RINTAW: A ROBUST INVISIBLE WATERMARK FOR TABULAR GENERATIVE MODELS

Liancheng Fang[1] , **Aiwei Liu**[2], **Henry Peng Zou**[1], **Hengrui Zhang**[1], **Philip S. Yu**[1]
[1] University of Illinois Chicago [2] Tsinghua University
lfang87@uic.edu, liuaw20@mails.tsinghua.edu.cn, psyu@uic.edu

ABSTRACT

Watermarking tabular generative models is critical for preventing misuse of synthetic tabular data. However, existing watermarking methods for tabular data often lack robustness against common attacks (e.g., row shuffling) or are limited to specific data types (e.g., numerical), restricting their practical utility. To address these challenges, we propose RINTAW, a novel watermarking framework for tabular generative models that is robust to common attacks while preserving data fidelity. RINTAW embeds watermarks by leveraging a subset of column values as seeds. To ensure the pseudorandomness of the watermark key, RINTAW employs an adaptive column selection strategy and a masking mechanism to enforce distribution uniformity. This approach guarantees minimal distortion to the original data distribution and is compatible with any tabular data format (numerical, categorical, or mixed) and generative model architecture. We validate RINTAW on six real-world tabular datasets, demonstrating that the quality of watermarked tables remains nearly indistinguishable from non-watermarked ones while achieving high detectability even under strong post-editing attacks. The code is available at this link.

1 INTRODUCTION

Recent advances in table generation models - including diffusion models (Zhang et al., 2024b; Kotelnikov et al., 2023), autoregressive models (Gulati & Roysdon, 2024; Solatorio & Dupriez, 2023), and masked generative models (Zhang et al., 2024a) - have made significant progress in generating synthetic data that closely aligns with real data distributions. These models have proven valuable in applications such as data augmentation (Lin & Tsai, 2020), privacy protection (Gascón et al., 2016), and missing data imputation (Lin & Tsai, 2020). However, the misuse of synthetic data can lead to serious issues, including financial fraud (Cartella et al., 2021) and data pollution (Padhi et al., 2021). To prevent the misuse of synthetic data and ensure data traceability and authenticity, watermarking techniques have emerged as an important solution.

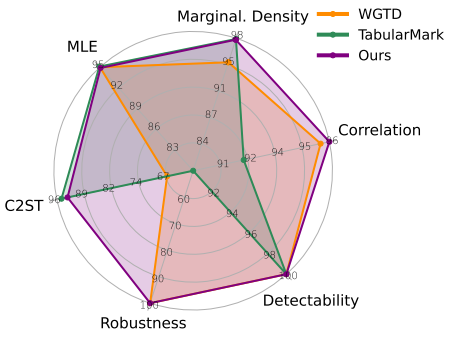

Figure 1: Our proposed RINTAW achieves the overall best performance in generation quality, detectability, and robustness.

Existing tabular watermarking approaches can be categorized into two types: post-processing watermarks and sampling-phase watermarks. Post-processing methods (Zheng et al., 2024; He et al., 2024) embed watermarks by modifying data after generation. While these methods are designed to introduce a small amount of post-editing, they may still introduce distortions to the data distribution. For example, (He et al., 2024) embeds watermarks by replacing the decimal part of all the numerical values of a table with a value from the nearest green list. Similarly, TabularMark (Zheng et al., 2024) modifies values by perturbing a small set of key cells with randomly sampled values from the green list. Although this specific design keeps the post-editing to a minimum, due to the heterogeneous nature of the tabular data, a small perturbation can possibly introduce systematic bias to the data distribution. For example, in a column with values ranging from 0 to 0.1, a decimal replacement

Table 1: Comparison of our method with baseline methods.

Method	Phase	Data modality	Distribution-preserving	Robust
Zheng et al. (2024)	Post-processing	Cat+Num	✓	✗
He et al. (2024)	Post-processing	Num only	✗	✓
RINTAW (Ours)	Sampling	Cat+Num	✓	✓

will create out-of-range values, completely invalidating the generated data. Sampling-phase watermarking methods, on the other hand, embed watermarks during the data generation process. Existing sampling-phase watermark methods are mainly for invertible diffusion models (e.g. DDIM (Song et al., 2020)). Since the sampling process of diffusion models maps a Gaussian noise to the data distribution, (Wen et al., 2023; Yang et al., 2024) embeds detectable patterns into the latent space, sampling pseudorandom gaussian noise for watermarked generation. While previous sampling-phase watermarks can produce invisible watermarks for individual samples, they are limited to invertible diffusion models. Furthermore, as the DDIM inversion is not exact, the detectability of watermarked samples is compromised due to error accumulation during the inversion process (Hu et al., 2025).

Our approach. In this work, we present RINTAW, a robust and invisible watermarking method for tabular generative models. Our method reconsiders where to embed the watermark in the generation process: since embedding the watermark signal in the latent space ($t = 0$ for diffusion models) will inevitably suffer from the inversion error, also inspired by the tournament sampling mechanism proposed in (Dathathri et al., 2024), we propose to embed the watermark at the end of the sampling stage ($t = 1$). The core idea of RINTAW is to embed watermark information by sampling from the current data distribution using pseudorandomly sampled keys. Specifically, we design a tournament sampling mechanism: After selecting a key, we sample multiple rows and score each row using a pseudorandom function. The row achieving the highest score under the function corresponding to that key is selected as the watermark sample. The keys are generated by applying a hash function to combinations of several special columns from the table. To ensure the generated keys have good randomness and uniqueness properties, we develop an adaptive column selection strategy that dynamically chooses columns for key generation based on the ranking of values within their distributions. Additionally, we introduce a repeated key masking mechanism to prevent reusing the same keys. Through these mechanisms, RINTAW can effectively embed watermarks while preserving the data distribution without distortion. Notably, RINTAW is model-agnostic and applicable to any table generation model that enables repeated sampling.

We highlight the main contributions of this paper below:

- We propose robust invisible tabular watermarking (RINTAW), to embed robust and distortion-free watermarks into generative tabular data.
- We conduct extensive experiments to demonstrate the effectiveness of our method. Our experiments show that RINTAW achieves the overall best performance in generation quality, detectability, and robustness (Figure 1).

