GDIFFTABLE: GRAPH DIFFUSION FOR TABULAR DATA SYNTHESIS

Anonymous authors
Paper under double-blind review

ABSTRACT

Tabular synthesis requires preserving dependencies among heterogeneous columns, yet diffusion progressively corrupts the row-level evidence for those dependencies. We introduce GDiffTable, a mixed-type tabular diffusion model that represents columns as graph nodes and jointly learns a dataset-level weighted adjacency with a time-conditioned message-passing denoiser. The adjacency is shared across diffusion timesteps to provide a stable scaffold for cross-column coupling, while the exchanged messages adapt to the current noisy state and time embedding. We further introduce correctable graph initialization: an offline schema-derived relation prior determines only the initial adjacency, and all off-diagonal weights remain trainable under the denoising objective. Consequently, initially absent relations can emerge, selected relations can be weakened, and the formulation remains compatible with non-LLM initializers. Across seven benchmarks, GDiffTable reports the highest aggregate point estimates for Shape (98.74%) and α-Precision (97.99%), together with the smallest average downstream gap to real data (4.50%) among the evaluated synthesizers. Adult and News comparisons indicate that cross-column communication improves over node-wise denoising, learned non-uniform routing improves over fixed uniform mixing, and refinement improves over a frozen prior. The full and data-only graph variants attain similar final quality; in the reported runs, LLM initialization reduces their early-stopping times by 38.5% and 59.3%. These results support graph-structured diffusion denoising while positioning the LLM prior as an optional optimization warm start.

1 INTRODUCTION

High-quality tabular synthesis requires modeling a joint distribution rather than a collection of column-wise marginals. A generated value can be plausible in isolation yet inconsistent with the remaining values in its row, and such violations of cross-column dependence directly affect both statistical fidelity and downstream utility. Diffusion models have become competitive tabular synthesizers by accommodating heterogeneous numerical and categorical corruption processes Kotelnikov et al. (2023); Lee et al. (2023); Zhang et al. (2024); Shi et al. (2025). Their reverse process, however, must recover coherent feature relationships from states in which the evidence for those relationships has itself been corrupted.

This creates a diffusion-specific structural challenge. As independent Gaussian noise increases, cross-column dependence becomes progressively less reliable relative to marginal noise. Absorbing-state corruption creates an analogous effect for categorical columns because fewer clean column pairs remain jointly observable at high masking rates. Yet the reverse denoiser must still coordinate predictions across columns throughout the trajectory. The problem is not to infer a new relation graph from every noisy state, but to equip the reverse process with a stable relational scaffold while allowing its use of that scaffold to adapt to the current noise level.

An explicit column graph provides such a scaffold: its adjacency controls the strength of information exchange between columns, and a time-conditioned message-passing network determines what information is useful at a particular diffusion step. Previous graph-based tabular generators learn relational structures for synthesis Liu et al. (2023); Lee & Min (2025). A remaining difficulty is how to initialize and adapt the graph for diffusion. The denoising objective supervises reconstructed values rather than individual edges, so it provides only indirect guidance about routing structure,

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.

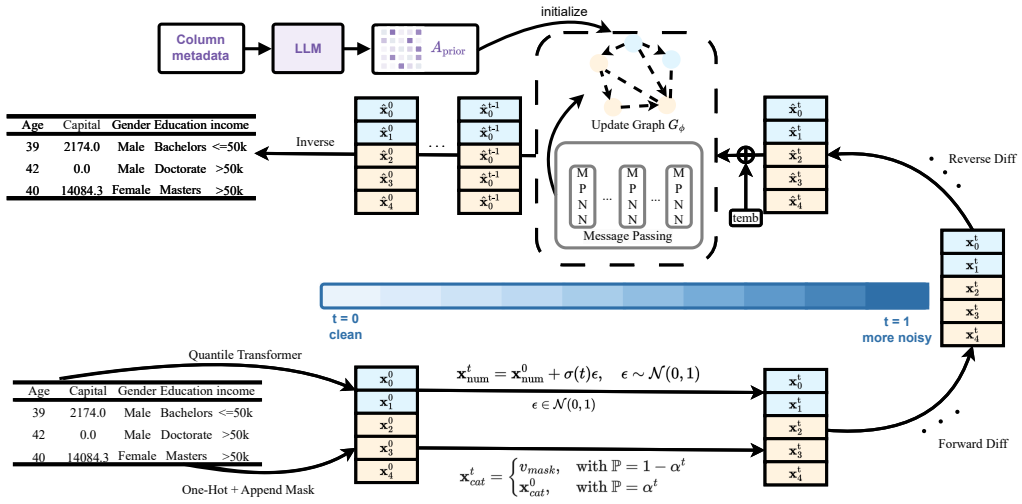


Figure 1: Overview of GDiffTable. An offline LLM uses schema descriptors, task information, and training-split marginal summaries to select directed relations, which are binarized into $\mathbf{A}_{\text{prior}}$. After initialization, all off-diagonal adjacency weights remain trainable under the denoising objective. The learned adjacency is shared across timesteps while messages are time-conditioned; it is fixed during sampling, and the LLM is not invoked during training or sampling.

particularly when relational evidence is weak. Conversely, fixing a graph from external knowledge can preserve plausible but dataset-inappropriate relations. We therefore treat the graph as a trainable denoising inductive bias rather than a recovered causal or ground-truth dependency graph.

Graph initialization is separable from graph-based denoising. A non-informative initializer can be optimized solely through the denoising objective, while schema metadata offers a practical warm start whose signal is unaffected by diffusion corruption. Column names, types, units, and descriptions can indicate plausible relations even when current feature values are heavily noised. Prior work such as SPADA demonstrates that an LLM can convert such metadata into a dependency graph for conditional tabular generation Yang et al. (2025). We use an offline LLM to select an initial edge set and convert it to initial weights through a fixed mapping. The graph denoiser itself does not require an LLM, and the present experiments do not establish that this initializer is uniquely superior to statistical, domain-informed, or other informative alternatives.

We instantiate this design in **GDiffTable**, a graph-structured diffusion model for mixed-type tabular synthesis. Its core component is a trainable, dataset-level adjacency matrix coupled to a time-conditioned message-passing denoiser. In the reported configuration, an offline LLM converts schema descriptors and training-split marginal summaries into a sparse pairwise warm start without receiving row-level records. The initialized adjacency subsequently undergoes *denoising-driven refinement*: every candidate edge is optimized jointly with the diffusion denoiser under the generation objective. Numerical and categorical columns undergo Gaussian and absorbing-state corruption, respectively, and are encoded as graph nodes at every reverse step. The adjacency is shared across diffusion time because it represents the routing strengths for cross-column coupling, whereas the messages are conditioned on the noisy node states and time embedding because the reliability and content of relational evidence change with the noise level. Thus, the adjacency controls how strongly information is exchanged between columns, while the time-conditioned denoiser determines the content of the exchanged messages. The LLM is never called during diffusion training or sampling.

We evaluate GDiffTable on seven real-world datasets using downstream predictive utility, statistical fidelity, manifold precision and recall, a near-duplicate indicator, and runtime. The benchmark results show strong aggregate Shape, Trend, and α-precision together with competitive downstream utility, while also revealing that SPADA attains higher β-recall. In the Adult and News point-estimate

comparisons, uniform dense communication improves over node-wise denoising, learned non-uniform routing improves further, and refining the initialized graph outperforms freezing it. The full and data-only trainable graphs attain similar final point estimates; in the reported runs, however, LLM initialization reduces time to the common early-stopping criterion by 38.5% on Adult and 59.3% on News. We therefore position the LLM-derived prior as a practical optimization warm start, not as a uniquely necessary source of final generation quality. Synthetic data can facilitate access and augmentation, but these fidelity measures and the reported near-duplicate indicator do not constitute a formal privacy guarantee Assefa et al. (2020); Hernandez et al. (2022).

Our contributions are twofold:

- We introduce GDiffTable, a mixed-type tabular diffusion model that represents columns as graph nodes and jointly learns a dataset-level weighted adjacency with a time-conditioned message-passing denoiser. The adjacency is shared across diffusion timesteps to provide a stable scaffold for cross-column coupling, while the exchanged messages adapt to the current noisy state and time embedding.
- We develop a correctable graph-initialization scheme in which an offline schema-derived relation prior serves only as a warm start. All off-diagonal adjacency weights remain trainable under the denoising objective, allowing initially absent relations to emerge and initially selected relations to be weakened. The graph-denoising formulation therefore does not require an LLM and is compatible with data-only, statistical, or domain-informed initializers.

