Tree-Ring Watermarks: Fingerprints for Diffusion Images that are Invisible and Robust

(LINK)

Yuxin Wen, John Kirchenbauer, Jonas Geiping, Tom Goldstein

Group 7
Chaitanya Sethi (2020B3A71961P)
Dhawal Mehta (2020B3A71965P)
Suraj Phalod (2020B3A71959P)

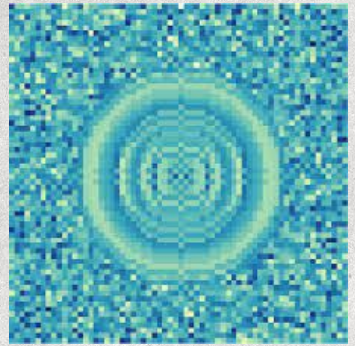
Table of contents

01

Abstract

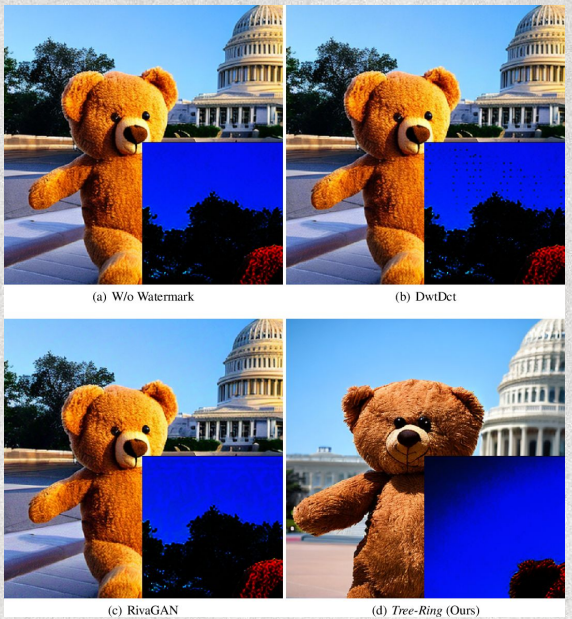
Tree-Ring Watermarks

Abstract



Watermarking the outputs of generative models is a crucial technique for **tracing copyright and preventing potential harm from AI-generated content**. In this paper, we introduce a novel technique called **Tree-Ring Watermarking** that robustly fingerprints diffusion model outputs. Unlike **existing methods that perform post-hoc modifications** to images after sampling, Tree-Ring Watermarking subtly influences the entire sampling process, resulting in a model fingerprint that is **invisible to humans**.

Abstract



(a) W/o Watermark (b) DwtDct

(c) RivaGAN (d) *Tree-Ring* (Ours)

The watermark **embeds a pattern into the initial noise vector used for sampling**. These patterns are structured in **Fourier space** so that they are invariant to convolutions, crops, dilations, flips, and rotations. After image generation, **the watermark signal is detected by inverting the diffusion process to retrieve the noise vector**, which is then checked for the embedded signal. We demonstrate that **this technique can be easily applied to arbitrary diffusion models**, including **text-conditioned Stable Diffusion**, as a plug-in with negligible loss in FID. Our watermark is semantically hidden in the image space and is **far more robust than watermarking alternatives that are currently deployed**.

02

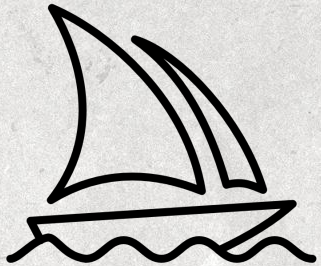
Introduction

Tree-Ring Watermarks

The Background

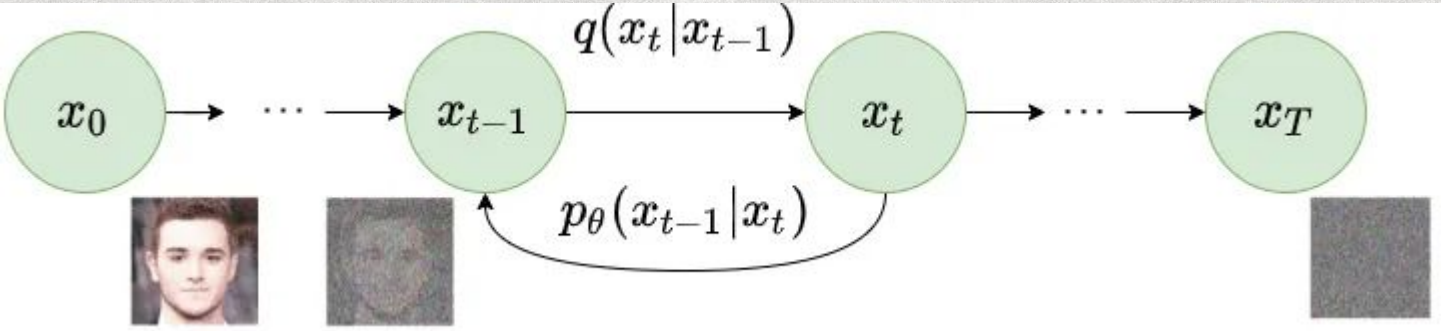
The development of diffusion models has led to a surge in image generation quality. Modern text to-image diffusion models, like **Stable Diffusion and Midjourney**, are capable of generating a wide variety of novel images in an innumerable number of styles. These systems are general-purpose image generation tools, able to generate new art just as well as photo-realistic depictions of fake events for malicious purposes.

Research and applications of watermarking for digital content have a long history, with many approaches being considered over the last decade. However, so far **research has always conceptualized the watermark as a minimal modification imprinted onto an existing image**

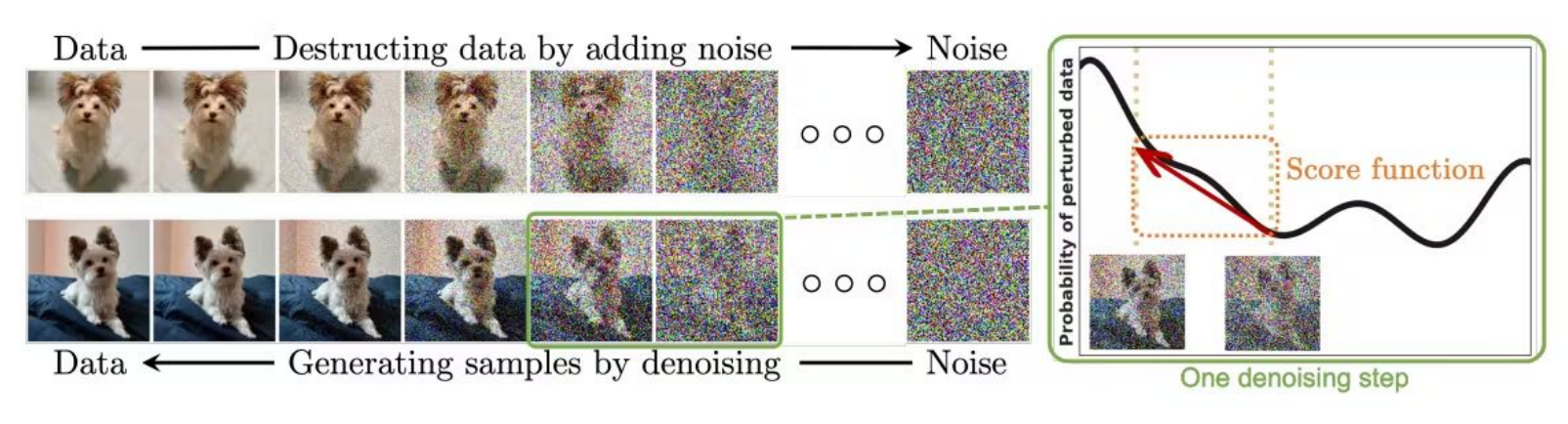


What do diffusion models do?

