expands into several tokens, prediction is performed over a general vocabulary, and numerical values are emitted as strings. We introduce FAITH, which keeps a frozen causal backbone while replacing its language interface with typed cell encoders and schema-local output distributions. Every cell occupies one causal position; categorical columns are predicted on their observed support, and numerical columns combine quantile-bin selection with a continuous conditional Beta density. This construction separates the sequence model that shares context across columns from the distributions that define individual columns. Across ten datasets and five baselines, FAITH obtains the highest primary synthetic-train-to-real-test utility on four datasets and its strongest joint quality on the dependency-intensive Allstate and HBA1C cases. On these datasets, FAITH trains 14.7–$16.0\times$ and samples 29.8–$47.4\times$ faster than the token-level GReaT baseline. The evaluation separately reports fidelity, utility, runtime, and empirical disclosure behavior.

1 INTRODUCTION

Synthetic data generation supports data sharing, augmentation, and imputation in settings where releasing or repeatedly accessing the original table is undesirable (Fonseca & Bacao, 2023; Assefa et al., 2020; Hernandez et al., 2022; Zheng & Charoenphakdee, 2022). Its difficulty comes from the object being generated: a row may combine categories, counts, continuous measurements, and missing-value atoms, while its usefulness depends on relationships that cross those incompatible domains. A generator must therefore preserve both local column structure and the dependencies that make a row meaningful.

Successive model families have addressed these demands by changing where heterogeneity is handled. CTGAN introduced type-aware representations and conditional adversarial training (Xu et al., 2019); TabDDPM and TabDiff assigned different diffusion processes to different field types (Kotelnikov et al., 2023; Shi et al., 2025); and TabSyn learned a common latent space before generation (Zhang et al., 2024). Autoregressive and masked alternatives instead decomposed the joint problem into field-level predictions, from MADE to TabMT and TabNAT (Germain et al., 2015; Gulati & Roysdon, 2023; Zhang et al., 2025). These models make column domains explicit, but learn from scratch the contextual transformation that must recover the dependencies among them. That burden is mild for simple pairwise regularities and more demanding when a finite table encodes domain-specific relations among many fields.

Pretrained language models suggest a different source for that shared mechanism. Their causal Transformers already compose heterogeneous context into a representation for the next prediction, and pretraining may supply transformations that need not be estimated from one table alone (Vaswani et al., 2017; Brown et al., 2020; Meng et al., 2022). This prior is most relevant when the table contains complex or narrow-domain dependencies, such as medical measurements whose joint meaning is not captured by their marginals. GReaT reused the pretrained mechanism by serializing feature–value pairs, and Tabby added column-oriented experts while retaining tokenized

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

values (Borisov et al., 2023; Cromp et al., 2026). They made pretrained contextualization available to tables, but coupled it to an interface designed for text.

That mismatch matters in three linked ways. Serialization expands one cell into a variable number of causal positions, asks a general vocabulary to represent a column-specific support, and expresses numerical magnitude through token identity rather than numerical density. Non-textual tabular models avoid these costs but do not test whether the pretrained causal operator can be retained independently of its language interface. Consequently, two lines of work have remained largely separate: native typed prediction has been developed with table-specific context models, while pretrained context modeling has been coupled to textual input and output.

We connect these lines by treating the language model as a context operator rather than a table decoder. FAITH maps each cell to one typed vector, passes the sequence through a frozen causal backbone, and predicts the next value with a distribution defined only on the target column. The central account is that pretrained causal computation is potentially useful for difficult inter-column dependencies, whereas token generation is incidental to that computation and inefficient for tables. If this distinction is correct, a typed interface should retain the strongest gains on dependency-intensive datasets while removing much of the cost of text serialization. The method develops the interface from factorization to normalized sampling; the experiments test its quality and cost, with Allstate and HBA1C serving as the principal stress cases.

Contributions. We make three contributions. First, we formulate typed cell-level generation as an interface around a frozen pretrained causal backbone, including a normalized mixed discrete–continuous conditional for numerical columns. Second, we demonstrate this construction with LFM2.5-230M, a 230M-parameter backbone (AI, 2026); the method therefore tests whether useful pretrained contextualization can be obtained from a very small language model rather than relying on scale. Third, we evaluate whether the resulting interface preserves difficult inter-column structure while separating distributional fidelity, predictive utility, runtime, and empirical disclosure behavior.

2 Related work

Specialized tabular generators. Specialized generators first made heterogeneity an explicit modeling concern. CTGAN combines type-specific encodings with conditional adversarial training (Xu et al., 2019); TabDDPM couples Gaussian and multinomial diffusion (Kotelnikov et al., 2023); TabSyn learns a continuous latent representation before diffusion (Zhang et al., 2024); and TabDiff retains mixed variables with feature-specific noise schedules (Shi et al., 2025). Their common achievement is to preserve native field structure without relying on language tokens. Their common separation from FAITH is the contextual model: it is learned for the tabular task rather than inherited as a frozen causal backbone.

Non-textual conditional tabular models. Conditional tabular models move the same type distinction into a factorized prediction problem. MADE established tractable autoregressive density estimation (Germain et al., 2015); TabMT uses field-specific embeddings and quantized outputs (Gulati & Roysdon, 2023); and TabNAT combines column-local categorical predictors with conditional diffusion for continuous fields (Zhang et al., 2025). These methods show that cells can be the units of conditional modeling. They do not, however, determine whether the context shared across those conditionals can come from a pretrained language model without converting the cells back into text.

Text-serialized language-model generators. Language-model generators approach the missing context mechanism from the opposite direction. GReaT fine-tunes an autoregressive model on feature–value strings (Borisov et al., 2023), and Tabby adds column experts while retaining tokenized string generation (Cromp et al., 2026). Their results establish that language backbones can model rows, but do not isolate the backbone from the serialization through which it is used. A matched cell-versus-text comparison is therefore necessary to distinguish reusable causal computation from the effects of the language interface.

Privacy and practical disclosure risk. Synthetic release introduces a second distinction between resemblance and disclosure. Membership and attribute inference can reveal information not captured

by fidelity metrics (Stadler et al., 2022); language models can memorize sequences (Carlini et al., 2021; Shi et al., 2024); and related behavior has been studied in tabular diffusion models (Fang et al., 2025). Differential privacy provides a formal objective under an explicit budget (Abadi et al., 2016; Li et al., 2022), whereas distance and match statistics describe empirical behavior. We retain that distinction when evaluating a generator that reuses a pretrained sequence model.

3 METHOD

3.1 THE TYPED AUTOREGRESSIVE PROBLEM

A causal backbone consumes an ordered sequence, whereas a table is defined by a product of heterogeneous domains. The first step is therefore to create a causal problem without erasing the type of the value being predicted.

