Beyond Invisibility: Learning Robust Visible Watermarks for Stronger Copyright Protection

Tianci Liu[1] Tong Yang[2] Quan Zhang[3] Qi Lei[4]

[1]Purdue University
[2]Carnegie Mellon University
[3]Michigan State University
[4]New York University

Abstract

As AI advances, copyrighted content faces growing risk of unauthorized use, whether through model training or direct misuse. Building upon invisible adversarial perturbation, recent works developed copyright protections against specific AI techniques such as unauthorized personalization through DreamBooth that are misused. However, these methods offer only short-term security, as they require retraining whenever the underlying model architectures change. To establish long-term protection aiming at better robustness, we go beyond invisible perturbation, and propose a universal approach that embeds *visible* watermarks that are *hard-to-remove* into images. Grounded in a new probabilistic and inverse problem-based formulation, our framework maximizes the discrepancy between the *optimal* reconstruction and the original content. We develop an effective and efficient approximation algorithm to circumvent a intractable bi-level optimization. Experimental results demonstrate superiority of our approach across diverse scenarios.

1 INTRODUCTION

Deep generative models (DGMs), such as diffusion models [71, 28], have shown remarkable success in various vision tasks, including text-to-image generation [65], image editing [10, 69], and style transfer [35]. Moreover, these models exhibit impressive personalization capabilities [24, 66]. For example, DreamBooth [66] fine-tunes diffusion models using a few representative reference images, enabling the generation of personalized images with high fidelity. This capability significantly reduces the cost of AI-assisted personalized generation, paving the way for a wider range of AI-driven applications [81].

However, these advances also introduce new risks. Artists and photographers frequently share their works online for promotional purposes. Yet, off-the-shelf AI tools enable malicious users to obtain unauthorized copies without purchasing rights or to directly plagiarize art styles by fine-tuning personalized models using these images [74, 52]. These threats greatly undermine the profits of art creators [68].

Recent studies developed *adversarial attacks* to defend against unauthorized use of AI tools [67, 68, 43, 74]. These methods learn *invisible perturbations* to disrupt the image generation process in DGMs like diffusion models. For example, [67] push the latent codes of text-to-image diffusion models toward unrelated targets, and AdvDM [44] minimizes the likelihood of perturbed images from diffusion models to degrade their performance on them. Poisoning attacks have also been used to trick fine-tuning based DreamBooth into learning false correlations, preventing it from capturing desired styles [74], and MetaCloak [52] incorporated meta-learning to attack an ensemble of diffusion models, improving the poisoning transferability.

Although effective on targeted models, these invisible attack-based solutions heavily rely on adversarial vulnerabilities, resulting in two key limitations. First, the *adversarial attack-based* mechanism makes them fall short to generalize well to broader DGMs [31, 52]. Specifically, their performance on black-box DGMs is largely unpredictable [16], and on white-box DGMs, they only provide *short-term* protection: when facing new DGMs, the perturbation must also be updated or retrained [80]. Second, their *invisibility* inherently limits their strength from two aspects. On one hand, invisible watermarks are prone to distortion and purification attacks [3, 52, 86]. On the other, since these protections are designed to be invisible, they cannot prevent direct misuse such as scraping copyrighted content for commercial use without authorization.

In response, we propose a new paradigm for copyright protection. Our approach revisits the visible watermark, a traditional tool for copyright protection. We demonstrate that

visible watermarks offer strong protection: with clear copyright information displayed, the image becomes largely unusable. Additionally, when a prominent visible watermark is present, AI tools like DreamBooth learn the watermark pattern due to their backdoor mechanism [64, 59, 11], resulting in unsatisfactory outputs. Finally, our protection is agnostic to misuse: unlike attack-based methods, visible watermarking does not target specific misuses or DGMs, thus providing a universal protection.

Another advantage of visible watermarking is its robustness to distortion attack, such as JPEG compression and Gaussian blur, that can easily compromise its invisible counterpart [3, 86]. The existence of *watermark removal* as *targeted* attack on visible watermarking also poses a significant challenge [51, 45, 55, 48]. Since standard mechanism that adds visible watermarks in a consistent way can be bypassed by specialized attacks [15], and manually placing watermarks in appropriate areas [76] can be labor-intensive and not scalable, we propose HARVIM, which *learns* a visible watermark that is hard to remove in an automated way. *To our best knowledge, this is the first learning-based visible watermark for copyright protection of human-created content in the AI era. This new exploration is a key contributions.*

Formally, HARVIM transforms watermark removal into an inpainting problem of reconstructing the watermarked area, and learns a watermark to make the reconstruction harder. This entails a bi-level optimization. The lower-level optimization reconstructs the watermarked area, and the upper-level optimization adjusts the watermark to push the reconstruction away from the original image. Through this formulation, HARVIM identifies a hard-to-reconstruct region of the image, usually containing rich visual details. Importantly, This region is an *intrinsic* characteristic of the image, allowing HARVIM to create watermarks that are *inherently* hard to remove, regardless of the removal method used. *The new hard-to-remove watermark formulation as a universal copyright protection is also a key contribution of this work.*

In execution, HARVIM uses a pre-trained generative model as a *prior* to guide lower-level optimization [5, 2, 58]. However, this generative prior makes the bi-level problem NP-hard [40, 70], due to the complexity of how the watermark impacts the lower-level optimal solution involving a deep neural network (DNN). Following prior work [31, 52], we use meta-learning [22] for an approximate solution, replacing the exact lower-level solution with one that takes K gradient descent steps from the initial value [22]. Expressing the gradients as functions of the watermark allows the approximation to be written as an explicit function of the watermark. Meta-learning requires K to be small, usually leading to approximation errors [31, 25]. Nonetheless, recent work showed that a special family of deep generative priors allows the lower-level optimization to be replaced by a series of subproblems, each solvable *in a few steps* [48]. Built upon this, we derive a new, effective solution to learn watermark. *This new bi-level solver is our third contribution.*

Our paper is organized as follows. Sec 2 discusses the HARVIM formulation and its approximate solution. Sec 3 evaluates HARVIM's performance on various image sets and tests its robustness against different watermark removers. Sec 4 reviews related works, and Sec 5 conclude the paper.

2 PROPOSED METHOD

Grounded in a probabilistic view, We propose HARVIM to learn *hard-to-remove* visible watermarks to protect copyrighted images from direct and AI-assisted misuse.

2.1 PRELIMINARY

Notations. As in previous works [58, 79, 48], we represent (flattened) images as vectors denoted by lowercase boldface letters. Uppercase boldface letters mark matrices.