Diffusion models are a class of generative AI models that **generate high-resolution images of varying quality**. They work by **gradually adding Gaussian noise to the original data** in the **forward diffusion process** and then l**earning to remove the noise in the reverse diffusion process**. They are latent variable models referring to a hidden continuous feature space, look similar to VAEs(Variational Autoencoders), and are loosely based on non-equilibrium thermodynamics.

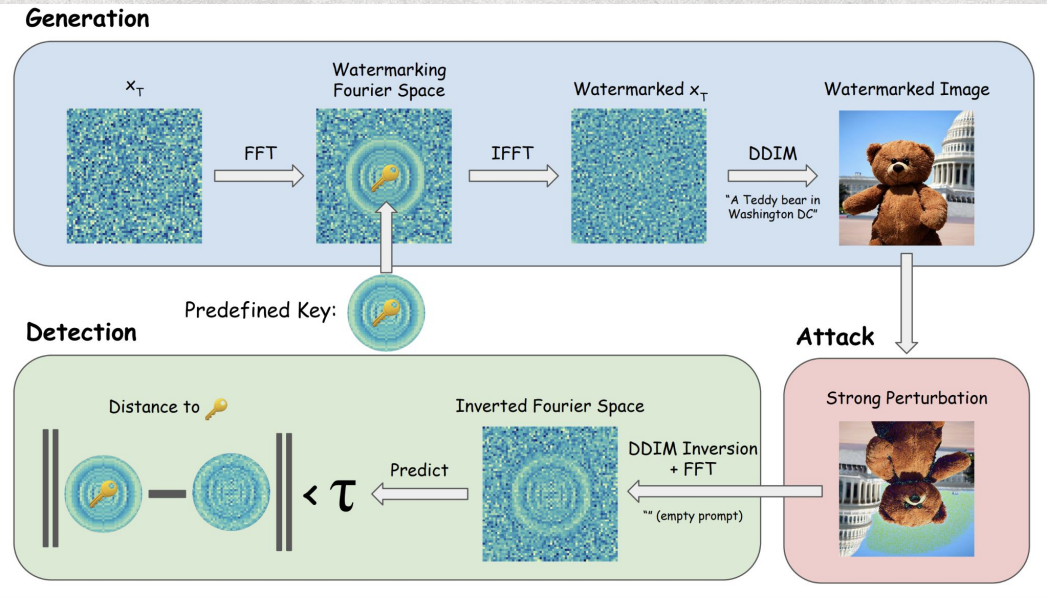


Another visual for diffusion



The Approach

The watermarking approach we propose in this work is conceptually different: This is the first watermark that is **truly invisible**, as **no post-hoc modifications** are made to the image. Instead, the distribution of generated images is imperceptibly modified and an **image is drawn from this modified distribution**. This way, the actual sample carries no watermark in the classical additive sense, **however an algorithmic analysis of the image can detect the watermark with high accuracy**. From a more practical perspective, the watermark materializes in minor changes in the potential layouts of generated scenes, that cannot be distinguished from other random samples by human inspection.



The features

This new approach to watermarking, which we call Tree-Ring Watermarking b**ased on the patterns imprinted into the Fourier space of the noise vector of the diffusion model**, can be **easily incorporated** into existing diffusion model APIs and is invisible on a per-sample basis. Most importantly, Tree-Ring Watermarking is **far more robust** than existing methods against a large battery of common image transformations, such as **crops, color jitter, dilation, flips, rotations, or noise**.

Tree-Ring Watermarking requires **no additional training or fine tuning to implement**, and the watermark can only be detected by parties in control of the image generation model. We validate the watermark in a number of tests, measuring **negligible impact on image quality scores, high robustness to transformations, the low false-positive rate in detection, and usability for arbitrary diffusion models both with and without text conditioning.**

(a) 75° Rotation (b) 25% JPEG Compression (c) 75% Random Cropping + Scaling

(d) Gaussian Blurring with an 8 × 8 Filter (e) Gaussian Noise with $\sigma = 0.1$ (f) Color Jitter with a Brightness Factor of 6

The Setup

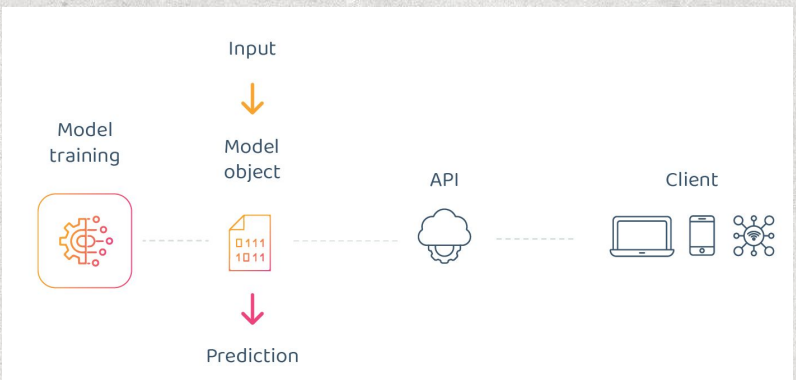
The aim is to identify whether an image was produced by one of the models.

The setup is that the model is going to be behind an API, which when called, will help people generate diffusion images.

The weights of the model need to be hidden, for obvious reasons.

If watermarking is done correctly, we can always prove that this image was generated by a particular model, by embedding the indistinguishable watermark.

The watermark is very difficult to get rid off, even with very strong perturbations.



03

Related Work

Tree-Ring Watermarks

Literature review - Diffusion Models

Paper 1: Generative Modeling by Estimating Gradients | Yang Song, Stefano Ermon

- Explores generative models using denoising auto-encoders and Laplacian eigenmaps for dimensionality reduction.
- Focuses on the theoretical aspects of generative modeling and its applications in image processing.

Paper 2: Improved Denoising Diffusion Probabilistic Models (DDPM) | Alex Nichol, P. Dhariwal

- Compares diffusion models with GANs using precision and recall metrics, showing higher recall for diffusion models.
- Discusses the benefits of diffusion models over GANs in covering modes of distribution
- Emphasizes DDIM Sampling as the most prominent sampling algorithm.

Comparison:

The main paper's focus on invisible watermarking through Tree-Ring Watermarking distinguishes it from Paper 2, which emphasizes performance metrics, and Paper 1, which delves into dimensionality reduction techniques.

Literature Review - Watermarking Digital Content

Paper 1: Watermarking digital images for copyright protection | JJKO Ruanaidh, W.J. Dowling et al.

