When Tables Leak: Attacking String Memorization in LLM-Based Tabular Data Generation

Joshua Ward
University of California Los Angeles

Bochau Gu
University of California Los Angeles

Chi-Hua Wang
University of California Los Angeles

Guang Cheng
University of California Los Angeles

Abstract

Large Language Models (LLMs) have recently demonstrated remarkable performance in generating high-quality tabular synthetic data. In practice, two primary approaches have emerged for adapting LLMs to tabular data generation: (i) fine-tuning smaller models directly on tabular datasets, and (ii) prompting larger models with examples provided in context. In this work, we show that popular implementations from both regimes exhibit a tendency to compromise privacy by reproducing memorized patterns of numeric digits from their training data. To systematically analyze this risk, we introduce a simple No-box Membership Inference Attack (MIA) called LevAtt that assumes adversarial access to only the generated synthetic data and targets the string sequences of numeric digits in synthetic observations. Using this approach, our attack exposes substantial privacy leakage across a wide range of models and datasets, and in some cases, is even a perfect membership classifier on state-of-the-art models. Our findings highlight a unique privacy vulnerability of LLM-based synthetic data generation and the need for effective defenses. To this end, we propose two methods, including a novel sampling strategy that strategically perturbs digits during generation. Our evaluation demonstrates that this approach can defeat these attacks with minimal loss of fidelity and utility of the synthetic data. [1]

Keywords

Membership Inference Attacks, Tabular Synthetic Data, Large Language Models, Privacy

1 Introduction

Machine learning systems across diverse domains—from healthcare databases to financial risk assessment platforms—rely heavily on structured tabular datasets [5, 16, 20]. This widespread dependence has driven significant interest in synthetic tabular data generation, where computational models learn to produce artificial records that statistically mirror the patterns of real datasets while avoiding direct replication [17]. The utility of synthetic data stems from two primary advantages: it can supplement limited training samples to improve model performance on underrepresented populations or rare events, and it facilitates private data sharing by generating records that do not directly correspond to actual individuals. These benefits are especially valuable in privacy-sensitive sectors such as medicine [49] and banking [57], where regulatory constraints often limit access to original data. As a result, synthetic data generation has emerged as an essential technique for expanding machine learning capabilities in data-constrained and privacy-regulated environments[34, 39].

Large Language Models (LLMs) have recently emerged as state-of-the-art tabular synthetic data generators. Unlike traditional approaches—such as Generative Adversarial Networks (GANs), Variational Autoencoders (VAEs), and diffusion models—that operate directly over original feature space, LLMs encode tabular rows into string representations and leverage two primary methodologies for generation. In-Context Learning (ICL) approaches include existing tabular rows within the LLM's context window and prompt the LLM to generate additional rows following the observed patterns [24, 29, 32, 43], while Supervised Fine-Tuning (SFT) methods train LLMs on larger quantities of string-encoded tabular data to learn the underlying distribution before sampling synthetic records [7, 46, 51]. Both approaches generate synthetic data by producing new string sequences that are subsequently decoded back into tabular format. In both cases, LLM-based generators have been shown to produce synthetic data that exhibits superior statistical fidelity to original datasets and maintains high utility for downstream machine learning tasks. Moreover, initial evaluations suggest these methods may offer enhanced privacy protection relative to conventional approaches [46].

However, these architectural differences raise unresolved questions about privacy. Extensive research has demonstrated that LLMs exhibit tendencies to memorize training data, particularly when exposed to repeated patterns [11, 28], longer sequences [10, 52], or through supervised fine-tuning processes [13]. These memorization behaviors, which can be beneficial for language modeling tasks, present unique privacy risks in the context of tabular data generation where training data often contain structured, long sequences of repeated values across many observations.

In order to measure privacy risk, Membership Inference Attacks (MIAs) [9, 45] have emerged as the linchpin of privacy auditing for tabular synthetic data generators, serving as the primary tool for evaluating privacy risks across diverse generative approaches [12, 22, 50, 54, 55]. However, these methods focus exclusively on the feature space over which traditional generative models operate, potentially missing the string-space vulnerabilities introduced by LLM-based generation entirely.

[1] A code repository is available here.

Proceedings on Privacy Enhancing Technologies 2026(3), 408–425

https://doi.org/10.56553/popets-2026-0088

To bridge this gap, we examine privacy risks in LLM-based tabular data generation by introducing LevAtt, an MIA that exploits Levenshtein Distance on the string representations of tabular data—the actual format LLMs generate—rather than the feature space alone. We find through extensive experimentation in both the ICL and SFT regimes that state-of-the-art LLMs often memorize and replicate numeric values from training data, exposing sensitive information digit-for-digit in synthetic outputs. Even under a conservative no-box threat model, where we assume only adversarial access to the synthetic data, we uncover that LLMs can leak private data through memorized digit patterns, revealing vulnerabilities that conventional feature-space MIAs fail to detect.

To address this new-found risk, we study two defenses: an ad-hoc post-processing algorithm we call Digit Modifier (DM) that flips digits to break sequential patterns and a novel Tendency-based Logit Processor (TLP) that strategically perturbs digits at sample time. We find that both strategies can defeat these attacks, however TLP can effectively control for privacy leakage with minimal effect on the statistical fidelity or downstream machine learning utility of the synthetic data.

2 Related Work

To situate LevAtt within the broader landscape of synthetic data research and privacy auditing, we review prior work on tabular data generation, LLM-based modeling, and membership inference attacks, highlighting how LLMs introduce fundamentally new vulnerabilities not present in conventional generators.

2.1 Tabular Data Generation

Tabular generative models aim to learn a generator G from training data T that approximates the true data distribution $p_X(X)$. We represent tabular data as a matrix $\mathbf{X} \in \mathcal{X}^{n\times d}$, where n denotes the number of samples, d the number of features, and $\mathcal{X}$ is the feature value domain. Each row $\mathbf{x}_i \in \mathcal{X}^d$ represents a data point drawn from the underlying distribution $p_X(X)$, while columns correspond to features that may have heterogeneous data types. We denote the j-th feature value of the i-th sample as $\mathbf{x}_{i,j}$. The training dataset $T = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$ comprises n independent samples from $p_X(X)$. The learned model generates synthetic samples $\tilde{\mathbf{x}} \sim G$, forming a synthetic dataset $S = \{\tilde{\mathbf{x}}_1, \tilde{\mathbf{x}}_2, \ldots, \tilde{\mathbf{x}}_m\}$. In this work, we assume that features can be of a continuous, ordinal, or categorical type.

Deep Learning-Based Generation. In recent years, a variety of conventional tabular synthetic data generators have been proposed including Generative Adversarial Networks [58–60], likelihood-based methods [4, 14, 56], and diffusion models [30, 48, 62] Each of these operate directly over the feature space $\mathcal{X}^d$, learning mappings $\mathcal{G} : \mathcal{Z} \to \mathcal{X}^d$ that model the joint distribution $p_X(X)$. Crucially, these approaches fundamentally treat each sample as an atomic unit, generating a complete feature vector $\mathbf{x} \in \mathcal{X}^d$ simultaneously.

LLM-Based Generation In contrast, LLM-based approaches reframe tabular generation as autoregressive text generation. Training samples are encoded into strings, tokenized into sequences from vocabulary $\mathcal{W}$, and the LLM generates according to $p(t) = \prod_{k=1}^{j} p(w_k|w_1, \ldots, w_{k-1})$. Rather than modeling the joint distribution $p_X(X)$ directly, this approach decomposes it through sequential conditioning, generating feature values $x_{i,j}$ token by token based on previously generated values. This sequential generation process fundamentally breaks the atomic unit assumption of other architectures, introducing new dynamics where generated tokens influence later ones through the autoregressive chain.

Within this paradigm, two complementary generation strategies have emerged based on data availability and computational resources. In-context-based methods leverage large foundation models by presenting tabular examples directly within the context window, enabling few-shot generation without parameter updates in low-data regimes [24, 29, 32, 43]. When larger datasets are available, SFT-based methods instead fine-tune smaller language models through direct optimization on tabular generation tasks [7, 46, 51]. Both strategies maintain the core autoregressive formulation while differing in how they leverage available data and computational resources.

2.2 Membership Inference Attacks on Synthetic Tabular Data

Membership Inference Attacks (MIAs) aim to classify whether a specific observation was a member of the original dataset used to train a model [9, 12, 45]. Given the generative model G trained on dataset T as defined above, which generates synthetic dataset S, an adversary $\mathcal{A} : X \to \{0, 1\}$ aims to determine if a test sample x^* is an element of T. Formally, this classification or MIA can be expressed as:

$$\mathcal{A}(x^\star) = \mathbb{I}\left[f(x^\star) > \gamma\right] \tag{1}$$

where $\mathbb{I}$ is the indicator function, $f(x^\star)$ is a scoring function of the test observation x^*, and γ is an adjustable decision threshold. The success of the attack can be measured using traditional binary classification metrics and can be interpreted as a measure of privacy leakage from a model of the training data.

MIAs have become a mainstay tool to audit the privacy of tabular synthetic data as they represent a material privacy risk associated with the output of a model [25, 53]. Indeed, a number of distance [12, 54], density [22, 50], and likelihood-based [35, 47, 55] attacks have been proposed under a wide variety of threat models. However, these existing methods all attack over the tabular feature space $\mathcal{X}^d$. LLM-based generators introduce a fundamentally different vulnerability: they generate in an intermediate string representation space before parsing to a tabular format. This autoregressive string generation process creates a novel attack surface that bypasses traditional feature-space-targeting attacks.

3 Attacking Implied Strings of LLM-Based Generators

LLM-based tabular data generators operate on records encoded as sequences of characters, often representing numeric values in fixed formats. From a privacy perspective, these numeric strings constitute a *distinct threat surface*: even minor changes at the character level can correspond to meaningful alterations in sensitive information, such as financial amounts, medical measurements, or personally identifiable metrics. Unlike free-form text, where approximate matches may be semantically ambiguous, we hypothesize that numeric strings are rigid and highly informative, making them particularly vulnerable to memorization by LLMs and thus adversarial signal.

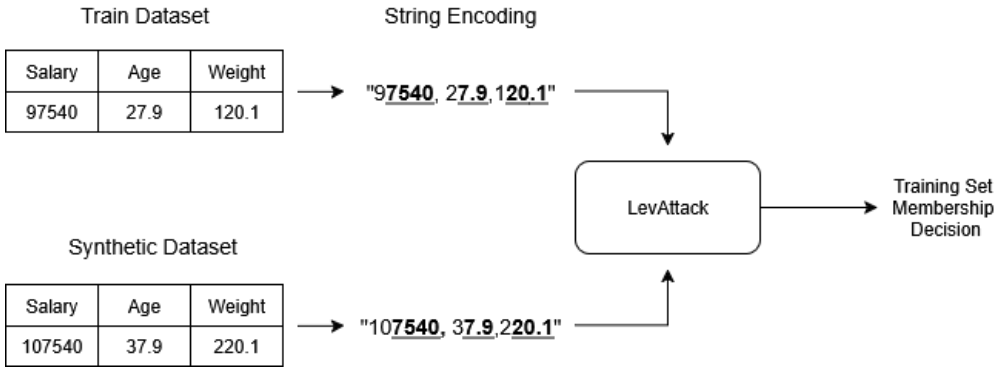


Figure 1: Diagram of Levenshtein Attack. We simply encode rows of tabular data into a string representation from which to attack. LevAtt finds signal in the highly constrained and often duplicated sequences of digits in synthetic tabular data generated by LLMs. In bold and underline: copied sequences of such patterns. Where these rows would be relatively far in Euclidean distance, repeated sequences in their string representations are the source of LevAtt's adversarial advantage.

In this section, we focus on *attacks that exploit these implied strings*. We show that by measuring string similarity between synthetic outputs and potential training records, an adversary can infer membership without access to the model internals, queries, or auxiliary data. This approach exposes a realistic privacy risk inherent in current LLM-based tabular data generation pipelines, emphasizing the need to examine character-level leakage beyond traditional feature-space analyses.

3.1 Threat Model

In this work, we adopt a conservative No-box threat model [25, 53], in which the adversary has access only to the synthetic data S produced by the model. We make no assumptions that the adversary has knowledge of the model implementation and hyperparameters, query access, or even a reference dataset for constructing calibrated attacks. This threat model is motivated by two key considerations:

(1) **Realism**. No-box threat models represent realistic privacy scenarios for tabular synthetic data practitioners. A common scenario involves a user training a tabular data generator on private datasets and sharing the generated data with public or private parties without disclosing implementation details. From the adversary's perspective, this synthetic data may be maliciously acquired or discovered on open data sharing platforms, with no additional context about the generation process.

(2) **Difficulty**. No-box threat models are recognized as particularly challenging for constructing effective attacks due to the severe limitations placed on the adversary [12, 25]. Specifically, the adversary cannot analyze the model's loss function, construct shadow models, or query the model directly. Successful attacks under these restrictive conditions therefore highlight fundamental vulnerabilities in these synthetic data generation methods.

3.2 Levenshtein Attack

String similarity is a well-established concept, with Levenshtein edit distance providing a fundamental measure of character-level overlap [31, 37, 61]. Recent work has applied normalized variants of this distance to quantify approximate memorization in LLMs, showing that models can reproduce training examples with minor variations in natural language text [26, 44]. However, these studies stop short of embedding string similarity into an adversarial framework, as open-ended natural language provides many valid paraphrases and thus yields weak signals for membership inference.