Let $\mathcal{D} = \{x^{(i)}\}_{i=1}^{n}$ contain rows $x = (x_1, \ldots, x_d)$, where cell x_j belongs to the native domain $\mathcal{X}_j$ of column j. We write $\mathcal{C}$ and $\mathcal{N}$ for categorical and numerical column indices, respectively, and use a distinguished atom $\perp$ for observed missingness. A supervised target is treated as another column. After fixing a column order, the row distribution factorizes as

$$p_\theta(x) = \prod_{j=1}^{d} p_{\theta,j}(x_j \mid x_{<j}), \qquad x_{<j} = (x_1, \ldots, x_{j-1}). \tag{1}$$

Each factor is normalized on $\mathcal{X}_j$. The factorization supplies the causal structure, but not the interface between one cell and a language model. Serialization resolves that interface by changing x_j into tokens. The alternative is to preserve $\mathcal{X}_j$ and change only how the backbone receives and predicts a cell, which makes the modeling question unavoidable.

Research question

Q1. How can a frozen pretrained causal backbone be connected to native categorical and numerical cells without text serialization while retaining one normalized autoregressive model?

FAITH separates the cell interface from the computation shared across columns. A schema $\mathcal{S}$ maps x_j to a typed state q_j; an encoder $E_{\theta,j}$ maps that state to one vector $z_j \in \mathbb{R}^H$; the frozen causal backbone F_{W_0} maps the committed prefix to $h_{j-1} \in \mathbb{R}^H$; and a local decoder $D_{\theta,j}$ defines the next-cell distribution. These four operations are

$$\begin{aligned} q_j &= q_{\mathcal{S},j}(x_j), & z_j &= E_{\theta,j}(q_j), \\ h_{j-1} &= [F_{W_0}(z_{\mathrm{BOS}}, z_1, \ldots, z_{j-1})]_{j-1}, & p_{\theta,j}(\cdot \mid x_{<j}) &= D_{\theta,j}(h_{j-1}; \mathcal{S}), \end{aligned} \tag{2}$$

where W_0 is fixed and θ contains all trainable schema interfaces. Equation 2 makes the separation explicit: columns share the transformation F_{W_0}, but neither their input codes nor their output supports are forced to coincide.

3.2 TYPED INPUT AND FROZEN CONTEXTUALIZATION

The first step is to preserve the information that serialization would otherwise entangle. FAITH separates column identity, coarse value identity, and continuous within-bin position, then combines them in one vector occupying one causal position.

For every column, let $\mathcal{A}_j$ denote its finite set of local codes. A categorical value maps directly to $a_j \in \mathcal{A}_j$. For a numerical column, the training values define approximately equal-frequency bins $I_{j,b} = [\ell_{j,b}, u_{j,b}]$. A value in a non-singleton bin is represented by the bin code a_j and the normalized position

$$s_j = \frac{x_j - \ell_{j,b}}{u_{j,b} - \ell_{j,b}} \in [0, 1]. \tag{3}$$

Missing values and singleton bins need only a code because the code already determines the decoded value. Appendix B specifies boundary construction and endpoint clipping.

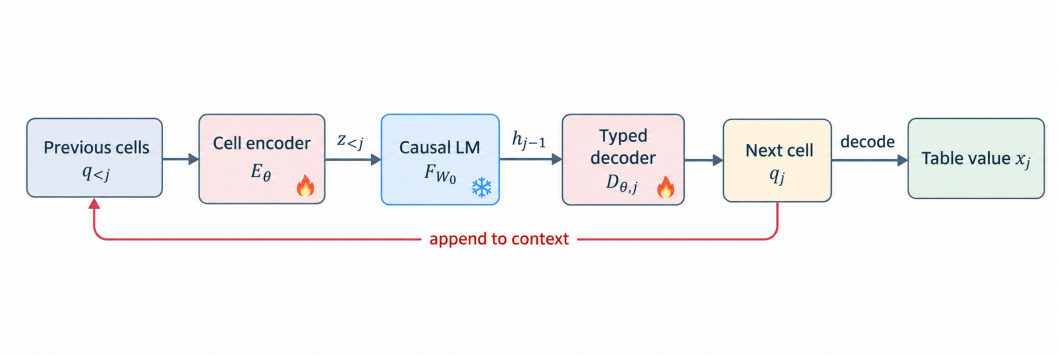

Figure 1: The FAITH generation path. A fitted schema and trainable encoder map one raw cell to one backbone vector. The frozen causal model contextualizes the committed prefix; the target-column head then returns a distribution only on that column's domain. The sampled cell is decoded, re-encoded, and appended. Trainable and frozen components are distinguished in the diagram.

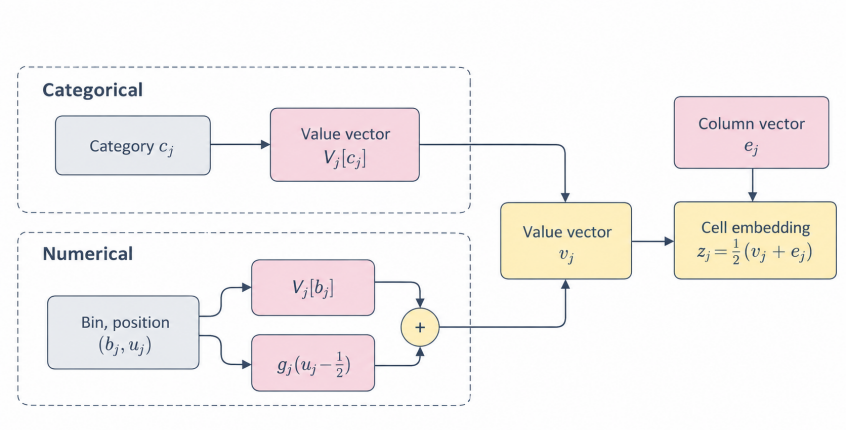

Figure 2: Typed cell encoder. Categorical cells use a learned local-code vector; numerical cells add an encoding of their within-bin position. Averaging the value and column vectors produces one cell embedding.

The cell encoder combines a trainable column vector e_j, a trainable code vector v_{j,a_j}, and, for an active numerical bin, a learned scalar-position map g_j. Its output is

$$z_j = \frac{1}{2}\left(e_j + v_{j,a_j} + I_j g_j(s_j - \tfrac{1}{2})\right), \tag{4}$$

where $I_j = 1$ only for a numerical value in a non-singleton bin and the final term is zero otherwise. Column names, category labels, and bin representatives initialize e_j and $v_{j,a}$ from the pretrained token embedding table; the vectors are then optimized. Text is thus an initialization source, not the repeated representation of every row.

Every cell occupies exactly one causal position regardless of the number of tokens needed to spell its name or value. Under teacher forcing, a row supplies all prefix states in one backbone pass. Freezing W_0 removes backbone gradients and optimizer states, while gradients still pass through its operations into the trainable interfaces. The backbone can therefore contextualize native cell vectors without becoming a table-specific set of updated weights.

3.3 Schema-local categorical and numerical conditionals

The shared prefix state must next return to a native column domain. Sharing context does not imply sharing an output vocabulary: each target column asks a different typed question of the same contextual representation.

A target-column query combines h_{j-1} with e_j and produces logits only for $\mathcal{A}_j$. We denote the resulting mass function by $\pi_{\theta,j}(a \mid h_{j-1})$. For a categorical column, this softmax is the complete conditional distribution, so no probability is assigned outside the observed support of that column.

