Prism: Private Relational Data Synthesis with Language Models

GUOHUI GUAN, University of Minnesota, USA
CHANG GE, University of Minnesota, USA

Organizations increasingly rely on relational data containing sensitive personal information for downstream tasks. A common approach is to release synthetic data as a privacy-preserving substitute, typically by learning a generative model and sampling from it. Differential privacy (DP) provides a formal guarantee of data privacy, and existing data synthesis systems aim to balance its inherent trade-off between data privacy and task utility. The advent of pretrained large language models (LLMs) has reshaped this trade-off landscape: on the utility side, LLMs possess stronger representational capabilities and encode extensive prior knowledge that does not consume the privacy budget, leading to improved task utility; on the other hand, enforcing differential privacy for LLM-based relational data synthesis introduces new technical challenges.

We present Prism, an end-to-end framework for DP relational data synthesis using LLMs. Prism privatizes knowledge transfer from an ensemble of teacher LLMs to a student generator via DP output aggregation. We introduce three key innovations in the aggregation mechanism to address LLM-specific challenges, including 1) token pruning with tries to minimize teacher queries and hence the privacy cost; 2) a data-dependent DP mechanism that adaptively scales noise for efficient budget use; and 3) probabilistic sampling with threshold filtering to reduce bias and preserve diversity. Extensive empirical results across twelve experiments show that Prism achieves substantially higher predictive utility than eight state-of-the-art methods under the same privacy budget.

CCS Concepts: • **Security and privacy → Privacy-preserving protocols**.

Additional Key Words and Phrases: Differential Privacy, Data Synthesis, Large Language Models

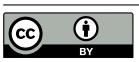
ACM Reference Format:
Guohui Guan and Chang Ge. 2026. Prism: Private Relational Data Synthesis with Language Models. *Proc. ACM Manag. Data* 4, 3 (SIGMOD), Article 225 (June 2026), 28 pages. https://doi.org/10.1145/3802102

1 Introduction

Organizations increasingly rely on personal data to support a wide range of data analytics and machine learning applications [20, 34]. Such data are often stored in relational tables due to their structured nature [60], and contain sensitive personal information. Uncontrolled sharing of these data can lead to serious privacy risks [51, 78] and violations of data protection regulations [1, 93].

To enable responsible use of sensitive data, differential privacy (DP) [22] has emerged as a rigorous definition and has been widely adopted by government agencies [3, 41, 58, 65, 92] and industry [6, 45, 69, 70, 87]. Informally, a DP mechanism guarantees that the inclusion or exclusion of any individual's record has only a negligible effect on the output, making it nearly impossible to infer the participation of any specific individual. In practice, DP is typically enforced by adding calibrated randomness governed by a fixed privacy budget, concealing individual contributions but inevitably introducing a trade-off between data privacy and task utility.

Authors' Contact Information: Guohui Guan, University of Minnesota, Minneapolis, MN, USA, gguan@umn.edu; Chang Ge, University of Minnesota, Minneapolis, MN, USA, cge@umn.edu.



ACM 2836-6573/2026/6-ART225
https://doi.org/10.1145/3802102

Broadly, there are two common paradigms for applying DP. The first is task-specific, where DP mechanisms are tailored to a particular analytical goal. Examples include statistical reporting [11, 83], SQL query answering [31, 42, 48], and graph analysis [95, 98]. This paradigm enables fine-grained control over the privacy–utility trade–off for the target workload. However, each task consumes part of the overall privacy budget and requires its own DP mechanism. Conducting additional tasks on the same dataset therefore incurs further privacy costs, making this approach difficult to scale when multiple tasks or users need to access the data [44].

The second paradigm is data-generative, where a generative model is privately learned from the sensitive data, with randomness introduced during the learning process to ensure DP. A synthetic dataset is then sampled from the trained model, aiming to preserve the utility of the original data for downstream tasks. Owing to the post-processing property of DP [23], the released synthetic data can be freely used across multiple applications without incurring additional privacy cost. This paradigm removes the need to design separate DP mechanisms for each task, and the synthetic data, which share the same schema and cardinality as the original, can serve as a drop-in replacement reusable for different tasks.

While releasing private synthetic data has been attractive [4, 8, 29, 43, 52, 54, 79, 82, 91, 96, 99–101] and has been an important topic in the database community [4, 12, 55, 82, 99], most if not all methods spend the privacy budget to learn a generative model from scratch. Given the limited privacy budget and the representational constraints of commonly used models such as probabilistic graphical models [54, 99], GANs [29, 96], autoencoders [4, 82], and diffusion models [52, 91], these approaches often struggle to capture the complex correlations present in real-world datasets [38].

Recently, pretrained large language models (LLMs) have emerged as a promising alternative that fundamentally reshapes the privacy-utility trade-off space. Unlike earlier approaches that train generative models entirely from scratch, LLMs bring rich prior knowledge learned from large-scale corpora and possess strong representational capacity to capture complex data correlations [30]. These properties suggest that LLMs can deliver high utility while consuming less of the privacy budget by leveraging pretrained knowledge without incurring additional privacy cost, making them appealing backbone models for relational data synthesis. However, on the privacy side, applying standard DP model training such as DP-SGD [2] to LLMs remains challenging [76, 77, 89]. Injecting noise into model weights during gradient descent can overwhelm meaningful signals, distorting the pretrained knowledge embedded in the LLM and thereby degrading its downstream utility [97].

Therefore, to harness the strengths of LLMs while avoiding the limitations of model perturbation, we develop an alternative strategy based on output perturbation for enforcing differential privacy. Specifically, our approach builds upon the framework of private aggregation of teacher ensembles (PATE) [62]. In this framework, an ensemble of teacher LLMs is fine-tuned on disjoint partitions of the sensitive data, and their DP aggregation serves as the mechanism for transferring knowledge to the student. For each query, the teachers produce votes, which are then privatized before being used to train a student LLM. This approach eliminates the need to inject noise directly into model parameters and instead provides privatized supervision signals for training the student generator.

While PATE avoids direct perturbation of model weights, applying PATE-style guarantees to LLM data synthesis introduces new challenges. PATE was designed for single-step classification over a small label set. Autoregressive LLM data synthesis instead requires a sequence of token-level queries over a large vocabulary, fundamentally changing the problem in three ways.

First (**C1**), aggregation in autoregressive synthesis must simultaneously ensure *schema-domain validity* and *query efficiency* under a large, context-dependent token space. At each generation step, the candidate token set is large and context-dependent; under DP noise, the aggregated vote can favor tokens that are linguistically plausible but invalid for the attribute domain, which breaks schema validity. Meanwhile, generating a single tuple requires many sequential aggregation queries

across tokens and attributes, so a naïve token-by-token querying strategy can rapidly consume the privacy budget. Second (**C2**), *teacher consensus varies* substantially across generation steps. Unlike classification with a fixed small label set, autoregressive generation predicts over a large, context-dependent token space, so vote histograms can be highly concentrated or highly dispersed. A uniform DP noise scale is therefore suboptimal: it either wastes privacy budget under strong consensus or overwhelms weak signals and hurts utility. Third (**C3**), *knowledge transfer must preserve diversity under noise.* Naïve noisy-max aggregation used in PATE induces mode collapse toward frequent values, which is particularly harmful for data synthesis.

Motivated by the above challenges, we present PRISM, an end-to-end framework for differentially private relational data synthesis using large language models. PRISM addresses these challenges with a tuple distillation mechanism that incorporates three key innovations, each of which has deep roots in prior database research: trie-guided pruning to short-circuit queries while enforcing schema validity (addressing C1), data-dependent DP to adapt noise for efficient budget use (addressing C2), and probabilistic sampling with threshold filtering to mitigate noise-induced bias and spurious token selection (addressing C3). These components are tightly coupled and collectively enable feasible and high-utility DP synthesis.

In summary, *our novelty* is an end-to-end framework that instantiates PATE-style private aggregation for autoregressive LLM generation to enable DP synthesis of structured relational data, where knowledge transfer proceeds via sequential token-level aggregations over a large, dynamic vocabulary while still producing schema-valid tuples. We summarize our contributions as follows:

- To the best of our knowledge, PRISM is the first to adapt output perturbation for differentially private relational data synthesis with LLMs, achieved through private tuple distillation (§ 3).
- We develop an end-to-end tuple distillation mechanism with three key innovations (i.e., trie-guided token querying, data-dependent DP noise scaling, and thresholded probabilistic sampling) to address the fundamental challenges that arise when applying PATE-style guarantees beyond single-step classification to autoregressive LLM-based data synthesis (§ 5).
- We release an open-source prototype[1] of PRISM. Through extensive evaluation consisting of twelve experiments, we demonstrate that PRISM achieves substantially higher predictive utility than eight state-of-the-art methods (§ 7).

2 Preliminaries

Differential Privacy (DP). A randomized algorithm $\mathcal{M}$ achieves (ϵ, δ)-DP [22, 25] if for all output $\mathcal{S} \subseteq \text{Range}(\mathcal{M})$ and for any two database instances $D, D' \in \mathcal{D}$ that differ only in one tuple, $\Pr[\mathcal{M}(D) \in \mathcal{S}] \leq e^{\epsilon} \Pr[\mathcal{M}(D') \in \mathcal{S}] + \delta$. The parameters (ϵ, δ) quantify the privacy cost and are commonly referred to as the privacy budget. A smaller privacy budget corresponds to a stronger privacy guarantee.

Complex DP algorithms can be built from the basic algorithms following two important properties of differential privacy: 1) Post-processing [23] states that for any function g defined over the output of the mechanism $\mathcal{M}$, if $\mathcal{M}$ satisfies (ϵ, δ)-DP, so does $g(\mathcal{M})$; and 2) Composability [22] states that if $\mathcal{M}_1, \mathcal{M}_2, \cdots, \mathcal{M}_n$ satisfy (ϵ_1, δ_1)-, (ϵ_2, δ_2)-, $\cdots$, (ϵ_n, δ_n)-DP, then a mechanism sequentially applying $\mathcal{M}_1$, $\mathcal{M}_2$ to $\mathcal{M}_n$ satisfies $(\sum_{i=1}^{n} \epsilon_i, \sum_{i=1}^{n} \delta_i)$-DP.

Private Aggregation of Teacher Ensembles (PATE). The PATE framework [62, 63] is designed to train a differentially private classifier (i.e., the student model) via noisily aggregating predictions from a set of classifiers (i.e., the teacher models). In PATE, the sensitive dataset D is partitioned

[1] https://github.com/golden-eggs-lab/prism

into k disjoint sets $\{D_{1,\cdots,k}\}$ (i.e., $D = \cup_{i=1}^{k} D_i$ and $D_i \cap D_j = \emptyset, \forall i \neq j, i, j \in [1, k]$). PATE consists of three components:

- Teacher models: Each teacher is a model trained independently only on one partition $D_{i\in[1,k]}$. At inference, teachers independently predict class labels.
- Aggregation mechanism: To classify an input $\mathbf{x}$, each teacher model votes for the class. The votes are counted, Gaussian noise $\mathcal{N}(0, \sigma^2)$ is added to each count, and the class with the highest noisy count is returned, i.e., $\arg\max_{j\in[1,t]}(n_j(\mathbf{x}) + \mathcal{N}(0, \sigma^2))$, where $n_j(\mathbf{x})$ is the number of teachers predicting class j. This implements a noisy max-vote mechanism for DP [22].
- Student model: The student learns by querying the teacher ensemble on unlabeled data. Since each query consumes part of the privacy budget, the number of queries is limited. The student receives only noisy labels, ensuring DP through DP's post-processing property [23].

3 Problem and Solution Overview

3.1 Problem Statement

Given a private relational table D in schema R and a DP budget (ϵ, δ), we aim to design a process $\mathcal{P}$ that generates a synthetic dataset $\tilde{D}$ in same schema R and cardinality $|D|$, such that:

- (Predictive Utility) The synthetic $\tilde{D}$ preserves predictive characteristics of the original D, i.e., for a downstream classification task $T(D_{train}, D_{test})$, trained on D_{train} and tested on D_{test}, and an evaluation metric E, $E(T(\tilde{D}, \bar{D})) \approx E(T(D \setminus \bar{D}, \bar{D}))$, where testing data $\bar{D} \subset D$.
- (Privacy Guarantee) The process $\mathcal{P}$ that generates $\tilde{D}$ satisfies (ϵ, δ)-differential privacy, i.e., for any set of instances $\mathbb{D}$ output by $\mathcal{P}$, $\Pr(\mathcal{P}(D_1) \in \mathbb{D}) \leq e^{\epsilon} \Pr(\mathcal{P}(D_2) \in \mathbb{D}) + \delta$, for any two neighboring D_1 and D_2 that differ in one tuple.

3.2 Solution Overview

We propose Prism, an end-to-end framework for differentially private table synthesis using large language models. Prism adopts a teacher–student paradigm to privately transfer knowledge from an ensemble of teacher models to a student model through tuple distillation. The framework is agnostic to the choice of pretrained LLMs; for notational simplicity, we denote all instances using a single LLM type M_0.

Given a private relational table D in schema R, a DP budget (ϵ, δ), an ensemble size k, and pretrained LLM M_0, Prism performs private data synthesis in three steps, which are illustrated by Figure 1 and more formally described in Algorithm 1.

Step 1: Fine-tune Teacher LLMs (§ 4). Prism first partitions the original table D into k disjoint subsets of equal size. Each subset D_i is used to fine-tune a teacher model M_i initialized from the input model M_0, where each tuple is represented as a tuple sentence in the format `[attribute is value]`. This step enables each teacher to learn independently from its partition of the private data (Algorithm 1, Lines 2–5).

Step 2: Transfer Knowledge From Teachers (§ 5). After the teacher ensemble is trained, Prism privately transfers their knowledge through a tuple distillation process. To distill a tuple sentence, Prism performs autoregressive generation, where the ensemble predicts one token at a time until the entire tuple is completed. To reduce privacy cost, Prism constructs a token trie for every attribute domain (§ 5.2). These tries guide generation by pruning out-of-domain tokens and short-circuiting deterministic completions, thereby reducing the number of teacher queries. The teachers' token-level votes are then aggregated into histograms, to which Prism applies a data-dependent differential privacy mechanism that adaptively scales Gaussian noise based on the histogram range

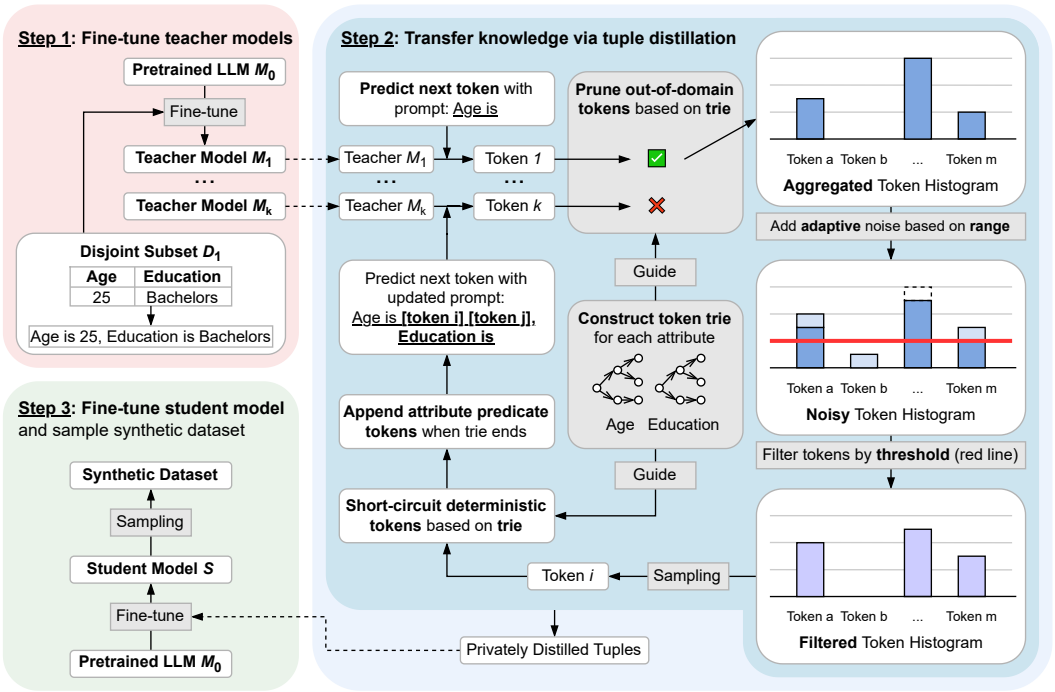


Fig. 1. An overview of Prism 's end-to-end workflow. The process comprises three main stages: 1) fine-tuning teacher LLMs on disjoint subsets of the sensitive dataset (§ 4); 2) transferring teachers' knowledge through private tuple distillation (§ 5); and 3) fine-tuning a student LLM on the distilled data to generate the final synthetic dataset (§ 4).