- Discusses watermarking digital images for copyright protection, highlighting challenges in unauthorized copying and the limitations of encryption systems like RSA.
- Discusses algorithms such as:
  - ***Block-Mean Approach*** **-** Division of image into blocks, the mean of which is manipulated
    - **Bidirectional Coding:** Encode '1' (increment) or decode to '0' (decrement)
    - **Unidirectional Coding:** Encode '1' (increment) or leave unchanged
  - ***Transform Domain Watermark*** **-** Allows for adaptive bits positioning within Image Block
    - **Step I:** Simple form of Modulation for bits placement
    - **Step II:** Technique to determine No. of bits to be placed at given locations in image described.
- The paper further discusses reliable communication under Gaussian Noise assumption,

Paper 2: A Method for Watermark Casting on Digital Images | Ionannis Pitas

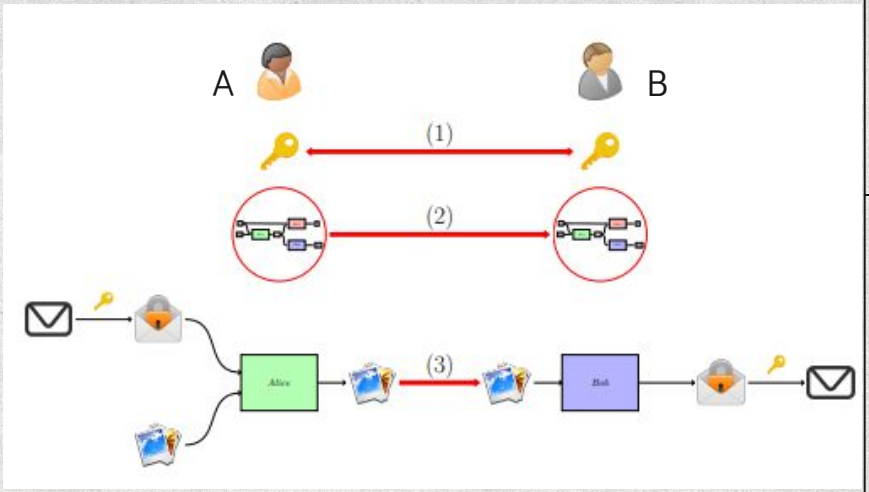
- Predetermined small luminance value (undetectable by eye) is added to randomly selected image pixels, by altering intensity levels of pixels in subsets of the Image using a binary pattern containing equal '1s' and '0s' (in pixels, defined as the Digital Watermark S)
- The seed of the random pixel generator is essentially the copyright holder watermark.

$$S = \{s_{nm},\, n \in \{0, 1, \cdots, N-1\},\, m \in \{0, 1, \cdots, M-1\}\}$$

Literature review- Fingerprinting and Watermarking Generative Models

Paper 1: Generating Steganographic Images via adversarial training | Jamie Hayes, George Danezis

1) 2 Parties establish a shared key
2) **A** trains the scheme on a set of images, whose model info. Is encrypted in the shared key. (**Fingerprint** encoding)
3) Key is sent to **B,** who decrypts the local copy of the models.
4) **A** uses **Green** model to embed a secret encrypted message (Stenographic Image creation) which is sent to **B** who uses **Purple** model to decode and decrypt.



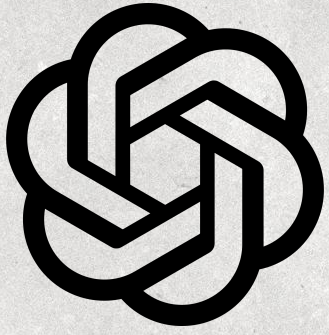
Paper 2: HiDDeN: Hiding Data with Deep Networks | J. Zhu, R. Kaplan, J. Johnson, Li Fei-Fei

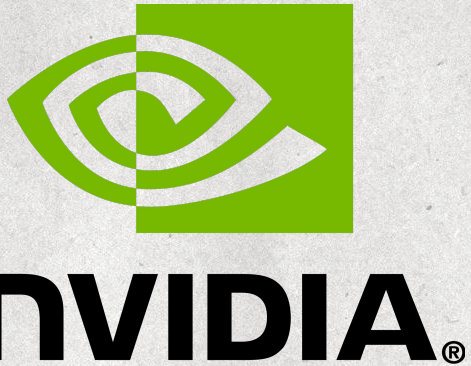
- Discusses hiding data in generative models - using trained (optimized on adversarial objectives) for maximizing transmission and robustness
- Explores methods for concealing data within generative models to protect sensitive information.

How Diffusion Models work

Diffusion models are a new class of state-of-the-art generative models that generate diverse high-resolution images. They have already attracted a lot of attention after OpenAI, Nvidia and Google managed to train large-scale models.

Diffusion models are fundamentally different from all the previous generative methods. Intuitively, they aim to decompose the image generation process (sampling) in many small “denoising” steps.





Forward Diffusion

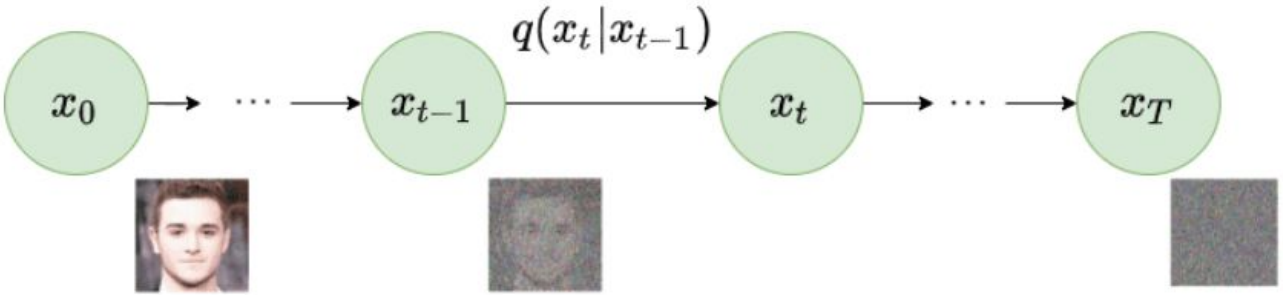
The basic idea behind diffusion models is rather simple. They take the **input image Xo and gradually add Gaussian noise to it through a series of T steps**. We will call this the forward process. Notably, this is unrelated to the forward pass of a neural network. If you'd like, this part is necessary to generate the targets for our neural network (the image after applying t<T noise steps).

In practice, they are formulated using a **Markov chain of T steps**. Here, a Markov chain means that each step only depends on the previous one, which is a mild assumption. Importantly, we are not constrained to using a specific type of neural network, unlike flow-based models.

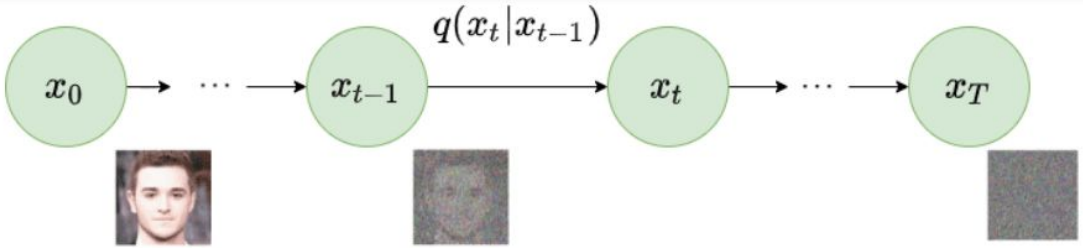
Forward Diffusion (continued)

Given a **data-point x_0 sampled from the real data distribution q(x) (x_0~q(x))**, one can define a forward diffusion process by adding noise. Specifically, at each step of the Markov chain we add Gaussian noise with **variance β_t** to x_t-1, producing a new latent variable **x_t with distribution $q(x|x_{t-1})$**. This diffusion process can be formulated as follows:

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



Forward diffusion process. Image modified by Ho et al. 2020



Forward diffusion process. Image modified by Ho et al. 2020

Since we are in the multi-dimensional scenario $\mathbf{I}$ is the identity matrix, indicating that each dimension has the same standard deviation β_t. Note that $q(\mathbf{x}_t|\mathbf{x}_{t-1})$ is still a normal distribution, defined by the mean $\boldsymbol{\mu}$ and the variance $\boldsymbol{\Sigma}$ where $\boldsymbol{\mu}_t = \sqrt{1-\beta_t}\mathbf{x}_{t-1}$ and $\boldsymbol{\Sigma}_t = \beta_t\mathbf{I}$. $\boldsymbol{\Sigma}$ will always be a diagonal matrix of variances (here β_t)

Thus, we can go in a closed form from the input data $\mathbf{x}_0$ to $\mathbf{x}_T$ in a tractable way. Mathematically, this is the posterior probability and is defined as:

$$q(\mathbf{x}_{1:T}|\mathbf{x}_0) = \prod_{t=1}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1})$$

The symbol : in $q(\mathbf{x}_{1:T})$ states that we apply q repeatedly from timestep 1 to T. It's also called trajectory.