Numerical columns require both coarse region selection and continuous local precision. FAITH first predicts the local code a_j, which selects missingness, a singleton value, or a bin b. For an active bin b, a second head produces positive $\alpha_{\theta,j,b}$ and $\beta_{\theta,j,b}$ and defines

$$p_{\theta,j}(a_j, s_j \mid x_{<j}) = \pi_{\theta,j}(a_j \mid h_{j-1}) \operatorname{Beta}(s_j; \alpha_{\theta,j,b}(h_{j-1}), \beta_{\theta,j,b}(h_{j-1})) . \tag{5}$$

The local softmax allocates mass across separated regions, while the bounded Beta density controls skew and concentration within the selected region. Missingness and singleton bins remain discrete atoms. The raw-scale density and normalization proof are given in Appendix C.

3.4 Normalized training and sampling

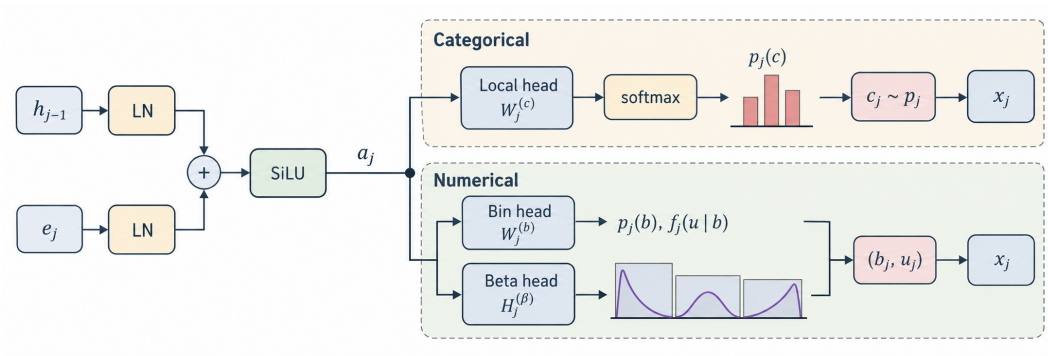

Figure 3: Schema-local decoder. The prefix state and target-column vector parameterize either a categorical softmax or a numerical bin distribution with a conditional within-bin Beta density.

The typed input and local output heads become one generator only if training and sampling use the same normalized conditionals. Teacher forcing evaluates those conditionals in parallel, while generation commits to them sequentially.

Teacher forcing evaluates all next-cell conditionals in parallel. Let $a_j^{(i)}$ be the local code of $x_j^{(i)}$, let $b_j^{(i)}$ be its numerical bin when defined, and let $I_j^{(i)}$ mark an active numerical bin. With s_j^{δ} denoting the endpoint-clipped position used for stable Beta likelihoods, the empirical objective is

$$\begin{aligned} \mathcal{L}_{\mathrm{AR}}(\theta) &= \frac{1}{nd} \sum_{i=1}^{n} \sum_{j=1}^{d} \ell_{ij}(\theta), \qquad W_0 \text{ fixed}, \\ \ell_{ij}(\theta) &= -\log \pi_{\theta,j}(a_j^{(i)} \mid h_{j-1}^{(i)}) \\ &\quad - \omega_{\mathrm{B}} I_j^{(i)} \log \operatorname{Beta}\Big(s_j^{(i),\delta}; \alpha_{\theta,j,b_j^{(i)}}(h_{j-1}^{(i)}), \beta_{\theta,j,b_j^{(i)}}(h_{j-1}^{(i)})\Big) . \end{aligned} \tag{6}$$

Thus categorical, missing, and singleton targets use only code cross-entropy, whereas active numerical targets also train a continuous density. Full algorithms and implementation rules are provided in Appendix D.

At generation time, FAITH begins with a learned BOS vector and follows the fitted column order. It samples a local code, samples a Beta position when the code identifies an active numerical bin, decodes the value, and appends its re-encoding to the prefix. The procedure uses d autoregressive cell positions for a d-column row and conditions every next draw on an actual committed prefix.

Answer to Q1. FAITH connects a frozen causal backbone to heterogeneous tables by assigning one vector and one local predictive distribution to each cell. The encoder preserves column identity and numerical position, the backbone shares prefix information, and the local head returns a normalized categorical or mixed numerical conditional. The construction gives a precise structural

consequence: one causal position per column. It does not yet give an empirical conclusion, so we must determine whether this representation remains useful on real tables and whether the shorter causal sequence changes the cost relative to token-level generators.

4 EXPERIMENTS

The method changes both the representation of a row and the boundary of adaptation. Its value therefore depends on two linked outcomes: the cell interface must preserve the dependencies that make a synthetic row useful, and the shorter causal sequence must reduce the cost inherited from token-level generation. We evaluate these outcomes together and then examine whether the resulting samples remain distinguishable from individual training records.

Research question

Q2. Does the typed cell-level interface preserve complex inter-column dependencies while reducing training and generation cost relative to text-serialized language-model generators, particularly on large or domain-specific tables?

4.1 EXPERIMENTAL DESIGN

We compare FAITH with CTGAN, the diffusion models TabDDPM and TabSyn, and the token-level language generators GReaT and Tabby on ten classification and regression datasets. FAITH uses the frozen 230M-parameter LFM2.5-230M backbone (AI, 2026). All runs use an NVIDIA H100 GPU and the same train, validation, and test partitions. For every method and dataset, we generate exactly as many synthetic rows as occur in the real training split. Language-model-based methods use validation early stopping with patience 5; diffusion baselines use patience 100 to accommodate their slower convergence. Every comparison orders methods by generator family, beginning with adversarial models, followed by diffusion models and pretrained causal sequence models. The FAITH row is shaded.

Statistical fidelity uses SDMetrics *Column Shapes* and *Column Pair Trends*, together with integrated alpha-precision and beta-recall. Machine learning efficiency (MLE) trains a predictor on synthetic data and evaluates it on the held-out real test split, using macro-F1 for classification and R^2 for regression. Runtime records training and generation separately. DCR, NNDR, identical-match statistics, and membership-inference attacks characterize empirical disclosure rather than formal privacy.

All scores are shown to two decimal places with a leading zero; bold face follows the unrounded point estimate. MLE brackets are 95% percentile bootstrap confidence intervals over test rows and do not measure variability across generator runs. Dataset provenance and complete secondary metrics are provided in Appendices A and E.

4.2 QUALITY AND COMPLEX DEPENDENCIES

The first part of Q2 asks whether replacing tokens with cells preserves useful row structure. Table 1 evaluates the strongest test of this structure: whether a model trained on synthetic rows transfers to held-out real data.

Table 1: Synthetic-train-to-real-test utility (↑): macro-F1 for classification and R^2 for regression. Brackets are 95% bootstrap confidence intervals.

