MUSE: Model-Agnostic Tabular Watermarking via Multi-Sample Selection

Liancheng Fang[1], **Aiwei Liu**[2]*, **Henry Peng Zou**[1], **Yankai Chen**[1], **Hengrui Zhang**[1], **Zhongfen Deng**[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

We introduce MUSE, a watermarking algorithm for tabular generative models. Previous approaches typically leverage DDIM invertibility to watermark tabular diffusion models, but tabular diffusion models exhibit significantly poorer invertibility compared to other modalities, compromising performance. Simultaneously, tabular diffusion models require substantially less computation than other modalities, enabling a multi-sample selection approach to tabular generative model watermarking. MUSE embeds watermarks by generating multiple candidate samples and selecting one based on a specialized scoring function, without relying on model invertibility. Our theoretical analysis establishes the relationship between watermark detectability, candidate count, and dataset size, allowing precise calibration of watermarking strength. Extensive experiments demonstrate that MUSE achieves state-of-the-art watermark detectability and robustness against various attacks while maintaining data quality, and remains compatible with any tabular generative model supporting repeated sampling, effectively addressing key challenges in tabular data watermarking. Specifically, it reduces the distortion rates on fidelity metrics by $\mathbf{81-89}\%$, while achieving $\mathbf{1.0}$ TPR@0.1%FPR detection rate. Implementation of MUSE can be found at https://github.com/fangliancheng/MUSE.

1 Introduction

The rapid development of tabular generative models [Kotelnikov et al., 2023, Gulati and Roysdon, 2024, Castellon et al., 2023, Zhang et al., 2024c, Shi et al., 2024, Zhang et al., 2024a, Fang et al., 2025] has significantly advanced synthetic data generation capabilities for structured information. These breakthroughs have enabled the creation of high-quality synthetic tables for applications in privacy preservation, data augmentation, and missing value imputation [Zhang et al., 2024b, Hernandez et al., 2022, Fonseca and Bacao, 2023, Assefa et al., 2020]. However, this advancement concurrently raises serious concerns about potential misuse, including data poisoning [Padhi et al., 2021] and financial fraud [Cartella et al., 2021]. To address these risks, watermarking techniques have emerged as

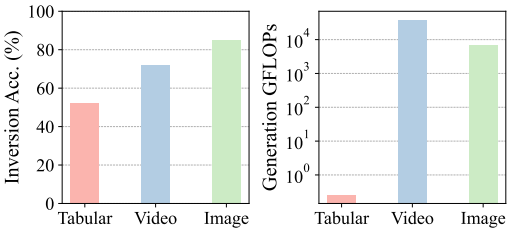

Figure 1: **Left**: Tabular diffusion models exhibit the lowest inversion accuracy (bit accuracy) when compared to video and image diffusion models. **Right**: Tabular diffusion models require much fewer generation GFLOPs than video and image diffusion models. Models used: TabSyn [Zhang et al., 2024c] (tabular), Stable Diffusion [Rombach et al., 2022, Blattmann et al., 2023] (image/video).

*Corresponding author.

Preprint. Under review.

a pivotal technique. By embedding imperceptible yet robust signatures into synthetic data, watermarking facilitates traceability, ownership verification, and misuse detection [Liu et al., 2024].

Earlier works on tabular data watermarking [Zheng et al., 2024, He et al., 2024] utilize ***edit-based watermarking***, embedding signals by modifying table values. However, this approach has a fundamental limitation with tabular data: direct value alterations, especially in columns with discrete or categorical data, **can easily corrupt information** or render entries invalid. For instance, such edits might introduce non-existent categories or generate values outside permissible numerical ranges, significantly compromising data integrity. Recently, ***generative watermarking*** has emerged as an alternative approach for tabular data, drawing from successful techniques in diffusion models for images and videos [Wu et al., 2025, Yang et al., 2024, Wen et al., 2023, Hu et al., 2025]. This approach exploits the reversibility of DDIM samplers by setting patterned initial Gaussian noise and measuring its correlation with noise recovered through the inverse process. TabWak [Zhu et al., 2025] applies this concept to tabular diffusion models. Unlike edit-based watermarking, this method maintains better generation quality since the watermark is embedded within noise patterns that closely resemble Gaussian distributions, minimizing impact on the generated content.

However, watermarking tabular diffusion models is **significantly more challenging** than for image and video diffusion models. This stems from the **substantially lower accuracy of DDIM inverse processes** in tabular diffusion models, as shown in Figure 1 (left). When using the same Gaussian shading algorithm [Yang et al., 2024], tabular modality exhibits the lowest reversibility accuracy. This challenge arises because tabular diffusion models incorporate multiple additional algorithmic components that are difficult to reverse, such as quantile normalization [Wikipedia contributors, 2025] and Variational Autoencoders (VAEs) [Kingma and Welling, 2013] used in TabSyn[Zhang et al., 2024c]. During watermark detection, the entire data processing pipeline must be inverted to recover the watermark signal, but this process accumulates errors as precisely reversing each step is often difficult or impossible. Key challenges in the inversion process include: (1) inverting quantile normalization is inherently problematic as this transformation is non-injective; (2) VAE decoder inversion relies on optimization methods without guarantees of perfect implementation. Due to limitations in tabular DDIM inversion accuracy, watermark detectability becomes highly dependent on model implementation, severely restricting its application scope and practical utility.

In this work, we introduce MUSE, a **model-agnostic** watermarking algorithm for tabular data that operates without relying on the invertibility of diffusion models. A key insight enabling our approach is that tabular data generation demands **significantly less computation** than image or video generation, as shown in Figure 1 (right). This computational efficiency makes a multi-sample selection process practical. MUSE leverages this by generating m candidate samples for each data row. A watermark is embedded by selecting one candidate based on a score function, which is calculated using values from specific columns. Because it does not depend on model invertibility, MUSE is broadly applicable to any tabular generative model. We also show that our method can maintain the original data distribution (distortion-free) when using certain column selection techniques. Furthermore, we establish a mathematical relationship between detectability, the number of candidates (m), and the number of rows (N), ensuring reliable watermark detection.

With these mechanisms, MUSE achieves high watermark detectability while preserving the generation quality of the underlying model, as validated across diverse tabular datasets and evaluation metrics. Moreover, MUSE exhibits strong robustness against a wide range of post-generation attacks specific to tabular data. Importantly, MUSE is model-agnostic and compatible with any tabular generative model supporting repeated sampling. This flexibility allows MUSE to integrate seamlessly with sampling-efficient diffusion models, such as TabSyn [Zhang et al., 2024c], thereby effectively mitigating the computational overhead introduced by repeated sampling processes. We summarize the main contributions of this paper as follows:

- We propose tabular watermarking via multi-sample selection (MUSE), a novel generative watermarking method for tabular data that completely avoids the inversion process in the data processing and sampling pipeline.
- We provide a theoretical analysis of the detectability of MUSE, which enables precise calibration of watermarking strength under a given detection threshold.
- Extensive experiments across multiple tabular datasets demonstrate that MUSE consistently achieves state-of-the-art performance in generation quality, watermark detectability, and robustness against various tabular-specific attacks.

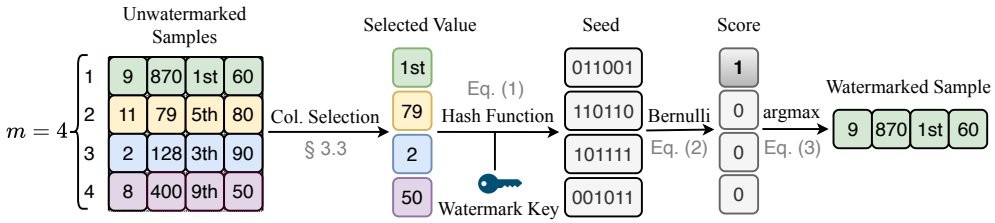

Figure 2: An overview of the MUSE watermark generation process. MUSE operates by generating multiple samples and selecting the highest-scoring sample (ties are broken randomly). The selected row is appended to the watermarked table, while others are discarded.

2 Preliminaries

Tabular Generative Models. A tabular dataset with N rows and M columns consists of $i.i.d.$ samples $(\mathbf{x}_i)_{i=1}^N$ drawn from an unknown joint distribution $p(\mathbf{x})$, where each $\mathbf{x}_i \in \mathbb{R}^M$ (or mixed-type space) represents a data row with M features. A tabular generative model aims to learn a parameterized distribution $p_\theta(\mathbf{x}) \approx p(\mathbf{x})$ to generate new realistic samples.

Watermark for Tabular Generative Models. Tabular watermark involves two main functions.

1. **Generate**: Given a secret watermark key k, this function produces a watermarked table. Similar to standard generation, each row of this table is sampled i.i.d., but from a distribution $p(\mathbf{x}, k)$.
2. **Detect**: Provided with a table and a specific key k, this function examines the table to determine if it carries the watermark associated with that particular key.

Threat Model. We consider the following watermarking protocol between three parties: the tabular data provider, the user, and the detector.

1. The tabular data provider shares a watermark key k with the detector.
2. The user asks the tabular data provider to generate a table T.
3. The user publishes a table T', which can either be an (edited version of the) original table T or an independent table.
4. The detector determines whether the table T' is watermarked or not.

3 Proposed Method

In this section, we introduce MUSE, a model-agnostic watermarking method for tabular generative models. Considering the low reverse accuracy of tabular diffusion models and fast generation compared to other modalities, our method is based on multi-sample selection instead of DDIM inversion. We first introduce the overall watermark generation process in Section 3.1 and then introduce the Watermark Scoring Function and Column Selection Mechanism in Section 3.2 and Section 3.3 respectively, followed by the detailed watermark detection in Section 3.4.

3.1 Watermark Generation Framework

We define the overall generation process of our MUSE method in this section. The generation of each watermarked row can be decomposed into the following two steps:

1. Sample m candidate rows from the tabular generative model $p(\mathbf{x})$.
2. Apply a *watermark scoring function* $s_k(\mathbf{x})$ to each candidate using watermark key k and select the **highest-scoring candidate** as the watermarked row. Details of the watermark scoring function will be introduced in section 3.2.