Inverse problems. Given a corrupted observation $\boldsymbol{y} \in \mathbb{R}^m$ of an unknown image $\boldsymbol{x}_T \in \mathbb{R}^n$ ($m \leq n$), inverse problems aim to reconstruct clean $\boldsymbol{x}_T$ assuming that $\boldsymbol{y}$ is generated by

$$\boldsymbol{y} = f(\boldsymbol{x}_T) + \boldsymbol{e}, \tag{1}$$

where $f(\cdot)$ is a known forward operator that corrupts $\boldsymbol{x}_T$, and $\boldsymbol{e}$ is a noise that has independently and identically distributed elements [5, 58]. Inverse problems, like compressed sensing and inpainting, are associated with a specific operator f. For more background, see [58]. Our work focuses on image inpainting.

Image inpainting. This task aims to recover an image with masked content. Formally, inpainting assumes that $\boldsymbol{y} = \mathbf{A}\boldsymbol{x}_T + \boldsymbol{e}$, where $\mathbf{A} \in \mathbb{R}^{n \times n}$ is a diagonal matrix with binary entries indicating whether a pixel is observed or missing, and $\boldsymbol{e} \sim \mathcal{G}(\mathbf{0}; \sigma^2\mathbf{I})$ is an isotropic Gaussian noise with known variance σ^2.

Deep Generative Prior. Inverse problems are generally under-determined, in the sense that Eq (1) admits infinitely many possible solutions. To address this, deep generative models (DGMs) pre-trained on large datasets can be used as *priors* to assess the plausibility of reconstructions and help find the optimal one [58]. From a Bayesian perspective, this entails a *maximum-a-posterior* (MAP) problem. Let G be a DGM prior. We solve the inverse problem by finding

$$\begin{aligned} \boldsymbol{x}^* &= \operatorname{argmax}_{\boldsymbol{x}} \log p_G(\boldsymbol{x} \mid \boldsymbol{y}; \lambda) \\ &= \operatorname{argmax}_{\boldsymbol{x}} \log p_e(\boldsymbol{y} - f(\boldsymbol{x})) + \lambda \log p_G(\boldsymbol{x}), \end{aligned} \tag{2}$$

$\log p_e(\cdot)$ and $\log p_G(\cdot)$ represent the log-likelihood of noise $\boldsymbol{e}$ and image $\boldsymbol{x}$, respectively. The hyperparameter $\lambda > 0$ controls the weight of the prior G, acting as a regularizer [79].

Copyrighted Image Protection. The advance of DGMs also enables unauthorized use of copyrighted content. For

instance, DreamBooth [66] allows text-to-image diffusion models [65] to generate personalized images. However, by fine-tuning on a few of an artist's work, it can mimic and plagiarize their style [74]. This has raised significant concerns about copyright protection [68]. To counter this, recent works [43, 74] proposed *targeted* attacks on DGMs like DreamBooth being misused. Conceptually, given a misused DGM G with training loss $\ell(G; \boldsymbol{x})$ for any $\boldsymbol{x}$, these works protected copyrighted image $\boldsymbol{x}_T$ by learning an invisible perturbation $\boldsymbol{\delta}$ via $\max_{\boldsymbol{\delta}:\|\boldsymbol{\delta}\|_\infty<\varepsilon} \ell(G; \boldsymbol{x}+\boldsymbol{\delta})$ to degrade G's performance on $\boldsymbol{x}+\boldsymbol{\delta}$, where ε limits pixel-level perturbation. This defines an adversarial attack on G. When G is inaccessible, δ is learned by attacking (an ensemble of) open-source surrogate models [52].

2.2 HARVIM: TOWARDS A UNIVERSAL PROTECTION BY VISIBLE WATERMARKING

As outlined before, although attack-based safeguards can effectively address targeted misuse, they have key weakness. First, the attack-based formulation limits their applicability in *untargeted* scenarios. Specifically, their performance on black-box AI is largely unpredictable due to the nature of the attack [16, 52], and in white-box settings, they provide only *short-term* protection, in the sense that new personalization techniques may render current safeguards (e.g., those against DreamBooth) ineffective [52, 80]. In addition, the *invisible* nature of existing protections also poses two inherent limitations. First, these protections are prone to distortion or purification attack [3, 52, 86]. Second, the *invisibility* offers no protection against *direct misuse*. We refer to a misuse as direct if it does not involve AI, but rather unauthorized use such as piracy. For instance, users may scrape copyrighted images for commercial purposes without purchasing rights, undermining creators' profits. Such misuse doesn't involve AI tools and cannot be addressed by existing attack-based methods. Consequently, existing safeguards often provide unsatisfactory protection in execution [52, 86]. Hence, a more general formulation for protection is needed.

In light of these limitations, we resort to visible watermarking for stronger protection. First, visible watermarks render protected images largely unusable in direct use. In AI-involved misuse scenarios, when a prominent watermark presents, AI such as personalization with DreamBooth will also be affected due to the backdoor mechanism [64, 59, 11]. As shown in Fig 1, DreamBooth learns watermark patterns from watermarked training images, leading to unusable outputs. Notably, adding visible watermarks requires no prior domain knowledge of misuse scenarios or mechanisms. Thus, it provides a broader protection. In addition, visible watermarking are much more robust to distortion attacks. In Fig 2 we applied JPEG compression [20, 4] and Gaussian blur [84] at varied intensities to distort watermarked images, and observed that the watermarks remain readable even when the images are greatly destroyed.

Our finding indicates that visible watermark offers an excellent level of robustness against standard transformation attack, and pave the way for more reliable copyright protection than existing attempts. Nonetheless, conventional watermarks are typically added in a consistent manner to the images, which offers limited resistance against more targeted watermark removal attack [15, 45, 73]. While manual or rule-based watermark placement can provide some protection [32], it requires significant human effort and lacks scalability. To address this, we propose an *automated* solution by *learning* a visible watermark that is resistant to remove. We refer to our approach as *hard-to-remove visible watermark* (HARVIM) and provide details below.

2.3 FORMAL FORMULATION OF HARVIM

We formulate the proposed HARVIM as an optimization problem. To this end, we define a watermark $\boldsymbol{m} \in \mathbb{R}^n$ as an image with the same dimensions[1] as the copyrighted image $\boldsymbol{x}_T$. Then, watermark removal can be formulated as an inverse problem [58], where the watermarked observation is[2] $\boldsymbol{y}(\boldsymbol{m}) = \mathbf{A}_m \boldsymbol{x}_T + \boldsymbol{e}$. Similar to inpainting, $\mathbf{A}_m$ is a diagonal matrix where entries indicate if a pixel is watermarked. Treating the watermarked area as missing, inpainting serves as a surrogate for visible watermark removal [30].