2 METHOD

In this section, we provide a detailed introduction to the proposed watermark algorithm named RINTAW. First, in Section 2.1, we introduce the overall process of watermark generation. Then, in Section 2.2, we present details of the column selection strategy. Finally, in Section 2.3, we present a statistical detection method for RINTAW.

2.1 WATERMARK GENERATION

A watermarking scheme for generative models typically contains two components: 1) a random seed generator and 2) a sampling method. To generate a watermarked sample, the random seed generator first generates a random seed, then the sampling method takes the random seed and the generative model as input to generate a watermarked sample. Previous sampling methods used for watermarking usually rely on the closed-form expression of the data distribution (density function for continuous

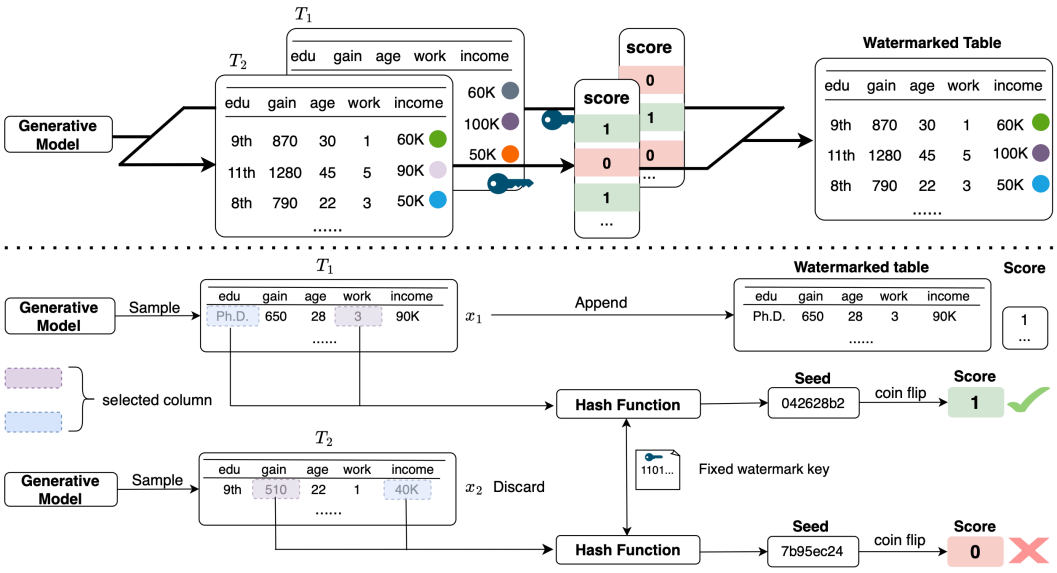

Figure 2: **Top:** An overview of the watermark generation process. Two $i.i.d$ sampled tables T_1, T_2 are generated from the generative model p. Each row gets a score based on a pseudorandom function. The row with the higher score is kept and appended to the watermarked table, while the row with the lower score is discarded. **Bottom:** detail of the pseudorandom score function: for each row, a subset of columns are selected by an adaptive column selection strategy (Algorithm 2), then a hash function is applied to the selected columns and the watermark key to generate a score.

data and probability mass function for discrete data) (Kirchenbauer et al., 2023; Kuditipudi et al., 2023; Yang et al., 2024). However, for diffusion models, the density function is expensive to compute or estimate (Song et al., 2021). To tackle this problem, previous works (Yang et al., 2024; Wen et al., 2023) typically rely on an approximate procedure to inverse the diffusion model and then embed the watermark on the Gaussian noise, for which we know the closed-form expression of the data distribution.

Our work reconsiders how to embed the watermark into the sampling phase of diffusion models: previous works embed the watermark into the initial Gaussian noise (i.e., $t = 0$), which necessitates a cumbersome recursive inversion of the diffusion model to detect the watermark. Inspired by the tournament sampling method used for LLM watermark (Dathathri et al., 2024), we propose to embed the watermark on a fully denoised sample (i.e., $t = T$). See Figure 2 (Top). By doing so, we can directly detect the watermark on the original sample, avoiding the iterative inversion of the diffusion model. The key idea of Tournament sampling is to sample duplicated $i.i.d$ samples from the data distribution, then use a tournament-like process to choose a sample that scores highly with respect to some pseudo-random watermarking functions (score function). This procedure embeds watermark by biasing the sampling process towards samples that achieve high scores.

To illustrate the main idea of this method, we first describe a simple version of the method in Algorithm 1, where we use a fixed column of the table to seed the score function.

Algorithm 1 Tabular watermark with fixed seed columns

1: **Input:** watermark key k, table generative model $p(\mathbf{x})$.
2: Draw two $i.i.d$ samples $\mathbf{x}_1, \mathbf{x}_2$ from $p(\mathbf{x})$.
3: Compute a hash of the the watermark key k and the first element of $\mathbf{x}_1, \mathbf{x}_2$: $r_1 = \text{hash}(k, \mathbf{x}_1[0])$, $r_2 = \text{hash}(k, \mathbf{x}_2[0])$. ▷ Fixed seed columns.
4: Use r_1, r_2 as seeds, randomly sample $s_1, s_2 \in \{0, 1\}$.
5: Pick $\mathbf{x}_i$ with the higher score s_i, break ties randomly.
6: **return** $\mathbf{x}_i$.

Computation time. To sample a watermarked table with N rows from a generative model p, we can run Algorithm 1 N times. Since the algorithm takes the same watermark key and generative model as input, in practice, we can first generate two unwatermarked tables with N rows from p, then randomly pair rows from these tables, and finally run Algorithm 1 on each pair in batch. This approach results in a generation time approximately twice that of producing an unwatermarked table. Compared to (Wen et al., 2023; Yang et al., 2024), which embeds watermark in the initial Gaussian noise, our method takes twice the time in the watermark generation phase but saves the time in the watermark detection phase (see details in Section 2.3), effectively balancing the overall computational cost.

