Ripple Watermarking for Latent Tabular Diffusion Models

by

Jiayi Tang

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Monday June 17th, 2024 at 12:30 PM.

Student number: 5817714
Thesis committee: Dr. Lydia Chen, TU Delft, supervisor
Dr. Avishek Anand, TU Delft

An electronic version of this thesis is available at `http://repository.tudelft.nl/`.

Preface

The generation of synthetic data through advanced generative models represents a transformative method for enhancing and sharing data. While watermarking techniques have been widely developed for synthetic images and texts, their application to synthetic tables has been limited. To addresses the essential need for traceability and auditability in synthetic tabular data to prevent misuse and potential harm, this work introduces `Ripple`, an innovative watermarking algorithm specifically designed for latent tabular diffusion models, bridging a significant gap in the application of watermarking techniques to synthetic tabular data.

`Ripple` is a novel watermarking algorithm tailored for tabular diffusion models during the sampling phase. By embedding watermarks in the Fourier space of the initial noise matrix, `Ripple` effectively balances data quality, detectability, and robustness. This approach ensures the watermark remains invisible to human observers while being detectable by algorithms and resilient to various post-editing operations.

Extensive evaluation across four diverse datasets demonstrates `Ripple`'s effectiveness in preserving synthetic data quality in terms of resemblance, discriminability, and downstream utility. The average quality difference between watermarked and non-watermarked data is less than 0.6%, ensuring the synthetic tables remain useful for machine learning tasks and other applications. In terms of detectability, `Ripple` delivers outstanding results, with statistical p-values for watermark detection averaging more than 25 times lower than 0.02 across all datasets, representing an improvement of more than 2 orders of magnitude compared to the tree-ring baseline. Additionally, `Ripple` demonstrates strong robustness against various post-editing attacks, including the deletion and distortion of rows, columns, and values. Robustness analysis reveals that 85% of the p-values remain below 0.05 across different attack scenarios, highlighting `Ripple` 's capability to maintain watermark integrity under malicious conditions.

By successfully embedding watermarks while preserving data quality and ensuring high detectability and robustness, `Ripple` sets a new benchmark for traceability and auditability in synthetic tabular data. This contribution is significant as the use of synthetic data continues to expand in fields such as healthcare, finance, and social sciences.

This master's thesis is completed under the dedicated supervision of Prof. Lydia Chen, with valuable guidance and support from Prof. Robert René Maria Birke and PhD student Chaoyi Zhu. Their academic expertise significantly contributed to the depth and rigor of this work. I am also deeply grateful for their warm patience and assistance, which have been instrumental in my personal and professional development. Additionally, I extend my sincere gratitude to Prof. Avishek Anand, for his support as my committee member, and to my family for their unwavering support throughout this journey.

Jiayi Tang
Delft, June 2024

Contents

1

Research Paper

RIPPLE WATERMARKING FOR LATENT TABULAR DIFFUSION MODELS

ABSTRACT

Synthetic tabular data generated by tabular generative models represent an effective means of augmenting and sharing data. It is of paramount importance to trace and audit such synthetic data, avoiding potential harms and risks associated with inappropriate usage. While watermarking techniques are increasingly used for synthetic images, little is known about how to watermark synthetic tables such that they are imperceptible for humans, detectable by algorithms, and robust against post-editing. In this paper, we present the first watermarking algorithm for tabular diffusion models, which inserts novel ripple watermarks into the latent space of tables. For every synthetic table, the watermark initiates from a central ring within the Fourier-transformed latent of the table, extending gradually across a large portion of the space. The watermark can be detected by calculating the distance between the Fourier-transformed tabular latent and the ground-truth watermark patch. Additionally, we develop post-editing attacks, including row/column/value deletion and distortion, to evaluate the robustness of the watermark. Our evaluation on four datasets demonstrates that our watermarking scheme effectively preserves the quality of synthetic tables in terms of resemblance, discriminability, and downstream utility. The average quality difference is less than 0.6% compared to non-watermarked data, while maintaining high detectability, with average statistical p-values over $25\times$ lower than 0.02. Additionally, our robustness analysis shows that the watermark is resilient against various post-editing actions, with 85% of the p-values remaining below 0.05 across all 18 attack settings on four datasets.

1 INTRODUCTION

Synthetic data from generative models is becoming integral to today's data management and artificial intelligence services. Synthetic tables generated from tabular generative adversarial networks [55, 47] and tabular diffusion models [23] are used to augment the data for training machine learning models and substitute the original data for protecting privacy [13]. As such synthetic data is increasingly adopted for critical tasks, it is paramount to ensure its traceability and auditability to avoid harm and misusages. Recent advancements in watermarking technology [22, 24, 45, 56, 48] have demonstrated significant promise in texts from language models and images from diffusion models. Research indicates that embedded watermarking keys do not compromise the quality of synthetic data, remaining imperceptible to human users while being detectable by algorithms. Besides balancing data quality and detectability, another essential criterion for watermarking technologies is their robustness against post-editing operations, such as deletions and insertions [24].

Existing studies on image and language generative models focus on embedding watermarking keys during the training [11], sampling [45, 22], and post-editing phases [44, 3, 15]. Watermarking during the training phase involves modifying the model weights, which enhances the trade-off between quality and robustness but is time-consuming [11]. Conversely, post-editing watermarking methods, which apply watermarks directly to existing data, significantly degrade data quality and are susceptible to removal or theft by third parties. Sampling-phase watermarking, which alters only the sampling process without changing the model weights, yet maintains high data quality [45, 22] and offers a favorable trade-off between computational overhead and robustness. This approach has garnered significant attention due to its efficiency and effectiveness. In the context of language models like GPT, secret keys [22] are used to modify the logit values of vocabulary tokens, thereby adjusting token probabilities for the next-word generation according to the context and keys. For image mod-

els, watermarking is proposed to be embedded in the latent space of image diffusion models [45]. To minimize the distortion to the image quality, Fourier transformation is first applied to the latent space to identify the low-frequency area of latent space that has a lower impact on perceived quality. Despite substantial research on watermarking synthetic texts and images, there is, unfortunately **no study on watermarking tabular generative models during the sampling phase**.

Synthetic tables can be generated using a variety of models, including Bayesian networks [2], variational autoencoder networks [1], generative adversarial networks [47, 55], and diffusion models [52, 23]. Prior research often addresses the challenge of representing discrete column variables, such as gender and education levels, through advanced encoding schemes and network architectures. One-hot encoding is a common method for representing these categorical variables [55, 47]. Tabular-specific encoding schemes [1] unify the representation of continuous and categorical variables in a single space, achieving scalability yet they suffer from poor scalability. Recently, language models have been applied to table generation [6] due to their advantage in generalized encoding across text, numerical, and discrete columns. However, the large model sizes and computational demands of language models limit their practicality for tabular generation, which is inherently structured and requires fewer vocabulary tokens than natural language. Consequently, tabular generative models inspired by image generative models are more commonly adopted in industry [10], offering a better balance between computational overhead and data quality, i.e., the similarity between real and synthetic tables and the differences in downstream utility between using real and synthetic tables.

In this paper, we propose the first watermarking scheme, `Ripple`, for tabular generative models in the sampling phase. The particular type of generative model considered is the latent tabular diffusion model [52] that first encodes all types of column variables into a unified space via autoencoder networks and then uses diffusion models to synthesize the latent codes. `Ripple` embeds watermark keys in the Fourier space of the initial latent noise matrix that is used to generate a new table. Watermarking in the Fourier space enhances coherence, as changes spread more evenly into the spatial domain, and improves robustness, as attacks in the spatial domain are less effective in the Fourier space. Existing tree-ring watermarking method [45], which embeds watermarks in the Fourier space of images, is limited when applied to tables due to the different shapes of tables. This method only watermarks the central low-frequency domain constrained by its employed radius for images. In contrast, `Ripple` embeds a ripple pattern across a broader frequency spectrum, ensuring that many more cells carry watermark keys, thus improving detectability. Additionally, `Ripple` maintains good data utility by embedding the watermark in the noise matrix in the latent space, minimizing the impact on the actual data. Another key component of `Ripple` is its watermark detection algorithm. Given a synthetic table, the invertibility of diffusion models allows the table to be reversed to its noise matrix. Detectability is then evaluated using statistical p-values, which describe the distributional differences of the distances in the Fourier domain between the ground-truth patch and the reversed noise matrices from watermarked and non-watermarked data.

We evaluate `Ripple` on four datasets with synthetic tables generated by TabSyn [52] for data quality comparison, and we further watermark these synthetic tables with tree-ring [45] and `Ripple` respectively for watermark detectability comparison. `Ripple` achieves data quality comparable to the original synthetic data in terms of shape quality, trend quality, discriminability, and machine learning efficacy. Additionally, `Ripple` demonstrates significantly higher detectability, measured by lower p-values across all four datasets. To assess the robustness of `Ripple`, we develop six post-editing attacks: deletion and distortion at the row, column, and value levels. Among these, row deletion is the most effective, reducing detectability by a significant margin on two of the four datasets. Lastly, we conduct ablation studies to examine the impact of the watermarking area, watermarking target, and attacking target. The results indicate that the combined use of the diffusion model and variational autoencoder enhances data quality, watermark detectability, and watermark robustness with the area to be watermarked being a hyperparameter to tune.

Our key contribution lies in designing and validating `Ripple`, the first watermarking scheme for latent tabular diffusion models. The specific technical contributions are summarized as follows:

- The novel ripple watermarking pattern enables watermarking across cells in the synthetic table, significantly increasing detectability by more than 2 orders of magnitude compared to the tree-ring watermarking baseline.
- Six post-editing attacks specific to synthetic tables are developed for robustness evaluation, involving the deletion and distortion of columns, rows, and cells.

- Comprehensive empirical evidence demonstrates that `Ripple` achieves three critical goals: i) maintaining high synthetic data quality with an average difference of less than 0.6% in quality measures despite the inclusion of the watermark, ii) ensuring high watermark detectability across all datasets with p-values on average more than 25 times lower than 0.02, and iii) ensuring high robustness with 85% of the p-values remaining under 0.05 in the presence of different settings of post-editing attacks.

2 Related Works

2.1 Watermarks on synthetic data

With the ability to create contents that mimic human creativity, generative AI models have achieved notable proficiency in generating high-fidelity images, videos, texts and more [7, 29]. However, this progress also introduces challenges, notably the potential for misuse, such as deepfakes and misinformation enabling fraud and scams [35, 14, 19]. To control potential misuse and risks, watermarking across various data modalities has been proposed as an effective strategy by embedding hidden messages into all generated content that are later detectable to enhance traceability.

In image synthesis, watermarking can be integrated into generative models by modifying their training procedures. This approach involves embedding a watermark into the training data, ensuring that the generated images inherently contain the watermark [50, 51, 54]. Alternatively, watermarking can be implemented without requiring watermarked training data or retraining the generator from scratch. Pivotal Tuning Watermarking [26] offers a method for watermarking pre-trained GANs by adjusting the models during post-training. Methods that use the invertibility of diffusion models [45] or employ additional encoders and decoders to embed a watermark message-matrix [46] can also be used for watermarking without retraining the generative model.

In text synthesis, watermarking for large language models (LLMs) has seen significant development. One approach involves embedding watermark messages into a subset of the training data using a watermark embedding function [42, 41]. However, this method requires retraining the model, and the model generates watermarked outputs only for specific inputs. Another approach is watermarking during the logits generation, where the probability distribution of the next predicted token is modified using algorithms based on previous tokens. The watermark is detected based on the partitions in the modified distributions [22, 56]. Alternatively, watermarking can be done during the token sampling phase, where the logits for the next token are not altered, but the watermark message guides the sampling of tokens. This can be implemented at the word level [24] or sentence level [17].

Watermarking has also been applied to other modalities of synthetic data. For instance, RivaGAN [53] watermarks videos by training a deep neural network adversarially to stamp a pattern on synthetic images using a custom attention-based mechanism. Additionally, an approach that predefines a text-image pair and fine-tunes the model allows for watermarking large-scale text-to-image models without starting from scratch [54]. These diverse watermarking techniques ensure the authenticity and integrity of AI-generated content, addressing potential misuse and preserving trust in digital media. While little attention has been paid to synthetic tabular data, we extend watermarking to latent tabular diffusion models with the proposed `Ripple`, which achieves high synthetic data quality and superior watermark detectability while demonstrating remarkable resilience to post-editing attacks.

2.2 Diffusion Models

Diffusion models, initially introduced by Sohl-Dickstein et al. [37], have gained considerable attention in recent years owing to their exceptional performance across a diverse array of domains. These domains include image synthesis [16, 20], text-to-image generation [32, 33, 31], spatial-temporal data modeling [43], tabular data synthesis [23, 52], and more. Based on their underlying assumptions and optimization objectives, the diffusion models can be broadly categorized into variational diffusion models and score-based models [27].

Variational diffusion models operate by employing two Markov chains: a forward process that perturbs the data into pure Gaussian noise, and a reverse process that learns to recover the data from this noise. These models are parameterized by neural networks that either aim to predict the original

data from the noised data at various timesteps [37] or learn to predict the source noise influencing the original data to the noised data at different timesteps [38].

In contrast, instead of the use of Markov chains, score-based models [39] originate from the concept of the Stein score [18], also known as the score function. The score function is defined as the gradient of the log probability density of a given probability distribution. Thus, given the original data, score-based models perturb the data with a sequence of intensifying Gaussian noise and simultaneously estimate the score functions for all noisy data distributions by training a deep neural network model conditioned on noise levels.

These foundational diffusion models have been further refined through various enhancements, leading to models with more efficient sampling methods [9, 40, 20], improved likelihoods [28, 25] and cross-modality integration [12, 30]. These developments enable diffusion models to excel in generating high-quality synthetic data that closely mimic real datasets, making it challenging to distinguish between the two. Consequently, it is crucial to develop advanced watermarking techniques that can effectively identify AI-generated content. This necessity underscores our approach, which integrates robust watermarking into the diffusion process, ensuring the traceability and authentication of synthetic data while maintaining its quality and usability.

3 METHODS

3.1 LATENT TABULAR DIFFUSION MODEL

Latent tabular diffusion models are state-of-the-art tabular data generative models [52, 36]. In this paper, we use them as the foundation to insert invisible watermarks in the synthesized tables. Indeed, their latent space provides a safe environment where watermarks can be seamlessly injected and detected since it requires access to the trained model, which we assume only the owner has. State-of-the-art tabular diffusion models consist of a diffusion model, generating a latent variable, and an autoencoder, decoding the latent variable into synthetic tabular data [52].

Autoencoder The autoencoder allows to unify the representation of continuous and categorical columns. Further, it adds a layer of disguise for watermarking, making the watermark pattern less discernable in the synthesized table.

The encoder $\mathcal{E}$ initially maps the original tabular data denoted as X, both continuous and categorical columns, into a unified, continuous latent space, represented as $Z = \mathcal{E}(X)$. Subsequently, the decoder $\mathcal{D}$ reconstructs the latent representation Z back into the original data space, yielding $\tilde{X} = \mathcal{D}(Z)$. Following the state-of-the-art work [52], we use variational autoencoders that can be trained by minimizing the negative evidence lower-bound (ELBO) [21].

