國立臺灣大學電機資訊學院資訊工程學研究所

碩士論文

Department of Computer Science and Information Engineering

College of Electrical Engineering and Computer Science

National Taiwan University

Master Thesis

加強潛在擴散模型中根源浮水印的強韌性

Enhancing the Robustness of Rooting Watermarks in Latent Diffusion Models

劉旭庭

Hsu-Ting Liu

指導教授: 吳家麟 博士

Advisor: Ja-Ling Wu, Ph.D.

中華民國 112 年 6 月

June, 2023

doi:10.6342/NTU202302023

國立臺灣大學碩士學位論文
口試委員會審定書
MASTER'S THESIS ACCEPTANCE CERTIFICATE
NATIONAL TAIWAN UNIVERSITY

加強潛在擴散模型中根源浮水印的強韌性

Enhancing the Robustness of Rooting Watermarks in Latent Diffusion Models

本論文係劉旭庭君（學號 R10922129）在國立臺灣大學資訊工程學系完成之碩士學位論文，於民國 112 年 6 月 30 日承下列考試委員審查通過及口試及格，特此證明。

The undersigned, appointed by the Department of Computer Science and Information Engineering on 30 June 2023 have examined a Master's thesis entitled above presented by HSU-TING LIU (student ID: R10922129) candidate and hereby certify that it is worthy of acceptance.

口試委員 Oral examination committee:

吳家麟		
（指導教授 Advisor）		
許超雲	陳文進	

系主任/所長 Director: 洪士灝

Acknowledgements

首先，感謝吳家麟老師的栽培。起初接觸研究時並沒太多興趣，覺得就是畢業的基本條件而已。在修過老師的課程後，以及每一次 meeting，感受到老師滿腔熱忱，這樣的能量如春風化雨般熏陶著我，不知不覺，埋藏在我心中的種子已經發芽了。在研究領域之外，老師對於各類議題往往能提出我不曾想過的見解，其中蘊含的邏輯思維也讓我獲益良多。談吐間更不乏幽默感，跟老師對話總是愉快，雖然老師常自嘲自己是 screwdriver. But to me, in the ocean of research, you are the true diver. 十分慶幸能接受老師指導！未來我會努力讓小苗成長茁壯。

接著，感謝陳駿丞學長對我的幫助，撥出時間跟我討論，分享許多學術活動和資源並且邀請我參與，進一步帶領我深入研究文化之中。也謝謝博班的學長姐們，一路走來充滿關懷，在學業上給出建議。感謝實驗室的同儕相互照應，有你們的這兩年氣氛很溫暖。特別是林家毅，在各方面與我暢談、腦力激盪，讓我得到不少靈感，感謝你的慷慨無私。

最後，感謝父母在背後默默付出，給予我愛與包容，讓我心裡特別踏實，無後顧之憂盡情探索。還有我的小貓 Vincent，Love u so much!

劉旭庭

國立臺灣大學 06/2023

doi:10.6342/NTU202302023

摘要

近期許多研究人員深入探索複雜的潛在擴散模型領域，這些模型能依據多模態條件生成逼真的圖片，這種特性賦予了它們強大的潛力，適用於各種應用。同時，這類模型也牽涉到迫在眉睫的倫理困境，主要受限於如何負責任地部署這項技術。惡意使用者製造的圖像有可能造成重大的社會問題，需要嚴格的控制措施以減輕可能的危害。根源浮水印技術是解決此問題可行的方法，這項技術允許我們追踪那些惡意創建的圖像來源，從而利用法律向為非作歹為非作歹者咎責。本篇論文基於現有的方法加以改善，著重於提升根源浮水印對於圖像失真的抵抗能力。通過對現有方法模組的正規化和在訓練期間加入圖像噪聲，本篇論文加強了根源浮水印的強韌性同時保持生成圖像的視覺品質。研究結果顯示，在圖片縮小為原圖的 10%後，浮水印解碼之為元準確率相較現有方法高出 24%，此結果展現本篇論文的有效性，也在潛在擴散模型之根源浮水印領域中取得重要的進展。

關鍵字：強韌浮水印、潛在擴散模型、圖像取證、生成模型、變分自動編碼器

doi:10.6342/NTU202302023

Abstract

Recently, many researchers dived into the intricate realm of latent diffusion models, renowned for their competence in generating images reflecting a multimodal range of conditions. This property endows them with enormous potential, rendering them fit for a host of applications across various disciplines. Simultaneously, the power of such models introduces pressing ethical dilemmas, primarily revolving around the responsible deployment of this technology. The indiscriminate use of such models has the potential to cause significant harm, emphasizing the need for rigorous control measures to mitigate potential misuse. Root watermarking is introduced as a countermeasure to address these ethical concerns. This technology allows for the tracing of images that have been maliciously created back to the originators. By establishing this link, we can assign responsibility for any misuse, facilitating the enforcement of legal sanctions and deterring misuse. In this work, we focus on enhancing the resilience of common distortions in the generated images. We have expanded upon an existing method, refining it through the normalization

doi:10.6342/NTU202302023

of its module and the strategic injection of noise during the training. This approach has allowed us to significantly enhance the model's robustness, all the while maintaining the perceptual quality of the generated images. Our quantitative analysis attests to the efficacy of our approach, as it has resulted in a substantial improvement in bitwise accuracy by up to 24%, even when the images undergo a 10% resizing. This outcome not only showcases the effectiveness of our method but also represents a significant stride forward in the ongoing journey to watermarking latent diffusion models.

Keywords: Robust watermarking, Latent diffusion model, Image forensics, Generative model, Variational autoencoders

doi:10.6342/NTU202302023

Contents

doi:10.6342/NTU202302023

doi:10.6342/NTU202302023

List of Figures

doi:10.6342/NTU202302023

doi:10.6342/NTU202302023

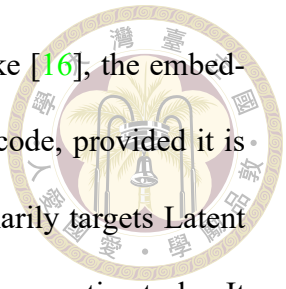
List of Tables

doi:10.6342/NTU202302023

Chapter 1 Introduction

The digital creation realm has witnessed a seismic shift in recent years, propelled by groundbreaking advancements in generative modeling and natural language processing. Cutting-edge AI technologies have unlocked new potentialities in generating and manipulating images, thus marking the dawn of a new epoch in digital arts. Pioneering models such as DALL·E 2[14] and Stable Diffusion[16] have demonstrated their capabilities in transforming text into images, often producing results that mimic genuine artworks with uncanny accuracy. This progress has spurred the development of numerous image editing tools like ControlNet[26] and Instruct-Pix2Pix[1], laying the foundation for emerging artists, designers, and average users to utilize AI for their creative exploits. This meteoric rise of generative AI has not been without its share of ethical dilemmas that need immediate attention. AI-generated images have reached a level of sophistication such that they closely resemble real pictures, making differentiating between them increasingly complex. This complexity carries profound implications—it probes the core tenets of image verification and traceability, which brings concerns over possible misuse of AI, such as infringing copyrights, creating deep fakes, and aiding impersonation. Traditional methods employed to mitigate these challenges have been image forensics, passive strategies to discern generated or manipulated images, and watermarking techniques that imprint a hidden message into the image after it is generated. However, these techniques bear

doi:10.6342/NTU202302023



significant flaws. For instance, in the case of open-source models like [16], the embedded watermark can be conveniently removed by altering the source code, provided it is publicly accessible. Stable Signature[3] is a new approach that primarily targets Latent Diffusion Models (LDM), given their versatility in executing various generative tasks. It weaves the watermarking process into the core of image generation by fine-tuning a minor part of the model, which is architecturally non-invasive and does not disrupt the diffusion process, ensuring its compatibility with most LDM-based generative methodologies. Our main contribution to this work is to enhance the robustness of the [3] methodology. We prioritize the robustness performance by introducing a noise layer during the fine-tuning stage of the VAE decoder, enabling it to learn how to embed invisible watermarks robustly while being exposed to various types of distortions. However, we encountered challenges when introducing the noise layer, as it disrupted the stability of the fine-tuning process. To address this issue, we introduced a normalization step to ensure the stability of the partial module in the model. Additionally, we discovered an issue of accuracy imbalance among different watermark bits, which may be attributed to a specific module in the model. This observation highlights the importance of further investigation of this imbalance to ensure the overall effectiveness of the watermarking technique. The subsequent chapters of this thesis are structured as follows: Chapter 2 provides a comprehensive review of related works in the field. Chapter 3 delves into the intricacies of our implementation, presenting detailed information about the methodology employed. Chapter 4 presents the experimental results, showcasing the outcomes of our research. Finally, Chapter 5 concludes this essay by summarizing the essential findings and offering insights into potential future directions for further exploration.

doi:10.6342/NTU202302023

Chapter 2 Related Works

In this chapter, we present an overview of the DNN-based watermark method in Section 2.1. Subsequently, in Section 2.2, we review the concept of rooting watermarks for model-generated image attribution. Section 2.3 introduces the state-of-the-art image generation model, known as the diffusion model. Finally, in Section 2.4, we discuss the recent advancements and developments in rooting watermark techniques related to diffusion models.