Forward Diffusion (closed form sampling)

If we define $\alpha_t = 1 - \beta_t$, $\bar{\alpha}_t = \prod_{s=0}^{t} \alpha_s$ where $\epsilon_0, ..., \epsilon_{t-2}, \epsilon_{t-1} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, one can use the reparameterization trick in a recursive manner to prove that:

$$\begin{aligned}
\mathbf{x}_t &= \sqrt{1-\beta_t}\mathbf{x}_{t-1} + \sqrt{\beta_t}\boldsymbol{\epsilon}_{t-1} \\
&= \sqrt{\alpha_t}\mathbf{x}_{t-2} + \sqrt{1-\alpha_t}\boldsymbol{\epsilon}_{t-2} \\
&= \dots \\
&= \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_0
\end{aligned}$$

> Note: Since all timestep have the same Gaussian noise we will only use the symbol $\boldsymbol{\epsilon}$ from now on.

Thus to produce a sample $\mathbf{x}_t$ we can use the following distribution:

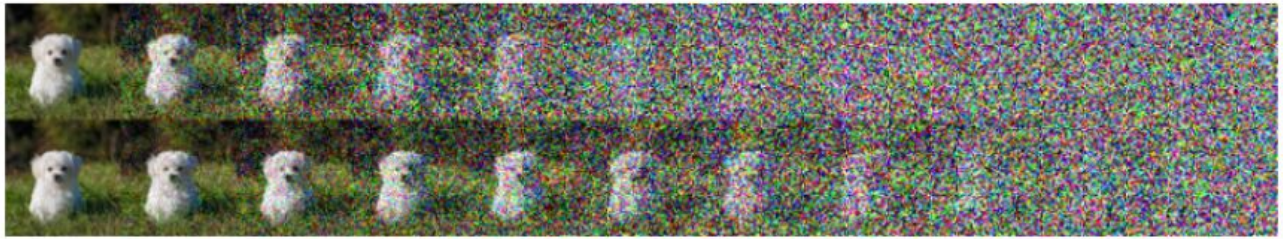
$$\mathbf{x}_t \sim q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1-\bar{\alpha}_t)\mathbf{I})$$

Since β_t is a hyperparameter, we can precompute α_t and $\bar{\alpha}_t$ for all timesteps. This means that we sample noise at any timestep t and get $\mathbf{x}_t$ in one go. Hence, we can sample our latent variable $\mathbf{x}_t$ at any arbitrary timestep. This will be our target later on to calculate our tractable objective loss L_t.

Reference
https://theaisummer.com/diffusion-models/

Variance Schedule

The variance parameter **βt** can be **fixed to a constant or chosen as a schedule over the T timesteps**. In fact, one can define a variance schedule, which can be linear, quadratic, cosine etc. The original **DDPM authors utilized a linear schedule** increasing from β1=10^−4 to βT=0.02. Nichol et al. 2021 showed that employing a **cosine schedule** works even better.



Latent samples from linear (top) and cosine (bottom) schedules respectively. Source: Nichol & Dhariwal 2021

Reverse Diffusion

Afterward, a neural network is trained to recover the original data by reversing the noising process. By being able to model the reverse process, we can generate new data. This is the so-called reverse diffusion process or, in general, the sampling process of a generative model.

For the reverse diffusion process, DDIM [Song and Ermon, 2020] is an efficient deterministic sampling strategy, mapping from a Gaussian vector $x_T \sim \mathcal{N}(0, 1)$ to an image $x_0 \in q(x)$. For each denoising step, a learned noise-predictor ϵ_θ estimates the noise $\epsilon_\theta(x_t)$ added to x_0. According to Equation (1), we can derive the estimation of x_0 as:

$$\hat{x}_0^t = \frac{x_t - \sqrt{1-\bar{\alpha}_t}\epsilon_\theta(x_t)}{\sqrt{\bar{\alpha}_t}}.$$

Then, we add the estimated noise to $\hat{x}_0$ to find x_{t-1}:

$$x_{t-1} = \sqrt{\bar{\alpha}_{t-1}}\hat{x}_0^t + \sqrt{1-\bar{\alpha}_{t-1}}\epsilon_\theta(x_t).$$

We denote such a recursively denoising process from x_T to x_0 as $x_0 = \mathcal{D}_\theta(x_T)$.

Reverse Diffusion

However, given the learned model $\epsilon_\theta(x_t)$, it is also possible to move in the opposite direction[1]. Starting from an image x_0, Dhariwal and Nichol [2021] describes an inverse process that retrieves an initial noise vector x_T which maps to an image $\hat{x}_0$ close to x_0 through DDIM, where $\hat{x}_0 = \mathcal{D}_\theta(x_T, 0) \approx x_0$. This inverse process depends on the assumption that $x_{t-1} - x_t \approx x_{t+1} - x_t$. Therefore, from $x_t \to x_{t+1}$, we follow:

$$x_{t+1} = \sqrt{\bar{\alpha}_{t+1}}\hat{x}_0^t + \sqrt{1 - \bar{\alpha}_{t+1}}\epsilon_\theta(x_t).$$

We denote the whole inversion process from a starting real image x_0 to x_T as $x_T = \mathcal{D}_\theta^\dagger(x_0)$.

In this work, we **re-purpose DDIM inversion $D^{\dagger}_{\theta}$ for watermark detection**. Given a generated image x_0 with a starting noise x_T, we apply DDIM inversion to find $\hat{}xT$. We empirically find DDIM's inversion performance to be quite strong, and $\hat{}xT \approx xT$. While it may not be surprising that inversion is accurate for unconditional diffusion models, inversion also succeeds well-enough for conditional diffusion models, even when the conditioning c is not provided. This property of inversion will be exploited heavily by our watermark