Diffusion model The diffusion model denoises a latent representation of the tabular data from a noise matrix, which can be infused with a watermark and later detected using the ground-truth watermark patch. While the architecture is oblivious to the specific choice of autoencoder architecture, the choice of diffusion model requires careful consideration to guarantee deterministic diffusion and sampling processes. Ensuring both deterministic processes allows for accurate recovery of the noise matrix from the synthesized table, thereby enabling sound detection of the watermark.

Among the various diffusion models, Denoising Diffusion Implicit Model (DDIM) stands out for its ability to facilitate both deterministic diffusion and sampling processes. DDIM extends the classical Markovian diffusion process into a broader class of non-Markovian diffusion processes. Within the DDIM framework, given the noise matrix Z_T in the latent space, and a neural network ϵ_θ that predicts the noise $\epsilon_\theta(t, Z_t)$ at each diffusion time step t, the generation of a sample Z_{t-1} from Z_t during the sampling process is described by the equation:

$$Z_{t-1} = \sqrt{\alpha_{t-1}}\left(\frac{Z_t - \sqrt{1-\alpha_t}\epsilon_\theta(t, Z_t)}{\sqrt{\alpha_t}}\right) + \sqrt{1-\alpha_{t-1}-\sigma_t^2}\cdot\epsilon_\theta(t, Z_t) + \sigma_t\epsilon_t \tag{1}$$

where $\alpha_1, \ldots, \alpha_T$ are computed from a predefined variance schedule, $\epsilon_t \sim \mathcal{N}(0, I)$ denotes standard Gaussian noise independent of Z_t, and the σ_t values can be varied to yield different generative

processes. Specifically, by setting σ_t to 0 for all t, the sampling process becomes deterministic:

$$Z_{t-1} = \sqrt{\frac{\alpha_{t-1}}{\alpha_t}} Z_t + (\sqrt{1-\alpha_{t-1}} - \sqrt{\frac{\alpha_{t-1}}{\alpha_t} - \alpha_{t-1}})\epsilon_\theta(t, Z_t) \tag{2}$$

This deterministic sampling process ensures that a given noise matrix Z_T consistently generates the same latent matrix Z_0. Consequently, when Z_0 is fed into the decoder $\mathcal{D}$, the resulting table $\tilde{X} = \mathcal{D}(Z_0)$ will also be consistently the same.

Notably, in the limit of small steps (large value of T), we can traverse the timesteps in the reverse direction towards increasing levels of noise, yielding a deterministic diffusion process from Z_0 to Z_T:

$$Z_{t+1} = \sqrt{\frac{\alpha_{t+1}}{\alpha_t}} Z_t + (\sqrt{1-\alpha_{t+1}} - \sqrt{\frac{\alpha_{t+1}}{\alpha_t} - \alpha_{t+1}})\epsilon_\theta(t, Z_t) \tag{3}$$

Therefore, given the tabular latent $Z_0 = \mathcal{E}(X)$ of a table X, the noise matrix Z_T that is used to sample the corresponding table can be derived. This latent tabular diffusion model with deterministic sampling and diffusion processes enables the secure watermarking of synthetic tabular data. By embedding the watermark into the noise matrix Z_T, the watermark remains imperceptible to humans and exerts minimal influence on the quality of the synthetic tables. By reversing the tabular data back to the noise matrix, the watermark's presence can be smoothly detected by assessing the distance between the ground-truth watermark patch and the reversed noise matrix $\tilde{Z}_T$.

3.2 Ripple watermarking

3.2.1 Watermarking in the Fourier domain

As Section 3.1 mentions, the watermark is embedded within the noise matrix inputted to the diffusion model. The deterministic nature of our latent diffusion model suggests that embedding the watermark directly into the latent noise matrix could introduce discernible patterns in the sampled tabular latent, which may subsequently manifest as noticeable patterns in the generated tables.

To mitigate this, we adopt a method inspired by the Tree-Ring watermarking technique [45]. Instead of embedding the watermark directly into the noise matrix, we first apply a fast Fourier transform (FFT) to the matrix and then embed the watermark in the Fourier domain. Watermarking in the Fourier domain offers two significant advantages over watermarking in the original spatial domain.

First, it enhances the imperceptibility of the watermark. Modifications in the Fourier domain are more evenly distributed when transformed back to the spatial domain, allowing the watermark to be spread across the entire matrix. This distribution minimizes the risk of noticeable artefacts that could arise from embedding the watermark directly in the spatial domain. Second, watermarking in the Fourier domain increases resistance to common spatial domain attacks such as cropping and scaling. These operations typically have a less pronounced impact on the frequency components of a matrix in its Fourier domain, thereby preserving the robustness of the watermark.

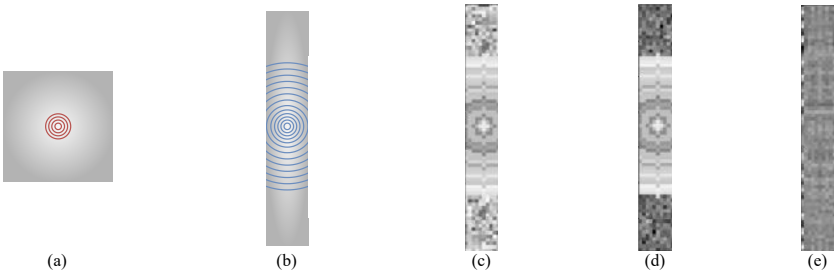


Figure 1: (a) The tree-ring watermark. (b) The ripple watermark. (c) The ground-truth ripple watermark patch. (d) The ripple watermarked FFT transformed noise matrix. (e) The resulting watermarked noise matrix.

The tree-ring watermark is specifically designed for images, taking into account the inherent characteristics of image data. In the context of image synthesis, the dimensions of an image are typically

square, such as a 256×256 matrix. The tree-ring watermark is centrally positioned and resembles the concentric rings of a tree, as illustrated in Figure 1 (a).

In contrast, tabular data is typically generated with a significantly larger number of rows than columns, resulting in a tall rectangular shape, for example, a 10000×40 matrix. Embedding a small, centrally positioned tree-ring watermark in such a tall rectangle would render the watermark undetectable. Another critical distinction between image and tabular watermarking is that image watermarking applies to individual data samples, that is, a single image. On the other hand, tabular data is often generated in batches containing multiple data samples, that is, multiple rows. Consequently, we propose the ripple watermark, characterized by circles spreading across the rows, akin to a ripple effect, watermarking a batch of rows at a time, as depicted in Figure 1 (b).

Specifically, given a predefined radius r representing the outermost circle of the ripple watermark, a watermark patch K with the ripple originating from its center is generated using Algorithm 1. Initially, a random matrix of the same size as the noise matrix is generated using Gaussian noise. Its 2D Fourier transform with the zero-frequency component shifted at the center, serves as the base for the watermark patch. Subsequently, a ripple is created wherein each concentric circle of the ripple has the same value, which is randomly sampled from the base matrix. The value of r is determined empirically according to different datasets.

Algorithm 1 Generate watermark patch

Input: Radius r, shape (n, m) of the noise matrix where n is the number of rows and m is the dimension of the latent for each row
$N \leftarrow (N_{ij} \sim \mathcal{N}(0,1))_{1 \le i \le n, 1 \le j \le m}$ // Initialize a random Gaussian matrix
$K \leftarrow \text{fftshift}(\text{fft2d}(N))$ // Apply 2D Fourier transform to N and shift the components
$K_{\text{tmp}} \leftarrow \text{copy}(K)$ // Copy K for value sampling in ripple circles
for $k \leftarrow r$ **downto** 1 **do**
 // From the outermost circle inward
  $v \leftarrow \text{sample}(K_{\text{tmp}})$ // Sample a random value v for the i-th circle
  for all (i, j) where $1 \le i \le n, 1 \le j \le m, (i - \frac{n}{2})^2 + (j - \frac{m}{2})^2 \le k^2$ **do**
    $K(i, j) \leftarrow v$ // Assign v to position (i, j)
  end for
end for
return K // Return the modified matrix

An example of the generated watermark patch is shown in Figure 1 (c). Plots of the watermarked Fourier-transformed noise matrix and its inverse fast Fourier transform (IFFT), which is used to sample synthetic tabular data, are presented in Figure 1 (d) and (e). As previously mentioned, modifications in the Fourier domain spread out more evenly when transferred back to the spatial domain. Therefore, the central ripple in the Fourier domain would impart a watermarking effect across all rows and columns in the spatial domain. Furthermore, as illustrated in the figure, while the ripple watermark is readily detectable in the Fourier domain, it does not exhibit an obvious pattern in the spatial domain. The IFFT noise matrix resembles a Gaussian noise matrix, making it suitable for sampling via the diffusion model.

3.2.2 The injection and detection of watermarking

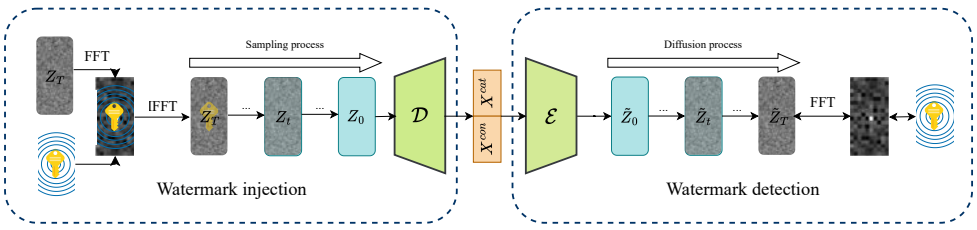


Figure 2: The injection and detection phases of ripple watermarking.

The watermarking injection and detection phases are depicted in Figure 2. In the injection phase, specific matrix elements in the Fourier-transformed noise matrix are replaced by values from a predefined watermark patch. Specifically, given the predefined watermark patch K and a binary mask M with values of 1 in the ripple watermark region and 0 otherwise, the FFT-transformed noise matrix $F(Z_T)$ of an initial noise matrix Z_T is watermarked as follows:

$$F(Z_T)_{i,j} = \begin{cases} K_{i,j}, & \text{if } M_{i,j} = 1 \\ F(Z_T)_{i,j}, & \text{otherwise} \end{cases} \quad \text{where } M_{i,j} = \begin{cases} 1, & \text{if } (i - \frac{n}{2})^2 + (j - \frac{m}{2})^2 \leq r^2 \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

The FFT-transformed noise matrix, which is watermarked, will then be applied with an inverse fast Fourier transform (IFFT). Subsequently, a synthetic table incorporating the watermark is generated through the diffusion model's sampling process and the decoder's decoding process.

In the detection phase, the synthetic tables are encoded back into the latent space and diffused back to the noise matrix $\tilde{Z}_T$. An FFT is applied again, and the detection process is based on the L_1 distance between the ground-truth watermark patch K and the watermarked area of the Fourier transformed latent noise matrix $F(\tilde{Z}_T)$. The distance metric is defined as:

$$d = \frac{1}{|M|} \sum_{M_{i,j}=1} \left| K_{i,j} - F(\tilde{Z}_T)_{i,j} \right| \tag{5}$$

The watermark is then detected if the calculated distance d is less than a threshold τ, which is empirically calibrated for different datasets.

4 Evaluation

4.1 Experimental setup

Datasets We used four widely utilized tabular datasets, including two small and two larger datasets, to evaluate the influence of the proposed watermarking technique on synthetic data quality, the effectiveness of watermark detection, and its robustness against post-editing attacks. The **Shoppers** [34] includes 12,330 samples featuring 18 mixed-type columns (10 continuous and 8 categorical) that capture online shoppers' purchasing intentions. The **Magic** [5] dataset is designed for simulating the registration of high-energy gamma particles and consists of 19,020 instances with 11 columns (10 continuous and 1 categorical). The **Credit** [49] dataset provides data on the default payments of credit card clients, comprising 30,000 instances with a total of 24 mixed-type columns (14 continuous and 10 categorical). Finally, the **Adult** [4] dataset contains information on individuals' annual incomes, consisting of 48,842 instances with 15 mixed-type columns (6 continuous and 9 categorical).

Tabular generative model All experiments are conducted using a consistent latent tabular model architecture. Watermarking is exclusively applied during the sampling processes; consequently, each model requires training only once per dataset. The trained model can then be sampled multiple times with various watermarked noise matrices to assess the watermark's efficacy. Specifically, the autoencoder module comprises an encoder and a decoder, each following a 2-layer Transformer architecture. The hidden dimension of the Transformer's feed-forward network (FFN) is set to 128. The diffusion model comprises a 4-layer multi-layer perceptron (MLP) with a hidden dimension of 1024. For both the diffusion and sampling processes within the diffusion model, 1000 timesteps are used. With these hyperparameters, the latent tabular model consistently generates high-quality data in the absence of watermarking, achieving similarity metrics above 0.88, discriminability metrics above 0.63, and utility metrics around 0.79 across all datasets. Therefore, the same architecture is employed for all four datasets, while the number of training epochs is tuned for each dataset individually.

Baselines Since our watermarking technique is applied during the sampling process, the baselines differ in the initialization of noise matrices for sampling tabular data. Two baselines are established: the non-watermarked baseline and the state-of-the-art tree-ring (TR) baseline [45]. The

non-watermarked baseline, referred to as W/O in the tables, uses a random Gaussian noise matrix without modifications. Data sampled from this baseline are used to evaluate the impact of watermarking on synthetic data quality. The **tree-ring** baseline involves injecting the noise matrix with a tree-ring watermark (TR_{rings} from [45]), where the injection is the same as the injection of the ripple watermark. However, unlike a broadly spread ripple watermark, the tree-ring watermark forms a complete tree-ring shape that extends to the matrix's edge. Specifically, given a noise matrix of size (n, m) where n is much larger than m, this baseline is applied with the largest possible tree-ring, meaning that for each dataset, the outermost radius of the tree-ring watermark is $m/2$. The tree-ring baseline is used for both data quality and watermark detection comparison. This allows us to analyze how different watermark types influence the synthesis quality and the effectiveness of watermark detection.

4.2 DATA QUALITY

The quality of the synthetic data is evaluated on three aspects: similarity, discriminability, and utility.

Similarity This aspect assesses the statistical similarity between the synthetic data and the original data through the following metrics:

- **Shape quality**: This metric evaluates resemblance by comparing the distribution of each column in the synthetic data with its counterpart in the original data. A similarity score is computed for each column based on the distributions in both datasets. In particular, we use the complement of the Kolmogorov-Smirnov statistic for continuous columns and of the total variation distance for categorical columns, respectively [8]. The average of these scores across all columns reflects the shape quality of the synthetic table. A higher score indicates a greater similarity between the synthetic and original tables.
- **Trend quality**: Since columns in the data may or may not relate to each other, this metric assesses how well the synthetic data preserves these relationships. This involves calculating the correlation between every pair of columns in both the synthetic and original datasets and comparing them. A higher trend quality score indicates that the synthetic data accurately represents the inter-column relationships found in the original data. The average of these scores across all column pairs is used to represent the trend quality of the synthetic table.

