CS726: Advanced Machin Learning
Project Report
Dilwaale Diffusion Le Jaayenge

Harsh Shah 200050049
Sandeepan Naskar 200050126
Shrey Bavishi 200050132
Jash Kabra 200050054

Contents

1 Introduction

1.1 What is Watermarking?

Watermarking is a technique used to embed information, typically in the form of digital data, into a digital signal or a piece of content such as an image, audio, or video file. The purpose of watermarking can vary, but it's often used for copyright protection, authentication, or to convey additional information about the content.

There are two main types of watermarks:
Visible Watermarks: These are typically logos, text, or symbols overlaid on top of the content and are visible to anyone who views it. Visible watermarks are often used to indicate ownership or to discourage unauthorized use of the content.

Invisible Watermarks: Also known as digital watermarks, these are embedded within the content itself and are not readily visible to the viewer.

1.2 Importance

Modern diffusion models can generate photo-realistic depictions of fake scenarios for malicious purposes. Watermarks document the use of image generation systems thus enabling news organizations, social media, etc. to mitigate harms and cooperate with law enforcement to identify the origin of an image as well as protecting copyright of your original work.

2 Problem Statement

We intend to construct invisible watermarks overlayed onto the image with minimal modification imprinted onto an existing image and making it invisible in order to prevent deterioration of image quality.

Tasks to be completed in the solution:

- Need a mechanism for generating high-quality images with watermarks invisible to the human eye.
- Model owner should be able to identify if given image is generated from their own model.
- Watermark is robust against adversarial attacks, consider
  - Cropping/Dilation
  - Rotation/Flipping
  - Blurring/Noise
  - Color Jitter

3 Implementation details and Approach

3.1 Mechanism

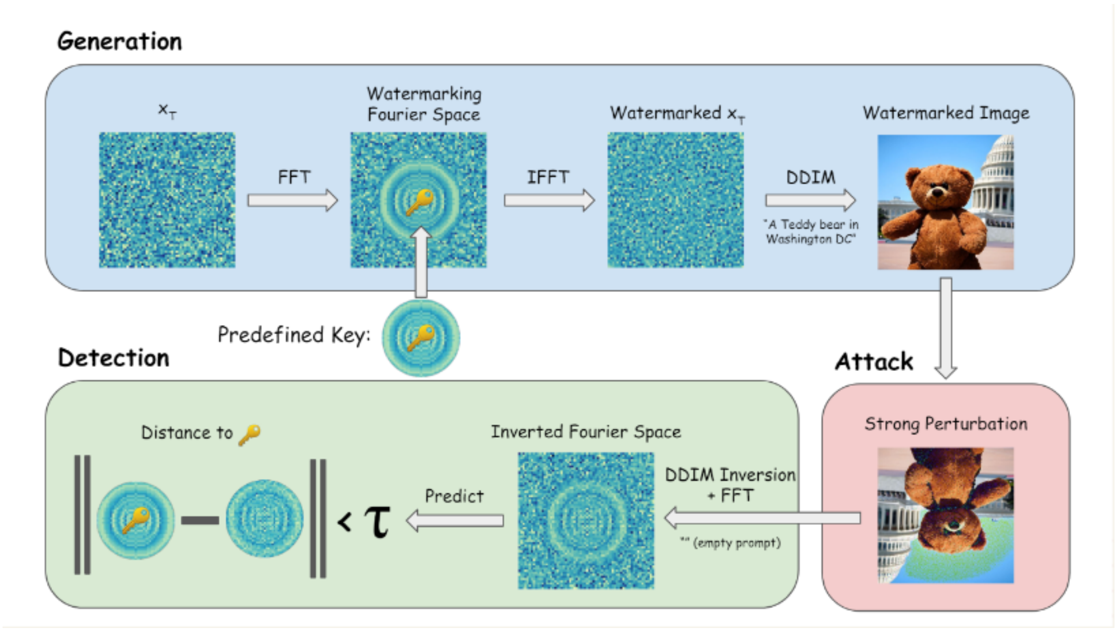


Figure 1: Model Architecture

3.2 Watermark Detection

A reliable watermark detector should provide an interpretable P-value to indicate the likelihood of the observed watermark occurring by random chance in a natural image. This P-value also sets the detection threshold (α), controlling false positive rates.