Method	Adult F1 ↑	Allstate R^2 ↑	Beijing R^2 ↑	Default F1 ↑	Diabetes F1 ↑	HBA1C F1 ↑	House R^2 ↑	Magic F1 ↑	News R^2 ↑	Shoppers F1 ↑
Adversarial										
CTGAN	0.77 [0.76, 0.78]	−0.03 [−0.06, −0.01]	0.16 [0.15, 0.18]	0.58 [0.57, 0.60]	0.58 [0.49, 0.66]	0.34 [0.34, 0.35]	0.42 [0.39, 0.46]	0.55 [0.53, 0.57]	**−0.03** [−0.09, −0.02]	0.69 [0.66, 0.72]
Diffusion										
TabDDPM	0.79 [0.78, 0.80]	0.38 [0.35, 0.42]	0.53 [0.51, 0.55]	0.66 [0.64, 0.67]	0.74 [0.65, 0.82]	0.32 [0.31, 0.32]	0.79 [0.77, 0.81]	0.85 [0.84, 0.87]	−492.70 [−1354.14, −239.64]	0.71 [0.69, 0.74]
TabSyn	0.79 [0.78, 0.80]	0.48 [0.44, 0.51]	0.48 [0.46, 0.50]	0.64 [0.62, 0.66]	**0.76** [0.67, 0.83]	0.38 [0.37, 0.38]	**0.79** [0.77, 0.81]	**0.86** [0.85, 0.87]	−1.95 [−5.72, −0.88]	0.78 [0.76, 0.81]
Pretrained causal sequence models										
GReaT	0.80 [0.79, 0.81]	0.44 [0.40, 0.47]	0.53 [0.51, 0.55]	**0.68** [0.67, 0.70]	0.60 [0.52, 0.69]	0.38 [0.37, 0.39]	-7.38×10^8 $[-2.32 \times 10^9, -827.48]$	0.52 [0.50, 0.53]	−0.91 [−2.54, −0.45]	**0.79** [0.77, 0.82]
Tabby	0.80 [0.79, 0.81]	0.46 [0.43, 0.50]	0.53 [0.51, 0.55]	0.65 [0.64, 0.67]	0.67 [0.59, 0.76]	0.37 [0.36, 0.38]	0.66 [0.64, 0.68]	0.70 [0.69, 0.72]	−0.77 [−1.82, 0.28]	0.77 [0.75, 0.80]
FAITH	**0.80** [0.79, 0.81]	**0.52** [0.48, 0.55]	**0.55** [0.53, 0.57]	0.66 [0.65, 0.68]	0.65 [0.56, 0.73]	**0.41** [0.40, 0.42]	0.71 [0.69, 0.73]	0.81 [0.80, 0.83]	−0.45 [−1.26, −0.21]	0.78 [0.75, 0.81]

No generator dominates all datasets. FAITH leads primary utility on Adult, Allstate, Beijing, and HBA1C, while other model families lead elsewhere. The concentration on Allstate and HBA1C is important because these datasets motivate the use of pretrained contextual computation: their predictive targets depend on heterogeneous, domain-specific relations rather than on isolated marginals. Table 2 therefore pairs downstream utility with pairwise fidelity on these two stress cases. Complete marginal, pairwise, precision, and coverage results are reported in Appendix Tables 6 and 7.

Table 2: Utility and pairwise fidelity on the two dependency-intensive stress cases (↑).

	Allstate		HBA1C	
Method	R^2	Pair	F1	Pair
Adversarial				
CTGAN	−0.03	0.77	0.34	0.82
Diffusion				
TabDDPM	0.38	0.90	0.32	0.36
TabSyn	0.48	0.90	0.38	0.91
Pretrained causal sequence models				
GReaT	0.44	0.90	0.38	0.80
Tabby	0.46	0.88	0.37	0.82
FAITH	**0.52**	**0.94**	**0.41**	**0.93**

Finding: the advantage concentrates on dependency-intensive tables. FAITH is the only method that leads both utility and pairwise fidelity on both stress cases. On HBA1C, the from-scratch generators lose either predictive performance or dependency fidelity; GReaT and Tabby retain the pretrained context model but trail FAITH when values remain tokenized. The result supports the central interface argument: pretrained contextual computation is most useful for difficult relations when the table is represented at cell level.

4.3 Computational profile

The second part of Q2 asks whether the structural reduction from tokens to cells appears in measured cost. Table 3 reports training and generation time under the common H100 protocol and equal synthetic sample counts.

Table 3: Training and generation time in seconds (↓).

Phase	Method	Adult	Allstate	Beijing	Default	Diabetes	HBA1C	House	Magic	News	Shoppers
Train	*Adversarial*										
	CTGAN	1.2×10^{4}	1.2×10^{5}	8.1×10^{3}	1.1×10^{4}	5.6×10^{1}	4.5×10^{4}	3.1×10^{3}	2.2×10^{3}	2.4×10^{4}	3.8×10^{3}
	Diffusion										
	TabDDPM	7.2×10^{2}	$\mathbf{4.6\times10^{3}}$	6.5×10^{2}	$\mathbf{6.9\times10^{2}}$	8.4×10^{1}	$\mathbf{7.0\times10^{2}}$	4.2×10^{2}	$\mathbf{2.6\times10^{2}}$	$\mathbf{5.5\times10^{2}}$	4.4×10^{2}
	TabSyn	3.3×10^{3}	3.3×10^{4}	2.9×10^{3}	3.2×10^{3}	3.6×10^{2}	1.2×10^{4}	1.8×10^{3}	1.2×10^{3}	2.5×10^{3}	7.6×10^{2}
	Pretrained causal sequence models										
	GReaT	1.0×10^{4}	2.3×10^{5}	1.6×10^{4}	4.8×10^{3}	1.4×10^{2}	5.8×10^{4}	1.6×10^{3}	4.1×10^{3}	1.3×10^{4}	1.7×10^{3}
	Tabby	1.4×10^{4}	3.1×10^{5}	2.2×10^{4}	6.5×10^{3}	1.9×10^{2}	7.8×10^{4}	2.2×10^{3}	5.5×10^{3}	1.8×10^{4}	2.3×10^{3}
	FAITH	$\mathbf{5.4\times10^{2}}$	1.5×10^{4}	$\mathbf{6.4\times10^{2}}$	8.8×10^{2}	$\mathbf{2.1\times10^{1}}$	4.0×10^{3}	$\mathbf{3.4\times10^{2}}$	4.6×10^{2}	2.3×10^{3}	$\mathbf{2.8\times10^{2}}$
Generate	*Adversarial*										
	CTGAN	1.4×10^{1}	1.3×10^{2}	9.8	1.5×10^{1}	**0.33**	7.5×10^{1}	5.1	5.3	2.6×10^{1}	5.5
	Diffusion										
	TabDDPM	4.5×10^{1}	2.7×10^{3}	3.1×10^{1}	3.0×10^{1}	0.84	4.5×10^{2}	5.7	5.9	1.5×10^{1}	1.5×10^{1}
	TabSyn	**6.7**	$\mathbf{4.4\times10^{1}}$	**7.4**	**5.8**	0.37	$\mathbf{1.3\times10^{1}}$	**4.4**	**1.9**	**4.5**	**2.3**
	Pretrained causal sequence models										
	GReaT	6.4×10^{4}	$\approx 2.1\times10^{5}$	5.6×10^{4}	2.0×10^{4}	7.3×10^{2}	8.9×10^{4}	1.8×10^{4}	6.8×10^{4}	7.3×10^{4}	9.9×10^{3}
	Tabby	8.6×10^{4}	2.8×10^{5}	7.6×10^{4}	2.7×10^{4}	9.9×10^{2}	1.2×10^{5}	2.4×10^{4}	9.2×10^{4}	9.9×10^{4}	1.3×10^{4}
	FAITH	2.2×10^{2}	7.1×10^{3}	1.9×10^{2}	5.5×10^{2}	6.1	1.9×10^{3}	1.4×10^{2}	2.4×10^{2}	2.5×10^{2}	1.8×10^{2}