In contrast, *the string representations of values in tabular data are highly constrained*. Unlike free-form text—where two sentences can express identical meaning with entirely different surface forms—tabular records have rigid schemas. Numerical fields follow fixed precision, delimiters appear in consistent positions, categorical fields draw from small vocabularies, and missing values are encoded in standardized ways. As a result, even a single-character modification (e.g., "17.5" → "17.6") corresponds to a meaningful change. This rigidity turns Levenshtein distance from a fragile signal in natural language into a highly sensitive indicator of memorization in tabular domains.

To operationalize this idea, we convert each record into a canonical string representation that preserves its structure. Numerical features are formatted with consistent precision, categorical variables mapped to stable tokens, and delimiters inserted to mark column boundaries. This step ensures that edit distance reflects true content similarity rather than superficial formatting differences. When a synthetic generator memorizes a training record, its outputs often contain near-exact replicas—matching digit sequences, categorical codes, and formatting patterns. Because such close matches are extremely unlikely to arise by chance, unusually low edit distances between a real record and its nearest synthetic neighbor provide strong evidence of membership.

Motivated by these observations, we design a membership inference attack based on Levenshtein distance. For each test observation x^* and synthetic set S, we extract the ordered numeric and categorical values and encode them as strings (see Figure 1), obtaining $x^*\text{str}$ and S_{str}. The LevAtt score for Equation 1 is defined as:

$$f_{\text{LevAtt}}(x^*_{\text{str}}, S_{\text{str}}) = -\min_{s \in S_{\text{str}}}, \ell(x^*_{\text{str}}, s), \tag{2}$$

where ℓ denotes the Levenshtein edit distance. Lower distances (i.e., higher scores) indicate closer matches and therefore stronger evidence of memorization.

LevAtt takes as input any type of data including numeric, categorical, missing, and open text, as each of these can be encoded as a string. As LevAtt requires only synthetic outputs, it applies directly to our No-box threat model highlighting a realistic privacy risk in LLM-based tabular data generation. Since the scale of Levenshtein distances varies by dataset, a decision threshold γ can be chosen relative to the distribution of scores, such as selecting a value from the Inter-Quartile Range. This approach mirrors the evaluation of DOMIAS [50], where the median score serves as a threshold to derive binary classification metrics and assess overall attack performance.

4 Experiments

We evaluate the privacy leakage of string memorization for in-context learning (ICL) and supervised fine-tuning (SFT) LLM-based tabular generators. Here, we use differing experimental designs to correspond to their popular use-case settings. For clarity, we organize the section into three subsections: experimental design details for both regimes and then separate subsections for results. We include the full experimental and implementation details in Appendix: A.

4.1 Experimental Design

4.1.1 ICL Approaches. Replicating the design of [8], we evaluate ICL approaches on the OpenML CTR23 benchmark [15], consisting of 35 real-world regression datasets with 500–100,000 rows and up to 5,000 features, including both numerical and categorical attributes (see Table 6). We partition each dataset into 80/20 training and test sets. To simulate the low-data regime these methods are commonly used for, we subsample 32, 64, 128 training rows without replacement. These sampled rows are provided as exemplars to the ICL models.

We consider three ICL models: LLaMA 3.3 70B [36] an open-source foundation model, GPT-4o-mini [38] a close-source foundation model, and TabPFN-v2 [23] a transformer-based model trained on tabular data due to their wide use and availability. For each sampled training subset, we generate an equal amount of synthetic data. Evaluation is conducted on all exemplar rows plus an equal holdout partition of the test set. We evaluate privacy leakage using the following No-box MIAs using the Synth-MIA package [53]: LevAtt, Euclidean Distance to Closest Record (DCR) [12], a Monte Carlo density estimation (MC) method [22], and a kernel density estimation method [25, 50]. We report MIA success using mean AUC-ROC and TPR at fixed FPR thresholds [9] to capture potential information leakage. We repeat this experimentation across 3 seeds, totaling 315 datasets/ subsample sizes/ seeds per model.

4.1.2 SFT Approaches. For SFT-approaches, we benchmark the original SFT-based tabular generation method GREAT [7] and RealTabFormer [46], which reports improved privacy performance due to enforcing a minimum Euclidean Distance to Closest to Record distribution in its training. As both of these methods use GPT-2 [42]as a base model, we modify RealTabFormer to accept more modern, larger foundation models: LLaMA 3.2 (1B, 3B) [21], Qwen2.5-3B [41], and Mistral v0.3 7B [27]. Additionally, we introduce as a control CT-GAN and TVAE [58], conventional deep learning-based generators. Lastly, we benchmark DP2Stage [1], a differentially private framework that SFT's GPT-2 with DP-SGD. Training follows default hyperparameters from the original GREAT, RealTabFormer, and DP2Stage implementations while CTGAN and TVAE are implemented through Synthcity [40].

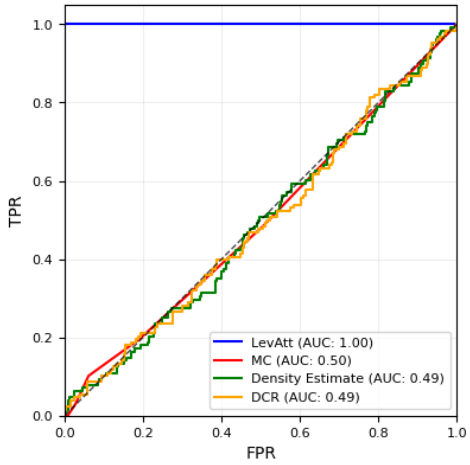


Figure 2: ROC plot for various No-box MIAs against TabPFN-V2 with 128 in-context samples from the MoneyBall dataset. LevAtt (blue) is able to achieve perfect classification for all in-context samples whereas MIAs that target the feature space of tabular data fail to capture the privacy leakage.

Experiments are conducted on five tabular datasets: CASP, Abalone, Diabetes, CA-Housing, and Faults (see Table 7), selected for their common use in synthetic tabular data benchmarking and containing numeric data. Similarly to ICL, we create 80/20 train-test partitions and following common synthetic tabular data benchmarking [62], synthetic datasets equal in size to the original training sets are generated post-training. For privacy evaluation, up to 1,000 training and holdout samples are partitioned as test sets, and the same MIAs are applied. Again, we repeat this experimentation across 3 seeded runs.

4.2 Metrics

We evaluate the synthetic data generators across three primary dimensions: privacy, fidelity, and utility. To quantify privacy, we measure the success of LevAtt using the Area Under the ROC Curve (AUC) and the True Positive Rate (TPR) at low False Positive Rate (FPR) thresholds. Following Carlini et al. [9], we prioritize these non-thresholded metrics because fixed-threshold accuracy can mask score separability, thereby underestimating the risk of memorization. Fidelity is assessed by calculating the Maximum Mean Discrepancy (MMD) with an RBF kernel and the Wasserstein distance between the real and synthetic distributions, where lower values signify superior distributional alignment. Finally, we evaluate utility via a "Train on Synthetic, Test on Real" (TSTR) framework [40]; we train an XGBoost classifier on the synthetic data and report

Table 1: LevAtt performance against ICL models. Mean (STD) values are reported for all datasets, training sizes, and seeds (315 runs Overall) and for the top 20 runs for each model with the highest AUC-ROC (Top 20). GPT-4o-mini shows relatively little privacy leakage, whereas LLama-3.3-70b and TabPFN-V2 show significant privacy failure- especially amongst their worst datasets. At the same time, these models generally have the best fidelity and utility measures.

		LevAtt			Fidelity		ML Utility	
Run	Model	AUC	TPR@0	TPR@0.1	MMD ↓	Wass ↓	AUC ↑	F1 ↑
Overall	LLaMA-3.3-70B	**0.63 ± 0.13**	**0.08 ± 0.20**	**0.28 ± 0.24**	**0.04 ± 0.02**	1369.27 ± 22068.62	0.63 ± 0.12	0.41 ± 0.18
	TabPFN-V2	0.58 ± 0.11	0.04 ± 0.17	0.19 ± 0.21	0.04 ± 0.03	**1.01 ± 1.14**	**0.66 ± 0.12**	**0.41 ± 0.18**
	GPT-4o-mini	0.54 ± 0.05	0.00 ± 0.01	0.13 ± 0.06	0.05 ± 0.07	4458.30 ± 72617.27	0.59 ± 0.10	0.34 ± 0.17
Top 20	LLaMA-3.3-70B	**0.97 ± 0.03**	**0.64 ± 0.27**	0.93 ± 0.07	**0.03 ± 0.02**	**1.38 ± 1.62**	0.65 ± 0.11	**0.41 ± 0.14**
	TabPFN-V2	0.97 ± 0.07	0.57 ± 0.41	**0.94 ± 0.14**	0.05 ± 0.05	2.66 ± 1.62	**0.68 ± 0.16**	0.40 ± 0.20
	GPT-4o-mini	0.60 ± 0.09	0.01 ± 0.03	0.27 ± 0.09	0.07 ± 0.05	1.84 ± 1.80	0.54 ± 0.05	0.26 ± 0.08

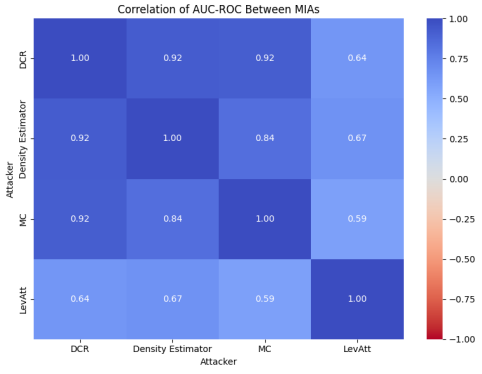


Figure 3: Correlation plot for No-box MIA AUC-ROC across the ICL experiment. While the feature-space targeting DCR, Density Estimate, and MC are nearly perfectly correlated, LevAtt is much less correlated. This highlights that while privacy leakage over tabular string representations and the feature space are related, LevAtt finds unique adversarial advantage.

the AUC and F1 score on a held-out real dataset to determine how effectively the synthetic samples preserve task-relevant structure.

4.3 Results: ICL Methods

Our ICL experiments reveal clear evidence that LLM-based tabular generators can exhibit substantial privacy leakage, even under an extremely restrictive threat model. We organize our findings around three central observations.

LevAtt identifies substantial privacy leakage in ICL-based tabular generators. Table 1 reports the Mean (STD) of LevAtt AUC-ROC and TPR@FPR $\in \{0.0, 0.1\}$ across all datasets, training sizes, and seeds. Because privacy audits often emphasize worst-case behavior, we also report metrics for the top 20 runs (ranked by AUC-ROC) for each model. This analysis reveals that even a simple string-similarity attack can be highly effective against state-of-the-art ICL generators. LLama 3.3-70B and TabPFN-V2 are particularly vulnerable, with mean TPR@FPR=0 values of 0.64 and 0.57 respectively in their highest-performing runs. Strikingly, LevAtt even achieves *perfect* membership classification in several TabPFN-V2 settings (Figure 2), demonstrating that memorized digit sequences can sometimes be reproduced verbatim in generated samples.

Additionally, we analyze the privacy leakage of each model based on the number of in-context examples in Table 8 (Appendix C). Here, we find that for all models, runs with fewer examples consistently experienced greater privacy leakage. This indicates that at small sample sizes, ICL data generation strategies have a greater tendency to directly memorize the string representations of example data.

Privacy leakage scales with model size. These results are consistent with prior evidence that memorization grows with model capacity [10, 11]. Although the parameter counts of GPT-4o-mini and TabPFN-V2 are not publicly disclosed, empirical benchmarks place GPT-4o-mini near the 7B scale while TabPFN-V2 is lightweight enough for single-GPU inference. In contrast, the 70B-parameter LLaMA 3.3 shows the largest privacy leakage across our benchmark. This pattern reinforces that larger models, even in an ICL setting without fine-tuning, have greater capacity to memorize and regenerate specific training examples.

LevAtt captures a distinct leakage signal relative to feature-space MIAs. To evaluate whether LevAtt overlaps with traditional feature-space MIAs, we analyze its correlation with DCR, Density Estimation, and MC attacks (Figure 3). The three feature-space attacks are nearly perfectly correlated with each other, reflecting their shared reliance on density discrepancies. In contrast, LevAtt exhibits substantially weaker correlation with all three, indicating that it identifies leakage signals orthogonal to feature-space methods. Indeed, in some instances, we found that LevAtt was a perfect classifier of membership in some experimental runs whereas traditional feature-space MIAs with compatible threat models were no better than random. This highlights that LLM-based tabular generators introduce a novel, string-level memorization vector that would be entirely missed by conventional synthetic-data MIAs.

4.4 Results: SFT Methods

We now turn to the SFT regime, where models are explicitly trained to approximate the full joint distribution of the training data. While leakage is generally less severe than in ICL, our findings demonstrate that SFT does not eliminate privacy risk and that leakage systematically depends on model design, sampling volume, and data structure.

Table 2: (Mean ± STD) LevAtt performance for each model and dataset across seeds. RealTabFormer experiences significant privacy leakage and LevAtt finds some signal for most LLM-based models in various datasets. However, LevAtt identifies no leakage in conventional deep learning-based methods CT-GAN and TVAE as they do not generate strings. (- indicates the model did not converge.)