Discriminability This aspect assesses how difficult it is for a machine learning model to distinguish between synthetic data and original data through the following method:

- **Logistic Detection**: A logistic regression model is trained to differentiate between the two datasets. First, all rows from both the real and synthetic datasets are combined and then split into training and validation sets. The machine learning model is trained on the training set and evaluated on the validation sets. The performance of the model is measured based on the averaged complement of ROC AUC score across all validation splits. A higher score indicates that the model can not differentiate between synthetic and real data, suggesting higher in-discriminability for the synthetic data, i.e. synthetic data being undistinguishable from real data.

Utility This aspect evaluates the quality of the synthetic data in terms of their performance in downstream machine-learning tasks. This evaluation is conducted using the following method:

- Machine Learning Efficacy (MLE): Following the training-on-synthetic and test-on-real setting, each dataset's classification or regression model is trained on the synthetic data and evaluated using the real testing set. The performance of the model is measured by the AUC score for classification tasks and the RMSE for regression tasks. A higher MLE score indicates better machine learning utility of the synthetic data.

The results of the quality metrics are presented in Table 6. As shown, the synthetic data without a watermark, with a tree-ring watermark, and with a ripple watermark exhibit generally similar performance in terms of quality. The most notable difference across the entire table is the logistic detection score between the no-watermark and ripple watermark in the Shoppers dataset, with a

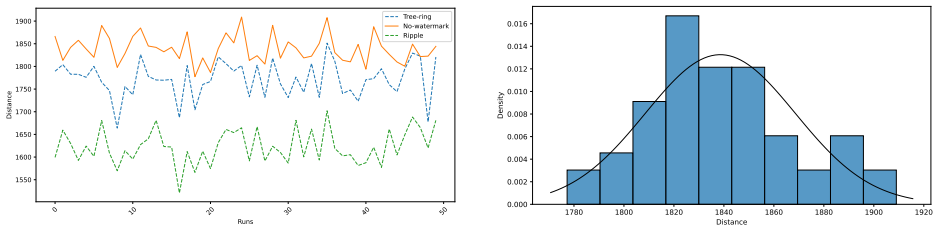
Table 1: Quality results of watermarking.

	Shape quality			Trend quality			Logistic Detection			MLE		
	W/O	TR	RP	W/O	TR	RP	W/O	TR	RP	W/O	TR	RP
Shoppers	0.922	0.922	0.917	0.907	0.907	0.901	0.637	0.635	0.620	0.894	0.893	0.893
Magic	0.917	0.917	0.912	0.939	0.939	0.937	0.711	0.710	0.704	0.834	0.833	0.834
Adult	0.933	0.933	0.930	0.887	0.886	0.879	0.652	0.652	0.651	0.824	0.823	0.824
Credit	0.929	0.929	0.923	0.904	0.904	0.899	0.741	0.742	0.741	0.790	0.790	0.789

difference of 0.017, which is still quite small. This underlines how the watermarking does not compromise the quality of the synthesized data.

4.3 Detection effectiveness

As described in Section 3.2.2, the distance between the FFT-transformed inversed noise matrix $F(\tilde{Z}_T)$ and the ground-truth watermark patch K is used for watermark detection. For all detection results, we conduct 50 random runs for each dataset under three scenarios: no-watermark, tree-ring watermark, and ripple watermark. Figure 3 (a) illustrates the resulting distances for the Adult dataset. As depicted in the figure, the distances under the no-watermark setting are generally the highest. While tree-ring watermarked matrices exhibit distances close to the no-watermark matrices, ripple watermarked matrices consistently show significantly lower distances. This can be attributed to distortions arising from reconstruction errors in the VAE and the inverse diffusion process, which can alter the values in the latent matrices and consequently distort the watermark in the FFT-transformed noise matrix. These distortions are predominantly noisy and smoothy, tending to concentrate around the low-frequency region of the tabular spectrum. As a result, the tree-ring watermark, which primarily occupies the center in the Fourier space, experiences greater distortion. In contrast, the ripple watermark covers more regions beyond the center where fewer distortions occur, leading to smaller distances between the ripple watermarked noise matrices and the ground-truth watermark patch.



(a) (b)

Figure 3: (a) The distances between the FFT-transformed reversed noise matrices and the ground-truth watermark patches for the Adult dataset. (b) The distribution of the distances in the no-watermark scenario for the Adult dataset.

For each dataset, the ripple watermark consistently induces a clear separation in the measured distances compared to the no-watermark scenario. This characteristic allows for the selection of a threshold that achieves perfect detection accuracy (100%). However, relying solely on detection accuracy can be overly simplistic. To provide a more nuanced assessment of detection effectiveness, we leverage **p-values**. Specifically, given that the distances in the no-watermark scenario follow a Gaussian distribution as shown in Figure 3 (b), the p-value quantifies the probability that a given distance is sampled from this distribution. Statistically, a p-value less than 0.05 shows that the distance is unlikely to arise from the no-watermark distribution, while a p-value less than 0.02 indicates that the distance is significantly unlikely to be observed from this distribution.

The p-values for every distance given by both the tree ring and ripple watermarking methods are calculated. The average and maximum p-values are presented in Table 2. As shown in the table, tree ring watermarking achieves a p-value below 0.05 in only one dataset, and its highest p-value reaches

Table 2: Detection results of watermarking.

	Tree-ring		Ripple	
	mean p-value	max p-value	mean p-value	max p-value
Shoppers	3.33e-2	4.85e-1	9.65e-8	3.27e-6
Magic	2.32e-1	8.50e-1	7.47e-4	1.01e-2
Adult	1.59e-1	1.00	7.07e-8	2.80e-6
Credit	7.58e-2	5.07e-1	8.40e-8	3.20e-6

1. In contrast, the ripple watermarking consistently exhibits significantly lower average p-values, and its highest p-value still remains below 0.02.

4.4 Robustness against attacks

To assess the robustness of ripple watermarking, we introduce six attacks specifically designed for tabular data. These attacks are categorized into two types: distortion attacks and deletion attacks.

Distortion attacks In distortion attacks, the attacker distorts a portion of rows, columns, or values in the synthetic tables using neighboring distortion. Specifically, given an attack percentage p, a neighboring distance d, and the total number of rows (n_r), columns (n_c), and values (n_v), the following attacks are performed:

- **Row distortion attack**: The i-th row in the synthetic table is replaced by a random row from its neighbors, ranging from the $(i-d)$-th row to the $(i+d)$-th row. A total number of $n_r \times p$ rows are attacked.
- **Value distortion attack**: The value at index (i, j) is replaced by a random neighboring value in the same column, within the indices from $(i, (j-d))$ to $(i, (j+d))$. A total number of $n_v \times p$ values are attacked.
- **Column distortion attack**: In the j-th column, every value in the column is attacked by a value distortion attack. A total number of $\max(1, n_c \times p)$ columns are attacked.

Deletion attacks In deletion attacks, a portion of rows, columns, or values in the synthetic tables is randomly deleted by the attacker. Given an attack percentage p, the number of rows/columns/values to be deleted is calculated as the total number of rows (n_r), columns (n_c), or values (n_v) multiplied by p. Thus, in the row deletion attack, $n_r \times p$ rows are randomly sampled and deleted. In the column deletion attack, $\max(1, n_c \times p)$ columns are randomly sampled and deleted to ensure that at least one column is deleted if the number of columns is small. For the value deletion attack, $n_v \times p$ values across the entire table are randomly sampled and deleted.

As for the detection of the watermark after attacking, it follows the methodology outlined in our watermark detection phase in Section 3.2.2. However, deletion attacks alter the table's shape and, consequently, the shape of the FFT-transformed reversed noise, making the computation of the distance to the ground-truth watermark patch tricky. To address this, we introduce an additional step to detect deletion-attacked tables.

Specifically, for all deletion-attacked tables, we first standardize the shape of the data by adding rows, columns, or values from a non-watermarked dataset to match the shape of the ground-truth watermark patch. If rows are missing, an equivalent number of rows are randomly sampled from a non-watermarked synthetic table and added to the attacked table. If columns are missing, the corresponding columns from a non-watermarked synthetic table are added. For missing values, a random value from the same column in a non-watermarked synthetic table replaces the missing value. Once these additions are made, the deletion-attacked tables are restored to a complete shape, enabling them to undergo detection as described in the watermark detection phase.

The results of deletion and distortion attacks on ripple-watermarked data are presented in Table 3 and Table 4. In this context, a p-value less than 0.02 indicates that the ripple watermarking is highly robust, a p-value between 0.02 and 0.05 indicates reasonable robustness, and for a p-value greater than 0.05 we consider that ripple watermarking fails to detect the attack. As shown in the tables,

attacks targeting rows are more detrimental to the detection of the watermark compared to attacks on columns and values. This suggests that the structural integrity of the watermark is more closely related to the row structure. Removing entire rows significantly disrupts the underlying distribution within the watermark, leading to a more substantial distortion than modifications to individual values or columns.

Table 3: Deletion attack results.

	No attack	Row deletion			Column deletion			Value deletion		
		5%	10%	15%	5%	10%	15%	5%	10%	15%
Shoppers	9.65e-8	1.33e-1	1.65e-1	1.80e-1	3.93e-6	6.08e-4	1.36e-3	2.22e-7	1.36e-7	1.04e-5
Magic	7.47e-4	1.06e-2	2.16e-2	4.27e-2	2.67e-3	6.86e-3	7.54e-3	2.59e-3	4.61e-3	1.43e-2
Adult	7.07e-8	1.66e-2	3.31e-2	3.78e-2	5.00e-7	8.45e-7	1.92e-4	1.27e-6	1.52e-5	6.63e-5
Credit	8.40e-8	3.41e-2	5.14e-2	7.71e-2	8.00e-7	1.33e-4	1.95e-5	8.21e-6	1.67e-4	2.02e-3

For deletion attacks, row deletions are the most damaging. In the Shoppers dataset, the p-value reaches 0.133 with just 5% row deletion. In the Credit dataset, ripple watermarking remains reasonably robust with 5% row deletion but fails at 10%. The Magic and Adult datasets show p-values of 0.0427 and 0.0378 with 15% row deletion, suggesting that ripple watermarking is reasonably robust to row deletion in these datasets. Despite the significant impact of row deletions, ripple watermarking demonstrates strong robustness to column and value deletion attacks across all datasets, consistently producing p-values below 0.02 for attack percentages ranging from 5% to 15%.

Table 4: Distortion attack results.

	No attack	Row distortion			Column distortion			Value distortion		
		5%	10%	15%	5%	10%	15%	5%	10%	15%
Shoppers	9.65e-8	1.12e-3	4.34e-2	3.42e-1	1.31e-6	1.35e-6	1.22e-4	3.09e-5	1.20e-3	3.93e-3
Magic	7.47e-4	1.28e-2	6.98e-2	2.29e-1	4.41e-2	4.96e-2	8.69e-2	8.84e-3	3.70e-2	1.47e-1
Adult	7.07e-8	2.45e-5	1.21e-3	2.22e-2	2.06e-5	1.80e-5	4.01e-2	2.37e-5	1.27e-3	1.50e-2
Credit	8.40e-8	1.31e-5	5.75e-4	7.84e-3	7.10e-6	1.00e-5	2.02e-4	5.56e-5	3.46e-3	4.78e-2

For distortion attacks, row distortions are again the most harmful, yielding generally higher p-values. Ripple watermarking shows less robustness in smaller datasets like Shoppers and Magic, with p-values of 0.342 at 15% and 0.0698 at 10% row distortions, respectively. However, it remains reasonably robust in larger datasets like Adult and Credit, where the p-value is 0.0222 for the Adult dataset at 15% row distortion and consistently below 0.02 for the Credit dataset with all attack percentages. Similar to deletion attacks, ripple watermarking exhibits stronger robustness to column and value distortion attacks. While it fails at the Magic dataset given 15% of column distortion and value distortion, all other p-values remain below 0.05 across all datasets and attack percentages.

4.5 Ablation studies

4.5.1 Impact of the watermarking area

The radius of the outermost circle in the ripple watermark is a critical hyperparameter in our watermarking technique. For instance, if this radius is too small, similar to the tree-ring watermark at the center of the table, the watermark becomes undetectable, as demonstrated in Section 4.3. To investigate the optimal radius r for the outermost circle that ensures stable detectability of the watermark, experiments with various settings of r are conducted. Specifically, for a given total number of rows n, the radius r is set to the integer value of $n/100$, $n/50$, $n/30$, and $n/10$.

Table 5: Detection results on different sizes of ripple watermarks.

	Ripple (r∼n/100)	Ripple (r∼n/50)	Ripple (r∼n/30)	Ripple (r∼n/10)
Shoppers	8.81e-4	9.65e-8	7.85e-11	6.76e-23
Magic	4.18e-3	7.47e-4	8.99e-4	1.24e-1
Adult	2.98e-4	7.07e-8	4.00e-2	2.00e-1
Credit	9.34e-5	8.40e-8	6.50e-9	1.84e-20

The average results of 50 random runs on the detectability of the ripple watermark with different outermost radii are presented in Table 5. As indicated in the table, when r is approximately $n/50$, ripple watermarking achieves a relatively high and stable detectability across all datasets, with all p-values less than 0.02. However, when r is smaller, such as $n/100$, the p-values decrease, indicating reduced detectability of the watermark. Conversely, when r is too large, the detectability becomes unstable; the p-values decrease in the Shoppers and Credit datasets, while they increase in the Magic and Adult datasets. This instability is likely due to the larger watermarks spanning both the low-frequency (central) and high-frequency (peripheral) areas of the shifted Fourier transform. Reversion errors during the VAE and diffusion processes may distort different areas of the reversed FFT noise matrices across datasets, damaging the embedded watermark and thereby introducing sensitivity and instability.

4.5.2 IMPACT OF THE WATERMARKING TARGET

As described in Section 3.2.2, the ripple watermark is injected into the FFT-transformed noise matrix Z_T before the sampling process in the diffusion model and is detected in the inverse FFT-transformed $\tilde{Z}_T$ after the diffusion process. Alternatively, another potential watermarking target is the FFT-transformed Z_0, where the watermark is embedded after the sampling process of the diffusion model but before the decoding process of the VAE, with detection occurring in the inverse FFT-transformed $\tilde{Z}_0$. To investigate how different watermarking targets might affect synthetic data quality and watermark detectability, experiments are conducted using the same ripple watermark on Z_T and Z_0.

Table 6: Quality results of watermarking on Z_T and Z_0.

	Shape quality			Trend quality			Logistic Detection			MLE		
	W/O	W-Z_T	W-Z_0	W/O	W-Z_T	W-Z_0	W/O	W-Z_T	W-Z_0	W/O	W-Z_T	W-Z_0
Shoppers	0.922	0.917	0.909	0.907	0.901	0.907	0.637	0.620	0.639	0.894	0.893	0.892
Magic	0.917	0.912	0.918	0.939	0.937	0.940	0.711	0.704	0.636	0.834	0.834	0.817
Adult	0.933	0.930	0.908	0.887	0.879	0.851	0.652	0.651	0.490	0.824	0.824	0.831
Credit	0.929	0.923	0.912	0.904	0.899	0.899	0.741	0.741	0.680	0.790	0.789	0.790

Table 7: Detection results of watermarking on Z_T and Z_0.