Then we could repeat the above process N times to generate the watermarked table. In practice, the selection procedure can be parallelized across the N groups since each group contains $i.i.d.$ samples. The overall process is illustrated in Figure 2.

Discussions. The advantages of this approach are twofold: (1) *Model-agnostic*: since we do not modify the row generation process of the tabular generative model, the method is applicable to any tabular generative model that supports repeated sampling. (2) *Distribution-preserving*: we demonstrate in subsequent sections that the watermark remains unbiased under specific watermark scoring functions.

3.2 Watermark Scoring Function

In this section, we detail the design of our watermark scoring function, denoted as $s(\mathbf{x}, k)$, which involves two key steps as follows:

1. **Column Selection Mechanism**: Given an input sample $\mathbf{x}$ which consists of M columns, the first step involves selecting a specific subset of n columns from these M available columns. Let $\pi(\mathbf{x})$ represent the operation that extracts the values from this chosen subset of n columns of $\mathbf{x}$. The specific strategies for how these n columns are selected (e.g., adaptive or fixed methods) will be further elaborated in Section 3.3.
2. **Score Generation**: The second step takes the selected column values $\pi(\mathbf{x})$ from the previous stage and the predefined watermark key k as inputs. These are processed by a cryptographic hash function $H(\cdot)$ to produce a deterministic hash value:

$$h = H(\pi(\mathbf{x}), k). \tag{1}$$

   This resulting hash value h is then used to seed or as an input to a pseudorandom function f. This function f maps the hash h to a scalar score $s \in [0, 1]$. In our implementation, we define f as a Bernoulli distribution with a mean of 0.5:

$$s_k(\mathbf{x}_i) = f(h_i), \quad f \sim \text{Bernoulli}(0.5). \tag{2}$$

   The rationale behind choosing a Bernoulli(0.5) distribution is discussed in Lemma 3.2.

Discussion on Scoring Design. An important question is why our scoring function incorporates column selection rather than simply using $H(k)$ as the score. The key insight is that content-dependent scoring through $\pi(\mathbf{x})$ makes each row's selection appear **pseudo-random** while still respecting the original data distribution $p(\mathbf{x})$. If we used a fixed score based solely on the watermark key, the multi-sample selection process would become arbitrary, potentially biasing the resulting distribution away from $p(\mathbf{x})$.

3.3 Column Selection Mechanism

In this section, we introduce two strategies for the column selection procedure $\pi(\cdot)$ used in our watermark scoring function (Equation (1)):

1. **Fixed Column Selection**: A straightforward approach that uses a predefined set of columns for all samples, regardless of data characteristics.
2. **Adaptive Column Selection**: A dynamic approach that selects columns based on properties of each individual sample, as detailed below.

For the adaptive strategy, we leverage each sample's statistical properties. Unlike sequential language models that naturally provide randomness [Kirchenbauer et al., 2023, Dathathri et al., 2024, Zhao et al., 2023], tabular generation is often non-sequential. Therefore, we use the sample itself as a source of entropy.

Specifically, we select columns based on how sample $\mathbf{x}$ deviates from the training data distribution T. For each column $j \in \{1, \ldots, M\}$, we compute the empirical quantile rank:

$$r_j = \frac{1}{N} \sum_{i=1}^{N} \mathbb{1}\{T_{i,j} \leq \mathbf{x}_j\}, \tag{3}$$

where N is the number of training samples. We then select columns (typically 3) corresponding to the smallest, median, and largest r_j values, resulting in an index set $\mathcal{J} = \pi(\mathbf{x})$ for hash computation.

Discussion. The adaptive column selection offers several advantages: (1) *Enhanced Security*: By varying columns between samples, it prevents attackers from targeting specific columns; (2) *Increased Diversity*: The diversity in column selection improves the uniformity of hash values, minimizing distortion in watermarked data; (3) *Improved Robustness*: By linking selection to the data's empirical distribution, the method better withstands post-processing attacks that don't significantly alter the overall statistical properties. The complete MUSE watermark generation algorithm is presented in Algorithm 1.

Repeated Column Masking. While our column selection mechanism introduces diversity, achieving **true pseudo-randomness** requires an additional safeguard. We adopt a repeated column masking strategy that detects when previously selected column values reappear. In such cases, the scoring function assigns a random score, thereby skipping watermark embedding for that instance. This mechanism is inspired by the *repeated key masking* technique used in LLM watermarking [Hu et al., 2023, Dathathri et al., 2024]. Further implementation details and experimental results are provided in Section 4.3.

3.4 Watermark Detection

Detection Statistic. Since MUSE biases the selection process toward high-scoring samples, the sum of scores for a watermarked table is expected to be higher than that of an unwatermarked table. Given a (watermarked or unwatermarked) table T consists of N rows: $T := (\mathbf{x}_1, \ldots, \mathbf{x}_N)$. We detect watermark by computing the mean score:

$$S(T) = \frac{1}{N} \sum_{i=1}^{N} s_k(\mathbf{x}_i). \tag{4}$$

We then compare with the mean score $S(T_{\text{no-wm}})$ of an unwatermarked table $T_{\text{no-wm}}$. We conclude that T is watermarked if $S(T) > S(T_{\text{no-wm}})$.

Calibrating the Number of Repeated Samples. Given the detection statistic Equation (4), we move on to show how the detectability of MUSE depends on (1) the number of watermarked samples N and (2) the number of repeated samples m.

Lemma 3.1. *Denote a watermarked table as T_{wm} and an unwatermarked table as $T_{\text{no-wm}}$, each consisting of N rows. Let $\mathbf{x} \sim p(\mathbf{x})$ be a random variable drawn from the data distribution, and let $\mathbf{x}_1, \ldots, \mathbf{x}_m$ be i.i.d. samples from $p(\mathbf{x})$. Define $\mu_{\text{no-wm}} = \mathbb{E}_{\mathbf{x}\sim p(\mathbf{x})}[s_k(\mathbf{x})]$ as the expected score of an unwatermarked sample, and define $\mu^m_{\text{wm}} = \mathbb{E}_{\mathbf{x}_i \sim p(\mathbf{x})}\left[\max_{i\in[m]} s_k(\mathbf{x}_i)\right]$ as the expected score of a watermarked sample obtained via m repeated samples. Suppose the scoring function satisfies $s_k(\cdot) \in [0, 1]$, then, the False Positive Rate (FPR) of the watermark detection satisfies:*

$$\Pr\left(S(T_{\text{no-wm}}) > S(T_{\text{wm}})\right) \leq \exp\left(-\frac{N(\mu^m_{\text{wm}} - \mu_{\text{no-wm}})^2}{2}\right). \tag{5}$$

Lemma 3.2 (Optimal Scoring Distribution)**.** *Let $s_k(\mathbf{x})$ be any random variable supported on $[0, 1]$ with mean 0.5, the right-hand-side of Equation (5) is minimized when $s_k(\mathbf{x})$ follows a Bernoulli(0.5) distribution.*

Theorem 3.3 (Minimum Watermarking Signal)**.** *Under the same assumptions as in Lemma 3.1, suppose the scoring function $s_k(\mathbf{x})$ is instantiated as a hash-seeded pseudorandom function such that $s_k(\mathbf{x}) \sim$ Bernoulli(0.5), the FPR is upper-bounded by:*

$$\Pr\left(S(T_{\text{no-wm}}) > S(T_{\text{wm}})\right) \leq \exp\left(-\frac{N}{2}\left(0.5 - 0.5^m\right)^2\right). \tag{6}$$

To ensure the FPR does not exceed a target threshold α, it suffices to set the number of repeated samples m as:

$$m = \max\left(2, \left\lceil \log_{0.5}\left(0.5 - \sqrt{\tfrac{2\log(1/\alpha)}{N}}\right)\right\rceil\right), \tag{7}$$

where $\lceil\cdot\rceil$ denotes the ceiling function. This expression is valid when $N > 8\log(1/\alpha)$.

Algorithm 1 MUSE Watermark Generation

Input: watermark key k, unwatermarked table $T \in \mathbb{R}^{N \times M}$, False Positive Rate α
Compute the number of repeated samples m based on N and α via Equation (7)
Randomly split rows of T into N/m groups: $(\mathcal{G}_i)_{i=1}^{N/m}$, each containing m rows
Initialize an empty set $\mathcal{R}$, and a list T_{wm} to store the watermarked table
for $i \leftarrow 1$ to N/m **do**
  $\mathbf{x}_1, \ldots, \mathbf{x}_m \leftarrow \mathcal{G}_i$
  for $t \in \{1, \ldots, m\}$ **do** ▷ Adaptive column selection.
    Let $\mathbf{x} \leftarrow \mathbf{x}_t$
    For each column j, compute quantile rank r_j of $\mathbf{x}_j$ in $T[:, j]$ ▷ See Equation (3).
    Sort $\{r_j\}_{j=1}^{M}$ and identify column indices with min, median, and max ranks
    Let $\mathcal{J}_t$ be the set of selected column indices
  end for
  for $t \in \{1, \ldots, m\}$ **do** ▷ Multi-sample selection.
    $r_t = \text{hash}(k, \mathbf{x}_t[\mathcal{J}_t])$
    Seed Bernoulli distribution with r_t
    $s_t \sim \text{Bernoulli}(0.5)$
  end for
  $i \leftarrow \arg\max_{t \in \{1,\ldots,m\}} s_t$
  Append $\mathbf{x}_i$ to T_{wm}
end for
return T_{wm}

Theorem 3.3 enables MUSE to calibrate the number of repeated samples m to achieve a target false positive rate with theoretical guarantees. This allows the method to embed *just enough* watermarking signal to ensure the desired detectability. Intuitively, since no redundant watermarking signal is embedded, the impact of watermarking on the generation quality is minimal. In Figure 3, we plot m as a function of table size N for various target FPRs, based on Equation (7) (omitting the ceiling operation for clarity). We observe that m quickly saturates as N increases. For instance, to achieve a 0.01% FPR, $m = 2$ suffices when $N \geq 300$, and even for $N = 100$, $m = 4$ is enough. In the rest of the paper, MUSE's m is set by Equation (7) unless otherwise specified.