Built upon this formulation, HARVIM seeks an $\boldsymbol{m}$ that makes $\boldsymbol{x}_T$ *hard to reconstruct* from observation $\boldsymbol{y}(\boldsymbol{m})$. The *reconstruction hardness* is measured by a similarity score $s(\boldsymbol{x}^*(\boldsymbol{m}), \boldsymbol{x}_T)$ between the optimal reconstruction $\boldsymbol{x}^*(\boldsymbol{m})$ from $\boldsymbol{y}(\boldsymbol{m})$ to the ground truth $\boldsymbol{x}_T$.

Watermarking constraints. When learning $\boldsymbol{m}$ for copyrighted image protection, two standard *readability* constraints must be met [57, 32]. First, **image readability** requires that the watermarked observation's readability must remain. Otherwise, while an excessive watermark occupying the entire image can make it unrecoverable, audience will also fail to recognize the image content, which could negatively compromise the creator's financial gains and public visibility. This constraint is solved by adding a regularization term $\mathcal{R}(\boldsymbol{m})$ to penalize the size of watermark. Second, **watermark readability** requires that the watermark itself should convey clear copyright information, such as the creator's logo or name. To satisfy this constraint, we use a small pre-trained generative model to control $\boldsymbol{m}$, as detailed in Appendix A.1 due to page limit.

Put together, HARVIM learns $\boldsymbol{m}$ to watermark image $\boldsymbol{x}_T$ by

[1] The background is also part of the image.

[2] We write the observation $\boldsymbol{y}$ (or reconstruction $\boldsymbol{x}^*$) as a function of $\boldsymbol{m}$ (and hyperparameter λ) to highlight the dependence.

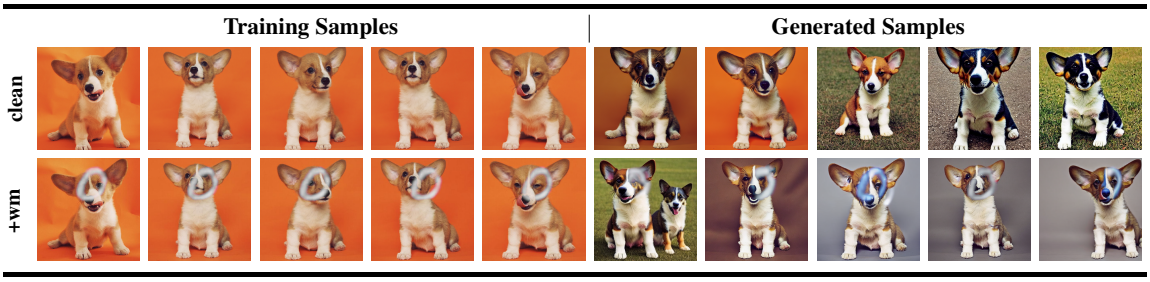

Figure 1: Visible Watermarking can provide strong protection: DreamBooth trained on watermarked ("+wm") images learn watermark patterns as well. Examples and implementations are from [75].

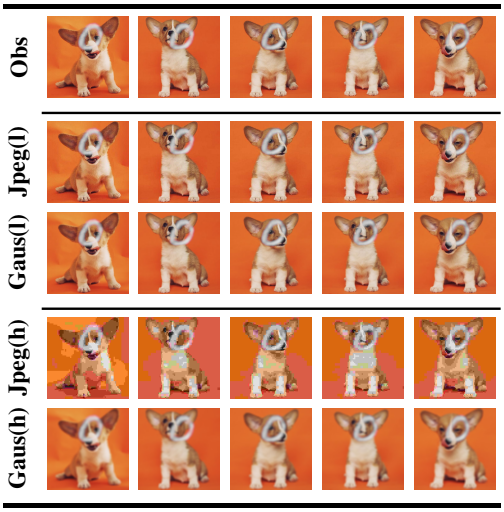

Figure 2: Visible watermarks remain resilient to strong distortion attacks JPEG compression and Gaussian blur, at low- (top) and high-intensity (bottom) levels.

solving a bi-level optimization problem

$$\begin{aligned} &\boldsymbol{m}^* = \min_{\boldsymbol{m}} s(\boldsymbol{x}^*(\boldsymbol{m}), \boldsymbol{x}_T) + \mathcal{R}(\boldsymbol{m}), \\ &\text{s.t. } \boldsymbol{x}^*(\boldsymbol{m}) = \operatorname{argmax}_{\boldsymbol{x}} \log p_G(\boldsymbol{x} \mid \boldsymbol{y}(\boldsymbol{m}); \lambda). \end{aligned} \tag{3}$$

Remark 2.1. We want to emphasize that Eq (3) provides a general framework for image protection for two key reasons. First, the concept of *hard-to-reconstruct region* underlying HARVIM reflects an intrinsic characteristic of an image, rather than a property specific to any particular prior G. Second, Eq (3) is not limited to any specific choice of G. The next section presents an implementation, but HARVIM by definition can incorporate any generative prior G capable of modeling the real image distribution.

2.4 AN APPROXIMATE SOLUTION FOR HARVIM

The bi-level optimization Eq (3) is non-trivial to solve, with difficulties lying in two folds. First, inpainting requires matrix $\mathbf{A}_m$ containing binary entries, which cannot be optimized by gradient-based method. Second, its feasible set, as specified by the lower-level optimization that involves some deep neural network G, is NP-hard to identify [70]. Therefore, further approximations are needed.

Mathematically, the first challenge arises from that $\boldsymbol{m}$'s gradient is undefined due to the discrete nature of $\mathbf{A}_m$. To address this issue, we construct a differentiable approximation for it based on continuous-valued learnable watermark $\boldsymbol{m}$. Specifically, given $\boldsymbol{m} \in \mathbb{R}^n$, denote the sigmoid function by $\text{sig} : \mathbb{R} \to \mathbb{R}$, we define

$$\mathbf{A}_m = \text{diag}\left(\text{sig}\left(\frac{m_1 - \alpha}{\beta}\right), \ldots, \text{sig}\left(\frac{m_n - \alpha}{\beta}\right)\right), \tag{4}$$

where α, β are hyperparameters such that $\text{sig}((m_i - \alpha)/\beta) \approx 1$ when m_i lies within the watermark area, and 0 otherwise. Additional implementation details are provided in Appendix A.1.