	Watermerking on Z_T		Watermarking on Z_0	
	mean p-value	max p-value	mean p-value	max p-value
Shoppers	9.65e-8	3.27e-6	6.67e-10	2.75e-8
Magic	7.47e-4	1.01e-2	3.63e-16	1.16e-14
Adult	7.07e-8	2.80e-6	4.24e-1	1.00
Credit	8.40e-8	3.20e-6	5.24e-7	1.13e-5

The experiments comprised 50 random runs, with results for synthetic data quality and watermark detectability presented in Table 6 and Table 7, respectively. Generally, watermarking on Z_0 has a more noticeable and unstable impact on both synthetic data quality and watermark detectability compared to watermarking on Z_T.

In the Shoppers, Magic, and Credit datasets, watermarking on Z_0 always results in larger differences in data quality scores (Table 6), and similar or better detectability (Table 7). This may be due to the fact that Z_0 is closer to the tabular data space, causing watermarking to have a greater influence on the synthetic data. Additionally, detecting the watermark before the diffusion process leads to possibly less distortion, thereby enhancing detectability.

However, watermarking on Z_0 exhibits unusual behavior with the Adult dataset, as evidenced by a logistic detection score of 0.490 for data quality and a mean p-value of 0.424 for watermark detection. The Adult dataset is unique among the four datasets because it contains more categorical columns than continuous ones. This suggests that the VAE alone may be insufficient for effectively unifying and describing the latent space. The diffusion component appears to aid in further expressing the latent space, enhancing both watermark embedding and data sampling. This additional step may account for the observed differences in the Adult dataset.

4.5.3 IMPACT OF THE ATTACKING TARGET

Apart from the watermarking target, the attacking target is also an interesting point of investigation. We can either attack the synthetic table X before the VAE encoder or the latent matrix $\tilde{Z}_0$ just after the VAE encoder but before diffusion. Although in practice, an attacker would rarely have the opportunity to attack the latent matrix $\tilde{Z}_0$, comparing attacks on X and $\tilde{Z}_0$ allows us to assess whether the VAE preserves the consistency of attacks between the original space and the latent space. Fifty random runs of row deletion and distortion attacks, identified as the most harmful attacks in Section 4.4, are conducted, with results presented in Table 8 and Table 9.

Table 8: Row deletion results on X and $\tilde{Z}_0$.

	No attack	Row deletion on X			Row deletion on $\tilde{Z}_0$		
		5%	10%	15%	5%	10%	15%
Shoppers	9.65e-8	1.33e-1	1.65e-1	1.80e-1	1.15e-1	1.65e-1	1.90e-1
Magic	7.47e-4	1.06e-2	2.16e-2	4.27e-2	1.05e-2	2.69e-2	5.51e-2
Adult	7.07e-8	1.66e-2	3.31e-2	3.78e-2	1.56e-1	1.89e-1	2.40e-1
Credit	8.40e-8	3.41e-2	5.14e-2	7.71e-2	3.46e-2	5.26e-2	7.76e-2

Table 9: Row distortion results on X and $\tilde{Z}_0$.

	No attack	Row distortion on X			Row distortion on $\tilde{Z}_0$		
		5%	10%	15%	5%	10%	15%
Shoppers	9.65e-8	1.12e-3	4.34e-2	3.42e-1	6.83e-4	3.94e-2	3.63e-1
Magic	7.47e-4	1.28e-2	6.98e-2	2.29e-1	8.80e-3	6.82e-2	2.25e-1
Adult	7.07e-8	2.45e-5	1.21e-3	2.22e-2	2.68e-5	1.64e-3	3.02e-2
Credit	8.40e-8	1.31e-5	5.75e-4	7.84e-3	1.12e-5	5.15e-4	8.68e-3

Intuitively, attacking the latent matrix $\tilde{Z}_0$ would cause more significant harm to watermark detection. However, this effect is only observed in the Adult dataset. In this dataset, while the watermark remains reasonably robust under a 15% row deletion in the synthetic table X, it fails at a 5% row deletion on $\tilde{Z}_0$, with a p-value of 0.156. No significant differences are observed in other scenarios, indicating that attacks before and after the VAE are generally consistent. The unique behavior in the Adult dataset, which contains more categorical columns, suggests that the VAE's unified latent space offers some protection for the watermark when attacked in the original space. This finding underscores the VAE's role in maintaining attack consistency and preserving watermark robustness in datasets with diverse column types.

5 CONCLUSION

In this paper, we introduce `Ripple`, the first watermarking algorithm specifically designed for tabular diffusion models during the sampling phase. Our novel approach embeds ripple watermarks in the Fourier space of the initial noise matrix, effectively balancing data quality, detectability, and robustness.

Our comprehensive evaluation on four datasets demonstrates that `Ripple` preserves synthetic data quality, with an average quality difference of less than 0.6% compared to non-watermarked data. This ensures that the synthetic tables retain their resemblance, discriminability, and downstream utility. Moreover, `Ripple` achieves high watermark detectability, with average statistical p-values over $25\times$ lower than 0.02 across all datasets, representing an improvement of more than 2 orders of magnitude compared to the tree-ring baseline. This significant enhancement in detectability is attributed to our innovative ripple watermarking pattern, which embeds watermarks across a broad frequency spectrum. Additionally, our robustness analysis shows that `Ripple` is resilient against various post-editing attacks, including the deletion and distortion of rows, columns, and values. Specifically, 85% of the p-values remain below 0.05 in the presence of different attack settings,

demonstrating Ripple's effectiveness in maintaining watermark integrity under malicious conditions.

In summary, Ripple represents a significant advancement in the field of watermarking for synthetic tabular data. By effectively embedding watermarks while preserving data quality and ensuring high detectability and robustness against post-editing, Ripple sets a new standard for traceability and auditability in synthetic tabular data. While Ripple is effective, several aspects warrant further development. First, since each row in a table represents a single data sample, and Ripple watermarks the entire noise matrix used to generate a complete table, Ripple watermarking operates at the batch level. More fine-grained watermarking techniques focused on the sample/row level of tabular data are an interesting area for further research to enhance watermark detectability and robustness. Additionally, exploring the application of Ripple to other types of generative models or developing a generic watermarking scheme for all types of generative tabular models would be valuable directions for future investigation as well.

REFERENCES

[1] Haleh Akrami, Sergul Aydore, Richard M Leahy, and Anand A Joshi. Robust variational autoencoder for tabular data with beta divergence. *arXiv preprint arXiv:2006.08204*, 2020.

[2] Laura Aviñó, Matteo Ruffini, and Ricard Gavaldà. Generating synthetic but plausible healthcare record datasets. *arXiv preprint arXiv:1807.01514*, 2018.

[3] Mauro Barni, Christine I Podilchuk, Franco Bartolini, and Edward J Delp. Watermark embedding: Hiding a signal within a cover image. *IEEE Communications magazine*, 39(8):102–108, 2001.

[4] Barry Becker and Ronny Kohavi. Adult. UCI Machine Learning Repository, 1996. DOI: https://doi.org/10.24432/C5XW20.

[5] R. Bock. MAGIC Gamma Telescope. UCI Machine Learning Repository, 2007. DOI: https://doi.org/10.24432/C52C8B.

[6] Vadim Borisov, Kathrin Seßler, Tobias Leemann, Martin Pawelczyk, and Gjergji Kasneci. Language models are realistic tabular data generators. *arXiv preprint arXiv:2210.06280*, 2022.

[7] Zalán Borsos, Raphaël Marinier, Damien Vincent, Eugene Kharitonov, Olivier Pietquin, Matt Sharifi, Dominik Roblek, Olivier Teboul, David Grangier, Marco Tagliasacchi, et al. Audiolm: a language modeling approach to audio generation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 2023.

[8] DataCebo, Inc. *Synthetic Data Metrics*, 10 2023. Version 0.12.0.

[9] Prafulla Dhariwal and Alexander Quinn Nichol. Diffusion models beat gans on image synthesis. In Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan, editors, *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems (NeurIPS)*, pages 8780–8794, 2021.

[10] Peter Eigenschink, Thomas Reutterer, Stefan Vamosi, Ralf Vamosi, Chang Sun, and Klaudius Kalcher. Deep generative models for synthetic sequential data: A survey. *IEEE Access*, 2023.

[11] Pierre Fernandez, Guillaume Couairon, Hervé Jégou, Matthijs Douze, and Teddy Furon. The stable signature: Rooting watermarks in latent diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 22466–22477, 2023.

[12] Shansan Gong, Mukai Li, Jiangtao Feng, Zhiyong Wu, and LingPeng Kong. Diffuseq: Sequence to sequence text generation with diffusion models. *arXiv preprint arXiv:2210.08933*, 2022.

[13] Xu Guo and Yiqiang Chen. Generative ai for synthetic data generation: Methods, challenges and the future. *arXiv preprint arXiv:2403.04190*, 2024.

[14] Maanak Gupta, CharanKumar Akiri, Kshitiz Aryal, Eli Parker, and Lopamudra Praharaj. From chatgpt to threatgpt: Impact of generative ai in cybersecurity and privacy. *IEEE Access*, 2023.

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

[16] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2020.

[17] Abe Bohan Hou, Jingyu Zhang, Tianxing He, Yichen Wang, Yung-Sung Chuang, Hongwei Wang, Lingfeng Shen, Benjamin Van Durme, Daniel Khashabi, and Yulia Tsvetkov. Semstamp: A semantic watermark with paraphrastic robustness for text generation. *arXiv preprint arXiv:2310.03991*, 2023.

[18] Aapo Hyvärinen and Peter Dayan. Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research*, 6(4), 2005.

[19] Stamatis Karnouskos. Artificial intelligence in digital media: The era of deepfakes. *IEEE Transactions on Technology and Society*, 1(3):138–147, 2020.

[20] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022.

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

[22] 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.

[23] 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.

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

[25] Cheng Lu, Kaiwen Zheng, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Maximum likelihood training for score-based diffusion odes by high order denoising score matching. In *International Conference on Machine Learning*, pages 14429–14460. PMLR, 2022.

[26] Nils Lukas and Florian Kerschbaum. {PTW}: Pivotal tuning watermarking for {Pre-Trained} image generators. In *32nd USENIX Security Symposium (USENIX Security 23)*, pages 2241–2258, 2023.

[27] Calvin Luo. Understanding diffusion models: A unified perspective. *arXiv preprint arXiv:2208.11970*, 2022.

[28] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International conference on machine learning*, pages 8162–8171. PMLR, 2021.

[29] OpenAI. Gpt-4 technical report. 2023.

[30] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.

[31] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 1(2):3, 2022.

[32] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution Image Synthesis with Latent Diffusion Models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR*, pages 10674–10685, 2022.

[33] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L. Denton, Seyed Kamyar Seyed Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, Jonathan Ho, David J. Fleet, and Mohammad Norouzi. Photorealistic Text-to-image Diffusion Models with Deep Language Understanding. In *NeurIPS*, 2022.

[34] C. Sakar and Yomi Kastro. Online Shoppers Purchasing Intention Dataset. UCI Machine Learning Repository, 2018. DOI: https://doi.org/10.24432/C5F88Q.

[35] Marco Schreyer, Timur Sattarov, Bernd Reimer, and Damian Borth. Adversarial learning of deepfakes in accounting. *arXiv preprint arXiv:1910.03810*, 2019.

[36] Aditya Shankar, Hans Brouwer, Rihan Hai, and Lydia Chen. Silofuse: Cross-silo synthetic data generation with latent tabular diffusion models. *arXiv preprint arXiv:2404.03299*, 2024.

[37] Jascha Sohl-Dickstein, Eric A. Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep Unsupervised Learning using Nonequilibrium Thermodynamics. In *Proceedings of the 32nd International Conference on Machine Learning, ICML*, volume 37, pages 2256–2265, 2015.

[38] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021.

[39] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in neural information processing systems*, 32, 2019.

[40] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021.

[41] Zhensu Sun, Xiaoning Du, Fu Song, and Li Li. Codemark: Imperceptible watermarking for code datasets against neural code completion models. In *Proceedings of the 31st ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, pages 1561–1572, 2023.

[42] Ruixiang Tang, Qizhang Feng, Ninghao Liu, Fan Yang, and Xia Hu. Did you train on my dataset? towards public dataset protection with cleanlabel backdoor watermarking. *ACM SIGKDD Explorations Newsletter*, 25(1):43–53, 2023.

[43] Yusuke Tashiro, Jiaming Song, Yang Song, and Stefano Ermon. CSDI: Conditional Score-based Diffusion Models for Probabilistic Time Series Imputation. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems*, pages 24804–24816, 2021.

[44] Umut Topkara, Mercan Topkara, and Mikhail J Atallah. The hiding virtues of ambiguity: quantifiably resilient watermarking of natural language text through synonym substitutions. In *Proceedings of the 8th workshop on Multimedia and security*, pages 164–174, 2006.

[45] Yuxin Wen, John Kirchenbauer, Jonas Geiping, and Tom Goldstein. Tree-rings watermarks: Invisible fingerprints for diffusion images. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[46] Cheng Xiong, Chuan Qin, Guorui Feng, and Xinpeng Zhang. Flexible and secure watermarking for latent diffusion model. In *Proceedings of the 31st ACM International Conference on Multimedia*, pages 1668–1676, 2023.

[47] Lei Xu, Maria Skoularidou, Alfredo Cuesta-Infante, and Kalyan Veeramachaneni. Modeling tabular data using conditional gan. In *Advances in Neural Information Processing Systems*, 2019.

[48] Zijin Yang, Kai Zeng, Kejiang Chen, Han Fang, Weiming Zhang, and Nenghai Yu. Gaussian shading: Provable performance-lossless image watermarking for diffusion models. *arXiv preprint arXiv:2404.04956*, 2024.

[49] I-Cheng Yeh. Default of Credit Card Clients. UCI Machine Learning Repository, 2016. DOI: https://doi.org/10.24432/C55S3H.

[50] Ning Yu, Vladislav Skripniuk, Sahar Abdelnabi, and Mario Fritz. Artificial fingerprinting for generative models: Rooting deepfake attribution in training data. In *Proceedings of the IEEE/CVF International conference on computer vision*, pages 14448–14457, 2021.

[51] Ning Yu, Vladislav Skripniuk, Dingfan Chen, Larry S Davis, and Mario Fritz. Responsible disclosure of generative models using scalable fingerprinting. In *International Conference on Learning Representations*, 2021.

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

[53] Kevin Alex Zhang, Lei Xu, Alfredo Cuesta-Infante, and Kalyan Veeramachaneni. Robust invisible video watermarking with attention. *arXiv preprint arXiv:1909.01285*, 2019.

[54] Yunqing Zhao, Tianyu Pang, Chao Du, Xiao Yang, Ngai-Man Cheung, and Min Lin. A recipe for watermarking diffusion models. *arXiv preprint arXiv:2303.10137*, 2023.

[55] Zilong Zhao, Aditya Kunar, Robert Birke, and Lydia Y Chen. Ctab-gan: Effective table data synthesizing. In *Asian Conference on Machine Learning*, pages 97–112. PMLR, 2021.