Model	Metric	Housing	Diabetes	CASP	Faults	Abalone
RealTabFormer	Lev-AUC	**0.62 ± 0.01**	**0.67 ± 0.01**	**0.70 ± 0.01**	**0.69 ± 0.00**	**0.59 ± 0.00**
	TPR@FPR=0.1	**0.17 ± 0.02**	0.25 ± 0.02	**0.35 ± 0.02**	**0.33 ± 0.01**	**0.17 ± 0.00**
LLaMA 3.2-1B	Lev-AUC	0.58 ± 0.00	0.57 ± 0.00	0.51 ± 0.01	0.56 ± 0.01	0.52 ± 0.00
	TPR@FPR=0.1	0.13 ± 0.00	0.21 ± 0.00	0.09 ± 0.01	0.16 ± 0.01	0.12 ± 0.01
LLaMA 3.2-3B	Lev-AUC	0.59 ± 0.01	0.57 ± 0.01	0.50 ± 0.00	0.55 ± 0.01	0.57 ± 0.00
	TPR@FPR=0.1	0.14 ± 0.01	0.17 ± 0.02	0.09 ± 0.01	0.13 ± 0.01	0.17 ± 0.01
GPT2	Lev-AUC	0.59 ± 0.00	0.51 ± 0.00	0.51 ± 0.01	0.52 ± 0.00	—
	TPR@FPR=0.1	0.14 ± 0.01	0.09 ± 0.01	0.10 ± 0.00	0.14 ± 0.01	—
GReaT	Lev-AUC	0.58 ± 0.01	0.50 ± 0.01	0.52 ± 0.00	—	0.51 ± 0.00
	TPR@FPR=0.1	0.12 ± 0.02	0.09 ± 0.00	0.10 ± 0.00	—	0.11 ± 0.01
Qwen 2.5-3B	Lev-AUC	0.59 ± 0.00	0.64 ± 0.00	0.51 ± 0.01	0.54 ± 0.01	0.51 ± 0.01
	TPR@FPR=0.1	0.13 ± 0.01	**0.27 ± 0.02**	0.10 ± 0.01	0.11 ± 0.01	0.11 ± 0.01
Mistral-7B	Lev-AUC	0.58 ± 0.00	0.52 ± 0.00	0.51 ± 0.02	0.51 ± 0.00	0.50 ± 0.00
	TPR@FPR=0.1	0.13 ± 0.01	0.10 ± 0.00	0.10 ± 0.02	0.11 ± 0.01	0.11 ± 0.01
TVAE	Lev-AUC	0.51 ± 0.00	0.50 ± 0.01	0.48 ± 0.00	0.50 ± 0.00	0.49 ± 0.01
	TPR@FPR=0.1	0.09 ± 0.00	0.10 ± 0.01	0.10 ± 0.00	0.08 ± 0.01	0.08 ± 0.01
CT-GAN	Lev-AUC	0.50 ± 0.01	0.48 ± 0.00	0.49 ± 0.01	0.49 ± 0.01	0.50 ± 0.00
	TPR@FPR=0.1	0.09 ± 0.01	0.09 ± 0.00	0.10 ± 0.01	0.09 ± 0.02	0.06 ± 0.00
DP2Stage ($\epsilon = 10$)	Lev-AUC	0.53 ± 0.00	0.50 ± 0.01	0.48 ± 0.00	—	0.48 ± 0.00
	TPR@FPR=0.1	0.10 ± 0.01	0.11 ± 0.02	0.09 ± 0.01	—	0.08 ± 0.01

SFT approaches exhibit noticeable but lower privacy leakage. Table 2 summarizes LevAtt performance across all datasets and seeds. Compared to ICL, SFT models typically show reduced leakage; however, RealTabFormer consistently exhibits the highest vulnerability—despite incorporating privacy-aware training—and does so more prominently than GREAT, despite their shared GPT-2 base. LevAtt also detects measurable leakage in LLaMA-3.2 (1B and 3B) and Qwen-2.5-3B. By contrast, LevAtt is ineffective against CT-GAN and TVAE, which generate full tabular rows rather than token sequences. This reinforces that token-level generation provides an attack surface absent in conventional deep-learning tabular synthesizers.

Privacy leakage increases with the volume of synthetic data. To understand how synthetic data scale affects leakage, we generate datasets at 0.25×, 0.5×. 1×, 5×, and 10× the size of the original training data using RealTabFormer. As shown in Figure 4, LevAtt's AUC-ROC increases monotonically with synthetic dataset size, with some datasets (e.g., Faults) exhibiting up to a 20% absolute increase from 1× to 10×. At lower levels, 0.25× and 0.5×, LevAtt sees worse performance as there is less information released from the generator. This suggests a simple mechanism: emitting more samples increases the likelihood that a memorized training example is reproduced. This has direct implications for practitioners who rely on synthetic data to replace or augment sensitive datasets—larger synthetic releases can inadvertently amplify privacy risk.

Memorization increases with sequence length in the training data. We study how digit sequence length affects memorization. Here, we create training and holdout sets using identical multivariate Gaussian distributions with a controlled numbers of digit columns. We then vary the total sequence length of digits for a row while keeping these distribution fixed. After training RealTabFormer on datasets with progressively longer sequences, we observe that LevAtt performance increases accordingly (Figure 5) before leveling off at 100 digits. This replicates earlier findings that longer sequences encourage LLM memorization [11] and highlights that dataset structure—not only model architecture—plays a direct role in privacy risk. Datasets containing long numeric strings (e.g., identifiers, timestamps, financial fields) inherently create more opportunities for memorization and unintended reproduction.

LevAtt Privacy vs Fidelity Tradeoff. Finally, the empirical results in Table 3 reveal a tension between generative utility and LevAtt susceptibility. Most notably, RealTabFormer achieves state-of-the-art performance (0.92 ± 0.11 ML AUC) but incurs the highest empirical privacy risk (0.65 ± 0.04 Lev AUC), suggesting that high-fidelity synthesis often correlates with increased record-level leakage. This trade-off is further quantified by the negative correlation between Lev AUC and statistical distance metrics like MMD (−0.163) and Wasserstein distance (−0.084), indicating that as LLMs minimize distributional divergence, they become more vulnerable to string similarity attacks. We also observe that incorporating larger, more modern language models into the RealTabFormer framework does not yield improvements in synthetic data quality. We hypothesize that this performance gap stems from a mismatch between model architecture and hyperparameter configuration: the original models were carefully tuned for the framework, whereas the newer

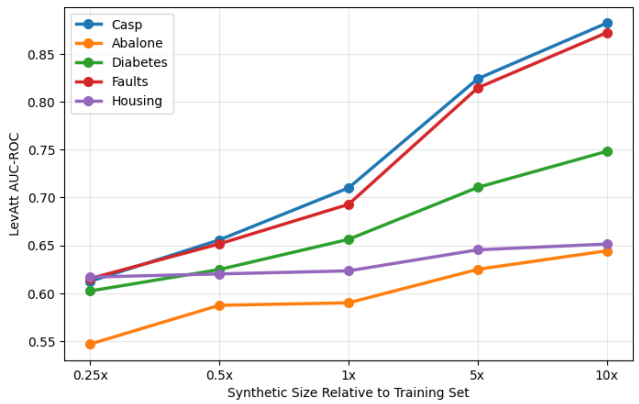


Figure 4: LevAtt AUC-ROC for various datasets generated by RealTabFormer with increasing synthetic dataset sizes relative to the training set.

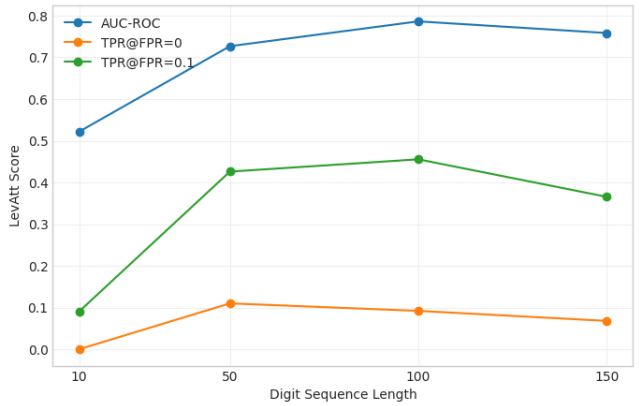


Figure 5: LevAtt performance on RealTabFormer at various training digit sequence lengths.

models likely require different optimization settings to fully realize their capacity.

5 Defenses Against Levenshtein Attack

The efficacy of LevAtt against LLM-based tabular models motivates us to explore methodologies that can defend generated synthetic data. Here, recognizing that LevAtt gains adversarial advantage from replicated patterns in strings of digits, we devise two strategies based on introducing controlled "noise" into string sequences. The first is Digit Modifier (DM), a post-processing method that operates independently of the generative model and is applied after the synthetic data have been produced. The second protection strategy is Tendency-based Logit Processor (TLP), a method compatible with any open-source LLM that strategically perturbs digits at sample time.

Table 3: Model Privacy and Fidelity Tradeoffs. We report the (Mean ± STD) of LevAtt's AUC, Maximum Mean Discrepency, Wasserstein Distance, and Downstream ML AUC for each model over all datsets as well as the Pearson Correlation of all metrics over all runs with LevAtt AUC.

Model	Lev AUC	MMD ↓	Wass ↓	ML AUC ↑
RealTabFormer	**0.65 ± 0.04**	0.01 ± 0.00	**0.13 ± 0.17**	**0.92 ± 0.11**
LLaMA-1B	0.54 ± 0.03	0.03 ± 0.03	0.31 ± 0.34	0.89 ± 0.14
LLaMA-3.2-3B	0.56 ± 0.03	0.02 ± 0.02	0.33 ± 0.48	0.83 ± 0.12
GPT-2	0.53 ± 0.04	0.01 ± 0.00	0.45 ± 0.65	0.75 ± 0.19
GReaT	0.53 ± 0.03	0.01 ± 0.00	7.00 ± 23.44	0.83 ± 0.26
Mistral	0.53 ± 0.03	0.02 ± 0.01	16.40 ± 33.60	0.64 ± 0.16
Qwen-3B	0.56 ± 0.05	0.01 ± 0.00	0.20 ± 0.27	0.88 ± 0.16
TVAE	0.50 ± 0.01	0.01 ± 0.00	0.19 ± 0.26	0.88 ± 0.13
CTGAN	0.49 ± 0.01	0.01 ± 0.00	0.24 ± 0.34	0.85 ± 0.09
DP-2Stage	0.50 ± 0.02	0.04 ± 0.06	1.08 ± 0.57	0.50 ± 0.01
Corr. Lev AUC	–	−0.163	−0.084	0.388

5.1 Digit Modifier

We propose a Digit Modifier (DM), a method for injecting controlled randomness into tabular data by perturbing numerical digits. Motivated by bit-flipping techniques for adding noise to binary representations of relational databases [2, 3], DM operates by replacing tokens in a record's tokenized sequence with alternatives sampled from a noise distribution. When tokens represent numerical values, these substitutions correspondingly alter the digits of the underlying sequences, yielding perturbed records. In this sense, DM can be viewed as a principled method for randomly replacing digits within a record.

To balance data fidelity and protection, we parameterize the mechanism as $\mathrm{DM}(p_{min}, p_{max})$ with $0 \leq p_{min} < p_{max} \leq 1$. For a numerical column $\mathbf{X_i}$ and entry $\mathbf{x_{k,i}}$, a probability function $g : \mathbf{x_{k,i}} \times \mathbf{X_i} \to [p_{min}, p_{max}]$ assigns digit-flipping probabilities such that larger-magnitude entries receive higher perturbation probabilities. Each digit of $\mathbf{x_{k,i}}$ is then independently flipped according to $g(\mathbf{x_{k,i}}, \mathbf{X_i})$, while smaller-magnitude values—being more sensitive—undergo smaller changes to preserve fidelity. In our experiment, we first define

$$M(\mathbf{X_i}) = \max_{\mathbf{x_{k,i}} \in \mathbf{X_i}} |\mathbf{x_{k,i}}|,$$

which represents the largest absolute value in the numerical column. We can then constructed the probability function by

$$g(\mathbf{x_{k,i}}, \mathbf{X_i}; p_{\min}, p_{\max}) = p_{\min} + (p_{\max} - p_{\min}) \frac{|\mathbf{x_{k,i}}|}{M(\mathbf{X_i})}.$$

At a high level, DM operates by iterating through each numerical entry in a tabular record, computing its digit-flipping probability using the function g, and independently perturbing each digit according to that probability. This produces a randomized yet fidelity-preserving transformation in which large-magnitude values receive stronger perturbations while small values remain close to their originals. We summarize the overall DM procedure in Algorithm 1.

5.2 Tendency-Based Logit Processor

We additionally propose Tendency-based Logit Processor (TLP), a mechanism for injecting controlled noise into synthetic data by perturbing the generator's logits at inference time. The TLP(t)

Algorithm 1 Digit Modifier (DM)

Input: Training dataset $D = \{\mathbf{x}_k\}_{k=1}^{n}$, generator G, parameters $0 \le p_{\min} < p_{\max} \le 1$, probability function g
Output: Perturbed synthetic dataset $\tilde{D}_{\text{syn}}$
Train generator G on dataset D
Generate synthetic dataset $D_{\text{syn}} = \{\mathbf{x}_k\}_{k=1}^{n'}$ using G
Let $\mathcal{N} \subseteq \{1, \ldots, d\}$ denote the indices of numerical columns
for all $i \in \mathcal{N}$ **do**
  $\mathbf{X}_i \leftarrow \{x_{1,i}, x_{2,i}, \ldots, x_{n',i}\}$ ▷ Column i values
  $M(\mathbf{X}_i) \leftarrow \max_{\mathbf{x}_{k,i} \in \mathbf{X}_i} |\mathbf{x}_{k,i}|$ ▷ Maximum absolute value
end for
Initialize $\tilde{D}_{\text{syn}} \leftarrow D_{\text{syn}}$
for all $k \in \{1, \ldots, n'\}$ **do** ▷ For each record
  for all $i \in \mathcal{N}$ **do** ▷ For each numerical column
    $p \leftarrow g(\mathbf{x}_{k,i}, \mathbf{X}_i; p_{\min}, p_{\max})$ ▷ Compute flip probability
    Let $\mathcal{R}(\mathbf{x}_{k,i})$ denote the set of digit positions in $\mathbf{x}_{k,i}$
    for all $r \in \mathcal{R}(\mathbf{x}_{k,i})$ **do** ▷ For each digit position
      Sample $b \sim \text{Bernoulli}(p)$
      if $b = 1$ **then**
        Let d_r be the digit at position r in $\mathbf{x}_{k,i}$
        Sample $d'_r \sim \text{Uniform}(\{0, 1, \ldots, 9\} \setminus \{d_r\})$
        Replace digit at position r in $\tilde{\mathbf{x}}_{k,i}$ with d'_r
      end if
    end for
  end for
end for
return $\tilde{D}_{\text{syn}}$

selectively amplifies lower-valued logits while suppressing higher-valued ones, making the generator more likely to select tokens that were originally less probable. The strength of this effect is controlled by the tendency parameter t: higher values of t induce stronger curvature, increasing the randomness of the generated sequence. In this way, TLP(t) acts as a principled method for introducing variability into synthetic outputs while still preserving the generator's learned distribution.