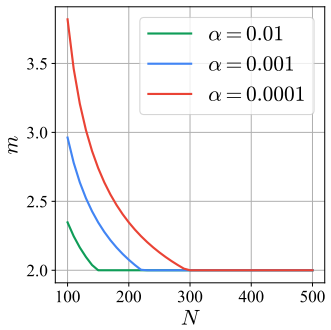

Figure 3: m vs. N under different α values (smoothed).

4 Experiments

In this section, we provide a comprehensive empirical evaluation of MUSE. We aim to answer the following research questions: **Q1: Detectability v.s. Invisibility** (§4.2): Can MUSE achieve strong detectability while preserving the distribution of the generated data? **Q2: Robustness** (§4.2): How resilient is the watermark to a range of post-processing attacks, such as row/column deletion or value perturbation? **Q3: Component-wise Analysis** (§4.3): How does MUSE perform under different design choices of its components?

4.1 Setup

Datasets. We select six real-world tabular datasets containing both numerical and categorical attributes: `Adult`, `Default`, `Shoppers`, `Magic`, `Beijing` and `News`. Due to space constraints, we defer the results on `News` and `Beijing` to Appendix A.1. The statistics of the datasets are summarized in Table 5 in Appendix C.2.

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 0.1%, denoted as *TPR@0.1%FPR*. (b) To evaluate the distortion of the watermarked data, we follow standard fedelity and utility metrics used in tabular data generation

Table 1: Watermark generation quality and detectability, indicates best performance, indicates second-best performance. For clarity, only our method is highlighted in detection.

Dataset	Method	Watermark Generation Quality				Watermark Detectability			
		Num. Training Rows				100		500	
		Marg.	**Corr.**	**C2ST**	**MLE Gap**	**AUC**	**T@0.1%F**	**AUC**	**T@0.1%F**
Adult	w/o WM	0.994	0.984	0.996	0.017	-	-	-	-
	TR	0.919	0.870	0.676	0.046	0.590	0.004	0.774	0.171
	GS	0.751	0.619	0.058	0.084	1.000	1.000	1.000	1.000
	TabWak	0.935	0.885	0.769	0.048	0.844	0.089	0.990	0.592
	TabWak*	0.933	0.879	0.713	0.085	0.999	0.942	1.000	1.000
	MUSE	0.979	0.963	0.883	0.017	1.000	1.000	1.000	1.000
Default	w/o WM	0.990	0.934	0.979	0.000	-	-	-	-
	TR	0.895	0.888	0.564	0.161	0.579	0.001	0.848	0.034
	GS	0.701	0.678	0.059	0.182	1.000	1.000	1.000	1.000
	TabWak	0.911	0.902	0.568	0.156	0.896	0.071	0.997	0.611
	TabWak*	0.906	0.894	0.550	0.176	0.965	0.218	1.000	0.995
	MUSE	0.983	0.925	0.963	0.002	1.000	1.000	1.000	1.000
Magic	w/o WM	0.990	0.980	0.998	0.008	-	-	-	-
	TR	0.898	0.936	0.621	0.129	0.652	0.014	0.592	0.102
	GS	0.688	0.838	0.030	0.064	1.000	1.000	1.000	1.000
	TabWak	0.905	0.929	0.605	0.120	0.904	0.067	0.997	0.737
	TabWak*	0.891	0.916	0.520	0.100	0.873	0.050	0.995	0.687
	MUSE	0.991	0.982	0.999	0.010	1.000	1.000	1.000	1.000
Shoppers	w/o WM	0.985	0.974	0.974	0.017	-	-	-	-
	TR	0.888	0.880	0.501	0.077	0.575	0.001	0.830	0.058
	GS	0.729	0.688	0.061	0.154	1.000	1.000	1.000	1.000
	TabWak	0.903	0.886	0.548	0.132	0.860	0.106	0.990	0.353
	TabWak*	0.897	0.879	0.525	0.384	0.742	0.002	0.981	0.185
	MUSE	0.982	0.974	0.950	0.015	1.000	1.000	1.000	1.000

[Zhang et al., 2024c, 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., 2024c] for a more detailed definition of each evaluation metric.

Baselines. Since our method belongs to the class of generative watermarking techniques, we primarily compare it with TabWak [Zhu et al., 2025], which is the only existing generative watermarking approach for tabular data. We use the official implementations of both TabWak and its improved variant TabWak* in all experiments to ensure consistency. In addition, following the experimental setup in TabWak, we include two image watermarking methods, TreeRing (TR) and Gaussian Shading (GS), as auxiliary baselines. For completeness, we also evaluate against representative edit-based watermarking methods, including TabularMark [Zheng et al., 2024] and WGTD [He et al., 2024]. Due to space constraints, detailed descriptions and results for these methods are deferred to Appendix A.3.

Implementation Details. We use the same tabular generative model as TabWak, namely TabSyn [Zhang et al., 2024c], and train it using the official codebase. For TabWak, we use its official implementation to generate watermarked data. Notably, it bypasses the inversion of quantile normalization, which assumes access to ground-truth data not available for watermark detection, potentially giving it an advantage under our evaluation protocol. Generation quality is evaluated across ten repetitions, and we report the averaged results.

4.2 Main Results

Distortion and Detectability. We address the first question: whether the watermarking method achieves high watermark detectability while introducing minimal distortion to the generated data. As shown in Table 1 and Appendix A.1, MUSE consistently achieves strong performance across both fidelity and detection metrics on all six datasets. It yields the highest marginal statistics,

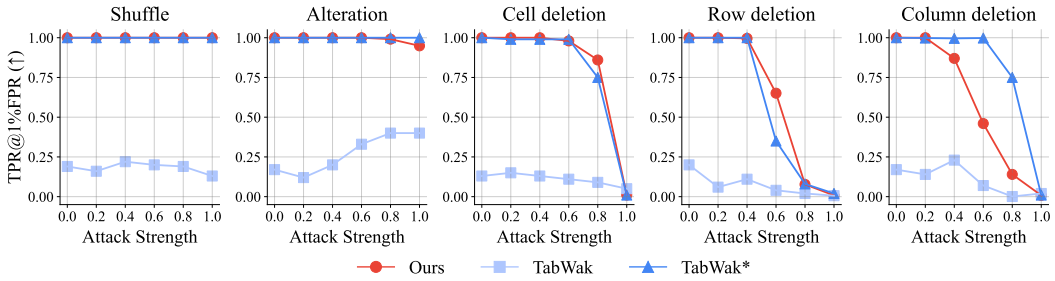

Figure 4: Detection performance of watermarking methods against different types of tabular data attacks across varying attack intensities.

correlation, C2ST and MLE, often closely matching the unwatermarked baseline. This suggests that the underlying data distribution is well preserved. In terms of detectability, MUSE achieves nearly perfect detection performance across all datasets and detection budgets, as measured by both AUC and T@0.1%F. For example, on the `Default` dataset, it attains a marginal statistic of 0.983 and AUC of 1.000, compared to 0.911 and 0.896 for TabWak. Notably, while GS also achieves strong detection scores, this comes at the cost of significantly higher distortion across all fidelity metrics. For instance, on the `Adult` dataset, GS results in a C2ST of only 0.058 and correlation of 0.619, in contrast to the higher values of 0.883 and 0.963 from MUSE, respectively. These results indicate that MUSE embeds a detectable watermark signal while preserving the statistical properties of the generated data to a greater extent than existing approaches.

Robustness against Attacks. We assess the robustness of watermarking methods under five representative attacks on tabular data: *row shuffling*, *row deletion*, *column deletion*, *cell deletion*, and *value alteration*. Each attack is applied at varying intensities, with attack percentages ranging from 0.0 to 1.0 in increments of 0.2. For deletion-based attacks, a specified fraction of rows, columns, or cells is randomly removed and replaced with unwatermarked values independently sampled from the same generative model. In the value alteration attack, selected numerical values are perturbed by multiplying each with a scalar drawn uniformly from $(0.8, 1.2)$. Row shuffling permutes a fraction of the dataset's rows. We benchmark the detectability of MUSE against TabWak and TabWak* on the `Adult` dataset, using N=500 and m=2. As shown in Figure 4, MUSE consistently outperforms or matches the performance of TabWak and TabWak* under four of the five attacks—row shuffling, row deletion, cell deletion, and value alteration. Under column deletion, however, MUSE shows a drop in performance due to its watermark being embedded via selected columns, which are partially removed by this attack. For benchmark results on other datasets, we refer readers to Appendix A.2.

Table 2: Component-wise ablation study of MUSE. Each color block indicates a different component of the method. Details of the experimental setup are in §4.3.

Model	Score func.	Col. Select	Mask	Num. Col.	z-stat.↑	Marg.↑	Corr.↑	C2ST↑	MLE Gap↓
TabSyn	Bernoulli	Adaptive	No	3	7.348	**0.979**	**0.963**	**0.883**	**0.017**
TabDAR	Bernoulli	Adaptive	No	3	7.270	0.977	0.958	0.880	0.018
DP-TBART	Bernoulli	Adaptive	No	3	**7.544**	0.951	0.931	0.759	0.020
TabSyn	Bernoulli	Adaptive	No	3	**7.348**	**0.979**	**0.963**	**0.883**	0.017
TabSyn	Uniform	Adaptive	No	3	5.012	0.964	0.940	0.808	**0.015**
TabSyn	Bernoulli	Adaptive	No	3	**7.348**	**0.979**	**0.963**	**0.883**	0.017
TabSyn	Bernoulli	Fixed	No	3	5.439	0.949	0.907	0.601	**0.015**
TabSyn	Bernoulli	Adaptive	No	3	**7.348**	0.979	0.963	0.883	**0.017**
TabSyn	Bernoulli	Adaptive	Yes	3	4.819	**0.985**	**0.973**	**0.940**	**0.017**
TabSyn	Bernoulli	Adaptive	No	1	4.987	0.931	0.879	0.544	**0.015**
TabSyn	Bernoulli	Adaptive	No	3	7.348	0.979	0.963	0.883	0.017
TabSyn	Bernoulli	Adaptive	No	5	8.624	0.989	0.969	0.983	0.017
TabSyn	Bernoulli	Adaptive	No	7	**8.728**	**0.990**	**0.976**	**0.995**	0.018

4.3 Ablation Study and Further Analysis

We perform a component-wise ablation study to evaluate the contribution of each design choice in our watermarking framework. All experiments are conducted on the `Adult` dataset and we generate

watermarked table with $N = 100$ rows, if not otherwise specified. For detectability, we report the z-statistic defined as $\frac{\sum_{i=1}^{N} s_k(\mathbf{x}_i) - N/2}{\sqrt{N/4}}$.