2.1 DNN-based watermarking

HiDDeN[28] presents the first DNN-based image watermarking method. It contains an embedder, extractor, and noise layer. Given a cover image, the embedder hides the invisible watermark inside it called watermarked image. Because the watermark is invisible, the watermarked image is perceptually non-differentiable from the original cover image. The watermark extractor can extract the hidden message from the watermarked image, achieving IPR protection and other usages. This method allows the user to train an end-to-end model without adding noises in a specific hand-craft embedding space. This approach can be replaced by adding anticipated distortion into the noise layer during training. DA[11] improves the robustness making the noise layer trainable. With the adversar-

doi:10.6342/NTU202302023

ial training of the noise layer, the model can enhance the robustness even more on unseen noise. SSL watermarking[4] uses a pre-trained self-supervise feature extractor to convert images from the image to the latent spaces. Furthermore, aligning the latent representation with an anchor in a latent space as a watermark is also an adversarial attack base on a self-supervised feature extractor for watermark embedding.

2.2 Rooting watermark

Deepfakes, which refer to artificially manipulated images and videos using AI techniques, have gained significant attention in recent years. Initially, these manipulations predominantly relied on GAN methods[2, 5, 8]. However, as the issues related to Deepfakes became more prominent, a solution [24] emerged that enables tracing the source of Deepfake images by proactively embedding watermarks into the images. This approach allows attributing GAN models if the embedded watermark can be detected. [24] leverages the DNN-based watermark method to embed user-specific watermarks into the training data for the deployed GAN model. Consequently, the watermark becomes transferred to the model parameters, enabling the detection of the user-specific watermark in every image generated by this GAN model.

Despite its effectiveness, this method encounters scalability challenges when dealing with many watermarks. This inefficiency is due to the need to pre-process the training data and re-train a generator for each watermark. In contrast, [25] offers fundamental advantages by directly and efficiently watermarking generative models. By training a single generic watermark model, the model distributor can instantiate numerous generators with different user-specific watermarks through watermark encoding and filter modulation

doi:10.6342/NTU202302023

within the generative model. This approach provides a more scalable and flexible solution for watermarking generative models.

2.3 Diffusion Models

The diffusion model has recently emerged as the leading image generation model, building upon the early concept proposed in [19]. Subsequent advancements in this architecture have been made through essential training and sampling methods such as Denoising Diffusion Probabilistic Model (DDPM)[7], Denoising Diffusion Implicit Model (DDIM)[20], and score-based diffusion[21]. These techniques have greatly improved the model's ability to generate high-quality images. Furthermore, additional conditions, such as text, have been integrated with models like [14]. Leveraging powerful CLIP[13] embeddings, [14] can produce diverse and photorealistic outputs based on given inputs, representing a desirable property of image generation. Another notable text-to-image engine is Imagen[18], diffusing image pixels using a pyramid structure. Palette[17] is a versatile diffusion model capable of multitasking, including super-resolution, image inpainting, and JPEG reconstruction. On the other hand, [16] is another multitasking model that supports multi-modality conditions, such as text, semantic maps, and depth maps, providing enhanced capabilities. In addition to multi-modality, [16] performs the diffusion process in the latent space by converting the image into a latent representation using a well-pretrained VAE encoder. Following the diffusion process, the latent representation is transformed into a photo using the VAE decoder. This approach significantly reduces training and inference time while benefiting from the VAE's ability to transform input images into a more learnable distribution.

doi:10.6342/NTU202302023

2.4 Diffusion Models-related rooting watermark

[27] introduces the rooting watermark technique specifically designed for the diffusion model. The model architecture closely resembles [24], with the key distinction being the shift from a GAN-based model to a diffusion model. Still, it has to watermark all training datasets pre-user which is non-scalable because it is challenging to integrate the watermark extractor into the end-to-end training process when dealing with generated images. The inferencing phase of the diffusion model involves sampling an image through multiple diffusion steps, resulting in difficulties and instability in backpropagation during training. Stable Signature[3] presents an efficient approach for rooting models, primarily focusing on the latent diffusion model. The methodology involves deploying a diffusion U-Net and a VAE decoder to generate images. The notable advantage of Stable Signature lies in its requirement for minimal fine-tuning of the VAE decoder for each user, using a small amount of training data (≤ 500) and requiring only a few minutes of training time. Additionally, this work incorporates a whitening layer to decorrelate the extracted watermark from the vanilla image (i.e., non-watermarked image), which will be discussed further in Section 3.5.

doi:10.6342/NTU202302023

Chapter 3 Proposed Method

3.1 Overview

The primary focus of our research is to strengthen the robustness of the Stable Signature methodology by introducing a supplementary noise layer during the fine-tuning stage. However, we encountered challenges during the noise and whitening layers implementation as they did not synergize effectively. Specifically, we observed a substantial increase in output variance following the addition of the whitening layer. We normalized the whitening layer to address this issue and ensure a more stable fine-tuning stage. We adopt a two-step approach as Stable Signature. The overview of our method is shown in Figure 3.1. First, we engage in pre-training a watermark embedder and extractor to effectively extract the watermark message from a watermarked image. In the second stage, we fine-tune the VAE[9, 22] decoder of LDM with a fixed user watermark. This fine-tuning process ensures that all latent representations transformed to image space by the decoder serve to conceal the watermark information, which the watermark extractor can subsequently extract. To achieve the attribution of rooting watermark.

doi:10.6342/NTU202302023

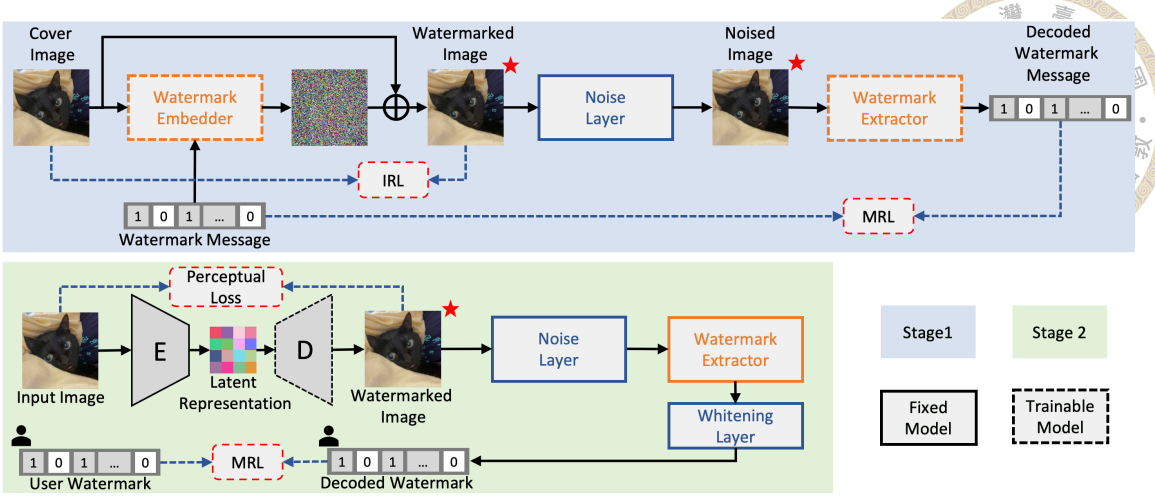


Figure 3.1: Block Diagram and Information Flow of our proposed two stages of training. Stage 1 involves pre-training a watermark extractor, while Stage 2 focuses on fine-tuning a VAE decoder to embed a user-specific watermark.(IRL refers to Image Reconstruction Loss, which measures the fidelity of reconstructed images. MRL stands for Message Reconstruction Loss, which quantifies the accuracy of reconstructed watermark messages).

3.2 Pre-training the watermark embedder/extractor

We adopt[28] as our watermark backbone, a well-established and widely used network in deep watermarking literature. [28] operates by jointly optimizing the parameters of the watermark embedder and extractor networks to robustly embed k-bit messages into images, accounting for various transformations in the noise layer during the training process. After pre-training, only the watermark extractor will be used during the fine-tuning phase. The embedder will no longer need and will be discarded.

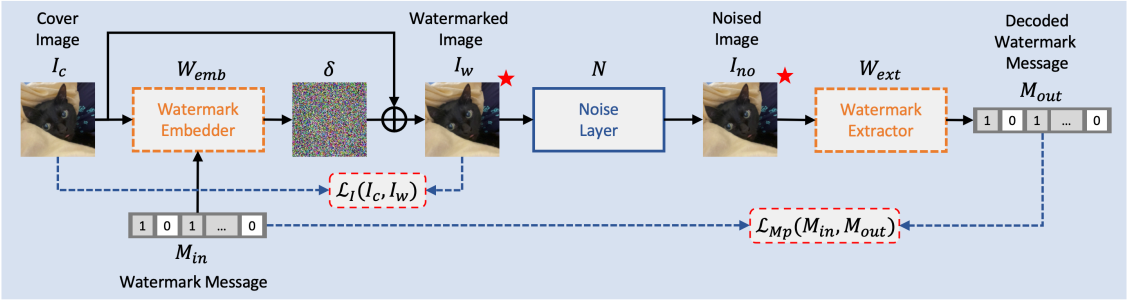


Figure 3.2: The detailed information flow in the Pre-training phase.