2 Related Work

2.1 Tabular Data Synthesizers

Early tabular synthesis adapted VAEs and GANs Kingma & Welling (2014); Goodfellow et al. (2014) to heterogeneous columns. Representative designs include TableGAN Park et al. (2018), CTGAN and TVAE Xu et al. (2019), CTAB-GAN Zhao et al. (2021), and OCT-GAN Kim et al. (2021). These models support augmentation and controlled sharing, but fidelity alone is not a privacy guarantee Assefa et al. (2020); Zheng & Charoenphakdee (2022). Strong discriminative tabular baselines also motivate joint feature contextualization Gorishniy et al. (2021).

Relational models draw on graph convolution, graph attention, and order-independent aggregation Kipf & Welling (2017); Veličković et al. (2018); Zaheer et al. (2017); connectivity can also be learned jointly with GNN parameters Franceschi et al. (2019). For tabular synthesis, GOGGLE Liu et al. (2023) learns a feature graph within a VAE, CG-TGAN Lee & Min (2025) uses a graph-based GAN, and hypergraph models capture higher-order interactions Ouyang et al. (2025). SPADA Yang et al. (2025) uses an LLM-derived dependency graph for conditional generation. In contrast, we jointly optimize a dataset-level column-routing graph with an unconditional diffusion denoiser and treat the schema-derived graph only as correctable initialization rather than a fixed conditioning structure.

Diffusion generation builds on DDPM Ho et al. (2020) and later high-fidelity extensions Dhariwal & Nichol (2021). For heterogeneous tables, TabDDPM Kotelnikov et al. (2023) separates numerical and categorical processes, CoDi Lee et al. (2023) uses co-evolving models, and TabSyn Zhang et al. (2024) diffuses in a continuous latent space. CDTD Mueller et al. (2025) uses continuous mixed-type diffusion, Diffuse Everything Rojas et al. (2025) supports arbitrary state spaces, and TabDiff Shi et al. (2025) learns feature-wise schedules.

3 Methodology

We describe the mixed-type diffusion model and its time-conditioned graph denoiser, followed by correctable initialization and joint optimization of the column graph. The graph formulation permits different initializers. In the reported configuration, an offline LLM supplies a practical warm start for the adjacency matrix; it is not part of diffusion training or sampling.

3.1 Problem Formulation

Let $\mathcal{T} = \{\mathbf{x}^{(i)}\}_{i=1}^{n}$ denote a mixed-type tabular dataset after deterministic preprocessing fitted on the training split, with M_{num} numerical and M_{cat} categorical columns, where $M = M_{\text{num}} + M_{\text{cat}}$. A clean row in this preprocessed tabular space is $\mathbf{x}^0 = [\mathbf{x}^0_{\text{num}}, \mathbf{x}^0_{\text{cat}}]$, with $\mathbf{x}^0_{\text{num}} \in \mathbb{R}^{M_{\text{num}}}$ and $\mathbf{x}^0_{\text{cat}} \in \mathcal{S}_1 \times \cdots \times \mathcal{S}_{M_{\text{cat}}}$. For categorical column i, $\mathcal{S}_i = \{1, \ldots, C_i\}$ is augmented with an absorbing state v_{mask}, giving $\tilde{\mathcal{S}}_i = \mathcal{S}_i \cup \{v_{\text{mask}}\}$. We denote the column metadata by $\mathcal{M} = \{s_1, \ldots, s_M\}$, where s_i contains a column name and type, and may contain a unit or a short description. The metadata are associated with columns, not individual rows.

Our goal is to learn $p_\theta(\mathbf{x})$ from $\mathcal{T}$ and generate realistic, diverse samples $\hat{\mathbf{x}} \sim p_\theta$. Mixed feature types require different corruption processes, while useful predictions of a corrupted column may depend on other columns. The central structural question is which cross-column messages should be available to the denoiser.

We represent column interactions by a weighted adjacency matrix $\mathbf{A} \in \mathbb{R}^{M \times M}$, with A_{ij} weighting a message from source column i to target column j and $A_{ii} = 0$. Reciprocal interactions can have different weights. The adjacency controls the strength of cross-column information exchange in the denoiser; its edges are not asserted to be causal effects or ground-truth dependencies.

The graph is a relational inductive bias for denoising Battaglia et al. (2018). A reconstruction loss constrains the generated values but does not directly supervise individual edges. We instantiate $\mathbf{A}$ with a schema-derived warm start and subsequently optimize its weights with the denoiser. This initialization is an implementation choice, not a requirement of the graph denoiser, and the optimized adjacency is not interpreted as a recovered dependency graph.

Dependency attenuation under corruption. The role of a shared relational scaffold can be seen directly from the forward processes. For numerical columns, let $x^i_t = x^i_0 + \sigma_t \epsilon_i$, where the noise variables are independent of the clean data and of one another, with zero mean and unit variance. For two distinct columns $i \neq j$,

$$\text{Corr}(x^i_t, x^j_t) = \frac{\text{Cov}(x^i_0, x^j_0)}{\sqrt{(\text{Var}(x^i_0) + \sigma_t^2)(\text{Var}(x^j_0) + \sigma_t^2)}}. \tag{1}$$

Although the clean cross-covariance remains in expectation, its normalized contribution decreases as the marginal noise variance grows. For categorical columns i and j, the probability that both remain observable under independent absorbing-state corruption is $\alpha^t_i \alpha^t_j$, which also decreases with the masking level. Thus, the current noisy state provides progressively weaker evidence about clean cross-column dependence, even though the reverse model must continue to coordinate its predictions.

Shared-graph assumption. We use one dataset-level adjacency matrix across diffusion timesteps. It is intended as a relational scaffold for the clean-data distribution, not as the correlation graph observable in every noisy state. In particular, heavy corruption can obscure empirical cross-column dependence. Sharing $\mathbf{A}$ across t is a modeling choice; the messages still change with the noisy node states and the time embedding. We do not infer a separate graph from each $\mathbf{x}^t$.

3.2 Model Overview

As illustrated in Fig. 1, GDiffTable has an offline graph-initialization stage followed by a feature encoder, a graph denoiser, and numerical and categorical prediction heads. The initialized adjacency remains trainable during diffusion training and is fixed only after training for sampling.

The encoder embeds each column into a shared latent space, producing a set of node embeddings that serve as the input to the graph backbone. Let $\mathcal{Q}$ denote the feature-wise quantile transformation Pedregosa et al. (2011) fitted on the numerical columns of the training split. Numerical diffusion is performed on $\mathcal{Q}(\mathbf{x}_{\text{num,raw}})$, while categorical states are represented as one-hot vectors before encoding. The reconstructor g decodes denoised node embeddings through two heads, i.e., a numerical head that predicts clean values in the quantile-transformed space and a categorical head that predicts class logits. After sampling, $\mathcal{Q}^{-1}$ maps the generated numerical values back to their original units.

The same learned $\mathbf{A}$ is used at every diffusion timestep; "shared" refers to diffusion time, not to frozen parameters during training. At each timestep, the encoder produces noisy column-node representations and the denoiser exchanges weighted messages along $\mathbf{A}$. Time conditioning changes the message content without introducing a separate adjacency matrix for every noise level.

The graph denoiser is integrated with EDM-style Gaussian corruption for numerical features and absorbing-state corruption for categorical features, as detailed below. It produces clean-value predictions used by reverse updates in the preprocessed tabular space, before the final inverse transformation to raw numerical units.

3.3 Graph Initialization and Denoising-Driven Refinement

For the reported GDiffTable configuration, we use one practical schema-derived warm start. For each dataset schema $\mathcal{M}$, we run an offline extraction stage with a local Qwen3-1.7B model to obtain a set $\mathcal{R}_{\mathrm{LLM}}$ of plausible directed *pairwise* column relations, following the general use of schema semantics in SPADA Yang et al. (2025). The prompt contains column names, types, descriptions, and training-split marginal summaries—numerical ranges and up to 12 categorical values—computed locally from the first 5,000 training rows; no row-level records are provided to the model. The LLM determines only which directed pairs belong to the initial graph; it does not output continuous edge weights. We convert its relation set to a binary initialization by the fixed mapping

$$(A_{\mathrm{prior}})_{ij} = \mathbb{I}[(i,j) \in \mathcal{R}_{\mathrm{LLM}}], \qquad (A_{\mathrm{prior}})_{ii} = 0, \tag{2}$$

so selected relations start at one and all other off-diagonal relations start at zero. Schemas are processed in chunks of at most 14 target columns, and the resulting $\mathbf{A}_{\mathrm{prior}} \in \{0,1\}^{M\times M}$ is cached. The extraction stage is run once per schema, although chunking and retries may require multiple generation calls. No LLM call is made during diffusion training or sampling. Appendix A.3 gives the complete prompt and extraction protocol. This choice does not imply that an LLM is uniquely optimal: statistical, domain-informed, uniform, or random initializers can be substituted without changing the denoising architecture.