(§ 5.3). Next (§ 5.4), threshold-based filtering removes low-support tokens introduced by noise, and probabilistic sampling selects the next token from the remaining candidates. This process iteratively forms a set of privately distilled tuples, which together constitute the distilled dataset $\hat{D}$ satisfying (ϵ, δ)-DP. This step corresponds to Line 6 of Algorithm 1.

Step 3: Fine-tune Student LLM and Synthesize Data (§ 4). The distilled dataset $\hat{D}$, being synthetic and already differentially private, is then used to fine-tune a student model S initialized from M_0. Since this training relies entirely on $\hat{D}$, it incurs no additional privacy cost. Once trained, the student model S serves as the generative synthesizer that samples a complete synthetic dataset $\tilde{D}$ matching the schema and cardinality of the original table. The resulting $\tilde{D}$ can be safely released for downstream tasks. This step corresponds to Lines 7–8 of Algorithm 1. □

In the rest, we elaborate on each step of the Prism workflow. The fine-tuning of teacher and student LLMs, as well as the synthetic data generation are described in § 4. The subsequent § 5 describes the knowledge transfer via data distillation, which focuses on three key innovations of Prism: token pruning with tries (§5.2), a data-dependent DP mechanism (§5.3), and probabilistic sampling with threshold filtering (§5.4).

4 Model Fine-Tuning and Data Synthesis

This section describes the shared fine-tuning procedure for both the teacher and student LLMs, as well as how the final synthetic dataset is sampled from the student LLM.

Algorithm 1 PRISM: End-to-end workflow

Require: Private relational table D in schema R, DP budget (ϵ, δ), ensemble size k, pretrained LLM M_0

1: **procedure** PRISM($D, R, (\epsilon, \delta), k, M_0$)
2: Randomly partition D into k disjoint subsets $\{D_{1,\cdots,k}\}$
3: **for** $i \in [1, k]$ **do**
4: $M_i \leftarrow$ TRAIN(D_i, R, M_0) ▷ Algorithm 2
5: **end for**
6: $\hat{D} \leftarrow$ DISTILL$\{M_{1,\cdots,k}\}, R, (\epsilon, \delta))$ ▷ Algorithm 5
7: $S \leftarrow$ TRAIN($\hat{D}, R, M_0$) ▷ Algorithm 2
8: $\tilde{D} \leftarrow$ SYNTHESIZE($S, R, |D|$) ▷ Algorithm 3
9: **return** $\tilde{D}$
10: **end procedure**

Model Fine-Tuning. As the first work to apply PATE to LLM-based DP data synthesis, we adopt a consistent LLM architecture for both teachers and the student to keep the workflow aligned and allow the same fine-tuning and generation procedures for both sides, enabling us to focus on the core technical challenges. Empirically, this design is already competitive with prior DP data synthesizers, as shown in § 7. In PRISM, both teacher and student models are initialized from pretrained LLMs. Although teacher and student models are fine-tuned on different datasets, with the teacher trained on a partition of the true dataset and the student on the distilled dataset, both datasets share the same schema. As a result, the fine-tuning procedure remains consistent.

Generative LLMs are designed to model and generate sequences of natural language. To apply them to a relational table, we first convert each tuple into a textual representation (i.e., tuple sentence). Various encoding strategies exist for this transformation [8, 30, 100], and a comparative analysis of encodings is beyond the scope of this work. In PRISM, we choose to leverage the schema context and represent each tuple as a sequence of attribute-value statements in the format `[attribute predicate value]` [21, 36]. Formally, given a tuple $t = [v_1, \cdots, v_k]$ under schema $R = [A_1, \cdots, A_k]$, we construct a tuple sentence as "A_1 `is` $v_1, \cdots, A_k$ `is` v_k".

After converting to tuple sentences, we apply fine-tuning using the causal language modeling (CLM) objective [67], which is implemented as token-level cross-entropy over each tuple sentence. Specifically, for each training sequence $x = (x_1, \ldots, x_n)$, the model is trained to minimize: $\mathcal{L}_{\text{CLM}}(x) = -\sum_{i=1}^{n} \log P_w(x_i \mid x_{<i})$, where P_w is the predicted distribution over tokens parameterized by w.

Example 1. A tuple with schema $R = [age, education]$ and values (30, *Bachelors*) is textualized as "`age is 30, education is Bachelors`." The resulting corpus of sentences is then used to fine-tune a pretrained language model.

Algorithm Details. Algorithm 2 shows the process to fine-tune a LLM. Each tuple $t \in D$ is first converted into a tuple sequence $\hat{t}$ in natural language through the schema-driven textualization procedure using TEXTUALIZE (Line 3). The resulting corpus $\{\hat{t}_{1,\cdots,n}\}$ is then used to fine-tune a language model M_0 using a CLM objective.

Data Synthesis from Student Model. Once the student model S has been fine-tuned, it serves as the generative synthesizer for producing the final synthetic dataset (Algorithm 3). PRISM employs a sentence-level sampling procedure to generate complete tuples that conform to the input schema

Algorithm 2 Train: LLM fine-tuning using relational table

Require: Dataset $D = \{t_{1,\cdots,n}\}$, schema R, pretrained LLM M_0

1: **procedure** Train(D, R, M_0)
2: **for** each tuple $t \in D$ **do**
3: $\hat{t} \leftarrow$ Textualize(t, R) ▷ Convert to a tuple sentence
4: **end for**
5: Fine-tune M_0 on corpus $\{\hat{t}_{1,\cdots,n}\}$
6: **return** M_0
7: **end procedure**

Algorithm 3 Synthesize: Sentence sampling from student LLM

Require: Student model S, schema R, cardinality m

1: **procedure** Synthesize(S, R, m)
2: $\tilde{D} \leftarrow \emptyset$
3: **while** $|\tilde{D}| < m$ **do**
4: $prompt \leftarrow$ "A_1 `is`"; $sentence \leftarrow S$.Generate($prompt$)
5: $t \leftarrow$ ParseSentence($sentence, R$)
6: **if** t contains every attribute in R **then**
7: $\tilde{D} \leftarrow \tilde{D} \cup \{t\}$
8: **end if**
9: **end while**
10: **return** $\tilde{D}$
11: **end procedure**

R. Each generation begins with a schema-aware prompt "A_1 `is`", from which the model autoregressively generates a textual representation of a tuple (Line 4). The generated sentence is then parsed back into a structured tuple following the reverse of the textualization process (Line 5). This process repeats until the number of synthesized tuples reaches the desired cardinality m, resulting in the final synthetic table $\tilde{D}$.

Example 2. Assume the same schema $R = [age, education]$. Generation starts from a prompt "`age is`". The student model autoregressively generates a sentence such as "`age is 30, education is Bachelors`". This sentence is then parsed back into a structured tuple [30, *Bachelors*]. Repeating this process yields the final synthetic dataset $\tilde{D}$.

5 Knowledge Transfer via Tuple Distillation

The original PATE framework was designed for training a student classifier, in which knowledge is transferred from the teacher ensemble through noisy aggregation over class labels (§ 2). However, in the context of data synthesis, the student is a generative model rather than a classifier, necessitating a different knowledge transfer mechanism. Prism develops a novel knowledge transfer approach called tuple distillation. That is, the teacher ensemble produces a small set of synthetic tuple sentences that approximate the distribution of the true dataset, while ensuring differential privacy.

5.1 Overview of Tuple Distillation

Tuple distillation follows a schema-guided autoregressive generation process at the token level. Generation proceeds at three levels: individual attributes, complete tuples, and the distilled dataset.

Individual Attribute. For a given attribute, tuple distillation generates its value token by token. At each step, the partially generated tuple serves as a prompt. All fine-tuned teacher models independently predict the next token, and their predictions are aggregated into a histogram. After adding DP noise, one token is selected and appended. The process repeats until the attribute value is completed. It is necessary for tuple distillation to perform voting at the token level because LLMs are pretrained to generate tokens, and their knowledge is encoded at the token level. Treating entire categorical values (e.g., *HighSchool*) as atomic outputs would require redefining them as new tokens, for which pretrained models have no contextual knowledge, making it impossible to leverage their knowledge.

Example 3.
Consider a schema $R = [age, education]$. Assume the domain of *age* is {30, 40} and the domain of *education* is {*HighSchool*, *Bachelors*}. Each value is represented as a sequence of tokens by the tokenizer. For example, the value *30* is tokenized as [“3”, “0”], *40* as [“4”, “0”], *HighSchool* as [“high”, “school”], and *Bachelors* as [“ba”, “che”, “lors”].
Tuple distillation starts with the prompt “age is”. Each teacher predicts a next token such as “3” or “4”, and their predictions are aggregated into a histogram. After adding noise, the token “3” is selected and appended, yielding “age is 3”. The same procedure then selects “0”, completing the value *30*.

Complete Tuple. Tuple distillation proceeds attribute by attribute. This process repeats until all attributes in the schema have been generated, producing a complete tuple.

Example 4.
Continuing the previous example, once *age* is completed, the prompt is extended to “age is 30, education is”. The same procedure is applied to the *education* attribute. A valid value *Bachelors* is produced. At this point, all attributes have been generated, yielding the tuple $[30, Bachelors]$.

Distilled Dataset. Each tuple generated in this manner is added to the distilled dataset $\hat{D}$. The tuple distillation procedure repeats, generating new tuples while tracking the cumulative privacy cost. Generation stops once the total privacy budget is exhausted. The resulting dataset $\hat{D}$ is used to train the student model.

Example 5.
Continuing the previous example, the tuple $[30, Bachelors]$ is added to the distilled dataset $\hat{D}$. The same procedure is repeated to generate additional tuples under the same schema until the privacy budget is fully consumed.

Directly applying this procedure in a naïve manner would rapidly exhaust the privacy budget and introduce noise-induced bias. In the remainder of this section, we describe the technical innovations in Prism that enable efficient and private knowledge transfer via tuple distillation.

5.2 Token Pruning with Tries

During tuple distillation, we query the teacher ensemble for the next token to iteratively form a tuple sentence. Since each token query consumes DP budget, minimizing the number of queries allows Prism to generate more tokens and distill more tuples under a fixed privacy budget. Prism reduces the number of token queries in two ways. First, it queries only the `value` portion of a sentence. Since only the `value` tokens are derived from sensitive data in the `[attribute predicate value]` format, Prism appends the `attribute predicate` tokens during autoregressive generation without querying the ensemble.

Second, Prism further minimizes the number of queries by short-circuiting the deterministic token sequences. Specifically, for each attribute A_i, a token trie [10] $\mathcal{T}_i$ is built from the domain of A_i, which is a set of valid values specified by the schema. When generating a tuple sentence, tries can enable deterministic completions. In such cases, Prism appends the rest of tokens to the tuple sentence, which in this example saves three queries and the privacy cost. As a result, for most categorical values, only a single teacher query is needed to resolve the value, even if it is represented by multiple tokens. Conceptually, this trie-based pruning strategy builds on classical database techniques for prefix-based enumeration and dependency discovery [49, 61, 72].

Example 6. Continuing the previous example. After the prompt is extended to "`age is 30, education is`", if the first token `ba` is selected by teacher voting, the trie uniquely determines the remaining tokens `che` and `lors`, which are appended deterministically without further teacher queries.

Furthermore, since each trie encodes all valid values for an attribute as sequences of tokens, Prism leverages these tries to constrain token predictions from the teacher models, which effectively prunes any out-of-domain token not found at the current trie node from the token histogram. By pruning out-of-domain tokens, the trie enforces semantic validity and narrows the sampling space to only valid tokens, which leads to more compact token histograms for subsequent sampling.

5.3 Data Dependent DP Mechanism

At each generation step, every teacher first votes for its most probable token. After being pruned using the attribute-specific tries to remove out-of-domain tokens (§ 5.2), the votes are aggregated into a count histogram $\{c[\tau]\}$, where $c[\tau]$ denotes the number of teachers voting for token τ. To ensure differential privacy, we add calibrated noise to the aggregated token histogram $\{c[\tau]\}$ to form a noisy one for subsequent filtering and sampling (§ 5.4).

However, naïvely applying the same noise level to all token histograms can lead to inefficient use of the privacy budget [25]. We observe that when teachers' votes are tightly concentrated on a few tokens, a larger DP noise can be added without harming utility while consuming less privacy budget; conversely, when votes are spread across many tokens, a smaller noise needs to be added to preserve voting quality.

Motivated by this observation, Prism develops a data-dependent DP mechanism that adapts the noise scale according to the range of the vote histogram $r = \max_\tau c[\tau] - \min_\tau c[\tau]$. Specifically, for each attribute A_i, Prism keeps track of the minimum and maximum observed range $(r_{\min}, r_{\max})$ of all token histograms belonging to A_i. Given the predefined minimum and maximum noise levels for histogram perturbation $(\sigma^{\mathrm{h}}_{\min}, \sigma^{\mathrm{h}}_{\max})$, for a token histogram with range r, Prism determines the appropriate noise level σ^{h} via linear interpolation, i.e.,

$$\sigma^{\mathrm{h}} = \sigma^{\mathrm{h}}_{\min} + \frac{\sigma^{\mathrm{h}}_{\max} - \sigma^{\mathrm{h}}_{\min}}{r_{\max} - r_{\min}} \times (r - r_{\min}). \tag{1}$$

Algorithm 4 GetAdaptiveNoise: Data-dependent noise scaling

Require: Raw vote histogram $\{c[\tau]\}$, current minimum and maximum observed range $(r_{\min}, r_{\max})$, privacy accountant $\mathcal{A}_{\text{DP}}$
Require: Hyperparameters: minimum and maximum noise scales for histogram $(\sigma^{\text{h}}_{\min}, \sigma^{\text{h}}_{\max})$, noise scale for range σ^{r}
Require: Sensitivity of the range query $\Delta_{\text{r}} = 2$
1: **procedure** GetAdaptiveNoise($\{c[\tau]\}, (r_{\min}, r_{\max}), \mathcal{A}_{\text{DP}}$)
2: $r \leftarrow \max_\tau c[\tau] - \min_\tau c[\tau]$ ▷ True vote range
3: $\tilde{r} \leftarrow r + \mathcal{N}(0, (\sigma^{\text{r}})^2)$; $\mathcal{A}_{DP}$.Step($\sigma^{\text{r}}/\Delta^{\text{r}}$)
4: $\tilde{r} \leftarrow \max(0, \min(\tilde{r}, k))$ ▷ Clip to valid range
5: **if** $r_{\min} = \perp$ **then** $r_{\min} \leftarrow \tilde{r}$ **end if**
6: **if** $r_{\max} = \perp$ **then** $r_{\max} \leftarrow \tilde{r}$ **end if**
7: $r_{\min} \leftarrow \min(r_{\min}, \tilde{r})$; $r_{\max} \leftarrow \max(r_{\max}, \tilde{r})$
8: **if** $r_{\min} = r_{\max}$ **then** $\sigma^{\text{h}} \leftarrow (\sigma^{\text{h}}_{\min} + \sigma^{\text{h}}_{\max})/2$
9: **else** Sample σ^{h} based on Equation (1) **end if**
10: **return** $(\sigma^{\text{h}}, (r_{\min}, r_{\max}))$
11: **end procedure**

Equation (1) maps the current range $r \in [r_{\min}, r_{\max}]$ to a $\sigma^{\text{h}} \in [\sigma^{\text{h}}_{\min}, \sigma^{\text{h}}_{\max}]$. Intuitively, a high-range token histogram is added with larger noise to save privacy budget, whereas a low-range histogram receives smaller noises for preserving utility.