For ease of explanation, we redraw the Pre-training phase of Figure 3.1 in Figure 3.2,

doi:10.6342/NTU202302023

as above. The watermark embedder W_{emb} receives a cover image $I_c \in \mathbb{R}^{C\times H\times W}$ and a k-bit watermark message $M_{in} \in \{0,1\}^k$, outputs a residual watermark image $\delta \in \mathbb{R}^{C\times H\times W}$, and produces a watermarked image $I_w = I_c + \alpha\delta$, where α is a scaling factor. After conducting the transformation in the noise layer, we get I_{no} from I_w. The watermark extractor W_{ext} extracts a soft message $M_{out} \in [0,1]^k$ from I_{no}.

To maintain the imperceptibility of watermarked images, they should be visually close to the original cover images. We use Image Reconstruction Loss(IRL) , the MSE loss between I_c and I_w: $\mathcal{L}_I(I_c, I_w) = \|I_c - I_w\|_2^2/(CHW)$, to measure the objective quality difference between the original and the watermarked images. Moreover, the decoded watermark message should be the same as the input one. We impose a Message Reconstruction Loss(MRL) also based on the MSE loss obtained during the pre-training between the M_{in} and M_{out}: $\mathcal{L}_{Mp}(M_{in}, M_{out}) = \|M_{in} - M_{out}\|_2^2/k$. We optimize our watermark model according to the loss defined over the distributions of watermark messages and input images, that is

$$\mathbb{E}_{I_c,M_{in}}\left[\lambda_I\mathcal{L}_I(I_c, I_w) + \mathcal{L}_{Mp}(M_{in}, M_{out})\right]$$

The way we pass the watermark message to the watermark embedder is to replicate the watermark message across spatial dimensions, creating a watermark message tensor. Then we concatenate this watermark message tensor with the feature map obtained from the output of $conv4$ (cf. Figure 3.3). By combining these elements, we enable the integration of the watermark message within the feature map, thereby imbuing it with the watermark information. We draw its network architecture and information flow in Figure 3.3 to better understand how our watermark embedder works.

doi:10.6342/NTU202302023

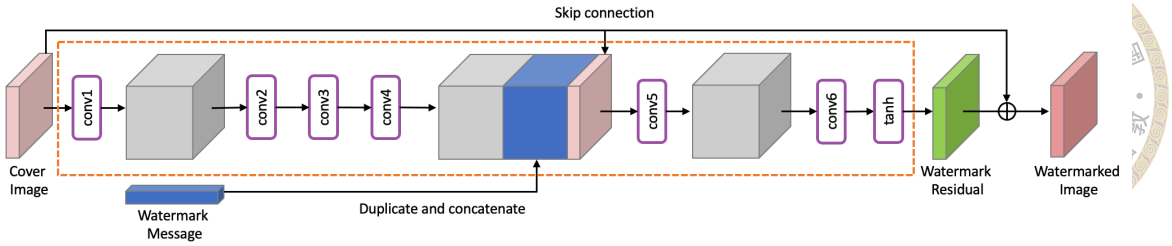


Figure 3.3: Watermark embedder architecture. The Grey tensor represents the feature map of convolution. The Blue tensor represents the watermark message-related tensor.

In Figure 3.3, the watermark extractor gets the watermark message tensor after the convolutional layer, which is then fed into the average pooling layer. This approach proves advantageous as it removes concerns about different image sizes since the spatial dimensions are compressed into one dimension through the average pooling process, so we can train at size 256 and input 512 size images during testing. We expect randomness in the output watermark bits extracted from the vanilla image (non-watermarked image). That is why after pre-training, we will append a linear layer called the whitening layer after the watermark extractor (cf. the green-colored sub-block of Figure 3.1) to remove the output bit correlation of the vanilla image. After pre-training, the watermark embedder will be discarded, and only the watermark extractor will participate in the subsequent fine-tuning phase. We draw our watermark extractor's network architecture and information flow for completeness and better comprehension in Figure 3.4.

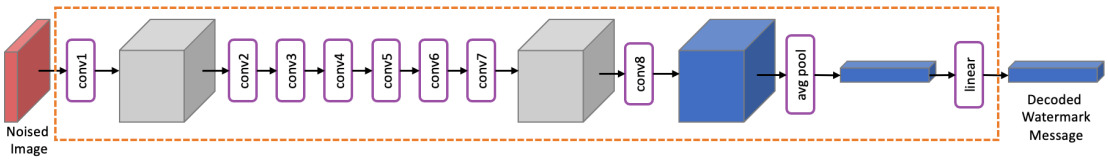


Figure 3.4: Watermark extractor architecture. The Grey tensor represents the feature map of convolution. The Blue tensor represents the watermark message-related tensor.

doi:10.6342/NTU202302023

3.3 Fine-tuning the VAE decoder

A critical aspect of the LDM (Latent Diffusion Models) framework involves utilizing a well-pre-trained VAE (Variational Autoencoder) encoder-decoder pair. This pre-trained VAE serves as a fixed component. The diffusion process exclusively occurs in the latent space encoded by the VAE encoder, denoted as E, generating a latent representation z_0 and converting z_0 by the VAE decoder, D, which maps z_0 to its corresponding image representation, denoted as x_0.

As shown in Figure 3.5, we fine-tune D to incorporate a predetermined message, denoted as m, into the generated image. This fine-tuning process ensures that the resulting image contains the desired message, which can be extracted using the dedicated watermark extractor, W_{ext}, as shown in Figure 3.5. An essential advantage of this approach is its compatibility with various tasks achieved by different latent diffusion models using the same VAE encoder/decoder. This compatibility stems from the fact that modifications made exclusively to D do not impact the diffusion process.

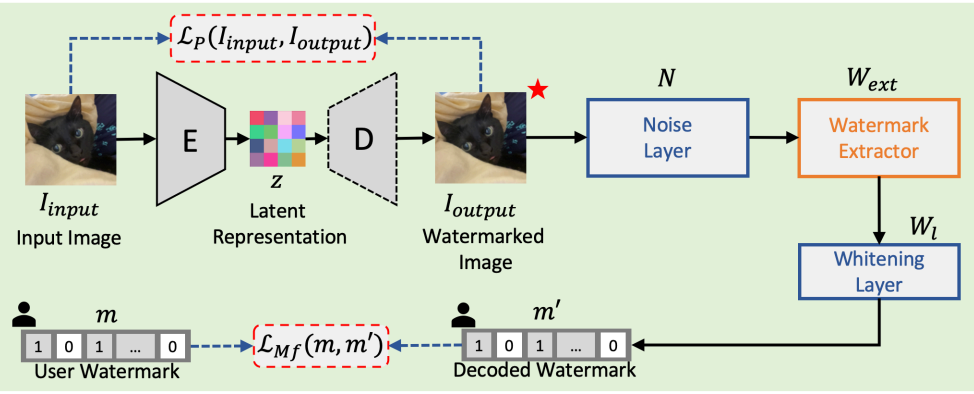


Figure 3.5: Fine-tuning phase. The only trainable module is the VAE decoder.

First, we define a fixed user-specific watermark message $m \in \{0,1\}^k$, then feed the input image I_{input} into E to get the latent representation z. Reconstruct the image I_{output} by

doi:10.6342/NTU202302023

D. By transforming I_{output} from the noise layer, we will have the noised image I_{no}. We are then passing I_{no} through W_{ext} to extract $m' \in [0,1]^k$. Considering the generative nature of the diffusion model, the absence of ground truth during the inferencing phase. In this regard, we utilize Watson-VGG as the perceptual loss, which ensures the preservation of the perceptual quality. The MSE loss is used for Message Reconstruction Loss(MRL) during the fine-tuning phase. We optimize the VAE decoder according to the loss over distributions of input images, that is

$$\mathbb{E}_{I_{input}} \left[\lambda_P \mathcal{L}_P(I_{input}, I_{output}) + \mathcal{L}_{Mf}(m, m') \right]$$

3.4 Noise layer

To ensure the robustness of images against diverse image transformations, the model incorporates the capability of handling such variations effectively. To achieve this, we introduce a noise layer incorporating four distinct types of transformations (cf. Figure 3.6 for details) as [3], each randomly chosen from the available set with equal probabilities for every batch processed. The **Identity** layer lets the image remain the same as I_w. The **Crop** layer randomly squares crops encoded images with size $S \times S$. We can control the size of the cropped image using the parameter $p \in [0,1]$, which is the ratio compared to the original image $(S \times S)/(H \times H)$. The **Resize** layer resizes the encoded image to size $H' \times W'$ controlled by $p \in [0,1]$, now the ratio of $(H' \times W')/(H \times H)$. The **JPEG** layer provides a differentiable approximation representation of the JPEG because the original JPEG distortion is non-differentiable. The JPEG compression technique partitions the image into 8×8 regions and applies a discrete cosine transformation (DCT) within each region. The frequency-domain coefficients obtained are quantized to varying degrees of

doi:10.6342/NTU202302023

coarseness, preserving only the perceptually important information. It is important to note that the quantization step in JPEG compression is non-differentiable, making it unsuitable for optimization based on gradient-based methods in training. To simulate the effects of JPEG compression, we employed a masking technique that restricted the representation of higher frequency information within the DCT coefficients. Specifically, we retained 25 low-frequency DCT coefficients in the Y channel and 9 in both the U and V channels, per the JPEG compression standard, which prioritizes preserving more information in the Y channel. The remaining DCT coefficients were effectively eliminated by setting them to zero.