We initialize the trainable off-diagonal parameters by $A_{ij}^{(0)} = (A_{\mathrm{prior}})_{ij}$ for $i \neq j$. Although Eq. (2) is binary, it determines only the starting values: it is neither a ground-truth target nor a permanent binary mask. All off-diagonal coefficients become continuous trainable weights, so the denoising objective can weaken an LLM-selected relation or increase a coefficient that was initialized at zero. We call this joint optimization of the initialized adjacency and denoiser *denoising-driven refinement*. During training, gradients of the total diffusion objective in Eq. (10) update both the denoiser parameters and $\mathbf{A}$. The full model and data-only graph use the same adjacency parameterization and sparsity penalty and differ only in their starting values; the fixed-prior control instead freezes $\mathbf{A}$ after initialization. Consequently, any benefit attributed to the LLM in our current evaluation concerns optimization from this starting point rather than a change to the denoising objective or model capacity.

3.4 Mixed-Type Diffusion Framework

Numerical columns are corrupted in the quantile-transformed feature space with EDM-style Gaussian noise, while categorical columns undergo absorbing-state masking on the same diffusion time axis. Together these variables form the preprocessed tabular state $\mathbf{x}^t$. The denoiser predicts clean numerical values and categorical logits from the complete mixed-type state, and these predictions are used by the reverse ODE and unmasking transitions, respectively. Diffusion acts on $\mathbf{x}^t$ before column encoding rather than on the latent node representation $\mathbf{H}^t$. The schedules and reverse updates are detailed in Appendix A.1.

3.5 Graph-Based Denoising

At each diffusion time t, the noisy preprocessed sample $\mathbf{x}^t$ is processed by the feature encoder τ, which consists of a set of feature-wise mapping layers. Concretely, τ is implemented as lightweight per-column MLP blocks (in practice, primarily linear projections), and it projects each individual column into a shared latent space as follows:

$$\mathbf{H}^t = \tau\big(\mathbf{x}^t\big) = [\mathbf{h}_1^t, \ldots, \mathbf{h}_M^t]^\top, \tag{3}$$

where each column i is processed by its own mapping function and represented by a d-dimensional node embedding $\mathbf{h}_i^t$ in the same dimension. These embeddings form the nodes of a latent feature graph and serve as intermediate representations on which the denoising network operates. Crucially, corruption is applied only to the preprocessed tabular state $\mathbf{x}^t$; the embeddings $\mathbf{H}^t$ are deterministic transformations of that noisy state via the column-specific mapping layers and the encoder.

We encode the diffusion timestep t as a d-dimensional embedding $\mathbf{e}_t$ using sinusoidal encoding Vaswani et al. (2017). The embedding enters each message-passing layer, allowing the message function to depend on the noise level.

At each reverse step, the current noisy data $\mathbf{x}^t$ are encoded afresh as $\mathbf{H}^t = \tau(\mathbf{x}^t)$. The graph denoiser Φ_θ computes an intermediate representation

$$\mathbf{Z}^t = \Phi_\theta\big(\mathbf{H}^t, \mathbf{e}_t, \mathbf{A}\big), \tag{4}$$

where $\mathbf{A}$ is the dataset-level adjacency shared across diffusion timesteps. This representation is decoded into clean-data predictions:

$$\big[\hat{\mathbf{x}}^0_{\text{num}}, \mathbf{L}_{\text{cat}}\big] = g\big(\mathbf{Z}^t\big), \tag{5}$$

where $\hat{\mathbf{x}}^0_{\text{num}}$ denotes predicted clean numerical values in the quantile-transformed space and $\mathbf{L}_{\text{cat}}$ denotes categorical logits. Similar to the encoder, g is implemented as per-column linear projections. These outputs parameterize the reverse update at the current step and are used by the sampler to construct the next preprocessed-space iterate.

Within message-passing layer l, each source node i forms a time-conditioned message by concatenating its current state with $\mathbf{e}_t$:

$$\mathbf{m}_i^{(l)} = f_{\text{msg}}^{(l)}\Big([\mathbf{h}_i^{(l-1)} \| \mathbf{e}_t]\Big), \tag{6}$$

For target node j, an order-independent operator $\oplus^{(l)}$ combines the incoming messages Battaglia et al. (2018). We consider all other columns as candidate sources, $\mathcal{N}(j) = \{i : i \neq j\}$, and weight their contributions by the corresponding trainable coefficients:

$$\mathbf{h}_{\mathcal{N}(j)}^{(l)} = \oplus^{(l)}\Big\{ A_{i,j}\mathbf{m}_i^{(l)} \;\Big|\; i \in \mathcal{N}(j) \Big\}, \tag{7}$$

The embedding of node j is updated by $f_{\text{upd}}^{(l)}$, which jointly considers the aggregated neighborhood representation and the previous state $\mathbf{h}_j^{(l-1)}$:

$$\mathbf{h}_j^{(l)} = f_{\text{upd}}^{(l)}\Big(\mathbf{h}_{\mathcal{N}(j)}^{(l)}, \mathbf{h}_j^{(l-1)}\Big). \tag{8}$$

Here $f_{\text{msg}}^{(l)}$ transforms a source representation, $\oplus^{(l)}$ aggregates weighted messages, and $f_{\text{upd}}^{(l)}$ updates the target node. Even when $(A_{\text{prior}})_{ij} = 0$, its trainable coefficient A_{ij} can receive gradients because no prior-derived hard mask removes the candidate edge. The realized message depends on both the fixed-for-this-step coefficient A_{ij} and the time-dependent node state and embedding; sharing $\mathbf{A}$ does not force identical messages at all noise levels. We regularize the weights as described below, without interpreting an edge weight alone as a verified dependency.

3.6 Loss Function

Let $\mathcal{D}_{\text{num}}$ and $\mathcal{D}_{\text{cat}}$ index numerical and categorical columns, respectively. Given the prediction heads above, the unified training objective is

$$\begin{aligned}\mathcal{L} = \mathbb{E}_{t,\,\mathbf{x}^0,\,\epsilon}\Bigg[&\sum_{i\in\mathcal{D}_{\text{num}}} w_t \left\|\hat{x}_i^0 - x_i^0\right\|^2 \\ &+ \sum_{i\in\mathcal{D}_{\text{cat}}} \lambda_t \,\text{CE}(\mathbf{L}_i, x_i^0)\Bigg],\end{aligned} \tag{9}$$

where the expectation is taken over clean preprocessed samples $\mathbf{x}^0$, diffusion times t, and the corruption applied in the preprocessed tabular space. The weights w_t and λ_t are time-dependent for

numerical and categorical losses, respectively. In this formulation, diffusion acts on $\mathbf{x}$ after numerical quantile transformation but before column encoding, while $\mathbf{H}^t$ is used solely as an intermediate node representation for graph-based denoising. Generated numerical samples are returned to the raw feature space only after the reverse trajectory through $\mathcal{Q}^{-1}$.

We encourage a parsimonious weighted graph by adding an L_1 penalty Tibshirani (1996) to Eq. (9):

$$\mathcal{L}_{\text{total}} = \mathcal{L} + \lambda \|\mathbf{A}\|_1, \tag{10}$$

where λ controls regularization strength. The penalty can shrink weak coefficients but does not, by itself, guarantee exact zero edges, recovery of true dependencies, or reduced dense-matrix computation. The selected prior affects initialization, not the training objective: $\mathcal{L}_{\text{total}}$ has no separate term that forces $\mathbf{A}$ to stay close to $\mathbf{A}_{\text{prior}}$.

Algorithm 1 in the appendix summarizes the training and sampling procedures of GDiffTable.

4 Experiments

We organize the evaluation around two goals. First, we assess whether GDiffTable generates useful and faithful tables across heterogeneous benchmarks. Second, we use the Adult and News ablation to examine cross-column communication, the practical effect of the selected warm start, and denoising-driven adjacency updates.

4.1 Experimental Setup

We evaluate on seven tabular benchmarks—Adult, Default, News, Insurance, Magic, Nursery, and California—covering numerical, categorical, and mixed-type data as well as classification and regression tasks. Baselines include representative diffusion, VAE, GAN, graph-based, and LLM-based synthesizers under consistent real-data splits. Dataset statistics, preprocessing, hyperparameters, and implementation details appear in Appendix A.6. All experiments were conducted on a single NVIDIA RTX 6000 GPU using CUDA 12.1, Python 3.10, and PyTorch 2.2.2.