Score Function. We compare two scoring distributions: (1) a Bernoulli distribution with mean 0.5, and (2) a uniform distribution over $[0, 1]$. As shown in Table 2, MUSE with the Bernoulli score yields higher detectability. This result aligns with our theoretical analysis in Lemma 3.2, which identifies Bernoulli(0.5) as the optimal scoring distribution under our detection formulation.

Column Selection. We compare adaptive column selection strategy with a fixed set strategy that selects the first three columns. As shown in Table 2, adaptive column selection leads to higher detectability. We also investigate the effect of varying the number of selected columns. Increasing the number of selected columns generally improves both detectability and generation quality due to improved diversity for the hash function. However, using more columns increases vulnerability to column deletion attacks. Additionally, detectability tends to saturate beyond three columns. Thus, we use three adaptively selected columns in all main experiments for a balanced trade-off.

Distortion-Free Watermarking. Ideally, the selection process described in Section 3 would introduce no distortion to the data distribution if it were entirely independent of sample values. In practice, however, some dependence is necessary to ensure watermark detectability. To approximate this ideal, we leverage the insight that if a value in the selected columns has not been previously used for watermarking, its selection can be considered effectively random. To enforce this, we implement a masking mechanism that tracks previously watermarked values and skips watermarking on samples that would reuse them. As shown in Table 2, this mechanism helps preserve the underlying data distribution and improves generation quality. However, it also reduces the number of watermarked samples, slightly compromising overall detection strength.

Model-Agnostic Applicability. While our primary experiments are based on a diffusion model [Zhang et al., 2024c], MUSE is universally applicable to any generative model and pre-allocated tabular data. To demonstrate this, we evaluate MUSE on two additional representative paradigms of tabular generative modeling: (1) *Autoregressive models*: we adopt DP-TBART [Castellon et al., 2023], a transformer-based autoregressive model that predicts each tabular entry conditioned on preceding entries; and (2) *Masked generative models*: we use TabDAR [Zhang et al., 2024a], a masked autoencoder model that predicts randomly masked values. As shown in Table 2, MUSE consistently achieves strong detectability and generation quality across all three model families, confirming its generality and robustness across diverse generative architectures.

Computation Time. We compare the effective watermarking time (generation + detection) of MUSE with baselines that rely on DDIM inversion. We generate 10K watermarked rows of the `Adult` dataset. As shown in Figure 5, MUSE achieves significantly lower detection time by avoiding the costly inversion process. Notably, its generation time is also lower than that of the baselines, despite using multi-sample generation ($m = 2$). This efficiency arises from MUSE's compatibility with fast score-based diffusion models [Zhang et al., 2024c, Karras et al., 2022], which require only 50 sampling steps, compared to the 1,000 steps typically needed for DDIM inversion to ensure sufficient discretization.

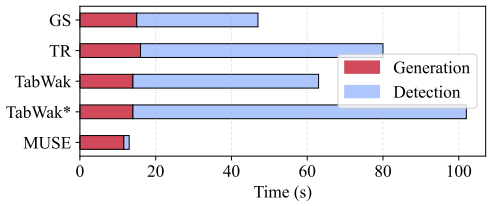

Figure 5: Watermark generation and detection time of MUSE and inversion-based baselines.

5 Related Work

Generative Watermarking. Generative watermarking embeds watermark signals during the generation process, typically by manipulating the generation randomness through pseudorandom seeds. This approach has proven effective and efficient for watermarking in image, video, and large language model (LLM) generation. In image and video generation, where diffusion-based models are the *de facto* standard, watermarking methods inject structured signals into the noise vector in latent space [Wen et al., 2023, Yang et al., 2024, Huang et al., 2024]. Detection involves inverting the diffusion sampling process [Dhariwal and Nichol, 2021, Hong et al., 2024, Pan et al., 2023] to recover

the original noise vector and verify the presence of the embedded watermark. For LLMs, generative watermarking methods fall into two categories: (1) *Watermarking during logits generation*, which embeds signals by manipulating the model's output logits distribution [Kirchenbauer et al., 2023, Zhao et al., 2023, Hu et al., 2023, Dathathri et al., 2024, Giboulot and Furon, 2024, Liu et al., 2023]; and (2) *Watermarking during token sampling*, which preserves the logits distribution but replaces the stochastic token sampling process (e.g., multinomial sampling) with a pseudorandom procedure seeded for watermarking [Aaronson and Kirchner, 2022, Kuditipudi et al., 2023, Christ et al., 2024]. In this sense, sampling-based watermarking is conceptually similar to inversion-based watermarking used in diffusion models. We refer the reader to [Liu et al., 2024, Pan et al., 2024] for a comprehensive survey of watermarking for LLMs. Closest to our approach are SynthID [Dathathri et al., 2024] and Watermax [Giboulot and Furon, 2024], both of which embed watermarks via repeated logit generation. However, our approach is specifically designed for unconditional tabular data generation, unlike these methods which primarily target discrete text. This focus on tabular data introduces unique challenges due to its distinct data structure. Consequently, our watermarking technique is engineered for robustness against a different set of attacks prevalent in the tabular domain.

Watermarking for Tabular Data Traditional tabular watermarking techniques are edit-based, injecting signals by modifying existing data values. WGTD [He et al., 2024] embeds watermarks by altering the fractional parts of continuous values using a green list of intervals, but it is inapplicable to categorical-only data. TabularMark [Zheng et al., 2024] perturbs values in a selected numerical column using pseudorandom domain partitioning, but relies on access to the original table for detection, limiting its robustness in adversarial settings. Another significant drawback of such methods is the potential to distort the original data distribution or violate inherent constraints. To overcome this, TabWak [Zhu et al., 2025] introduced the first generative watermarking approach for tabular data. Analogous to inversion-based watermarks in diffusion models, TabWak embeds detectable patterns into the noise vector within the latent space. It also employs a self-clone and shuffling technique to minimize distortion to the data distribution. While TabWak avoids post-hoc editing, its reliance on inverting both the sampling process (e.g., DDIM [Song et al., 2020]) and preprocessing steps (e.g., quantile normalization [Wikipedia contributors, 2025]) can introduce reconstruction errors. These errors will in turn impair the watermark's detectability.

6 Conclusion

We propose MUSE, a model-agnostic watermarking method that embeds signals via multi-sample selection, eliminating the need for costly and error-prone inversion procedures. MUSE achieves strong detectability while introducing negligible distributional distortion and seamlessly scales across a wide range of generative models. Extensive experiments on benchmark datasets validate its effectiveness, consistently outperforming prior edit-based and inversion-based approaches in both generation fidelity and watermark robustness. As synthetic tabular data becomes increasingly adopted in high-stakes domains such as healthcare, finance, and social science, ensuring data traceability and integrity is critical. MUSE provides a practical and generalizable solution for watermarking synthetic data, enabling reliable provenance tracking, ownership verification, and misuse detection. We believe this work opens new avenues for trustworthy synthetic data generation and highlights the importance of integrating security considerations into the core of data-centric AI systems.

References

Scott Aaronson and Hendrik Kirchner. Watermarking gpt outputs. https://www.scottaaronson.com/talks/watermark.ppt, 2022. Presentation.

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

Andreas Blattmann, Tim Dockhorn, Sumith Kulal, Daniel Mendelevitch, Maciej Kilian, Dominik Lorenz, Yam Levi, Zion English, Vikram Voleti, Adam Letts, et al. Stable video diffusion: Scaling latent video diffusion models to large datasets. arXiv preprint arXiv:2311.15127, 2023.

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.

Rodrigo Castellon, Achintya Gopal, Brian Bloniarz, and David Rosenberg. Dp-tbart: A transformer-based autoregressive model for differentially private tabular data generation. arXiv preprint arXiv:2307.10430, 2023.

Miranda Christ, Sam Gunn, and Or Zamir. Undetectable watermarks for language models. In The Thirty Seventh Annual Conference on Learning Theory, pages 1125–1139. PMLR, 2024.

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.

Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. Advances in neural information processing systems, 34:8780–8794, 2021.

Liancheng Fang, Aiwei Liu, Hengrui Zhang, Henry Peng Zou, Weizhi Zhang, and Philip S Yu. Tabgen-icl: Residual-aware in-context example selection for tabular data generation. arXiv preprint arXiv:2502.16414, 2025.

Joao Fonseca and Fernando Bacao. Tabular and latent space synthetic data generation: a literature review. Journal of Big Data, 10(1):115, 2023.

Eva Giboulot and Teddy Furon. Watermax: breaking the llm watermark detectability-robustness-quality trade-off. arXiv preprint arXiv:2403.04808, 2024.

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

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

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

Seongmin Hong, Kyeonghyun Lee, Suh Yoon Jeon, Hyewon Bae, and Se Young Chun. On exact inversion of dpm-solvers. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 7069–7078, 2024.

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.

Zhengmian Hu, Lichang Chen, Xidong Wu, Yihan Wu, Hongyang Zhang, and Heng Huang. Unbiased watermark for large language models. arXiv preprint arXiv:2310.10669, 2023.

Huayang Huang, Yu Wu, and Qian Wang. Robin: Robust and invisible watermarks for diffusion models with adversarial optimization. Advances in Neural Information Processing Systems, 37: 3937–3963, 2024.