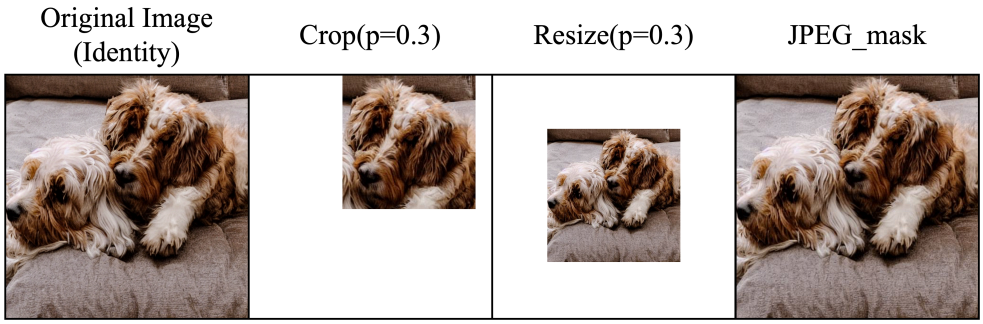


Figure 3.6: Illustration of all transformations used in the noise layer.

3.5 Whitening module

To address safety considerations, the extracted message obtained from vanilla images (i.e., non-watermarked images) must adhere to the Bernoulli distribution, representing a random bitstream distribution. This inherent randomness is paramount as it directly impacts the False Positive Rate (FPR) associated with vanilla images, explicitly referring to the probability of collision between the user-specific watermark message and the message extract from vanilla images. As an illustration, consider an 8-bit message where the desired collision probability is ideally $1/2^8$. However, assuming a scenario where

doi:10.6342/NTU202302023

the first 4 bits exhibit a strong correlation, such as the extreme case of the correlation coefficient 1, the consequence is that the first 4 bits extracted from vanilla images will invariably be identical. Consequently, any user whose watermark commences with the bit sequence (0,0,0,0) or (1,1,1,1) will face a substantially heightened probability of collision with vanilla images, specifically amounting to $1/2^5$.

To address this issue, a solution proposed by [3] involves implementing a PCA whitening transformation, which ensures that the extracted watermark message follows a Bernoulli distribution. After the pre-training stage, a set of vanilla images is required, from which the extracted messages are denoted as m_v. By calculating the mean, μ, and the covariance matrix Σ of m_v, we do an eigendecomposition of the covariance matrix $\Sigma = U \wedge U^T$. We construct a linear layer, named as the whitening layer is constructed, with bias $(b) = - \wedge^{-1/2} U^T \mu$ and weight $(w) = \wedge^{-1/2} U^T$. Because the whiten message's mean=0 and variance=1, we need to modify the bias and weight with $b = b/2 + 0.5$ and $w = w/2$ so that the new mean=0.5 and variance=0.25, which follows the Bernoulli distribution of random bit stream. This whitening layer is appended to the watermark extractor, and the parameters within the whitening layer are frozen during the fine-tuning stage. The combined watermark extractor and whitening layer can be viewed as a whitened watermark extractor, effectively mitigating the issue and aligning the extracted watermark messages with the desired Bernoulli distribution.

Nevertheless, it is observed that the original soft output generated by the watermark extractor from vanilla images exhibits a notably slight variance. In contrast, the soft output produced by the whitened watermark extractor displays a significantly more significant variance. This difference in variances disrupts training stability when a noise layer is introduced during the fine-tuning stage. To address this challenge, we propose the imple-

doi:10.6342/NTU202302023

mentation of a normalizing that squeezes the output distribution by leveraging a whitening scaling factor. This whitening scaling factor is derived from $f = tr(\Sigma_o)/tr(\Sigma_w)$, where $tr(\Sigma_o)$ is the trace of the covariance matrix observed from the soft output of the original watermark extractor, and Σ_w is the covariance matrix observed from the soft output of whitened watermark extractor. We seamlessly integrate the whitening scaling factor into the existing whitening layer, where $b = f \times (b - 0.5) + 0.5$ and $w = f \times w$. By incorporating this whitening scaling factor, we effectively mitigate the issue and ensure a more balanced and stable fine-tuning process.

3.6 Training procedure

Algorithm 1 Pre-Training Watermark Embedder/Extractor

Input: Cover image I_c
Watermark embedded: W_{emb}
Watermark extractor: W_{ext}
Noise layer: N
Watermark length: k
Initialize: Initialize parameters of W_{emb} and W_{ext}

for $steps \leftarrow 1$ **to** $total\_steps$ **do**
  sample Watermark message $M_{input} \sim \{0, 1\}^k$
  $I_w \leftarrow W_{emb}(I_c, M_{input})$
  $\mathcal{L}_I \leftarrow MSE(I_c, I_w)$ ▷ Image Reconstruction Loss
  $I_{no} \leftarrow N(I_w)$
  $M_{output} \leftarrow W_{ext}(I_{no})$
  $\mathcal{L}_{Mp} \leftarrow MSE(M_{input}, M_{output})$ ▷ Message Reconstruction Loss
  $\mathcal{L}_{pre\_training} \leftarrow \mathcal{L}_{Mp} + \lambda_I \mathcal{L}_I$
  update parameters of W_{emb} and W_{ext} with $\mathcal{L}_{pre\_training}$
end for
return

Our methodology can be divided into four stages to achieve the desired outcomes. Firstly, we initiate the process by pre-training the watermark embedder/extractor (Algorithm 1), yielding the watermark extractor used in stages 2, 3, and 4. Secondly, we construct the whitening layer (Algorithm 2), which plays a crucial role in stages 3 and 4 by

doi:10.6342/NTU202302023

whitening the output derived from the watermark extractor.

Algorithm 2 Construct Whitening Layer

Input: Vanilla image I_v
Input: Pre-trained Watermark Extractor W_{ext}
Watermark length: k
Whitening Layer: W_l
Whitening Layer bias: b
Whitening Layer weight: w

$M \leftarrow W_{ext}(I_v)$ ▷ $M \in \mathbb{R}^{N\times k}, I_v \in \mathbb{R}^{N\times H\times W\times C}$
$\mu \leftarrow Mean(M)$
$\Sigma \leftarrow Covariance\_Matrix(M)$ ▷ $\Sigma \in \mathbb{R}^{k\times k}$
$U \wedge U^T \leftarrow \Sigma$ ▷ eigendecomposition
$b \leftarrow - \wedge^{-1/2} U^T \mu/2 + 0.5$ ▷ $b \in \mathbb{R}^k$
$w \leftarrow \wedge^{-1/2} U^T/2$ ▷ $w \in \mathbb{R}^{k\times k}$
$M_w \leftarrow W_l(M)$ ▷ $M_w \in \mathbb{R}^{N\times k}$
$\Sigma_w \leftarrow Covariance\_Matrix(M_w)$
$f \leftarrow (tr(\Sigma)/tr(\Sigma_w))^{1/2}$ ▷ scaling factor
$b \leftarrow f \times (b - 0.5) + 0.5$ ▷ re-weighting bias
$w \leftarrow f \times w$ ▷ re-weighting weight
return

Moving on to the third stage, we fine-tune a VAE decoder for each user with a user-specific watermark (Algorithm 3). This stage ensures the effective integration of the user-specific watermark into the decoder. Thus, the image decoded from the decoder can be extracted from the user watermark by the extractor.

Finally, in the fourth stage (Algorithm 4), we carry out the watermark detection process, which determination of the user who generated the image or discerns whether the image is a vanilla image without any embedded watermark. This critical stage enables the accurate detection and attribution of watermarked images to their respective users, thereby ensuring the integrity and security of the watermarking system.

doi:10.6342/NTU202302023

Algorithm 3 Fine-tuning VAE decoder

Input: Input image I_{input}
Input: Pre-trained Watermark Extractor W_{ext}
Input: Constructed Whitening Layer W_l
Input: Pre-trained VAE Encoder E
Input: Pre-trained VAE Decoder D
Noise layer: N
User-specific watermark message: M_u
(Only D will be fine-tuned. Other modules will be frozen.)

for $steps \leftarrow 1$ **to** $total\_steps$ **do**
  $z \leftarrow E(I_{input})$ ▷ Encoding image
  $I_{output} \leftarrow D(z)$ ▷ Decoding latent representation
  $\mathcal{L}_P \leftarrow Watson\_VGG(I_{input}, I_{output})$ ▷ Perceptual Loss
  $I_{no} \leftarrow N(I_{output})$
  $M_{output} \leftarrow W_{ext}(I_{no})$
  $M_{whiten} \leftarrow W_l(M_{output})$
  $\mathcal{L}_{Mf} \leftarrow MSE(M_{input}, M_{whiten})$ ▷ Message Reconstruction Loss
  $\mathcal{L}_{fine\_tuning} \leftarrow \mathcal{L}_{Mf} + \lambda_P \mathcal{L}_P$
  update parameters of D with $\mathcal{L}_{fine\_tuning}$
end for
return

Algorithm 4 Watermark detection

Input: Detected image I
Input: Pre-trained Watermark Extractor W_{ext}
Input: Constructed Whitening Layer W_l
Matching threshold: τ
User watermark message database: DB

$M_{output} \leftarrow W_{ext}(I)$
$M_{whiten} \leftarrow W_l(M_{output})$
$M_r \leftarrow Round(M_{whiten})$ ▷ Rounding message bit with threshold 0.5
for M_i in DB **do**
  if $Matching(M_r, M_i) > \tau$ **then**
    $user\_id \leftarrow i$
    Image is generated by $user_i$
    return
  end if
end for
Image is vanilla image
return

doi:10.6342/NTU202302023

Chapter 4 Experiments

4.1 Setting

This section details the training dataset and hyperparameters employed in each experiment stage. The watermark message length is set to 48. We choose [3] as the baseline.