We evaluate downstream utility with Train on Synthetic, Test on Real (TSTR) Xu et al. (2019); Kotelnikov et al. (2023): XGBoost models are trained only on synthetic records and evaluated on a held-out real test set. Classification tasks use AUC and regression tasks use RMSE. Distributional quality is assessed with Shape and Trend, while α-precision and β-recall characterize manifold fidelity and coverage. DCR is reported only as a distance-based near-duplicate indicator and does not constitute a formal privacy guarantee. Complete metric definitions are given in Appendix A.4.

4.2 Overall Generation Quality

4.2.1 Downstream Predictive Utility

Table 1 shows that GDiffTable leads the evaluated synthesizers on Default, Insurance, and News, ranks second on Nursery, Adult, and Magic, and trails the strongest diffusion baselines on California. This supports competitive utility across heterogeneous tables rather than uniform dominance.

4.2.2 Distributional Fidelity and Coverage

Table 2 shows that GDiffTable reports the highest aggregate point estimates for Shape and α-precision, a near-best Trend value, and the smallest reported downstream gap to REAL. SPADA's higher Trend and β-recall reveal a fidelity–coverage trade-off. GDiffTable's DCR is close to the ideal 50% balance, but this diagnostic is not a formal privacy guarantee. GReaT's Average Gap is an estimate because News is OOM, as detailed below the table.

4.3 Mechanistic Analysis of Relational Denoising

The core ablation asks whether cross-column communication helps (**RQ1**), learned non-uniform routing improves over global mixing (**RQ2**), an LLM warm start changes full-data optimization (**RQ3**), and refinement improves over a fixed prior (**RQ4**).

Table 1: TSTR performance. Synthetic methods report mean ± standard deviation over five runs; REAL (red) is a point estimate. Bold and underlined values are the best and second-best synthetic results. OOM indicates that News exceeds GReaT's available memory budget.

	Method	Cont.	Disc.	Mixed				
		California ↓	Nursery ↑	Adult ↑	Default ↑	Insurance ↓	Magic ↑	News ↓
	REAL	0.227	0.984	0.925	0.771	0.469	0.946	0.843
	SMOTE	0.253±0.006	0.738±0.004	0.899±0.007	0.729±0.009	0.632±0.011	0.867±0.008	0.991±0.036
Diffusion	TabDiff	0.253±0.003	0.892±0.004	0.901±0.003	0.758±0.005	0.501±0.009	0.932±0.004	0.884±0.018
Diffusion	TabDDPM	**0.248**±0.005	0.900±0.003	0.911±0.001	0.759±0.004	0.529±0.011	0.914±0.005	4.635±1.870
Diffusion	STaSy	0.257±0.006	0.897±0.003	0.904±0.001	0.749±0.006	0.652±0.014	0.924±0.003	0.934±0.012
Diffusion	CoDi	0.250±0.003	0.893±0.004	0.829±0.006	0.498±0.006	0.723±0.021	0.930±0.003	0.999±0.018
Diffusion	TabSyn	0.256±0.005	0.894±0.003	0.911±0.002	0.760±0.004	0.490±0.008	0.934±0.002	0.863±0.017
LLM	GReaT	0.306±0.008	0.099±0.012	0.892±0.003	0.712±0.006	0.942±0.013	0.913±0.008	OOM
LLM	SPADA	0.369±0.007	**0.984**±0.002	**0.925**±0.003	0.732±0.006	0.618±0.012	**0.936**±0.004	2.316±0.564
VAE	GOGGLE	0.559±0.012	0.853±0.009	0.778±0.012	0.585±0.005	1.377±0.025	0.655±0.024	1.091±0.018
VAE	TVAE	0.386±0.007	0.842±0.005	0.846±0.004	0.745±0.005	0.936±0.011	0.912±0.003	0.979±0.016
GAN	CTGAN	0.354±0.009	0.891±0.006	0.875±0.009	0.736±0.005	0.950±0.019	0.875±0.006	0.895±0.016
GAN	CG-TGAN	0.576±0.011	0.897±0.005	0.882±0.007	0.747±0.006	0.924±0.017	0.876±0.005	0.905±0.015
Ours	**GDiffTable**	0.264±0.004	0.903±0.003	0.913±0.002	**0.764**±0.004	**0.478**±0.009	0.935±0.003	**0.857**±0.014

Table 2: Aggregate evaluation across seven datasets. Shape, Trend, α-Precision, β-Recall, and DCR are unweighted macro-averages of their dataset-level scores. DCR follows the equal-size synthetic-vs.-holdout protocol (50% is ideal). Average Gap is the mean absolute relative downstream gap to REAL across the same datasets.

	Method	Statistical Fidelity				Privacy Diagnostic	Utility
		Shape ↑ %	Trend ↑ %	α-Precision ↑ %	β-Recall ↑ %	DCR → 50 %	Average Gap ↓ %
Diffusion	STaSy	85.70	86.50	78.20	44.40	51.27	11.33
Diffusion	TabDDPM	85.33	83.18	97.52	51.40	65.19	69.55
Diffusion	CoDi	81.20	85.60	90.50	48.28	52.15	19.93
Diffusion	TabDiff	98.48	97.86	97.76	53.80	52.62	5.46
Diffusion	TabSyn	98.52	97.82	97.80	54.90	51.85	4.71
LLM	GReaT	76.61	53.55	70.92	20.70	**50.73**	36.63†
LLM	SPADA	98.58	**98.88**	95.24	**68.35**	96.23	39.31
VAE	TVAE	84.20	82.60	81.80	29.10	51.36	30.81
VAE	GOGGLE	82.80	70.90	78.20	9.20	51.27	64.77
GAN	CTGAN	82.10	81.60	82.30	30.90	51.28	27.37
GAN	CG-TGAN	83.60	82.70	83.10	33.20	52.39	40.30
Ours	**GDiffTable**	**98.74**	97.92	**97.99**	56.10	51.95	**4.50**

Average Gap for method m is computed as $\frac{100}{7}\sum_{d=1}^{7}\frac{|s_{m,d}-r_d|}{|r_d|}$, where $s_{m,d}$ and r_d are the synthetic-data and REAL downstream scores on dataset d, respectively. For GReaT (†), the News result is unavailable because the large number of columns causes an out-of-memory failure. We impute this entry only for the aggregate using the median News score (0.979) of the other 11 baselines in Table 1, excluding REAL, GReaT, and GDiffTable. Thus, its Average Gap is an estimate rather than a fully observed result.

4.3.1 Core Ablation: Graph, Prior, and Denoising-Driven Refinement

We compare five variants on Adult and News. No Graph uses a node-wise denoiser; Uniform Dense Graph fixes equal weights on every off-diagonal edge; Data-only Graph learns adjacency from a non-informative initialization; Fixed LLM Prior freezes the schema-derived graph; and GDiffTable refines

Table 3: Core ablation on Adult and News. Entries are measured point estimates; parentheses give wall-clock time to the common early-stopping criterion. Shape and Trend are percentages.

Variant	Adult AUC ↑	News RMSE ↓	Avg. Shape ↑	Avg. Trend ↑
No Graph	0.634 (20.4 min)	1.357 (39.1 min)	99.04	53.61
Uniform Dense Graph	0.813 (41.6 min)	0.989 (157.6 min)	95.53	84.72
Data-only Graph	0.911 (39.2 min)	0.861 (163.2 min)	98.83	97.21
Fixed LLM Prior	0.864 (25.7 min)	0.985 (91.5 min)	97.32	95.26
GDiffTable	0.913 (24.1 min)	0.857 (66.4 min)	98.84	97.54

that initialization. All variants share the diffusion setup, all graph variants share the message-passing backbone, and the two trainable graphs share their parameterization and sparsity penalty.

Table 3 shows that uniform dense communication improves utility and Trend over No Graph, and learning non-uniform routing improves further. No Graph nevertheless has the highest Shape, confirming that marginal fidelity alone does not ensure cross-column consistency. GDiffTable also outperforms Fixed LLM Prior, supporting denoising-driven refinement.

GDiffTable and Data-only Graph have similar final point estimates and identical per-step computation, differing only in initialization. In the reported runs, GDiffTable reaches the common early-stopping criterion 38.5% sooner on Adult and 59.3% sooner on News. Thus, the LLM prior acts mainly as an optimization warm start; this comparison does not establish superiority to every informative initializer.

5 Conclusion

We presented GDiffTable, a mixed-type tabular diffusion model motivated by the observation that forward corruption progressively weakens the row-level evidence for cross-column dependence. Its dataset-level weighted adjacency is shared across diffusion time, while message content adapts to the current noisy state and timestep. We additionally introduced a correctable initialization scheme: a schema-derived prior determines only the starting adjacency, and every off-diagonal weight remains trainable under the denoising objective. Across seven datasets, GDiffTable reports the highest aggregate point estimates for Shape and α-precision together with competitive downstream utility. Adult and News point-estimate comparisons indicate that cross-column communication helps, learned non-uniform routing improves over fixed uniform mixing, and refinement improves over a frozen prior. Full and data-only graphs have similar final point estimates, while LLM initialization reduces early-stopping time by 38.5% and 59.3% in the reported runs. We therefore view the LLM prior as a replaceable optimization warm start. The adjacency is a denoising routing structure, not a causal graph; DCR is not a formal privacy guarantee, and repeated-run timing and wider-table scaling remain for future study.