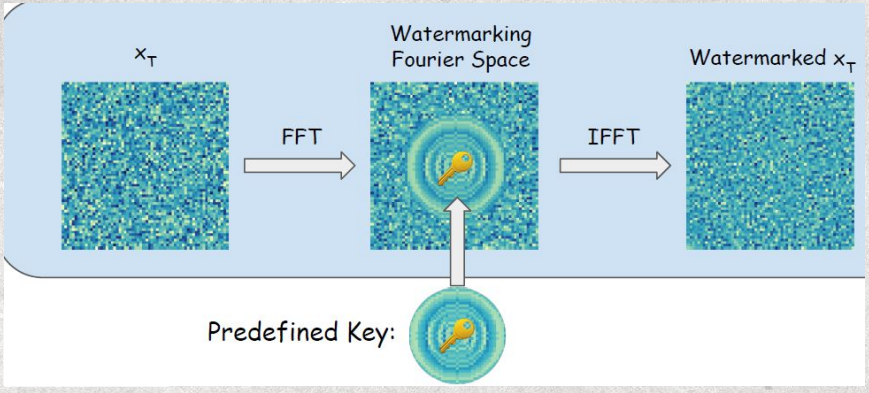
04

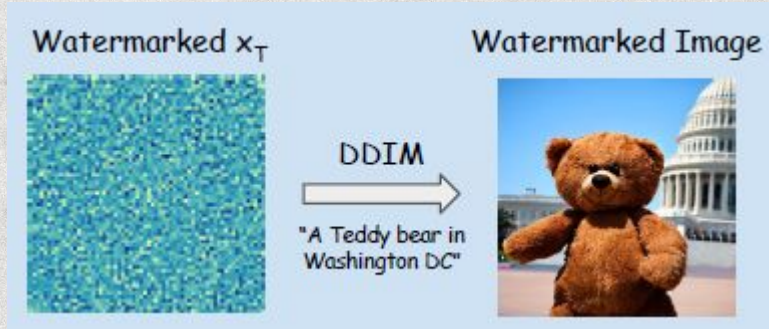
Methodology

Tree-Ring Watermarks

Overview

Diffusion models convert an array of Gaussian noise into a clean image. Tree-Ring Watermarking chooses the **initial noise array so that its Fourier transform contains a carefully constructed pattern** near its center. This pattern is called the "**key**." This **initial noise vector** is then converted into an image using the standard diffusion pipeline with no modifications. To detect the watermark in an image, the diffusion model is inverted using the process of diffusion inversion to retrieve the original noise array used for generation.

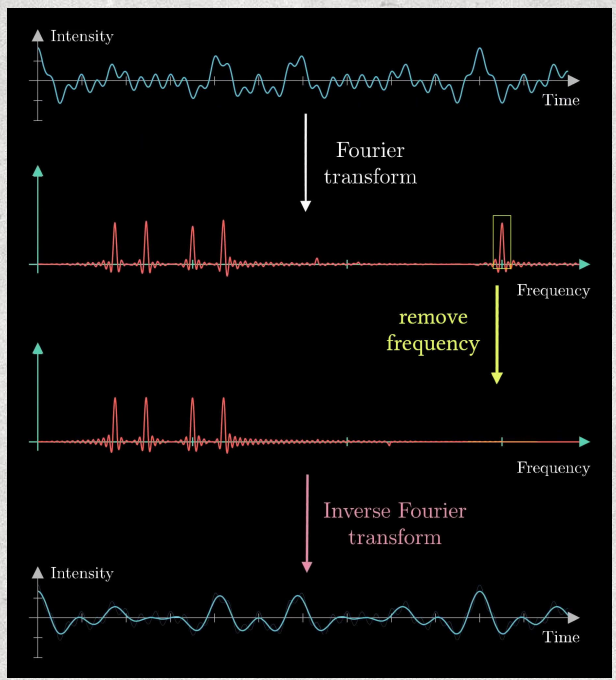




Overview

Rather than imprint the key into the Gaussian array directly, which might cause noticeable patterns in the resulting image, **we imprint the key into the Fourier transform of the starting noise vector**. We choose a binary mask M, and sample the key $k^* \in C^{|M|}$. As such, the initial noise vector $x_T \in R^L$ can be described in Fourier space as

$$\mathcal{F}(x_T)_i \sim \begin{cases} k_i^* & \text{if} \quad i \in M \\ \mathcal{N}(0,1) & \text{otherwise.} \end{cases}$$



Overview

At detection time, given an image x'_0, the model owner can obtain an approximated initial noise vector x'_T through the DDIM inversion process: $x'_T = D^{\dagger}_{\theta}(x'_0)$. The final metric is calculated as the L1 distance between the inverted noise vector and the key in the Fourier space of the watermarked area M, i.e.

$$d_{\text{detection distance}} = \frac{1}{|M|} \sum_{i \in M} |k_i^* - \mathcal{F}(x'_T)_i|,$$

and the watermark is detected if this falls below a tuned threshold τ . The process described above is straightforward. However, its success depends strongly on the construction of the “key” pattern.

Constructing a Tree-Ring Key

We watermark images by placing a “key” pattern into the Fourier space of the original Gaussian noise array. Our patterns can exploit several classical properties of the Fourier transform for periodic signals

- **A rotation in pixel space corresponds to a rotation in Fourier space.**

- A translation in pixel space multiplies all Fourier coefficients by a constant complex number.

- **A dilation/compression in pixel space corresponds to a compression/dilation in Fourier space.**

- **Color jitter in pixel space (adding a constant to all pixels in a channel) corresponds to changing the magnitude of the zero-frequency Fourier mode.**

(a) 75° Rotation

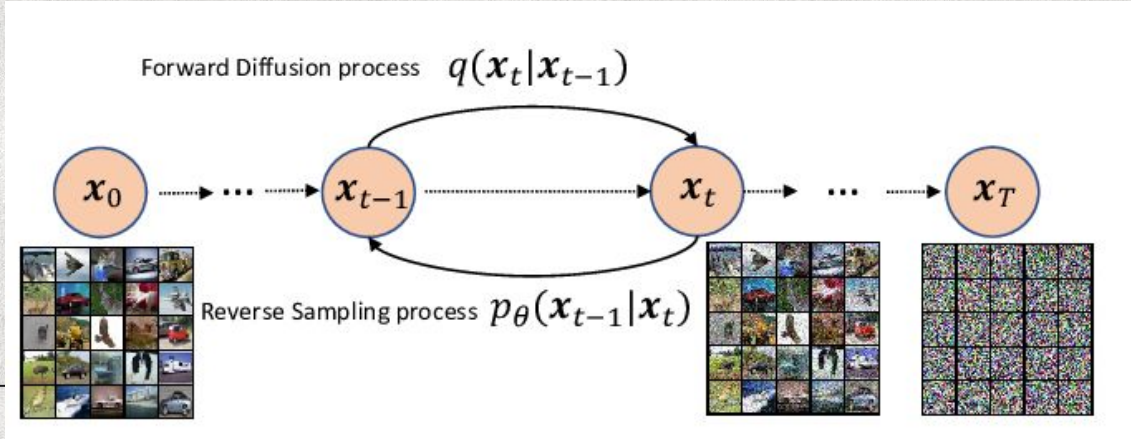
(b) 25% JPEG Compression

(f) Color Jitter with a Brightness Factor of 6

Constructing a Tree-Ring Key

Our watermark departs from classical methods by applying a Fourier watermark to a random noise array before diffusion takes place. Curiously, we will observe below that the **invariant properties above are preserved in x_T even when image manipulations are done in pixel space of x_0.**

In addition to exploiting the invariances above, the **chosen key should also be statistically similar to Gaussian noise**. Note that the Fourier transform of a Gaussian noise array is also distributed as Gaussian noise. **For this reason, choosing a highly non-Gaussian key may cause a distribution shift that impacts the diffusion model.**