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. Advances in neural information processing systems, 35:26565–26577, 2022.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114, 2013.

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, pages 17061–17084. PMLR, 2023.

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

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

Aiwei Liu, Leyi Pan, Xuming Hu, Shiao Meng, and Lijie Wen. A semantic invariant robust watermark for large language models. ArXiv, abs/2310.06356, 2023.

Aiwei Liu, Leyi Pan, Yijian Lu, Jingjing Li, Xuming Hu, Xi Zhang, Lijie Wen, Irwin King, Hui Xiong, and Philip Yu. A survey of text watermarking in the era of large language models. ACM Computing Surveys, 57(2):1–36, 2024.

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), pages 3565–3569. IEEE, 2021.

Leyi Pan, Aiwei Liu, Zhiwei He, Zitian Gao, Xuandong Zhao, Yijian Lu, Binglin Zhou, Shuliang Liu, Xuming Hu, Lijie Wen, et al. Markllm: An open-source toolkit for llm watermarking. arXiv preprint arXiv:2405.10051, 2024.

Zhihong Pan, Riccardo Gherardi, Xiufeng Xie, and Stephen Huang. Effective real image editing with accelerated iterative diffusion inversion. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 15912–15921, 2023.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 10684–10695, 2022.

Juntong Shi, Minkai Xu, Harper Hua, Hengrui Zhang, Stefano Ermon, and Jure Leskovec. Tabdiff: a unified diffusion model for multi-modal tabular data generation. In NeurIPS 2024 Third Table Representation Learning Workshop, 2024.

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

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.

Wikipedia contributors. Quantile normalization – Wikipedia, the free encyclopedia. https://en.wikipedia.org/wiki/Quantile_normalization, 2025. Accessed: 2025-05-11.

Junchao Wu, Shu Yang, Runzhe Zhan, Yulin Yuan, Lidia Sam Chao, and Derek Fai Wong. A survey on llm-generated text detection: Necessity, methods, and future directions. Computational Linguistics, pages 1–66, 2025.

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, pages 12162–12171, 2024.

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.

Hengrui Zhang, Liancheng Fang, and Philip S Yu. Unleashing the potential of diffusion models for incomplete data imputation. arXiv preprint arXiv:2405.20690, 2024b.

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, 2024c.

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

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, pages 3570–3584, 2024.

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.

Appendix

A Additional Experiments Results

A.1 Omitted Results on Distortion and Detectability

We present the omitted results on distortion and detectability in Table 3.

Table 3: Watermark generation quality and detectability, indicates best performance, indicates second-best performance. For clarity, only our method is highlighted in detection.

Dataset	Method	Watermark Generation Quality				Watermark Detectability			
		Num. Training Rows				100		500	
		Marg.	**Corr.**	**C2ST**	**MLE Gap**	**AUC**	**T@0.1%F**	**AUC**	**T@0.1%F**
Beijing	w/o WM	0.977	0.958	0.934	0.199	-	-	-	-
	TR	0.914	0.873	0.734	0.396	0.577	0.000	0.548	0.007
	GS	0.656	0.529	0.097	0.715	1.000	1.000	1.000	1.000
	TabWak	0.923	0.871	0.792	0.375	0.925	0.096	0.999	0.978
	TabWak*	0.917	0.860	0.761	0.403	0.996	0.734	1.000	1.000
	MUSE	0.972	0.955	0.926	0.209	1.000	1.000	1.000	1.000
News	w/o WM	0.960	0.973	0.899	0.024	-	-	-	-
	TR	0.899	0.963	0.641	0.041	0.547	0.000	0.549	0.005
	GS	0.673	0.907	0.031	0.065	1.000	1.000	1.000	1.000
	TabWak	0.929	0.968	0.749	0.066	0.998	0.869	1.000	1.000
	TabWak*	0.924	0.964	0.719	0.044	1.000	0.991	1.000	1.000
	MUSE	0.959	0.973	0.883	0.033	1.000	1.000	1.000	1.000

A.2 Omitted Results on Robustness

We present the omitted robustness results in Figure 6, where MUSE is compared against TabWak and TabWak* on the `Beijing`, `Default`, `Magic`, `News`, and `Shoppers` datasets. Overall, MUSE demonstrates stronger robustness under cell deletion and row deletion attacks, while achieving comparable performance on alteration and column deletion attacks. Both MUSE and TabWak/TabWak* remain resilient to shuffle attacks, due to embedding watermarks at the individual row level. Notably, we observe that TabWak and TabWak* exhibit instability on certain datasets, such as `Shoppers` and `Beijing`, where detection performance fluctuates—first decreasing and then increasing—as attack intensity increases. We hypothesize that this behavior stems from the inherent instability of the VAE inversion process.

A.3 Omitted Results on Edit-based Watermarking

We compare our method against two representative **edit-based** watermarking baselines, which embed watermarks by directly altering table entries. Since the official implementations of these methods are not publicly available, we reimplement them based on the descriptions in their original papers. We first outline their core methodologies and our reimplementation details, then present the comparative results in Table 4. **Our reproduced codes are provided in the supplementary material.** Below are the detailed implementations of the baselines.

WGTD [He et al., 2024]. WGTD embeds watermarks by modifying the fractional part of continuous data points, replacing them with values from a predefined green list. Consequently, **it is limited to continuous data and cannot be applied to tables containing only categorical features**.

The watermarking process in WGTD involves three main steps: (i) dividing the interval $[0, 1]$ into $2m$ equal sub-intervals to form m pairs of consecutive intervals; (ii) randomly selecting one interval from each pair to construct a set of m "green list" intervals; and (iii) replacing the fractional part of each data point with a value sampled from the nearest green list interval, if the original does not already fall within one. Detection is performed via a hypothesis-testing framework that exploits the statistical properties of the modified distribution to reliably identify the presence of a watermark. For reproducibility, we adopt the original hyperparameter setting with $m = 5$ green list intervals.

TabularMark [Zheng et al., 2024]. TabularMark embeds watermarks by perturbing specific cells in the data. It first pick a selected attribute/column to embed the watermark, then it generate pesudorandom partition of a fixed range into multiple unit domains, and label them with red and green domains, and finally perturb the selected column with a random number from the green domain.

Table 4: Watermark generation quality and detectability, indicates best performance, indicates second-best performance. For clarity, only our method is highlighted in detection.

Dataset	Method	Watermark Generation Quality				Watermark Detectability			
		Num. Training Rows				100		500	
		Marg.	**Corr.**	**C2ST**	**MLE Gap**	**AUC**	**T@0.1%F**	**AUC**	**T@0.1%F**
Adult	w/o WM	0.994	0.984	0.996	0.017	-	-	-	-
	TabularMark	0.983	0.949	0.987	0.021	1.000	1.000	1.000	1.000
	WGTD	0.987	0.972	0.978	0.019	1.000	1.000	1.000	1.000
	MUSE	0.979	0.963	0.883	0.017	1.000	1.000	1.000	1.000
Beijing	w/o WM	0.977	0.958	0.934	0.199	-	-	-	-
	TabularMark	0.935	0.789	0.941	0.528	1.000	1.000	1.000	1.000
	WGTD	0.964	0.948	0.929	0.527	1.000	1.000	1.000	1.000
	MUSE	0.972	0.955	0.926	0.209	1.000	1.000	1.000	1.000
Default	w/o WM	0.990	0.934	0.979	0.000	-	-	-	-
	TabularMark	0.987	0.939	0.961	0.004	1.000	1.000	1.000	1.000
	WGTD	0.989	0.913	0.919	0.000	1.000	1.000	1.000	1.000
	MUSE	0.983	0.925	0.963	0.002	1.000	1.000	1.000	1.000
Magic	w/o WM	0.990	0.980	0.998	0.008	-	-	-	-
	TabularMark	0.985	0.975	0.999	0.026	1.000	1.000	1.000	1.000
	WGTD	0.979	0.977	0.998	0.019	1.000	1.000	1.000	1.000
	MUSE	0.991	0.982	0.999	0.010	1.000	1.000	1.000	1.000
News	w/o WM	0.960	0.973	0.811	0.024	-	-	-	-
	TabularMark	0.959	0.969	0.877	0.130	1.000	1.000	1.000	1.000
	WGTD	0.903	0.968	0.861	0.131	1.000	1.000	1.000	1.000
	MUSE	0.959	0.973	0.883	0.033	1.000	1.000	1.000	1.000
Shoppers	w/o WM	0.985	0.974	0.974	0.017	-	-	-	-
	TabularMark	0.974	0.930	0.975	0.013	1.000	1.000	1.000	1.000
	WGTD	0.964	0.944	0.887	0.008	1.000	1.000	1.000	1.000
	MUSE	0.982	0.974	0.950	0.015	1.000	1.000	1.000	1.000

In our implementation, we choose the first numerical column as the selected attribute, and set the number of unit domains $k = 500$, the perturbation range controlled by $p = 25$, and configure n_w as 10% of the total number of rows.

During detection, TabularMark leverages the original unwatermarked table to reverse the perturbations and verify whether the restored differences fall within the green domain. However, **this approach assumes access to the original unwatermarked table**, which is often impractical, especially in scenarios where the watermarked table can be modified by adversaries.

Discussions. As demonstrated in Table 4, both WGTD and TabularMark exhibit strong detection performance across all datasets. Furthermore, their generation quality is generally comparable to that of MUSE. However, a notable observation is the significant performance degradation measured by the MLE metric for both WGTD and TabularMark on the `Beijing` and `News` datasets. We hypothesize that this performance drop stems from the post-editing process, which may introduce substantial artifacts into the data. These artifacts, in turn, could negatively impact the performance of downstream machine learning tasks.

B Further Analysis of the Inversion-Based Watermarking

We first introduce the overall pipeline of inversion-based watermarking in Figure 7. The difficulty lies in the inversion of three components, in sequential order: (1) inverse Quantile Transformation (IQT) §B.2, (2) the VAE decoder §B.3, and (3) the DDIM sampling process §B.4. Finally, we analyze the error accumulation and detection performance across the inversion stages in §B.5.