Reproducibility Statement

The model architecture, corruption processes, objective, and sampling procedure are described in the main text and Appendix A.1. Appendix A.3 documents the offline LLM model, prompt construction, schema summaries, chunking, parsing, retry, binary relation-to-initialization mapping, graph construction, and caching procedures. Decoding parameters and random seeds are versioned with the anonymous implementation and cached metadata. Dataset processing and evaluation details are provided in Appendix A.6. The anonymous implementation is available at `https://anonymous.4open.science/r/GDiffTable-131F`.

AI Use Statement

Generative AI tools were used to retrieve and discover potentially relevant literature and to assist with language editing and polishing. The authors independently verified the cited literature and reviewed all AI-assisted edits. The authors take full responsibility for the final content of this paper.

REFERENCES

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

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.

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

Peter W Battaglia, Jessica B Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, et al. Relational inductive biases, deep learning, and graph networks. *arXiv preprint arXiv:1806.01261*, 2018.

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

R. Bock et al. Uci machine learning repository: Magic gamma telescope data set. University of California, Irvine, School of Information and Computer Sciences, 2004. URL https://archive.ics.uci.edu/dataset/159/magic+gamma+telescope. Accessed: 2026-04-24.

Vadim Borisov, Kathrin Seßler, Tobias Leemann, Martin Pawelczyk, and Gjergji Kasneci. Language models are realistic tabular data generators. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=cEygmQNOeI.

Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357, 2002.

Miri Choi. Medical cost personal datasets. Kaggle, 2018. URL https://www.kaggle.com/datasets/mirichoi0218/insurance. Dataset uploaded by Miri Choi; accessed 2026-04-24.

Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/49ad23d1ec9fa4bd8d77d02681df5cfa-Abstract.html.

Kelwin Fernandes, Pedro Vinagre, Paulo Cortez, and Pedro Sernadela. Online news popularity. UCI Machine Learning Repository, 2015. URL https://doi.org/10.24432/C5NS3V. Dataset.

Luca Franceschi, Mathias Niepert, Massimiliano Pontil, and Xiao He. Learning discrete structures for graph neural networks. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 1972–1982. PMLR, 2019. URL https://proceedings.mlr.press/v97/franceschi19a.html.

Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014. URL https://proceedings.neurips.cc/paper/2014/hash/f033ed80deb0234979a61f95710dbe25-Abstract.html.

Yury Gorishniy, Ivan Rubachev, Valentin Khrulkov, and Artem Babenko. Revisiting deep learning models for tabular data. *Advances in neural information processing systems*, 34: 18932–18943, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/9d86d83f925f2149e9edb0ac3b49229c-Abstract.html.

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

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/4c5bcfec8584af0d967f1ab10179ca4b-Abstract.html.

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. In *Advances in Neural Information Processing Systems*, volume 35, pp. 26565–26577, 2022.

Jayoung Kim, Jinsung Jeon, Jaehoon Lee, Jihyeon Hyeong, and Noseong Park. Oct-gan: Neural ode-based conditional tabular gans. In *Proceedings of the Web Conference 2021*, pp. 1506–1515, 2021.

Jayoung Kim, Chaejeong Lee, and Noseong Park. Stasy: Score-based tabular data synthesis. In *11th International Conference on Learning Representations, ICLR 2023*, 2023.

Diederik P Kingma and Max Welling. Auto-encoding variational Bayes. In *International Conference on Learning Representations*, 2014. URL https://arxiv.org/abs/1312.6114.

Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=SJU4ayYgl.

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

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

Seungcheol Lee and Moohong Min. Cg-tgan: Conditional generative adversarial networks with graph neural networks for tabular data synthesizing. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 18145–18153, 2025.

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

Markus Mueller, Kathrin Gruber, and Dennis Fok. Continuous diffusion for mixed-type tabular data. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/61f4e5747b1b753cb35546b15d981f76-Paper-Conference.pdf.

Jin Ouyang, Xu Jia, Sheng Xiang, Ying Zhang, and Lu Qin. Hypergraph-driven tabular data synthesis with multi-objective optimization. In *International Conference on Document Analysis and Recognition*, pp. 451–467. Springer, 2025.

R. Kelley Pace and Ronald Barry. California housing data. StatLib Repository, Carnegie Mellon University, 1997. URL https://www.dcc.fc.up.pt/~ltorgo/Regression/cal_housing.html. Accessed: 2026-04-24.

Noseong Park, Mahmoud Mohammadi, Kshitij Gorde, Sushil Jajodia, Hongkyu Park, and Youngmin Kim. Data synthesis based on generative adversarial networks. *Proceedings of the VLDB Endowment*, 11(10):1071–1083, 2018. doi: 10.14778/3231751.3231757. URL https://www.vldb.org/pvldb/vol11/p1071-park.pdf.

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

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830, 2011.

Vladislav Rajkovic. Nursery. UCI Machine Learning Repository, 1989. URL https://doi.org/10.24432/C5P88W. Dataset.

Kevin Rojas, Yuchen Zhu, Sichen Zhu, Felix X-F. Ye, and Molei Tao. Diffuse everything: Multimodal diffusion models on arbitrary state spaces. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=AjbiIcRt6q.

Jiaxin Shi, Kehang Han, Zhe Wang, Arnaud Doucet, and Michalis Titsias. Simplified and generalized masked diffusion for discrete data. *Advances in neural information processing systems*, 37: 103131–103167, 2024.

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*, volume 2025, pp. 37353–37375, 2025.

Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 58(1):267–288, 1996.

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, 2017.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=rJXMpikCZ.

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

Shuo Yang, Zheyu Zhang, Bardh Prenkaj, and Gjergji Kasneci. Doubling your data in minutes: Ultra-fast tabular data generation via LLM-induced dependency graphs. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 10337–10358. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.emnlp-main.525. URL https://aclanthology.org/2025.emnlp-main.525/.

I-Cheng Yeh. Default of credit card clients. UCI Machine Learning Repository, 2009. URL https://doi.org/10.24432/C55S3H. Dataset.

Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Russ R Salakhutdinov, and Alexander J Smola. Deep sets. *Advances in neural information processing systems*, 30, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/f22e4747da1aa27e363d86d40ff442fe-Abstract.html.

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 *The twelfth International Conference on Learning Representations*, 2024.

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

Shuhan Zheng and Nontawat Charoenphakdee. Diffusion models for missing value imputation in tabular data. In *NeurIPS 2022 First Table Representation Workshop*, 2022.

A Supplementary Experimental Details

A.1 Mixed-Type Diffusion Details

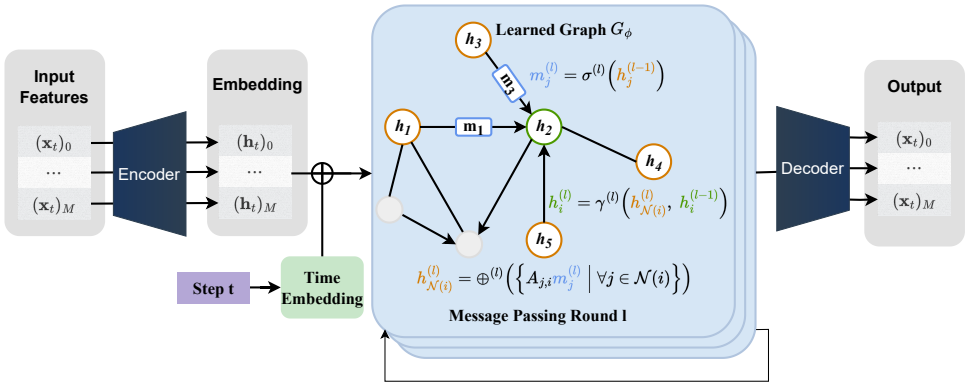


Figure 2: One GDiffTable denoising evaluation. The corrupted row $\mathbf{x}^t$ is encoded as column nodes; the shared adjacency $\mathbf{A}$ weights time-conditioned messages; and the decoder returns numerical predictions and categorical logits. The adjacency is trainable during training and fixed during sampling.