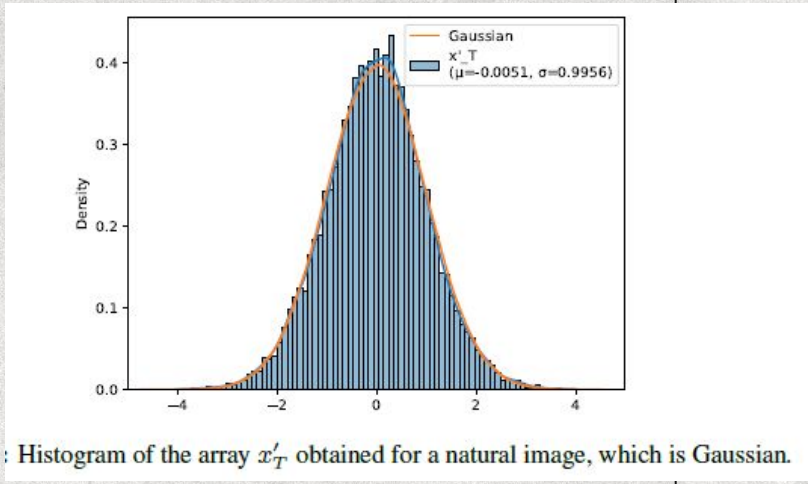
Types of Keys

1. **Tree-Ring$_{Zeros}$:** We choose the mask to be a **circular region** to **preserve invariance to rotations in image space**. The key is chosen to be an **array of zeros**, which creates invariance to shifts, crops, and dilations. This **key is invariant to manipulations**, but at the **cost of departing severely from the Gaussian distribution**. It also prevents multiple keys from being used to distinguish between models.
2. **Tree-Ring$_{Rand}$:** We draw the a fixed key k* from a Gaussian distribution. The k**ey has the same iid Gaussian nature** as the original Fourier modes of the noise array, and so we anticipate this strategy will have the l**east impact on generation quality**. This method also offers the flexibility for the model owner to possess multiple keys. However, it is **not invariant to make image manipulations.**
3. **Tree-Ring$_{Rings}$:** We introduce a **pattern comprised of multiple rings**, and constant value along each ring. This makes the **watermark invariant to rotations**. We choose the constant ring values from a Gaussian distribution. This **provides some invariance to multiple types of image transforms, while also ensuring that the overall distribution is only minimally shifted from an isotropic Gaussian.**

Deriving P-values for watermark detection

A key desideratum for a **reliable watermark detector** is that it provide an **interpretable P-value** that communicates to the user how likely it is that the observed watermark could have occurred in a natural image by random chance. In addition to **making detection results interpretable, P-values an be used to set the threshold of detection**, i.e., the watermark is "detected" when p is below a chosen threshold α. By doing so, one can explicitly control the false positive rate α, making false accusations statistically unlikely.

We construct a statistical test for the presence of the watermark that **produces a rigorous P-value**. The forward diffusion process is designed to map images onto Gaussian noise, and **so we assume a null hypothesis in which the entries in the array x'_T obtained for a natural image are Gaussian**.



Histogram of the array x'_T obtained for a natural image, which is Gaussian.

Hypothesis Testing

For any test image x'_0, we compute the approximate initial vector x'_T and then set $y = \mathcal{F}(x'_T)$. We then define the following null hypothesis

$$\mathrm{H}_0 : y \textit{ is drawn from a Gaussian distribution } \mathcal{N}(0, \sigma^2 I_{\mathbb{C}}). \tag{4}$$

Here, σ^2 is an unknown variance, which we estimate for each image[2] using the formula $\sigma^2 = \frac{1}{M}\sum_{i\in M}|y_i|^2$. To test this hypothesis, we define the score

$$\eta = \frac{1}{\sigma^2}\sum_{i\in M}|k^*_i - y_i|^2. \tag{5}$$

When H_0 is true, the distribution of η is exactly a *noncentral* χ^2 *distribution* [Patnaik, 1949], with $|M|$ degrees of freedom and non-centrality parameter $\lambda = \frac{1}{\sigma^2}\sum_i |k^*_i|^2$.

We declare an image to be watermarked if the value of η is too small to occur by random chance.

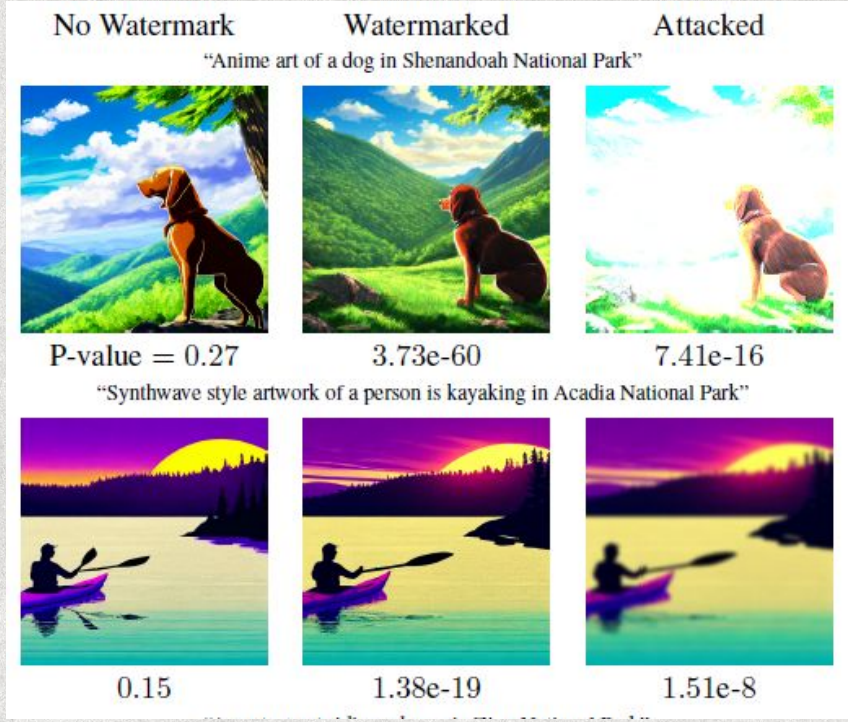
The probability of observing a value as small as η is given by the cumulative distribution function Φ_{χ^2} of the noncentral χ^2 distribution:

$$p = \Pr\left(\chi^2_{|M|,\lambda} \le \eta \middle| H_0\right) = \Phi_{\chi^2}(z). \tag{6}$$

Φ_{χ^2} is a standard statistical function [Glasserman, 2003], available in `scipy` and many other statistics libraries.