B.1 Pipeline of Inversion-based Watermarking

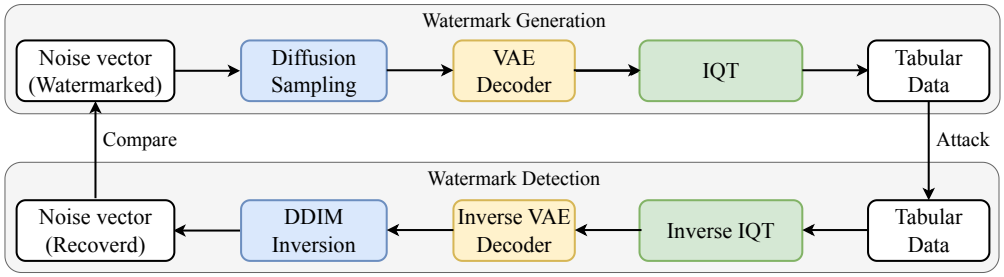

Figure 7: Pipeline of Inversion-based Watermarking. **Top**: The watermark signal is embedded in the noise vector in the latent space, a watermarked table is subsequently generated. **Bottom**: To detect the watermark signal, we need to reverse the entire pipeline. IQT stands for the inverse map of Quantile Transformation.

B.2 Inversion of (Inverse) Quantile Transformation

The Quantile Transformation [Wikipedia contributors, 2025] is a widely used [Zhang et al., 2024c,a, Shi et al., 2024, Kotelnikov et al., 2023] data preprocessing step in tabular data synthesis. It regularizes the data distribution to a standard normal distribution. The Quantile Transformation can be implemented as follows:

1) Estimate the empirical cumulative distribution function (CDF) of the features.
2) Map to uniform distribution with the estimated CDF.
3) Map to standard normal distribution with inverse transform sampling: $z = \Phi^{-1}(u)$, where Φ is the CDF of the standard normal distribution.

Note that in the second step, only the ordering of the data is preserved, and the exact values are not preserved, making the map non-injective, therefore, the inverse of the Quantile Transformation is inherently error-prone. Based on the official codebase, TabWak [Zhu et al., 2025] bypass the inversion of quantile normalization by caching the original data during watermarking, this is infeasible in practical scenarios where the ground truth is unavailable. To study the impact of the inversion error of the Quantile Transformation, we apply the original Quantile Transformation to the sampled tabular data to inverse the inverse quantile transformation.

B.3 Inversion of VAE decoder

Denote the VAE decoder as f_θ, and the VAE decoder output as $\mathbf{x} = f_\theta(\mathbf{z})$. To get $\mathbf{z}$ from $\mathbf{x}$, [Zhu et al., 2025] employ a gradient-based optimization to approximate the inverse of the VAE decoder. Specifically, we can parametrize the unknown $\mathbf{z}$ with trainable parameters, and optimize the following objective with standard gradient descent:

$$\mathbf{z} = \arg\min_{\mathbf{z}} \|\mathbf{x} - f_\theta(\mathbf{z})\|_2^2 .$$

where $\mathbf{z}$ is inilitaized as $g(f_\theta(\mathbf{x}))$, and $g(\cdot)$ is a VAE encoder. However, there is no guarantee that the above optimization will converge to the true $\mathbf{z}$, and we observed that the optimization process is unstable (sometimes produce NaN) for tabular data and introduce significant error in the inversion process.

B.4 DDIM Inversion

The DDIM diffusion forward process is defined as:

$$q(\mathbf{x}_t \mid \mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t \mathbf{I}),$$

where $\mathbf{x}_0$ is the original data, $\mathbf{x}_t$ is the data at time t, and β_t is the variance of the noise at step t. Based on the above definition, we can write $\mathbf{x}_t$ as:

$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\mathbf{x}_{t-1} + \sqrt{1-\bar{\alpha}_t}\epsilon, \qquad \text{(Forward process)}$$

where $\bar{\alpha}_t = \prod_{i=0}^{t}(1-\beta_i)$, $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$.

Starting from $\mathbf{x}_T$, we sample $\mathbf{x}_{T-1}, \ldots, \mathbf{x}_0$ recursively according to the following process:

$$\begin{aligned} \mathbf{x}_0^t &= \left(\mathbf{x}_t - \sqrt{1-\bar{\alpha}_t}\epsilon_\theta(\mathbf{x}_t, t)\right)/\sqrt{\bar{\alpha}_t} \\ \mathbf{x}_{t-1} &= \sqrt{\bar{\alpha}_{t-1}}\mathbf{x}_0^t + \sqrt{1-\bar{\alpha}_{t-1}}\epsilon_\theta(\mathbf{x}_t, t), \end{aligned} \qquad \text{(Reverse process)}$$

where $\epsilon_\theta(\mathbf{x}_t, t)$ is noise predicted by a neural network.

The **DDIM inversion process** is defined as the inverse of the DDIM reverse process. Specifically, starting from $\mathbf{x}_0$, our goal is to recover the original noise vector $\mathbf{x}_T$ in the latent space. We introduce the basic DDIM inversion process proposed in [Dhariwal and Nichol, 2021], and is widely adapted in inversion-based watermark methods [Wen et al., 2023, Yang et al., 2024, Zhu et al., 2025, Hu et al., 2025].

We can abtain the inverse of the DDIM forward process by replacing the $t-1$ subscript with $t+1$ in Equation (Reverse process), but use $\mathbf{x}_t$ to approxiate the unknown $\mathbf{x}_{t+2}$:

$$\mathbf{x}_{t+1} = \sqrt{\bar{\alpha}_{t+1}}\mathbf{x}_0^t + \sqrt{1-\bar{\alpha}_{t+1}}\epsilon_\theta(\mathbf{x}_t, t),$$

Due to the approximation $\mathbf{x}_t \approx \mathbf{x}_{t+2}$, the inversion process generally demands a finer discretization of the time steps. For instance, inversion-based watermarking methods [Wen et al., 2023, Zhu et al., 2025] typically adopt $T = 1000$ steps, whereas diffusion models optimized for fast inference [Karras et al., 2022, Zhang et al., 2024c] often operate with a coarser discretization of $T = 50$ steps.

Advanced Inversion Methods. To address the inexactness of the above inversion process, recent works [Hong et al., 2024, Pan et al., 2023] have proposed more accurate inversion methods based on iterative optimization. However, we empirically found that those methods still suffer from inversion error due to already noisy input from the previous steps (VAE decoder and Quantile Transformation).

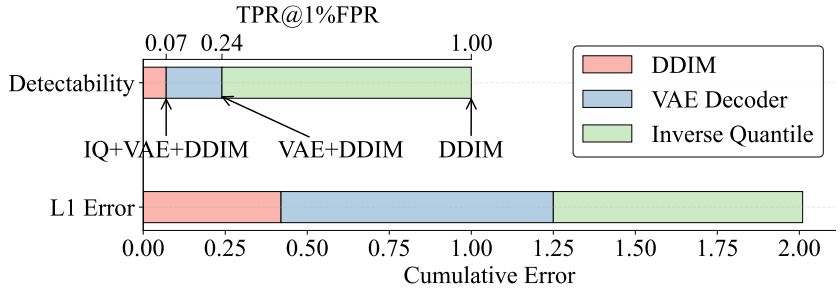

Figure 8: Error Accumulation and Detection Performance Across Inversion Stages of TabWak. The ℓ_1 error is computed between the estimated and ground truth noise vectors in latent space.

B.5 Error Accumulation

In Figure 8, we analyze the error accumulated at each inversion stage and its impact on detection performance using the `Adult` dataset. Specifically, we compute the TPR@1%FPR over 100 watermarked tables, each with 100 rows. The top bar chart shows detection performance when progressively inverting different parts of the pipeline. From left to right:

- When we invert the entire pipeline (IQ $\rightarrow$ VAE $\rightarrow$ DDIM), the detection performance drops to 0.07 TPR@1%FPR.
- When we provide the ground-truth IQ and only invert the VAE decoder and DDIM, the performance improves to 0.24 TPR@1%FPR.
- When both the ground-truth IQ and VAE decoder outputs are provided (i.e., only DDIM is inverted), detection reaches a perfect 1.0 TPR@1%FPR.

The bottom bar chart reports the ℓ_1 error between the estimated and ground-truth noise vectors in the latent space. From left to right, the bars correspond to:

- Inverting only DDIM (given the ground-truth VAE output),
- Inverting both the VAE decoder and DDIM (given the ground-truth IQ), and
- Inverting the full pipeline (IQ $\rightarrow$ VAE $\rightarrow$ DDIM).

This comparison highlights how errors accumulate through the inversion stages and directly affect watermark detectability.

C Experimental Details

C.1 Hardware Specification

We use a single hardware for all experiments. The hardware specifications are as follows:

- GPU: NVIDIA RTX 4090
- CPU: Intel 14900K

C.2 Dataset Statistics

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[2]: Adult[3], Default[4], Shoppers[5], Magic[6], Beijing[7], and News[8], which contains varies number of numerical and categorical features. The statistics of the datasets are presented in Table 5.

Table 5: 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 5, **# 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.

C.3 Fidelity Metrics

The fidelity metrics used in this paper (Marginal, Correlation, C2ST and MLE) are standard metrics in the field of tabualr data synthesis. Here is a reference:

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

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

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

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

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

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

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

- Marginal: Appendix E.3.1 in [Zhang et al., 2024c].
- Correlation: Appendix E.3.2 in [Zhang et al., 2024c].
- C2ST: Appendix F.3 in [Zhang et al., 2024c].
- MLE: Appendix E.4 in [Zhang et al., 2024c].

Below is a summary of how these metrics work.

C.3.1 Marginal Distribution

The **Marginal** metric assesses how well the marginal distribution of each column is preserved in the synthetic data. For continuous columns, we use the Kolmogorov–Smirnov Test (KST); for categorical columns, we use the Total Variation Distance (TVD).