Formally, TLP(t) first maps the generator's logits $l = (l_1, l_2, ., l_k)$ into the range $[0, 1]$ using a shifted min–max scaler S_l. Specifically, let $m_l = \min_j l_j$, $M_l = \max_j l_j$, and $\varepsilon > 0$ (small constant). Then, we define

$$\left[S_l(l_1, l_2, \ldots, l_n)\right]_i = \frac{l_i - m_l}{M_l - m_l + \varepsilon}, \qquad i = 1, \ldots, k.$$

Similarly, we can also define S_l^{-1} componentwise as

$$\left[S_l^{-1}(s_1, s_2, \ldots, s_n)\right]_i = m_l + s_i\,(M_l - m_l + \varepsilon), \qquad i = 1, \ldots, n,$$

which maps processed logits back to their original scale. Next, TLP applies a monotone increasing, concave curving function $f_t : [0, 1] \to [0, 1]$, parameterized by t and satisfying $f_t(0) = 0$, to each scaled logit. Finally, the processed logits are transformed back to their original scale using the inverse scaler S_l^{-1}. The design of f_t is central to TLP(t). Monotonicity ensures that the relative order of logits is preserved, so high-probability tokens remain more likely than lower-probability ones, allowing controlled noise injection without overwhelming the generator's learned signal. Concavity, combined with $f_t(0) = 0$, guarantees that lower logits are curved

Algorithm 2 Tendency-Based Logit Processor (TLP)

Input: Training dataset D, generator G, tendency parameter $t > 0$, curving function $f_t : [0, 1] \to [0, 1]$
Output: Synthetic sequence generated by G with TLP applied
Train generator G on dataset D
Initialize output sequence $\mathbf{y} \leftarrow \emptyset$
while generation not complete **do**
  Compute raw logits $\mathbf{l} = (l_1, \ldots, l_k) \leftarrow G(\mathbf{y})$ ▷ k = vocabulary size
  $m_l \leftarrow \min_{j \in [k]} l_j, \quad M_l \leftarrow \max_{j \in [k]} l_j$
  $\mathbf{s} \leftarrow S_l(\mathbf{l})$ where $s_i = \frac{l_i - m_l}{M_l - m_l + \varepsilon}$ for $i \in [k]$ ▷ Scale to $[0, 1]$
  $\tilde{\mathbf{s}} \leftarrow f_t(\mathbf{s})$ where $\tilde{s}_i = f_t(s_i)$ for $i \in [k]$ ▷ Apply curving function
  $\tilde{\mathbf{l}} \leftarrow S_l^{-1}(\tilde{\mathbf{s}})$ where $\tilde{l}_i = m_l + \tilde{s}_i(M_l - m_l + \varepsilon)$ for $i \in [k]$ ▷ Rescale
  $\mathbf{p} \leftarrow \text{softmax}(\tilde{\mathbf{l}})$ where $p_i = \frac{\exp(\tilde{l}_i)}{\sum_{j=1}^{k} \exp(\tilde{l}_j)}$
  Sample token $y_{\text{next}} \sim \text{Categorical}(\mathbf{p})$
  $\mathbf{y} \leftarrow \mathbf{y} \cup \{y_{\text{next}}\}$
end while
return $\mathbf{y}$

upward, increasing their chance of being sampled. Too see why we need concavity, let's fix $0 < a < b \le 1$. Since $a = \theta b$ for some $\theta \in (0, 1)$, concavity of f_t implies

$$f_t(a) = f_t(\theta b + (1 - \theta) \cdot 0) \;\ge\; \theta f_t(b) + (1 - \theta) f_t(0) = \theta f_t(b),$$

where we used $f(0) = 0$. Dividing both sides by $a = \theta b$ gives

$$\frac{f_t(a)}{a} \;\ge\; \frac{f_t(b)}{b}.$$

Thus, the ratio $f_t(x)/x$ is non-increasing in x. This means smaller logits receive a proportionally larger boost compared to larger logits. Therefore, f_t preserves the ordering of the logits (by monotonicity) while compressing their differences (by concavity), which corresponds to a "curving-up" transformation that favors lower logits.

In particular, the curving function we use in our experiment is $f_t(x) = x^{\frac{1}{t}}$. Figure 10 below demonstrates the graph of our f_t under different t. Figure 6 shows how f_t transform the distribution of logits so that lower logits get higher probability of being selected. We summarize the overall procedure of TLP in Algorithm 2. Please refer to appendix B for more details on DM and TLP.

5.3 Evaluating Defense Mechanisms

To evaluate the effectiveness of DM and TLP, we use RealTabFormer, the SFT model exhibiting the highest degree of privacy leakage from LevAtt, and measure how much privacy improvement each method provides against LevAtt while preserving synthetic data fidelity. Here, we use two high privacy leakage datasets for RealTabFormer: a simulated dataset from a Multivariate Gaussian N(300, 5) designed to have 100 digits, and the CASP dataset. We copy the experiment design of Section 4.1.2 and sample models at varying synthetic set sizes to induce different levels of privacy leakage in the synthetic datasets.

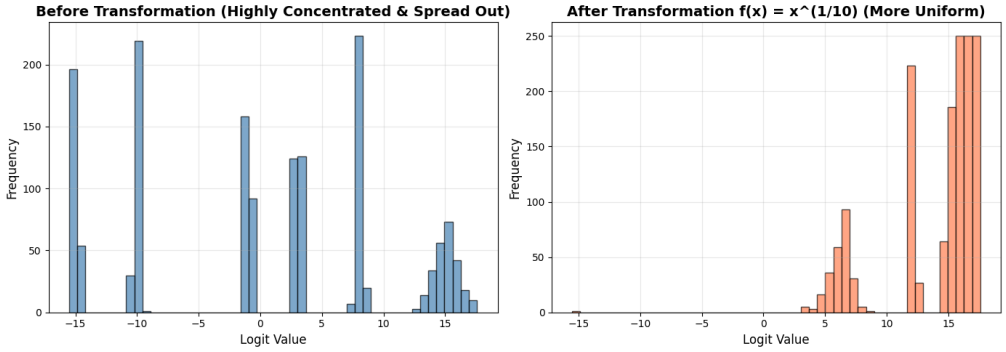


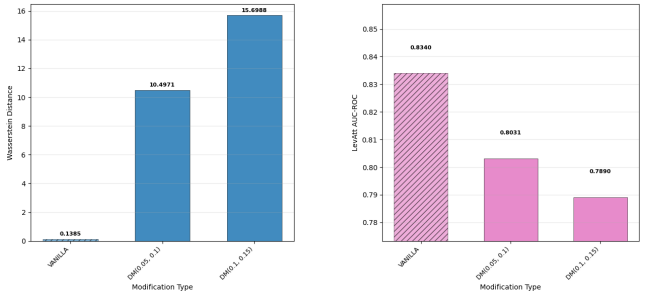
Figure 6: Effect of the TLP transformation on logit distributions. Before transformation (left), lower logits are tightly concentrated near the bottom of the range and have little chance of being selected. After applying the TLP function $f(x) = x^{1/10}$ (right), the transformation increases their relative magnitude of lower value logits, making them more likely to be sampled.

We then apply DM and TLP across different scaler and tendency parameter levels respectively, and assess performance in terms of privacy (LevAtt AUC-ROC and TPR@FPR=0.1), fidelity (Wasserstein Distance and Maximum Mean Discrepancy (MMD) between training and synthetic datasets), and utility (RMSE of XGBoost models trained on synthetic data and evaluated on real holdout data) [40].

Overall, we find that while effective in reducing privacy leakage, DM suffers large fidelity costs. In Figure 7, we applied DM to the simulation dataset, which is highly vulnerable to LevAtt (showing an MIA AUC-ROC above 83% without any protection). As we increased the amount of noise injected into the synthetic data, LevAtt's AUC-ROC decreased. However, the fidelity gap measured by Wasserstein between the original training data and the noised synthetic data rose sharply. This sharp trade-off stems from the simulation dataset's small dynamic range and low variance - even small amounts of noise push values into out-of-domain regions. This illustrates that while DM is convenient- it can be applied post-hoc to any synthetic dataset- it is not an elegant protection mechanism: its agnostic approach to noise injection cannot respect the structural constraints that make synthetic data useful.

On the other hand, TLP- when equipped with a tuned tendency parameter t-controllably reduces attack efficacy while preserving the fidelity between the processed synthetic data and the training data. In our experiment, we begin with a small t and evaluate its privacy protection effect. If the resulting synthetic data does not meet the privacy threshold (LevAtt AUC–ROC below 0.55 or TPR below 0.125 at FPR = 0.1), we increment t and repeat the procedure, stopping once we identify the smallest t that satisfies the criterion. Across both the highly unprivate simulation setting and the CASP dataset, this tuned TLP reduces LevAtt's AUC from as high as 0.79 to 0.55 and drives the TPR@FPR=0.1 from 0.48 to below 0.125, all while incurring virtually no penalty in the Maximum Mean Discrepancy of the TLP-generated data across all synthetic data sizes (see Figure 8).

TLP not only preserves the fidelity of the training dataset while meeting the privacy threshold, but also maintains the downstream utility of the CASP dataset. To evaluate this, we train XGBoost models on three versions of the data: real training subsets, vanilla synthetic data generated without protection, and TLP-processed synthetic data that satisfies the privacy requirement (LevAtt AUC-ROC below 0.55 or TPR below 0.125 at FPR = 0.1). For each training size, model performance is assessed on the same held-out real test set using RMSE, and we observe that the utility degradation remains within a reasonable range (see Figure 11).



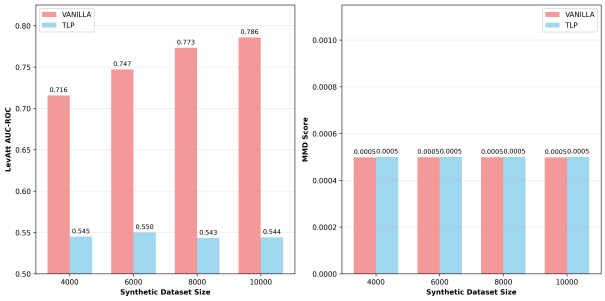
(a) Wasserstein distances between training and synthetic datasets.

(b) LevAtt AUC-ROC of synthetic datasets evaluated with an equal number of training and non-training records.

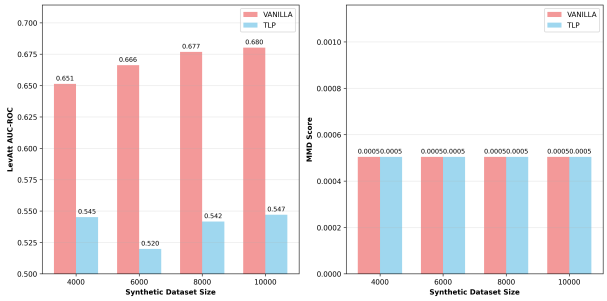
Figure 7: Privacy–fidelity comparison of DM on RealTabFormer synthetic data from the simulated dataset (Section 5.3). Vanilla corresponds to plain sampling without protection. Panel (a) reports Wasserstein distances; panel (b) shows LevAtt AUC-ROC. While DM is able to induce reductions in privacy leakage, the resulting synthetic data are of low fidelity.

6 Discussion

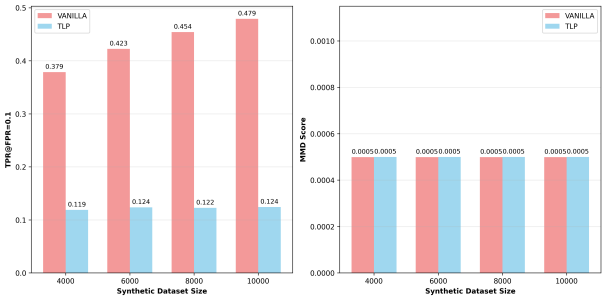
To contextualize our findings and their implications, we discuss LevAtt's broader significance, evaluate the effectiveness of proposed defenses, examine what LLMs truly learn from tabular data, and outline limitations and future research directions.