4.2 Watermark embedder/extractor

Following the HiDDeN[28] framework, we establish the watermark embedder and extractor using 64 hidden channels. The model is trained on the MS-COCO 2017[10] training dataset, employing random cropping to the images with size 256×256. The training process spans 150 epochs with a batch size of 32. Within the watermark embedder, a scaling factor of $\alpha = 0.3$ is utilized. The Image Reconstruction Loss parameter λ_I is set to 0.7. We employ the Adam optimizer for optimization, with the learning rate following a cosine annealing schedule. This schedule includes 2.5 epochs of linear warmup, gradually increasing the learning rate to 10^{-2} and decreasing it to 10^{-6} at the end of training. The model's training is conducted on a single GTX3090 GPU, which requires approximately three days to complete.

doi:10.6342/NTU202302023

4.2.1 Noise layer

During the training process, we incorporated different noise levels to enhance the model's robustness. These included Identity, Crop_p=0.3, Crop_p=0.7, Resize_p=0.3, Resize_p=0.7, and JPEG_mask. A noise type was randomly selected from this set of available noises with equal probabilities for each batch processed.

4.2.2 Whitening layer

We leverage a subset of 10,000 vanilla images randomly sampled from the MS-COCO 2017 training set to construct the whitening layer. We are employing the PyTorch library for the eigendecomposition calculations.

4.2.3 Fine-tuning VAE decoder

For our experimentation, we utilized the Stable Diffusion v1.5 VAE encoder/decoder architecture provided by HuggingFace[23]. To fine-tune the decoder, we randomly selected 400 images from the MS-COCO 2017 training dataset and applied random cropping with a size of 256×256. The training procedure encompassed five epochs, with a batch size of 4. Additionally, a Perceptual Loss parameter λ_P value of 0.2 was employed to enhance the perceptual quality of the generated images. We employed the Adam optimizer to optimize the model with a cosine annealing schedule for the learning rate. This schedule encompassed one epoch of linear warmup, gradually increasing the learning rate to 10^{-4}, and subsequently decreasing it to 10^{-6} towards the end of training. The model was trained on a single GTX3090 GPU, with an approximate training duration of three minutes.

doi:10.6342/NTU202302023

4.2.4 Testing

During the testing phase, we employed Stable Diffusion v1.5, obtained from HuggingFace, and replaced the VAE decoder with the fine-tuned VAE decoder. Following the established protocol in [14–16, 18], we generated 5000 images based on the captions provided in the MS-COCO 2017 validation set. We used DDPM sampler with a guidance scale of 3.0 and 50 diffusion steps. The generated image size is 512×512. The validation set consisted of 5000 images, each accompanied by five captions. We utilized the first caption for each image to generate the testing images, resulting in a total of 5000 testing images. Subsequently, we computed the Fréchet Inception Distance (FID)[6] score by comparing these generated images to the MS-COCO 2017 validation set, which was resized to 512×512. During the testing phase, four distortions, namely brightness, contrast, saturation, and sharpness, were applied to the images. These distortions provided by the PIL and Torchvision[12] libraries were not included in the training or fine-tuning phases but were employed to evaluate the robustness of the model. We adjust the default factor utilized for applying these distortions.

4.3 Metric

Researchers usually employ three key metrics when evaluating digital watermarking performance: imperceptibility, robustness, and capacity (cf. Figure 4.1). These metrics serve as the primary factors for analyzing the effectiveness of watermarking schemes. **Capacity** refers to the ability to embed important watermark information within the cover image. **Imperceptibility** ensures that the embedded watermark remains undetectable to human observers. **Robustness** measures the resistance of the cover image against

doi:10.6342/NTU202302023

various attacks or distortions encountered during normal usage. Recognizing that these three characteristics are interconnected is essential, giving rise to an inherent trade-off. **Capacity-Imperceptibility** trade-off: When the payload capacity is high, the detectability of the watermark message increases, leading to a decrease in the level of imperceptibility. **Robustness-Capacity** trade-off: The watermarked image incorporates additional redundancy to enhance its resilience against distortions, thus harming the capacity. **Imperceptibility-Robustness** trade-off: The redundancy to maintain robustness have a negative impact on imperceptibility. Conversely, a lack of redundancy makes detecting the watermark message harder. Our primary focus in this work is on the aspect of robustness.

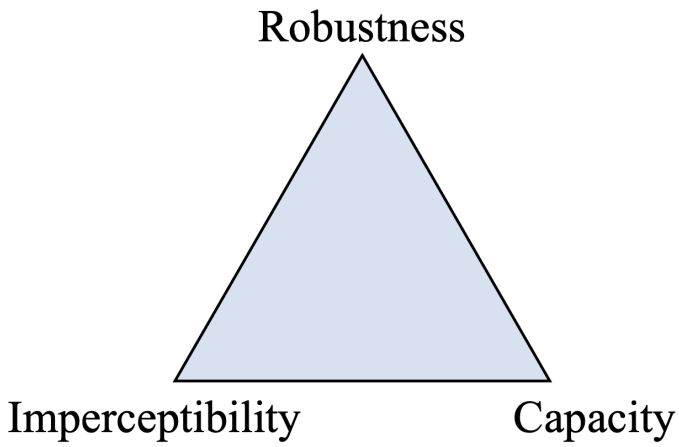


Figure 4.1: The watermark metric trade-off triangle.

Capacity: We fixed the watermark bit length in our entire experiment, which is set to 48 as the setting in [3]. **Imperceptibility**: Due to the inherent sampling nature of the inferencing phase in the diffusion model, ground truth for comparison is unavailable. As a result, we employ the Fréchet Inception Distance (FID) score as a metric to evaluate imperceptibility, which is commonly used for assessing the performance of generated images from the diffusion model. By utilizing FID, we can ensure that the original model's capabilities remain intact even after incorporating watermark embedding functionality. The lower the FID score, the better. **Robustness**: We utilize bitwise accuracy as a quantitative

doi:10.6342/NTU202302023

measure to assess the performance of our method under various levels of image distortion. The higher the bitwise accuracy, the better.

4.4 Adding noise layer during fine-tuning

In this section, we present the experimental results concerning our main contribution, which focuses on enhancing robustness by introducing a noise layer during the fine-tuning phase.

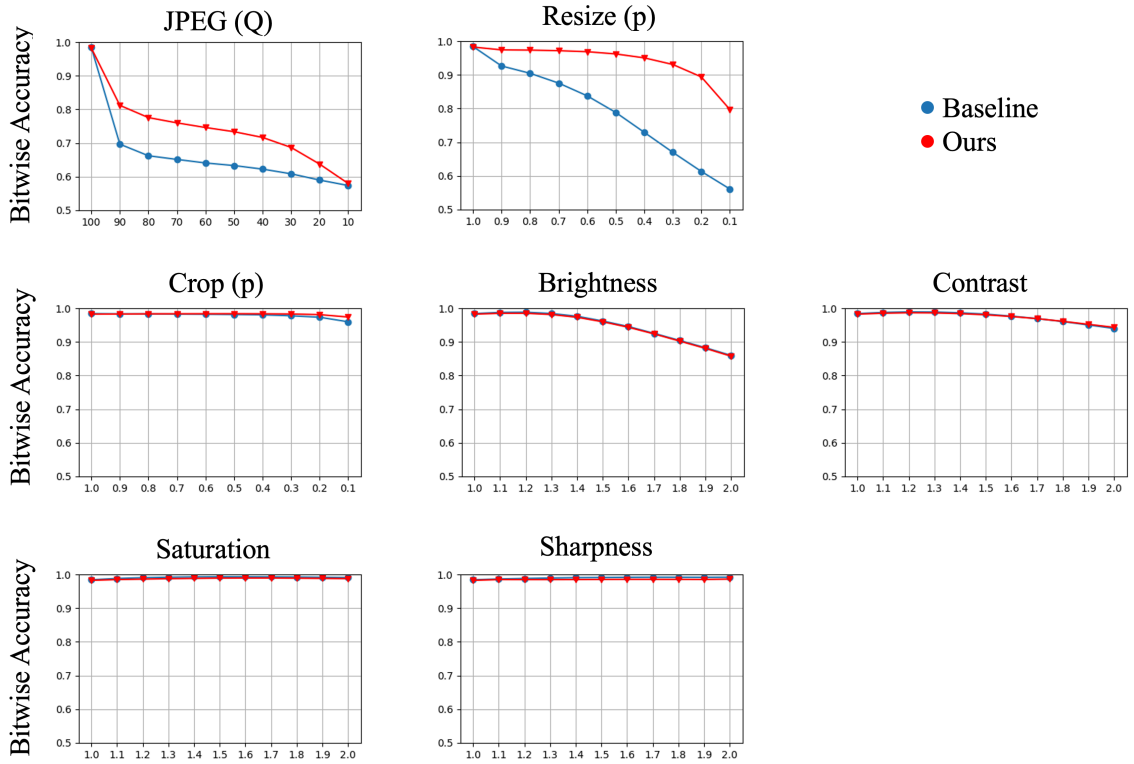


Figure 4.2: Bitwise accuracy between baseline and our method. (Top two charts depict the results associated with non-naturally robust distortions.)