The differentiable $\boldsymbol{m}$ can be optimized with gradient

$$\begin{aligned} &\nabla_{\boldsymbol{m}} \left(s(\boldsymbol{x}^*(\boldsymbol{m}), \boldsymbol{x}_T) + \mathcal{R}(\boldsymbol{m})\right) \\ =&\nabla_{\boldsymbol{x}^*} s(\boldsymbol{x}^*(\boldsymbol{m}), \boldsymbol{x}_T)^\top \mathbf{J}_{\boldsymbol{m}}(\boldsymbol{x}^*(\boldsymbol{m})) + \nabla_{\boldsymbol{m}} \mathcal{R}(\boldsymbol{m}), \end{aligned} \tag{5}$$

where the second line holds from the chain rule, and $\mathbf{J}_{\boldsymbol{m}}(\boldsymbol{x}^*(\boldsymbol{m}))$ denotes the Jacobian of $\boldsymbol{x}^*$ with respect to $\boldsymbol{m}$. Unfortunately, this Jacobian is intractable due to the unknown form of $\boldsymbol{x}^*(\boldsymbol{m})$, making the problem remain unsolved. We resort to meta-learning for an approximate solution [31] by replacing the exact $\boldsymbol{x}^*(\boldsymbol{m})$ with an approximate solution $\tilde{\boldsymbol{x}}(\boldsymbol{m})$ that is computed from K-step gradient descent [22]. By treating $\nabla_{\boldsymbol{x}} \log p_G(\boldsymbol{x} \mid \boldsymbol{y}(\boldsymbol{m}); \lambda)$ as a function of $\boldsymbol{m}$, $\tilde{\boldsymbol{x}}(\boldsymbol{m})$ can be expressed as an explicit function of $\boldsymbol{m}$, making approximation Eq (5) viable.

In practice, however, meta-learning requires small K to maintain affordable computational cost, and 1 or 2 is often used [31]. Such a small value often results in highly inaccurate approximation [25], Critically, when the approximation fails to reflect the faithful progress made by current $\boldsymbol{m}$, the upper-level optimization will be misled as well, resulting in poor or failed solutions [26, 23].

Idea. While this difficulty cannot be resolved in general, for inverse problems solvers that use normalizing flows [60]

Algorithm 1 HARVIM algorithm

Input: copyrighted image $\boldsymbol{x}_T$, $\lambda > 0$ and its update steps $T > 0$, random noise variance $\sigma^2 > 0$, generative prior $G : \mathbb{R}^n \to \mathbb{R}^n$, inpainting mask hyperparameters α, β (Eq (4)), unrolled steps K
Initialize: $\lambda_0 = 0$, randomly initialize $\boldsymbol{m}_0$ and inpainting mask $\mathbf{A}_{m,0}$ based on Eq (4), watermarked image $\boldsymbol{y}_0 = \mathbf{A}_{m,0}\boldsymbol{x}_T + \boldsymbol{e}$ where $\boldsymbol{e} \sim \mathcal{G}(\mathbf{0}, \sigma^2\mathbf{I})$
Ignoring dependency on $\boldsymbol{m}_0$, find the MLE solution $\tilde{\boldsymbol{x}}_0 = \tilde{\boldsymbol{x}}(\boldsymbol{m}_0, \lambda_0)$ for $\boldsymbol{y}_0$ [48]
for $t = 1, \ldots, T$ **do**
  Treat $\boldsymbol{y}_{t-1}(\boldsymbol{m}_{t-1}) = \mathbf{A}_{m,t-1}\boldsymbol{x}_T + \boldsymbol{e}, \boldsymbol{e} \sim \mathcal{G}(\mathbf{0}, \sigma^2\mathbf{I})$ as a function of $\boldsymbol{m}_{t-1}$
  $\lambda_t = \lambda_{t-1} + \frac{\lambda}{T}$
  $\tilde{\boldsymbol{x}}_t = \tilde{\boldsymbol{x}}_{t-1}$
  for $k = 1, \ldots, K$ **do**
    $\tilde{\boldsymbol{x}}_t = \tilde{\boldsymbol{x}}_t + \nabla_{\boldsymbol{x}} \log p(\boldsymbol{x} \mid \boldsymbol{y}_{t-1}(\boldsymbol{m}_{t-1}); \lambda_t)$
  end for
  Denote current solution as $\tilde{\boldsymbol{x}}_t(\boldsymbol{m}_{t-1}, \lambda_t)$
  Update $\boldsymbol{m}_t$ based on Eq (6)
end for
return Learned $\boldsymbol{m}_t$

as generative priors, it can be largely alleviated. In specific, denote $\boldsymbol{x}^*(\boldsymbol{y}; \lambda) = \text{argmax}_{\boldsymbol{x}} \log p_G(\boldsymbol{x} \mid \boldsymbol{y}; \lambda)$, [48] showed that under regular conditions, $\log p_G(\boldsymbol{x} \mid \boldsymbol{y}; \lambda')$ is locally convex at $\boldsymbol{x}^*(\boldsymbol{y}; \lambda)$ when λ' is close enough to λ. Therefore, using $\boldsymbol{x}^*(\boldsymbol{y}; \lambda)$ as an initial value, $\boldsymbol{x}^*(\boldsymbol{y}; \lambda')$ by nature can be obtained within *a few* gradient descent steps. Motivated by this, we expect $\log p_G(\boldsymbol{x} \mid \boldsymbol{y}; \lambda)$ to preserve a local convexity around $\boldsymbol{x}^*(\boldsymbol{y}'; \lambda')$ if $\boldsymbol{y}'$ is close to $\boldsymbol{y}$ and λ' is close to λ. Built upon this, we optimize $\boldsymbol{m}$ along with $\tilde{\boldsymbol{x}}(\boldsymbol{m})$ and λ as in [48] together in an iterative way.

Solution. Our solution starts with a randomly initialized watermark $\boldsymbol{m}_0$, hyperparameter $\lambda_0 = 0$, and an approximate solution $\tilde{\boldsymbol{x}}(\boldsymbol{m}_0; \lambda_0)$ solved by gradient descent. Here the approximate solution $\tilde{\boldsymbol{x}}$ is expressed as a function of both watermark $\boldsymbol{m}$ and λ. In each round t, we first update hyperparameter λ_t by taking a small step towards the final λ. Next, given current $\boldsymbol{m}_{t-1}$ and λ_t, we solve $\tilde{\boldsymbol{x}}_t(\boldsymbol{m}_{t-1}; \lambda_t)$ by taking K gradient descent steps from the last round solution $\tilde{\boldsymbol{x}}_{t-1}$. Finally, we update $\boldsymbol{m}_t$ by unrolling updates on $\tilde{\boldsymbol{x}}_t(\boldsymbol{m}_{t-1}; \lambda_t)$ as a function of $\boldsymbol{m}_{t-1}$ and take

$$\boldsymbol{m}_t = \boldsymbol{m}_{t-1} - \nabla_{\boldsymbol{m}} \left(s(\underbrace{\tilde{\boldsymbol{x}}_t(\boldsymbol{m}_{t-1}; \lambda_t)}_{\text{func. of } \boldsymbol{m}_{t-1}}, \boldsymbol{x}_T) + \mathcal{R}(\boldsymbol{m}_{t-1}) \right). \tag{6}$$

We repeat the following steps until λ_t reaches the prespecified value λ. The solution is outlined in Algo 1.