Weakness of using fixed seed columns. The above approach is a simple way to embed a watermark for tabular data, but it has several potential issues:

1) **Venerability to robust attacks.**. Note that in Algorithm 1, every sample is chosen based on the same column. If an attacker identifies the column used to seed the score function, they can completely remove the watermark by simply deleting that column from the table.
2) **Low watermark strength.** If the column used to seed the score function has a small number of possible values, the watermark strength will be low. Consider an extreme scenario where the column used is constant, the score function will always return the same value. Consequently, the watermark strength will be zero.
3) **High distribution distortion.** When the column used to seed the score function has a small set of possible values, due to repeated usage of the same value for seeding, the strategy of picking the sample that scores higher may significantly bias the data distribution.

Issues (1) and (2) are evident. Regarding issue (3), It can be shown (Dathathri et al., 2024) that the watermark scheme defined in Algorithm 1 is distribution-preserving in expectation if the watermark key is uniformly distributed over $\mathbb{Z}_n$.

Definition 2.1 (Watermark distribution)**.** Let $\tilde{\mathbf{x}}$ be the output of Algorithm 1 with data distribution p and watermark key k as input. We call the probability distribution of $\tilde{\mathbf{x}}$ as *watermark distribution*, denoted by $p_{wm}(\tilde{\mathbf{x}} \mid p, k)$.

Theorem 2.2 (Algorithm 1 is single sample distribution-preserving)**.** *Suppose a sample $\tilde{\mathbf{x}}$ is generated from Algorithm 1 with data distribution p and watermark key k as input. Then it holds that marginalizing over the watermark key k, the watermark distribution is the same as the original data distribution:*

$$\mathbb{E}_{k\sim\text{Unif}(\mathbb{Z}_n)}\left[p_{wm}(\mathbf{x} \mid p, k)\right] = p(\mathbf{x})$$

Theorem 2.2 shows that the watermark scheme defined in Algorithm 1 is single-sample distortion-free. In reality, we call Theorem 2.2 repeatedly with the same watermark key k to generate multiple samples to form a table. In the following, we show that we can extend the above result to K-sample distribution-preserving.

Corollary 2.3 (Algorithm 1 is K-sample distribution-preserving)**.** *Suppose a sequence of K samples $(\tilde{\mathbf{x}}_1, \ldots, \tilde{\mathbf{x}}_K)$ are generated consecutively from Algorithm 1 with the same watermark key k and data distribution $p(\mathbf{x})$. Then it holds that, marginalizing over the watermark key k, the joint distribution of $(\tilde{\mathbf{x}}_1, \ldots, \tilde{\mathbf{x}}_K)$ is the same as the joint distribution of K $i.i.d$ samples from the original data distribution:*

$$\mathbb{E}_{k\sim\text{Unif}(\mathbb{Z}_n)}\left[p_{wm}(\tilde{\mathbf{x}}_1, \ldots, \tilde{\mathbf{x}}_K \mid p, k)\right] = \prod_{i=1}^{K} p(\tilde{\mathbf{x}}_i)$$

2.2 Column selection strategy

Based on the above analysis, preserving the original data distribution requires using a uniformly random distributed watermark key k as input to the random seed generator. While (Kuditipudi et al., 2023) suggests maintaining a list of pseudo-random watermark keys to generate watermarked samples, this approach faces challenges with large-scale data. When generating numerous samples (e.g., tables with $N > 10^5$ rows), the pseudo-random key list becomes exhaustive, approaching complete coverage of the possible key space. This comprehensive coverage significantly increases the false positive rate (FPR) during watermark detection.

Algorithm 2 Column Selection

1: **Input:** Unwatermarked table T, a sample $\mathbf{x}$ from T.
2: For each column index j, sort $T[:, j]$ in ascending order, denote the index of $\mathbf{x}[j]$ within the sorted list as $r[j]$.
3: Sort the list r in ascending order, denote the column indices that attain the largest, median, and smallest of list r as $j_{\texttt{min}}, j_{\texttt{med}}, j_{\texttt{max}}$, respectively.
4: **return** column indices $\{j_{\texttt{min}}, j_{\texttt{med}}, j_{\texttt{max}}\}$.

In this paper, we use an alternative strategy: we use a fixed watermark key k, but introduce the needed uniform randomness by selecting a subset of columns of $\mathbf{x}$ as the second input to the random seed generator. This strategy is conceptually similar to some LLM watermarking methods (Kirchenbauer et al., 2023; Dathathri et al., 2024; Zhao et al., 2023), which use the previous token(s) as the second input to the random seed generator.

However, determining the proper columns for seeding the score function presents a significant challenge. As discussed in Section 2.1, tabular data's heterogeneous nature makes simple fixed-column selection strategies sub-optimal. In the following, we present two techniques for better column selection. The complete watermark generation algorithm is presented in Algorithm 3.

Adaptive column selection. Given a non-watermarked table T with N rows and M columns, suppose we want to determine which columns of a sample $\mathbf{x}$ to use for seeding the score function. We propose the following procedure:

1) For each column $j \in \{1, 2, ..., M\}$, denote the empirical cumulative distribution function (CDF) of the column j of T as $\tilde{F}_{T[:,j]}(\cdot)$, compute its evaluation at $\mathbf{x}_j$: $r_j = \tilde{F}_{T[:,j]}(\mathbf{x}_j)$.
2) Sort the columns based on the rank $\{r_j\}_{j=1}^{M}$,
3) Select the columns that attain the largest, median, and smallest r_j.

Different from the simple strategy in Algorithm 1, the above procedure is adaptive to each individual sample of $\mathbf{x}$. This procedure achieves adaptivity by leveraging the deviation from the data distribution of each individual sample. In our experiments, we choose three quantiles $0, 0.5$, and 1 of the sorted rank to select the columns. Choosing more fine-grained quantiles will result in higher watermark strength, but potentially harm watermark robustness.

Masking mechanism. Suppose the space of the watermark key is $\mathbb{Z}_n$, our goal is to use selected columns of $\mathbf{x}$ as a proxy of the uniformly distributed watermark key over $\mathbb{Z}_n$. In reality, so long as the value at the selected columns is not repeatedly used for watermarking, we consider this as a uniformly distributed watermark key. For this purpose, we propose a masking mechanism to prevent repeated usage of the same value for watermarking. It works by maintaining a set of values that have been used for watermarking and skipping watermarking the current sample if the value has been used. To control the effect of the masking mechanism, we deploy a parameter $\alpha \in [0, 1]$ to control the probability of skipping watermarking, when repetition is found. This parameter can also be used to control the watermark strength.