4.4.1 Robustness

In the fine-tuning stage, our approach incorporates a noise layer between the VAE decoder and the watermark extractor. Figure 4.2 shows the results of our robustness com-

doi:10.6342/NTU202302023

Prompt	Model	Watermark method	FID
MS-COCO 2017 Validation dataset caption	Stable Diffusion V1.5	None	19.00
		Baseline	19.81
		Ours	19.70

Table 4.1: Imperceptibility performance in terms of FID scores

parison, highlighting the performance of our model relative to the baseline method. The results demonstrate that our model outperforms the baseline method when exposed to non-naturally robust distortions, such as Resize and JPEG. Conversely, our method performs on par with the baseline method when considering naturally robust distortions, such as Crop, Brightness, Contrast, Saturation, and Sharpness. These findings indicate that including non-naturally robust distortions during the fine-tuning process can enhance robustness compared to the baseline method.

4.4.2 Imperceptibility

In evaluating the effectiveness of our method, it is crucial to consider not only the bitwise accuracy of the embedded watermark message but also the impact on the functionality of the VAE decoder; especially, on the perceptual quality. Thus, we conducted a comparative analysis using the FID score, as presented in Table 4.1. Moreover, Figure 4.3 shows a Qualitative comparison between our method and the baseline approach.

The results obtained from the evaluation demonstrate that our method successfully maintains a balance between robustness and perceptual quality. Specifically, our approach did not compromise the perceptual quality of the generated images while enhancing their robustness against various image distortions. This outcome signifies a favorable outcome, as it affirms that our method achieves improved robustness without sacrificing the perceptual quality of the generated images.

doi:10.6342/NTU202302023

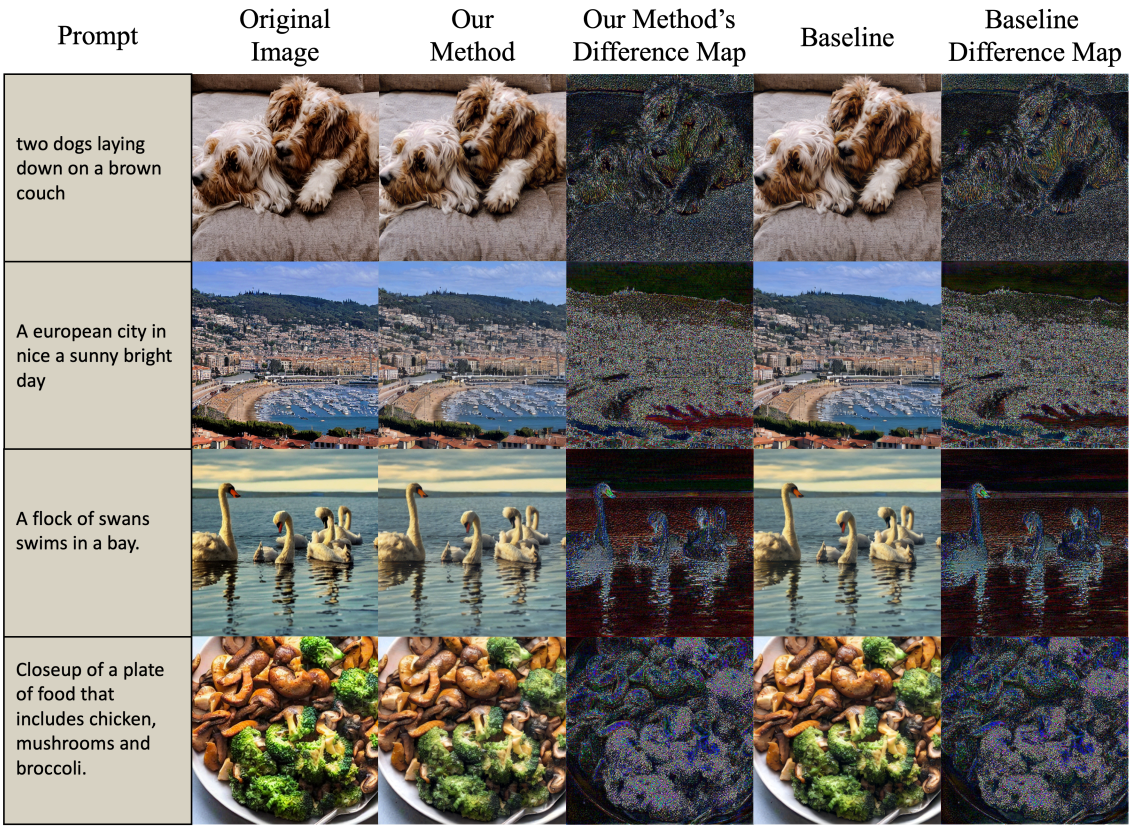


Figure 4.3: Qualitative comparison between our method and the baseline approach, associated with the same Prompt and resolution (set to 512 for all cases). Difference map is pixel-wise difference $\times 10$

doi:10.6342/NTU202302023

4.5 Whitening layer

Figure 4.4 depicts the covariance matrix of the bit stream output obtained from the original watermark extractor using 10k vanilla images. The left part of the figure reveals that the diagonal elements of the covariance matrix are characterized by small values, indicating a limited variance in the soft outputs (i.e., non-rounded outputs). The right part of the figure clearly illustrates that the covariance matrix of the rounded outputs deviates noticeably from the expected Bernoulli distribution, thereby highlighting the necessity of the whitening process.

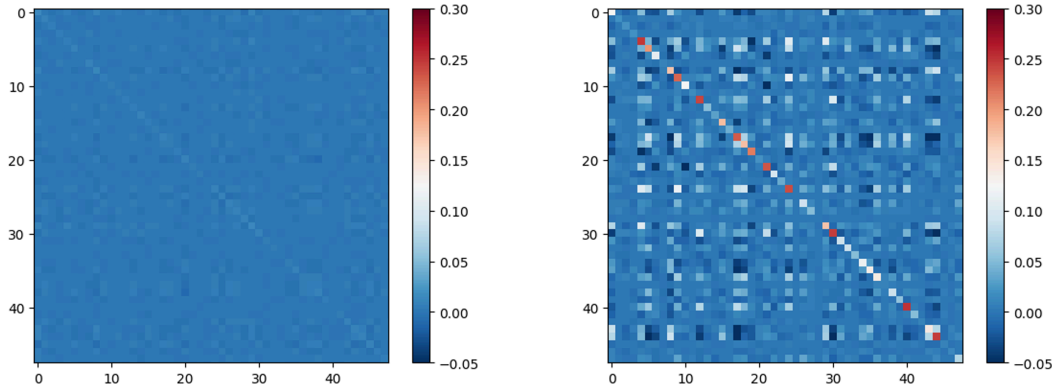


Figure 4.4: The vanilla images' Covariance matrices were extracted from the original watermark extractor. (Left) the non-rounded outputs and (right) the rounded outputs.

Following the whitening process in [3], the resulting covariance matrix is depicted in Figure 4.5. Notably, an observation can be made that the diagonal elements of the non-rounded covariance matrix closely align with those of the rounded one, both exhibiting conformity to the desired Bernoulli distribution. This alignment signifies a significant improvement, indicating that the issue of imbalanced collision probabilities among vanilla images has been effectively addressed. The achievement of this alignment is of utmost importance as it ensures a more equitable distribution of collision probabilities among vanilla images. This outcome enhances the fairness and reliability of the watermarking

doi:10.6342/NTU202302023

process, ensuring that the extracted bit stream adheres more consistently to the desired Bernoulli distribution and mitigating previously existing imbalances.

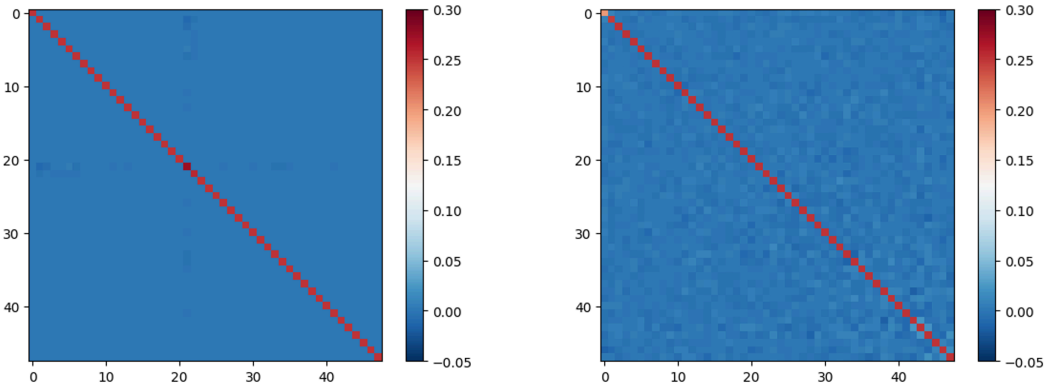


Figure 4.5: The vanilla images' Covariance matrices were extracted from the whitened watermark extractor. (Left) the non-rounded outputs and (right) the rounded outputs.

4.5.1 Re-weighting

However, it is worth noting that the whitening, as mentioned above, may introduce a potential issue. This challenge can be observed by comparing the soft output distributions before and after whitening, as depicted in Figure 4.4 and the left part of Figure 4.5, respectively. Particularly, when examining a single dimension of the bit stream, the distributions can be plotted, as shown in Figure 4.6. Following the whitening process, the new distribution exhibits a significantly wider spread than the original distribution.