Some qualitative examples for P-value

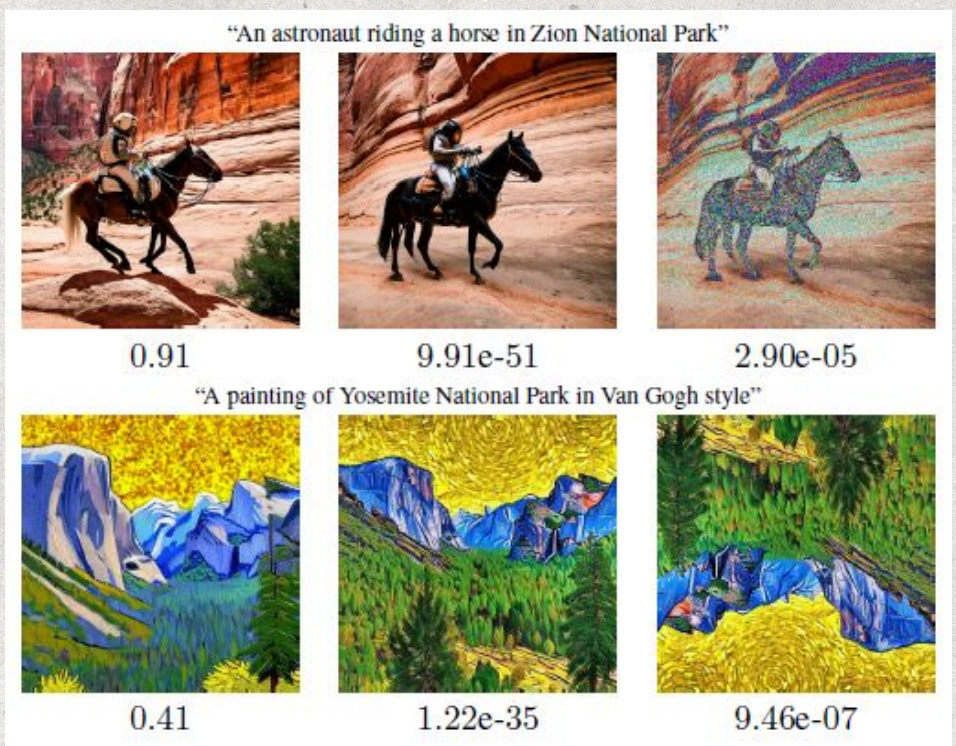
We show qualitative examples of the proposed watermarking scheme and accompanying P-values. For each prompt, we show the generated image with and without the watermark, and also a watermarked image subjected to a transformation. For each image, we report a P-value. As expected, **these values are large for non-watermarked images, and small (enabling rejection of the null hypothesis) when the watermark is present. Transformations reduce the watermark strength as reflected in the increased P-value.**



Some qualitative examples for P-value

The qualitative results show three types of images: non-watermarked, Tree-RingRings watermarked, and attacked watermarked images.

A P-value is provided below each image, which corresponds to the probability of the detected watermark structure occurring by random chance



05

Experiments

Tree-Ring Watermarks

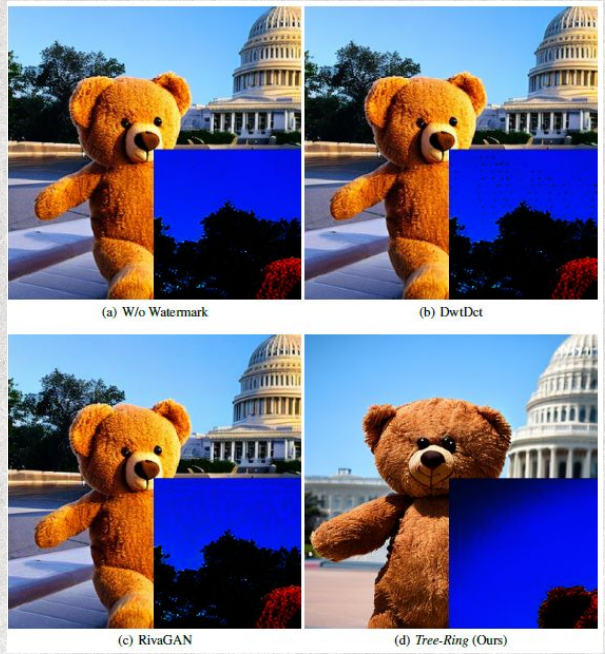
Experimental Setting

- **Diffusion models used for the experiment**
  - **Stable Diffusion-v2**
    - Training dataset used - MS-COCO-2017
  - **256x256 ImageNet Diffusion Model**
    - Training dataset used - ImageNet-1k
- **Number of inference steps = T = 50**
  - Number of steps taken to generate the image from random noise vector
- **Guidance scale = 7.5**
  - Defines the extent to which the model is guided by the text-prompt while generating the image
- **Empty prompt for DDIM inversion**
- **Watermark radius = r = 10**

Comparative Analysis

DwtDctSvd

- Wavelet and Cosine transformation based
- No training required



DwtDct

- Wavelet and Cosine transformation based
- No training required

RivaGAN

- GAN based watermarking
- Pre-training required

Stable Signature

VAE decoder based watermarking
Pre-training required
Exclusively used for stable diffusion

Benchmarking Accuracy

- **Area Under Curve (AUC) of the Receiver Operating Characteristic (ROC) curve**
  - The ROC curve is created by plotting the True Positive Rate (TPR) against the False Positive Rate (FPR) at various threshold settings.
  - The AUC ranges from 0 to 1, with 1 indicating perfect classification (i.e. images with watermark versus images without watermark)
  - AUC provides an aggregate measure of the model's performance across all possible thresholds, capturing its overall discriminative ability.
- **True Positive Rate (TPR) at 1% False Positive Rate (FPR)**
  - It represents the proportion of true positive instances correctly identified by the model when the false positive rate is controlled at 1%.
  - Unlike AUC, TPR@1%FPR is specific to a particular operating point (the thresholds are fixed so that FPR is 1%) and does not provide information about the model's performance across all thresholds.
- **Frechet Inception Distance (FID)**
  - It measures the similarity between distribution of generated images and real images
  - Used specifically for GAN based image generation models
  - A lower FID score indicates low differences between real and generated images
- **CLIP Score**
  - Used to compute similarity between text prompt and generated images
  - Used for Stable Diffusion models

Table 1: Main Results. T@1%F represents TPR@1%FPR. We evaluate watermark accuracy in both benign and adversarial settings. Adversarial here refers to average performance over a battery of image manipulations. An extended version with additional details and standard error estimates can be found in Appendix Table 3.

Model	Method	AUC/T@1%F (Clean)	AUC/T@1%F (Adversarial)	FID ↓	CLIP Score ↑
Stable Diff. FID = 25.29 CLIP Score = 0.363	DwtDct	0.974 / 0.624	0.574 / 0.092	$25.10_{.09}$	$0.362_{.000}$
	DwtDctSvd	1.000 / 1.000	0.702 / 0.262	$25.01_{.09}$	$0.359_{.000}$
	RivaGAN	0.999 / 0.999	0.854 / 0.448	**$24.51_{.17}$**	$0.361_{.000}$
	***Tree-Ring*$_{Zeros}$**	0.999 / 0.999	0.963 / **0.715**	$26.56_{.07}$	$0.356_{.000}$
	***Tree-Ring*$_{Rand}$**	1.000 / 1.000	0.918 / 0.702	$25.47_{.05}$	$0.363_{.001}$
	***Tree-Ring*$_{Rings}$**	1.000 / 1.000	**0.975** / 0.694	$25.93_{.13}$	**$0.364_{.000}$**
ImageNet FID = 17.73	DwtDct	0.899 / 0.244	0.536 / 0.037	$17.77_{.01}$	-
	DwtDctSvd	1.000 / 1.000	0.713 / 0.187	$18.55_{.02}$	-
	RivaGAN	1.000 / 1.000	0.882 / 0.509	$18.70_{.02}$	-
	***Tree-Ring*$_{Zeros}$**	0.999 / 1.000	0.921 / 0.476	$18.78_{.00}$	-
	***Tree-Ring*$_{Rand}$**	0.999 / 1.000	0.940 / 0.585	$18.68_{.09}$	-
	***Tree-Ring*$_{Rings}$**	0.999 / 0.999	**0.966 / 0.603**	**$17.68_{.16}$**	-

Benchmarking Robustness

- **Number of different perturbations included in watermarked images**
  - 75° rotation
  - 25% compression
  - 75% random cropping and scaling
  - Gaussian blur (8x8 filter size)
  - Gaussian noise ($\sigma = 0.1$)
  - Colour jitter (brightness factor 6)
- **Ablation experiments on various hyperparameters**
  - Changing the number of inference steps
  - Changing the watermark radius
  - Changing the guidance scale
  - Changing the attack strength of different perturbations

Table 2: AUC under each Attack for Stable Diffusion, showing the effectiveness of *Tree-Ring*$_{Rings}$ over a number of augmentations. Cr. & Sc. refers to random cropping and rescaling. Additional results for the ImageNet model can be found in Appendix Table 4.