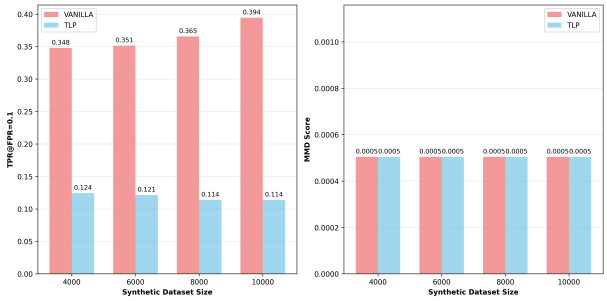
(a) Simulation Dataset- LevAtt AUC and MMD



(b) CASP Dataset- LevAtt AUC and MMD



(c) Simulation Dataset- LevAtt TPR@FPR=0.1 and MMD



(d) CASP Dataset- LevAtt TPR@FPR=0.1 and MMD

Figure 8: Privacy–fidelity trade-off of TLP on RealTabFormer synthetic data. We plot the AUC and Maximum Mean Discrepancy (MMD) for Vanilla and TLP-protected RealTabFormer models in (a)-(b). For both simulation (Section 5.3) and CASP datasets, TLP consistently reduces privacy leakage (AUC-ROC to ∼55%) while creating synthetic data that matches the MMD of unmodified Vanilla data at various sizes. This trend holds as well for TPR@FPR=0.1 in (c)-(d) where TLP successfully reduces the TPR@FPR=0.1 to below 0.125 without loss to fidelity.

6.1 LevAtt and the Privacy of LLM-Based Synthetic Data Generation

LevAtt reveals that LLM-based tabular data generation is uniquely unsafe relative to conventional deep learning approaches. While being a simple string-similarity attack operating under an extremely restrictive threat model, LevAtt uncovers that state-of-the-art ICL models can catastrophically leak training membership by copying sequential patterns of digits and text from prompt exemplar examples. This phenomenon was further demonstrated in SFT-generators, revealing that the base implementation of RealTabFormer—a method recognized for its privacy-preserving capabilities—was susceptible to significant privacy leakage. In contrast, conventional tabular generators such as CT-GAN and TVAE were resistant to string-based attacks, and LevAtt exhibited low correlation with feature-space oriented MIAs.

This vulnerability stems from fundamental differences in how LLMs generate synthetic data. Unlike GANs and VAEs that model joint distributions directly in the feature space, LLMs decompose generation into sequential token prediction, where each digit is conditioned on previously generated values. This autoregressive process creates opportunities for the model to reproduce memorized digit sequences, particularly when training data contains repeated patterns, long numeric strings, or low-variance columns—structural characteristics common in real-world tabular datasets. These results highlight that autoregressive token-based generation in LLMs exposes a distinct attack surface not shared by other deep learning architectures, emphasizing the need for dedicated privacy audits in this domain.

6.2 Sampling Defenses for LLMs

While highly deployable and powerful, LevAtt can be defeated. In this work, we proposed an intuitive post-hoc defense called Digit Modifier (DM), which alters digits after generation. However, we found that DM failed to preserve the fidelity of the synthetic dataset. In contrast, our Tendency-based Logit Processor (TLP) successfully reduced LevAtt's AUC-ROC to below 55% and TPR to below 12.5% at FPR=10% across both simulation and real-world datasets, while maintaining Maximum Mean Discrepancy nearly identical to vanilla generation. DM, despite achieving similar privacy improvements, incurred substantial fidelity costs, with Wasserstein distances increasing sharply as noise injection intensified.

The key advantage of TLP lies in its integration with the generation process itself. By perturbing logits during inference, TLP allows the model to maintain coherent dependencies across features while strategically introducing uncertainty in high-confidence digit predictions. DM, operating post-hoc, must blindly flip digits without access to the model's learned correlations, making it difficult to balance privacy protection with fidelity preservation. This is

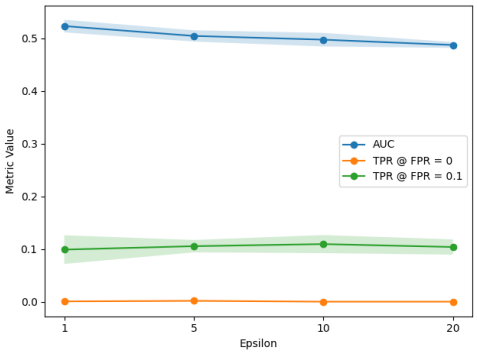


Figure 9: LevAtt performance on the Diabetes dataset at various ϵ levels for DP2Stage.

particularly problematic for datasets with narrow distributions or tight inter-feature dependencies, where even minor perturbations can push synthetic samples into low-density regions. TLP can be appended to any open-source LLM through the Hugging Face API, effectively controlling privacy leakage by smoothing the logits of digits with disproportionately high probabilities without substantially altering the statistical fidelity of the resulting synthetic data.

6.3 Differential Privacy as Defense

To mitigate the privacy risks exposed by LevAtt, we evaluate the effectiveness of implementing Differential Privacy (DP) during the training process. Our results indicate that across all datasets (See Table 2, DP2Stage successfully defeats LevAtt. Indeed, we plot LevAtt's performance on the Diabetes dataset at varying levels of ϵ from DP2Stage in Figure 9 and further find that all ϵ levels protect against LevAtt. However, this defense at training time comes at a substantial cost to data fidelity and utility as Table 3 shows that DP2Stage had the highest Maximum Mean Discrepancy value and the worst ML AUC over all other models.

The application of DP in the context of Large Language Models (LLMs) faces two critical hurdles: first, DP methods are fundamentally incompatible with In-Context Learning (ICL), as ICL lacks a traditional training stage to privatize, necessitating our defenses at sample time instead. Second, growing evidence suggests LLMs are often pre-trained on common open-source benchmarking datasets [6]; if a model's weights have already been exposed to the target data prior to the private training stage, the technical validity of the DP guarantee is undermined. While DP2Stage offers superior protection against string-matching attacks like LevAtt, the trade-off between privacy and utility remains a primary bottleneck for high-fidelity tabular generation.

6.4 On the Nature of LLM-Based Tabular Learning

While TLP provides an effective defense against LevAtt, the need for such inference-time mitigation raises a broader question: *What are these models exactly learning?* A prevailing assumption is that LLM-based tabular generators approximate the joint distribution of the training set T through sequential string modeling, similar to conventional generative approaches. However, our findings suggest that in some cases these models behave more like perturbation mechanisms, producing outputs that closely resemble training examples with only minor modifications. This behavior naturally yields high fidelity and downstream utility but does so precisely because of its proximity to the training data—thereby exposing the system to privacy leakage.

This question is not merely theoretical—our experimental results provide concrete evidence of memorization-like behavior. The perfect membership classification achieved by LevAtt on some TabPFN-V2 runs, combined with the scaling of privacy leakage with model size, suggests that larger models increasingly rely on storing and retrieving training examples rather than abstracting general patterns. Furthermore, our finding that privacy leakage increases with both the volume of synthetic data generated (Figure 4) and the length of digit sequences in training data (Figure 5) aligns more closely with a retrieval mechanism than with principled distribution learning.

This question echoes similar debates in natural language processing, where LLMs demonstrably memorize training data under certain conditions [11]. However, tabular data may be particularly susceptible to memorization due to its rigid structure and lack of paraphrase. In natural language, identical semantic content can be expressed in countless surface forms, creating ambiguity about whether a model has memorized a specific sentence or learned underlying concepts. Tabular data offers no such ambiguity—a sequence of digits either matches or does not. This rigidity may push LLM-based tabular generators toward memorization even when natural language applications achieve genuine generalization. Understanding when LLMs genuinely learn tabular distributions versus when they rely on approximate memorization remains an important direction for future work.

6.5 Limitations

While LevAtt and our proposed defenses demonstrate significant privacy vulnerabilities and mitigation strategies, several limitations warrant discussion.

First, LevAtt operates under a conservative No-box threat model, which—while demonstrating that privacy leakage occurs even under minimal adversarial assumptions—likely underestimates the true privacy risk in other possible scenarios. More permissive threat models that grant adversaries knowledge of model implementation details, training hyperparameters, or access to reference datasets would enable substantially more powerful attacks. For instance, knowledge of the specific tokenization scheme or access to auxiliary data could allow adversaries to adapt existing LLM-based membership inference techniques from natural language to tabular domains, potentially achieving even higher attack success rates than those reported here.

Second, while DM and TLP effectively reduce LevAtt's success rate, neither method provides formal privacy guarantees comparable to differential privacy. Both defenses are empirically validated against a specific attack rather than offering provable protection against arbitrary privacy audits. Consequently, more sophisticated attacks—particularly those that exploit model properties beyond string similarity—may successfully defeat these defenses [18, 19, 33].

Third, we do not address adaptive adversaries who are aware that protective mechanisms are in place. An adversary with knowledge of TLP deployment, for example, might develop attacks that specifically target the statistical artifacts introduced by logit perturbation, or exploit correlations that remain intact despite digit modifications. Such adaptive strategies could potentially circumvent our defenses, highlighting the need for more robust, defense-aware attack evaluations and iterative improvement of protection mechanisms.

Finally, despite its effectiveness, LevAtt's performance is contingent upon the synthetic generator's adherence to a consistent string representation of the underlying data. Because Levenshtein distance operates at the character level, it can be sensitive to formatting discrepancies; for instance, if a model reproduces a memorized float like "17.5" as "17.500," the resulting edit distance may obscure the underlying membership signal despite the numerical values being identical. Furthermore, while the models in this study do not utilize open text, such data types present a unique challenge for string-based attacks. Unlike the rigid schemas of numerical or categorical fields, open text allows for semantic paraphrasing where the same information can be expressed through entirely different character sequences. In such instances, LevAtt might fail to detect memorized information if the surface form is altered, suggesting that extending this framework to unstructured text would require augmenting character-level metrics with semantic similarity measures.

6.6 Future Work

Our findings open several avenues for future research aimed at both strengthening the understanding of privacy leakage in LLM-based tabular data generators and developing more robust protective mechanisms.

A first direction is the exploration of richer threat models beyond the No-box setting. While our results show that even minimally informed adversaries can perform highly accurate membership inference, more permissive models may reveal additional vulnerabilities. Future work could investigate attacks that leverage model internals, surrogate data, or tokenization details, and evaluate how such enhanced adversarial capabilities affect memorization dynamics in structured data domains. This includes adapting or extending gradient-based or embedding-space attacks from NLP to tabular contexts, potentially uncovering deeper patterns of leakage.

Second, our proposed defenses—while effective against LevAtt—lack provable privacy guarantees. Future research should aim to establish theoretical foundations for privacy in LLM-based tabular generation, analogous to differential privacy frameworks used in classical synthetic data generation. One promising direction is designing logit- or representation-level perturbation mechanisms that preserve distributional fidelity while offering quantifiable bounds on memorization risk. Similarly, exploring how architectural choices, training objectives, or regularization schemes influence memorization could inform principled defense design.

7 Conclusion

In this work we introduce LevAtt, a No-box threat model MIA that exposes substantial privacy risk for in-context learning and supervised-finetuned LLM-based tabular data generators. LevAtt shows that LLMs are vulnerable to memorization from the structured, often duplicated patterns of tabular data. By attacking the string encodings of autoregressively generated tabular data, LevAtt finds unique adversarial signal compared to existing methods. While less restrictive threat models would likely lead to a better attack, we believe No-box carries a powerful message: an attack with the minimal assumptions reasonably possible for an adversary can perfectly classify training membership in state-of-the-art generators. Lastly propose two defenses against LevAtt, showing that Tendency-Based Logit Processor can effectively defeat LevAtt with minimal loss in synthetic data fidelity. Future research directions could involve developing even more powerful attacks under less restrictive threat models, finding more efficient and provable defenses, and studying mechanistically how LLMs learn and represent tabular distributions.

8 Statement of Ethics

The potential for adversaries to determine whether an individual's data was included in the original dataset presents significant privacy risks, especially in fields such as healthcare, finance, and social sciences, where sensitive personal information is commonly used. Synthetic data that fails to sufficiently mask membership information could inadvertently enable re-identification. Although this work introduces a method for assessing such risks, its primary objective is to empower researchers and practitioners to perform more rigorous privacy evaluations before deploying synthetic datasets. We emphasize that adversarial approaches are essential for advancing the development of robust privacy-preserving systems.

Acknowledgments

The authors used LLMs to assist in the programming of our experiments, designing data visualizations, making the codebase more user-friendly, finding clearer phrasings in our writing, and formatting Latex code. All revisions by LLMs were checked and validated by the authors.

This work was financially supported by NSF – CNS (2247795), Office of Naval Research, (ONR N00014-22-1-2680), a gift from CISCO, and the National AI Research Resource (NAIRR-250187).

References

[1] Tejumade Afonja, Hui-Po Wang, Raouf Kerkouche, and Mario Fritz. 2025. DP-2Stage: Adapting Language Models as Differentially Private Tabular Data Generators. https://openreview.net/forum?id=6nBIweDYzZ

[2] Rakesh Agrawal and Jerry Kiernan. 2002. Watermarking relational databases. In *VLDB'02: Proceedings of the 28th International Conference on Very Large Databases*. Morgan Kaufmann, Hong Kong, China, 155–166.

[3] Abd S Alfagi, A Abd Manaf, B Hamida, S Khan, and Ali A Elrowayati. 2016. Survey on relational database watermarking techniques. *ARPN-JEAS* 11 (2016), 422–423.