[56] Chaoyi Zhu, Jeroen Galjaard, Pin-Yu Chen, and Lydia Y Chen. Duwak: Dual watermarks in large language models. *arXiv preprint arXiv:2403.13000*, 2024.

2

Extended Related Works

2.1. Generative Models for Data Synthesis

Data generation has been a cornerstone of machine learning, with techniques progressing from basic statistical models to sophisticated deep learning architectures. Early works rely on simpler models for data synthesis. The Gaussian Mixture Model (GMM) [26] is one of the fundamental machine learning models for data synthesis, by representing data as a combination of Gaussian distributions and enabling data synthesis by sampling from the learned distributions. However, GMMs struggle with complex data structures and high dimensionality. Another representative and simple generative model is Naive Bayes [12], which learns joint probability distributions and generates new data by sampling from them. Similar to GMMs, Naive Bayes lacks the flexibility to model intricate relationships within data. While these early approaches laid the groundwork for generative modeling, they are limited in their ability to capture complex data patterns, particularly for tabular datasets. In recent years, the advent of deep learning has significantly advanced generative modeling capabilities.

Generative Adversarial Networks (GANs) [13; 4; 53; 8]are one of the most popular deep generative models, which pit a generative network against a discriminative network in a game-like setting. The generator learns to produce data that the discriminator cannot distinguish from real data. While powerful, GANs can be challenging to train due to potential convergence issues and mode collapse, where the generator gets stuck in a loop, producing a limited set of outputs.

Variational Autoencoders (VAEs) [20; 16; 33] offer a probabilistic approach to data generation by learning a latent representation of the data, which models the distribution more explicitly than GANs. These models use an encoder-decoder architecture. The encoder compresses input data into a latent space, and the decoder reconstructs the data from that latent representation. Generating new data points is done by sampling from the latent space. However, VAEs can suffer from information loss during the encoding process, potentially impacting the fidelity of the generated data.

Flow-based generative models, also known as normalizing flows [11; 30; 28; 43] represent another class of explicit generative models. They leverage invertible transformations to map a simple distribution (e.g., Gaussian) into a complex data distribution. Using the Change of Variable Theorem, they estimate the density of the transformed data based on the original distribution. This allows for efficient sampling and the

generation of new data points. However, the effectiveness of flow-based models can be hampered by the complexity of the data distribution, particularly for tabular data with mixed data types.

Diffusion models [34] are a recent development in generative modeling, achieving impressive results in various applications. These models [17; 31; 19] operate in two stages: a forward diffusion process that gradually adds noise to the data, and a reverse diffusion process that aims to recover the original data from the noise. By reversing the diffusion process with controlled noise, synthetic data can be generated. While these models exhibit great promise, their application raises concerns such as training data privacy and synthetic data authenticity, therefore requiring further investigation.

2.2. Watermarking on Generative Models

Recent advancements in generative artificial intelligence (AI) have blurred the lines between human-produced and synthetic content. These models can now generate high-fidelity images, text, and various forms of synthetic data indistinguishable from their real-world counterparts, highlighting its significant utility [29; 37]. While this technology holds immense potential across various fields, it also presents significant challenges, particularly concerning its potential for misuse in disseminating misinformation.

Sophisticated AI models are capable of creating deepfakes – highly realistic but fabricated audio, video, or images that purport to depict real-world events or individuals engaging in actions or uttering statements they never did (Deepfakes Web). This technology can be weaponized to erode trust in institutions, manipulate public opinion, and damage reputations [1; 18; 7]. Furthermore, generative AI can be used to generate convincing text, emails, or voices that mimic individuals or organizations, facilitating phishing scams, identity theft, and other forms of fraud [32; 14]. To mitigate these risks, watermarking AI-generated content has emerged as a promising solution. By embedding an invisible identifier within the content itself, watermarks can aid in tracing the origin of the content and enhancing its authenticity.

2.2.1. Watermarking in Image Generation

For image synthesis, watermarking methods have been mainly proposed for two types of deep generative models: the generative adversarial networks (GANs) and diffusion models. In the context of generative adversarial networks (GANs), watermarking during training involves inserting watermarks into all training data, thereby embedding the watermark into generated images during sampling. Examples of this approach include methods [45; 46] that modify the training procedure to achieve watermarking in subsequent image generation. PTW [25] introduces a method that embeds an n-bit watermarking message into the latent space used by the generator, allowing for watermarking with pre-trained generators and eliminating the need for watermarked training data.

In diffusion models, similar strategies are employed to implant watermarks into the training data. Zhao et al. describe an approach for unconditional image generation where an encoder embeds watermarks into the training data, and a decoder extracts the watermark from generated images [51]. For conditional image synthesis, the same

work predefines text-image pairs to fine-tune the generative model, thus avoiding the need to train large-scale models like Stable Diffusion from scratch. Tree-ring watermarks [40] offer a method that bypasses the need for training or fine-tuning. This technique embeds a detectable watermark in the frequency domain of the initial noise matrix and utilizes the invertibility of the diffusion model to recover the noise matrix and detect the watermark.

Xiong et al. [41] propose an end-to-end method for watermarking images generated by latent diffusion models (LDMs) using an encoder-decoder framework. The message encoder transforms a user-specific message into a message matrix, which is then embedded into the image by the fine-tuned LDM. The message can be extracted from the watermarked image using a message decoder. This method allows for flexible message embedding and robustly defends against message embedding escape attacks, outperforming existing post-hoc and LDM-specific watermarking methods.

While several studies have developed innovative watermarking techniques and evaluated their robustness against known attacks, concerns have been raised regarding the adequacy of these robustness assessments. Adaptive attacks, which assume that the attacker knows the watermarking algorithm but not the secret key, have been shown to compromise the robustness of many watermarking methods in image classifiers [24]. This highlights the need for more rigorous evaluations of watermarking robustness in generative models.

2.2.2. Watermarking in Text Generation

In the realm of large language models (LLMs) such as GPT-4 [3], Gemini [36], and Llamma [39], impressive capabilities have been showcased across various tasks, spanning from machine translation to dialogue systems and code generation. However, as these models grow in size and complexity, their capacity to rapidly disseminate information, including misinformation, presents significant challenges concerning intellectual property and content authenticity. Text watermarking emerges as a viable solution to embed unique, subtle markers within content generated by LLMs, facilitating tracking and attribution to address concerns regarding accountability and intellectual property rights in LLM deployment.

Recent advancements in LLM security have enabled the development of robust watermarking techniques. These techniques effectively embed watermarks within the generated text and maintain their detectability even after post-generation modifications. This is achieved by manipulating the model's logits and probabilities associated with token selection during next-token generation. A prominent strategy involves segmenting the vocabulary into 'green' (preferred) and 'red' (non-preferred) lists based on a hashed value derived from a secret key and previous tokens. The logit values, which represent the model's internal preference for each token, are then adjusted for green tokens. This adjustment, controlled by a bias parameter δ and a hyper-parameter γ, increases the likelihood of selecting green tokens during generation, subtly embedding the watermark within the text. This approach was pioneered by Kirchenbauer et al [21], who innovatively organized the model's vocabulary to bias the decoding process towards green tokens.

Several advancements have been proposed to improve watermark robustness against user modifications. Zhao et al. [50] introduced a technique maintaining a consistent red-

green split, achieving double the robustness compared to earlier models. Additionally, Kirchenbauer et al. [21] explored detection schemes effective for identifying texts embedded within larger documents. Kuditipudi [23] proposed a watermarking scheme that utilizes a key as long as the generated text for calculating alignment costs. This ensures the watermark's persistence even after post-watermarking edits like insertions or deletions. Sampling methods proposed by OpenAI [2] maintain the original probability estimates while incorporating controlled randomness. These approaches utilize an exponential scheme to select tokens based on unaltered probabilities and a vector of random numbers influenced by historical inputs and the secret key. Instead of a single watermark, a recent development, Duwak [52], significantly improves watermark detection efficiency and text quality. This method utilizes dual watermarking, embedding two independent secret patterns into the token probability distribution and sampling scheme. Compared to single watermarking, Duwak significantly improves watermark detection efficiency and maintains high text quality.

2.2.3. Watermarking in Other Modalities

While most existing research focuses on watermarking image generative models and large language models, watermarking techniques are also crucial in other modalities such as tables, videos and text-image multimodalities.

One watermarking approach applied to the generation of tabular data is by He et al. [15]. This method is a post-processing technique, where the watermark is added after the original table has been generated. It strategically embeds watermarks through data binning, dividing the range of feature values into finely segmented intervals and inserting watermarks within selected "green list" intervals. The effectiveness of this technique is supported by a statistical hypothesis-testing framework, ensuring watermark detection under minimal assumptions. The WavMark [9] framework demonstrates robust audio watermarking through an invertible encoder-decoder architecture that embeds watermarks in the frequency domain, ensuring durability against signal alterations and noise. In video synthesis, techniques like RivaGAN [49] use adversarial networks with attention-based mechanisms to embed watermarks into videos, providing security against digital tampering. In the rapidly evolving field of synthetic text-to-image synthesis, models like DALL-E 2 and Imagen adopt methods such as SteganoGAN [48], which integrate watermarks directly into the image generation process, ensuring the authenticity and origin verification of AI-generated visual content. These innovations illustrate the expanding scope of digital watermarking, adapting to the complexities of modern media while safeguarding intellectual property.

3

Additional Experimental Results

In this chapter, supplementary experimental results are presented, focusing on the comparison between different generative models and providing additional visualizations of the `Ripple` watermarking method. Section 3.1 compares six different generative models to evaluate their respective capabilities in generating tabular data. Section 3.2 provides visualization results to demonstrate the imperceptibility and detectability of the `Ripple` watermarking technique.

3.1. Comparison Between Tabular Generative Models

To evaluate the performance of various tabular generative models, six different tabular synthesizers are trained and compared. These include three GAN-based models: CTGAN [42], CopulaGAN [35], and ADS-GAN [44]; one statistical model: Gaussian Copula [27]; and two diffusion-based models: TabDDPM [22] and a latetn tabular diffusion model(LTDM) based on TabSYN [47].

GAN-based models

Three distinct Generative Adversarial Networks (GAN) models are considered:

- CTGAN [42] focuses on conditional generation using GANs. It addresses non-Gaussian and multimodal distributions in continuous columns through mode-specific normalization and mitigates class imbalance in categorical columns with a conditional generator that employs a conditional vector and a training-by-sampling method.
- CopulaGAN [35] enhances CTGAN by using cumulative distribution function-based transformations with Gaussian Copulas, which model dependencies through multivariate normal errors. This approach improves the inference process using a likelihood-based method, thereby enhancing the model's ability to capture real data trends.
- ADS-GAN [44] is a conditional GAN framework designed to generate synthetic data while minimizing re-identification risk. It incorporates a record-level identifiability metric based on weighted Euclidean distances into the generator's loss function, achieving a balance between data utility and anonymization.

Statistical models

While deep neural networks are often preferred for synthesizing tabular data, simpler statistical modeling approaches, such as Gaussian Copulas, have also demonstrated considerable success. The Gaussian Copula (GC) method [27] utilizes the training data to fit Gaussian Copula functions to the empirical distribution. These functions model a Gaussian joint probability distribution that effectively captures both the marginal distributions and interdependence structures of the data.

Difussion models

Diffusion models have recently become a prominent paradigm in generative modeling for computer vision and natural language processing. Here, two models are considered for tabular synthesis:

- TabDDPM[22] employs the Gaussian diffusion process, a fundamental component of the original DDPM[17], to effectively model numerical columns. It also utilizes a multinomial diffusion process to model categorical and binary features by introducing uniform noise across classes, thereby strategically corrupting the data to facilitate accurate modeling.
- Latent Tabular Diffusion Model (LTDM), based on TabSYN [47], transforms both continuous and categorical features into a unified continuous latent space using an autoencoder. A unified diffusion model is then applied to noise and denoise the continuous latent features. Unlike TabSYN, which employs a score-based diffusion model, LTDM uses the reversible diffusion model DDIM.

Two Python libraries are utilized for implementing five of the mentioned tabular data generative models. Specifically, Synthetic Data Vault (SDV) [1] is employed for CTGAN, CopulaGAN, and Gaussian Copula and Synthcity [2] is used for ADS-GAN and TabDDPM. LTDM is implemented based on the repository of TabSYN [3] while the diffusion model is re-implemented. To ensure a fair comparative analysis, neural networks used across all models were configured with the same architecture, consisting of three multi-layer perceptron (MLP) layers, each with 256 dimensions.

Metrics and datasets

Given an original dataset, these synthesizers generate synthetic tabular data, and the quality of the synthetic data is measured from three aspects: i) **resemblance**, ii) **discriminability**, and iii) **utility**. These metrics assess whether the synthetic results are similar to the original data and practically useful. The metrics are developed based on common practices in synthetic data generation, encompassing both statistical and machine learning-based approaches, and are reported as scores in the 0-100 range.

The resemblance metric measures how closely the distribution and inter-correlation of the columns in the synthetic data match those of the original data, ensuring that the synthetic data captures the statistical patterns and characteristics of the original data. The resemblance metric is composed of five similarity measures:

[1]https://github.com/sdv-dev/SDV
[2]https://github.com/vanderschaarlab/synthcity
[3]https://github.com/amazon-science/tabsyn

- Column Similarity This calculates the correlation between each original and synthetic column, using Pearson's coefficient for numerical columns and Theil's U for categorical columns.
- Correlation Similarity This measures the correlation between the correlation coefficients of each column pair. First, the Pearson correlation for numerical pairs, Theil's U for categorical pairs, and the correlation ratio for numerical-categorical cases are calculated. Then, the correlation between these coefficients is calculated.
- Statistical Similarity This employs Spearman's Rho to correlate descriptive statistics (minimum, maximum, median, mean, and standard deviation) of numerical columns in synthetic and original data.
- Jensen-Shannon Similarity This uses the Jensen-Shannon distance, a symmetric distance measure between the probability distributions of the original and synthetic columns. One minus this distance is used so that higher scores are better, as in the other metrics.
- Kolmogorov-Smirnov Similarity This uses the Kolmogorov-Smirnov distance to measure the maximum difference between the cumulative distributions of each original and synthetic column. Once again, one minus the distance is used so that a higher score is better.

The discriminability metric measures how closely the synthetic data resembles the real data such that a binary classifier (XGBoost) cannot differentiate between the two. This is measured with the mean-absolute error between the classifier's probabilities and the uniform distribution (50% probability for either class), which is 0 when the classifier cannot distinguish between the two datasets. One minus the mean-absolute error is used and scaled to a score of 0-100 so that higher scores are better.

The utility metric measures how well the synthetic data performs like the original data in downstream machine learning tasks. For each column, a classifier or regressor (XGBoost) is trained with 3-fold cross-validation to predict the column from the remaining columns. Models are trained either on real or synthetic data, but in both cases, evaluated on a hold-out set of real data. The downstream performance is calculated by taking the 90th percentile of macro-averaged F1 scores for categorical columns and D2 absolute error scores (clipped to 0 and 1) for continuous columns. The utility score is derived from the ratio of the downstream performance of the synthetic data to that of the real data and scaled to 0-100.