3 EXPERIMENTS

In this section, we evaluate the performance of HARVIM in learning various types of watermarks across diverse image distributions. Importantly, HARVIM employs a simpler G to acquire information about *reconstruction hardness* for guiding watermark optimization, and learned watermarks are capable of resisting more advanced watermark removal techniques. These results confirmed the versatility of HARVIM.

3.1 EXPERIMENT SETUP

HARVIM Setup. Following Whang et al. [79], Liu et al. [48], we use representative normalizing flow model RealNVP [19] as pre-trained on CelebA [53] as a reliable generative prior G [47]. More training details can be found in Whang et al. [79]. We adopt peak-signal-to-ratio (PSNR) to measure similarity between reconstruction $\tilde{\boldsymbol{x}}$ and ground truth $\boldsymbol{x}_T$ that HARVIM seeks to minimize in Eq (3).

Learnable Watermarks. We consider two families of learnable watermarks for empirical study. The **logo-styled** watermarks are simulated by MNIST digits [39], and we use all digits 0-9. The **initial-styled** watermarks, on the other hand, are constructed from handwritten English letters [12], and we choose two randomly selected initials, "NJ" and "OS". All watermark generators are implemented by lightweight variational auto-encoder (VAE, Kingma and Welling [36]) using fully-connected layers and can be trained with CPU only. We provide more details in Appendix A.1.

Image Datasets. We consider three image sets to protect. The **In-distribution** set is a validation subset of CelebA whereon G was trained. This dataset helps understand the scenario where G can be maintained by the copyright owners. For further evaluations of HARVIM in scenarios where copyrighted images are not allowed to be used for training G, we consider two **out-of-distribution** sets: a validation subset of ImageNet [17], and 10 manually selected Cartoon images. Due to budget constraints, on CelebA and ImageNet we randomly choose 100 images respectively, see Appendix A.4 for more details.

Watermark Removal Methods. After constructing *hard-to-remove* watermark $\boldsymbol{m}$, we conduct two classes of watermark removal methods. The first *worst-case* class have access to the ground truth location of watermarks (i.e., exact $\mathbf{A}_m$ is assumed known) and remove them by solving inverse problems. To this end, **Flow-R** uses the same flow-based model G to solve the inpainting task with random initialized $\boldsymbol{x}$ [48], and **RePaint** is a representative diffusion model-based inpainting method [55]. The second *Blind-case* class contains **SLBR** [45] and **DeNet** [73], which are blind watermark removal models that are pretrained on diverse images and watermarks to *locate-and-remove* watermark in an end-to-end manner. Notably, HARVIM *is not optimized for any*

of these methods.

Evaluation Metrics. We evaluate the performance of HARVIM based on the reconstruction quality of $\tilde{\boldsymbol{x}}(\boldsymbol{m}_0)$ and $\tilde{\boldsymbol{x}}(\boldsymbol{m}_T)$, where $\boldsymbol{m}_0$ and $\boldsymbol{m}_T$ denotes the initial and learned watermarks respectively. Following the literature [45, 55], the reconstruction quality is measured by peak-signal-to-ratio (PSNR), structural similarity (SSIM), and learned perceptual image patch similarity (LPIPS) [83]. As will be detailed shortly, we manipulate the three metrics to make sure that higher indicates better reconstruction, and thus weaker copyright protection.

3.2 QUANTITATIVE EVALUATION OF HARVIM

When conducting watermark removal, We noted all of the four methods suffered from notable performance degradation in challenging scenarios. As an extreme case, SLBR and DeNet failed to recognize watermarks, and produced reconstruction nearly identical to the watermarked observation, as shown in Fig 3. Consequently, a direct comparison of PSNR and other metrics may fail to correctly measures the effectiveness of HARVIM: when a reconstruction $\tilde{\boldsymbol{x}}$ is identical to the observation $\boldsymbol{y}$, metric $\text{PSNR}(\tilde{\boldsymbol{x}}, \boldsymbol{x}_T) = \text{PSNR}(\boldsymbol{y}, \boldsymbol{x}_T)$ in essence quantifies *how much watermark* $\boldsymbol{m}$ *distorts the image*, other than *how difficult it is to be removed*.

To avoid this misleading evaluation, we check *to what extent a reconstructed image is better than the watermarked observation* by computing how much PSNR or SSIM from a reconstruction to the ground truth is higher than from the observation. Specifically, we defined $v_{\text{PSNR}}(\boldsymbol{x}) \triangleq \text{PSNR}(\tilde{\boldsymbol{x}}, \boldsymbol{x}_T) - \text{PSNR}(\boldsymbol{y}, \boldsymbol{x}_T)$ as a measure of how good reconstruction $\tilde{\boldsymbol{x}}$ is in terms of PSNR, the measures of SSIM and LPIPS are defined similarly[3]. We report these results (mean$\pm$se) in Tab 1. Due to page limit, we defer the results from SLBR that failed on our watermarks to App B. Original metrics are also reported in Tab 4 in App B for more comprehensive evaluation.

From Tab 1, HARVIM successfully learned watermarks resisting both flow- and diffusion-based worst-case methods, Flow-R and RePaint, in all cases. Blind-case methods failed to identify added watermarks, possibly due to the substantial style and semantic difference between our learned watermarks and their pre-trained data. This highlights the limitation of blind-case methods.

When comparing the defense performance against the two worst-case methods, HARVIM exhibited better performance on Flow-R than on RePaint. We hypothesize that this superior performance can be attributed to the fact that HARVIM and Flow-R share the same generative prior G and employ a similar *maximum-a-posteriori* Bayesian optimization framework. In contrast, images reconstructed by RePaint undergo a significantly different optimization process. Conceptually, this distinction is similar to attacking a *gray-box* model versus *black-box* model [61]. Furthermore, HARVIM shows strong transferability in both scenarios, which are challenging for traditional adversarial attacks [16]. We attribute this success to the fact that HARVIM's target, the *hard-to-reconstruct region* of image $\boldsymbol{x}_T$, is an intrinsic characteristic of the real $\boldsymbol{x}_T$. Consequently, any generative model pre-trained on real images will inherently reflect this property. As a result, HARVIM offers a general protection. In contrast, previous adversarial attack-based protections targeted on models-specific shortcuts that are not shared across different models, often resulting in unsatisfactory transferability [31].