The range r itself is computed from the teachers' vote counts and thus may reveal information about the underlying private data distribution. For example, a very large range may indicate that most teachers strongly agree on a particular token, revealing a strong consensus that could indirectly leak information about patterns in the training data. To mitigate such leakage, Prism treats the range as a differentially private quantity. Specifically, before using it to determine the adaptive noise scale, we perturb the raw range r with Gaussian noise $\mathcal{N}(0, (\sigma^{\text{r}})^2)$, where σ^{r} is a hyperparameter controlling the privacy level of this auxiliary query.

Using noisy range $\tilde{r}$ in place of the raw range ensures that even the adaptive noise selection remains differentially private, thereby ensuring that the entire mechanism remains end-to-end differentially private. Each query, including both the noisy range computation and the histogram perturbation, is recorded by a shared privacy accountant, which tracks the cumulative privacy loss and ensures that it remains below the target budget (ϵ, δ). We will show a rigorous proof of end-to-end differential privacy in §6. Conceptually, this adaptive noise bounding strategy follows prior work on data-dependent differential privacy in the database community, where private statistics are used to modulate noise while preserving end-to-end DP guarantees [35, 50, 59].

Example 7. Continuing the running example, suppose that teacher votes for the attribute *age* are highly concentrated on a single token (e.g., "3"), resulting in a large vote range. According to Equation (1), this large range leads to a larger noise scale being assigned to the histogram, which preserves utility while reducing privacy budget consumption.

Algorithm Details. Algorithm 4 describes our proposed data-dependent DP mechanism. Given the raw vote histogram $\{c[\tau]\}$, current minimum and maximum observed range $(r_{\min}, r_{\max})$ and the predefined minimum and maximum noise scales for histogram perturbation $(\sigma^{\text{h}}_{\min}, \sigma^{\text{h}}_{\max})$, the algorithm first computes the true vote range r from the histogram and then perturbs it with Gaussian noise $\mathcal{N}(0, (\sigma^{\text{r}})^2)$ to obtain a differentially private range $\tilde{r}$ (Line 3). The privacy accountant

records this range query (Line 3) with a sensitivity of $\Delta^{\mathrm{r}} = 2$. Finally, the algorithm applies linear interpolation between $(\sigma_{\min}, \sigma_{\max})$ according to $\tilde{r}$ to determine the appropriate noise level σ^{h} (Lines 8–9), which will later be used for adding DP noise to the teachers' aggregated votes.

5.4 Probabilistic Sampling with Threshold Filtering

After the noisy token histogram is formed, Prism prunes the histogram by removing tokens with low consensus. In particular, the added noise for privacy can cause tokens with zero votes to receive non-zero noisy counts. To mitigate this, Prism filters any token whose noisy vote count falls below a threshold θ. Although Gaussian noise is two-sided and can push small counts above the threshold or large counts below it, we can use Lemma 1 to bound the probability of such filtering errors.

Lemma 1 (Filtering Error). *For any token τ with true and noisy vote counts c and $\tilde{c}$ respectively, given the filtering threshold θ, the probability of filtering error is:*

$$\Pr(\textit{error}) = \begin{cases} 1 - \Phi\left(\frac{\theta - c}{\sigma}\right), & c < \theta,\ \tilde{c} > \theta, \\ \Phi\left(\frac{\theta - c}{\sigma}\right), & c > \theta,\ \tilde{c} < \theta. \end{cases} \tag{2}$$

where $\Phi(\cdot)$ is the CDF of the standard normal, and σ is the standard deviation of the Gaussian distribution used to generate the noisy counts (i.e., $\tilde{c} = c + \mathcal{N}(0, \sigma^2)$). A false positive occurs when $c < \theta$ but $\tilde{c} > \theta$, i.e., τ should be filtered but is not. A false negative occurs when $c > \theta$ but $\tilde{c} < \theta$, i.e., τ is filtered but should not be.

Proof. For any token τ, let c and $\tilde{c}$ represent its true and noisy vote counts, respectively. Recall that $\tilde{c} = c + \mathcal{N}(0, \sigma^2)$, a random variable $X = \frac{\tilde{c} - c}{\sigma}$ has a distribution of standard Gaussian, i.e., $X \sim \mathcal{N}(0, 1)$. Let $\Phi(\cdot)$ be the CDF of X. Given the threshold θ,

Case 1: False positive. If $c < \theta$ and $\tilde{c} > \theta$,

$$\Pr(\tilde{c} > \theta) = \Pr(\frac{\tilde{c} - c}{\sigma} > \frac{\theta - c}{\sigma}) = \Pr(X > \frac{\theta - c}{\sigma}) = 1 - \Phi(\frac{\theta - c}{\sigma})$$

Case 2: False negative. If $c > \theta$ and $\tilde{c} < \theta$,

$$\Pr(\tilde{c} < \theta) = \Pr(\frac{\tilde{c} - c}{\sigma} < \frac{\theta - c}{\sigma}) = \Pr(X < \frac{\theta - c}{\sigma}) = \Phi(\frac{\theta - c}{\sigma})$$

Therefore, Equation (2) holds. □

Empirically, we set the filtering threshold $\theta = 2\sigma^{\mathrm{h}}$. For example, if a token has no vote (i.e., $c = 0$), then the probability of being included in sampling is very small (i.e., $\Pr(\tilde{c} > 2\sigma^{\mathrm{h}} \mid c = 0) = 0.0228$); on the other hand, if it has reasonable vote count (e.g., $c = 4\sigma^{\mathrm{h}}$), then the probability of keeping it for sampling is large (i.e., $\Pr(\tilde{c} > 2\sigma^{\mathrm{h}} \mid c = 4\sigma^{\mathrm{h}}) = 0.9772$). Unlike other works [19, 103], our threshold filtering step applies to the noisy counts as a post-processing step, and hence it does not spend the privacy budget. As we will show in § 7, it effectively ensures that the selected token remains closely consistent with the true data.

Finally, Prism selects the next token through probabilistic sampling, normalizing the token histogram instead of deterministically choosing the token with the highest vote count as in the PATE framework (§ 2). This strategy mitigates bias towards the most frequent values when generating the distilled data. We will empirically demonstrate its effectiveness in § 7. Conceptually, probabilistic sampling and threshold filtering parallel database methods used for sampling-driven approximate query processing [15, 40].

Algorithm 5 DISTILL: Token-wise aggregation from teacher ensemble with DP-adaptive noise

Require: Teacher models $\{M_{1,\cdots,k}\}$, DP budget (ϵ, δ), Schema $R = [(A_1, Dom_{A_1}), \ldots, (A_{|R|}, Dom_{A_{|R|}})]$, where A_i is the i-th attribute and Dom_{A_i} its domain
Require: Sensitivity of the histogram perturbation $\Delta^{\mathrm{h}} = \sqrt{2}$
procedure DISTILL($\{M_{1,\cdots,k}\}, R, (\epsilon, \delta)$)
  Initialize accountant $\mathcal{A}_{\mathrm{DP}}$ with (ϵ, δ); $\hat{D} \leftarrow \emptyset$
  for $A_i \in R$ **do**
    $r_{\min}[A_i], r_{\max}[A_i] \leftarrow \perp$; $\mathcal{T}_{A_i} \leftarrow$BUILDTRIE($Dom_{A_i}$)
  end for
  while $\neg \mathcal{A}_{\mathrm{DP}}$.ISEXHAUSTED() **do**
    $t \leftarrow \langle\rangle$ ▷ Initialize an empty tuple
    for $A_i \in R$ **do**
      Construct *prompt* based on previous columns
      node $\leftarrow \mathcal{T}_{A_i}$.root; *valueTokens* $\leftarrow$ []
      while *node* not leaf **do**
        if $|node.children| = 1$ **then**
          $\tau^* \leftarrow$ sole child; append to *valueTokens*
          node $\leftarrow$ *node.children*$[\tau^*]$
          continue
        end if
        Query each $M_i \in \{M_{1,\cdots,k}\}$ with *prompt*;
        Aggregate teacher votes $c[\tau]$ for each token $\tau \in$ *node.children* to obtain the vote histogram $\{c[\tau]\}$
        $(\sigma^{\mathrm{h}}, (r_{\min}[A_i], r_{\max}[A_i])) \leftarrow$ GETADAPTIVENOISE($\{c[\tau]\}, (r_{\min}[A_i], r_{\max}[A_i]), \mathcal{A}_{\mathrm{DP}}$) ▷ Algorithm 4
        $\tilde{c}[\tau] \leftarrow [c[\tau] + \mathcal{N}(0, (\sigma^{\mathrm{h}})^2)]_+$ for each token τ
        $\mathcal{A}_{\mathrm{DP}}$.STEP($\sigma^{\mathrm{h}}/\Delta^{\mathrm{h}}$)
        Set $\tilde{c}[\tau] \leftarrow 0$ if $\tilde{c}[\tau] < 2\sigma^{\mathrm{h}}$ for each token τ
        Sample $\tau^* \sim$ Categorical$(\tilde{c}[\tau]/\sum_{\tau'} \tilde{c}[\tau'])$
        Append τ^* to *valueTokens*
        node $\leftarrow$ *node.children*$[\tau^*]$
      end while
      $t[A_i] \leftarrow$DETOKENIZE(*valueTokens*)
    end for
    $\hat{D} \leftarrow \hat{D} \cup \{t\}$
  end while
  return $\hat{D}$
end procedure

Example 8. Continuing the running example, assume that at a certain generation step, no teacher predicts the token "`high`". After adding Gaussian noise, this token may obtain a small positive noisy count. If the noisy count is below the threshold $2\sigma^{\mathrm{h}}$, the token is filtered out and excluded from sampling. In contrast, tokens such as "`ba`", which receive sufficient support, are retained and sampled probabilistically.

Algorithm Details. Algorithm 5 describes the complete tuple distillation procedure, which integrates the three key innovations introduced in the previous subsections: token pruning with tries (§ 5.2), data-dependent DP noise (§ 5.3), and probabilistic sampling with threshold filtering (§ 5.4). Together, these components enable PRISM to efficiently transfer knowledge from the teacher ensemble to the distilled dataset $\hat{D}$ under the target privacy budget (ϵ, δ).

The algorithm synthesizes tuples token-by-token while tracking the cumulative privacy cost using a privacy accountant $\mathcal{A}_{\text{DP}}$. For each attribute A_i in the schema R, a trie $\mathcal{T}_{A_i}$ is constructed from its domain Dom_{A_i} to encode all valid token sequences (Line 4), enabling pruning of out-of-domain tokens and deterministic completions. During tuple generation, PRISM iterates over attributes in schema order (Line 8) and constructs a *prompt* representing the partially generated tuple (Line 9).

Within each attribute, the trie guides token-level generation. If the current node has only one child, a deterministic completion is performed without querying the teacher ensemble (Lines 12–16), reducing both query cost for efficiency and privacy cost for better budget usage. Otherwise, PRISM queries all teacher models with the current *prompt* and aggregates teacher votes $c[\tau]$ for each token τ in the current trie node's children to obtain the vote histogram $\{c[\tau]\}$ (Lines 17–18), effectively pruning out-of-domain tokens.

Next, PRISM applies the data-dependent DP mechanism described in § 5.3. Specifically, it determines an adaptive noise scale σ^{h} based on the current vote range and the tracked minimum and maximum observed ranges, and accounts for the privacy cost of this process through Algorithm 4 (invoked in Line 19). Gaussian noise with variance $(\sigma^{\text{h}})^2$ is then added to each token count (Line 20), and this perturbation step is recorded by the accountant with sensitivity Δ^{h} (Line 21). To further avoid sampling spurious tokens caused by added noise, PRISM performs threshold-based filtering, setting noisy counts below $2\sigma^{\text{h}}$ to zero (Line 22).

Finally, the remaining noisy histogram is normalized and used for probabilistic token selection (Line 23). The selected token advances the trie traversal (Line 25), and once the attribute's token sequence is completed, it is detokenized into a textual value (Line 27). After all attributes are generated, the completed tuple t is appended to the distilled dataset $\hat{D}$ (Line 29). The process repeats until the privacy accountant signals that the total budget (ϵ, δ) has been fully consumed (Line 31).

Overall, Algorithm 5 integrates trie-guided token pruning, adaptive noise scaling, and thresholded sampling into a unified DP aggregation mechanism, enabling PRISM to synthesize differentially private tuples efficiently and effectively.

6 Data Privacy Analysis

This section presents a formal privacy analysis of PRISM and establishes its end-to-end (ε, δ)-differential privacy guarantee. We begin by defining the analysis setting and introducing the Rényi DP (RDP) framework [56] used for privacy accounting. We then bound the sensitivity of the released statistics (§ 6.1), derive the per-round RDP cost and its adaptive composition across token-query rounds (§ 6.2), and finally prove the differential privacy of both the tuple distillation mechanism and the overall system (§ 6.3).

Settings. Let D and D' be neighboring datasets that differ in exactly one tuple. The private dataset D is partitioned into k disjoint subsets $D_i \in \{D_{1,\cdots,k}\}$, each used to fine-tune a teacher model M_i from the pretrained base model M_0; hence modifying one tuple can affect at most one teacher. At each token-query round $t \in \{1, \ldots, Q\}$, the mechanism queries all k teachers to obtain a trie-filtered vote histogram $c_t(D) = \{c_t[\tau]\}_{\tau \in \mathcal{V}_t} \in \mathbb{R}^{|\mathcal{V}_t|}$, where each coordinate $c_t[\tau]$ records the number of teachers that vote for token τ in the valid token set $\mathcal{V}_t$. $\mathcal{V}_t$ is determined by the public schema and the previously released privatized outputs, and is therefore independent of the private dataset D.

At round t, the mechanism releases two Gaussian-perturbed quantities:

$$\tilde{r}_t \;=\; r(c_t) + \mathcal{N}\big(0, (\sigma^{\mathrm{r}})^2\big), \quad r(c_t) \coloneqq \max_{\tau} c_t[\tau] - \min_{\tau} c_t[\tau], \tag{3}$$

$$\tilde{c}_t[\tau] \;=\; c_t[\tau] + \mathcal{N}\Big(0, (\sigma_t^{\mathrm{h}})^2\Big), \quad \forall \tau \in \mathcal{V}_t, \tag{4}$$

where σ_t^{h} is a deterministic function of the previously released privatized range history $H_t = (\tilde{r}_1, \ldots, \tilde{r}_t)$, as defined in §5.3.

Privacy Framework. We analyze privacy under the framework of RDP, which provides tighter accounting for adaptive Gaussian releases. During distillation, RDP is used to accumulate the privacy cost across token-query rounds until the target privacy budget is reached. Formally, if a mechanism satisfies $\varepsilon_{\mathrm{RDP}}(\alpha)$-RDP at order $\alpha > 1$, then it satisfies (ε, δ)-DP for any $\delta \in (0, 1)$, where

$$\big(\varepsilon_{\mathrm{RDP}}(\alpha) + \tfrac{\log(1/\delta)}{\alpha - 1},\ \delta\big)\text{-DP}. \tag{5}$$

For a Gaussian mechanism that releases $f(D) + \mathcal{N}(0, \sigma^2 I)$ with ℓ_2 sensitivity Δ, the corresponding RDP cost is

$$\varepsilon_{\mathrm{G}}(\alpha) = \frac{\alpha\, \Delta^2}{2\sigma^2}. \tag{6}$$

6.1 Sensitivity Bounds

We characterize the sensitivity of the quantities released at each token-query round. These bounds will be directly used in the following RDP analysis and composition.