We construct a statistical test for the watermark's presence, producing a rigorous P-value. Assuming Gaussian noise, we define a null hypothesis (H_0) where x'_T entries for natural images are Gaussian. We compute η, a score related to a noncentral χ^2 distribution. A small η indicates a watermarked image.

$$H_0 : y \text{ is drawn from a Gaussian distribution } N(0, \sigma^2_{IC})$$

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

We declare a watermark if η is too small to occur randomly. The probability of η is given by Φ_{χ^2}, a standard statistical function.

Qualitative examples of the proposed watermarking scheme and P-values are shown in Figure 3. P-values are large for non-watermarked images and small when the watermark is present, even after transformations.

3.3 Models and Parameters

- Modified the code to extract intermediate results
- Many configurations possible (diffusion model, key shape, key pattern)
  - Model : Stable diffusion 2.1
  - Key shape (mask shape) : Circle
  - Key pattern : Rings
- Tradeoff between generation quality and detection of watermark

4 Code Survey

4.1 Existing Approaches

- Traditional watermark casting strategies imprinted a watermark in a suitable frequency decomposition of the image, constructed through
  - Complex wavelet transformations as described in Boland [1996], Cox et al. [1996], O'Ruanaidh and Pun [1997]
  - SVD based decompositions done by Chang et al.[2005] and Al-Haj [2007]
- These traditional approaches are not robust to attacks done via Image manipulations as decribed in Kutter and Petitcolas [1999]
- Hayes and Danezis [2017] and Zhu et al. [2018] propose strategies to learn watermarking end-to-end, where both the watermark encoder and the watermark decoder are learned models
- Yu et al. [2022] proposed a two-stage process where the trained encoder is used to imprint the watermark onto the training data for a generative model
- Existing image watermarking approaches first learn a watermark signal and then learn to either embed it into generated data or the generating model which stands in contrast to watermarking approaches for language models

4.2 Most relevant Code/Papers

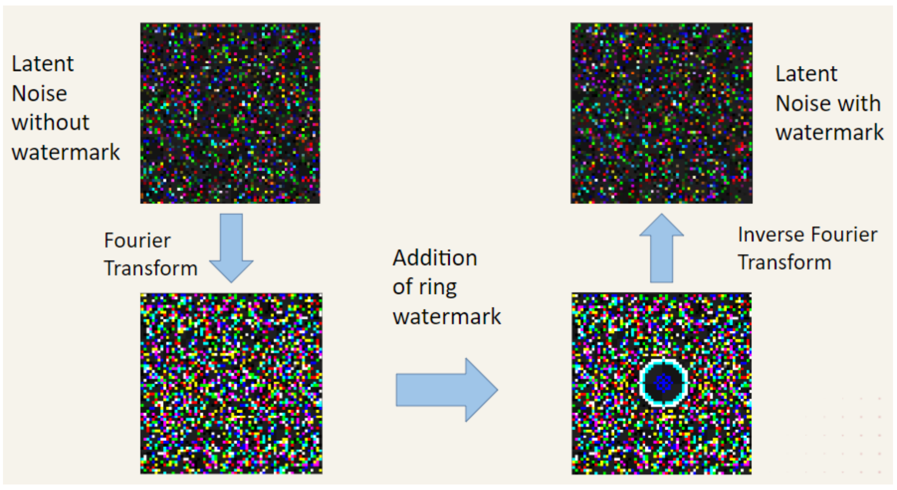
- Tree-Ring Watermarks: Fingerprints for Diffusion Images that are Invisible and Robust (https://arxiv.org/pdf/2305.20030.pdf)
- Official Code: https://github.com/YuxinWenRick/tree-ring-watermark
- This approach alters the output distribution of diffusion models, making it the first watermark that does not rely on minor modification of generated images

5 Dataset

We have used a pre-trained **Stable diffusion 2.1** model to generate our images and then applied watermarking on them. We have sampled the prompts from the **Gustavosta/Stable-Diffusion-Prompts** Dataset with 80,000 prompts.