As further evidence, although the generative prior G was trained on CelebA, HARVIM still offers comparable defense performance on out-of-distribution ImageNet and Cartoon datasets. As pointed in previous studies [2, 79], Flow as a generative prior provides a certain degree of generalizability across different image distributions for measuring the likelihood of an image. Our results further demonstrate that this flexibility can be leveraged to identify the *hard-to-reconstruct region* in out-of-distribution images as well.

3.3 QUALITATIVE EVALUATION OF HARVIM

We conclude this section by providing careful analysis on how HARVIM learns watermarks in order to make them hard-to-remove. We visualize reconstructions generated using different methods applied to random and HARVIM learned watermarks. Results are shown in Fig 3. Due to space constraints, one sample is presented for each watermark.

From Fig 3, HARVIM increased the difficulty of watermark removal while simultaneously preserving both image and watermark readability. To achieve this, it selected regions with abundant visual details as hard-to-reconstruct regions to place watermarks. Importantly, we found these details are likely overlooked even by human readers. For example, in three out of four CelebA images, HARVIM placed watermarks along the boundaries between human hair and the background. These placements caused both Flow-R and RePaint to fail in accurately reconstructing the textures. Similarly, watermarks were put on leafy backgrounds on ImageNet images, leading to further failures of the two models.

Interestingly, in column 8, both model failed to reconstruct the smaller lizard masked by the digit logo watermark. Furthermore, they both misinterpreted this lizard as part of the larger one. Given that Flow-R and RePaint employed generative priors of different architectures trained on distinct datasets, this "coincidence" can be considered as concrete evidence that the *hard-to-reconstruct region* is an intrinsic characteristic of the image, learned by different generative priors trained in diverse scenes. By targeting this intrinsic

[3] As lower LPIPS implies higher similarity, we flip its subtraction order to make larger v_{LPIPS} indicate better reconstruction.

Table 1: Worse-case performance of HARVIM when removers know the exact position of watermarks. The performance is evaluated based on to what extent the watermark removing performance is better than the observation in terms of PSNR, SSIM, and LPIPS respectively. Lower indicates worse reconstruction quality, thus stronger protection.

		CelebA								
		PSNR			SSIM × 100			LPIPS × 100		
		Random	HARVIM	Imp (↑)	Random	HARVIM	Imp (↑)	Random	HARVIM	Imp (↑)
DIG	Flow-R	$13.02_{\pm 0.37}$	$7.57_{\pm 0.66}$	5.44	$6.71_{\pm 0.24}$	$3.82_{\pm 0.40}$	2.89	$3.22_{\pm 0.27}$	$2.21_{\pm 0.24}$	1.01
	RePaint	$13.45_{\pm 0.41}$	$11.57_{\pm 0.46}$	1.89	$9.63_{\pm 0.25}$	$8.62_{\pm 0.33}$	1.01	$5.96_{\pm 0.27}$	$5.74_{\pm 0.21}$	0.22
NJ	Flow-R	$9.31_{\pm 0.34}$	$7.33_{\pm 0.36}$	1.98	$11.40_{\pm 0.58}$	$9.34_{\pm 0.61}$	2.06	$7.53_{\pm 0.39}$	$6.44_{\pm 0.44}$	1.08
	RePaint	$10.25_{\pm 0.34}$	$9.99_{\pm 0.38}$	0.26	$14.08_{\pm 0.60}$	$13.07_{\pm 0.67}$	1.01	$9.36_{\pm 0.34}$	$8.93_{\pm 0.41}$	0.43
OS	Flow-R	$9.57_{\pm 0.33}$	$7.21_{\pm 0.31}$	2.36	$11.43_{\pm 0.62}$	$8.01_{\pm 0.53}$	3.42	$7.90_{\pm 0.41}$	$7.02_{\pm 0.39}$	0.88
	RePaint	$10.39_{\pm 0.36}$	$9.44_{\pm 0.37}$	0.96	$15.68_{\pm 0.64}$	$12.86_{\pm 0.62}$	2.81	$10.62_{\pm 0.43}$	$10.09_{\pm 0.41}$	0.53
		ImageNet								
		PSNR			SSIM × 100			LPIPS × 100		
		Random	HARVIM	Imp (↑)	Random	HARVIM	Imp (↑)	Random	HARVIM	Imp (↑)
DIG	Flow-R	$9.61_{\pm 0.45}$	$6.09_{\pm 0.62}$	3.52	$4.49_{\pm 0.35}$	$2.72_{\pm 0.45}$	1.77	$3.22_{\pm 0.27}$	$2.21_{\pm 0.24}$	1.01
	RePaint	$9.61_{\pm 0.48}$	$8.56_{\pm 0.53}$	1.05	$8.80_{\pm 0.39}$	$8.40_{\pm 0.44}$	0.41	$5.96_{\pm 0.27}$	$5.74_{\pm 0.21}$	0.22
NJ	Flow-R	$8.46_{\pm 0.37}$	$7.21_{\pm 0.42}$	1.25	$5.12_{\pm 0.73}$	$3.42_{\pm 0.75}$	1.70	$7.53_{\pm 0.39}$	$6.44_{\pm 0.44}$	1.08
	RePaint	$7.20_{\pm 0.31}$	$6.69_{\pm 0.40}$	0.50	$3.76_{\pm 0.73}$	$2.79_{\pm 0.82}$	0.96	$9.36_{\pm 0.34}$	$8.93_{\pm 0.41}$	0.43
OS	Flow-R	$8.58_{\pm 0.34}$	$7.18_{\pm 0.39}$	1.40	$4.28_{\pm 0.69}$	$1.82_{\pm 0.69}$	2.46	$7.90_{\pm 0.41}$	$7.02_{\pm 0.39}$	0.88
	RePaint	$7.44_{\pm 0.33}$	$6.60_{\pm 0.38}$	0.84	$4.03_{\pm 0.68}$	$2.03_{\pm 0.77}$	2.00	$10.62_{\pm 0.43}$	$10.09_{\pm 0.41}$	0.53
		Cartoon								
		PSNR			SSIM × 100			LPIPS × 100		
		Random	HARVIM	Imp (↑)	Random	HARVIM	Imp (↑)	Random	HARVIM	Imp (↑)
DIG	Flow-R	$6.53_{\pm 1.18}$	$-0.86_{\pm 1.74}$	7.39	$2.55_{\pm 0.77}$	$-0.88_{\pm 1.25}$	3.42	$3.22_{\pm 0.27}$	$2.21_{\pm 0.24}$	1.01
	RePaint	$4.75_{\pm 1.36}$	$3.01_{\pm 1.26}$	1.74	$4.76_{\pm 1.04}$	$3.84_{\pm 1.23}$	0.92	$5.96_{\pm 0.27}$	$5.74_{\pm 0.21}$	0.22
NJ	Flow-R	$2.78_{\pm 1.27}$	$1.83_{\pm 1.41}$	0.95	$-0.59_{\pm 1.85}$	$-3.58_{\pm 2.08}$	2.98	$7.53_{\pm 0.39}$	$6.44_{\pm 0.44}$	1.08
	RePaint	$2.46_{\pm 1.14}$	$1.86_{\pm 1.08}$	0.60	$-2.95_{\pm 1.77}$	$-3.96_{\pm 2.33}$	1.01	$9.36_{\pm 0.34}$	$8.93_{\pm 0.41}$	0.43
OS	Flow-R	$2.97_{\pm 0.44}$	$1.52_{\pm 0.44}$	1.45	$-1.20_{\pm 1.72}$	$-5.25_{\pm 1.76}$	4.05	$7.90_{\pm 0.41}$	$7.02_{\pm 0.39}$	0.88
	RePaint	$3.52_{\pm 0.36}$	$1.82_{\pm 0.25}$	1.70	$-1.21_{\pm 2.14}$	$-5.03_{\pm 1.70}$	3.82	$10.62_{\pm 0.43}$	$10.09_{\pm 0.41}$	0.53