Lemma 2 (Range sensitivity). *For any neighboring D, D' and any round t, $\big|r(c_t(D)) - r(c_t(D'))\big| \le 2$.*

Proof. Disjoint teacher partitions imply that one tuple change affects at most one teacher's vote, changing both the max and the min by at most 1; hence the difference of their gap is at most 2. Since $r(\cdot)$ outputs a scalar, its ℓ_2 sensitivity equals this bound. □

Lemma 3 (Histogram ℓ_2 sensitivity). *For any neighboring D, D' and any round t, $\|c_t(D) - c_t(D')\|_2 \le \sqrt{2}$.*

Proof. Let e_τ be the standard basis vector for coordinate τ. One vote change yields Δc_t of the form $e_{\tau_b} - e_{\tau_a}$ (valid→valid, norm $\sqrt{2}$), e_{τ_b} (invalid→valid, norm 1), or $-e_{\tau_a}$ (valid→invalid, norm 1). The worst case is $\sqrt{2}$. □

6.2 Per-round and Composed RDP Analysis

Next, we analyze the privacy cost of each token-query round and then compose across rounds.

Per-round RDP under Adaptive Noise. Conditioning on the released privatized range history $H_t = (\tilde{r}_1, \ldots, \tilde{r}_t)$, both σ^{r} and σ_t^{h} are deterministic constants. Hence, each release at round t can be analyzed as an independent Gaussian mechanism.

Proposition 1 (RDP of range release). *Releasing $\tilde{r}_t$ in (3) satisfies $\varepsilon^{\mathrm{r}}(\alpha) = \frac{2\alpha}{(\sigma^{\mathrm{r}})^2}$.*

Proof. By Lemma 2, the ℓ_2 sensitivity is $\Delta^{\mathrm{r}} = 2$. Applying the Gaussian RDP formula (6) yields the result. □

Proposition 2 (RDP of histogram release). *Conditioned on H_t, releasing $\tilde{c}_t$ in (4) satisfies $\varepsilon_t^{\mathrm{h}}(\alpha \mid H_t) = \frac{\alpha}{(\sigma_t^{\mathrm{h}})^2}$.*

Proof. By Lemma 3, the histogram sensitivity is $\Delta^{\mathrm{h}} = \sqrt{2}$. Since σ_t^{h} is fixed given H_t, the RDP cost is $\frac{\alpha(\sqrt{2})^2}{2(\sigma_t^{\mathrm{h}})^2} = \frac{\alpha}{(\sigma_t^{\mathrm{h}})^2}$. □

Corollary 1 (Per-round composition). *For any H_t, the total RDP at round t is $\varepsilon_t(\alpha \mid H_t) = \varepsilon^{\mathrm{r}}(\alpha) + \varepsilon_t^{\mathrm{h}}(\alpha \mid H_t) = \frac{2\alpha}{(\sigma^{\mathrm{r}})^2} + \frac{\alpha}{(\sigma_t^{\mathrm{h}})^2}$.*

Proof. RDP composes additively under adaptive dependence, because the histogram noise scale σ_t^{h} is a deterministic function of the privatized history H_t, while the range noise σ^{r} is fixed. □

Remark 1 (Degenerate single-token rounds). If $|\mathcal{V}_t| = 1$, the vote histogram is deterministic from the trie. The release of $\tilde{c}_t$ can be omitted (equivalently, $\sigma_t^{\mathrm{h}} = +\infty$ and $\sigma^{\mathrm{r}} = +\infty$), incurring no additional privacy cost.

Composition Across Rounds. Having established the per-round privacy guarantees, we now aggregate them over all Q query rounds using adaptive composition of RDP. Let $H_{\le Q} = (H_1, \ldots, H_Q)$ denote the entire privatized history. By adaptive composition of RDP applied to Corollary 1, the total RDP cost for Q rounds satisfies

$$\varepsilon_{\mathrm{total}}^{(Q)}(\alpha \mid H_{\le Q}) = \frac{2Q\alpha}{(\sigma^{\mathrm{r}})^2} + \sum_{t=1}^{Q} \frac{\alpha}{(\sigma_t^{\mathrm{h}})^2}. \tag{7}$$

This bound holds for every privatized history $H_{\le Q}$, and therefore for all possible executions of the mechanism.

Conversion to (ε, δ)-DP. For any $\alpha > 1$ and $\delta \in (0, 1)$, the standard conversion (5) from RDP to (ε, δ)-DP gives

$$\varepsilon(\delta \mid H_{\le Q}) = \varepsilon_{\mathrm{total}}^{(Q)}(\alpha \mid H_{\le Q}) + \frac{\log(1/\delta)}{\alpha - 1}. \tag{8}$$

6.3 End-to-End Privacy Guarantee

We now connect the per-round analysis to the overall privacy guarantee of both the distillation mechanism and the complete workflow of Prism. The first theorem establishes that the distillation process itself is differentially private, followed by a second theorem that extends this guarantee to the full end-to-end system.

Theorem 1 (Differential privacy of the distillation mechanism of Prism). *Given a privacy budget (ε, δ), ensemble size k, pretrained base model M_0, and training dataset D, the distillation mechanism of Prism, which queries k teacher models over Q token-query rounds and releases a privatized transcript $\mathcal{H} = (\tilde{r}_1, \tilde{c}_1, \ldots, \tilde{r}_Q, \tilde{c}_Q)$, satisfies $(\varepsilon(\delta), \delta)$-DP for the training dataset D.*

Proof. Consider the distillation mechanism of Prism that, at each round, queries all k teachers and accumulates privacy cost under RDP until the target budget is reached. Let Q denote the total number of token-query rounds executed before the privacy budget is exhausted. By Corollary 1 and the adaptive composition rule of RDP, the overall mechanism satisfies $\varepsilon(\alpha)$-RDP, where $\varepsilon(\alpha)$ is defined in (7). Applying the standard conversion from RDP to (ε, δ)-DP (8) yields that the transcript $\mathcal{H} = (\tilde{r}_1, \tilde{c}_1, \ldots, \tilde{r}_Q, \tilde{c}_Q)$ satisfies $(\varepsilon(\delta), \delta)$-differential privacy for the training dataset D. □

Having established the privacy of the distillation mechanism, we show that the end-to-end workflow of Algorithm 1 privately releases a synthetic dataset under (ε, δ)-differential privacy.

Theorem 2 (Differential privacy of Prism). *Given a privacy budget (ε, δ), ensemble size k, and pretrained base model M_0, and training dataset D, the randomized mechanism that maps a private relational table D to a synthetic dataset $\tilde{D}$ through the workflow in Algorithm 1 satisfies (ε, δ)-DP.*

Table 1. Dataset details used in the experiments.

Dataset	Cardinality	# Num. Attr.	# Cat. Attr.
Abalone [57]	4,177	8	1
Tax [17]	30,000	5	5
Adult [5]	32,561	3	11
Cardio [80]	70,000	5	7
Diabetes [14]	236,378	4	18

Proof. By Theorem 1, the distillation mechanism of Prism satisfies $(\varepsilon(\delta), \delta)$-differential privacy for the training dataset D, and produces a privatized transcript $\mathcal{H} = (\tilde{r}_1, \tilde{c}_1, \ldots, \tilde{r}_Q, \tilde{c}_Q)$. All subsequent operations in Algorithm 1, including thresholding, probabilistic sampling, detokenization, student training, and synthetic data generation, depend solely on $\mathcal{H}$. Since differential privacy is preserved under arbitrary post-processing [22], these steps incur no additional privacy cost. Therefore, the overall mechanism that maps the private dataset D to the released synthetic dataset $\tilde{D}$ satisfies $(\varepsilon(\delta), \delta)$-differential privacy, completing the proof. □

7 Evaluation

7.1 Evaluation Setup

Datasets. We choose five benchmarking datasets: Abalone [57], Tax [17], Adult [5], Cardio [80], and Diabetes [14], following prior work [12, 43, 55, 82, 84] to ensure fair comparison with existing methods. These datasets cover a range of cardinalities up to 10^6, dimensionalities up to 22, and diverse attribute types. Table 1 lists the details of the datasets used in our evaluation.

Baselines. We select eight state-of-the-art methods for differentially private relational data synthesis, representing the major methodological families. ① DP-LLM Full [97] and ② DP-LLM LoRA fine-tune LLMs using DP-SGD, applying full-parameter and low-rank adaptation [37], respectively. ③ PATE-CTGAN [43, 85] and ④ DP-CTGAN [84] are CTGAN [96]-based synthesizers that achieve DP by PATE and DP-SGD, respectively. ⑤ P3GM [82] is a variational autoencoder trained with DP-SGD, and ⑥ DP-TabDDPM is a diffusion-based synthesizer adapted from TabDDPM [47], with DP-SGD in the training loop. ⑦ AIM [55] is a marginal-based solution using probabilistic graphical models, and ⑧ PrivBayes [99] is a Bayesian network–based synthesizer that perturbs low-dimensional marginals to ensure DP.

We emphasize that our goal is DP data synthesis via knowledge transfer from teacher models. Without teacher signals, a pretrained LLM prompted only with schema cannot acquire dataset information and therefore does not constitute a meaningful baseline. Moreover, DistilGPT2 is not instruction-tuned and cannot generate complete tuples when prompted solely with the schema.

Evaluation Metrics. To standardize evaluation across datasets of different sizes, we uniformly sample 2,000 tuples from the generated synthetic dataset and 2,000 tuples from the real dataset. Following prior work [32], we train a suite of 9 classification models: Logistic Regression, Random Forest, AdaBoost, Gradient Boosting, XGBoost, Decision Tree, Bernoulli Naive Bayes, Multi-Layer Perceptron, and Bagging Classifier. For each classifier, we split the sampled synthetic tuples into 90% for training and split the sampled real tuples into 10% for testing (i.e., Train-on-Synthetic, Test-on-Real [26]). We report accuracy, precision, recall, F1, AUROC, and AUPRC, averaged across all 9 classifiers. All classifiers are implemented using the scikit-learn [64] and xgboost [16] libraries with default hyperparameters. As a reference, we also report the performance of models trained and tested on the true data, denoted as True.

Table 2. Parameter settings and corresponding experiments.

Parameter	Default Value	Experiment
Privacy budget (ϵ, δ)	$(6.8, 10^{-5})$	Exp. 5
Number of teachers k	1800	Exp. 8
Histogram noise scales $(\sigma^{\text{h}}_{\text{min}}, \sigma^{\text{h}}_{\text{max}})$	$(80, 130)$	Exp. 9
Range noise scale σ^{r}	500	Exp. 10
RDP orders α	Opacus [18] default	

Our work, together with most prior works in the literature [12, 43, 55, 82, 85, 99], focuses on single-table data synthesis. These works likewise evaluate using single-table metrics, and our evaluation follows this practice. Topics such as multi-table synthesis or foreign keys are important but orthogonal to the scope.

Implementation Details. We implement Prism in PyTorch 2.4.1 with the Hugging Face transformers library [28], and use the Opacus library [18] to account for DP cost [56]. Baseline codes are adopted from dp-transformers package [39] (①–②), SmartNoise toolkit [86] (③–④,⑦), authors' code (⑤ [81]–⑥ [71]), and the DataSynthesizer package [66] (⑧). Models are trained with default parameters, except that we fix random_state = 0 for reproducibility. DistilGPT2 [27] is used by default for all LLMs. All experiments are conducted on a machine with 96× vCPUs, 1TB memory, 8× A100 GPUs. We report the mean and std of 3 independent runs.

Parameter Settings. Table 2 summarizes the default parameter configuration used in our experiments and indicates which experiments vary each parameter. ① We set the default DP budget to $\epsilon = 6.8$ and $\delta = 10^{-5}$, following prior work [97]. ② The parameter α is part of the RDP accountant [56] rather than a model hyperparameter. We adopt the default configuration provided by Opacus [18], namely `DEFAULT_ALPHAS = [1 + x / 10.0 for x in range(1, 100)] + list(range(12, 64))`. ③ We set the number of teachers to $k = 1800$ by default, following prior works [88]. ④ The noise scales $\sigma^{\text{h}}_{\text{min}}$, $\sigma^{\text{h}}_{\text{max}}$, and σ^{r} are practical settings guided by the ensemble size k. Specifically, $\sigma^{\text{h}}_{\text{min}}$ and $\sigma^{\text{h}}_{\text{max}}$ bound the amount of noise added to token histograms: they must be small enough not to overwhelm the signal from k teachers, yet sufficiently large to allow a reasonable number of queries within the (ϵ, δ) budget. Since vote counts are bounded by $k = 1800$, we set $\sigma^{\text{h}}_{\text{min}} = 80$, $\sigma^{\text{h}}_{\text{max}} = 130$, and $\sigma^{\text{r}} = 500$ for all datasets. We use a single hyperparameter configuration across all datasets. The consistent performance observed across datasets indicates that the gains of Prism arise from its design rather than hyperparameter adjustment.

Pretraining Exposure to Datasets. We argue that pretraining exposure to the benchmark tables is unlikely. The backbone model DistilGPT2 [27] is pretrained on OpenWebTextCorpus [33], which consists primarily of unstructured natural-language webpages. The benchmark datasets appear in CSV or tabular form and are rarely embedded verbatim in web text; occasional isolated example rows are insufficient for meaningful memorization. In addition, the Diabetes dataset [14] postdates OpenWebText but Prism still outperforms the other baselines on this dataset. This observation is also supported by Experiment 3, which shows that Prism continues to outperform non-LLM baselines even without pretrained knowledge.

7.2 Evaluation

Experiment 1: End-to-End Comparison with Baselines. Table 3 shows the overall prediction quality of training 9 classifiers on the synthetic data and evaluating on the true dataset. All methods use the default DP budget ($\epsilon = 6.8, \delta = 10^{-5}$) to ensure fair comparisons. As Table 3 shows, Prism

Table 3. Exp. 1: Evaluation of 9 classification models trained on synthetic and evaluated on the true dataset. Prism outperforms baselines in almost all measurements, and is closest to the true reference.