[4] Ankur Ankan and Abinash Panda. 2015. pgmpy: Probabilistic Graphical Models using Python. In *Proceedings of the Python in Science Conference (SciPy)*. SciPy, Austin, TX, USA, 6–11. https://doi.org/10.25080/majora-7b98e3ed-001

[5] Samuel A. Assefa, Danial Dervovic, Mahmoud Mahfouz, Robert E. Tillman, Prashant Reddy, and Manuela Veloso. 2021. Generating synthetic data in finance: opportunities, challenges and pitfalls. In *Proceedings of the First ACM International Conference on AI in Finance* (New York, New York) *(ICAIF '20)*. Association for Computing Machinery, New York, NY, USA, Article 44, 8 pages. https://doi.org/10.1145/3383455.3422554

[6] Sebastian Bordt, Harsha Nori, Vanessa Cristiny Rodrigues Vasconcelos, Besmira Nushi, and Rich Caruana. 2024. Elephants Never Forget: Memorization and Learning of Tabular Data in Large Language Models. https://openreview.net/forum?id=HLoWN6m4fS

[7] Vadim Borisov, Kathrin Seßler, Tobias Leemann, Martin Pawelczyk, and Gjergji Kasneci. 2023. Language Models are Realistic Tabular Data Generators. arXiv:2210.06280 [cs.LG] https://arxiv.org/abs/2210.06280

[8] Jessup Byun, Xiaofeng Lin, Joshua Ward, and Guang Cheng. 2025. Risk In Context: Benchmarking Privacy Leakage of Foundation Models in Synthetic Tabular Data Generation. arXiv:2507.17066 [cs.LG] https://arxiv.org/abs/2507.17066

[9] Nicholas Carlini, Steve Chien, Milad Nasr, Shuang Song, A. Terzis, and Florian Tramèr. 2021. Membership Inference Attacks From First Principles. , 1897-1914 pages. https://api.semanticscholar.org/CorpusID:244920593

[10] Nicholas Carlini, Daphne Ippolito, Matthew Jagielski, Katherine Lee, Florian Tramer, and Chiyuan Zhang. 2023. Quantifying Memorization Across Neural Language Models. arXiv:2202.07646 [cs.LG] https://arxiv.org/abs/2202.07646

[11] Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, Alina Oprea, and Colin Raffel. 2021. Extracting Training Data from Large Language Models. arXiv:2012.07805 [cs.CR] https://arxiv.org/abs/2012.07805

[12] Dingfan Chen, Ning Yu, Yang Zhang, and Mario Fritz. 2020. GAN-Leaks: A Taxonomy of Membership Inference Attacks against Generative Models. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security (CCS '20)*. ACM, Virtual Event, USA, 343–362. https://doi.org/10.1145/3372297.3417238

[13] Tianzhe Chu, Yuexiang Zhai, Jihan Yang, Shengbang Tong, Saining Xie, Dale Schuurmans, Quoc V Le, Sergey Levine, and Yi Ma. 2025. SFT Memorizes, RL Generalizes: A Comparative Study of Foundation Model Post-training. In *Forty-second International Conference on Machine Learning*, Vol. TBD. PMLR, Vancouver, Canada, XXXX–YYYY. https://openreview.net/forum?id=dYur3yabMj

[14] Conor Durkan, Artur Bekasov, Iain Murray, and George Papamakarios. 2019. Neural spline flows. In *Advances in Neural Information Processing Systems*, Vol. 32. Curran Associates Inc., Vancouver, Canada, 7627–7638.

[15] Sebastian Felix Fischer, Matthias Feurer, and Bernd Bischl. 2023. OpenML-CTR23 – A curated tabular regression benchmarking suite. https://openreview.net/forum?id=HebAOoMm94

[16] Brendan Flanagan, Rwitajit Majumdar, and Hiroaki Ogata. 2022. Fine Grain Synthetic Educational Data: Challenges and Limitations of Collaborative Learning Analytics. *IEEE Access* 10 (03 2022), 26230–26241. https://doi.org/10.1109/ACCESS.2022.3156073

[17] Joao Fonseca and Fernando Bação. 2023. Tabular and latent space synthetic data generation: a literature review. *Journal of Big Data* 10 (07 2023). https://doi.org/10.1186/s40537-023-00792-7

[18] Wenjie Fu, Huandong Wang, Chen Gao, Guanghua Liu, Yong Li, and Tao Jiang. 2024. Membership inference attacks against fine-tuned large language models via self-prompt calibration. In *Proceedings of the 38th International Conference on Neural Information Processing Systems* (Vancouver, BC, Canada) *(NIPS '24)*. Curran Associates Inc., Red Hook, NY, USA, Article 4290, 30 pages.

[19] Filippo Galli, Luca Melis, and Tommaso Cucinotta. 2024. Noisy Neighbors: Efficient membership inference attacks against LLMs. In *Proceedings of the Fifth Workshop on Privacy in Natural Language Processing*, Ivan Habernal, Sepideh Ghanavati, Abhilasha Ravichander, Vijayanta Jain, Patricia Thaine, Timour Igamberdiev, Niloofar Mireshghallah, and Oluwaseyi Feyisetan (Eds.). Association for Computational Linguistics, Bangkok, Thailand, 1–6. https://aclanthology.org/2024.privatenlp-1.1/

[20] Mauro Giuffré and Dennis L. Shung. 2023. Harnessing the power of synthetic data in healthcare: innovation, application, and privacy. https://api.semanticscholar.org/CorpusID:263802405

[21] Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurelien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Roziere, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, Danny Wyatt, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Francisco Guzmán, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Govind Thattai, Graeme Nail, Gregoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel Kloumann, Ishan Misra, Ivan Evtimov, Jack Zhang, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Karthik Prasad, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, Khalid El-Arini, Krithika Iyer, Kshitiz Malik, Kuenley Chiu, Kunal Bhalla, Kushal Lakhotia, Lauren Rantala-Yeary, Laurens van der Maaten, Lawrence Chen, Liang Tan, Liz Jenkins, Louis Martin, Lovish Madaan, Lubo Malo, Lukas Blecher, Lukas Landzaat, Luke de Oliveira, Madeline Muzzi, Mahesh Pasupuleti, Mannat Singh, Manohar Paluri, Marcin Kardas, Maria Tsimpoukelli, Mathew Oldham, Mathieu Rita, Maya Pavlova, Melanie Kambadur, Mike Lewis, Min Si, Mitesh Kumar Singh, Mona Hassan, Naman Goyal, Narjes Torabi, Nikolay Bashlykov, Nikolay Bogoychev, Niladri Chatterji, Ning Zhang, Olivier Duchenne, Onur Çelebi, Patrick Alrassy, Pengchuan Zhang, Pengwei Li, Petar Vasic, Peter Weng, Prajjwal Bhargava, Pratik Dubal, Praveen Krishnan, Punit Singh Koura, Puxin Xu, Qing He, Qingxiao Dong, Ragavan Srinivasan, Raj Ganapathy, Ramon Calderer, Ricardo Silveira Cabral, Robert Stojnic, Roberta Raileanu, Rohan Maheswari, Rohit Girdhar, Rohit Patel, Romain Sauvestre, Ronnie Polidoro, Roshan Sumbaly, Ross Taylor, Ruan Silva, Rui Hou, Rui Wang, Saghar Hosseini, Sahana Chennabasappa, Sanjay Singh, Sean Bell, Seohyun Sonia Kim, Sergey Edunov, Shaoliang Nie, Sharan Narang, Sharath Raparthy, Sheng Shen, Shengye Wan, Shruti Bhosale, Shun Zhang, Simon Vandenhende, Soumya Batra, Spencer Whitman, Sten Sootla, Stephane Collot, Suchin Gururangan, Sydney Borodinsky, Tamar Herman, Tara Fowler, Tarek Sheasha, Thomas Georgiou, Thomas Scialom, Tobias Speckbacher, Todor Mihaylov, Tong Xiao, Ujjwal Karn, Vedanuj Goswami, Vibhor Gupta, Vignesh Ramanathan, Viktor Kerkez, Vincent Gonguet, Virginie Do, Vish Vogeti, Vítor Albiero, Vladan Petrovic, Weiwei Chu, Wenhan Xiong, Wenyin Fu, Whitney Meers, Xavier Martinet, Xiaodong Wang, Xiaofang Wang, Xiaoqing Ellen Tan, Xide Xia, Xinfeng Xie, Xuchao Jia, Xuewei Wang, Yaelle Goldschlag, Yashesh Gaur, Yasmine Babaei, Yi Wen, Yiwen Song, Yuchen Zhang, Yue Li, Yuning Mao, Zacharie Delpierre Coudert, Zheng Yan, Zhengxing Chen, Zoe Papakipos, Aaditya Singh, Aayushi Srivastava, Abha Jain, Adam Kelsey, Adam Shajnfeld, Adithya Gangidi, Adolfo Victoria, Ahuva Goldstand, Ajay Menon, Ajay Sharma, Alex Boesenberg, Alexei Baevski, Allie Feinstein, Amanda Kallet, Amit Sangani, Amos Teo, Anam Yunus, Andrei Lupu, Andres Alvarado, Andrew Caples, Andrew Gu, Andrew Ho, Andrew Poulton, Andrew Ryan, Ankit Ramchandani, Annie Dong, Annie Franco, Anuj Goyal, Aparajita Saraf, Arkabandhu Chowdhury, Ashley Gabriel, Ashwin Bharambe, Assaf Eisenman, Azadeh Yazdan, Beau James, Ben Maurer, Benjamin Leonhardi, Bernie Huang, Beth Loyd, Beto De Paola, Bhargavi Paranjape, Bing Liu, Bo Wu, Boyu Ni, Braden Hancock, Bram Wasti, Brandon Spence, Brani Stojkovic, Brian Gamido, Britt Montalvo, Carl Parker, Carly Burton, Catalina Mejia, Ce Liu, Changhan Wang, Changkyu Kim, Chao Zhou, Chester Hu, Ching-Hsiang Chu, Chris Cai, Chris Tindal, Christoph Feichtenhofer, Cynthia Gao, Damon Civin, Dana Beaty, Daniel Kreymer, Daniel Li, David Adkins, David Xu, Davide Testuggine, Delia David, Devi Parikh, Diana Liskovich, Didem Foss, Dingkang Wang, Duc Le, Dustin Holland, Edward Dowling, Eissa Jamil, Elaine Montgomery, Eleonora Presani, Emily Hahn, Emily Wood, Eric-Tuan Le, Erik Brinkman, Esteban Arcaute, Evan Dunbar, Evan Smothers, Fei Sun, Felix Kreuk, Feng Tian, Filippos Kokkinos, Firat Ozgenel, Francesco Caggioni, Frank Kanayet, Frank Seide, Gabriela Medina Florez, Gabriella Schwarz, Gada Badeer, Georgia Swee, Gil Halpern, Grant Herman, Grigory Sizov, Guangyi, Zhang, Guna Lakshminarayanan, Hakan Inan, Hamid Shojanazeri, Han Zou, Hannah Wang, Hanwen Zha, Haroun Habeeb, Harrison Rudolph, Helen Suk, Henry Aspegren, Hunter Goldman, Hongyuan Zhan, Ibrahim Damlaj, Igor Molybog, Igor Tufanov, Ilias Leontiadis, Irina-Elena Veliche, Itai Gat, Jake Weissman, James Geboski, James Kohli, Janice Lam, Japhet Asher, Jean-Baptiste Gaya, Jeff Marcus, Jeff Tang, Jennifer Chan, Jenny Zhen, Jeremy Reizenstein, Jeremy Teboul, Jessica Zhong, Jian Jin, Jingyi Yang, Joe Cummings, Jon Carvill, Jon Shepard, Jonathan McPhie, Jonathan Torres, Josh Ginsburg, Junjie Wang, Kai Wu, Kam Hou U, Karan Saxena, Kartikay Khandelwal, Katayoun Zand, Kathy Matosich, Kaushik Veeraraghavan, Kelly Michelena, Keqian Li, Kiran Jagadeesh, Kun Huang, Kunal Chawla, Kyle Huang, Lailin Chen, Lakshya Garg, Lavender A, Leandro Silva, Lee Bell, Lei Zhang, Liangpeng Guo, Licheng Yu, Liron Moshkovich, Luca Wehrstedt, Madian Khabsa, Manav Avalani, Manish Bhatt, Martynas Mankus, Matan Hasson, Matthew Lennie, Matthias Reso, Maxim Groshev, Maxim Naumov, Maya Lathi, Meghan Keneally, Miao Liu, Michael L. Seltzer, Michal Valko, Michelle Restrepo, Mihir Patel, Mik Vyatskov, Mikayel Samvelyan, Mike Clark, Mike Macey, Mike Wang, Miquel Jubert Hermoso, Mo Metanat, Mohammad Rastegari, Munish Bansal, Nandhini Santhanam, Natascha Parks, Natasha White, Navyata Bawa, Nayan Singhal, Nick Egebo, Nicolas Usunier, Nikhil Mehta, Nikolay Pavlovich Laptev, Ning Dong, Norman Cheng, Oleg Chernoguz, Olivia Hart, Omkar Salpekar, Ozlem Kalinli, Parkin Kent, Parth Parekh, Paul Saab, Pavan Balaji, Pedro Rittner, Philip Bontrager, Pierre Roux, Piotr Dollar, Polina Zvyagina, Prashant Ratanchandani, Pritish Yuvraj, Qian Liang, Rachad Alao, Rachel Rodriguez, Rafi Ayub, Raghotham Murthy, Raghu Nayani, Rahul Mitra, Rangaprabhu Parthasarathy, Raymond Li, Rebekkah Hogan, Robin Battey, Rocky Wang, Russ Howes, Ruty Rinott, Sachin Mehta, Sachin Siby, Sai Jayesh Bondu, Samyak Datta, Sara Chugh, Sara Hunt, Sargun Dhillon, Sasha Sidorov, Satadru Pan, Saurabh Mahajan, Saurabh Verma, Seiji Yamamoto, Sharadh Ramaswamy, Shaun Lindsay, Shaun Lindsay, Sheng Feng, Shenghao Lin, Shengxin Cindy Zha, Shishir Patil, Shiva Shankar, Shuqiang Zhang, Shuqiang Zhang, Sinong Wang, Sneha Agarwal, Soji Sajuyigbe, Soumith Chintala, Stephanie Max, Stephen Chen, Steve Kehoe, Steve Satterfield, Sudarshan Govindaprasad, Sumit Gupta, Summer Deng, Sungmin Cho, Sunny Virk, Suraj Subramanian, Sy Choudhury, Sydney Goldman, Tal Remez, Tamar Glaser, Tamara Best, Thilo Koehler, Thomas Robinson, Tianhe Li, Tianjun Zhang, Tim Matthews, Timothy Chou, Tzook Shaked,