This disparity in distribution width can notably impact the fine-tuning process, as it can lead to divergent convergence losses. We introduce a noise layer to exacerbate this issue further, creating undesirable and unmanageable turbulence during fine-tuning. Consequently, this poses a challenge in achieving a stable and effective fine-tuning of the model when incorporating the noise layer. We propose a re-weighting method to address this issue, as discussed in Section 3.5. Figure 4.7 shows that the re-weighting process

doi:10.6342/NTU202302023

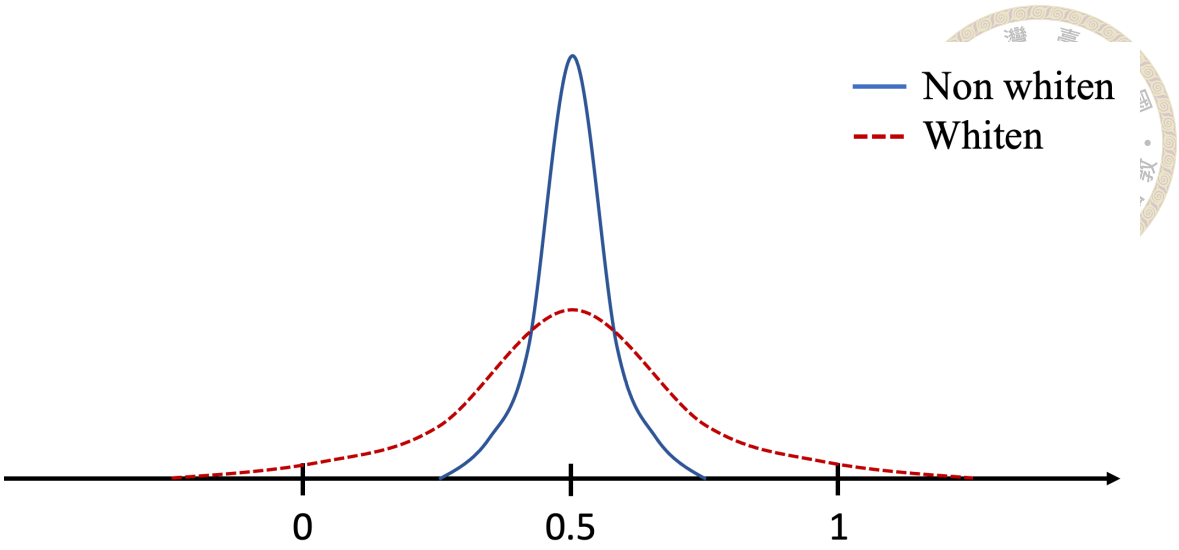


Figure 4.6: Distribution of non-rounded outputs (soft values) at single-bit dimension.

maintains compatibility between the diagonal elements in the covariance matrices of the non-rounded outputs with and without whitening while ensuring that the rounded output after the whitening layer follows the desired Bernoulli distribution. This enables us to achieve a stable fine-tuning stage even after introducing a noise layer, ultimately enhancing the robustness.

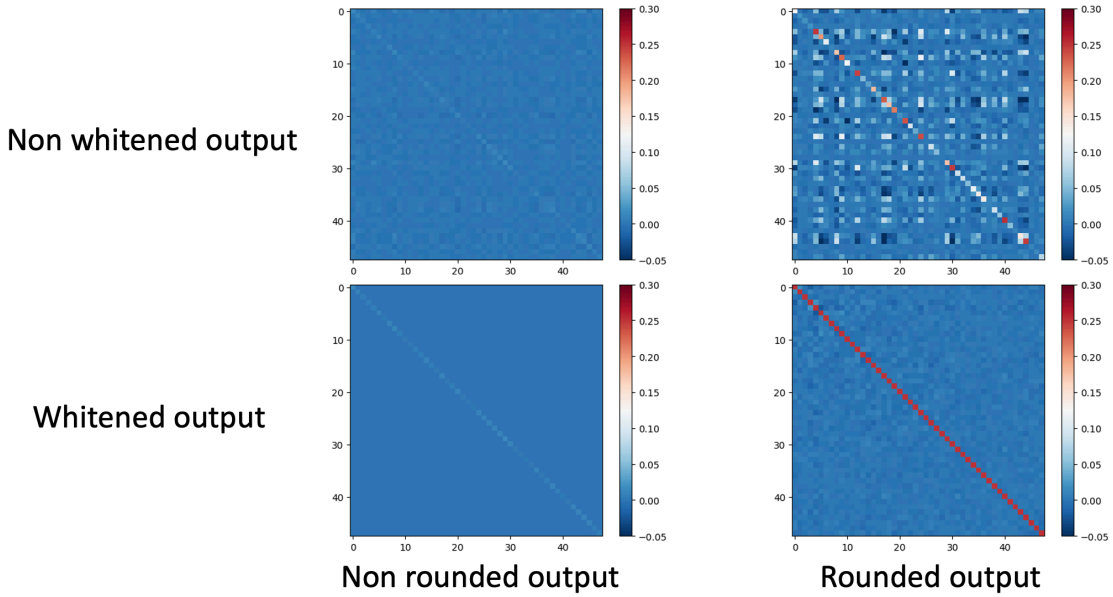


Figure 4.7: The Comparison of Covariance Matrices. (The bottom two images are the covariance matrices of the outputs extracted from the vanilla images by re-weighting the whitened watermark extractor.)

We explore another method to re-weight the whitening layer by applying bitwise

doi:10.6342/NTU202302023

normalization, which involves independently normalizing each output dimension's distribution. The results are illustrated in Figure 4.8. Interestingly, the robustness of the bitwise normalization method is lower when it comes to non-naturally robust distortions. This fact could be attributed to the whitening layer overfitting the characteristics of the 10k vanilla images as we manipulate the distribution on a per-bit basis.

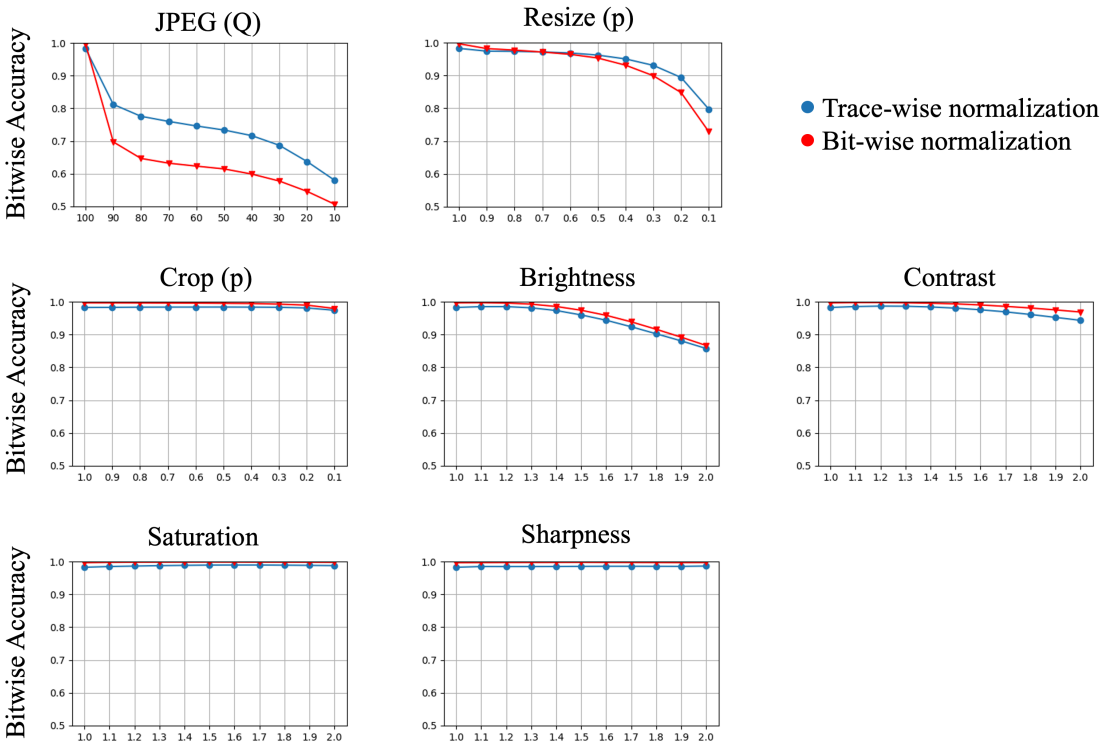


Figure 4.8: Bitwise accuracy between the trace-wise normalization method and the bit-wise normalization method. (Top two charts depict the results associated with non-naturally robust distortions.)