We first define a clean sample in the preprocessed tabular space as $\mathbf{x}^0 = [\mathbf{x}^0_{\text{num}}, \mathbf{x}^0_{\text{cat}}]$. Here $\mathbf{x}^0_{\text{num}} = \mathcal{Q}(\mathbf{x}^0_{\text{num,raw}})$ uses the feature-wise quantile transformation fitted on the training split, whereas $\mathbf{x}^0_{\text{cat}}$ contains categorical states. The forward process corrupts this preprocessed sample along a common time axis $t \in [0, 1]$, yielding $\mathbf{x}^t$ before column encoding.

For the numerical features, we adopt an EDM-style Gaussian perturbation Karras et al. (2022) in the quantile-transformed feature space:

$$\mathbf{x}^t_{\text{num}} = \mathbf{x}^0_{\text{num}} + \sigma(t)\boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$

where $\sigma(t)$ is the noise schedule controlling the magnitude of corruption at diffusion time t. Following the probability-flow ODE formulation for variance exploding diffusion in prior work on continuous-time tabular diffusion models Shi et al. (2025), the associated reverse-time dynamics for the numerical features can be written as

$$d\mathbf{x}^t_{\text{num}} = -\dot{\sigma}(t)\,\sigma(t)\,\nabla_{\mathbf{x}^t_{\text{num}}} \log p_t\big(\mathbf{x}^t_{\text{num}}, \mathbf{x}^t_{\text{cat}}\big)\,dt, \tag{11}$$

where $p_t(\mathbf{x}^t_{\text{num}}, \mathbf{x}^t_{\text{cat}})$ denotes the joint mixed-type distribution at time t, interpreted as a density over the numerical variables and a probability mass over the categorical states. The gradient is taken only with respect to the numerical coordinates, but it is conditioned on the complete noisy mixed-type row. This joint score is approximated by the graph-based denoiser and integrated in discretized form along the reverse trajectory.

In our implementation, we adopt the $\mathbf{x}^0$-parameterization for the denoising network. Given the complete noisy row $\mathbf{x}^t = [\mathbf{x}^t_{\text{num}}, \mathbf{x}^t_{\text{cat}}]$ and time t, the graph-based denoiser predicts the clean numerical values $\hat{\mathbf{x}}^0_{\text{num}}(\mathbf{x}^t, t)$. The score function used in Eq. (11) is then implicitly derived via the following EDM-style relationship:

$$\nabla_{\mathbf{x}^t_{\text{num}}} \log p_t\big(\mathbf{x}^t_{\text{num}}, \mathbf{x}^t_{\text{cat}}\big) \approx \frac{\hat{\mathbf{x}}^0_{\text{num}}(\mathbf{x}^t, t) - \mathbf{x}^t_{\text{num}}}{\sigma(t)^2}. \tag{12}$$

This allows the model to be trained with a simple Mean Squared Error loss (Eq. (9)) while maintaining the ODE-based sampling framework.

For each categorical feature $i \in \mathcal{D}_{\text{cat}}$, following Austin et al. (2021); Shi et al. (2024), the forward diffusion process is defined as:

$$x_{\text{cat}}^{t,i} = \begin{cases} v_{\text{mask}}, & \text{with } \mathbb{P} = 1 - \alpha_i^t \\ x_{\text{cat}}^{0,i}, & \text{with } \mathbb{P} = \alpha_i^t \end{cases}$$

where $\alpha_i^t \in [0, 1]$ is a strictly decreasing function of t.

For categorical features, the reverse transition $p_\theta(\mathbf{x}_{\text{cat}}^s \mid \mathbf{x}_{\text{cat}}^t)$ at time $s < t$ follows a categorical distribution based on the predicted clean logits $\mathbf{L}_i$. The state transition logic is defined as follows:

1. If $x_{\text{cat}}^{t,i} \neq v_{\text{mask}}$, the feature has already been de-masked, and we fix its value as $x_{\text{cat}}^{s,i} = x_{\text{cat}}^{t,i}$ to maintain consistency.
2. If $x_{\text{cat}}^{t,i} = v_{\text{mask}}$, we recover the clean state with probability

$$\beta_i^{(t,s)} = \frac{\alpha_i^s - \alpha_i^t}{1 - \alpha_i^t}, \tag{13}$$

   and sample its value from the column-wise prediction $\hat{p}_{\theta,i}(x_0^i \mid \mathbf{x}^t, t) = \text{softmax}(\mathbf{L}_i)$; otherwise, the state remains v_{mask}.

After the reverse trajectory reaches $t = 0$, the generated numerical state is mapped back to the raw feature units with $\mathcal{Q}^{-1}$, and the generated categorical states are decoded to their original labels. Thus, diffusion is performed in the preprocessed tabular space rather than in the raw numerical space or the latent node-embedding space.

A.2 Training and Sampling Procedure

Algorithm 1 Unified Training and Sampling Procedure of GDiffTable

Part 0: Offline schema initialization
Require: Column metadata $\mathcal{M}$, task descriptor κ, training-split marginal summaries $\mathcal{S}$, and pairwise LLM relation extractor q_{LLM}
Parse $\mathcal{R}_{\mathrm{LLM}} \leftarrow q_{\mathrm{LLM}}(\mathcal{M}, \kappa, \mathcal{S})$, without supplying row-level records
Construct $(A_{\mathrm{prior}})_{ij} \leftarrow \mathbb{I}[(i,j) \in \mathcal{R}_{\mathrm{LLM}}]$ for $i \neq j$; set $(A_{\mathrm{prior}})_{ii} = 0$
Cache $\mathbf{A}_{\mathrm{prior}}$ and initialize trainable $A_{ij} \leftarrow (A_{\mathrm{prior}})_{ij}$
Do not call q_{LLM} in Parts A or B

Part A: Training
Require: Preprocessed training data $\mathcal{T}$, time prior $p(t)$, schedules $\sigma(t)$ and $\{\alpha_i^t\}$
Require: Encoder τ, graph denoiser Φ_θ, reconstructor g, initialized adjacency $\mathbf{A}$
repeat
  Sample batch $\{\mathbf{x}^0\} \subset \mathcal{T}$ and time $t \sim p(t)$
  Split $\mathbf{x}^0 = [\mathbf{x}^0_{\mathrm{num}}, \mathbf{x}^0_{\mathrm{cat}}]$
  Sample $\epsilon \sim \mathcal{N}(0, \mathbf{I})$ and set
    $\mathbf{x}^t_{\mathrm{num}} = \mathbf{x}^0_{\mathrm{num}} + \sigma(t)\epsilon$
  for each categorical feature i **do**
    Set $x^{t,i}_{\mathrm{cat}} = v_{\mathrm{mask}}$ with probability $1 - \alpha_i^t$;
      otherwise keep $x^{t,i}_{\mathrm{cat}} = x^{0,i}_{\mathrm{cat}}$
  end for
  Form noisy input $\mathbf{x}^t = [\mathbf{x}^t_{\mathrm{num}}, \mathbf{x}^t_{\mathrm{cat}}]$
  Compute time embedding $\mathbf{e}_t$ from t
  Encode and denoise:
    $\mathbf{H}^t = \tau(\mathbf{x}^t)$, $\mathbf{Z}^t = \Phi_\theta(\mathbf{H}^t, \mathbf{e}_t, \mathbf{A})$
  Decode predictions:
    $[\hat{\mathbf{x}}^0_{\mathrm{num}}, \mathbf{L}_{\mathrm{cat}}] = g(\mathbf{Z}^t)$
  Compute $\mathcal{L}_{\text{total}} = \mathcal{L} + \lambda\|\mathbf{A}\|_1$ via Eq. (9) and Eq. (10)
  Update model parameters and $\mathbf{A}$ by gradient descent on $\mathcal{L}_{\text{total}}$
until convergence