6 Results

6.1 Inserting Latent Watermarks


Latent Noise without watermark
Fourier Transform
Addition of ring watermark
Latent Noise with watermark
Inverse Fourier Transform

6.2 Invisibilty of Watermarks

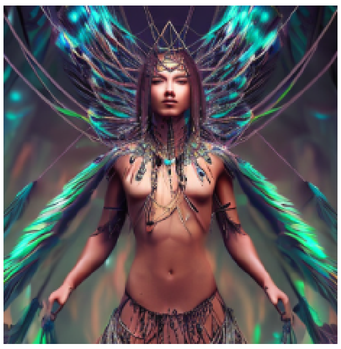


Figure 2: Original

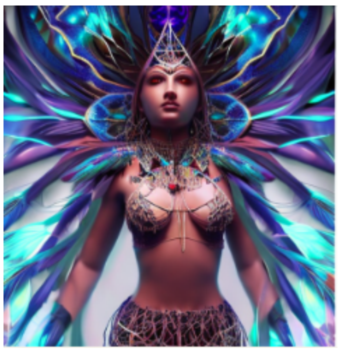


Figure 3: Watermarked

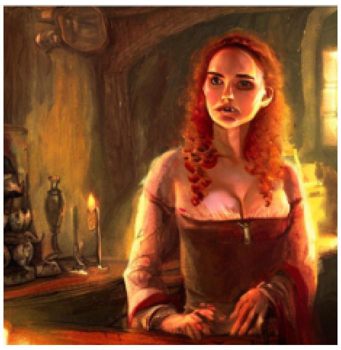


Figure 4: Original

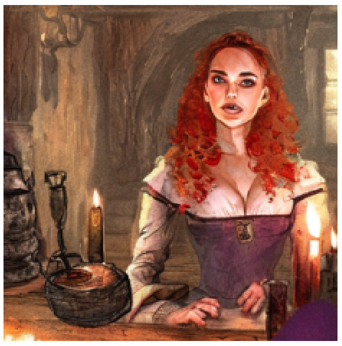


Figure 5: Watermarked

7 Intermediate Improvements

7.1 Challenges

The above examples only had one channel watermarked. If all three channels are watermarked, then the quality is significantly degraded which needs to be improved.

7.2 Other possible improvements

- Analyse other potholes.
- Get the image watermarked images closer to non-watermarked images using guidance by gradients of differential LPIPS loss.
- Try discrete cosine transform.

8 Improvements Post-Mid Presentation

8.1 DCT (Discrete Cosine Transform)

8.1.1 What is DCT?

The Discrete Cosine Transform (DCT) is a technique used in signal processing and data compression to convert a signal into elementary frequency components. It is defined by the formula:

$$X(u) = \sum_{x=0}^{N-1} x(n) \cdot \cos\left[\frac{\pi}{N}\left(n + \frac{1}{2}\right)u\right], \quad u = 0, 1, \ldots, N-1$$

where:

- $X(u)$ is the frequency component at frequency u,
- $x(n)$ is the input signal,
- N is the total number of samples in the input signal.

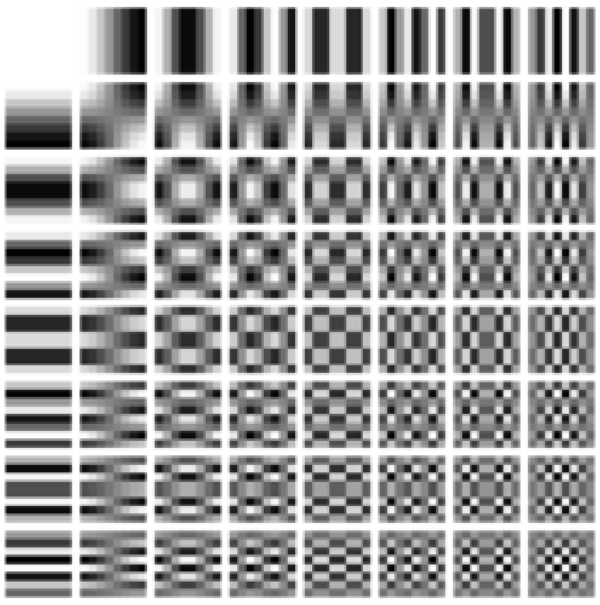
8.1.2 DCT for Images

In the context of images, the DCT is applied to blocks of pixels to transform them into the frequency domain. This is done by dividing the image into $N \times N$ blocks and applying the 2D DCT formula:

$$X(u,v) = \sum_{x=0}^{N-1}\sum_{y=0}^{N-1} x(n,m) \cdot \cos\left[\frac{\pi}{N}\left(n + \frac{1}{2}\right)u\right] \cdot \cos\left[\frac{\pi}{N}\left(m + \frac{1}{2}\right)v\right]$$

where:

- $X(u,v)$ is the frequency component at frequency (u,v),
- $x(n,m)$ is the pixel intensity at position (n,m) in the image.



8.1.3 Our Use for Increased Efficiency

We have incorporated the Discrete Cosine Transform (DCT) into our watermarking diffusion models to enhance efficiency. By utilizing DCT instead of the Fast Fourier Transform (FFT), we can significantly reduce the storage requirements for embedding the watermark.

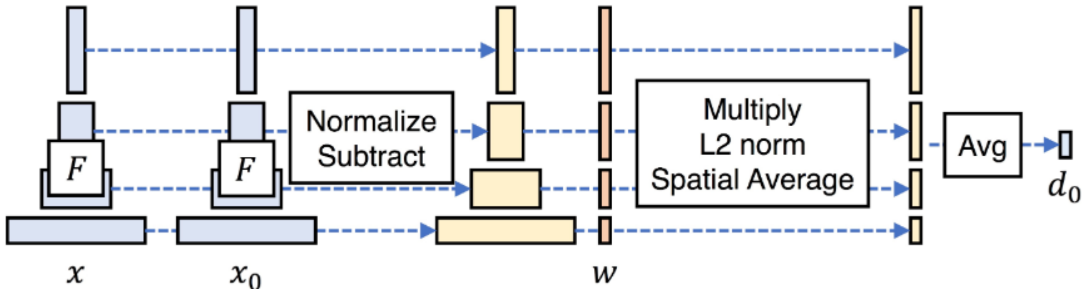
Unlike FFT, which uses complex numbers and requires storage for both real and imaginary parts, DCT results in purely real numbers. This means that for each pixel in the image, we only need to store one DCT coefficient instead of two complex numbers, thus halving the storage space needed for embedding the watermark. This reduction in storage requirements not only improves space efficiency but also simplifies the embedding process.

Additionally, DCT allows us to better distribute the watermark across the image while minimizing the perceptual impact on the visual quality. By modifying the DCT coefficients, we can embed the watermark information in a way that is imperceptible to the human eye, ensuring that the quality of the watermarked image remains high.

Overall, the use of DCT in place of FFT has significantly improved the space efficiency of our watermarking diffusion models, making them more practical for real-world applications where storage resources may be limited.

8.2 Learned Perceptual Image Patch Similarity (LPIPS)

LPIPS is a metric used to measure the perceptual similarity between two images. Unlike traditional metrics like Mean Squared Error (MSE) or Structural Similarity Index (SSIM), LPIPS is based on deep learning and aims to capture perceptual differences that are more aligned with human visual perception.



8.2.1 Overview

1. **Deep Learning Approach**: LPIPS is based on deep neural networks, particularly Convolutional Neural Networks (CNNs).
2. **Perceptual Features**: LPIPS captures perceptually meaningful features from images and computes similarity between these features.
3. **Patch-based Comparison**: LPIPS operates by comparing local image patches rather than the entire image.
4. **Training on Human Perception**: LPIPS networks are trained using human perceptual judgments.
5. **Robustness**: LPIPS is robust to common image distortions such as brightness changes, contrast, and noise.
6. **Applications**: LPIPS is used in image quality assessment, image super-resolution, image generation, and style transfer.