Finding: the cell interface removes text-generation cost, not sequential generation itself. On Allstate and HBA1C, FAITH is more than an order of magnitude faster than GReaT in both training and generation. The largest reduction appears in generation because a complete cell replaces a variable-length token string. CTGAN and TabSyn remain faster samplers because they avoid the same sequential cell-by-cell loop; the measured advantage is therefore specific to removing the textual interface from pretrained causal generation.

4.4 Empirical disclosure behavior

High fidelity does not determine whether individual training records are exposed. Table 4 reports the strict composite screen used in the evaluation: a dataset passes only when its DCR, NNDR, and identical-match tests all pass.

Table 4: Composite disclosure-risk screen across ten datasets.

Method	Passed	Failed
Adversarial		
CTGAN	9	Beijing
Diffusion		
TabDDPM	7	Adult, Beijing, Diabetes
TabSyn	8	Beijing, Diabetes
Pretrained causal sequence models		
GReaT	2	all except House, Magic
Tabby	3	all except House, Magic, Shoppers
FAITH	9	Beijing

FAITH passes on nine datasets, matching CTGAN and exceeding the other completed baselines. The membership-inference attacks in Appendix Table 9 remain close to chance in most runs. These measurements characterize empirical disclosure behavior rather than providing a formal privacy guarantee.

Answer to Q2. The typed interface remains competitive across the full benchmark and gives its clearest advantage on the two dependency-intensive stress cases, where FAITH leads both utility and pairwise fidelity. The same representation removes the dominant token-generation cost of GReaT while retaining a small frozen backbone. The quality, runtime, and disclosure results therefore support the system-level position tested here: pretrained contextual computation and typed cell prediction are complementary.

5 CONCLUSION

FAITH separates two roles that textual tabular generators combine: contextualizing a row and representing an individual value. A frozen pretrained causal backbone performs the first role; typed encoders and schema-local categorical or mixed numerical distributions perform the second. Across ten datasets, the distinction matters most where dependencies are difficult to estimate from one table: FAITH obtains the strongest joint utility and pairwise fidelity on Allstate and HBA1C, while the evaluated from-scratch generators lose either predictive or dependency structure. Relative to GReaT, the typed interface also reduces training and sampling time by more than an order of magnitude on both stress cases. The resulting position is precise: pretrained causal computation is a useful candidate prior for complex tabular dependencies, but it should be exposed through cells rather than through serialized strings.

AI USE STATEMENT

In this work, we used generative AI tools to aid and polish the writing and for retrieval and discovery, such as finding related work. We have not used generative AI tools for research ideation or execution, including proposing hypotheses, designing experiments, implementing methods and interpreting results, for drafting sections of the paper, for generating synthetic datasets, or for translation, and proving mathematical claims is not applicable to this work. Additionally, we used generative AI tools to create and modify scientific figures and to review software code for errors. We have reviewed all AI-assisted work. We read every cited paper and checked each reference against its original source. All plotted values come from our experimental outputs, and every figure was checked against them. Every code change suggested by these tools was inspected and tested by the authors before it was used. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

The main method specifies the probabilistic factorization and trainable/frozen boundary. Appendix A records benchmark provenance, Appendices B–D expand the construction used to answer Q1, and Appendix E provides the additional evidence used to interpret Q2.

REFERENCES

Martin Abadi, Andy Chu, Ian Goodfellow, H. Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pp. 308–318, 2016. doi: 10.1145/2976749.2978318.

Liquid AI. Lfm2.5-230m: Built to run anywhere. *Liquid AI Blog*, 2026. www.liquid.ai/blog/lfm2-5-230m.

Samuel A. Assefa, Danial Dervovic, Mahmoud Mahfouz, Robert E. Tillman, Prashant Reddy, and Manuela Veloso. Generating synthetic data in finance: Opportunities, challenges and pitfalls. In *Proceedings of the First ACM International Conference on AI in Finance*, pp. 1–8, 2020. doi: 10.1145/3383455.3422554.

Vadim Borisov, Kathrin Seßler, Tobias Leemann, Martin Pawelczyk, and Gjergji Kasneci. Language models are realistic tabular data generators. In *International Conference on Learning Representations*, 2023. URL https://arxiv.org/abs/2210.06280.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D. Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pp.

1877–1901, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/1457c0d6bfcb4967418bfb8ac142f64a-Abstract.html.

Nicholas Carlini, Florian Tramèr, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Úlfar Erlingsson, Alina Oprea, and Colin Raffel. Extracting training data from large language models. In *30th USENIX Security Symposium*, pp. 2633–2650. USENIX Association, 2021. URL https://www.usenix.org/conference/usenixsecurity21/presentation/carlini-extracting.

Sonia Cromp, Satya Sai Srinath Namburi GNVV, Mohammed Alkhudhayri, Catherine Cao, Samuel Guo, Nicholas Roberts, and Frederic Sala. Tabby: A language model architecture for tabular and structured data synthesis. *arXiv preprint arXiv:2503.02152v2*, 2026. URL https://arxiv.org/abs/2503.02152v2.

Zhengyu Fang, Zhimeng Jiang, Huiyuan Chen, Xiao Li, and Jing Li. Understanding and mitigating memorization in diffusion models for tabular data. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 15976–16005, 2025. URL https://proceedings.mlr.press/v267/fang25f.html.

Jorge Fonseca and Fernando Bacao. Tabular and latent space synthetic data generation: A literature review. *Journal of Big Data*, 10:115, 2023. doi: 10.1186/s40537-023-00792-7.

Mathieu Germain, Karol Gregor, Iain Murray, and Hugo Larochelle. MADE: Masked autoencoder for distribution estimation. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37, pp. 881–889, 2015. URL https://arxiv.org/abs/1502.03509.

Manbir Gulati and Keith Roysdon. TabMT: Generating tabular data with masked transformers. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/90debc7cedb5cac83145fc8d18378dc5-Abstract-Conference.html.

Mikel Hernandez, Gorka Epelde, Ane Alberdi, Rodrigo Cilla, and Debra Rankin. Synthetic data generation for tabular health records: A systematic review. *Neurocomputing*, 493:28–45, 2022. doi: 10.1016/j.neucom.2022.04.053.