Dataset	Method	F1	Accuracy	Recall	Precision	AUROC	AUPRC
Adult	True	$0.8587_{\pm 0.0015}$	$0.8435_{\pm 0.0027}$	$0.8662_{\pm 0.0011}$	$0.8558_{\pm 0.0015}$	$0.8005_{\pm 0.0053}$	$0.8804_{\pm 0.0038}$
	DP-SGD Full	$0.8036_{\pm 0.0106}$	$\mathbf{0.7903}_{\pm 0.0061}$	$0.8589_{\pm 0.0033}$	$0.7745_{\pm 0.0116}$	$0.6807_{\pm 0.0056}$	$0.8020_{\pm 0.0057}$
	DP-SGD LoRA	$0.7989_{\pm 0.0084}$	$0.7867_{\pm 0.0053}$	$0.8548_{\pm 0.0048}$	$0.7704_{\pm 0.0088}$	$0.6720_{\pm 0.0081}$	$0.7974_{\pm 0.0049}$
	DP-CTGAN	$0.7272_{\pm 0.0064}$	$0.7212_{\pm 0.0249}$	$0.7682_{\pm 0.0117}$	$0.7570_{\pm 0.0043}$	$0.6528_{\pm 0.0035}$	$0.7754_{\pm 0.0010}$
	PATE-CTGAN	$0.5499_{\pm 0.0149}$	$0.6113_{\pm 0.0100}$	$0.5676_{\pm 0.0162}$	$0.6410_{\pm 0.0112}$	$0.5199_{\pm 0.0091}$	$0.6644_{\pm 0.0076}$
	P3GM	$0.1907_{\pm 0.0394}$	$0.4086_{\pm 0.0223}$	$0.1885_{\pm 0.0568}$	$0.2897_{\pm 0.0143}$	$0.4939_{\pm 0.0146}$	$0.6512_{\pm 0.0143}$
	DP-TabDDPM	$0.5500_{\pm 0.0275}$	$0.5624_{\pm 0.0200}$	$0.6114_{\pm 0.0567}$	$0.5708_{\pm 0.0057}$	$0.5081_{\pm 0.0098}$	$0.6597_{\pm 0.0098}$
	AIM	$0.7226_{\pm 0.0093}$	$0.7284_{\pm 0.0132}$	$0.7321_{\pm 0.0126}$	$0.7336_{\pm 0.0076}$	$0.6916_{\pm 0.0003}$	$0.8111_{\pm 0.0039}$
	PrivBayes	$0.6861_{\pm 0.0111}$	$0.6807_{\pm 0.0172}$	$0.7089_{\pm 0.0166}$	$0.7281_{\pm 0.0153}$	$0.6780_{\pm 0.0153}$	$0.8040_{\pm 0.0010}$
	Prism	$\mathbf{0.8222}_{\pm 0.0052}$	$0.7873_{\pm 0.0051}$	$\mathbf{0.8696}_{\pm 0.0121}$	$\mathbf{0.7870}_{\pm 0.0036}$	$\mathbf{0.6941}_{\pm 0.0106}$	$\mathbf{0.8172}_{\pm 0.0047}$
Tax	True	$0.8223_{\pm 0.0086}$	$0.8768_{\pm 0.0061}$	$0.8095_{\pm 0.0157}$	$0.8575_{\pm 0.0151}$	$0.8951_{\pm 0.0040}$	$0.8665_{\pm 0.0075}$
	DP-SGD Full	$0.5417_{\pm 0.0106}$	$0.6362_{\pm 0.0073}$	$0.5618_{\pm 0.0147}$	$0.5755_{\pm 0.0128}$	$0.6099_{\pm 0.0134}$	$0.6022_{\pm 0.0149}$
	DP-SGD LoRA	$0.5555_{\pm 0.0026}$	$0.6612_{\pm 0.0047}$	$0.5819_{\pm 0.0078}$	$0.5724_{\pm 0.0105}$	$0.6427_{\pm 0.0168}$	$0.6108_{\pm 0.0033}$
	DP-CTGAN	$0.4862_{\pm 0.0798}$	$0.5854_{\pm 0.0278}$	$0.5638_{\pm 0.1685}$	$0.4824_{\pm 0.0307}$	$0.5499_{\pm 0.0376}$	$0.5396_{\pm 0.0296}$
	PATE-CTGAN	$0.2792_{\pm 0.0181}$	$0.5072_{\pm 0.0104}$	$0.3023_{\pm 0.0245}$	$0.3283_{\pm 0.0153}$	$0.5483_{\pm 0.0146}$	$0.5335_{\pm 0.0161}$
	P3GM	$0.2177_{\pm 0.0009}$	$0.5047_{\pm 0.0011}$	$0.2540_{\pm 0.0016}$	$0.2765_{\pm 0.0105}$	$0.4966_{\pm 0.0033}$	$0.4860_{\pm 0.0023}$
	DP-TabDDPM	$0.4340_{\pm 0.0543}$	$0.5313_{\pm 0.0115}$	$0.5018_{\pm 0.0937}$	$0.4976_{\pm 0.0130}$	$0.5423_{\pm 0.0206}$	$0.5222_{\pm 0.0144}$
	AIM	$0.4105_{\pm 0.0112}$	$0.6003_{\pm 0.0062}$	$0.4189_{\pm 0.0126}$	$0.4271_{\pm 0.0109}$	$0.6526_{\pm 0.0040}$	$\mathbf{0.6414}_{\pm 0.0044}$
	PrivBayes	$0.3597_{\pm 0.0011}$	$0.6086_{\pm 0.0096}$	$0.3606_{\pm 0.0119}$	$0.3906_{\pm 0.0079}$	$0.6438_{\pm 0.0041}$	$0.6304_{\pm 0.0043}$
	Prism	$\mathbf{0.5776}_{\pm 0.0076}$	$\mathbf{0.6637}_{\pm 0.0119}$	$\mathbf{0.5995}_{\pm 0.0011}$	$\mathbf{0.5958}_{\pm 0.0099}$	$\mathbf{0.6669}_{\pm 0.0148}$	$0.6235_{\pm 0.0174}$
Abalone	True	$0.8988_{\pm 0.0029}$	$0.8797_{\pm 0.0040}$	$0.8910_{\pm 0.0038}$	$0.9189_{\pm 0.0047}$	$0.9184_{\pm 0.0023}$	$0.9426_{\pm 0.0040}$
	DP-SGD Full	$0.6845_{\pm 0.0022}$	$0.6207_{\pm 0.0055}$	$0.8308_{\pm 0.0108}$	$0.6330_{\pm 0.0062}$	$0.5961_{\pm 0.0197}$	$0.7144_{\pm 0.0115}$
	DP-SGD LoRA	$0.5811_{\pm 0.0081}$	$0.5726_{\pm 0.0131}$	$0.6742_{\pm 0.0079}$	$0.5873_{\pm 0.0034}$	$0.5937_{\pm 0.0284}$	$0.7300_{\pm 0.0108}$
	DP-CTGAN	$0.7207_{\pm 0.0138}$	$0.6405_{\pm 0.0143}$	$\mathbf{0.8576}_{\pm 0.0091}$	$0.6966_{\pm 0.0164}$	$0.6538_{\pm 0.0035}$	$0.7598_{\pm 0.0065}$
	PATE-CTGAN	$0.7271_{\pm 0.0052}$	$0.6428_{\pm 0.0049}$	$0.8441_{\pm 0.0083}$	$0.6710_{\pm 0.0132}$	$0.6737_{\pm 0.0139}$	$0.7675_{\pm 0.0110}$
	P3GM	$0.1190_{\pm 0.0050}$	$0.4167_{\pm 0.0304}$	$0.1299_{\pm 0.0043}$	$0.1679_{\pm 0.0171}$	$0.5169_{\pm 0.0210}$	$0.6814_{\pm 0.0066}$
	DP-TabDDPM	$0.5772_{\pm 0.0129}$	$0.5566_{\pm 0.0142}$	$0.5262_{\pm 0.0293}$	$0.7526_{\pm 0.0104}$	$0.5723_{\pm 0.0269}$	$0.7259_{\pm 0.0108}$
	AIM	$\mathbf{0.7386}_{\pm 0.0113}$	$0.6481_{\pm 0.0141}$	$0.8080_{\pm 0.0193}$	$0.7105_{\pm 0.0101}$	$0.6405_{\pm 0.0184}$	$0.7635_{\pm 0.0110}$
	PrivBayes	$0.5073_{\pm 0.0058}$	$0.5728_{\pm 0.0006}$	$0.5258_{\pm 0.0012}$	$0.6431_{\pm 0.0085}$	$0.6629_{\pm 0.0281}$	$0.7653_{\pm 0.0205}$
	Prism	$0.6843_{\pm 0.0152}$	$\mathbf{0.6546}_{\pm 0.0121}$	$0.6951_{\pm 0.0168}$	$\mathbf{0.7546}_{\pm 0.0103}$	$\mathbf{0.6913}_{\pm 0.0156}$	$\mathbf{0.7943}_{\pm 0.0128}$
Cardio	True	$0.7635_{\pm 0.0114}$	$0.7994_{\pm 0.0136}$	$0.7909_{\pm 0.0054}$	$0.7437_{\pm 0.0200}$	$0.6490_{\pm 0.0149}$	$0.7784_{\pm 0.0125}$
	DP-SGD Full	$0.7163_{\pm 0.0022}$	$0.7629_{\pm 0.0022}$	$0.7655_{\pm 0.0066}$	$0.6936_{\pm 0.0100}$	$0.5614_{\pm 0.0085}$	$0.7224_{\pm 0.0025}$
	DP-SGD LoRA	$0.6980_{\pm 0.0056}$	$0.7679_{\pm 0.0055}$	$0.7464_{\pm 0.0036}$	$0.6906_{\pm 0.0139}$	$0.5168_{\pm 0.0122}$	$0.7089_{\pm 0.0076}$
	DP-CTGAN	$0.6063_{\pm 0.0120}$	$0.6797_{\pm 0.0120}$	$0.6060_{\pm 0.0060}$	$0.6917_{\pm 0.0159}$	$0.5617_{\pm 0.0148}$	$0.7252_{\pm 0.0070}$
	PATE-CTGAN	$0.7332_{\pm 0.0049}$	$0.7346_{\pm 0.0048}$	$0.8110_{\pm 0.0153}$	$0.6814_{\pm 0.0078}$	$0.4793_{\pm 0.0147}$	$0.6930_{\pm 0.0033}$
	P3GM	$0.3516_{\pm 0.0156}$	$0.5287_{\pm 0.0016}$	$0.3492_{\pm 0.0199}$	$0.4580_{\pm 0.0114}$	$0.4751_{\pm 0.0142}$	$0.7074_{\pm 0.0017}$
	DP-TabDDPM	$0.7332_{\pm 0.0055}$	$0.7563_{\pm 0.0080}$	$0.7725_{\pm 0.0094}$	$0.7149_{\pm 0.0086}$	$0.5768_{\pm 0.0135}$	$0.7364_{\pm 0.0033}$
	AIM	$0.7275_{\pm 0.0227}$	$0.7637_{\pm 0.0167}$	$0.7851_{\pm 0.0244}$	$0.6979_{\pm 0.0087}$	$\mathbf{0.5841}_{\pm 0.0365}$	$0.7389_{\pm 0.0056}$
	PrivBayes	$0.7285_{\pm 0.0030}$	$0.7613_{\pm 0.0015}$	$0.7618_{\pm 0.0194}$	$0.7148_{\pm 0.0159}$	$0.5307_{\pm 0.0002}$	$0.7323_{\pm 0.0155}$
	Prism	$\mathbf{0.7533}_{\pm 0.0049}$	$\mathbf{0.7683}_{\pm 0.0030}$	$\mathbf{0.7997}_{\pm 0.0099}$	$\mathbf{0.7204}_{\pm 0.0035}$	$0.5532_{\pm 0.0018}$	$\mathbf{0.7418}_{\pm 0.0032}$
Diabetes	True	$0.8040_{\pm 0.0056}$	$0.7672_{\pm 0.0059}$	$0.8299_{\pm 0.0053}$	$0.7895_{\pm 0.0074}$	$0.6933_{\pm 0.0134}$	$0.8308_{\pm 0.0072}$
	DP-SGD Full	$0.7864_{\pm 0.0022}$	$0.7452_{\pm 0.0007}$	$0.8334_{\pm 0.0013}$	$0.7551_{\pm 0.0031}$	$0.6022_{\pm 0.0149}$	$0.7845_{\pm 0.0034}$
	DP-SGD LoRA	$0.7899_{\pm 0.0003}$	$0.7479_{\pm 0.0042}$	$0.8392_{\pm 0.0001}$	$0.7567_{\pm 0.0022}$	$0.6216_{\pm 0.0139}$	$0.7906_{\pm 0.0039}$
	DP-CTGAN	$0.7576_{\pm 0.0073}$	$0.7208_{\pm 0.0080}$	$0.7897_{\pm 0.0058}$	$0.7580_{\pm 0.0107}$	$0.5499_{\pm 0.0086}$	$0.7686_{\pm 0.0093}$
	PATE-CTGAN	$0.7868_{\pm 0.0000}$	$0.7260_{\pm 0.0017}$	$\mathbf{0.8668}_{\pm 0.0055}$	$0.7333_{\pm 0.0015}$	$0.5325_{\pm 0.0065}$	$0.7474_{\pm 0.0002}$
	P3GM	$0.5866_{\pm 0.0105}$	$0.6174_{\pm 0.0096}$	$0.5786_{\pm 0.0144}$	$0.6441_{\pm 0.0121}$	$0.5007_{\pm 0.0134}$	$0.7438_{\pm 0.0099}$
	DP-TabDDPM	$0.7019_{\pm 0.0086}$	$0.6846_{\pm 0.0078}$	$0.7064_{\pm 0.0093}$	$0.7589_{\pm 0.0110}$	$0.5709_{\pm 0.0101}$	$0.7718_{\pm 0.0087}$
	AIM	$0.7910_{\pm 0.0042}$	$0.7526_{\pm 0.0057}$	$0.8176_{\pm 0.0007}$	$0.7755_{\pm 0.0135}$	$\mathbf{0.6451}_{\pm 0.0150}$	$0.8097_{\pm 0.0066}$
	PrivBayes	$0.7370_{\pm 0.0016}$	$0.7297_{\pm 0.0029}$	$0.7570_{\pm 0.0057}$	$0.7597_{\pm 0.0069}$	$0.6433_{\pm 0.0029}$	$0.8078_{\pm 0.0004}$
	Prism	$\mathbf{0.7939}_{\pm 0.0042}$	$\mathbf{0.7550}_{\pm 0.0020}$	$0.8069_{\pm 0.0104}$	$\mathbf{0.7934}_{\pm 0.0043}$	$0.6066_{\pm 0.0110}$	$\mathbf{0.8103}_{\pm 0.0042}$

outperforms all baselines in almost all metrics across all benchmarking datasets. Notably, it consistently achieves the highest AUROC and AUPRC scores across almost all datasets, demonstrating

Table 4. Exp. 2: Ablation study of PRISM's components on Adult: token pruning (§ 5.2), data-dependent DP (§ 5.3), and probabilistic sampling with threshold filtering (§ 5.4).

Method	F1	Accuracy	Recall	Precision	AUROC	AUPRC
PRISM w/o data-dep. DP	$0.8189_{\pm 0.00}$	$0.7818_{\pm 0.00}$	$0.8655_{\pm 0.00}$	$0.7843_{\pm 0.00}$	$0.6934_{\pm 0.00}$	$0.8162_{\pm 0.00}$
PRISM w/o prb. smp. thr. flt.	$0.8206_{\pm 0.00}$	$0.7749_{\pm 0.00}$	$\mathbf{0.9192}_{\pm 0.00}$	$0.7580_{\pm 0.00}$	$0.6726_{\pm 0.00}$	$0.7932_{\pm 0.00}$
PRISM w/o token pruning	$0.8141_{\pm 0.00}$	$0.7819_{\pm 0.01}$	$0.8642_{\pm 0.00}$	$0.7772_{\pm 0.01}$	$0.6884_{\pm 0.01}$	$0.8135_{\pm 0.00}$
PRISM	$\mathbf{0.8222}_{\pm 0.01}$	$\mathbf{0.7873}_{\pm 0.01}$	$0.8696_{\pm 0.01}$	$\mathbf{0.7870}_{\pm 0.00}$	$\mathbf{0.6941}_{\pm 0.01}$	$\mathbf{0.8172}_{\pm 0.00}$

Table 5. Exp. 3: Comparison of PRISM with PRISM using the randomly initialized LLM using the Adult dataset.

Method	F1	Accuracy	Recall	Precision	AUROC	AUPRC
PRISM w/ rand-init LLM	$0.7988_{\pm 0.01}$	$0.7599_{\pm 0.01}$	$0.8557_{\pm 0.01}$	$0.7568_{\pm 0.00}$	$0.6748_{\pm 0.01}$	$0.7923_{\pm 0.01}$
PRISM	$\mathbf{0.8222}_{\pm 0.01}$	$\mathbf{0.7873}_{\pm 0.01}$	$\mathbf{0.8696}_{\pm 0.01}$	$\mathbf{0.7870}_{\pm 0.00}$	$\mathbf{0.6941}_{\pm 0.01}$	$\mathbf{0.8172}_{\pm 0.00}$

superior preservation of class ranking and separability for downstream tasks. Furthermore, PRISM's results are closest to the true reference, showing the high utility of the produced synthetic data.