The generative models undergo evaluation across four distinct datasets. The **Loan dataset**[5] comprises demographic details of 5000 customers, encompassing 14 features categorized into binary, interval, ordinal, and nominal measurements. The **Housing dataset**[38] pertains to houses within a specified California district, offering summary statistics derived from the 1990 Census data. It comprises 20,640 instances, featuring 1 categorical and 9 numerical attributes, with a total of 207 missing values. The **Adult dataset**[6] encompasses data concerning individuals' annual incomes and associated variables. With 48,842 instances, it consists of 15 mixed datatype features and a total of 6465 missing values. The **Cardiovascular Heart Disease dataset**[10] furnishes detailed insights into the risk factors associated with cardiovascular disease, comprising 70,000 instances featuring 13 mixed-type columns. For each dataset, every

synthesizer generates a synthetic dataset of identical size to the training dataset for subsequent evaluation.

Results

Table 3.1: Synthetic data quality of six tabular generative models

Dataset	Method	Resemblance	Discriminability	Utility
Loan	CopulaGAN	92	95	70
	CTGAN	92	85	93
	ADS-GAN	93	95	73
	Gaussian Copula	86	82	78
	TabDDPM	98	100	97
	LTDM	96	98	100
Housing	CopulaGAN	94	90	62
	CTGAN	94	92	64
	ADS-GAN	93	87	74
	Gaussian Copula	91	84	32
	TabDDPM	96	98	93
	LTDM	98	98	95
Adult	CopulaGAN	93	97	81
	CTGAN	90	79	83
	ADS-GAN	88	59	83
	Gaussian Copula	80	50	56
	TabDDPM	96	98	98
	LTDM	95	98	100
Cardio	CopulaGAN	87	93	96
	CTGAN	84	68	97
	ADS-GAN	90	71	100
	Gaussian Copula	81	63	86
	TabDDPM	95	99	100
	LTDM	100	95	100

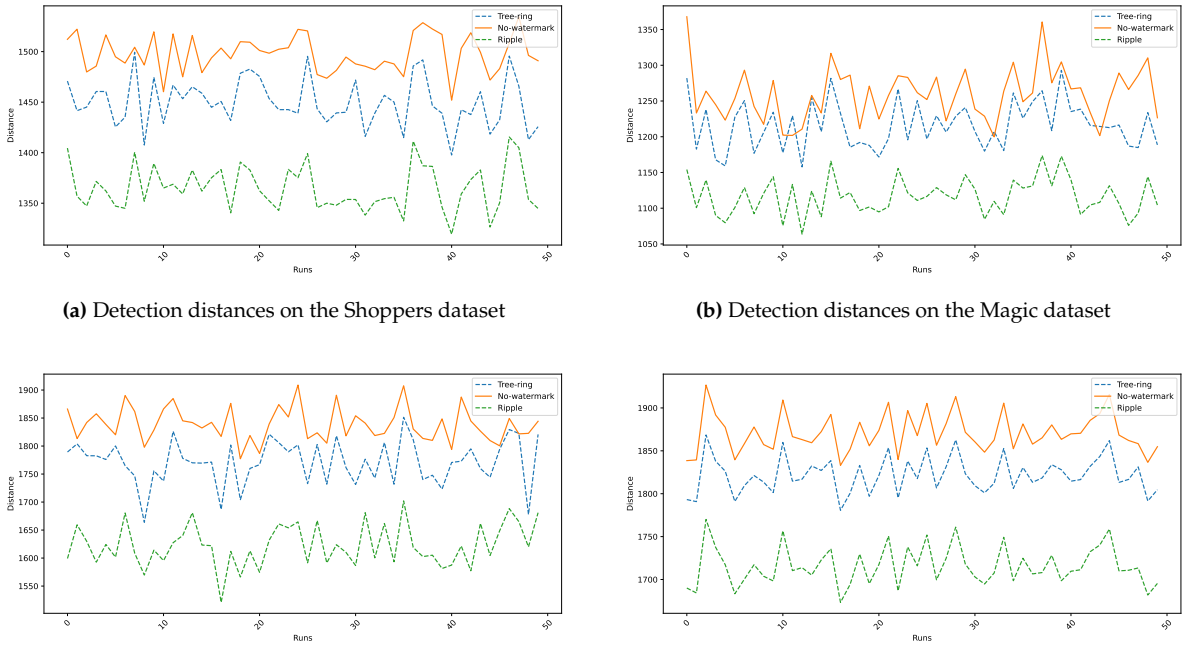
Table 3.1 presents detailed results quantifying the quality of synthetic data across all four datasets using the six described generative models. Regarding synthetic data quality, larger datasets (Adult and Cardio) generally exhibit lower resemblance and discriminability scores compared to smaller ones (Loan and Housing). These findings suggest that larger datasets pose more challenges to synthesizers due to increased dataset sizes potentially introducing greater diversity and complexity, thus making data synthesis more challenging.

However, utility scores tend to increase with dataset size. This phenomenon may be attributed to the utility metric being evaluated based on the performance of downstream machine learning tasks, which are inherently influenced by the size of the training data. In our experiments, the size of the synthetic dataset remains consistent with the corresponding real dataset. Consequently, small real datasets lead to small synthetic datasets, which may result in suboptimal performance in machine learning tasks and subsequently lower utility scores.

As depicted in the table, the latent tabular diffusion model (LTDM) consistently produces synthetic data of the highest quality, surpassing other synthesizers. Across all datasets, LTDM consistently achieves scores exceeding 95 in terms of resemblance, discriminability, and utility compared to other synthesizers. Following closely, the TabDDPM emerges as the second-best synthesizer. Conversely, the Gaussian Copula performs less favorably, being outperformed by other synthesizers across all datasets.

3.2. Attional Experimental Results of `Ripple`

The detection distances across all four datasets (Shoppers, Magic, Adult and Credit) used in the research paper in Chapter 1, under the no-watermark, tree-ring watermark, and `Ripple` watermarking settings, are presented in Figure 3.1. As illustrated in the figure, the distances under the no-watermark setting are generally the highest for all datasets. Distances under the Tree-ring watermarking exhibit distances that are close to those of the no-watermark ones. In contrast, distances under the `Ripple` watermarking consistently display significantly lower distances. These lower distances correspond to the much lower p-values achieved by `Ripple` watermarking compared to tree-ring watermarking in the research paper.



(a) Detection distances on the Shoppers dataset

(b) Detection distances on the Magic dataset

(c) Detection distances on the Adult dataset

(d) Detection distances on the Credit dataset

Figure 3.1: Detection distances of no-watermark, tree-ring watermarking and ripple watermarking on four datasets

Moreover, examples of ten rows of synthetic tables generated with no watermark, tree-ring watermark, and `Ripple` watermark across all four datasets are provided from Figure 3.2 to Figure 3.5. As demonstrated in these figures, no discernible differences are apparent to be noticed, thereby supporting the imperceptibility of the watermarks and their minimal impact on the quality of the synthetic data, as discussed in Chapter 1.

Administrative	Administrative_Duration	Informational	Informational_Duration	ProductRelated	ProductRelated_Duration	BounceRates	ExitRates	PageValues	SpecialDay	Month	OperatingSystems	Browser	Region	TrafficType	VisitorType	Weekend	Revenue
0.0	0.0	0.0	0.0	36.0	1419.2334	0.012470941	0.037224118	0.0	0.0	May	2	2	9	1	Returning_Visitor	False	False
0.0	0.0	0.0	0.0	6.0	32.30736	0.2	0.030620452	0.0	0.0	May	2	2	5	2	Returning_Visitor	False	False
11.0	3398.75	0.0	0.0	33.19255	3956.3347	0.009090909	0.028499221	0.0	0.0	Mar	2	2	1	10	Returning_Visitor	True	False
0.0	0.0	0.0	0.0	3.0	177.93471	0.008333333	0.054278462	0.0	0.0	May	2	10	1	18	Returning_Visitor	False	False
5.0	209.73753	0.0	0.0	39.65166	1301.1072	0.0	0.0	0.0	0.4	May	2	2	3	4	Returning_Visitor	False	False
11.0	3398.75	0.0	0.0	6.0	1000.98334	0.0	0.019284725	0.0	0.0	Nov	3	2	1	8	New_Visitor	False	False
5.0	230.6708	4.0	491.32297	19.0	1095.3977	0.0	0.008326806	129.43388	0.0	Dec	1	1	1	1	Returning_Visitor	False	True
1.0	53.994637	0.0	0.0	309.01248	159.36723	0.2	0.022222223	0.0	0.0	Dec	2	2	1	11	Returning_Visitor	False	False
0.0	0.0	0.0	0.0	2.0	37.989777	0.0	0.1083213	0.0	0.0	Dec	3	2	1	1	Returning_Visitor	False	False
7.0	208.27109	0.0	0.0	22.0	775.0539	0.0	0.005127804	97.61065	0.0	May	2	2	6	3	New_Visitor	False	True
0.0	0.0	0.0	0.0	42.77108	1320.3123	0.0	0.1477804	0.0	0.0	Dec	2	1	2	1	Returning_Visitor	False	False

(a) Ten rows of the synthetic table on the Shoppers dataset without watermark

Administrative	Administrative_Duration	Informational	Informational_Duration	ProductRelated	ProductRelated_Duration	BounceRates	ExitRates	PageValues	SpecialDay	Month	OperatingSystems	Browser	Region	TrafficType	VisitorType	Weekend	Revenue
0.0	0.0	0.0	0.0	40.19058	1581.5741	0.010526316	0.03333333	0.0	0.0	Mar	2	2	3	1	Returning_Visitor	False	False
0.0	0.0	0.0	0.0	6.350493	101.67554	0.0	0.029741054	0.0	0.0	Nov	2	2	7	2	Returning_Visitor	True	False
12.0	3394.9893	0.0	0.0	27.0	1642.1317	0.009090909	0.032520384	0.0	0.0	Mar	2	2	1	3	Returning_Visitor	True	False
0.0	0.0	0.0	0.0	2.0	66.10672	0.013152794	0.06666667	0.0	0.0	May	2	10	1	18	Returning_Visitor	False	False
5.0	135.55235	0.0	0.0	67.279106	1955.3083	0.0	0.0	0.0	0.0	Mar	2	2	3	2	Returning_Visitor	True	False
10.615869	3398.75	0.0	0.0	7.0	1131.8263	0.0	0.021352011	0.0	0.0	Nov	2	2	1	8	New_Visitor	True	False
13.0	3398.5696	17.55803	2523.0215	40.8462	3455.0322	0.0	0.008333333	63.192017	0.0	Dec	1	1	3	1	Returning_Visitor	False	True
5.0	57.188534	0.0	0.0	190.19827	583.88995	0.2	0.03083427	0.0	0.0	Dec	2	2	1	3	Returning_Visitor	True	False
0.0	0.0	0.0	0.0	2.0	27.018652	0.0	0.11403274	0.0	0.0	Dec	2	2	1	1	Returning_Visitor	False	False
10.0	304.19354	0.0	0.0	26.0	807.60315	0.0	0.005039437	119.50438	0.0	May	2	2	8	3	New_Visitor	False	False
5.0	0.0	0.0	0.0	59.0	1535.6113	0.0	0.12926589	0.0	0.0	Aug	2	1	2	18	Returning_Visitor	False	False

(b) Ten rows of the synthetic table on the Shoppers dataset with `Ripple` watermark

Administrative	Administrative_Duration	Informational	Informational_Duration	ProductRelated	ProductRelated_Duration	BounceRates	ExitRates	PageValues	SpecialDay	Month	OperatingSystems	Browser	Region	TrafficType	VisitorType	Weekend	Revenue
11.0	0.0	0.0	0.0	22.980406	817.04675	0.011626511	0.2	0.0	0.0	Aug	4	2	1	18	Returning_Visitor	False	False
0.0	0.0	0.0	0.0	16.0	516.3611	0.0	0.0	0.0	0.0	Mar	3	2	2	2	Returning_Visitor	False	False
10.885049	842.6082	0.0	0.0	24.0	1054.134	0.0	0.003842031	0.0	0.0	Dec	2	4	1	8	Returning_Visitor	False	False
8.0	2471.4114	0.0	0.0	4.0	122.33771	0.0	0.03333333	0.0	0.0	May	1	1	1	3	New_Visitor	False	False
0.0	0.0	0.0	0.0	14.0	2812.441	0.017515253	0.021665828	0.0	0.0	Nov	2	2	4	1	Returning_Visitor	False	False
0.0	0.0	0.0	0.0	79.03469	154.59235	0.022481292	0.08080037	0.0	0.0	Nov	3	2	6	3	Other	True	True
25.404018	3397.0461	22.921463	2530.3472	682.21704	63401.188	0.0	0.008333333	13.024882	0.0	Nov	2	2	1	2	Returning_Visitor	True	False
0.0	0.0	0.0	0.0	6.0	2.8845	0.2	0.05491632	0.0	0.0	Nov	2	10	1	1	Returning_Visitor	False	True
22.179543	3398.75	11.784934	1983.758	69.941696	5037.2793	0.005269276	0.013316727	0.0	0.0	May	1	1	3	2	Returning_Visitor	False	False
12.0	3398.75	0.0	0.0	80.3354	9228.197	0.005771693	0.020486265	0.0	0.0	Mar	2	2	1	2	Returning_Visitor	True	False
16.297164	3398.75	0.0	0.0	53.0	4440.7163	0.0	0.006375795?	35.660645	0.0	Mar	1	1	3	1	Returning_Visitor	False	False

(c) Ten rows of the synthetic table on the Shoppers dataset with Tree-ring watermark

Figure 3.2: Example of synthetic tables on the Shoppers dataset

Length	Width	Size	Conc	Conc1	Asym	M3Long	M3Trans	Alpha	Dist	class
29.50316	16.06165	2.4496179	0.39709762	0.20315146	53.773563	33.78526	-9.367778	24.631697	265.00458	h
17.136372	0.0	2.1388857	0.8415834	0.554925	26.037418	9.776217	4.9141445	88.5079	205.97107	h
34.49897	12.264082	2.799134	0.3750155	0.22525023	28.233248	-19.095558	-12.837495	1.957272	196.48692	g
74.72992	17.113258	2.6486304	0.29616234	0.16084436	100.98778	-54.09752	-12.060159	1.7024609	184.64902	g
25.97611	11.8562565	2.542755	0.47538552	0.2656954	21.594048	19.897678	-8.934775	46.408955	71.51735	g
28.009651	15.409279	2.4101448	0.19649765	0.244889	2.4382112	23.145662	11.474078	82.28989	123.640335	h
45.237206	7.3382106	2.7647102	0.36546847	0.1911917	-25.520956	-25.400356	3.504286	89.993576	198.70453	h
23.502459	16.53841	2.8302739	0.44337276	0.15362787	-7.514536	16.252663	-6.795991	34.763218	140.16533	h
59.866985	16.664644	2.4689484	0.32336682	0.17421067	88.79034	24.151981	14.311558	11.957585	206.63376	g
156.47713	50.012974	4.0026875	0.08306128	0.043591972	226.59204	142.38239	-35.127693	2.3645737	276.75076	g
112.35931	31.943472	3.3610618	0.1682879	0.083864145	-37.024055	72.93012	-20.228468	0.37044546	347.88327	g