characteristic, HARVIM shows strong transferability.

4 RELATED WORKS

Copyright Protection. DGM-based AI tools have raised concerns about unauthorized use of copyrighted images, including style transfer [35], personalization [24, 66], and image editing [10, 69]. Recent studies framed data protection as a problem of *adversarial attacks*. By introducing imperceptible perturbations to protected images, these approaches aim to degrade AI performance on the affected data [67, 45, 74]. Particularly, PhotoGuard [67] attacked a text-to-image model by perturbing its latent code, aligning generations with an unrelated dummy image. Glaze [68] further employed a style-transfer model to minimize the similarity of generated images to the protected content. AdvDM [44] targeted on diffusion-based models by minimizing the likelihood of perturbed images; and [43] added a texture-targeting loss for improved robustness. To defend against personalized DreamBooth [66], [74] learned perturbations to degrade its training performance using a bi-level optimization framework, with an approximate solution proposed by neglecting the trajectories in the lower-level optimization. [52] improved this process by using meta-learning to attack an ensemble of models. However, existing methods specifically targeted DGMs that cause the misuse, and their attack-based solutions are highly specialized, making generalization challenging [16]. In contrast, our HARVIM identifies a hard-to-reconstruct region of the image and places a visible watermark, rendering the image unusable. In this way, HARVIM provides protection agnostic to misuse scenarios.

Visible Watermarking and Removal. Visible watermarks have been widely used to prevent piracy [7, 13, 57, 32]. Early works resorted to signal processing technique to enhance the robustness of watermark [62, 32, 29]. In response, watermark removal has also accumulated a vast literature [15, 9, 41]. When the watermark location is known, inverse-problem-based solvers can provide strong reconstructions [34], as also verified in our experiments. However, these methods are ineffective when location information is unavailable, and obtaining human labeling is often impractical [49]. The advance of deep learning further stimulated the end-to-end blind watermark removal models. Early works used image translation methods to generate clean images from watermarked observations in a single step [8, 42]. Later, [14, 45, 49] separated the processes of locating and removing watermarks into two distinct steps, achieving more effective results. These methods have posed remarkable performance on removing visible watermarking [15]. However,