Akim Kotelnikov, Dmitry Baranchuk, Ivan Rubachev, and Artem Babenko. TabDDPM: Modelling tabular data with diffusion models. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 17564–17579, 2023. URL https://proceedings.mlr.press/v202/kotelnikov23a.html.

Xuechen Li, Florian Tramèr, Percy Liang, and Tatsunori Hashimoto. Large language models can be strong differentially private learners. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=bVuP3ltATMz.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in GPT. In *Advances in Neural Information Processing Systems*, volume 35, pp. 17359–17372, 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/6f1d43d5a82a37e89b0665b33bf3a182-Abstract-Conference.html.

Juntong Shi, Minkai Xu, Harper Hua, Hengrui Zhang, Stefano Ermon, and Jure Leskovec. TabDiff: A mixed-type diffusion model for tabular data generation. In *International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2410.20626.

Weijia Shi, Anirudh Ajith, Mengzhou Xia, Yangsibo Huang, Daogao Liu, Terra Blevins, Danqi Chen, and Luke Zettlemoyer. Detecting pretraining data from large language models. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/e32ad85fa27be4a9868d55703f01323e-Abstract-Conference.html.

Theresa Stadler, Bristena Oprisanu, and Carmela Troncoso. Synthetic data—anonymisation groundhog day. In *31st USENIX Security Symposium*, pp. 1451–1468. USENIX Association, 2022. URL https://www.usenix.org/conference/usenixsecurity22/presentation/stadler.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, pp. 5998–6008, 2017. URL https://proceedings.neurips.cc/paper/7181-attention-is-all-you-need.

Lei Xu, Maria Skoularidou, Alfredo Cuesta-Infante, and Kalyan Veeramachaneni. Modeling tabular data using conditional GAN. In *Advances in Neural Information Processing Systems*, volume 32, 2019. URL https://arxiv.org/abs/1907.00503.

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. In *International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2310.09656.

Hengrui Zhang, Liancheng Fang, Qitian Wu, and Philip S. Yu. TabNAT: A continuous-discrete joint generative framework for tabular data. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 74858–74881, 2025. URL https://proceedings.mlr.press/v267/zhang25t.html.

Xinyang Zheng and Nontawat Charoenphakdee. TabCSDI: Conditional score-based diffusion models for probabilistic time series imputation. *arXiv preprint arXiv:2210.17128*, 2022. URL https://arxiv.org/abs/2210.17128.

The appendix follows the argument of the main paper. Appendix A documents the empirical inputs. Appendices B–D expand the typed construction behind Q1 in the same order as a cell moves through FAITH. Appendix E then supplies the secondary evidence behind the quality and disclosure conclusions of Q2 without repeating the main utility, runtime, or composite disclosure tables.

A Benchmark provenance

The empirical study in Section 4 depends on ten public or competition datasets. Table 5 records the source associated with each benchmark and its stated access condition. UCI lists the included datasets under CC BY 4.0, whereas Allstate is governed by the Kaggle competition rules. HBA1C is distributed as a project benchmark copy without separate source metadata.

Table 5: Dataset sources and usage terms.

Dataset	Source	License or access condition
Adult	https://archive.ics.uci.edu/dataset/2/adult	CC BY 4.0
Allstate	https://www.kaggle.com/c/allstate-claims-severity/data	Kaggle competition rules
Beijing	https://archive.ics.uci.edu/dataset/501/beijing+multi+site+air+quality	CC BY 4.0
Default	https://archive.ics.uci.edu/dataset/350/default+of+credit+card+clients	CC BY 4.0
Diabetes	https://www.openml.org/api/v1/json/data/37	Public (OpenML metadata)
HBA1C	Project benchmark copy	Source license not distributed with the copy
House	https://figshare.com/articles/dataset/cal_housing_tgz/3829992	CC BY 4.0
Magic	https://archive.ics.uci.edu/dataset/159/magic+gamma+telescope	CC BY 4.0
News	https://archive.ics.uci.edu/dataset/332/online+news+popularity	CC BY 4.0
Shoppers	https://archive.ics.uci.edu/dataset/468/online+shoppers+purchasing+intention+dataset	CC BY 4.0

B TYPED SCHEMA AND CELL ENCODING

This section expands the schema map and cell encoder summarized in Equation 2. The order follows the generation path: a raw value is first assigned a column-local code, numerical values retain a within-bin coordinate, and the resulting state is mapped to one backbone vector.

B.1 COLUMN-LOCAL CODES

For categorical column j, let $\mathcal{A}_j = \{0, \dots, K_j - 1\}$ index the observed support and the missing atom when present. Then

$$q_{\mathcal{S},j}(x_j) = a_j \in \mathcal{A}_j. \tag{7}$$

Validation-time categories are not inserted after schema fitting.

For numerical column j, let finite training values define B_j intervals $I_{j,b} = [\ell_{j,b}, u_{j,b}]$, with candidate boundaries only between distinct values. Let $m_j \in \{0, 1\}$ indicate whether a missing code is present and let $b_j(x)$ denote the bin containing x. The numerical code is

$$q_{\mathcal{S},j}(x_j) = \begin{cases} (0, \varnothing), & x_j = \perp, \\ (m_j + b_j(x_j), \varnothing), & u_{j,b_j(x_j)} = \ell_{j,b_j(x_j)}, \\ (m_j + b_j(x_j), s_j), & \text{otherwise}, \end{cases} \tag{8}$$

where

$$s_j = \frac{x_j - \ell_{j,b_j(x_j)}}{u_{j,b_j(x_j)} - \ell_{j,b_j(x_j)}} \in [0, 1]. \tag{9}$$

Singleton bins are inactive because their code already determines the value. To avoid evaluating a Beta log density at an endpoint, training uses $s_j^\delta = \min\{1 - \delta, \max\{\delta, s_j\}\}$ for fixed $0 < \delta \ll 1$.

B.2 ONE VECTOR PER CELL

FAITH represents column j by $e_j \in \mathbb{R}^H$ and local code a by $v_{j,a} \in \mathbb{R}^H$. A categorical, missing, or inactive numerical value uses $z_j = \frac{1}{2}(e_j + v_{j,a_j})$. An active numerical value additionally uses

$$g_j(t) = A_{j,2}\,\mathrm{SiLU}(A_{j,1}t + b_{j,1}) + b_{j,2}, \qquad z_j = \tfrac{1}{2}\left(e_j + v_{j,a_j} + g_j(s_j - \tfrac{1}{2})\right). \tag{10}$$

The final affine map of g_j is initialized to zero so that optimization begins from the simpler column-plus-code representation.

B.3 SEMANTIC INITIALIZATION

If E_0 is the pretrained token embedding table and $\tau(t)$ tokenizes string t, semantic initialization uses

$$\mathrm{Init}(t) = \frac{1}{|\tau(t)|} \sum_{w \in \tau(t)} E_0(w). \tag{11}$$

Column names initialize e_j, category labels initialize categorical code vectors, and textual bin representatives initialize numerical code vectors. These vectors remain trainable.