2.3 Watermark statistical detection

In this section, we formalize the statistical detection process for identifying watermarks in a table. Consider a table T containing N samples (rows) $\mathbf{x}_1, \ldots, \mathbf{x}_N$. During watermarking, each row is assigned a binary score of 1 or 0 based on a pseudo-random function, and the row that scores higher is kept. Therefore, for a watermarked table, the total count of rows with a score 1, denoted by $|W|$, is expected to be significantly higher than random chance. To statistically validate this, we formulate watermark detection as a hypothesis testing problem, following (Kirchenbauer et al., 2023):

$$H_0 : \text{The table is generated without watermarking.}$$
$$\text{vs. } H_1 : \text{The table is generated with watermarking.}$$

Algorithm 3 Tabular watermark generation

Input: watermark key k, unwatermarked table T.
Randomly pairing rows of T into $N/2$ pairs: $\{p_i\}_{i=1}^{N/2}$.
Initialize an empty set $\mathcal{R}$, and a list T_{wm} to store the watermarked table.
for $i \leftarrow 1$ to $N/2$ **do**
  $(\mathbf{x}_1, \mathbf{x}_2) \leftarrow p_i$
  $\mathcal{I}_1 \leftarrow \textsc{ColSelect}(T, \mathbf{x}_1)$. ▷ Algorithm 2.
  $\mathcal{I}_2 \leftarrow \textsc{ColSelect}(T, \mathbf{x}_2)$. ▷ Algorithm 2.
  Compute a hash of the watermark key k and the selected columns: $r_1 = \text{hash}(k, \mathbf{x}_1[\mathcal{I}_1])$ $r_2 = \text{hash}(k, \mathbf{x}_2[\mathcal{I}_2])$
  if $r_1 \in \mathcal{R}$ and $r_2 \in \mathcal{R}$ **then**
    Append $\mathbf{x}_1$ or $\mathbf{x}_2$ to T_{wm} decided by coin flip.
    Continue to the next iteration. ▷ Skip watermarking.
  else
    Add r_1, r_2 to $\mathcal{R}$.
  end if
  Use r_1, r_2 as seeds, randomly sample $s_1, s_2 \in \{0, 1\}$, respectively.
  Pick the sample with the higher score and break ties randomly.
  Append the sample to T_{wm}.
end for
return T_{wm}.

Under the null hypothesis, $|W|$ follows a binomial distribution with mean $\mu = N/2$ and variance $\sigma^2 = N/4$. The standardized z-statistic is computed as:

$$z = \frac{|W| - \mu}{\sigma} = \frac{|W| - N/2}{\sqrt{N/4}}$$

We perform a *one-tailed* test (upper tail) since the alternative hypothesis predicts $|W| > N/2$. The z-statistic is compared against a critical value z_α corresponding to a desired significance level α (e.g. $\alpha = 0.05$ yields $z_\alpha = 1.645$). If $z > z_\alpha$, we reject the null hypothesis and conclude that the table is watermarked.

3 Experiments

We validate the performance of our watermarking method through extensive experiments. In particular, we investigate the following questions:

- Can RINTAW have good statistical power (detectability), with minimal distortion to the generated data (invisibility)? (Table 2)
- How robust is the watermark against various attacks? (Table 3)
- How does RINTAW perform under different masking ratios and the number of selected columns? (Figure 3)

3.1 Setup

Datasets. We select six real-world tabular datasets containing both numerical and categorical attributes: **Adult**, **Default**, **Shoppers**, **Magic**, **Beijing** and **News**. The statistics of the datasets are summarized in Table 4 in Appendix A.2.

Baselines. We use recent tabular watermarking algorithms as baselines, including TabularMark (Zheng et al., 2024) and WGTD (He et al., 2024). We re-implement both methods using the authors' specifications, as the official code was unavailable. The implementation details are provided in Appendix A.1.

Table 2: Quality and detectability, RINTAW: without masking, RINTAW$_{df}$: with masking. The best results are highlighted in bold.