(a) Ten rows of the synthetic table on the Magic dataset without watermark

Length	Width	Size	Conc	Conc1	Asym	M3Long	M3Trans	Alpha	Dist	class
35.30307	16.64472	2.5245123	0.38694882	0.1954228	46.70417	37.965256	-7.3183727	9.8207855	245.57162	g
13.062335	7.2635226	2.0781827	0.8664315	0.5752393	26.4537	8.316785	10.70612	85.97891	200.21384	h
40.524975	12.844649	2.7349188	0.36952683	0.20444609	34.750706	-24.953712	-11.650808	0.3556718	190.99416	g
45.505928	16.594114	2.6194267	0.3290552	0.17959978	59.461304	-32.29623	-8.444348	1.6397188	168.18362	g
25.67888	11.674168	2.5552065	0.47450054	0.2659148	15.444065	19.572388	-6.5131006	27.112617	73.268196	g
26.885931	15.534331	2.3730001	0.25279206	0.2631005	-4.2416224	17.202438	13.265056	78.76212	121.51649	h
27.589327	0.0	2.4141476	0.59803575	0.32005686	-39.763725	-17.585285	4.8646398	89.99968	212.41422	h
23.296146	16.836401	2.7585526	0.4370695	0.1678857	-11.364904	15.757224	7.7258067	11.883816	134.031	g
39.100918	13.33985	2.2458668	0.3922745	0.20304327	70.04794	18.370302	11.954219	16.58858	199.17256	g
97.981735	26.262829	3.239584	0.26989132	0.15214784	73.92106	83.37061	-21.477163	2.4486513	330.8961	g
106.55097	29.985685	3.3542128	0.17734703	0.091438055	-84.89384	72.538734	-19.269552	0.2206306	341.63913	g

(b) Ten rows of the synthetic table on the Magic dataset with `Ripple` watermark

Length	Width	Size	Conc	Conc1	Asym	M3Long	M3Trans	Alpha	Dist	class
208.68639	69.79462	3.1185472	0.18984744	0.13966085	56.81338	52.309174	-16.986662	71.44638	339.78656	h
320.13016	227.56863	5.1784062	0.013124045	0.0008054634	575.2407	-330.97327	-203.11964	66.871826	221.59763	h
69.38346	14.864264	2.9102473	0.6039451	0.18567702	-49.095432	-37.856884	11.437894	44.312267	124.532585	g
27.499226	0.0	2.6140897	0.726563	0.43497753	28.593939	-14.905209	5.8833833	88.89875	219.9429	h
37.079006	13.125928	2.4100616	0.3786842	0.1986684	70.21424	-21.964241	-13.339236	17.042065	197.8729	g
21.748087	12.045411	2.4047985	0.4842891	0.26615497	20.249588	8.193292	9.382959	1.9900787	222.11835	g
187.20859	83.46023	4.402744	0.0297663	0.007401169	-318.84048	163.65921	67.14243	2.0158803	194.94618	g
30.57708	13.872212	2.745715	0.5154582	0.31112507	14.441536	-13.944984	-24.758919	33.181526	59.399075	h
31.141603	13.693857	2.8263843	0.62804705	0.3524597	-1.8333604	-28.27726	7.720078	36.904804	63.103016	h
23.628618	24.422411	3.4936821	0.51250327	0.1286727	29.223211	52.452522	19.462797	13.253972	185.51393	g
31.047913	9.575477	3.1065223	0.42309067	0.1833185	17.17553	21.999958	-7.8504424	7.3477874	175.30763	g

(c) Ten rows of the synthetic table on the Magic dataset with Tree-ring watermark

Figure 3.3: Example of synthetic tables on the Magic dataset

age	workclass	fnlwgt	education	education.num	marital.status	occupation	relationship	race	sex	capital.gain	capital.loss	hours.per.week	native.country	income
74	Private	142912.86	Doctorate	16	Never-married	Handlers-cleaners	Not-in-family	White	Male	0	0	1.7616923	United-States	<=50K
28	Private	92000.26	HS-grad	9	Never-married	Machine-op-inspct	Own-child	Black	Female	0	0	40.0	United-States	<=50K
39	Private	168469.9	HS-grad	9	Married-civ-spouse	Sales	Husband	White	Male	0	0	40.0	United-States	>50K
31	Private	185862.73	Masters	14	Never-married	Handlers-cleaners	Unmarried	White	Male	0	0	40.0	United-States	<=50K
12	Self-emp-not-inc	55883.098	12th	9	Married-civ-spouse	Adm-clerical	Husband	White	Male	0	0	40.0	United-States	<=50K
89	Private	107622.016	Bachelors	13	Never-married	Prof-specialty	Not-in-family	White	Female	0	0	45.0	United-States	>50K
25	Private	23580.691	HS-grad	9	Separated	Craft-repair	Not-in-family	White	Male	0	0	40.0	United-States	<=50K
36	Private	204678.81	HS-grad	5	Married-civ-spouse	Exec-managerial	Husband	White	Male	0	0	38.0	United-States	<=50K
43	Private	209074.66	HS-grad	9	Never-married	Other-service	Not-in-family	White	Female	0	0	40.0	United-States	<=50K
72	State-gov	20041.28	Masters	14	Married-civ-spouse	Exec-managerial	Husband	White	Male	0	0	70.0	United-States	>50K
90	Private	60455.375	HS-grad	9	Never-married	Other-service	Own-child	White	Female	0	0	2.952189	United-States	<=50K

(a) Ten rows of the synthetic table on the Adult dataset without watermark

age	workclass	fnlwgt	education	education.num	marital.status	occupation	relationship	race	sex	capital.gain	capital.loss	hours.per.week	native.country	income
19	Private	201545.2	Some-college	10	Never-married	Handlers-cleaners	Not-in-family	White	Male	0	0	7.9444246	United-States	<=50K
17	Private	102290.56	11th	7	Never-married	Adm-clerical	Own-child	Black	Female	0	0	25.0	United-States	<=50K
25	Private	197866.8	HS-grad	9	Married-civ-spouse	Sales	Husband	White	Male	0	0	40.0	United-States	>50K
21	Private	213089.47	Assoc-acdm	12	Never-married	Handlers-cleaners	Unmarried	White	Male	0	0	40.0	United-States	<=50K
17	Federal-gov	97476.79	Assoc-voc	10	Married-civ-spouse	Sales	Other-relative	White	Male	0	0	50.0	United-States	<=50K
28	Private	126329.79	Bachelors	13	Never-married	Prof-specialty	Not-in-family	White	Female	0	0	99.0	United-States	>50K
19	Private	22148.027	Some-college	10	Married-civ-spouse	Other-service	Wife	White	Male	0	0	40.0	United-States	<=50K
21	Private	200636.6	HS-grad	9	Married-civ-spouse	Other-service	Husband	White	Male	0	0	40.0	United-States	<=50K
22	Private	235846.14	HS-grad	9	Never-married	Other-service	Not-in-family	White	Female	0	0	40.0	United-States	<=50K
28	State-gov	23695.045	Assoc-acdm	12	Married-civ-spouse	Exec-managerial	Husband	White	Male	0	0	99.0	United-States	<=50K
18	Private	30012.25	10th	7	Never-married	Other-service	Own-child	White	Female	0	0	40.0	Mexico	<=50K

(b) Ten rows of the synthetic table on the Adult dataset with **Ripple** watermark

age	workclass	fnlwgt	education	education.num	marital.status	occupation	relationship	race	sex	capital.gain	capital.loss	hours.per.week	native.country	income
45	Private	683944.0	11th	14	Married-civ-spouse	Farming-fishing	Not-in-family	White	Male	0	0	40.0	United-States	<=50K
43	Self-emp-inc	189298.31	12th	4	Married-civ-spouse	Exec-managerial	Husband	White	Male	0	0	40.0	United-States	>50K
90	Self-emp-inc	46515.07	HS-grad	9	Widowed	Sales	Not-in-family	Asian-Pac-Islander	Female	0	0	1.0001146	United-States	<=50K
74	Private	262124.05	HS-grad	9	Married-civ-spouse	Sales	Husband	White	Male	0	0	20.0	United-States	>50K
29	Private	78480.125	Bachelors	13	Never-married	Adm-clerical	Unmarried	White	Female	0	0	40.0	United-States	<=50K
31	Private	185271.81	11th	6	Never-married	Transport-moving	Own-child	White	Male	0	0	60.0	Honduras	<=50K
43	State-gov	45369.926	Masters	10	Married-civ-spouse	Prof-specialty	Wife	White	Male	0	0	40.0	United-States	<=50K
38	Federal-gov	12628.181	Bachelors	14	Never-married	Prof-specialty	Unmarried	White	Female	0	0	40.0	United-States	<=50K
47	Self-emp-not-inc	24133.389	Some-college	9	Married-civ-spouse	Tech-support	Own-child	White	Male	0	0	40.0	Honduras	<=50K
26	Private	225767.98	HS-grad	9	Married-civ-spouse	Sales	Not-in-family	White	Male	0	0	20.0	United-States	<=50K
27	Private	427466.12	Bachelors	13	Married-civ-spouse	Prof-specialty	Husband	White	Male	0	0	99.0	United-States	<=50K

(c) Ten rows of the synthetic table on the Adult dataset with Tree-ring watermark

Figure 3.4: Example of synthetic tables on the Adult dataset

LIMIT_BAL	SEX	EDUCATION	MARRIAGE	AGE	PAY_0	PAY_2	PAY_3	PAY_4	PAY_5	PAY_6	BILL_AMT1	BILL_AMT2	BILL_AMT3	BILL_AMT4	BILL_AMT5	BILL_AMT6	PAY_AMT1	PAY_AMT2	PAY_AMT3	PAY_AMT4	PAY_AMT5	PAY_AMT6	default payment next month
100000.0	1.0	1.0	1.0	33.0	2.0	0.0	0.0	2.0	2.0	0.0	91072.07	90660.79	97961.65	96949.19	82487.48	91973.98	3761.7832	8859.494	0.0	0.0	2987.2556	7000.0	1.0
50000.0	1.0	2.0	2.0	35.0	0.0	0.0	0.0	0.0	0.0	0.0	42065.6	42840.277	28171.018	43986.27	42295.918	41520.81	3072.1812	0.0	2000.0	2000.0	0.0	2000.0	0.0
50000.0	2.0	2.0	1.0	27.0	4.0	-1.0	0.0	0.0	0.0	0.0	8557.187	17124.283	18049.11	110609.83	20210.748	6903.548	316.35168	4500.0	1500.0	69.37378	11000.0	3370.736	0.0
220000.0	1.0	2.0	2.0	28.0	0.0	0.0	0.0	0.0	0.0	0.0	213664.56	212401.3	215268.77	209971.4	205908.88	207556.25	8670.784	9792.453	8001.257	8215.5	6165.684	9006.611	0.0
140000.0	2.0	1.0	1.0	32.0	-1.0	-1.0	-1.0	-1.0	-1.0	-1.0	103.59358	362.39655	390.0	205.76213	380.02808	390.0	166.97078	390.0	390.0	390.0	422.4124	503.69104	0.0
30000.0	1.0	3.0	2.0	27.0	0.0	0.0	0.0	0.0	0.0	0.0	4559.7427	17813.152	77690.09	4917.23	29314.723	-325.5894	27781.266	1839993.12	3500.0	0.0	0.0	0.0	0.0
10000.0	2.0	3.0	1.0	35.0	-1.0	-1.0	0.0	-1.0	0.0	0.0	412.76013	1801.2444	2137.53	2334.1714	3396.526	3878.2593	1880.2987	1159.3396	1800.0	406.7712	601.8035	876.6561	0.0
30000.0	1.0	2.0	2.0	29.0	1.0	2.0	2.0	2.0	2.0	2.0	15877.836	16915.332	19554.615	26416.115	22770.645	22874.314	2372.8005	1102.955	0.0	2000.0	648.63934	1000.0	0.0
110000.0	2.0	2.0	2.0	26.0	-1.0	-1.0	0.0	0.0	0.0	0.0	264.50412	761.49243	1402.1102	1628.6847	780.0	854.3064	1000.0	1000.0	1000.0	0.0	184.93259	401.54663	0.0
100000.0	1.0	3.0	2.0	27.0	0.0	0.0	0.0	0.0	0.0	2.0	47978.812	48416.29	39855.535	34534.81	90439.734	59448.047	39278.434	5900.3003	0.0	11060.513	0.0	4317.272	0.0
9999486.6	1.0	1.0	2.0	58.0	0.0	0.0	0.0	0.0	0.0	0.0	964511.0	983931.0	1663817.0	891524.56	92717.1.0	335812.03	1110314.79	135308.28	66102.7	108745.04	48846.746	0.0	0.0

(a) Ten rows of the synthetic table on the Credit dataset without watermark

LIMIT_BAL	SEX	EDUCATION	MARRIAGE	AGE	PAY_0	PAY_2	PAY_3	PAY_4	PAY_5	PAY_6	BILL_AMT1	BILL_AMT2	BILL_AMT3	BILL_AMT4	BILL_AMT5	BILL_AMT6	PAY_AMT1	PAY_AMT2	PAY_AMT3	PAY_AMT4	PAY_AMT5	PAY_AMT6	default payment next month
80000.0	1.0	1.0	1.0	33.0	2.0	0.0	0.0	2.0	2.0	0.0	71602.25	70928.62	79310.14	79751.05	68528.414	7667.5.7	3014.5173	7463.565	0.0	0.0	2329.8477	6269.427	1.0
38455.24	1.0	2.0	2.0	36.0	0.0	0.0	2.0	0.0	0.0	2.0	30807.248	36416.145	35834.176	36363.33	37751.3	38465.35	4159.132	0.0	1800.0	3225.1611	0.0	2000.0	0.0
30000.0	2.0	2.0	1.0	27.0	-1.0	-1.0	0.0	0.0	0.0	0.0	1804.6637	4648.249	14322.85	28471.4	14296.628	5781.4.76	0.0	3010.6128	1391.5848	308.79	5000.0	2572.9504	0.0
100000.0	1.0	2.0	2.0	30.0	0.0	0.0	0.0	0.0	0.0	0.0	96390.516	91415.69	90646.81	84676.83	82537.77	83839.695	4000.0	5000.0	4000.0	3632.4062	5972.8066	5000.0	0.0
200000.0	2.0	1.0	1.0	34.0	-1.0	-1.0	-1.0	-1.0	-1.0	-1.0	197.93697	390.0	391.8788	326.0	390.0	492.93848	219.2799	396.29715	35.626717	416.16876	626.0029	600.0	0.0
50000.0	2.0	2.0	1.0	29.0	0.0	0.0	0.0	0.0	0.0	0.0	18935.621	17464.39	40647.027	27009.342	15622.388	274.13714	7000.0	71945.28	3000.0	0.0	0.0	0.0	0.0
10000.0	2.0	3.0	1.0	59.0	-2.0	-2.0	-2.0	-2.0	-2.0	0.0	0.0	396.0	390.0	372.6952	390.0	549.8378	358.37967	730.4432	224.0587	510.73883	745.6274	51619.594	0.0
80000.0	1.0	2.0	1.0	35.0	1.0	2.0	2.0	2.0	2.0	2.0	946.342	8548.105	19941.62	45513.957	25297.838	12294.512	1977.5104	1427.9312	0.0	2100.0	50.777027	1000.0	0.0
230000.0	2.0	2.0	2.0	28.0	-1.0	-1.0	-1.0	-1.0	0.0	-1.0	316.0	390.0	397.90817	780.0	320.6721	479.4576	262.14572	447.51248	1000.0	0.0	565.6138	689.44794	0.0
100000.0	1.0	3.0	2.0	28.0	0.0	0.0	0.0	0.0	0.0	2.0	53967.53	114432.41	47753.465	31880.475	80112.71	55920.875	18867.768	5000.0	0.0	10000.0	0.0	5000.0	0.0
10000000.0	1.0	1.0	2.0	60.0	0.0	0.0	0.0	0.0	0.0	0.0	964511.0	983931.0	1664089.0	891539.3	92717.1.0	705982.56	133273.56	209266.42	133141.95	188432.08	40853.543	27854.514	0.0