8.2.2 LPIPS Loss

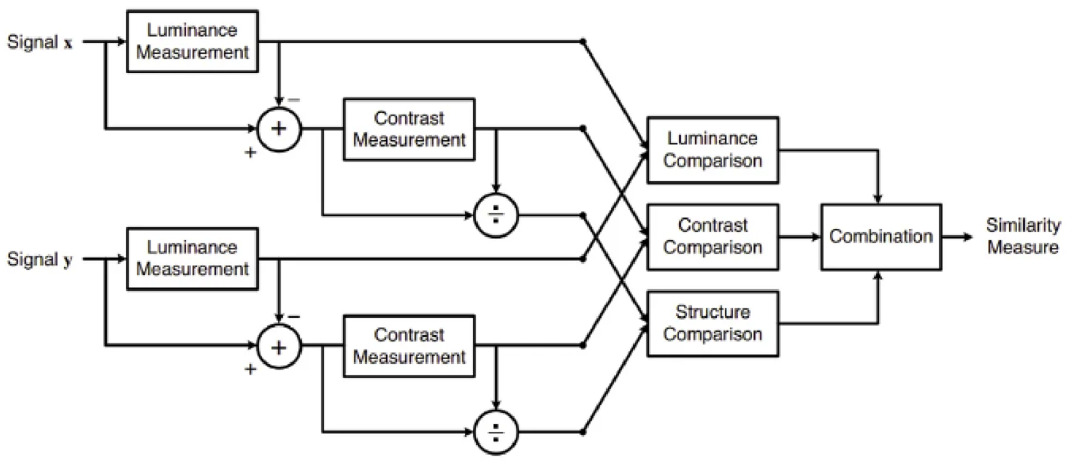
The LPIPS loss is a metric used in image processing tasks to measure the perceptual similarity between two images. Unlike traditional loss functions such as Mean Squared Error (MSE), which focus on pixel-wise differences, LPIPS loss is based on deep learning and captures perceptual differences that are more aligned with human visual perception.

The LPIPS loss is computed by passing the images through a pre-trained deep neural network (In our case using **AlexNet**), typically a Convolutional Neural Network (CNN), and comparing the feature representations of the images. The feature representations are extracted from intermediate layers of the network and compared using a perceptual similarity measure.

The LPIPS loss is particularly useful in tasks such as image generation, image super-resolution, and style transfer, where preserving perceptual quality is important. By optimizing the LPIPS loss during training, models can generate images that are visually similar to the target images, even if there are differences in pixel values.

8.3 Structural Similarity Index (SSIM)

The Structural Similarity Index (SSIM) is a metric used to measure the similarity between two images. It was designed to account for perceived changes in structural information, such as luminance, contrast, and structure, which are important for human visual perception.



SSIM compares three aspects of the images: luminance, contrast, and structure. The SSIM index is calculated using the mean luminance (μ_x, μ_y), the variance (σ_x^2, σ_y^2), and the covariance (σ_{xy}) of the image patches.

The SSIM index between two images x and y is defined as:

$$\text{SSIM}(x, y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)}$$

where:

$$\mu_x, \mu_y : \text{mean of images } x \text{ and } y,$$
$$\sigma_x^2, \sigma_y^2 : \text{variance of images } x \text{ and } y,$$
$$\sigma_{xy} : \text{covariance between images } x \text{ and } y,$$
$$C_1, C_2 : \text{constants to stabilize the division.}$$

SSIM returns a value between -1 and 1, where 1 indicates perfect similarity and -1 indicates perfect dissimilarity.