Dataset	Method	Quality				Detectability			
		Num. Training Rows				500		1000	
		Marg.↑	Corr.↑	C2ST↑	MLE Gap↓	p-value/z-stat	AUC/T@1%F	p-value/z-stat	AUC/T@1%F
Adult	w/o WM	0.994$_{\pm 0.001}$	0.984$_{\pm 0.001}$	0.996$_{\pm 0.003}$	0.017	-	-	-	-
	TabularMark	**0.992**$_{\pm 0.001}$	0.939$_{\pm 0.000}$	**0.995**$_{\pm 0.003}$	**0.017**	0.000/7.071	1.000/1.000	0.000/10.000	1.000/1.000
	WGTD	0.905$_{\pm 0.089}$	0.970$_{\pm 0.014}$	0.975$_{\pm 0.021}$	0.019	-	1.000/1.000	-	1.000/1.000
	RINTAW	0.979$_{\pm 0.001}$	0.963$_{\pm 0.001}$	0.883$_{\pm 0.005}$	**0.017**	0.000/7.830	1.000/1.000	0.000/8.610	1.000/1.000
	RINTAW$_{df}$	0.985$_{\pm 0.001}$	**0.973**$_{\pm 0.001}$	0.940$_{\pm 0.003}$	**0.017**	0.000/3.486	1.000/1.000	0.003/2.723	1.000/1.000
Beijing	w/o WM	0.977$_{\pm 0.001}$	0.958$_{\pm 0.003}$	0.934$_{\pm 0.003}$	0.199	-	-	-	-
	TabularMark	0.970$_{\pm 0.001}$	0.799$_{\pm 0.002}$	**0.934**$_{\pm 0.003}$	**0.204**	0.000/7.071	1.000/1.000	0.000/10.000	1.000/1.000
	WGTD	0.971$_{\pm 0.004}$	0.958$_{\pm 0.003}$	0.664$_{\pm 0.411}$	0.207	-	1.000/1.000	-	1.000/1.000
	RINTAW	0.972$_{\pm 0.001}$	0.955$_{\pm 0.002}$	0.926$_{\pm 0.002}$	0.209	0.000/11.886	1.000/1.000	0.000/16.961	1.000/1.000
	RINTAW$_{df}$	**0.977**$_{\pm 0.001}$	**0.959**$_{\pm 0.002}$	**0.934**$_{\pm 0.005}$	0.210	0.000/11.794	1.000/1.000	0.000/16.886	1.000/1.000
Default	w/o WM	0.990$_{\pm 0.001}$	0.934$_{\pm 0.018}$	0.979$_{\pm 0.006}$	0.000	-	-	-	-
	TabularMark	**0.990**$_{\pm 0.001}$	0.934$_{\pm 0.018}$	**0.982**$_{\pm 0.003}$	**0.000**	0.000/7.071	1.000/1.000	0.000/10.000	1.000/1.000
	WGTD	**0.990**$_{\pm 0.001}$	**0.944**$_{\pm 0.019}$	0.713$_{\pm 0.134}$	0.001	-	1.000/1.000	-	1.000/1.000
	RINTAW	0.983$_{\pm 0.001}$	0.925$_{\pm 0.000}$	0.963$_{\pm 0.003}$	0.002	0.001/3.277	1.000/1.000	0.000/13.786	1.000/1.000
	RINTAW$_{df}$	0.986$_{\pm 0.001}$	0.927$_{\pm 0.018}$	0.963$_{\pm 0.002}$	0.001	0.000/9.243	1.000/1.000	0.000/13.036	1.000/1.000
Magic	w/o WM	0.990$_{\pm 0.001}$	0.980$_{\pm 0.007}$	0.998$_{\pm 0.002}$	0.008	-	-	-	-
	TabularMark	0.989$_{\pm 0.001}$	0.971$_{\pm 0.007}$	0.998$_{\pm 0.002}$	**0.009**	0.000/7.071	1.000/1.000	0.000/10.000	1.000/1.000
	WGTD	0.950$_{\pm 0.010}$	0.955$_{\pm 0.006}$	0.932$_{\pm 0.069}$	0.014	-	1.000/1.000	-	1.000/1.000
	RINTAW	**0.991**$_{\pm 0.002}$	**0.982**$_{\pm 0.008}$	**0.999**$_{\pm 0.001}$	0.010	0.000/11.179	1.000/1.000	0.000/15.862	1.000/1.000
	RINTAW$_{df}$	0.990$_{\pm 0.002}$	0.981$_{\pm 0.007}$	**0.999**$_{\pm 0.002}$	**0.009**	0.000/11.122	1.000/1.000	0.000/15.817	1.000/1.000
News	w/o WM	0.960$_{\pm 0.000}$	0.973$_{\pm 0.003}$	0.811$_{\pm 0.004}$	0.024	-	-	-	-
	TabularMark	**0.960**$_{\pm 0.000}$	0.973$_{\pm 0.003}$	0.811$_{\pm 0.004}$	0.023	0.000/7.071	1.000/1.000	0.000/10.000	1.000/1.000
	WGTD	0.896$_{\pm 0.003}$	0.966$_{\pm 0.002}$	0.216$_{\pm 0.228}$	**0.009**	-	1.000/1.000	-	1.000/1.000
	RINTAW	0.959$_{\pm 0.001}$	0.902$_{\pm 0.001}$	**0.812**$_{\pm 0.007}$	0.033	0.000/10.235	1.000/1.000	0.000/19.966	1.000/1.000
	RINTAW$_{df}$	0.959$_{\pm 0.000}$	**0.974**$_{\pm 0.003}$	**0.812**$_{\pm 0.009}$	0.027	0.000/13.912	1.000/1.000	0.000/19.585	1.000/1.000
Shoppers	w/o WM	0.985$_{\pm 0.001}$	0.974$_{\pm 0.004}$	0.974$_{\pm 0.007}$	0.017	-	-	-	-
	TabularMark	**0.985**$_{\pm 0.001}$	0.923$_{\pm 0.004}$	**0.973**$_{\pm 0.006}$	0.016	0.000/7.071	1.000/1.000	0.000/10.000	1.000/1.000
	WGTD	0.983$_{\pm 0.002}$	0.955$_{\pm 0.006}$	0.514$_{\pm 0.425}$	0.025	-	1.000/1.000	-	1.000/1.000
	RINTAW	0.982$_{\pm 0.001}$	**0.974**$_{\pm 0.002}$	0.950$_{\pm 0.007}$	**0.015**	0.000/11.479	1.000/1.000	0.000/16.226	1.000/1.000
	RINTAW$_{df}$	0.982$_{\pm 0.001}$	**0.974**$_{\pm 0.002}$	0.952$_{\pm 0.008}$	0.018	0.000/10.789	1.000/1.000	0.000/15.149	1.000/1.000

Evaluation metrics. (a) To evaluate the detectability of the watermark, we report the area under the curve (AUC) of the receiver operating characteristic (ROC) curve, and the True Positive Rate when the False Positive Rate is at 1%, denoted as *TPR@1%FPR*. (b) To evaluate the distortion of the watermarked data, we follow standard fedelity and utility metrics used in tabular data generation (Zhang et al., 2024b; Kotelnikov et al., 2023): we report Marginal distribution (Marg.), Pair-wise column correlation (Corr.), Classifier Two Sample Test (C2ST), and Machine Learning Efficiency (MLE). For MLE, we report the gap between the downstream task performance of the generated data and the real test set (MLE Gap). We refer the readers to (Zhang et al., 2024b) for a more detailed definition of each evaluation metric.

Implementation details. We use TabSyn (Zhang et al., 2024b) as the generative model for all experiments. For most datasets, the number of selected columns is set to 3, except for the Default dataset, where it is set to 7. The masking ratio is configured to range from 0 to 0.5, with the specific masking ratio for each dataset automatically inferred based on the unique number of values in the selected columns. Experiments for generation quality evaluation are repeated 10 times and mean and standard deviation are reported.