Varun Vontimitta, Victoria Ajayi, Victoria Montanez, Vijai Mohan, Vinay Satish Kumar, Vishal Mangla, Vlad Ionescu, Vlad Poenaru, Vlad Tiberiu Mihailescu, Vladimir Ivanov, Wei Li, Wenchen Wang, Wenwen Jiang, Wes Bouaziz, Will Constable, Xiaocheng Tang, Xiaojian Wu, Xiaolan Wang, Xilun Wu, Xinbo Gao, Yaniv Kleinman, Yanjun Chen, Ye Hu, Ye Jia, Ye Qi, Yenda Li, Yilin Zhang, Ying Zhang, Yossi Adi, Youngjin Nam, Yu, Wang, Yu Zhao, Yuchen Hao, Yundi Qian, Yunlu Li, Yuzi He, Zach Rait, Zachary DeVito, Zef Rosnbrick, Zhaoduo Wen, Zhenyu Yang, Zhiwei Zhao, and Zhiyu Ma. 2024. The Llama 3 Herd of Models. arXiv:2407.21783 [cs.AI] https://arxiv.org/abs/2407.21783

[22] Benjamin Hilprecht, Martin Härterich, and Daniel Bernau. 2019. Monte Carlo and Reconstruction Membership Inference Attacks against Generative Models. *Proceedings on Privacy Enhancing Technologies* 2019 (2019), 232 – 249. https://api.semanticscholar.org/CorpusID:199546273

[23] Noah Hollmann, Samuel Müller, Lennart Purucker, Arjun Krishnakumar, Max Körfer, Shi Bin Hoo, Robin Tibor Schirrmeister, and Frank Hutter. 2025. Accurate predictions on small data with a tabular foundation model. *Nature* 637, 8045 (2025), 319–326.

[24] Noah Hollmann, Samuel Müller, Lennart Purucker, Arjun Krishnakumar, Max Körfer, Shi Hoo, Robin Schirrmeister, and Frank Hutter. 2025. Accurate predictions on small data with a tabular foundation model. *Nature* 637 (01 2025), 319–326. https://doi.org/10.1038/s41586-024-08328-6

[25] Florimond Houssiau, James Jordon, Samuel N Cohen, Owen Daniel, Andrew Elliott, James Geddes, Callum Mole, Camila Rangel-Smith, and Lukasz Szpruch. 2022. Tapas: a toolbox for adversarial privacy auditing of synthetic data.

[26] Daphne Ippolito, Florian Tramer, Milad Nasr, Chiyuan Zhang, Matthew Jagielski, Katherine Lee, Christopher Choquette Choo, and Nicholas Carlini. 2023. Preventing Generation of Verbatim Memorization in Language Models Gives a False Sense of Privacy. In *Proceedings of the 16th International Natural Language Generation Conference*, C. Maria Keet, Hung-Yi Lee, and Sina Zarrieß (Eds.). Association for Computational Linguistics, Prague, Czechia, 28–53. https://doi.org/10.18653/v1/2023.inlg-main.3

[27] Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7B. arXiv:2310.06825 [cs.CL] https://arxiv.org/abs/2310.06825

[28] Nikhil Kandpal, Eric Wallace, and Colin Raffel. 2022. Deduplicating Training Data Mitigates Privacy Risks in Language Models. https://api.semanticscholar.org/CorpusID:246823128

[29] Jinhee Kim, Taesung Kim, and Jaegul Choo. 2024. EPIC: Effective Prompting for Imbalanced-Class Data Synthesis in Tabular Data Classification via Large Language Models. https://openreview.net/forum?id=d5cKDHCrFJ

[30] Akim Kotelnikov, Dmitry Baranchuk, Ivan Rubachev, and Artem Babenko. 2022. TabDDPM: Modelling Tabular Data with Diffusion Models. arXiv:2209.15421 [cs.LG]

[31] Vladimir I Levenshtein. 1966. Binary Codes Capable of Correcting Deletions, Insertions and Reversals. *Soviet Physics Doklady* 10 (February 1966), 707.

[32] Junwei Ma, Apoorv Dankar, George Stein, Guangwei Yu, and Anthony Caterini. 2024. TabPFGen – Tabular Data Generation with TabPFN. arXiv:2406.05216 [cs.LG] https://arxiv.org/abs/2406.05216

[33] Justus Mattern, Fatemehsadat Mireshghallah, Zhijing Jin, Bernhard Schoelkopf, Mrinmaya Sachan, and Taylor Berg-Kirkpatrick. 2023. Membership Inference Attacks against Language Models via Neighbourhood Comparison. In *Findings of the Association for Computational Linguistics: ACL 2023*, Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (Eds.). Association for Computational Linguistics, Toronto, Canada, 11330–11343. https://doi.org/10.18653/v1/2023.findings-acl.719

[34] Ryan McKenna, Brett Mullins, Daniel Sheldon, and Gerome Miklau. 2022. AIM: an adaptive and iterative mechanism for differentially private synthetic data. *Proc. VLDB Endow.* 15, 11 (July 2022), 2599–2612. https://doi.org/10.14778/3551793.3551817

[35] Matthieu Meeus, Florent Guepin, Ana-Maria Creţu, and Yves-Alexandre de Montjoye. 2024. *Achilles' Heels: Vulnerable Record Identification in Synthetic Data Publishing*. Springer Nature Switzerland, Cham, Switzerland, 380–399. https://doi.org/10.1007/978-3-031-51476-0_19

[36] Meta AI. 2024. LLaMA-3.3 70B Instruct Model. https://huggingface.co/meta-llama/Llama-3.3-70B-Instruct. Released December 6, 2024; accessed 2025-06-13.

[37] Gonzalo Navarro. 2001. A Guided Tour to Approximate String Matching. *Comput. Surveys* 33, 1 (2001), 31–88.

[38] OpenAI. 2024. GPT-4o Mini Model in Chat Completions API. https://platform.openai.com/docs/models/gpt-4o-mini. Released July 18, 2024; accessed 2025-06-13.

[39] Michael Platzer and Thomas Reutterer. 2021. Holdout-based empirical assessment of mixed-type synthetic data. *Frontiers in big Data* 4 (2021), 679939.

[40] Zhaozhi Qian, Bogdan-Constantin Cebere, and Mihaela van der Schaar. 2023. Synthcity: facilitating innovative use cases of synthetic data in different data modalities. https://doi.org/10.48550/ARXIV.2301.07573

[41] Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. 2025. Qwen2.5 Technical Report. arXiv:2412.15115 [cs.CL] https://arxiv.org/abs/2412.15115

[42] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. *Language Models are Unsupervised Multitask Learners*. Technical Report. OpenAI.

[43] Nabeel Seedat, Nicolas Huynh, Boris van Breugel, and Mihaela van der Schaar. 2024. Curated LLM: Synergy of LLMs and Data Curation for tabular augmentation in low-data regimes. In *Forty-first International Conference on Machine Learning*, Vol. 235. PMLR, Vienna, Austria, 44060–44092.

[44] Igor Shilov, Matthieu Meeus, and Yves-Alexandre de Montjoye. 2025. The Mosaic Memory of Large Language Models. arXiv:2405.15523 [cs.CL] https://arxiv.org/abs/2405.15523

[45] R. Shokri, M. Stronati, C. Song, and V. Shmatikov. 2017. Membership Inference Attacks Against Machine Learning Models. In *2017 IEEE Symposium on Security and Privacy (SP)*. IEEE Computer Society, Los Alamitos, CA, USA, 3–18. https://doi.org/10.1109/SP.2017.41

[46] Aivin V Solatorio and Olivier Dupriez. 2023. Realtabformer: Generating realistic relational and tabular data using transformers.

[47] Theresa Stadler, Bristena Oprisanu, and Carmela Troncoso. 2022. Synthetic Data – Anonymisation Groundhog Day. In *31st USENIX Security Symposium (USENIX Security 22)*. USENIX Association, Boston, MA, 1451–1468. https://www.usenix.org/conference/usenixsecurity22/presentation/stadler

[48] Namjoon Suh, Xiaofeng Lin, Din-Yin Hsieh, Mehrdad Honarkhah, and Guang Cheng. 2023. AutoDiff: combining Auto-encoder and Diffusion model for tabular data synthesizing. https://openreview.net/forum?id=XhxOCXlXSh

[49] Vibeke Binz Vallevik, Aleksandar Babic, Serena E. Marshall, Severin Elvatun, Helga M.B. Brøgger, Sharmini Alagaratnam, Bjørn Edwin, Narasimha R. Veeraragavan, Anne Kjersti Befring, and Jan F. Nygård. 2024. Can I trust my fake data – A comprehensive quality assessment framework for synthetic tabular data in healthcare. *International Journal of Medical Informatics* 185 (2024), 105413. https://doi.org/10.1016/j.ijmedinf.2024.105413

[50] Boris van Breugel, Hao Sun, Zhaozhi Qian, and Mihaela van der Schaar. 2023. Membership Inference Attacks against Synthetic Data through Overfitting Detection. arXiv:2302.12580 [cs.LG]

[51] Yuxin Wang, Duanyu Feng, Yongfu Dai, Zhengyu Chen, Jimin Huang, Sophia Ananiadou, Qianqian Xie, and Hao Wang. 2025. HARMONIC: harnessing LLMs for tabular data synthesis and privacy protection. In *Proceedings of the 38th International Conference on Neural Information Processing Systems* (Vancouver, BC, Canada) *(NIPS '24)*. Curran Associates Inc., Red Hook, NY, USA, Article 3179, 17 pages.

[52] Zhepeng Wang, Runxue Bao, Yawen Wu, Jackson Taylor, Cao Xiao, Feng Zheng, Weiwen Jiang, Shangqian Gao, and Yanfu Zhang. 2024. Unlocking Memorization in Large Language Models with Dynamic Soft Prompting. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (Eds.). Association for Computational Linguistics, Miami, Florida, USA, 9782–9796. https://doi.org/10.18653/v1/2024.emnlp-main.546

[53] Joshua Ward, Xiaofeng Lin, , Chi-Hua Wang, and Guang Cheng. 2025. SynthMIA: A Testbed for Auditing Privacy Leakage in Tabular Data Synthesis. arXiv:2509.18014 [cs.CR] https://arxiv.org/abs/2509.18014

[54] Joshua Ward, Chi-Hua Wang, and Guang Cheng. 2024. Data Plagiarism Index: Characterizing the Privacy Risk of Data-Copying in Tabular Generative Models. arXiv:2406.13012 [cs.LG] https://arxiv.org/abs/2406.13012

[55] Joshua Ward, Chi-Hua Wang, and Guang Cheng. 2025. Privacy Auditing Synthetic Data Release through Local Likelihood Attacks. arXiv:2508.21146 [cs.LG] https://arxiv.org/abs/2508.21146

[56] David S. Watson, Kristin Blesch, Jan Kapar, and Marvin N. Wright. 2023. Adversarial Random Forests for Density Estimation and Generative Modeling. In *Proceedings of The 26th International Conference on Artificial Intelligence and Statistics (Proceedings of Machine Learning Research, Vol. 206)*, Francisco Ruiz, Jennifer Dy, and Jan-Willem van de Meent (Eds.). PMLR, Valencia, Spain, 5357–5375. https://proceedings.mlr.press/v206/watson23a.html

[57] Jinhong Wu, Konstantinos Plataniotis, Lucy Liu, Ehsan Amjadian, and Yuri Lawryshyn. 2023. Interpretation for Variational Autoencoder Used to Generate Financial Synthetic Tabular Data. *Algorithms* 16 (02 2023), 121. https://doi.org/10.3390/a16020121

[58] Lei Xu, Maria Skoularidou, Alfredo Cuesta-Infante, and Kalyan Veeramachaneni. 2019. Modeling Tabular data using Conditional GAN. In *Advances in Neural Information Processing Systems*, Vol. 32. Curran Associates, Inc., Vancouver, Canada, 7335–7345. https://proceedings.neurips.cc/paper/2019/hash/254ed7d2de3b23ab10936522dd547b78-Abstract.html

[59] Jinsung Yoon, Lydia N Drumright, and Mihaela Van Der Schaar. 2020. Anonymization through data synthesis using generative adversarial networks (ads-gan). *IEEE journal of biomedical and health informatics* 24, 8 (2020), 2378–2388.
[60] Jinsung Yoon, James Jordon, and Mihaela van der Schaar. 2019. PATE-GAN: Generating Synthetic Data with Differential Privacy Guarantees. In *International Conference on Learning Representations*. OpenReview.net, New Orleans, LA, USA, 1–15. https://openreview.net/forum?id=S1zk9iRqF7
[61] Li Yujian and Liu Bo. 2007. A Normalized Levenshtein Distance Metric. *IEEE Trans. Pattern Anal. Mach. Intell.* 29, 6 (June 2007), 1091–1095. https://doi.org/10.1109/TPAMI.2007.1078
[62] Hengrui Zhang, Jiani Zhang, Zhengyuan Shen, Balasubramaniam Srinivasan, Xiao Qin, Christos Faloutsos, Huzefa Rangwala, and George Karypis. 2024. Mixed-Type Tabular Data Synthesis with Score-based Diffusion in Latent Space. In *The Twelfth International Conference on Learning Representations*. OpenReview.net, Vienna, Austria, 4Ay23yeuz0. https://openreview.net/forum?id=4Ay23yeuz0

A Appendix

A.1 In-Context Learning details

A.1.1 Generator Details.