Part B: Sampling
Require: Trained τ, Φ_θ, g, learned $\mathbf{A}$, fitted inverse transform $\mathcal{Q}^{-1}$, reverse grid $1 = t_0 > t_1 > \cdots > t_K = 0$

for each synthetic sample **do**
  Initialize quantile-transformed numerical state $\mathbf{x}^{t_0}_{\mathrm{num}} \sim \mathcal{N}(0, \sigma(t_0)^2\mathbf{I})$
  Initialize categorical state as fully masked: $x^{t_0,i}_{\mathrm{cat}} = v_{\mathrm{mask}},\ \forall i$
  for $k = 0$ to $K - 1$ **do**
    $t \leftarrow t_k$, $s \leftarrow t_{k+1}$, and form $\mathbf{x}^t = [\mathbf{x}^t_{\mathrm{num}}, \mathbf{x}^t_{\mathrm{cat}}]$
    Compute time embedding $\mathbf{e}_t$ from t
    Encode and denoise:
      $\mathbf{H}^t = \tau(\mathbf{x}^t)$, $\mathbf{Z}^t = \Phi_\theta(\mathbf{H}^t, \mathbf{e}_t, \mathbf{A})$
    Decode predictions:
      $[\hat{\mathbf{x}}^0_{\mathrm{num}}, \mathbf{L}_{\mathrm{cat}}] = g(\mathbf{Z}^t)$
    Update numerical state with the discretized reverse ODE from Eq. (11)
    for each categorical feature i **do**
      if $x^{t,i}_{\mathrm{cat}} \neq v_{\mathrm{mask}}$ **then**
        Keep state: $x^{s,i}_{\mathrm{cat}} \leftarrow x^{t,i}_{\mathrm{cat}}$
      else
        Compute $\beta_i^{(t,s)}$ by Eq. (13)
        With probability $\beta_i^{(t,s)}$, sample $x^{s,i}_{\mathrm{cat}} \sim \mathrm{softmax}(\mathbf{L}_i)$;
          otherwise set $x^{s,i}_{\mathrm{cat}} = v_{\mathrm{mask}}$
      end if
    end for
    Set $(\mathbf{x}^t_{\mathrm{num}}, \mathbf{x}^t_{\mathrm{cat}}) \leftarrow (\mathbf{x}^s_{\mathrm{num}}, \mathbf{x}^s_{\mathrm{cat}})$
  end for
  Map numerical values to raw units: $\hat{\mathbf{x}}^0_{\mathrm{num,raw}} \leftarrow \mathcal{Q}^{-1}(\mathbf{x}^{t_K}_{\mathrm{num}})$
  Decode categorical states and output $\hat{\mathbf{x}}^0_{\mathrm{raw}} = [\hat{\mathbf{x}}^0_{\mathrm{num,raw}}, \mathrm{decode}(\mathbf{x}^{t_K}_{\mathrm{cat}})]$
end for

A.3 LLM Relation-Prior Extraction Details

Model and execution. We use Qwen3-1.7B (`Qwen/Qwen3-1.7B`) through Transformers 4.51.0 in `bfloat16`. The model is loaded from a local snapshot with `local_files_only=True`; no external API is used. The snapshot was accessed on July 16, 2026. Because it was retained as a plain file copy rather than an intact Hugging Face cache, its upstream commit hash is unavailable. We report this limitation rather than infer a revision. No system prompt is used. Thinking is disabled, and the single user message is formatted with the tokenizer's default chat template.

Schema construction. For each dataset, the prompt contains the task type and one descriptor per column. Numerical descriptors include the column name, type, description, and range; categorical descriptors include the name, type, description, and at most 12 observed values. Ranges and value lists are computed locally from the first 5,000 rows of the training split. Raw rows are not included in the prompt. Descriptions come from the dataset metadata and fall back to "numerical column" or "categorical column." Target annotations preserve the target's actual metadata type: a numerical regression target is tagged `[numerical target]`, whereas a categorical classification target is tagged `[categorical target]`. This type-aware rule is applied consistently to all datasets before relation extraction.

Prompt. The following template is used verbatim, with braces denoting substituted fields:

```
You are a feature-relationship extractor for a tabular dataset.
Dataset schema:
{schema}

Task: for EVERY column listed under 'Targets', select its DIRECT
parent columns from the other columns. A parent is a source column
with a strong, direct semantic or causal effect on the target
(e.g. 'education' determines 'education.num'; 'occupation' and
'hours.per.week' help explain income).
Rules:
- At most 3 parents per target; prefer 0-2.
- Never list the target itself; only use existing column names.
- A column that is a basic attribute (e.g. age, gender) usually
  has NO parent.
- Do not just list every other column: empty arrays are preferred
  over weak guesses.
Targets: {targets}
Return ONLY ONE JSON object mapping each target column to its parent
list, e.g.
{"colA": [], "colB": ["colA"],
 "colC": ["colA", "colB"]}
Do NOT explain. Do NOT include a thinking block.
Output the raw JSON only.
Available columns: {names}
```

Here `targets` contains only the columns in the current chunk, whereas `names` contains all columns in the dataset. The word "parent" is used only as a prompting heuristic for selecting directed candidate relations. The resulting edges are interpreted as denoising routes and are not claimed to be causal relations.

Chunking, parsing, and retries. Columns are processed in chunks of 14. Each chunk triggers one generation call, and an empty parsed relation set is retried up to three times. Parsing first removes stray thinking, response, and end-of-message markers; it then scans balanced brace spans and parses each candidate with a JSON decoder. If no JSON object is recovered, lines of the form `name: [...]` are used as a fallback. Parsed names are matched against existing columns with whitespace normalization. Unknown names, self-relations, and keys outside the current target chunk are discarded. Results are merged across chunks; a chunk that fails all retries contributes no relations rather than guessed edges.

Directed graph and caching. Each retained source-to-target relation defines a directed member of $\mathcal{R}_{\mathrm{LLM}}$; the relation set is not symmetrized, and reciprocal relations are retained as two directed edges. The LLM outputs parent lists rather than numerical scores. We therefore apply the fixed binary mapping in Eq. (2): selected relations receive an initial value of one, unselected off-diagonal relations receive zero, and the diagonal is fixed to zero. This binary matrix is only an initialization; in the full model, all off-diagonal entries subsequently become continuous trainable weights under the denoising objective. The extraction stage is executed before diffusion training. Its outputs are cached as `GDiffTable/ckpt/<dataset>/llm_prior.npy` with accompanying metadata in `llm_prior_meta.json`. Diffusion training and sampling read the cached prior and never invoke the LLM. Decoding parameters, random seeds, and dataset-specific cached outputs are versioned with the released implementation rather than repeated in the paper.

A.4 Evaluation Metric Details

We evaluate the quality of synthetic tabular data across four dimensions, following established protocols to ensure comparability. The downstream results in Table 1 are reported as mean $\pm$ standard deviation over five runs. The aggregate multidimensional evaluation and core ablation report measured point estimates unless otherwise stated. For each metric $m \in \{\mathrm{Shape}, \mathrm{Trend}, \alpha\text{-Precision}, \beta\text{-Recall}, \mathrm{DCR}\}$ in Table 2, we first compute a dataset-level score m_d and then report the unweighted macro-average over the seven benchmarks,

$$\bar{m} = \frac{1}{7} \sum_{d=1}^{7} m_d. \tag{14}$$

Thus, every dataset receives equal weight regardless of its number of rows, columns, or compatible column pairs; we do not pool columns, column pairs, or samples across datasets before aggregation. Average Gap is aggregated separately as defined below Table 2.

Statistical Similarity We use the single-table `QualityReport` from SDMetrics in the SDV framework Patki et al. (2016). The report contains two complementary properties. *Column Shapes* measures univariate distributional similarity. It applies `KSComplement` to numerical columns, defined as one minus the Kolmogorov–Smirnov statistic between the real and synthetic empirical cumulative distributions, and `TVComplement` to categorical and Boolean columns, defined as one minus the total-variation distance between their empirical category-frequency distributions. *Column Pair Trends* measures pairwise dependence similarity. For numerical–numerical pairs, `CorrelationSimilarity` compares the Pearson correlations in the real and synthetic data as $1 - |\rho_{\mathrm{real}} - \rho_{\mathrm{syn}}|/2$. For categorical–categorical and numerical–categorical pairs, `ContingencySimilarity` compares normalized two-dimensional contingency tables using one minus their total-variation distance; continuous values are discretized when a contingency-table comparison is required. The reported Shape and Trend scores are the averages over compatible columns and column pairs, respectively. Both scores lie in $[0, 1]$, with larger values indicating greater similarity, and we report them as percentages. Unless otherwise stated, we use the default settings of the SDMetrics version employed in our experiments.

Manifold Fidelity and Coverage Following recent advances in manifold-based evaluation Zhang et al. (2024); Alaa et al. (2022), we incorporate α-precision and β-recall to characterize high-order dependencies. These metrics respectively quantify the fidelity of synthetic samples within the real data support and the coverage of the ground-truth distribution.

DCR near-duplicate diagnostic. Following TabSyn's synthetic-vs.-holdout protocol Zhang et al. (2024), we use equally sized real training and holdout subsets for this evaluation and generate a synthetic set of the same size. Numerical columns are divided by their training-set ranges, categorical columns are one-hot encoded using categories fitted on the training subset, and the resulting representations are compared with the L_1 distance. For a synthetic record s, define

$$d_{\mathrm{tr}}(s) = \min_{x \in D_{\mathrm{tr}}} \|\phi(s) - \phi(x)\|_1, \qquad d_{\mathrm{ho}}(s) = \min_{x \in D_{\mathrm{ho}}} \|\phi(s) - \phi(x)\|_1. \tag{15}$$

The reported score is

$$\mathrm{DCRScore} = \frac{100}{|S|} \sum_{s \in S} \mathbb{I}[d_{\mathrm{tr}}(s) < d_{\mathrm{ho}}(s)]\,. \tag{16}$$

With equal-sized reference sets, a score close to 50% indicates no systematic nearest-neighbor preference for the training subset, whereas a value approaching 100% indicates that synthetic records are disproportionately closer to training records. DCR is only an empirical near-duplicate diagnostic; it neither constitutes a formal privacy guarantee nor rules out inference attacks.