Method	Clean	Rotation	JPEG	Cr. & Sc.	Blurring	Noise	Color Jitter	Avg
DwtDct	0.974	0.596	0.492	0.640	0.503	0.293	0.519	0.574
DwtDctSvd	**1.00**	0.431	0.753	0.511	0.979	0.706	0.517	0.702
RivaGan	0.999	0.173	0.981	0.999	0.974	0.888	0.963	0.854
Stable Sig.	**1.00**	0.658	0.989	**1.00**	0.565	0.731	0.976	0.880
***T-R*$_{Zeros}$**	0.999	**0.994**	0.984	0.999	0.977	0.877	0.907	0.963
***T-R*$_{Rand}$**	**1.00**	0.486	**0.999**	0.971	**0.999**	**0.972**	**0.994**	0.918
***T-R*$_{Rings}$**	**1.00**	0.935	**0.999**	0.961	**0.999**	0.944	0.983	**0.975**

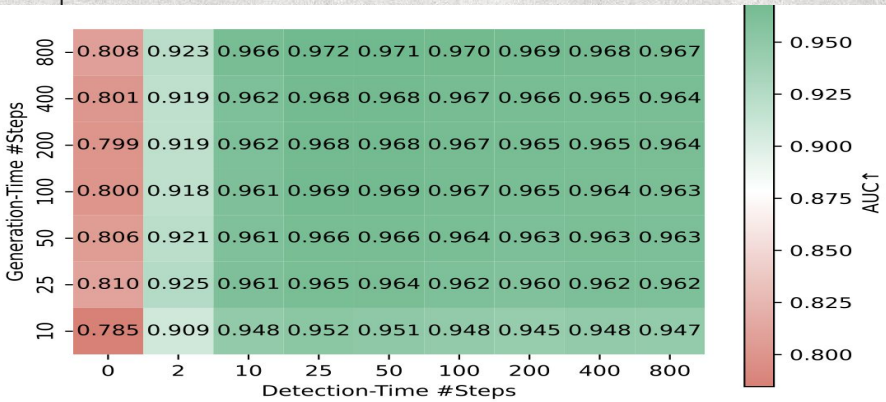
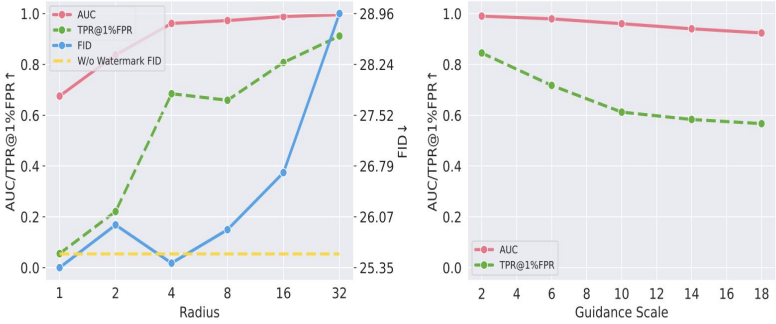
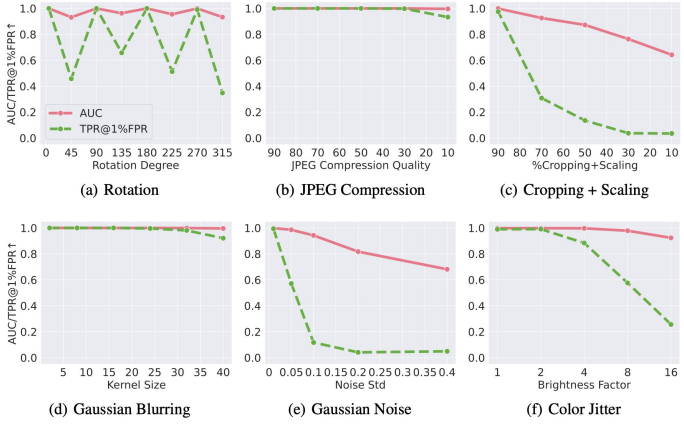


Figure 3: Ablation on Number of Generation Steps versus Detection Steps. Detection succeeds independent of the number of DDIM used to generate data.



(a) Ablation on Watermark Radii (b) Ablation on Guidance Scales

Figure 4: Ablation on Watermark Radii and Guidance Scales.



(a) Rotation (b) JPEG Compression (c) Cropping + Scaling

(d) Gaussian Blurring (e) Gaussian Noise (f) Color Jitter

Figure 5: Ablation on Different Perturbation Strengths.

Remarks on Robustness

- Surprisingly, even with a significant difference between the generation-time and detection-time #steps, the decrease in AUC is minimal when the model owner uses a reasonable number of inference steps for detection without knowledge of the true generation-time steps.
- The model owner (or the user) is free to choose the number of generation steps that achieve the best quality.
- As the watermarking radius increases, the watermark's robustness improves. Nevertheless, there is a trade-off with generation quality.
- Although a higher guidance scale does increase the error for DDIM inversion due to the lack of this ground-truth guidance during detection (i.e. the original text prompt used to generate the image is unavailable at detection), the watermark remains robust and reliable even at a guidance scale of 18 (whereas optimal guidance scale lies between 5 and 15).
- Even with extreme perturbations like Gaussian blurring with kernel size 40, Tree-Ring Watermarking can still be reliably detected.

06

Limitations and Future Work

Tree-Ring Watermarks

Limitations

- **The model requires that DDIM method be used during the inference stage (i.e. during the image generation process)**
- **The correctness of the watermark can only be identified by the model owner, since the detection phase requires that model parameters used during the inference stage be known.**
  - Can be a double edged sword
  - Advantage: adversaries can't verify whether their perturbations have affected the watermark
  - Disadvantage: Third-parties can't verify the correctness of the watermark detected
- **It is unclear how many keys k* can be created.**
  - The method is not scalable if unique keys can't be provided to each user of the API

Future Work

- Increasing the accuracy of the inversion process of DDIM diffusion models will increase the effectiveness of watermarking process

Another thought

Currently we haven't trained the diffusion model on the modified latent space after adding the key in the Fourier transformed space.
Can we train the diffusion model that has watermarked noises as an actual training distribution?

Even after watermarking, there might be a an approach to steer the outputs, which has not been tried yet.

07

Conclusions

Tree-Ring Watermarks

Conclusion

The paper emphasizes the importance of watermarking the output of generative models, distinguishing it from watermarking model weights for intellectual property reasons.

Technique:
Based on patterns imprinted into the Fourier space of the noise vector of the diffusion model.
This approach is invisible on a per-sample basis and highly robust against various image transformations.

Key Contributions:

- **Introduces Tree-Ring Watermarking**
  A novel watermarking technique for diffusion images based on patterns in the Fourier space of the noise vector.

- **Invisibility and Robustness**
  Tree-Ring Watermarking is invisible on a per-sample basis and highly robust against various image transformations.

- **Generative Model Output Protection**
  Focuses on watermarking the output of generative models, distinct from watermarking model weights for intellectual property protection.

08

Acknowledgements

Tree-Ring Watermarks

Thank You!

Group 7
Chaitanya Sethi (2020B3A71961P)
Dhawal Mehta (2020B3A71965P)
Suraj Phalod (2020B3A71959P)