(b) Ten rows of the synthetic table on the Credit dataset with **Ripple** watermark

LIMIT_BAL	SEX	EDUCATION	MARRIAGE	AGE	PAY_0	PAY_2	PAY_3	PAY_4	PAY_5	PAY_6	BILL_AMT1	BILL_AMT2	BILL_AMT3	BILL_AMT4	BILL_AMT5	BILL_AMT6	PAY_AMT1	PAY_AMT2	PAY_AMT3	PAY_AMT4	PAY_AMT5	PAY_AMT6	default payment next month
110000.0	1.0	1.0	1.0	33.0	0.0	0.0	0.0	0.0	-1.0	0.0	23667.592	0.0	0.0	3870.639	149453.75	0.0	0.0	5729.6626	2000.0	498.28143	0.0	1000.0	0.0
20000.0	2.0	2.0	2.0	22.0	0.0	0.0	0.0	0.0	0.0	0.0	14584.515	14996.015	15331.722	14646.04	13629.849	12767.647	1434.4639	1352.5215	925.73517	591.4808	769.27747	787.0758	0.0
90000.0	2.0	2.0	1.0	26.0	0.0	2.0	0.0	0.0	0.0	0.0	90977.555	101824.27	90770.21	85061.28	94324.984	53848.86	2001.9437	3696.651	3045.161	1841.2418	2500.0	2400.7705	0.0
160000.0	1.0	1.0	1.0	31.0	0.0	0.0	0.0	0.0	0.0	0.0	63282.21	37648.312	42846.58	70254.734	79041.92	60226.613	10971.515	2601.0754	6839.0987	486.49402	0.0	4378.6763	0.0
290000.0	1.0	2.0	1.0	48.0	0.0	0.0	0.0	0.0	0.0	0.0	69723.15	15370.212	92426.51	55315.99	20357.893	1373.82	3019.7297	1296.8167	6519.47	0.0	1300.0	0.0	0.0
1465423.53	2.0	1.0	1.0	41.0	-2.0	-2.0	-2.0	-2.0	-2.0	2.0	1918.2117	5380.3177	2010.2666	55353.016	19853.752	15708.672	2000.0	4000.0	4000.0	6045.6357	327439.9	113180.09	0.0
743007.2	2.0	1.0	1.0	50.0	0.0	0.0	0.0	0.0	0.0	0.0	951025.0	974905.0	1620647.9	882448.5	920895.56	373505.06	43935.117	49489.5	30000.0	46144.082	33114.45	43712.816	1.0
290000.0	2.0	2.0	2.0	29.0	0.0	0.0	0.0	0.0	0.0	0.0	203915.3	181392.14	146728.47	155547.19	156579.28	198783.83	4675.71	5000.0	5005.544	4082.1897	5032.9604	9625.043	0.0
50000.0	2.0	2.0	1.0	55.0	2.0	2.0	2.0	0.0	0.0	-1.0	450.97476	221.83966	150.35072	0.0	-79720.67	148.38493	0.0	1399.4729	0.0	0.0	9198.504	8451.303	1.0
30000.0	2.0	3.0	1.0	30.0	2.0	2.0	0.0	0.0	0.0	0.0	1677.24	21325.828	20443.957	29473.66	28696.158	32560.898	1400.5197	1500.0	3000.0	3300.6133	1000.1316	0.0	0.0
80000.0	2.0	1.0	1.0	31.0	0.0	0.0	0.0	0.0	0.0	0.0	2406.2847	17324.477	73509.12	19861.025	18368.504	39884.426	0.0	2500.0	5404.487	12887.499	0.0	1000.0	0.0

(c) Ten rows of the synthetic table on the Credit dataset with Tree-ring watermark

Figure 3.5: Example of synthetic tables on the Credit dataset

4

Conclusion

The generation of synthetic data using advanced generative models is a transformative method for augmenting and disseminating data, providing significant advantages for machine learning tasks and data privacy. Ensuring the traceability and auditability of synthetic data is crucial to prevent misuse and potential harm. While extensive research has developed watermarking techniques for synthetic images and texts, a notable gap exists in the application of these techniques to synthetic tabular data. To address this gap, this thesis answers three critical research questions (RQs):

- **RQ1:** Can a novel watermark be designed for synthetic tables such that it spreads across the entire table while remaining imperceptible?
- **RQ2:** What are the possible post-editing attacks on watermarks in synthetic tabular data?
- **RQ3:** To what extent does the tabular watermark achieve the three goals of watermarking: i) maintaining high synthetic data quality, ii) ensuring high detectability, and iii) remaining highly robust to post-editing attacks?

This work addresses these critical challenges by introducing `Ripple`, a novel watermarking algorithm specifically designed for latent tabular diffusion models. The contributions of this work are as follows:

Contribution 1: `Ripple` is the first watermarking algorithm tailored for tabular diffusion models during the sampling phase. By embedding `Ripple` watermarks across a broad spectrum in the Fourier space of the initial noise matrix during sampling, it enables watermarking across cells in the synthetic table, significantly increasing detectability.

Contribution 2: Six post-editing attacks specific to synthetic tables are developed for robustness evaluation against `Ripple` watermarking. These attacks involve the deletion and distortion of columns, rows, and cells.

Contribution 3: Comprehensive evaluation on four diverse datasets demonstrates `Ripple`'s effectiveness in preserving synthetic data quality. The average quality difference between watermarked and non-watermarked data is less than 0.6%, indicating that `Ripple` maintains the resemblance, discriminability, and downstream utility of the synthetic tables. In terms of detectability, `Ripple` achieves remarkable results.

The statistical p-values for watermark detection are, on average, more than 25 times lower than 0.02 across all datasets, showing an improvement of more than 2 orders of magnitude compared to the tree-ring watermarking baseline. Additionally, `Ripple` exhibits strong robustness against various post-editing attacks, including the deletion and distortion of rows, columns, and values. Our robustness analysis shows that 85% of the p-values remain below 0.05 across different attack settings, demonstrating `Ripple` 's ability to maintain watermark integrity under malicious conditions.

By effectively embedding watermarks while preserving data quality and ensuring high detectability and robustness, `Ripple` sets a new standard for traceability and auditability in synthetic tabular data. This advancement is particularly significant as the use of synthetic data continues to expand across various fields, including healthcare, finance, and social sciences. Despite its demonstrated effectiveness, `Ripple` opens several promising avenues for future research. One potential direction involves developing more fine-grained watermarking techniques that operate at the sample or single row level, thereby enhancing watermark detectability and robustness. Additionally, exploring the application of `Ripple` to other types of generative models, or creating a generic watermarking scheme applicable to all types of generative tabular models, would significantly broaden the impact and utility of this watermarking approach. In summary, `Ripple` paves the way for safer and more reliable use of synthetic data in various applications, contributing to the advancement of data management and artificial intelligence.

Bibliography

[1] Online deepfake maker. `https://deepfakesweb.com/`. Accessed: 2024-06-08.

[2] Scott Aaronson. My ai safety lecture for ut effective altruism, 2022.

[3] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[4] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223. PMLR, 2017.

[5] Thera Bank. Bank Loan Modelling, 2017.

[6] Barry Becker and Ronny Kohavi. Adult. UCI Machine Learning Repository, 1996. DOI: https://doi.org/10.24432/C5XW20.

[7] Lisa Bode, Dominic Lees, and Daniel Golding. Editorial the digital face and deepfakes on screen. *Convergence: The International Journal of Research into New Media Technologies*, 27(4):135485652110340–854, 2021.

[8] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. In *International Conference on Learning Representations*, 2018.

[9] Guangyu Chen, Yu Wu, Shujie Liu, Tao Liu, Xiaoyong Du, and Furu Wei. Wavmark: Watermarking for audio generation. *arXiv preprint arXiv:2308.12770*, 2023.

[10] Kuzak Dempsy. Cardiovascular Disease Dataset, 2021.

[11] Laurent Dinh, David Krueger, and Yoshua Bengio. Nice: Non-linear independent components estimation. *arXiv preprint arXiv:1410.8516*, 2014.

[12] Andrew Gelman, John B Carlin, Hal S Stern, and Donald B Rubin. *Bayesian data analysis*. Chapman and Hall/CRC, 1995.

[13] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 27, 2014.

[14] Jeffrey T Hancock and Jeremy N Bailenson. The social impact of deepfakes, 2021.

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

[16] Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. In *International conference on learning representations*, 2016.

[17] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2020.

[18] Felix Juefei-Xu, Run Wang, Yihao Huang, Qing Guo, Lei Ma, and Yang Liu. Countering malicious deepfakes: Survey, battleground, and horizon. *International journal of computer vision*, 130(7):1678–1734, 2022.

[19] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022.

[20] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *giomi2023unifiedt*, 1050:1, 2014.

[21] 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.

[22] 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.

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

[24] Nils Lukas, Edward Jiang, Xinda Li, and Florian Kerschbaum. Sok: How robust is image classification deep neural network watermarking? In *2022 IEEE Symposium on Security and Privacy (SP)*, pages 787–804. IEEE, 2022.

[25] Nils Lukas and Florian Kerschbaum. {PTW}: Pivotal tuning watermarking for {Pre-Trained} image generators. In *32nd USENIX Security Symposium (USENIX Security 23)*, pages 2241–2258, 2023.

[26] Geoffrey J McLachlan and Kaye E Basford. *Mixture models: Inference and applications to clustering*, volume 38. M. Dekker New York, 1988.

[27] Roger B Nelsen. *An introduction to copulas*. Springer, 2006.

[28] George Papamakarios, Eric Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan. Normalizing flows for probabilistic modeling and inference. *The Journal of Machine Learning Research*, 22(1):2617–2680, 2021.

[29] Walter HL Pinaya, Mark S Graham, Eric Kerfoot, Petru-Daniel Tudosiu, Jessica Dafflon, Virginia Fernandez, Pedro Sanchez, Julia Wolleb, Pedro F da Costa, Ashay Patel, et al. Generative ai for medical imaging: extending the monai framework. *arXiv preprint arXiv:2307.15208*, 2023.

[30] Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *International conference on machine learning*, pages 1530–1538. PMLR, 2015.

[31] 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.

[32] Marco Schreyer, Timur Sattarov, Bernd Reimer, and Damian Borth. Adversarial learning of deepfakes in accounting. *arXiv preprint arXiv:1910.03810*, 2019.

[33] Huajie Shao, Zhisheng Xiao, Shuochao Yao, Dachun Sun, Aston Zhang, Shengzhong Liu, Tianshi Wang, Jinyang Li, and Tarek Abdelzaher. Controlvae: Tuning, analytical properties, and performance analysis. *IEEE transactions on pattern analysis and machine intelligence*, 44(12):9285–9297, 2021.

[34] Jascha Sohl-Dickstein, Eric A. Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep Unsupervised Learning using Nonequilibrium Thermodynamics. In *Proceedings of the 32nd International Conference on Machine Learning*, *ICML*, volume 37, pages 2256–2265, 2015.

[35] Synthetic Data Vault. Copulagan synthesizer documentation, 2023.

[36] Gemini Team, Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

[37] Romal Thoppilan, Daniel De Freitas, Jamie Hall, Noam Shazeer, Apoorv Kulshreshtha, Heng-Tze Cheng, Alicia Jin, Taylor Bos, Leslie Baker, Yu Du, et al. Lamda: Language models for dialog applications. *arXiv preprint arXiv:2201.08239*, 2022.

[38] Luís Torgo. California Housing Prices, 1990.

[39] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

[40] Yuxin Wen, John Kirchenbauer, Jonas Geiping, and Tom Goldstein. Tree-rings watermarks: Invisible fingerprints for diffusion images. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[41] Cheng Xiong, Chuan Qin, Guorui Feng, and Xinpeng Zhang. Flexible and secure watermarking for latent diffusion model. In *Proceedings of the 31st ACM International Conference on Multimedia*, pages 1668–1676, 2023.

[42] Lei Xu, Maria Skoularidou, Alfredo Cuegiomi2023unified-Infante, and Kalyan Veeramachaneni. Modeling tabular data using conditional gan. *Advances in neural information processing systems*, 32, 2019.

[43] Yilun Xu, Ziming Liu, Max Tegmark, and Tommi Jaakkola. Poisson flow generative models. *Advances in Neural Information Processing Systems*, 35:16782–16795, 2022.

[44] Jinsung Yoon, Lydia N Drumright, and Mihaela Van Der Schaar. Anonymization through data synthesis using generative adversarial networks (ads-gan). *IEEE journal of biomedical and health informatics*, 24(8):2378–2388, 2020.

[45] Ning Yu, Vladislav Skripniuk, Sahar Abdelnabi, and Mario Fritz. Artificial fingerprinting for generative models: Rooting deepfake attribution in training data. In *Proceedings of the IEEE/CVF International conference on computer vision*, pages 14448–14457, 2021.

[46] Ning Yu, Vladislav Skripniuk, Dingfan Chen, Larry S Davis, and Mario Fritz. Responsible disclosure of generative models using scalable fingerprinting. In *International Conference on Learning Representations*, 2021.

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

[48] Kevin Alex Zhang, Alfredo Cuesta-Infante, Lei Xu, and Kalyan Veeramachaneni. Steganogan: High capacity image steganography with gans. *arXiv preprint arXiv:1901.03892*, 2019.

[49] Kevin Alex Zhang, Lei Xu, Alfredo Cuesta-Infante, and Kalyan Veeramachaneni. Robust invisible video watermarking with attention. *arXiv preprint arXiv:1909.01285*, 2019.

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

[51] Yunqing Zhao, Tianyu Pang, Chao Du, Xiao Yang, Ngai-Man Cheung, and Min Lin. A recipe for watermarking diffusion models. *arXiv preprint arXiv:2303.10137*, 2023.

[52] Chaoyi Zhu, Jeroen Galjaard, Pin-Yu Chen, and Lydia Y Chen. Duwak: Dual watermarks in large language models. *arXiv preprint arXiv:2403.13000*, 2024.

[53] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017.