Experiment 2: Ablation Study of PRISM Components. Table 4 presents an ablation study evaluating the contribution of each core technique (§ 5). We observe consistent trends across datasets and hence report only Adult as a representative example for the rest.

As shown in Table 4, each component individually contributes to the overall predictive quality, and their combination in PRISM yields the strongest performance across all metrics. Among the components, replacing probabilistic sampling and threshold filtering with PATE's noisy max-vote aggregation leads to the most significant degradation, which performs worst in 4 out of 6 metrics. However, this variant achieves the highest recall (i.e., 0.9192) because deterministically selecting the max-vote token at each step causes the model to over-predict frequent values, artificially boosting recall while hurting ranking-based metrics like AUROC and AUPRC.

Experiment 3: Impact of Pretrained Knowledge. This experiment aims to support our claim in § 1 that pretrained LLMs benefit data synthesis with prior knowledge and strong model capacity.

We disentangle the contribution of pretrained knowledge from that of our tuple distillation mechanism by constructing a variant, PRISM w/ rand-init LLM, which removes pretrained knowledge by replacing PRISM's pretrained DistilGPT2 with a randomly initialized LLM of identical architecture. As shown in Table 5, PRISM outperforms PRISM w/ rand-init LLM, demonstrating the substantial contribution of pretrained knowledge.

Experiment 4: Effectiveness of Teacher-Student Knowledge Transfer.

PRISM transfers knowledge from the teacher ensemble to the student by privately distilling a small set of tuples $\hat{D}$, which is then used to train the student model to produce $\tilde{D}$. Although $|\hat{D}| < |D|$, the distilled set satisfies differential privacy and could, in principle, be used directly as the final output. We construct two variants: PRISM teacher, which directly returns $\hat{D}$, and PRISM teacher-sampling, which returns $\tilde{D}$ by sampling from $\hat{D}$ with replacement to match the original dataset size such that $|\tilde{D}| = |D|$.

As shown in Table 6, PRISM performs on par with PRISM teacher, showing successful knowledge transfer. PRISM outperforms PRISM teacher-sampling, indicating better generalization. Specifically, out of the 32,561 tuples in the synthetic dataset, PRISM produces 29,035 unique tuples, while PRISM teacher-sampling only generates 501 unique tuples with less diversity.

Table 6. Exp. 4: Comparison of alternative knowledge transfer methods on Adult: PRISM teacher directly outputs distilled tuples, and PRISM teacher-sampling samples from distilled tuples with replacement.

Method	F1	Accuracy	Recall	Precision	AUROC	AUPRC
True	$0.8587_{\pm 0.00}$	$0.8435_{\pm 0.00}$	$0.8662_{\pm 0.00}$	$0.8558_{\pm 0.00}$	$0.8005_{\pm 0.01}$	$0.8804_{\pm 0.00}$
PRISM teacher	$\mathbf{0.8295}_{\pm 0.00}$	$\mathbf{0.7944}_{\pm 0.01}$	$\mathbf{0.8793}_{\pm 0.01}$	$\mathbf{0.7925}_{\pm 0.00}$	$\mathbf{0.7046}_{\pm 0.01}$	$\mathbf{0.8227}_{\pm 0.01}$
PRISM teacher-sampling	$0.8221_{\pm 0.00}$	$0.7837_{\pm 0.00}$	$0.8747_{\pm 0.00}$	$0.7829_{\pm 0.00}$	$0.6826_{\pm 0.00}$	$0.8123_{\pm 0.00}$
PRISM	$0.8222_{\pm 0.01}$	$0.7873_{\pm 0.01}$	$0.8696_{\pm 0.01}$	$0.7870_{\pm 0.00}$	$0.6941_{\pm 0.01}$	$0.8172_{\pm 0.00}$

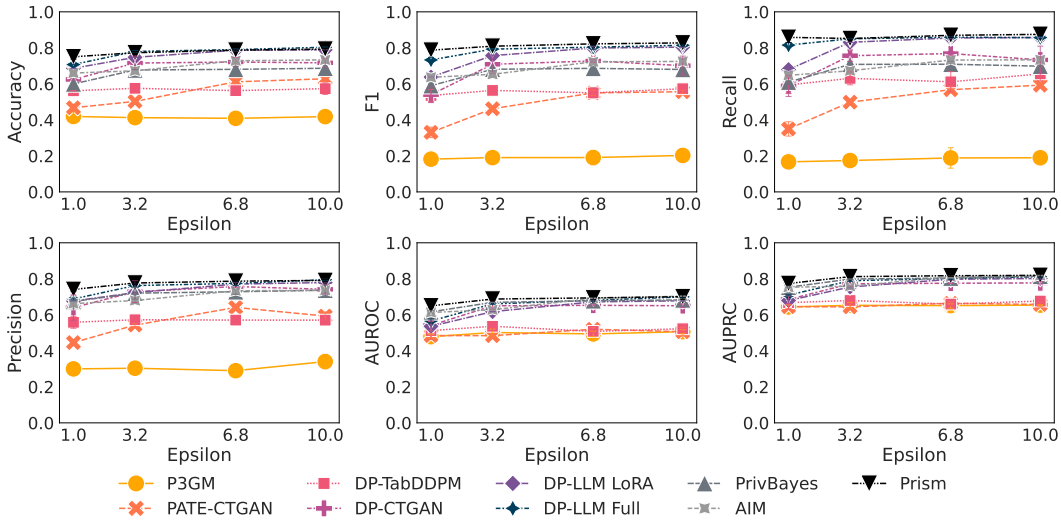


Fig. 2. Exp. 5: Comparison of PRISM with baselines at $\epsilon \in \{1.0, 3.2, 6.8, 10.0\}$ on Adult.

Table 7. Exp. 6: Impact of base LLMs on the Adult dataset.

Base Model	F1	Accuracy	Recall	Precision	AUROC	AUPRC
True	$0.8587_{\pm 0.00}$	$0.8435_{\pm 0.00}$	$0.8662_{\pm 0.00}$	$0.8558_{\pm 0.00}$	$0.8005_{\pm 0.01}$	$0.8804_{\pm 0.00}$
GPT2	$\mathbf{0.8232}_{\pm 0.01}$	$\mathbf{0.7890}_{\pm 0.00}$	$0.8681_{\pm 0.01}$	$\mathbf{0.7880}_{\pm 0.00}$	$\mathbf{0.7078}_{\pm 0.00}$	$\mathbf{0.8220}_{\pm 0.00}$
DistilGPT2	$0.8222_{\pm 0.01}$	$0.7873_{\pm 0.01}$	$\mathbf{0.8696}_{\pm 0.01}$	$0.7870_{\pm 0.00}$	$0.6941_{\pm 0.01}$	$0.8172_{\pm 0.00}$
GPT-Neo	$0.8167_{\pm 0.00}$	$0.7835_{\pm 0.00}$	$0.8670_{\pm 0.00}$	$0.7782_{\pm 0.01}$	$0.6934_{\pm 0.00}$	$0.8090_{\pm 0.00}$

Table 8. Exp. 7: Comparison of PRISM with and without noise adding on the Adult dataset.

Method	F1	Accuracy	Recall	Precision	AUROC	AUPRC
True	$0.8587_{\pm 0.00}$	$0.8435_{\pm 0.00}$	$0.8662_{\pm 0.00}$	$0.8558_{\pm 0.00}$	$0.8005_{\pm 0.01}$	$0.8804_{\pm 0.00}$
PRISM w/o DP	$0.8169_{\pm 0.01}$	$\mathbf{0.7972}_{\pm 0.00}$	$0.8448_{\pm 0.01}$	$\mathbf{0.7950}_{\pm 0.01}$	$\mathbf{0.7098}_{\pm 0.01}$	$\mathbf{0.8194}_{\pm 0.00}$
PRISM	$\mathbf{0.8222}_{\pm 0.01}$	$0.7873_{\pm 0.01}$	$\mathbf{0.8696}_{\pm 0.01}$	$0.7870_{\pm 0.00}$	$0.6941_{\pm 0.01}$	$0.8172_{\pm 0.00}$

Experiment 5: Varying DP Budgets. Figure 2 reports results under DP budgets $\epsilon \in \{1.0, 3.2, 6.8, 10.0\}$ on the Adult dataset.

This experiment demonstrates the privacy–utility trade-off: smaller ϵ enforces stronger privacy at the cost of reduced utility, whereas larger ϵ relaxes privacy and improves utility. Across all

Table 9. Exp. 8: Comparison of Prism using different number of teachers on the Adult dataset.

# Teachers	F1	Accuracy	Recall	Precision	AUROC	AUPRC
True	$0.8587_{\pm 0.00}$	$0.8435_{\pm 0.00}$	$0.8662_{\pm 0.00}$	$0.8558_{\pm 0.00}$	$0.8005_{\pm 0.01}$	$0.8804_{\pm 0.00}$
600	$0.8145_{\pm 0.01}$	$\mathbf{0.7944}_{\pm 0.00}$	$0.8403_{\pm 0.01}$	$0.8001_{\pm 0.00}$	$\mathbf{0.7209}_{\pm 0.00}$	$\mathbf{0.8286}_{\pm 0.00}$
1000	$\mathbf{0.8268}_{\pm 0.00}$	$0.7938_{\pm 0.00}$	$0.8635_{\pm 0.01}$	$\mathbf{0.8004}_{\pm 0.00}$	$0.7156_{\pm 0.00}$	$0.8277_{\pm 0.00}$
1400	$0.8237_{\pm 0.00}$	$0.7889_{\pm 0.01}$	$0.8685_{\pm 0.00}$	$0.7914_{\pm 0.00}$	$0.7031_{\pm 0.00}$	$0.8187_{\pm 0.01}$
1800	$0.8222_{\pm 0.01}$	$0.7873_{\pm 0.01}$	$\mathbf{0.8696}_{\pm 0.01}$	$0.7870_{\pm 0.00}$	$0.6941_{\pm 0.01}$	$0.8172_{\pm 0.00}$
2200	$0.8164_{\pm 0.00}$	$0.7833_{\pm 0.00}$	$0.8626_{\pm 0.00}$	$0.7813_{\pm 0.00}$	$0.6940_{\pm 0.00}$	$0.8129_{\pm 0.00}$

Table 10. Exp. 9: Comparison of Prism under different values of σ^{r} on the Adult dataset.

σ^{r}	F1	Accuracy	Recall	Precision	AUROC	AUPRC
True	$0.8587_{\pm 0.00}$	$0.8435_{\pm 0.00}$	$0.8662_{\pm 0.00}$	$0.8558_{\pm 0.00}$	$0.8005_{\pm 0.01}$	$0.8804_{\pm 0.00}$
200	$0.8170_{\pm 0.00}$	$0.7810_{\pm 0.00}$	$0.8624_{\pm 0.01}$	$0.7832_{\pm 0.00}$	$0.6935_{\pm 0.00}$	$0.8127_{\pm 0.00}$
500	$\mathbf{0.8222}_{\pm 0.01}$	$\mathbf{0.7873}_{\pm 0.01}$	$0.8696_{\pm 0.01}$	$\mathbf{0.7870}_{\pm 0.00}$	$\mathbf{0.6941}_{\pm 0.01}$	$0.8172_{\pm 0.00}$
800	$0.8208_{\pm 0.00}$	$0.7846_{\pm 0.00}$	$0.8658_{\pm 0.00}$	$0.7866_{\pm 0.00}$	$0.6929_{\pm 0.01}$	$\mathbf{0.8181}_{\pm 0.01}$
1100	$0.8190_{\pm 0.00}$	$0.7844_{\pm 0.00}$	$\mathbf{0.8700}_{\pm 0.00}$	$0.7796_{\pm 0.01}$	$0.6861_{\pm 0.01}$	$0.8123_{\pm 0.00}$

privacy levels and metrics, Prism consistently achieves the best or near-best utility. Notably, even under the strictest setting of $\epsilon = 1.0$, Prism outperforms all baselines, which implies its utility in high-privacy regimes. As ϵ increases, while the performance of all models improves, Prism remains consistently at or near the top across all metrics. This set of experiments demonstrates Prism's robustness under different privacy budgets.

Experiment 6: Varying LLMs. Table 7 evaluates the impact of different language model backbones in Prism, including DistilGPT2 [27], GPT2 [68], and GPT-Neo 125M [7], in increasing order of model parameters. All models are trained under the default privacy budget on the Adult dataset.

Comparing Prism using DistilGPT2 with GPT2, Table 7 shows that the predictive utility of the synthetic data increases with more powerful LLMs. This, in addition to Experiment 3, supports the claim from § 1 that pretrained LLMs can bring rich prior knowledge and offer strong representational capacity to model complex correlations. Notably, although Prism with GPT-Neo 125M shows lowest scores, it still outperforms all non-Prism baselines across all metrics (Table 3). This demonstrates that Prism's sampling and aggregation pipeline remains effective even with less-capable LLMs.

Experiment 7: Impact of Noise on Aggregation. This experiment evaluates how DP affects predictive utility in Prism. We compare Prism with default DP budget to a variant that omits noise when constructing token histograms, denoted as Prism w/o DP.

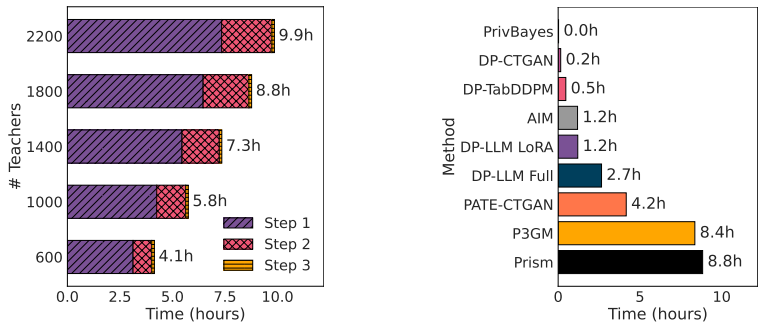
As shown in Table 8, Prism w/o DP achieves results very close to the default Prism, which suggests that Prism maintains strong utility even when a reasonable amount of noise is applied.

Experiment 8: Varying the Number of Teachers. This experiment evaluates how the number of teacher LLMs in the ensemble affects predictive utility. Given a dataset D and k teacher models, each teacher is trained on a partition of approximately $\left\lfloor \frac{|D|}{k} \right\rfloor$ tuples. A smaller k increases the maximum possible vote per token, but with the same noise scale, this results in higher perturbation, potentially masking the true signal. Conversely, a larger k reduces the data per teacher, weakening their individual predictive quality.

Table 9 reports Prism's utility across varying teacher counts $k \in \{600, 1000, 1400, 1800, 2200\}$. We observe a sweet spot; for example, 1000 teachers yield the best overall F1 (0.8268), while 1800

Table 11. Exp. 10: Comparison of Prism under different histogram noise scales $(\sigma^{\mathrm{h}}_{\min}, \sigma^{\mathrm{h}}_{\max})$ on the Adult dataset.