Kolmogorov–Smirnov Test (KST) Given two continuous distributions $p_r(x)$ and $p_s(x)$ (real and synthetic, respectively), the KST measures the maximum discrepancy between their cumulative distribution functions (CDFs):

$$\text{KST} = \sup_x \left| F_r(x) - F_s(x) \right|, \tag{8}$$

where $F_r(x)$ and $F_s(x)$ denote the CDFs of $p_r(x)$ and $p_s(x)$:

$$F(x) = \int_{-\infty}^{x} p(x)\,\mathrm{d}x. \tag{9}$$

Total Variation Distance (TVD) TVD measures the difference between the categorical distributions of real and synthetic data. Let Ω be the set of possible categories in a column. Then:

$$\text{TVD} = \frac{1}{2} \sum_{\omega \in \Omega} \left| R(\omega) - S(\omega) \right|, \tag{10}$$

where $R(\cdot)$ and $S(\cdot)$ denote the empirical probabilities in real and synthetic data, respectively.

C.3.2 Correlation

The **Correlation** metric evaluates whether pairwise relationships between columns are preserved.

Pearson Correlation Coefficient For two continuous columns x and y, the Pearson correlation coefficient is defined as:

$$\rho_{x,y} = \frac{\text{Cov}(x,y)}{\sigma_x \sigma_y}, \tag{11}$$

where $\text{Cov}(\cdot)$ is the covariance and σ denotes standard deviation. We evaluate the preservation of correlation by computing the mean absolute difference between correlations in real and synthetic data:

$$\text{Pearson Score} = \frac{1}{2} \mathbb{E}_{x,y} \left| \rho^R(x,y) - \rho^S(x,y) \right|, \tag{12}$$

where ρ^R and ρ^S denote correlations in real and synthetic data. The score is scaled by $\frac{1}{2}$ to ensure it lies in $[0,1]$. Lower values indicate better alignment.

Contingency Similarity For categorical columns A and B, we compute the Total Variation Distance between their contingency tables:

$$\text{Contingency Score} = \frac{1}{2} \sum_{\alpha \in A} \sum_{\beta \in B} \left| R_{\alpha,\beta} - S_{\alpha,\beta} \right|, \tag{13}$$

where $R_{\alpha,\beta}$ and $S_{\alpha,\beta}$ are the joint frequencies of (α, β) in the real and synthetic data, respectively.

C.3.3 Classifier Two-Sample Test (C2ST)

C2ST evaluates how distinguishable the synthetic data is from real data. If a classifier can easily separate the two, the synthetic data poorly approximates the real distribution. We adopt the implementation provided by the SDMetrics library.[9]

[9] https://docs.sdv.dev/sdmetrics/metrics/metrics-in-beta/detection-single-table

C.3.4 Machine Learning Efficiency (MLE)

MLE evaluates the utility of synthetic data for downstream machine learning tasks. Each dataset is split into training and testing subsets using real data. Generative models are trained on the real training set, and a synthetic dataset of equal size is sampled.

For both real and synthetic data, we use the following protocol:

- Split the training set into train/validation with an $8{:}1$ ratio.
- Train a classifier/regressor on the train split.
- Tune hyperparameters based on validation performance.
- Retrain the model on the full training set using the optimal hyperparameters.
- Evaluate on the real test set.

This process is repeated over 20 random train/validation splits. Final scores (AUC for classification task or RMSE for regression task) are averaged over the 20 trials for both real and synthetic training data. In our experiments, we report the MLE Gap which is the difference between the MLE score of the (unwatermarked) real data and the MLE score of the synthetic data.

C.4 Watermark Detection Metrics

For watermark detection metrics, we primaryly use the area under the curve (AUC) of the receiver operating characteristic (ROC) curve: **AUC**, and the True Positive Rate (TPR) at a given False Positive Rate (FPR): **TPR@x%FPR**.

z-statistic In addition, we can formalize a statistical test for watermark detection for MUSE. Specifically, consider a table T containing N samples (rows) $\mathbf{x}_1, \ldots, \mathbf{x}_N$. Recall that during watermarking, each row is assigned a binary score of 0 or 1 based on a pseudorandom 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:

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

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.

D Ommited Proofs in Section 3

Recall that for a table T (wateramarked or unwatermarked) with N rows: $\mathbf{x}_1, \ldots, \mathbf{x}_N$, we define the watermark detection score as

$$S(T) = \frac{1}{N} \sum_{i=1}^{N} s_k(\mathbf{x}_i), \tag{14}$$

where $s_k(\mathbf{x}_i)$ is the score of the i-th sample, k is the fixed watermark key.

Lemma 3.1. *Denote a watermarked table as T_{wm} and an unwatermarked table as $T_{\text{no-wm}}$, each consisting of N rows. Let $\mathbf{x} \sim p(\mathbf{x})$ be a random variable drawn from the data distribution, and let $\mathbf{x}_1, \ldots, \mathbf{x}_m$ be i.i.d. samples from $p(\mathbf{x})$. Define $\mu_{\text{no-wm}} = \mathbb{E}_{\mathbf{x} \sim p(\mathbf{x})}[s_k(\mathbf{x})]$ as the expected score of an unwatermarked sample, and define $\mu^m_{\text{wm}} = \mathbb{E}_{\mathbf{x}_i \sim p(\mathbf{x})}\left[\max_{i \in [m]} s_k(\mathbf{x}_i)\right]$ as the expected score*

of a watermarked sample obtained via m repeated samples. Suppose the scoring function satisfies $s_k(\cdot) \in [0,1]$, then, the False Positive Rate (FPR) of the watermark detection satisfies:

$$\Pr\left(S(T_{\text{no-wm}}) > S(T_{\text{wm}})\right) \leq \exp\left(-\frac{N(\mu_{\text{wm}}^m - \mu_{\text{no-wm}})^2}{2}\right). \tag{5}$$

Proof. Let $S(T_{\text{no-wm}}) = \sum_{i=1}^N \mathrm{c}_i$ denote the sum of N i.i.d. scores from the unwatermarked table, where each $\mathrm{c}_i = s_k(\mathbf{x}_i)$ for $\mathbf{x}_i \sim p(\mathbf{x})$, and similarly let $S(T_{\text{wm}}) = \sum_{i=1}^N \mathrm{c}'_i$ denote the sum of N i.i.d. scores from the watermarked table, where each $\mathrm{c}'_i = \max\{s_k(\mathbf{x}_{i1}), \ldots, s_k(\mathbf{x}_{im})\}$ with $\mathbf{x}_{ij} \sim p(\mathbf{x})$.

Define the expected values:

$$\mu_{\text{no-wm}} = \mathbb{E}[\mathrm{c}_i], \quad \mu_{\text{wm}}^m = \mathbb{E}[\mathrm{c}'_i].$$

We are interested in bounding the false positive rate:

$$\Pr(S(T_{\text{no-wm}}) > S(T_{\text{wm}})) = \Pr\left(\sum_{i=1}^N (\mathrm{c}_i - \mathrm{c}'_i) > 0\right).$$

Let $\mathrm{w}_i = \mathrm{c}_i - \mathrm{c}'_i$. Since $s_k(x) \in [0,1]$, we have $\mathrm{c}_i \in [0,1]$ and $\mathrm{c}'_i \in [0,1]$, so $\mathrm{w}_i \in [-1,1]$. Moreover, $\mathbb{E}[\mathrm{w}_i] = \mu_{\text{no-wm}} - \mu_{\text{wm}}^m =: -\delta$, where $\delta = \mu_{\text{wm}}^m - \mu_{\text{no-wm}} > 0$.

We apply Hoeffding's inequality to the sum of w_i's:

$$\Pr\left(\sum_{i=1}^N \mathrm{w}_i > 0\right) = \Pr\left(\sum_{i=1}^N \mathrm{w}_i - \mathbb{E}[\sum_{i=1}^N \mathrm{w}_i] > N\delta\right) \leq \exp\left(-\frac{2N^2\delta^2}{4N}\right).$$

Plug in the definition of δ, we have:

$$\Pr(S(T_{\text{no-wm}}) > S(T_{\text{wm}})) \leq \exp\left(-\frac{N^2\delta^2}{2}\right) = \exp\left(-\frac{N(\mu_{\text{wm}}^m - \mu_{\text{no-wm}})^2}{2}\right).$$

which proves the result. □

Lemma 3.2 (Optimal Scoring Distribution)**.** *Let $s_k(\mathbf{x})$ be any random variable supported on $[0,1]$ with mean 0.5, the right-hand-side of Equation (5) is minimized when $s_k(\mathbf{x})$ follows a Bernoulli(0.5) distribution.*

Proof. Let $s_1, \ldots, s_m$ be $i.i.d.$ copies of a random variable $s_k(\mathbf{x}) \in [0,1]$ with fixed mean $\mathbb{E}[s_k(\mathbf{x})] = 0.5$. Define:

$$\mu := \mathbb{E}[s_k(\mathbf{x})] = 0.5, \quad \mu_{\max} := \mathbb{E}[\max(s_1, \ldots, s_m)].$$

Let $\Delta := \mu_{\max} - \mu$ be the gap between the expected maximum score over m repetitions and the mean score. The upper bound in Equation (5) is:

$$\Pr(S_{\text{no-wm}} > S_{\text{wm}}) \leq \exp\left(-\frac{N\Delta^2}{2}\right),$$

so minimizing the FPR corresponds to maximizing Δ under the constraint that $\mathbb{E}[s_k(\mathbf{x})] = 0.5$ and $s_k(\mathbf{x}) \in [0,1]$.

We now show that Δ is maximized when $s_k(\mathbf{x}) \sim$ Bernoulli(0.5).

Step 1: Write $\mu_{\max}$ and μ as integrals over the CDF. Let F be the cumulative distribution function (CDF) of $s_k(\mathbf{x})$. Then the CDF of $\max(s_1, \ldots, s_m)$ is $F^m(x)$. By the tail integration formula, we can compute the expected maximum as:

$$\begin{aligned}\mu_{\max} &= \int_0^1 \Pr(\max(s_1, \ldots, s_m) > x) \\ &= \int_0^1 (1 - F(x)^m)\, dx.\end{aligned}$$

Similarly, we have: $\mu = \int_0^1 (1 - F(x))\, dx$.