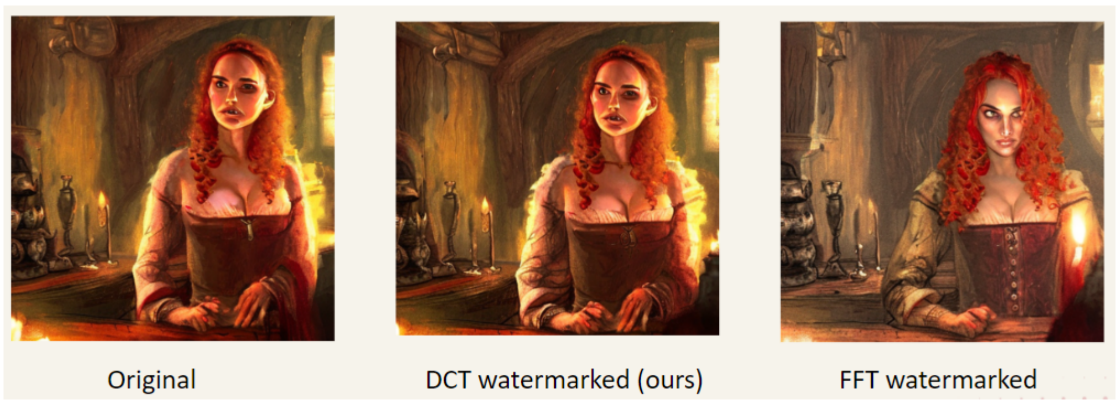
SSIM is widely used in image processing and computer vision tasks, particularly for evaluating the quality of image compression algorithms, image restoration, and super-resolution techniques.

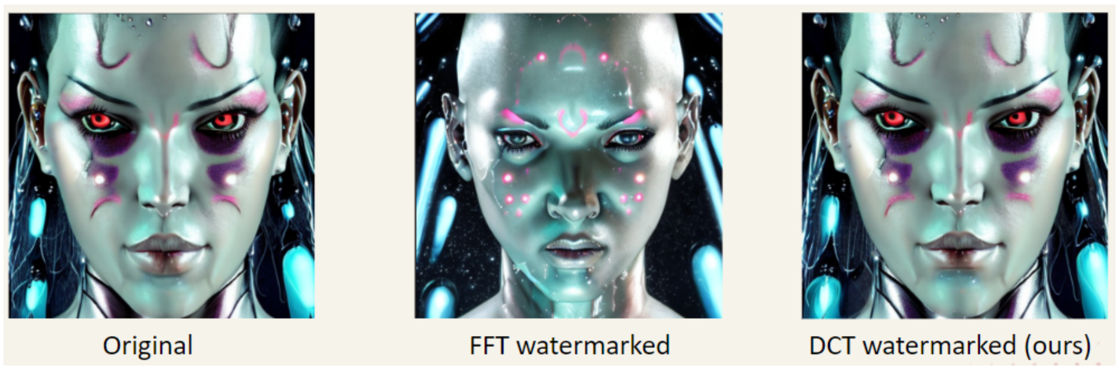
9 Final Results

9.1 Experiment details

- For all experiments, used 50 fixed prompts from the dataset
- Multi-ring watermark for both FFT and DCT based methods
- Model : Stable-Diffusion 2.1

9.2 Result Images





9.3 Metrics

Transform vs Metric	Accuracy	[Mean] Clip Score	[Var] Clip Score
FFT	1.0	0.36857	0.03899
DCT (our)	1.0	0.36695	0.02834

Method vs Metric	LPIPS loss	SSIM loss
FFT	0.25	0.43
DCT (ours)	0.09	0.22

9.4 Inferences

Why this works?

- Couldn't pin down the exact reason, but most plausible reason:
  - DCT method changes the higher frequency terms, possibly leading to a sample close to Gaussian distribution after inverse DCT.
  - FTF method changes the low frequency terms, greatly affecting the distribution in image space.
- **Caveats:**
  - Rotation equivariance is lost (still good recovery for small rotations $\sim$10 deg).

10 References

- Steganography : `https://www.sciencedirect.com/science/article/pii/S0165168422004479`
- Fourier transform : `https://homepages.inf.ed.ac.uk/rbf/HIPR2/fourier.htm`
- Discrete Cosine Transform : `https://in.mathworks.com/help/images/discrete-cosine-transform.html`
- Denoising schedulers : `https://blog.segmind.com/what-are-schedulers-in-stable-diffusion/`