$(\sigma^{\mathrm{h}}_{\min}, \sigma^{\mathrm{h}}_{\max})$	F1	Accuracy	Recall	Precision	AUROC	AUPRC
True	$0.8587_{\pm 0.00}$	$0.8435_{\pm 0.00}$	$0.8662_{\pm 0.00}$	$0.8558_{\pm 0.00}$	$0.8005_{\pm 0.01}$	$0.8804_{\pm 0.00}$
(40, 90)	$0.8082_{\pm 0.00}$	$0.7860_{\pm 0.00}$	$0.8436_{\pm 0.00}$	$0.7795_{\pm 0.00}$	$0.6942_{\pm 0.01}$	$0.8131_{\pm 0.00}$
(80, 130)	$0.8222_{\pm 0.01}$	$0.7873_{\pm 0.01}$	$0.8696_{\pm 0.01}$	$\mathbf{0.7870}_{\pm 0.00}$	$0.6941_{\pm 0.01}$	$\mathbf{0.8172}_{\pm 0.00}$
(120, 170)	$\mathbf{0.8242}_{\pm 0.00}$	$\mathbf{0.7895}_{\pm 0.00}$	$\mathbf{0.8740}_{\pm 0.01}$	$0.7868_{\pm 0.01}$	$\mathbf{0.7126}_{\pm 0.01}$	$0.8157_{\pm 0.00}$
(160, 210)	$0.8168_{\pm 0.01}$	$0.7868_{\pm 0.01}$	$0.8571_{\pm 0.00}$	$0.7861_{\pm 0.01}$	$0.6912_{\pm 0.01}$	$0.8169_{\pm 0.00}$



(a) Prism runtime breakdown by various number of teachers.

(b) End-to-end runtime of Prism and baselines.

Fig. 3. Exp. 11: Runtime evaluation using Adult. (a) The most time-consuming step is teacher model fine-tuning, particularly as the number of teachers increases. (Step 1: fine-tuning teachers; Step 2: tuple distillation; Step 3: fine-tuning student and data synthesis). (b) Prism offers better utility at the cost of longer runtime.

Table 12. Exp. 12: Comparison with Kamino with zero violations to the two denial constraints [32] on the Adult dataset.

Method	F1	Accuracy	Recall	Precision	AUROC	AUPRC
Kamino	$0.7385_{\pm 0.00}$	$0.7222_{\pm 0.00}$	$0.7276_{\pm 0.00}$	$0.7712_{\pm 0.00}$	$0.6474_{\pm 0.00}$	$0.7806_{\pm 0.00}$
Prism Kamino	$\mathbf{0.7982}_{\pm 0.03}$	$\mathbf{0.7719}_{\pm 0.02}$	$\mathbf{0.8309}_{\pm 0.04}$	$\mathbf{0.7872}_{\pm 0.00}$	$\mathbf{0.6830}_{\pm 0.00}$	$\mathbf{0.8023}_{\pm 0.01}$

teachers achieve the highest recall (0.8696). These results suggest that moderate ensemble sizes (e.g., 600–1000) offer the best balance between privacy and teacher quality.

Experiment 9: Different Range Noise Scales σ^{r}. Table 10 reports results under different values of the range-noise parameter $\sigma^{\mathrm{r}} \in \{200, 500, 800, 1100\}$ on the Adult dataset.

We observe that moderate values of σ^{r} (i.e., $\sigma^{\mathrm{r}} \in \{500, 800\}$) lead to the best overall utility across most metrics. Smaller values incur higher privacy budget consumption at the range-query stage, while larger values introduce excessive noise into the range estimate.

Experiment 10: Different Histogram Noise Scales $(\sigma^{\mathrm{h}}_{\min}, \sigma^{\mathrm{h}}_{\max})$. Table 11 reports results under different histogram noise scale ranges $(\sigma^{\mathrm{h}}_{\min}, \sigma^{\mathrm{h}}_{\max}) \in \{(40, 90), (80, 130), (120, 170), (160, 210)\}$.

We observe that moderate noise ranges (i.e., (80, 130) and (120, 170)) achieve the best overall performance across most metrics. Smaller noise ranges consume more privacy budget per query and therefore limit the number of tuples, while larger noise ranges introduce excessive perturbation, leading to degraded utility.

Experiment 11: Comparison on Efficiency. We evaluate the computational efficiency of Prism and analyze how its runtime varies with the number of teacher models. Figure 3a shows the runtime breakdown of Prism across its three main stages: teacher model fine-tuning (Step 1), tuple distillation (Step 2), and student model fine-tuning and data synthesis (Step 3), under different numbers of teachers. From Figure 3a, we observe that Step 1 (teacher fine-tuning) dominates the total runtime, and the cost increases nearly linearly with the number of teachers. Using 600 teachers reduces the total runtime to about half that of 1800 teachers (4.1h vs. 8.8h), while maintaining nearly identical utility, shown in Table 9.

Figure 3b compares the end-to-end runtime of Prism with existing baseline methods on the Adult dataset under the same privacy budget. As shown in Figure 3b, Prism requires longer runtime than lightweight methods, yet consistently achieves higher downstream utility (Table 3), indicating a favorable trade-off between computational cost and synthetic data quality.

Experiment 12: Comparison with Kamino. Relational data has structures, such as correlations and dependencies across tuples. Kamino [32] is a prior approach that generates differentially private synthetic data while preserving such structural constraints. It employs a sampling method that rejects values violating hard constraints. In Prism, we incorporate a similar constraint-aware sampling strategy as a byproduct of building tries, which help identify constraint-compliant tokens during token histogram generations.

Table 12 compares Prism and Kamino on downstream classification tasks under two hard denial constraints, such as one constraint enforces a functional dependency between education level and education number (i.e., $\neg(t_i[edu] = t_j[edu] \wedge t_i[edu\_num] \neq t_j[edu\_num])$). Prism outperforms Kamino across all metrics, achieving higher F1 (0.7982 vs. 0.7385), accuracy (0.7719 vs. 0.7222), and recall (0.8309 vs. 0.7276), while maintaining zero constraint violations in both methods.

8 Related Work

Relational Data Synthesis with Differential Privacy. There is a rich body of literature on releasing differentially private tabular data [9, 38, 102]. Broadly, existing methods can be categorized into three groups based on the type of generative models they employ.

The first category leverages probabilistic graphical models [46] such as Bayesian networks [99] and Markov random fields [12], to estimate tuple distributions. Due to 1) the limited representational power of statistical models, and 2) the curse of dimensionality when enforcing differential privacy [24], these methods typically estimate low-dimensional marginal distributions under DP and reconstruct the high-dimensional joint distribution [55]. Approaches in this category are generally efficient and highly interpretable.

The second category employs deep generative models such as GANs [96], autoencoders [82], and diffusion models [52, 75], to improve representational capacity. With advances in differentially private training methods such as privacy accounting [56], deep generative models can be trained while satisfying DP [2]. However, these models often need to learn both meaningful tuple representations and model parameters simultaneously, at the cost of privacy.

The third and most recent category utilizes transformer-based pretrained LLMs to address the limitations of earlier approaches. However, their complexity poses challenges for standard training techniques with differential privacy [97]; for instance, applying off-the-shelf DP methods like DP-SGD [2] and parameter-efficient methods like LoRA [37] to fine-tune LLMs distorts the information encoded in LLMs and leads to suboptimal utility.

Our work falls into the third category, presenting a practical framework that preserves the benefits of LLMs while achieving DP.

Privacy Definitions and Evaluations for Relational Data Synthesis. A variety of privacy definitions and evaluation metrics have been proposed for relational data synthesis, including k-anonymity [74], l-diversity [53], t-closeness [51], and empirical metrics such as distance to the closest record (DCR) and membership inferences [78]. Most existing LLM-based relational data synthesis methods rely on empirical post-evaluations of privacy rather than proactively offering formal guarantees. For example, GReaT [8], REaLTabFormer [79], and TAPTAP [100] assess privacy using DCR, while more recently, HARMONIC [94] extends this approach by introducing an ad-hoc data leakage test. Rather than seeking empirical or heuristic privacy, Prism is designed to guarantee DP [22, 25], which offers provable, composable, and quantifiable guarantees for data privacy.

DP for machine learning models can be achieved through two main paradigms. The first is DP-SGD [2]. To the best of our knowledge, all prior work that employs language models for relational data, such as DP-TBART [13], SynLM [73], and DP-LLMTGen [90], falls into this category. The second paradigm is PATE [62]. Although PATE has been explored in the context of data synthesis such as PATE-GAN [43], extending it to LLMs introduces challenges. To the best of our understanding, Prism is the first approach to leverage PATE with LLMs for relational data synthesis.

9 Conclusion

We presented Prism, a differentially private framework for relational data synthesis with large language models via tuple distillation. By leveraging output perturbation rather than direct model perturbation, Prism makes PATE-style private aggregation practical for autoregressive synthesis of structured relational data. Extensive experiments demonstrate that Prism achieves substantially stronger predictive utility than eight state-of-the-art methods under the same privacy budget. Our results highlight the effectiveness of the proposed approach and point to a promising direction for private relational data synthesis using LLMs.

References

[1] 2016-04-27. Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation). OJ (2016-04-27).

[2] Martín Abadi, Andy Chu, Ian J. Goodfellow, H. Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. 2016. Deep Learning with Differential Privacy. In CCS. ACM, 308–318.

[3] John M. Abowd, Robert Ashmead, Ryan Cumings-Menon, Simson Garfinkel, Micah Heineck, Christine Heiss, Robert Johns, Daniel Kifer, Philip Leclerc, Ashwin Machanavajjhala, Brett Moran, William Sexton, Matthew Spence, and Pavel Zhuravlev. 2022. The 2020 Census Disclosure Avoidance System TopDown Algorithm. Technical Report. U.S. Census Bureau / Harvard Data Science Review (preprint). https://arxiv.org/abs/2204.08986

[4] Gergely Ács, Luca Melis, Claude Castelluccia, and Emiliano De Cristofaro. 2019. Differentially Private Mixture of Generative Neural Networks. TKDE 31, 6 (2019), 1109–1121.

[5] Barry Becker and Ronny Kohavi. 1996. Adult [Dataset]. UCI Machine Learning Repository. doi:10.24432/C5XW20 https://archive.ics.uci.edu/dataset/2/adult.

[6] Sarah Bird. 2020. Putting differential privacy into practice to use data responsibly. Microsoft AI Blog for Business. https://blogs.microsoft.com/ai-for-business/differential-privacy/ Accessed: March 19, 2026.

[7] Sid Black, Leo Gao, Phil Wang, Connor Leahy, and Stella Biderman. 2021. GPT-Neo: Large Scale Autoregressive Language Modeling with Mesh-TensorFlow. http://github.com/eleutherai/gpt-neo

[8] Vadim Borisov, Kathrin Seßler, Tobias Leemann, Martin Pawelczyk, and Gjergji Kasneci. 2023. Language Models are Realistic Tabular Data Generators. In ICLR. OpenReview.net.

[9] Claire McKay Bowen and Fang Liu. 2020. Comparative Study of Differentially Private Data Synthesis Methods. Statist. Sci. 35, 2 (May 2020), 280–307. doi:10.1214/19-sts742

[10] Rene De La Briandais. 1959. File searching using variable length keys. In IRE-AIEE-ACM. ACM, 295–298.

[11] U.S. Census Bureau. Accessed on 2020-11-30. LEHD Origin-Destination Employment Statistics (2002-2017). https://onthemap.ces.census.gov/

[12] Kuntai Cai, Xiaoyu Lei, Jianxin Wei, and Xiaokui Xiao. 2021. Data Synthesis via Differentially Private Markov Random Field. VLDB 14, 11 (2021), 2190–2202.

[13] Rodrigo Castellon, Achintya Gopal, Brian Bloniarz, and David Rosenberg. 2023. DP-TBART: A Transformer-based Autoregressive Model for Differentially Private Tabular Data Generation. CoRR abs/2307.10430 (2023).

[14] CDC. 2023. Diabetes Health Indicators Dataset. Kaggle. https://www.kaggle.com/datasets/julnazz/diabetes-health-indicators-dataset.

[15] Surajit Chaudhuri, Gautam Das, and Vivek R. Narasayya. 2007. Optimized stratified sampling for approximate query processing. TDS 32, 2 (2007), 9.

[16] Tianqi Chen and Carlos Guestrin. 2016. XGBoost: A Scalable Tree Boosting System. In ACM SIGKDD (KDD '16).

[17] Xu Chu, Ihab F. Ilyas, and Paolo Papotti. 2013. Discovering Denial Constraints. VLDB 6, 13 (2013), 1498–1509. http://www.vldb.org/pvldb/vol6/p1498-papotti.pdf

[18] Opacus Contributors. 2025. Opacus: Train PyTorch Models with Differential Privacy. https://opacus.ai/. Accessed: March 19, 2026.

[19] Graham Cormode and Akash Bharadwaj. 2022. Sample-and-threshold differential privacy: Histograms and applications. In AISTATS (PMLR, Vol. 151). PMLR, 1420–1431.

[20] Fida Kamal Dankar and Khaled El Emam. 2013. Practicing Differential Privacy in Health Care: A Review. Trans. Data Priv. 6 (2013), 35–67.

[21] Tuan Dinh, Yuchen Zeng, Ruisu Zhang, Ziqian Lin, Michael Gira, Shashank Rajput, Jy-yong Sohn, Dimitris S. Papailiopoulos, and Kangwook Lee. 2022. LIFT: Language-Interfaced Fine-Tuning for Non-language Machine Learning Tasks. In NeurIPS.

[22] Cynthia Dwork. 2006. Differential Privacy. In ICALP, Vol. 4052. Springer, 1–12.

[23] Cynthia Dwork, Krishnaram Kenthapadi, Frank McSherry, Ilya Mironov, and Moni Naor. 2006. Our Data, Ourselves: Privacy Via Distributed Noise Generation. In EUROCRYPT, Vol. 4004. Springer, 486–503.

[24] Cynthia Dwork, Moni Naor, Omer Reingold, Guy N. Rothblum, and Salil P. Vadhan. 2009. On the complexity of differentially private data release: efficient algorithms and hardness results. In STOC. ACM, 381–390.

[25] Cynthia Dwork and Aaron Roth. 2014. The Algorithmic Foundations of Differential Privacy. Foundations and Trends in Theoretical Computer Science 9, 3-4 (2014), 211–407.

[26] Cristóbal Esteban, Stephanie L. Hyland, and Gunnar Rätsch. 2017. Real-valued (Medical) Time Series Generation with Recurrent Conditional GANs. CoRR abs/1706.02633 (2017).

[27] Hugging Face. 2024. distilbert/distilgpt2. https://huggingface.co/distilbert/distilgpt2. Accessed: March 19, 2026.

[28] Hugging Face. 2025. Transformers Documentation (Hugging Face). https://huggingface.co/docs/transformers/en/index. Accessed: March 19, 2026.

[29] Mei Ling Fang, Devendra Singh Dhami, and Kristian Kersting. 2022. DP-CTGAN: Differentially Private Medical Data Generation Using CTGANs. In AIME (Lecture Notes in Computer Science). Springer, 178–188.

[30] Xi Fang, Weijie Xu, Fiona Anting Tan, Ziqing Hu, Jiani Zhang, Yanjun Qi, Srinivasan H. Sengamedu, and Christos Faloutsos. 2024. Large Language Models (LLMs) on Tabular Data: Prediction, Generation, and Understanding - A Survey. TMLR 2024 (2024).

[31] Chang Ge, Xi He, Ihab F. Ilyas, and Ashwin Machanavajjhala. 2019. APEx: Accuracy-Aware Differentially Private Data Exploration. In SIGMOD. 177–194.

[32] Chang Ge, Shubhankar Mohapatra, Xi He, and Ihab F. Ilyas. 2021. Kamino: Constraint-Aware Differentially Private Data Synthesis. VLDB 14, 10 (2021), 1886–1899.

[33] Aaron Gokaslan, Vanya Cohen, Ellie Pavlick, and Stefanie Tellex. 2019. OpenWebText Corpus. http://Skylion007.github.io/OpenWebTextCorpus.

[34] Samuel Haney, Ashwin Machanavajjhala, John M. Abowd, Matthew Graham, Mark Kutzbach, and Lars Vilhuber. 2017. Utility Cost of Formal Privacy for Releasing National Employer-Employee Statistics. In SIGMOD. ACM, 1339–1354.