Therefore, the gap Δ can be written as:

$$\Delta = \mu_{\max} - \mu = \int_0^1 [F(x) - F(x)^m]\, dx.$$

Step 2: Leverage the concavity. By Lemma E.1, the integrand $F(x) - F(x)^m$ is concave in $F(x)$. By Lemma E.2, the integral is maximized when $F(x)$ is the CDF of a Bernoulli distribution with mean $\mu = 0.5$.

Therefore, among all $s_k(\mathbf{x}) \in [0, 1]$ with $\mathbb{E}[s_k(\mathbf{x})] = 0.5$, the Bernoulli(0.5) distribution maximizes Δ, which minimizes the upper bound on the FPR. Hence, the lemma holds. □

Theorem 3.3 (Minimum Watermarking Signal)**.** *Under the same assumptions as in Lemma 3.1, suppose the scoring function $s_k(\mathbf{x})$ is instantiated as a hash-seeded pseudorandom function such that $s_k(\mathbf{x}) \sim$ Bernoulli(0.5), the FPR is upper-bounded by:*

$$\Pr\left(S(T_{\text{no-wm}}) > S(T_{\text{wm}})\right) \leq \exp\left(-\frac{N}{2}\left(0.5 - 0.5^m\right)^2\right). \tag{6}$$

To ensure the FPR does not exceed a target threshold α, it suffices to set the number of repeated samples m as:

$$m = \max\left(2, \left\lceil \log_{0.5}\left(0.5 - \sqrt{\tfrac{2\log(1/\alpha)}{N}}\right)\right\rceil\right), \tag{7}$$

where $\lceil \cdot \rceil$ denotes the ceiling function. This expression is valid when $N > 8\log(1/\alpha)$.

Proof. When $s_k(\mathbf{x}) \sim$ Bernoulli(0.5), we have:

$$\mu_{\text{no-wm}} = \mathbb{E}[s_k(\mathbf{x})] = 0.5, \quad \mu_{\text{wm}}^m = \mathbb{E}[\max(s_1, \ldots, s_m)] = 1 - 0.5^m.$$

Plug in into the FPR bound Equation (16), we have:

$$\Pr\left(S(T_{\text{no-wm}}) > S(T_{\text{wm}})\right) \leq \exp\left(-\frac{N}{2}\left(0.5 - 0.5^m\right)^2\right),$$

which completes the proof. □

E Technical Lemmas

Lemma E.1. *For any integer $m \geq 2$, the function $f(x) = x - x^m$ is concave on the interval $[0, 1]$.*

Proof. To prove that $f(x) = x - x^m$ is concave on $[0, 1]$, we show that its second derivative is non-positive on this interval.

Compute the first derivative:

$$f'(x) = \frac{d}{dx}(x - x^m) = 1 - mx^{m-1}.$$

Compute the second derivative:

$$f''(x) = \frac{d}{dx}(1 - mx^{m-1}) = -m(m-1)x^{m-2}.$$

Observe that for all $x \in [0, 1]$ and $m \geq 2$: $m(m-1) > 0$ and $x^{m-2} \geq 0$.

Therefore,

$$f''(x) = -m(m-1)x^{m-2} \leq 0 \quad \text{for all } x \in [0, 1].$$

Hence, $f(x)$ is concave on $[0, 1]$. □

Lemma E.2. *Let $\phi : [0,1] \to \mathbb{R}$ be a concave function, and let F be the cumulative distribution function (CDF) of a random variable supported on $[0,1]$ with fixed mean $\mu \in (0,1)$. Then the integral*

$$\int_0^1 \phi(F(x))dx$$

is maximized when $F(x) = \begin{cases} 0 & \text{if } x < 0 \\ 1-\mu & \text{if } 0 \le x < 1 \\ 1 & \text{if } x \ge 1 \end{cases}$, i.e. the CDF of a Bernoulli distribution with mean μ.

Proof. **Step 1: Rewrite the Mean Constraint**

By the tail integration formula, the mean constraint for the random variable X with CDF $F(x)$ supported on $[0,1]$ is:

$$\int_0^1 (1 - F(x))\, dx = \mu.$$

Rearranging this equation gives the integral of $F(x)$:

$$\int_0^1 F(x)\, dx = 1 - \mu. \tag{15}$$

Step 2: Upper Bound the Integral

The function $\phi : [0,1] \to \mathbb{R}$ is concave. The CDF $F(x)$ takes values in $[0,1]$ for $x \in [0,1]$, so $\phi(F(x))$ is well-defined. We can apply Jensen's inequality for integrals, which for a concave function ϕ and an integrable function $g(x)$ on an interval $[a,b]$ states:

$$\frac{1}{b-a}\int_a^b \phi(g(x))\, dx \le \phi\left(\frac{1}{b-a}\int_a^b g(x)\, dx\right).$$

Plug in $a = 0$, $b = 1$, $g(x) = F(x)$. Jensen's inequality then becomes:

$$\int_0^1 \phi(F(x))\, dx \le \phi\left(\int_0^1 F(x)\, dx\right).$$

Substituting Equation (15) into the right hand side, we have:

$$\int_0^1 \phi(F(x))\, dx \le \phi(1-\mu). \tag{16}$$

Step 3: Verify $F(x)$ achieves the upper bound

It is straightforward to verify that $F(x)$ satisfies the mean constraint. Next, we will show that $F(x)$ achieves the upper bound $\phi(1-\mu)$. For $x \in [0,1)$, $F(x) = 1 - \mu$. Therefore, we have:

$$\int_0^1 \phi(F(x))\, dx = \int_0^1 \phi(1-\mu)\, dx = \phi(1-\mu).$$

We have shown that $F(x)$ satisfies the mean constraint and achieves the upper bound $\phi(1-\mu)$, which completes the proof. □

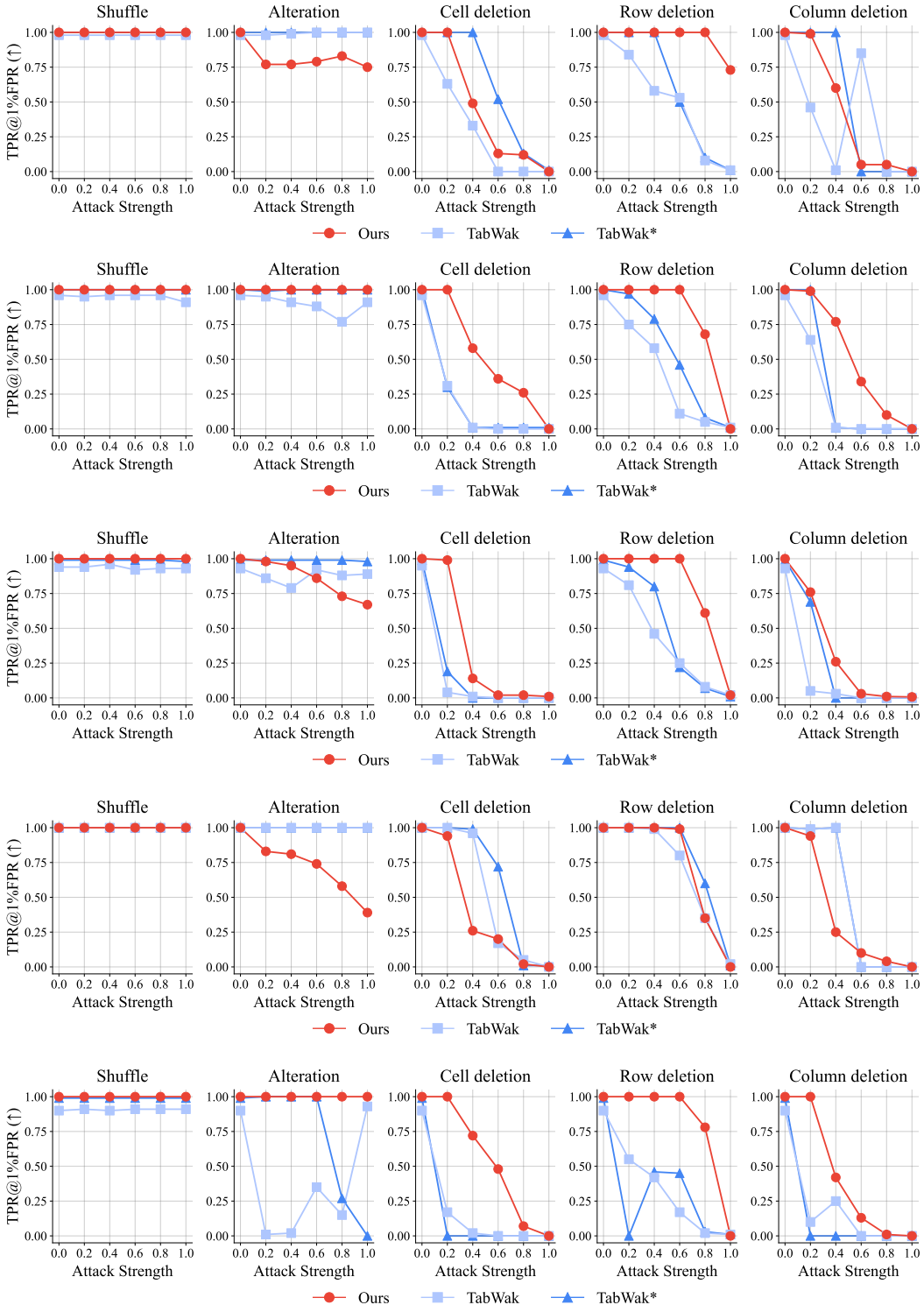

Figure 6: Detection performance of MUSE vs. TabWak/TabWak* against different types of tabular data attacks across varying attack intensities. From top to bottom: `Beijing`, `Default`, `Magic`, `News` and `Shoppers`.