C NORMALIZED LOCAL OUTPUT DISTRIBUTIONS

The main paper introduces the numerical head through bin selection and a conditional Beta density. This section gives the corresponding local code distribution, raw-scale density, and normalization argument.

C.1 TARGET-CONDITIONED LOCAL PREDICTION

The target-aware query used by each local head is

$$r_j(h) = \mathrm{SiLU}(\mathrm{LN}_{\gamma,\beta}(h) + \mathrm{LN}_0(e_j)), \tag{12}$$

where the first layer normalization is shared and trainable and the second has no learned affine parameters. The local code probabilities are

$$\pi_{\theta,j}(a \mid h) = \mathrm{softmax}(W_j r_j(h) + b_j)_a. \tag{13}$$

For each active numerical bin, positive parameters are produced with softplus plus a small $\epsilon > 0$. They are initialized so that $\alpha_{\theta,j,b} = \beta_{\theta,j,b} = 1$, corresponding to an initially uniform within-bin density.

C.2 Raw-scale likelihood and normalization

For a value x_j in active bin b, changing variables from normalized position to the raw scale gives

$$p_{\theta,j}(x_j \mid x_{<j}) = \frac{\pi_{\theta,j}(m_j + b \mid h_{j-1})}{u_{j,b} - \ell_{j,b}} \operatorname{Beta}\left(\frac{x_j - \ell_{j,b}}{u_{j,b} - \ell_{j,b}}; \alpha_{\theta,j,b}(h_{j-1}), \beta_{\theta,j,b}(h_{j-1})\right). \tag{14}$$

If $\mathcal{B}_j^+$ and $\mathcal{B}_j^0$ denote active and inactive bins, respectively, the mixed measure is normalized because

$$m_j \pi_{\theta,j}(0 \mid h) + \sum_{b \in \mathcal{B}_j^0} \pi_{\theta,j}(m_j + b \mid h) + \sum_{b \in \mathcal{B}_j^+} \int_{\ell_{j,b}}^{u_{j,b}} p_{\theta,j}(x \mid x_{<j})\, dx = 1. \tag{15}$$

The equality follows because every active-bin Beta density integrates to its selected-bin mass, while missing and singleton codes remain discrete atoms. Thus every column head defines one normalized conditional on its native mixed domain.

D Optimization and autoregressive decoding

The objective in Equation 6 is evaluated with teacher forcing, but deployment commits sampled cells sequentially. Algorithms 1 and 2 state the two procedures using the same schema and local distributions.

D.1 Training

Algorithm 1 Training FAITH

Require: Training table $\mathcal{D}$, frozen backbone F_{W_0}, weight ω_{B}
  $\mathcal{S} \leftarrow \mathrm{FitSchema}(\mathcal{D})$; initialize θ
  repeat
    $\mathcal{B} \leftarrow \mathrm{Minibatch}(\mathcal{D})$
    $Z \leftarrow E_\theta(q_{\mathcal{S}}(\mathcal{B}))$
    $H \leftarrow F_{W_0}(z_{\mathrm{BOS}}, Z_{1:d-1})$
    evaluate Equation 6 on $\mathcal{B}$
    update θ with AdamW; keep W_0 fixed
  until validation stopping criterion
  return $(\mathcal{S}, F_{W_0}, \theta)$

D.2 Generation

Algorithm 2 Sampling from FAITH

Require: $(\mathcal{S}, F_{W_0}, \theta)$, number of rows n
  for $r = 1, \ldots, n$ **do**
    $P \leftarrow (z_{\mathrm{BOS}})$
    for $j = 1, \ldots, d$ **do**
      $h \leftarrow \mathrm{Last}(F_{W_0}(P))$; sample $a_j \sim \pi_{\theta,j}(\cdot \mid h)$
      if a_j selects an active numerical bin b **then**
        sample $s_j \sim \mathrm{Beta}(\alpha_{\theta,j,b}(h), \beta_{\theta,j,b}(h))$
      else
        $s_j \leftarrow \varnothing$
      end if
      $\widetilde{x}_j \leftarrow \mathrm{Dec}_{\mathcal{S},j}(a_j, s_j)$
      $P \leftarrow P \,\|\, E_{\theta,j}(a_j, s_j)$
    end for
  end for
  return $\{\widetilde{x}^{(r)}\}_{r=1}^n$

For active numerical bin b, decoding returns $\widetilde{x}_j = \ell_{j,b} + s_j(u_{j,b} - \ell_{j,b})$. Singleton bins return their stored value. Integer-valued columns round to the nearest integer and clamp the result to the selected bin's integer support.

E ADDITIONAL EVIDENCE FOR Q2

The main paper reports the primary utility, full runtime, stress-case comparison, and composite disclosure screen. This section contains only complementary measurements needed to interpret those conclusions.

E.1 LOW-ORDER FIDELITY AND DEPENDENCE PRESERVATION

Column Shapes measures marginal agreement, while Column Pair Trends measures whether pairwise relations are retained. Table 6 provides the full benchmark behind the stress-case summary in Table 2.

Table 6: Low-order fidelity: SDMetrics Column Shapes and Column Pair Trends (↑).

Metric	Method	Adult	Allstate	Beijing	Default	Diabetes	HBA1C	House	Magic	News	Shoppers
Shape	*Adversarial*										
	CTGAN	0.88	0.93	0.92	0.86	0.67	0.96	0.94	0.89	0.84	0.81
	Diffusion										
	TabDDPM	0.99	0.97	**0.99**	0.98	0.92	0.74	0.97	0.97	0.24	0.61
	TabSyn	**0.99**	0.98	0.97	**0.99**	0.95	**0.98**	**0.99**	**0.99**	**0.93**	**0.99**
	Pretrained causal sequence models										
	GReaT	0.90	0.97	0.96	0.75	0.83	0.93	0.86	0.82	0.79	0.82
	Tabby	0.90	0.90	0.90	0.89	0.80	0.86	0.90	0.88	0.84	0.85
	FAITH	0.96	**0.98**	0.94	0.95	**0.96**	0.98	0.97	0.98	0.89	0.96
Pair	*Adversarial*										
	CTGAN	0.88	0.77	0.88	0.84	0.61	0.82	0.82	0.78	0.85	0.83
	Diffusion										
	TabDDPM	0.98	0.90	**0.96**	0.95	0.88	0.36	0.91	0.99	0.62	0.77
	TabSyn	**0.99**	0.90	0.96	**0.99**	**0.93**	0.91	**0.98**	**0.99**	0.90	**0.99**
	Pretrained causal sequence models										
	GReaT	0.88	0.90	0.74	0.84	0.85	0.80	0.77	0.84	0.87	0.89
	Tabby	0.85	0.88	0.81	0.85	0.86	0.82	0.85	0.85	0.90	0.92
	FAITH	0.93	**0.94**	0.94	0.93	0.72	**0.93**	0.94	0.96	**0.92**	0.96

Across the full benchmark, TabSyn most often leads the low-order scores. FAITH is distinguished by the concentration of its strongest pairwise results on Allstate and HBA1C, the same datasets on which it leads downstream utility in the main paper. This alignment is why Section 4 treats these datasets as stress cases rather than summarizing performance through an average over heterogeneous tasks.