4.5.2 Bit accuracy imbalance

We have observed an additional effect introduced by the whitening layer. This effect manifests as an accuracy imbalance among different bits, as Figure 4.9 illustrates. Especially the last few bits exhibit significantly lower accuracy, indicating they are less reliable as part of the watermark. Furthermore, we observed that this phenomenon becomes more pronounced when incorporating a noise layer during fine-tuning.

doi:10.6342/NTU202302023

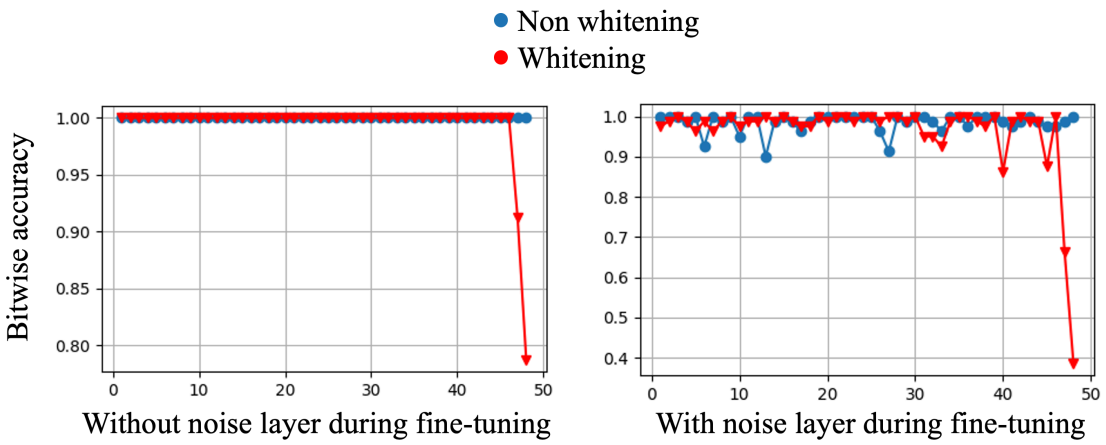


Figure 4.9: Bitwise accuracy in the validation phase of the fine-tuning stage.

We attribute this phenomenon to the disparity in eigenvalues. Some eigenvalues may be too small to contribute significantly to the overall data variance, resulting in higher noise-like characteristics. During eigendecomposition, the PyTorch library sorts the eigenvalues, causing the last few bits of the whitened bit stream to be associated with these noise-like eigenvalues. Consequently, noise-like eigenvalues can amplify, mainly when a noise layer is introduced during the fine-tuning stage.

doi:10.6342/NTU202302023

Chapter 5 Conclusions

In conclusion, our research significantly enhances the robustness of watermarking schemes by introducing a noise layer during the fine-tuning stage. Notably, we observe a significant improvement in the robustness of non-naturally robust distortions without adversely impacting the performance of naturally robust distortions. This improvement demonstrates our ability to enhance robustness without incurring any penalties. However, introducing the noise layer disrupts the stability of the fine-tuning process initially. We successfully addressed the issue by normalizing the whitening layer using a scaling factor derived from the trace from the covariance matrix of the whitened and non-whiten watermark messages.

Furthermore, our investigation uncovered a potential problem from the whitening layer: the accuracy imbalance among the extracted watermark bits. This issue poses a concern for the reliability and trustworthiness of the watermark bit stream. Overall, our work contributes to the diffusion model-related rooting watermarking field, particularly in improving robustness and addressing associated challenges.

For future research, one direction is to integrate a trainable attack network proposed in [11] as the noise layer. Eliminates the need for explicit modeling of image distortions during training. Additionally, further enhancing robustness can be achieved by explor-

doi:10.6342/NTU202302023

ing channel coding techniques for the watermark message. However, it is crucial to ensure mathematical proof regarding the safety enhancement and performance impact of the whitening layer, which remains a topic for future investigation.

doi:10.6342/NTU202302023

References

[1] T. Brooks, A. Holynski, and A. A. Efros. Instructpix2pix: Learning to follow image editing instructions. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 18392–18402, 2023.

[2] Y. Choi, M. Choi, M. Kim, J. Ha, S. Kim, and J. Choo. Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. In 2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018, pages 8789–8797. Computer Vision Foundation / IEEE Computer Society, 2018.

[3] P. Fernandez, G. Couairon, H. Jégou, M. Douze, and T. Furon. The stable signature: Rooting watermarks in latent diffusion models. CoRR, abs/2303.15435, 2023.

[4] P. Fernandez, A. Sablayrolles, T. Furon, H. Jégou, and M. Douze. Watermarking images in self-supervised latent spaces. In IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2022, Virtual and Singapore, 23-27 May 2022, pages 3054–3058. IEEE, 2022.

[5] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. C. Courville, and Y. Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors,

doi:10.6342/NTU202302023

Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada, pages 2672–2680, 2014.

[6] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, and R. Garnett, editors, Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA, pages 6626–6637, 2017.

[7] J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, editors, Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual, 2020.

[8] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila. Analyzing and improving the image quality of stylegan. In 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020, pages 8107–8116. Computer Vision Foundation / IEEE, 2020.

[9] D. P. Kingma and M. Welling. Auto-encoding variational bayes. In Y. Bengio and Y. LeCun, editors, 2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings, 2014.

[10] T. Lin, M. Maire, S. J. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft COCO: common objects in context. In D. J. Fleet, T. Pa-

doi:10.6342/NTU202302023

jdla, B. Schiele, and T. Tuytelaars, editors, Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V, volume 8693 of Lecture Notes in Computer Science, pages 740–755. Springer, 2014.

[11] X. Luo, R. Zhan, H. Chang, F. Yang, and P. Milanfar. Distortion agnostic deep watermarking. In 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020, pages 13545–13554. Computer Vision Foundation / IEEE, 2020.

[12] S. Marcel and Y. Rodriguez. Torchvision the machine-vision package of torch. In A. D. Bimbo, S. Chang, and A. W. M. Smeulders, editors, Proceedings of the 18th International Conference on Multimedia 2010, Firenze, Italy, October 25-29, 2010, pages 1485–1488. ACM, 2010.

[13] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever. Learning transferable visual models from natural language supervision. In M. Meila and T. Zhang, editors, Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event, volume 139 of Proceedings of Machine Learning Research, pages 8748–8763. PMLR, 2021.

[14] A. Ramesh, P. Dhariwal, A. Nichol, C. Chu, and M. Chen. Hierarchical text-conditional image generation with CLIP latents. CoRR, abs/2204.06125, 2022.

[15] A. Ramesh, M. Pavlov, G. Goh, S. Gray, C. Voss, A. Radford, M. Chen, and I. Sutskever. Zero-shot text-to-image generation. In M. Meila and T. Zhang, editors, Proceedings of the 38th International Conference on Machine Learning, ICML 2021,

doi:10.6342/NTU202302023

18-24 July 2021, Virtual Event, volume 139 of Proceedings of Machine Learning Research, pages 8821–8831. PMLR, 2021.

[16] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-resolution image synthesis with latent diffusion models. In IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022, pages 10674–10685. IEEE, 2022.

[17] C. Saharia, W. Chan, H. Chang, C. A. Lee, J. Ho, T. Salimans, D. J. Fleet, and M. Norouzi. Palette: Image-to-image diffusion models. In M. Nandigjav, N. J. Mitra, and A. Hertzmann, editors, SIGGRAPH '22: Special Interest Group on Computer Graphics and Interactive Techniques Conference, Vancouver, BC, Canada, August 7 - 11, 2022, pages 15:1–15:10. ACM, 2022.

[18] C. Saharia, W. Chan, S. Saxena, L. Li, J. Whang, E. L. Denton, S. K. S. Ghasemipour, R. G. Lopes, B. K. Ayan, T. Salimans, J. Ho, D. J. Fleet, and M. Norouzi. Photorealistic text-to-image diffusion models with deep language understanding. In NeurIPS, 2022.

[19] J. Sohl-Dickstein, E. A. Weiss, N. Maheswaranathan, and S. Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In F. R. Bach and D. M. Blei, editors, Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015, volume 37 of JMLR Workshop and Conference Proceedings, pages 2256–2265. JMLR.org, 2015.

[20] J. Song, C. Meng, and S. Ermon. Denoising diffusion implicit models. In 9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021. OpenReview.net, 2021.

doi:10.6342/NTU202302023

[21] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole. Score-based generative modeling through stochastic differential equations. In 9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021. OpenReview.net, 2021.

[22] A. van den Oord, O. Vinyals, and K. Kavukcuoglu. Neural discrete representation learning. In I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, and R. Garnett, editors, Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA, pages 6306–6315, 2017.

[23] P. von Platen, S. Patil, A. Lozhkov, P. Cuenca, N. Lambert, K. Rasul, M. Davaadorj, and T. Wolf. Diffusers: State-of-the-art diffusion models. https://github.com/huggingface/diffusers, 2022.

[24] N. Yu, V. Skripniuk, S. Abdelnabi, and M. Fritz. Artificial fingerprinting for generative models: Rooting deepfake attribution in training data. In 2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021, pages 14428–14437. IEEE, 2021.

[25] N. Yu, V. Skripniuk, D. Chen, L. S. Davis, and M. Fritz. Responsible disclosure of generative models using scalable fingerprinting. In The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022. OpenReview.net, 2022.

[26] L. Zhang and M. Agrawala. Adding conditional control to text-to-image diffusion models. CoRR, abs/2302.05543, 2023.

doi:10.6342/NTU202302023

[27] Y. Zhao, T. Pang, C. Du, X. Yang, N. Cheung, and M. Lin. A recipe for watermarking diffusion models. CoRR, abs/2303.10137, 2023.

[28] J. Zhu, R. Kaplan, J. Johnson, and L. Fei-Fei. Hidden: Hiding data with deep networks. In V. Ferrari, M. Hebert, C. Sminchisescu, and Y. Weiss, editors, Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part XV, volume 11219 of Lecture Notes in Computer Science, pages 682–697. Springer, 2018.

doi:10.6342/NTU202302023