[35] Michael Hay, Vibhor Rastogi, Gerome Miklau, and Dan Suciu. 2010. Boosting the Accuracy of Differentially Private Histograms Through Consistency. VLDB 3, 1 (2010), 1021–1032. https://vldb.org/pvldb/vol3/R91.pdf

[36] Stefan Hegselmann, Alejandro Buendia, Hunter Lang, Monica Agrawal, Xiaoyi Jiang, and David A. Sontag. 2023. TabLLM: Few-shot Classification of Tabular Data with Large Language Models. In AISTATS (PMLR, Vol. 206). PMLR, 5549–5581.

[37] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-Rank Adaptation of Large Language Models. In ICLR. OpenReview.net.

[38] Yuzheng Hu, Fan Wu, Qinbin Li, Yunhui Long, Gonzalo Munilla Garrido, Chang Ge, Bolin Ding, David A. Forsyth, Bo Li, and Dawn Song. 2024. SoK: Privacy-Preserving Data Synthesis. In SP. IEEE.

[39] Huseyin Inan, Andre Manoel, and Lukas Wutschitz. 2022. dp-transformers: Training Transformer Models with Differential Privacy. Microsoft Research / GitHub Repository. https://github.com/microsoft/dp-transformers Accessed: March 19, 2026.

[40] Yannis E. Ioannidis and Stavros Christodoulakis. 1991. On the Propagation of Errors in the Size of Join Results. In SIGMOD. ACM Press, 268–277.

[41] Ron Jarmin. 2019. Census Bureau Adopts Cutting Edge Privacy Protections for 2020 Census. U.S. Census Bureau Random Samplings Blog. https://www.census.gov/newsroom/blogs/random-samplings/2019/02/census_bureau_adopts.html Accessed: March 19, 2026.

[42] Noah M. Johnson, Joseph P. Near, and Dawn Song. 2018. Towards Practical Differential Privacy for SQL Queries. VLDB 11, 5 (2018), 526–539.

[43] James Jordon, Jinsung Yoon, and Mihaela van der Schaar. 2019. PATE-GAN: Generating Synthetic Data with Differential Privacy Guarantees. In ICLR. OpenReview.net.

[44] Peter Kairouz, Sewoong Oh, and Pramod Viswanath. 2017. The Composition Theorem for Differential Privacy. IEEE Trans. Inf. Theory 63, 6 (2017), 4037–4049.

[45] Krishnaram Kenthapadi and Thanh T. L. Tran. 2018. PriPeARL: A Framework for Privacy-Preserving Analytics and Reporting at LinkedIn. In CIKM. ACM, 2183–2191.

[46] Daphne Koller and Nir Friedman. 2009. Probabilistic Graphical Models - Principles and Techniques. MIT Press.

[47] Akim Kotelnikov, Dmitry Baranchuk, Ivan Rubachev, and Artem Babenko. 2023. TabDDPM: Modelling Tabular Data with Diffusion Models. In ICML (PMLR, Vol. 202). PMLR, 17564–17579.

[48] Ios Kotsogiannis, Yuchao Tao, Xi He, Maryam Fanaeepour, Ashwin Machanavajjhala, Michael Hay, and Gerome Miklau. 2019. PrivateSQL: A Differentially Private SQL Query Engine. VLDB 12, 11 (2019), 1371–1384.

[49] Sebastian Kruse and Felix Naumann. 2018. Efficient Discovery of Approximate Dependencies. VLDB 11, 7 (2018), 759–772.

[50] Chao Li, Gerome Miklau, Michael Hay, Andrew McGregor, and Vibhor Rastogi. 2015. The matrix mechanism: optimizing linear counting queries under differential privacy. VLDBJ 24, 6 (2015), 757–781.

[51] Ninghui Li, Tiancheng Li, and Suresh Venkatasubramanian. 2007. t-Closeness: Privacy Beyond k-Anonymity and l-Diversity. In ICDE. IEEE Computer Society, 106–115.

[52] Tongyu Liu, Ju Fan, Nan Tang, Guoliang Li, and Xiaoyong Du. 2024. Controllable Tabular Data Synthesis Using Diffusion Models. SIGMOD 2, 1 (2024), 28:1–28:29.

[53] Ashwin Machanavajjhala, Johannes Gehrke, Daniel Kifer, and Muthuramakrishnan Venkitasubramaniam. 2006. l-Diversity: Privacy Beyond k-Anonymity. In ICDE. IEEE Computer Society, 24.

[54] Ryan McKenna, Gerome Miklau, and Daniel Sheldon. 2021. Winning the NIST Contest: A scalable and general approach to differentially private synthetic data. J. Priv. Confidentiality 11, 3 (2021). doi:10.29012/JPC.778

[55] Ryan McKenna, Brett Mullins, Daniel Sheldon, and Gerome Miklau. 2022. AIM: An Adaptive and Iterative Mechanism for Differentially Private Synthetic Data. VLDB 15, 11 (2022), 2599–2612.

[56] Ilya Mironov. 2017. Rényi Differential Privacy. In CSF. IEEE Computer Society, 263–275.

[57] Warwick Nash, Tracy Sellers, Simon Talbot, Andrew Cawthorn, and Wes Ford. 1994. Abalone [Dataset]. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C55C7W.

[58] Joseph Near, David Darais, and Katie Boeckl. 2020. Differential Privacy Blog Series. NIST Privacy Engineering Program Blog. https://www.nist.gov/itl/applied-cybersecurity/privacy-engineering/collaboration-space/blog-series/differential-privacy Accessed: March 19, 2026.

[59] Kobbi Nissim, Sofya Raskhodnikova, and Adam D. Smith. 2007. Smooth sensitivity and sampling in private data analysis. In STOC, David S. Johnson and Uriel Feige (Eds.). ACM, 75–84.

[60] Carl W. Olofson. [n. d.]. Why Relational Databases Matter. https://www.idc.com/getdoc.jsp?containerId=US47408021.

[61] Thorsten Papenbrock and Felix Naumann. 2017. Data-driven Schema Normalization. In EDBT. 342–353.

[62] Nicolas Papernot, Martín Abadi, Úlfar Erlingsson, Ian J. Goodfellow, and Kunal Talwar. 2017. Semi-supervised Knowledge Transfer for Deep Learning from Private Training Data. In ICLR. OpenReview.net.

[63] Nicolas Papernot, Shuang Song, Ilya Mironov, Ananth Raghunathan, Kunal Talwar, and Úlfar Erlingsson. 2018. Scalable Private Learning with PATE. In ICLR. OpenReview.net.

[64] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. JMLR 12 (2011), 2825–2830.

[65] Andrea Pellandra and Alejandra Moreno Ramirez. 2025. The Microdata Library in 2024: Expanding Access to Data While Strengthening Privacy Protections. UNHCR Blog. https://www.unhcr.org/blogs/the-microdata-library-in-2024-expanding-access-to-data-while-strengthening-privacy-protections/ Accessed: March 19, 2026.

[66] Haoyue Ping, Julia Stoyanovich, and Bill Howe. 2017. DataSynthesizer: Privacy-Preserving Synthetic Datasets. In SSDM. ACM, 42:1–42:5.

[67] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. 2018. Improving language understanding by generative pre-training. (2018).

[68] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language Models are Unsupervised Multitask Learners. OpenAI Blog (2019). https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf URL: https://cdn.openai.com/better-language-models/language_

models_are_unsupervised_multitask_learners.pdf.
[69] Apple Machine Learning Research. 2025. Understanding Aggregate Trends for Apple Intelligence Using Differential Privacy. Apple Machine Learning Blog. https://machinelearning.apple.com/research/differential-privacy-aggregate-trends Accessed: March 19, 2026.
[70] Google Research. 2022. Federated Learning with Formal Differential Privacy Guarantees. Google Research Blog. https://research.google/blog/federated-learning-with-formal-differential-privacy-guarantees/ Accessed: March 19, 2026.
[71] Yandex Research. 2023. TabDDPM: Modelling Tabular Data with Diffusion Models — Official Implementation. GitHub Repository (yandex-research/tab-ddpm). https://github.com/yandex-research/tab-ddpm Accessed: March 19, 2026.
[72] Astrid Rheinländer, Martin Knobloch, Nicky Hochmuth, and Ulf Leser. 2010. Prefix Tree Indexing for Similarity Search and Similarity Joins on Genomic Data. In SSDBM, Vol. 6187. Springer, 519–536.
[73] Alexandre Sablayrolles, Yue Wang, and Brian Karrer. 2023. Privately generating tabular data using language models. CoRR abs/2306.04803 (2023).
[74] P. Samarati and L. Sweeney. 1998. Protecting Privacy when Disclosing Information: k-Anonymity and its Enforcement through Generalization and Suppression. (1998).
[75] Timur Sattarov, Marco Schreyer, and Damian Borth. 2023. FinDiff: Diffusion Models for Financial Tabular Data Generation. In ICAIF. ACM, 64–72.
[76] Weiyan Shi, Aiqi Cui, Evan Li, Ruoxi Jia, and Zhou Yu. 2022. Selective Differential Privacy for Language Modeling. In NAACL. Association for Computational Linguistics, 2848–2859.
[77] Weiyan Shi, Ryan Shea, Si Chen, Chiyuan Zhang, Ruoxi Jia, and Zhou Yu. 2022. Just Fine-tune Twice: Selective Differential Privacy for Large Language Models. In EMNLP, Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang (Eds.). Association for Computational Linguistics, 6327–6340.
[78] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. 2017. Membership Inference Attacks Against Machine Learning Models. In SP. IEEE Computer Society, 3–18.
[79] Aivin V. Solatorio and Olivier Dupriez. 2023. REaLTabFormer: Generating Realistic Relational and Tabular Data using Transformers. CoRR abs/2302.02041 (2023).
[80] Daria Sulianova and Others. 2019. Cardiovascular Disease Dataset. Kaggle. https://www.kaggle.com/datasets/sulianova/cardiovascular-disease-dataset.
[81] Shun Takagi, Tsubasa Takahashi, Yang Cao, and Masatoshi Yoshikawa. 2020. P3GM: Private High-Dimensional Data Release via Privacy Preserving Phased Generative Model. GitHub Repository / Project (tsubasat/P3GM). https://github.com/tsubasat/P3GM Corresponding to ICDE 2021 / preprint arXiv:2006.12101.
[82] Shun Takagi, Tsubasa Takahashi, Yang Cao, and Masatoshi Yoshikawa. 2021. P3GM: Private High-Dimensional Data Release via Privacy Preserving Phased Generative Model. In ICDE. IEEE, 169–180.
[83] Yuchao Tao, Amir Gilad, Ashwin Machanavajjhala, and Sudeepa Roy. 2022. DPXPlain: Privately Explaining Aggregate Query Answers. VLDB 16, 1 (2022), 113–126.
[84] SmartNoise Team. 2023. Differentially Private Conditional Tabular GAN (DPCTGAN) — SmartNoise Documentation. https://docs.smartnoise.org/synth/synthesizers/dpctgan.html. Accessed: March 19, 2026.
[85] SmartNoise Team. 2023. PATE-CTGAN (Conditional Tabular GAN with PATE) — SmartNoise Documentation. https://docs.smartnoise.org/synth/synthesizers/patectgan.html. Accessed: March 19, 2026.
[86] SmartNoise Team. 2025. SmartNoise: A Differential Privacy Toolkit for Analytics and Machine Learning. https://smartnoise.org/. Accessed: March 19, 2026.
[87] Uber Privacy & Security Team. 2016. Uber Releases Open Source Project for Differential Privacy. Uber Privacy / Medium Blog. https://medium.com/uber-security-privacy/differential-privacy-open-source-7892c82c42b6 Accessed: March 19, 2026.
[88] Zhiliang Tian, Yingxiu Zhao, Ziyue Huang, Yu-Xiang Wang, Nevin L. Zhang, and He He. 2022. SeqPATE: Differentially Private Text Generation via Knowledge Distillation. In NeurIPS.
[89] Florian Tramèr, Gautam Kamath, and Nicholas Carlini. 2024. Position: Considerations for Differentially Private Learning with Large-Scale Public Pretraining. In ICML. OpenReview.net.
[90] Toan V. Tran and Li Xiong. 2024. Differentially Private Tabular Data Synthesis using Large Language Models. arXiv:2406.01457 [cs.LG] https://arxiv.org/abs/2406.01457
[91] Gianluca Truda. 2023. Generating tabular datasets under differential privacy. CoRR abs/2308.14784 (2023). arXiv:2308.14784 doi:10.48550/ARXIV.2308.14784
[92] Task Team on Privacy Enhancing Techniques United Nations. 2025. UN Guide on Privacy-Enhancing Technologies for Official Statistics. UN Statistics Division / UN-CEBD Task Team on PETs. https://unstats.un.org/bigdata/task-teams/privacy/guide/ Accessed: March 19, 2026.
[93] U.S. Congress. 1996. Health Insurance Portability and Accountability Act of 1996 (HIPAA). Public Law 104-191, 110 Stat. 1936. https://www.hhs.gov/hipaa/index.html Accessed: 2025-03-20.

[94] Yuxin Wang, Duanyu Feng, Yongfu Dai, Zhengyu Chen, Jimin Huang, Sophia Ananiadou, Qianqian Xie, and Hao Wang. 2024. HARMONIC: Harnessing LLMs for Tabular Data Synthesis and Privacy Protection. In NeurIPS.
[95] Siyuan Xia, Beizhen Chang, Karl Knopf, Yihan He, Yuchao Tao, and Xi He. 2021. DPGraph: A Benchmark Platform for Differentially Private Graph Analysis. In SIGMOD. ACM, 2808–2812.
[96] Lei Xu, Maria Skoularidou, Alfredo Cuesta-Infante, and Kalyan Veeramachaneni. 2019. Modeling Tabular data using Conditional GAN. In NeurIPS. 7333–7343.
[97] Da Yu, Saurabh Naik, Arturs Backurs, Sivakanth Gopi, Huseyin A. Inan, Gautam Kamath, Janardhan Kulkarni, Yin Tat Lee, Andre Manoel, Lukas Wutschitz, Sergey Yekhanin, and Huishuai Zhang. 2022. Differentially Private Fine-tuning of Language Models. In ICLR. OpenReview.net.
[98] Quan Yuan, Zhikun Zhang, Linkang Du, Min Chen, Peng Cheng, and Mingyang Sun. 2023. PrivGraph: Differentially Private Graph Data Publication by Exploiting Community Information. In USENIX. USENIX Association, 3241–3258.
[99] Jun Zhang, Graham Cormode, Cecilia M. Procopiuc, Divesh Srivastava, and Xiaokui Xiao. 2017. PrivBayes: Private Data Release via Bayesian Networks. TDS 42, 4 (2017), 25:1–25:41.
[100] Tianping Zhang, Shaowen Wang, Shuicheng Yan, Li Jian, and Qian Liu. 2023. Generative Table Pre-training Empowers Models for Tabular Prediction. In EMNLP. Association for Computational Linguistics, 14836–14854.
[101] Zhikun Zhang, Tianhao Wang, Ninghui Li, Jean Honorio, Michael Backes, Shibo He, Jiming Chen, and Yang Zhang. 2021. PrivSyn: Differentially Private Data Synthesis. In USENIX. USENIX Association, 929–946.
[102] Tianqing Zhu, Gang Li, Wanlei Zhou, and Philip S. Yu. 2017. Differentially Private Data Publishing and Analysis: A Survey. TKDE 29, 8 (2017), 1619–1638.
[103] Wennan Zhu, Peter Kairouz, Brendan McMahan, Haicheng Sun, and Wei Li. 2020. Federated Heavy Hitters Discovery with Differential Privacy. In AISTATS (PMLR, Vol. 108). PMLR, 3837–3847.

Received October 2025; revised January 2026; accepted February 2026