3.2 DETECTABILITY VS TABLE QUALITY

We address the first question: whether the watermarking method achieves high statistical power (detectability) and remains invisible (causing minimal distortion) in the generated data. As shown in Table 2, while all baseline methods demonstrate high detectability, WGTD significantly falls short in generation quality. This result is expected, as WGTD modifies the decimal part of all numerical cells in the table, leading to substantial distortion. In contrast, TabularMark introduces less distortion by watermarking only a small subset of cells. However, this selective approach makes it more vulnerable to various attacks. Our proposed method, on the other hand, achieves high detectability with minimal distortion to the generated data. Previous tabular watermarking methods often cause significant degradation in data quality. For post-processing watermarking methods, the distortion arises from modifying the decimal parts of the generated data during the post-processing steps.

Table 3: Robustness to different attacks, AUC/T@1%F

Dataset	Method	Attack Type					
		Shuffle	Row Del.	Col Del.	Cell Del.	Alteration	Noise
Adult	TabularMark	0.374/0.002	0.374/0.000	-	0.996/0.904	0.366/0.000	0.411/0.002
	WGTD	1.000/1.000	1.000/1.000	**0.825/0.198**	1.000/1.000	0.653/0.008	0.469/0.004
	RINTAW	1.000/1.000	1.000/1.000	0.741/0.082	1.000/1.000	**1.000/0.994**	0.396/0.002
	RINTAW$_{df}$	1.000/1.000	1.000/1.000	0.717/0.022	0.991/0.822	0.950/0.580	**0.582/0.018**
Beijing	TabularMark	0.285/0.000	0.338/0.000	-	1.000/0.990	1.000/1.000	1.000/1.000
	WGTD	1.000/1.000	1.000/1.000	**0.885/0.232**	1.000/1.000	0.653/0.020	0.774/0.020
	RINTAW	1.000/1.000	1.000/1.000	0.754/0.550	1.000/1.000	0.768/0.054	0.456/0.014
	RINTAW$_{df}$	1.000/1.000	1.000/1.000	0.708/0.492	1.000/1.000	0.772/0.090	0.479/0.002
Default	TabularMark	0.464/0.002	0.505/0.002	-	0.981/0.750	0.500/0.000	0.412/0.012
	WGTD	1.000/1.000	1.000/1.000	1.000/1.000	1.000/1.000	0.669/0.005	**0.733**/0.008
	RINTAW	1.000/1.000	1.000/1.000	0.921/0.714	1.000/1.000	**0.963/0.509**	0.628/**0.023**
	RINTAW$_{df}$	1.000/1.000	1.000/1.000	0.905/0.683	1.000/1.000	0.955/0.360	0.589/0.017
Magic	TabularMark	0.458/0.006	0.458/0.000	-	0.988/0.736	0.323/0.000	0.415/0.000
	WGTD	1.000/1.000	1.000/1.000	1.000/1.000	1.000/1.000	**0.982/0.834**	0.484/0.000
	RINTAW	1.000/1.000	1.000/1.000	0.892/0.540	0.999/0.988	0.516/0.020	0.499/**0.022**
	RINTAW$_{df}$	1.000/1.000	1.000/1.000	0.883/0.504	1.000/0.998	0.496/0.006	**0.539**/0.004
News	TabularMark	0.482/0.006	0.465/0.008	-	0.977/0.756	0.234/0.000	0.400/0.000
	WGTD	1.000/1.000	1.000/1.000	1.000/1.000	1.000/1.000	1.000/1.000	**0.776/0.026**
	RINTAW	1.000/1.000	1.000/1.000	0.819/0.614	0.995/0.888	0.476/0.014	0.505/0.004
	RINTAW$_{df}$	1.000/1.000	1.000/1.000	0.795/0.550	0.993/0.880	0.498/0.004	0.523/0.010
Shoppers	TabularMark	0.310/0.000	0.326/0.002	-	1.000/0.990	0.002/0.000	0.424/0.002
	WGTD	1.000/1.000	1.000/1.000	**0.938/0.102**	1.000/1.000	0.969/**0.748**	**0.764/0.026**
	RINTAW	1.000/1.000	1.000/1.000	0.812/0.482	1.000/1.000	**0.980**/0.722	0.510/0.010
	RINTAW$_{df}$	1.000/1.000	1.000/1.000	0.813/0.458	1.000/0.998	0.952/0.544	0.547/0.010

3.3 ROBUSTNESS AGAINST ATTACKS

We next benchmark the robustness of the tabular watermarking method against six prevalent attacks tailored to tabular data: **row shuffling**, **column deletion**, **row deletion**, **cell deletion**, **Gaussian noise addition**, and **value alteration**. The intensity levels are defined as follows: For row deletion, we remove 20% of rows. For cell deletion, we remove 20% of randomly selected cells. For column deletion, 3 columns are deleted. For Gaussian noise, we inject noise into numeric columns with zero mean and identity variance. For value alteration, numeric column values are multiplied by a factor sampled uniformly from $(0.9, 1.1)$. In Table 3, we report the AUC and TPR@1%FPR of the watermarking methods under these different attacks. Our method demonstrates high detectability under row shuffling, row deletion, and cell deletion attacks, but is relatively more vulnerable to column deletion attacks. In contrast, TabularMark is the least robust method, as it watermarks only a small subset of cells at fixed positions, making it easier to target. WGTD generally shows strong robustness to most attacks, though it is more susceptible to value alteration and Gaussian noise injection. Overall, our method strikes a balance between detectability and robustness, outperforming other methods in key scenarios.

3.4 ABLATION STUDY

We perform ablation studies to analyze the impact of different components of the proposed method. Specifically, we examine the effects of the masking ratio and the number of selected columns, as shown in Figure 3. We observe that the statistical power of the watermarking method decreases as the masking ratio increases. This is expected, as a higher masking ratio results in fewer samples being watermarked. For the number of selected columns, since the number of unique values increases as more columns are selected, the statistical power of the watermarking method is intuitively expected to improve. However, selecting more columns also makes the watermark more vulnerable to column deletion attacks. Interestingly, we find that the statistical power generally saturates when the number of selected columns exceeds 3, except for the Adult dataset, where the z-statistic continues to increase with more selected columns. This suggests that selecting 3 columns strikes a good balance between statistical power and robustness against attacks.