A.5 Ablation Configuration Details

We compare five variants while keeping the diffusion processes, feature encoders, reconstruction heads, hidden dimension, and optimization budget unchanged:

- **No Graph** replaces inter-column message passing with a capacity-matched node-wise MLP denoiser.
- **Uniform Dense Graph** uses a fixed complete adjacency with identical off-diagonal weights and a zero diagonal. It retains global cross-column message passing but removes both learned non-uniform routing and semantic edge selection.
- **Data-only Graph** learns the adjacency matrix from the diffusion objective, starting from a non-informative initialization whose mean off-diagonal weight matches that of the semantic prior.
- **Fixed LLM Prior** uses the LLM-derived adjacency throughout training and does not update its topology or edge weights.
- **GDiffTable** initializes the learnable graph from the LLM prior and refines it jointly with the diffusion denoiser.

The same data split, diffusion setup, and early-stopping rule are used for all variants. The four graph variants share the same message-passing backbone. Uniform Dense Graph and Fixed LLM Prior keep their respective adjacencies fixed, whereas Data-only Graph and GDiffTable use the same trainable adjacency parameterization and sparsity coefficient and differ only in initialization. We conduct the ablation on Adult and News: Adult represents a standard mixed-type classification setting, whereas News provides a relatively wide regression setting. Table 3 reports measured point estimates of downstream utility and aggregate Shape and Trend, together with wall-clock training time to the common early-stopping criterion. The Uniform Dense Graph control isolates generic all-to-all communication from learned structured routing: improvement over No Graph supports the value of cross-column mixing, while improvement from Uniform Dense Graph to Data-only Graph or GDiffTable supports the value of non-uniform routing.

A.6 Dataset and Baseline Details

Datasets: To provide a comprehensive evaluation, we conduct extensive experiments on seven real-world tabular datasets with diverse characteristics, including Adult Becker & Kohavi (1996), Default Yeh (2009), News Fernandes et al. (2015), Insurance Choi (2018), Magic Bock et al. (2004), Nursery Rajkovic (1989), and California Pace & Barry (1997). As summarized in Table 4, these benchmarks represent a wide spectrum of data modalities: from purely categorical and purely numerical to mixed-type distributions. Furthermore, they cover both classification and regression tasks with varying scales, ranging from approximately 10^3 to nearly 5×10^4 samples. Following the protocol in TabDDPM Kotelnikov et al. (2023), we adopt standard train/test splits for all datasets to ensure a fair comparison. For datasets with pre-defined splits (e.g., Adult), we use the original test sets. For others, we employ a random $80/20\%$ split with a fixed seed to maintain consistency across all evaluated models.

Baselines: We compare against a broad range of methods: SMOTE Chawla et al. (2002), a classic interpolation-based reference; CTGAN and TVAE Xu et al. (2019), foundational deep generative models for tabular data; CG-TGAN Lee & Min (2025), a graph-enhanced GAN that models column relationships via GNNs; TabDDPM Kotelnikov et al. (2023), an early diffusion model using separate Gaussian and multinomial processes; CoDi Lee et al. (2023), which couples numerical and categorical diffusion for dependency modeling; STaSy Kim et al. (2023), a score-based model with self-paced fine-tuning; GOGGLE Liu et al. (2023), a VAE that jointly learns feature graph structure and generation; TabSyn Zhang et al. (2024), which performs score-based diffusion in a VAE latent space; TabDiff Shi et al. (2025), a joint continuous-time diffusion framework; and GReaT Borisov et al.

Table 4: Summary of datasets used in experiments.

Dataset	# Train	# Test	# Num.	# Cat.	Task
California	16,512	4,128	9	0	Regression
Nursery	10,363	2,592	0	8	Classification
Adult	32,561	16,281	6	9	Classification
Default	24,000	6,000	14	11	Classification
News	31,715	7,929	45	15	Regression
Insurance	1,070	268	4	3	Regression
Magic	15,216	3,804	10	1	Classification

Table 5: Wall-clock efficiency comparison. Train is total training time; Sample is the time required to generate a synthetic dataset matching the corresponding real training-set size.

Method	Insurance		Adult		News	
	Train	Sample	Train	Sample	Train	Sample
CTGAN	66.0s	0.1s	1295.0s	0.5s	2664.0s	1.8s
TVAE	47.0s	0.1s	496.0s	0.2s	1137.0s	0.6s
TabDDPM	334.0s	4.0s	1132.0s	30.9s	3380.0s	106.6s
TabSyn	899.0s	0.5s	2342.0s	12.7s	5840.0s	28.0s
GDiffTable	379.0s	0.8s	1448.0s	18.1s	3985.0s	50.3s

(2023), which fine-tunes an autoregressive language model on textualized table rows. All baselines use official implementations with default hyperparameters.

A.7 Efficiency Analysis

To investigate the computational efficiency of different generative paradigms, we report total wall-clock training time and the time required to generate a synthetic dataset matching the real training-set size for representative methods across three datasets (Insurance: $\sim$1K samples, Adult: $\sim$32K samples, News: $\sim$32K samples).

As shown in Table 5, one-shot models (CTGAN, TVAE) achieve sub-second dataset generation but lag in generation quality. Among diffusion-based methods, TabDDPM has the highest sampling cost due to its full DDPM trajectory, while TabSyn samples fastest through VAE compression but incurs the highest training overhead. GDiffTable strikes a middle ground: its training is faster than TabSyn in these comparisons (e.g., $3{,}985\,\mathrm{s}$ vs. $5{,}840\,\mathrm{s}$ on News), and its ODE-based sampling ($50.3\,\mathrm{s}$ on News) is cheaper than TabDDPM ($106.6\,\mathrm{s}$). From Adult to News, GDiffTable's observed training and sampling times increase by $2.75\times$ and $2.78\times$, respectively, compared with $2.99\times$ and $3.45\times$ for TabDDPM. Because these datasets differ in more than feature count, this comparison is descriptive rather than a controlled scalability claim.

A.8 Hyperparameter Sensitivity Analysis

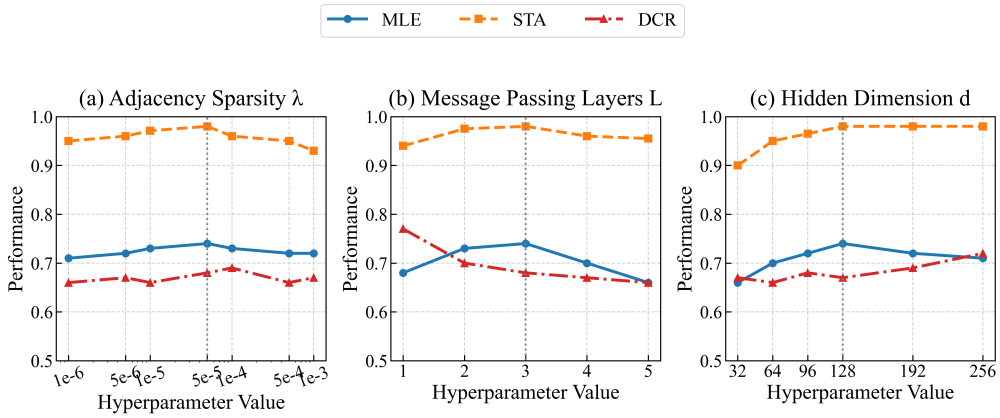


Figure 3: Sensitivity to adjacency sparsity λ, message-passing depth L, and hidden dimension d. MLE denotes machine-learning efficiency, STA $= (\text{Shape} + \text{Trend})/2$, and DCR is best near 0.5. Dotted lines mark the defaults $\lambda = 5 \times 10^{-5}$, $L = 3$, and $d = 128$.

We evaluate the sensitivity of GDiffTable to the three hyperparameters in Fig. 3. In panel (a), MLE and STA peak around $\lambda = 5 \times 10^{-5}$, while stronger sparsity regularization reduces utility. In panel (b), both metrics peak at $L = 3$, suggesting that moderate message-passing depth is sufficient. In panel (c), gains saturate around $d = 128$, while the DCR score moves farther from its ideal value of 0.5 at larger dimensions. The reported curves vary smoothly around the selected settings; we therefore use $\lambda = 5 \times 10^{-5}$, $L = 3$, and $d = 128$ as defaults.