(1) **TabPFN Generation [23]**: We followed the Prior Labs tutorial (https://priorlabs.ai/tutorials/unsupervised/) for unsupervised TabPFN. Each training split was loaded, shuffled, and divided into batches of 200 rows. Numeric features were cast to `float32`, while categorical variables were label-encoded (assigning unseen categories to −1). Columns with zero variance were removed prior to model fitting and reintroduced after sampling. For each batch, we fit TabPFN and generated synthetic rows using temperature $t = 1.0$ across three random permutations. Outputs were decoded, constant columns reattached, batches concatenated, and finally truncated to match the size of the original dataset.
(2) **LLaMA Generation [36]**: We used LLaMA 3.3 70B via the Groq API (https://console.groq.com/docs/models). Each training split was divided into batches of up to 32 rows, ensuring that all rows were fully included in the prompt. For each batch, we computed per-column summary statistics and serialized the data to CSV. We then queried `llama-3.3-70b-versatile` with temperature $t = 1.0$, requesting N rows in JSON format. If outputs contained parse errors or incorrect row counts, we retried the generation up to five times. Valid generations were concatenated, truncated, or re-prompted as necessary, and validated for type and dimensional consistency.[2]
(3) **GPT-4o-mini [38]**: We applied the same prompting and inference pipeline as with LLaMA 3.3 70B. However, we used OpenAI's structured output API, defining the target format as a JSON schema with column names as keys and corresponding cell values as entries.

Listing 1: ICL Prompt Template passed to Groq API

```
System role: You are a tabular synthetic data generation
     model.

Your goal is to produce data that mirrors the given
     examples in
causal structure and feature/label distributions,
while maximizing diversity.

Context: Leverage your in-context learning to generate
     realistic,
diverse samples.

Output format: JSON.

Dataset name: {dataset_name}

Column names (in order): {col_names}

Summary statistics:
{summary_stats}

CSV of full data:
{data}

Please generate {batch_size} rows of synthetic data.

Treat the rightmost column as the target. Return only a
     JSON object:
{
  "synthetic_data": "<CSV string>"
}

Do not include any additional text.
```

A.2 Black-Box MIAs

We apply feature space based black box MIAs in the same experimental design as LevAtt. Here, we treat the exemplar observations as the membership class and holdout data as non-membership. Before calling the attacks, we scale continuous and one-hot encode categorical variables we instead ordinal encode them. All implementations of these attacks are through the Synth-MIA library [53].

(1) **Distance to Closest Record (DCR)** [12]: DCR is a black-box attack that scores test data based on a score of the Euclidean distance to the nearest neighbor in the synthetic dataset.
(2) **MC** [22]: MC is based on counting the number of observations in the synthetic dataset that fall into the neighborhood of a test point (Monte Carlo Integration). However, this method does not consider a reference dataset, and the choice of distance metric for defining a neighborhood is a non-trivial hyperparameter to tune.
(3) **Density Estimate** [22, 50]: Density Estimate follows a similar strategy as MC, but rather than using a Monte Carlo approximation of density, instead uses a Kernel Density Estimator (KDE). The idea is that a test observation that is a training observation will have a higher density estimate on a KDE fit over a synthetic dataset.

A.3 SFT-Datasets

Dataset	# Instances	# Features
Abalone (OpenML)	4,177	9
CA Housing (OpenML)	20,640	9
CASP (OpenML)	45,730	9
Diabetes (Sklearn)	412	10
Steel Plates Faults (UCI)	1,941	27

[2]LLaMA 3.3-70B failed on *geographical-origin-of-music*, *pumadyn32nh*, *student-performance-por*, *superconductivity*, and *wave-energy* due to token limitations. TabPFN failed on *geographical-origin-of-music* due to extreme dimensionality.

A.4 SFT Model Details

We modify the original RealTabFormer implementation to use LLaMA 3.2 (1B, 3B) [21], Qwen2.5-3B [41], and Mistral v0.3 7B [27]. Here, we follow RealTabFormer's base training and sampling hyperparameters in Table 4. To SFT GREAT, we also use its original implementation of which the base hyperparameters can be found in Table 5. For CT-GAN and TVAE, we use the default hyperparameters and implementation found in Synthcity [40].

Hyperparameter	Default Value
epochs	1000
batch_size	8
train_size	1
output_max_length	512
early_stopping_patience	5
early_stopping_threshold	0
mask_rate	0
numeric_nparts	1
numeric_precision	4
numeric_max_len	10

Table 4: Numeric hyperparameters of REaLTabFormer.

Hyperparameter	Default Value
epochs	100
batch_size	8
float_precision	None
temperature	0.7
top-k sampling	100
max_length	100

Table 5: GReaT training and sampling hyperparameters.

A.5 Simulated Data Generation Details

For Figure 5, we initialize a Multivariate Gaussian of N(1e10,1e9). This ensures that there are up to 10 digits for a column as RealTabFormer can struggle to process exceptionally long decimal strings. We then sample 10,000 training and holdout rows for our experiment. At each level, we add additional columns to the training, holdout, and therefore synthetic dataset observation sequence lengths increase by 10 digits for each column.

B More Details on Defenses

Digit Substitution Rule in DM: For the Digit Modifier (DM), each selected digit is perturbed using a simple increment rule. Specifically, a digit $d \in \{0, 1, \ldots, 9\}$ is replaced by $(d + 1) \bmod 10$. Thus, 0 becomes 1, 1 becomes 2, and so on, with 9 wrapping around to 0. This cyclic increment operation provides a minimal yet consistent perturbation to each affected digit, ensuring that the modified values remain close to their originals while still introducing controlled randomness.

Handling Invalid Logits in ReaLTabFormer: In ReaLTabFormer, some logits take the value $-\infty$ or become NaN due to masking constraints in the tabular structure. These values arise when certain tokens are structurally disallowed, leading to zero-probability entries after masking. Since such logits cannot be meaningfully transformed by TLP, we simply ignore them: TLP is applied only to finite-valued logits, while any $-\infty$ or NaN logits are left unchanged to preserve the validity of the generator's masking logic.

Why TLP Is Applied Only at Inference Time: The Tendency-based Logit Processor (TLP) is designed exclusively for inference-time perturbations. Applying TLP during training severely disrupts the optimization dynamics: the modified logits distort the gradient signal, causing the loss to decrease extremely slowly and preventing the model from converging in a reasonable number of steps. In effect, the model must simultaneously learn both the underlying data distribution and the perturbation induced by TLP, which dramatically complicates training. By restricting TLP to inference time, we preserve stable training behavior while still introducing controlled variability into the generated samples.

Role of the Stabilization Constant ε. The shifted min–max scaling used in TLP requires a small positive constant ε to ensure numerical stability. In cases where all logits in a column share the same value, we have $M_l = m_l$, causing the denominator $M_l - m_l$ in the scaling expression to become zero. Without a stabilizing term, this would result in undefined values or NaN outputs after normalization. By adding a small $\varepsilon > 0$, we guarantee that the denominator remains strictly positive, preventing division-by-zero errors and ensuring that the scaled logits remain well defined. This safeguard is essential for applying TLP robustly, especially when the generator outputs nearly constant logits due to structural constraints in the data.

C Additional Tables and Figures

Table 6: ICL Experiment Dataset Characteristics

Dataset Name	Num. Numeric	Num. Categorical	Instances
Moneyball	9	6	1232
QSAR_fish_toxicity	7	0	908
abalone	8	1	4177
airfoil_self_noise	6	0	1503
auction_verification	6	2	2043
brazilian_houses	6	4	10692
california_housing	9	0	20640
cars	18	0	804
concrete_compressive_strength	9	0	1030
cps88wages	3	4	28155
cpu_activity	22	0	8192
diamonds	7	3	53940
energy_efficiency	9	0	768
fifa	28	1	19178
forest_fires	11	0	517
fps_benchmark	31	13	24624
geographical_origin_of_music	117	0	1059
grid_stability	13	0	10000
health_insurance	5	7	22272
kin8nm	9	0	8192
kings_county	18	4	21613
miami_housing	16	0	13932
naval_propulsion_plant	15	0	11934
physiochemical_protein	10	0	45730
pumadyn32nh	33	0	8192
red_wine	12	0	1599
sarcos	22	0	48933
socmob	2	4	1156
solar_flare	3	8	1066
space_ga	7	0	3107
student_performance_por	14	17	649
superconductivity	82	0	21263
video_transcoding	17	2	68784
wave_energy	49	0	72000
white_wine	12	0	4898

Table 7: SFT Experiment Dataset Characteristics

Dataset Name	Num. Numeric	Num. Numeric	Instances
Housing	1	9	20,640
Diabetes	2	7	768
CASP	0	10	45,730
Faults	10	24	1,941
Abalone	2	7	4,177

Table 8: Privacy and fidelity metrics across ICL generators and context size (Mean ± STD). Runs with fewer context example sizes demonstrate both higher risk to LevAtt and lower utilities.

Generator	Size	AUC	TPR@0	TPR@0.1	MMD	Wasserstein	ML AUC
LLaMA-3.3-70B	32	0.67 ± 0.146	0.11 ± 0.246	0.34 ± 0.265	0.06 ± 0.016	0.65 ± 0.581	0.61 ± 0.113
	64	0.62 ± 0.134	0.07 ± 0.184	0.26 ± 0.237	0.03 ± 0.025	0.67 ± 0.831	0.62 ± 0.115
	128	0.60 ± 0.112	0.07 ± 0.172	0.24 ± 0.208	0.02 ± 0.005	4090.75 ± 38150.682	0.66 ± 0.123
TabPFN-V2	32	0.59 ± 0.112	0.04 ± 0.176	0.20 ± 0.200	0.06 ± 0.006	1.12 ± 1.155	0.62 ± 0.103
	64	0.57 ± 0.111	0.04 ± 0.173	0.18 ± 0.207	0.03 ± 0.004	0.95 ± 1.100	0.68 ± 0.117
	128	0.56 ± 0.113	0.04 ± 0.172	0.17 ± 0.205	0.02 ± 0.003	0.81 ± 1.052	0.70 ± 0.115
GPT-4o-mini	32	0.55 ± 0.055	0.00 ± 0.009	0.15 ± 0.063	0.07 ± 0.019	14.45 ± 67.815	0.57 ± 0.087
	64	0.53 ± 0.045	0.00 ± 0.010	0.12 ± 0.040	0.04 ± 0.010	13486.80 ± 126698.02	0.59 ± 0.102
	128	0.52 ± 0.026	0.00 ± 0.003	0.11 ± 0.029	0.03 ± 0.106	72.58 ± 407.985	0.62 ± 0.099

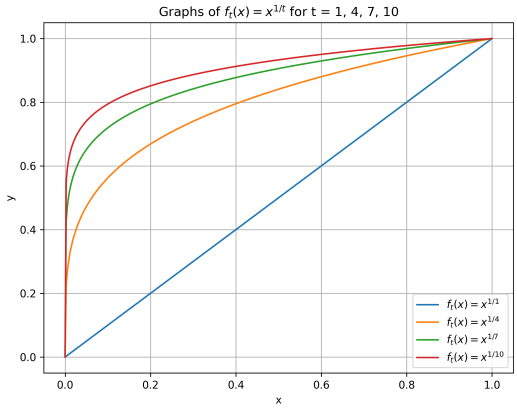
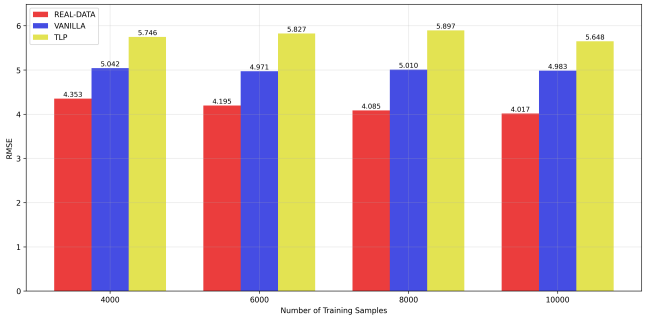
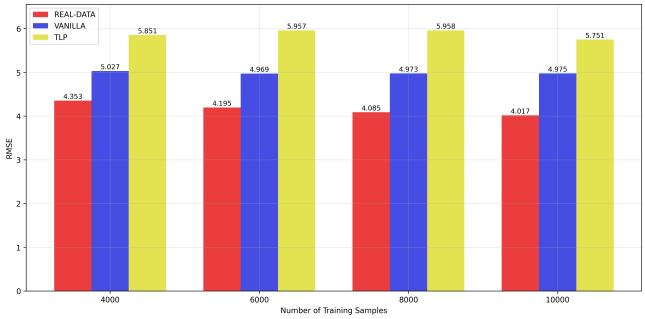


Figure 10: Visualization of the transformation function $f_t(x) = x^{1/t}$ under varying values of t. As t increases, the function becomes more concave, amplifying smaller logits proportionally more than larger ones. This behavior helps compress logit differences while preserving their ordering.



(a) RMSE of CASP dataset XGBoost models where TLP-protected datasets have a LevAtt AUC of below 0.55.



(b) RMSE of CASP dataset XGBoost models where TLP-protected datasets have a LevAtt TPR@FPR=0.1 of below 0.125.

Figure 11: Utility comparison of XGBoost models trained on real, vanilla synthetic, and TLP-protected synthetic data at various synthetic dataset (and thus privacy leakage) sizes. Real data achieves the lowest RMSE, vanilla synthetic shows moderate degradation, and TLP-protected data shows larger degradation. However, the gap between vanilla and TLP remains stable as training size increases, indicating that TLP provides a controllable privacy–utility trade-off even when stronger tendency levels are needed.