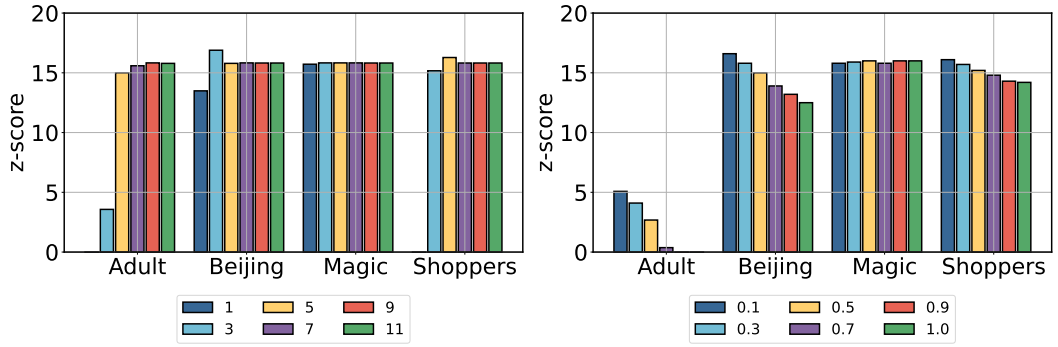

Figure 3: **Left:** z-statistic under different number of selected columns. The statistic power almost achieves the highest at 3 columns, except the Adult dataset. **Right:** z-statistic under different masking ratios. The statistic power decreases as the masking ratio increases. Negative values are clipped to 0.

4 Related Work

Existing table watermarking approaches can be broadly categorized into two types: post-processing watermarks and generation-time watermarks. Post-processing methods, such as the work by (Ngo et al., 2024), embed watermarks by modifying data after it has been generated. While these methods have minimal impact on data quality and downstream tasks, they can still introduce distortions to the data distribution. Similarly, TabularMark (Zheng et al., 2024) embeds watermarks by modifying values in select key cells. Although this approach reduces the impact on machine learning tasks, its ability to preserve global data distributions remains limited. Another method, proposed by (He et al., 2024), embeds watermarks through resampling within "green-listed" intervals. While this technique theoretically maintains low data distortion, it still relies on post-processing modifications, which can introduce systematic bias into the data distribution. In contrast, generation-time watermarking methods are more common in image watermarking, as demonstrated by approaches such as (Wen et al., 2023; Yang et al., 2024). Recently, a concurrent method (Zhu et al., 2025) introduced a generation-time watermarking technique for tabular data. However, these methods are often limited to specific diffusion models and depend on DDIM reversibility (Song et al., 2020), which may reduce the effectiveness of watermark detection. These limitations highlight that existing methods still have room for improvement in maintaining distortion-free data distributions, ensuring model independence, and enhancing robustness.

5 Conclusion

This paper presents RINTAW, a novel watermarking algorithm that integrates directly into the sampling process of generative models for tabular data. Unlike existing post-processing approaches, RINTAW achieves strong statistical detection power while maintaining minimal distribution distortion. Through extensive experiments, we demonstrate the method's robustness against common attacks including row shuffling/deletion, column deletion, cell deletion, alteration, and Gaussian noise addition. With its general applicability across different generative frameworks, RINTAW represents a promising advance in protecting synthetic tabular data integrity.

REFERENCES

Francesco Cartella, Orlando Anunciacao, Yuki Funabiki, Daisuke Yamaguchi, Toru Akishita, and Olivier Elshocht. Adversarial attacks for tabular data: Application to fraud detection and imbalanced data. arXiv preprint arXiv:2101.08030, 2021. 1

Sumanth Dathathri, Abigail See, Sumedh Ghaisas, Po-Sen Huang, Rob McAdam, Johannes Welbl, Vandana Bachani, Alex Kaskasoli, Robert Stanforth, Tatiana Matejovicova, Jamie Hayes, Nidhi Vyas, Majd Al Merey, Jonah Brown-Cohen, Rudy Bunel, Borja Balle, Taylan Cemgil, Zahra Ahmed, Kitty Stacpoole, Ilia Shumailov, Ciprian Baetu, Sven Gowal, Demis Hassabis, and Pushmeet Kohli. Scalable watermarking for identifying large language model outputs. Nature, 634(8035):818–823, Oct 2024. ISSN 1476-4687. doi: 10.1038/s41586-024-08025-4. URL https://doi.org/10.1038/s41586-024-08025-4. 2, 3, 4, 5

Adrià Gascón, Phillipp Schoppmann, Borja Balle, Mariana Raykova, Jack Doerner, Samee Zahur, and David Evans. Privacy-preserving distributed linear regression on high-dimensional data. Cryptology ePrint Archive, 2016. 1

Manbir Gulati and Paul Roysdon. Tabmt: Generating tabular data with masked transformers. Advances in Neural Information Processing Systems, 36, 2024. 1

Hengzhi He, Peiyu Yu, Junpeng Ren, Ying Nian Wu, and Guang Cheng. Watermarking generative tabular data. arXiv preprint arXiv:2405.14018, 2024. 1, 2, 6, 9

Runyi Hu, Jie Zhang, Yiming Li, Jiwei Li, Qing Guo, Han Qiu, and Tianwei Zhang. Videoshield: Regulating diffusion-based video generation models via watermarking. arXiv preprint arXiv:2501.14195, 2025. 2

John Kirchenbauer, Jonas Geiping, Yuxin Wen, Jonathan Katz, Ian Miers, and Tom Goldstein. A watermark for large language models. In International Conference on Machine Learning, pp. 17061–17084. PMLR, 2023. 3, 5

Akim Kotelnikov, Dmitry Baranchuk, Ivan Rubachev, and Artem Babenko. Tabddpm: Modelling tabular data with diffusion models. In International Conference on Machine Learning, pp. 17564–17579. PMLR, 2023. 1, 7

Rohith Kuditipudi, John Thickstun, Tatsunori Hashimoto, and Percy Liang. Robust distortion-free watermarks for language models. arXiv preprint arXiv:2307.15593, 2023. 3, 4

Wei-Chao Lin and Chih-Fong Tsai. Missing value imputation: a review and analysis of the literature (2006–2017). Artificial Intelligence Review, 53:1487–1509, 2020. 1