E.2 SUPPORT PRECISION AND COVERAGE

Low-order scores do not reveal whether a generator covers the real support or concentrates on a narrow high-density subset. Table 7 reports alpha-precision and beta-recall to separate these behaviors.

Table 7: Integrated alpha-precision and beta-recall (↑).

Metric	Method	Adult	Allstate	Beijing	Default	Diabetes	HBA1C	House	Magic	News	Shoppers
Alpha	*Adversarial*										
	CTGAN	0.74	0.13	0.89	0.90	0.90	0.96	0.86	0.45	0.20	0.84
	Diffusion										
	TabDDPM	**0.98**	0.44	0.97	0.98	**0.96**	0.48	0.96	0.99	0.02	0.33
	TabSyn	0.97	0.53	0.99	**0.99**	0.91	0.96	**0.99**	**0.99**	**0.64**	**0.99**
	Pretrained causal sequence models										
	GReaT	0.89	0.87	0.97	0.81	0.95	0.93	0.97	0.97	**0.73**	0.68
	Tabby	0.91	0.89	0.98	0.84	0.95	0.94	0.95	0.95	0.62	0.74
	FAITH	0.98	**0.98**	**1.00**	0.98	0.95	**0.99**	0.89	0.87	0.20	0.96
Beta	*Adversarial*										
	CTGAN	0.40	0.28	0.44	0.56	0.85	0.90	0.87	0.61	0.54	0.57
	Diffusion										
	TabDDPM	0.51	0.49	**0.61**	0.73	0.80	0.25	0.93	0.70	0.00	0.38
	TabSyn	0.49	0.50	0.42	0.58	0.71	0.85	**0.95**	0.69	**0.40**	**0.63**
	Pretrained causal sequence models										
	GReaT	**0.91**	0.61	0.43	**0.91**	**0.89**	0.92	0.93	**0.78**	**0.84**	**0.82**
	Tabby	0.83	0.65	0.43	0.86	0.86	0.92	0.92	0.75	0.72	0.78
	FAITH	0.52	**0.82**	0.42	0.64	0.75	**0.93**	0.86	0.65	0.23	0.62

FAITH has its strongest joint precision and recall on Allstate and HBA1C, but no method dominates support coverage across all datasets. In particular, the News results show that a high score on one coverage component need not imply useful predictive transfer. These measurements therefore complement, rather than replace, the primary MLE results.

E.3 Regression error metrics

The main paper reports R^2 because it measures predictive structure relative to a constant predictor. Table 8 adds scale-dependent MAE and RMSE for the four regression datasets. Classification macro-F1 is not repeated because it is already the primary metric in Table 1.

Table 8: Scale-dependent regression errors (↓).

Metric	Method	Allstate	Beijing	House	News
MAE	*Adversarial*				
	CTGAN	2263.05	60.32	0.69	**3504.17**
	Diffusion				
	TabDDPM	1372.95	42.42	**0.35**	210549.46
	TabSyn	1303.86	44.58	0.36	6455.79
	Pretrained causal sequence models				
	GReaT	1288.27	**40.53**	532.51	6651.36
	Tabby	1282.12	41.03	426.10	6452.05
	FAITH	**1257.54**	43.03	0.45	5654.79
RMSE	*Adversarial*				
	CTGAN	3072.38	86.52	0.88	**11878.51**
	Diffusion				
	TabDDPM	2369.40	64.79	0.53	259925.50
	TabSyn	2177.74	68.34	**0.53**	20083.05
	Pretrained causal sequence models				
	GReaT	2259.21	64.87	31642.35	16167.09
	Tabby	2225.50	64.59	25314.01	15752.19
	FAITH	**2090.65**	**63.47**	0.63	14092.60

FAITH obtains the lowest Allstate error and the lowest Beijing RMSE, which agrees with the R^2 comparison in the main paper. House also illustrates why multiple regression metrics are necessary: catastrophic GReaT errors are visible directly in MAE and RMSE, while the remaining methods occupy a much narrower range.

E.4 Membership-inference diagnostics

The composite screen in Table 4 summarizes whether all distance and matching checks pass. Table 9 provides the attack-level measurements that support the separate disclosure analysis. An AUC near 0.50 indicates chance-level discrimination; TPR is reported at a target FPR of at most 5%.

Table 9: Membership-inference attacks. TPR is measured at a target FPR of at most 5%.

Method	Attack metric	Adult	Allstate	Beijing	Default	Diabetes	HBA1C	House	Magic	News	Shoppers
Diffusion											
TabSyn	Distance AUC	0.50	0.50	0.51	0.50	0.56	0.51	0.51	0.51	0.51	0.50
	Distance TPR	0.05	0.05	0.05	0.05	0.06	0.05	0.05	0.05	0.06	0.04
	DOMIAS AUC	0.50	0.50	0.51	0.49	0.57	0.51	0.51	0.50	0.50	0.50
	DOMIAS TPR	0.05	0.05	0.05	0.05	0.18	0.05	0.06	0.05	0.05	0.05
Pretrained causal sequence models											
GReaT	Distance AUC	0.50	0.50	0.54	0.50	0.54	0.51	0.50	0.51	0.51	0.50
	Distance TPR	0.05	0.05	0.07	0.05	0.08	0.05	0.05	0.06	0.06	0.04
	DOMIAS AUC	0.50	0.51	0.50	0.50	0.50	0.51	0.50	0.49	0.51	0.49
	DOMIAS TPR	0.05	0.05	0.05	0.04	0.05	0.05	0.05	0.04	0.06	0.05
Tabby	Distance AUC	0.50	0.50	0.53	0.50	0.55	0.51	0.50	0.51	0.51	0.50
	Distance TPR	0.05	0.05	0.06	0.05	0.07	0.05	0.05	0.06	0.06	0.04
	DOMIAS AUC	0.50	0.51	0.50	0.50	0.50	0.51	0.50	0.49	0.51	0.49
	DOMIAS TPR	0.05	0.05	0.05	0.04	0.05	0.05	0.05	0.04	0.06	0.05
FAITH	Distance AUC	0.50	0.50	0.53	0.50	0.56	0.51	0.50	0.51	0.51	0.51
	Distance TPR	0.05	0.05	0.06	0.06	0.04	0.05	0.05	0.06	0.06	0.04
	DOMIAS AUC	0.51	0.50	0.50	0.50	0.51	0.50	0.50	0.49	0.51	0.50
	DOMIAS TPR	0.05	0.05	0.05	0.05	0.07	0.05	0.05	0.05	0.06	0.05

Across FAITH, TabSyn, and GReaT, most AUC values remain close to 0.50 and most TPR values remain near the target false-positive rate. The clearest exception is the TabSyn DOMIAS result on Diabetes. These attacks

therefore support the aggregate disclosure discussion in the main paper while remaining distinct from a formal differential-privacy guarantee.