Dung Daniel Ngo, Daniel Scott, Saheed Obitayo, Vamsi K Potluru, and Manuela Veloso. Adaptive and robust watermark for generative tabular data. arXiv preprint arXiv:2409.14700, 2024. 9

Inkit Padhi, Yair Schiff, Igor Melnyk, Mattia Rigotti, Youssef Mroueh, Pierre Dognin, Jerret Ross, Ravi Nair, and Erik Altman. Tabular transformers for modeling multivariate time series. In ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 3565–3569. IEEE, 2021. 1

Aivin V Solatorio and Olivier Dupriez. Realtabformer: Generating realistic relational and tabular data using transformers. arXiv preprint arXiv:2302.02041, 2023. 1

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. arXiv preprint arXiv:2010.02502, 2020. 2, 9

Yang Song, Conor Durkan, Iain Murray, and Stefano Ermon. Maximum likelihood training of score-based diffusion models. Advances in neural information processing systems, 34:1415–1428, 2021. 3

Yuxin Wen, John Kirchenbauer, Jonas Geiping, and Tom Goldstein. Tree-ring watermarks: Fingerprints for diffusion images that are invisible and robust. arXiv preprint arXiv:2305.20030, 2023. 2, 3, 4, 9

Zijin Yang, Kai Zeng, Kejiang Chen, Han Fang, Weiming Zhang, and Nenghai Yu. Gaussian shading: Provable performance-lossless image watermarking for diffusion models. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 12162–12171, 2024. 2, 3, 4, 9

Hengrui Zhang, Liancheng Fang, Qitian Wu, and Philip S Yu. Diffusion-nested auto-regressive synthesis of heterogeneous tabular data. arXiv preprint arXiv:2410.21523, 2024a. 1

Hengrui Zhang, Jiani Zhang, Balasubramaniam Srinivasan, Zhengyuan Shen, Xiao Qin, Christos Faloutsos, Huzefa Rangwala, and George Karypis. Mixed-type tabular data synthesis with score-based diffusion in latent space. In The twelfth International Conference on Learning Representations, 2024b. 1, 7

Xuandong Zhao, Prabhanjan Ananth, Lei Li, and Yu-Xiang Wang. Provable robust watermarking for ai-generated text. arXiv preprint arXiv:2306.17439, 2023. 5

Yihao Zheng, Haocheng Xia, Junyuan Pang, Jinfei Liu, Kui Ren, Lingyang Chu, Yang Cao, and Li Xiong. Tabularmark: Watermarking tabular datasets for machine learning. In Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security, pp. 3570–3584, 2024. 1, 2, 6, 9

Chaoyi Zhu, Jiayi Tang, Jeroen M. Galjaard, Pin-Yu Chen, Robert Birke, Cornelis Bos, and Lydia Y. Chen. Tabwak: A watermark for tabular diffusion models. In The Thirteenth International Conference on Learning Representations, 2025. URL https://openreview.net/forum?id=71pur4y8gs. 9

A Appendix

A.1 Implementation details

• **WGTD.** WGTD embeds watermarks into tabular data through a three-step process: i) dividing the continuous interval $[0, 1]$ into $2m$ equal parts to form m pairs of consecutive intervals; ii) randomly selecting one interval from each pair to create a set of m "green list" intervals; and iii) replacing the fractional part of data points with a value sampled from the nearest "green list" interval if the original falls outside. Detection is achieved through a hypothesis-testing framework that leverages the statistical properties of data distribution to reliably identify watermarks. For reproducibility, we use the same hyperparameter settings as the original study, setting the number of "green list" intervals to $m = 5$.

• **TabularMark.** TabularMark employs a hypothesis-testing approach for watermarking by partitioning data noise into domains and introducing controlled perturbations to specific cells. Detection relies on a one-proportion z-test, which examines the deviation characteristics in the data to identify watermarks. This method involves four hyperparameters: the selected attribute A_i, the number of key cells n_w, the perturbation range controlled by p, and the number of unit domains k. Similar to the original experiment setup, we use the first numerical column as A_i, set $k = 500$, $p = 25$, and configure n_w as 10% of the total number of rows.

A.2 Datasets

The dataset used in this paper could be automatically downloaded using the script in the provided code. We use 6 tabular datasets from UCI Machine Learning Repository[1]: Adult[2], Default[3], Shoppers[4], Magic[5], Beijing[6], and News[7], which contains varies number of numerical and categorical features. The statistics of the datasets are presented in Table 4.

Table 4: Dataset statistics.

Dataset	# Rows	# Continuous	# Discrete	# Target	# Train	# Test	Task
Adult	32,561	6	8	1	22,792	16,281	Classification
Default	30,000	14	10	1	27,000	3,000	Classification
Shoppers	12,330	10	7	1	11,098	1,232	Classification
Magic	19,021	10	1	1	17,118	1,903	Classification
Beijing	43,824	7	5	1	39,441	4,383	Regression
News	39,644	46	2	1	35,679	3,965	Regression

In Table 4, **# Rows** refers to the total records in each dataset, while **# Continuous** and **# Discrete** denote the count of numerical and categorical features, respectively. The **# Target** column indicates whether the prediction task involves a continuous (regression) or discrete (classification) target variable. All datasets except Adult are partitioned into training and testing sets using a 9:1 ratio, with splits generated using a fixed random seed for reproducibility. The Adult dataset uses its predefined official testing set. For evaluating Machine Learning Efficiency (MLE), the training data is further subdivided into training and validation subsets with an 8:1 ratio, ensuring consistent evaluation protocols across experiments.

[1] https://archive.ics.uci.edu/datasets

[2] https://archive.ics.uci.edu/dataset/2/adult

[3] https://archive.ics.uci.edu/dataset/350/default+of+credit+card+clients

[4] https://archive.ics.uci.edu/dataset/468/online+shoppers+purchasing+intention+dataset

[5] https://archive.ics.uci.edu/dataset/159/magic+gamma+telescope

[6] https://archive.ics.uci.edu/dataset/381/beijing+pm2+5+data

[7] https://archive.ics.uci.edu/dataset/332/online+news+popularity