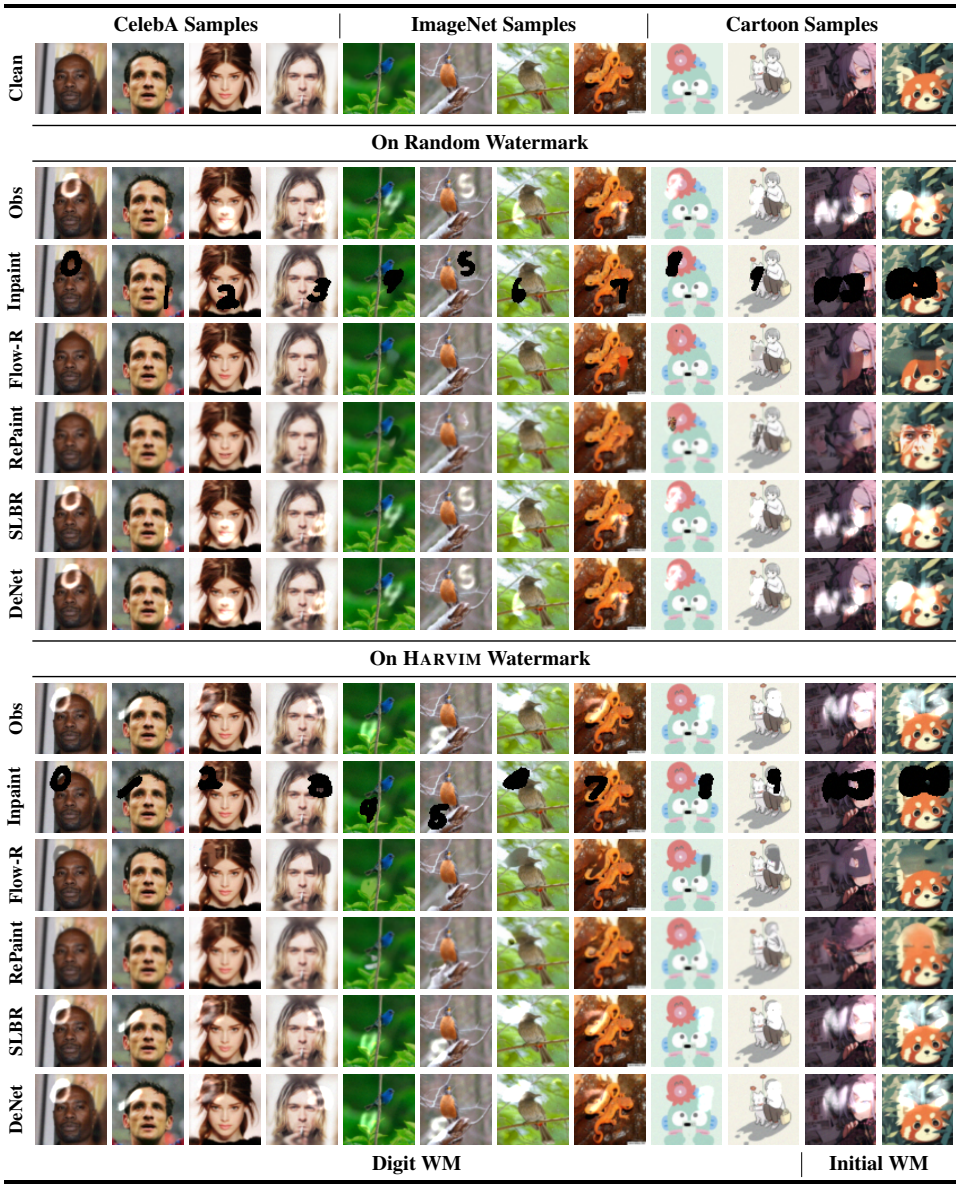

Figure 3: Watermark removal performance of *worse-case* Flow-R and RePaint, and *blind-case* SLBR and DeNet. "Obs"and "Inpaint" show watermarked and surrogate inpainting images respectively.

the primary focus of watermarking in the AI era has shifted to attack-based protection and invisible watermarking on AI-generated contents, leaving robust visible watermarking unsolved. In this work, we proposed a new learning-based visible watermarking and experimented with both inverse problem solver and two-stage blind watermark removal methods. Empirically, HARVIM learns stronger watermarks to defeat all these methods.

Watermarking in AI Era. The advance of vision and language foundation models [63, 18, 77, 65, 6, 50, 87] have raised ethical concerns about the potential misuse of AI-generated content (AIGC), such as deepfake [78], plagiarism [37, 38], and others [27]. To address these challenges, invisible watermarking have been proposed for embedding in the output data [85, 46, 33]. These watermarks do not affect normal use of AIGC, but in case of misuse such as fake news, they can be extracted to trace the source of the generated content. For example, in watermarking vision models, an encoder and decoder are trained to generate and extract watermarks, and the vision model is often fine-tuned jointly to avoid performance degradation [82, 21, 56, 1]. For language models, the generation process is altered by increasing the likelihood of certain words while decreasing that of others. This creates a traceable pattern in the generated texts [37, 46]. Notably, these watermarking techniques have objectives orthogonal to ours. They aim to ensure the traceability of AIGC, while ours targets to protect copy-

righted contents created by human artists.

5 CONCLUSION

In this paper, we introduce HARVIM, a new copyright protection paradigm in the AI era. Unlike existing attack-based approaches, HARVIM requires no domain knowledge of misuse scenarios or mechanisms. HARVIM bridges inverse problems and copyright protection and formulates learning-based *hard-to-remove* visible watermarking as a bi-level optimization problem. Built upon recent optimality guarantees for inverse problems, we propose a new meta-learning solution for HARVIM. We validate the effectiveness of our algorithm across watermarking scenarios. Encouraged by the promising results, we identify two exciting directions for future work. First, we will explore a training-free version of Harvim that embeds open-ended text watermarks, leveraging recent advances in image editing to enhance real-time efficiency. Second, we will investigate the theoretical guarantees of the Harvim framework from two perspectives: (1) determining the minimum distortion (masking) required to ensure a watermark is provably unremovable, and (2) characterizing the maximum tolerable noise or perturbation under which a personalized concept can still be provably learned using DreamBooth.

ACKNOWLEDGMENT

This material is based upon work supported by the U.S. Department of Energy, Office of Science Energy Earthshot Initiative as part of the project "Learning reduced models under extreme data conditions for design and rapid decision-making in complex systems" under Award #DE-SC0024721.

References

[1] Bang An, Mucong Ding, Tahseen Rabbani, Aakriti Agrawal, Yuancheng Xu, Chenghao Deng, Sicheng Zhu, Abdirisak Mohamed, Yuxin Wen, Tom Goldstein, et al. Benchmarking The Robustness Of Image Watermarks. *arXiv preprint arXiv:2401.08573*, 2024.

[2] Muhammad Asim, Max Daniels, Oscar Leong, Ali Ahmed, and Paul Hand. Invertible generative models for inverse problems: mitigating representation error and dataset bias. In *International Conference on Machine Learning*, pages 399–409, 2020.

[3] Anish Athalye, Logan Engstrom, Andrew Ilyas, and Kevin Kwok. Synthesizing Robust Adversarial Examples. In *International Conference On Machine Learning*, pages 284–293, 2018.

[4] Ayse Elvan Aydemir, Alptekin Temizel, and Tugba Taskaya Temizel. The Effects Of JPEG And JPEG2000 Compression On Attacks Using Adversarial Examples. *arXiv preprint arXiv:1803.10418*, 2018.

[5] Ashish Bora, Ajil Jalal, Eric Price, and Alexandros G Dimakis. Compressed Sensing Using Generative Models. In *International Conference on Machine Learning*, pages 537–546, 2017.

[6] Florian Bordes, Richard Yuanzhe Pang, Anurag Ajay, Alexander C Li, Adrien Bardes, Suzanne Petryk, Oscar Mañas, Zhiqiu Lin, Anas Mahmoud, Bargav Jayaraman, et al. An Introduction to Vision-language Modeling. *arXiv preprint arXiv:2405.17247*, 2024.

[7] Gordon W. Braudaway, Karen A. Magerlein, and Frederick Cole Mintzer. Protecting Publicly Available Images with a Visible Image Watermark. In *SPIE Proceedings*, volume 2659, pages 126–133, 1996.

[8] Zhiyi Cao, Shaozhang Niu, Jiwei Zhang, and Xinyi Wang. Generative Adversarial Networks Model For Visible Watermark Removal. *IET Image Processing*, 13(10):1783–1789, 2019.

[9] Danni Cheng, Xiang Li, Wei-Hong Li, Chan Lu, Fake Li, Hua Zhao, and Wei-Shi Zheng. Large-scale Visible Watermark Detection and Removal with Deep Convolutional Networks. In *Pattern Recognition and Computer Vision: First Chinese Conference*, pages 27–40, 2018.

[10] Jooyoung Choi, Yunjey Choi, Yunji Kim, Junho Kim, and Sungroh Yoon. Custom-Edit: Text-Guided Image Editing With Customized Diffusion Models. *arXiv preprint arXiv:2305.15779*, 2023.

[11] Sheng-Yen Chou, Pin-Yu Chen, and Tsung-Yi Ho. How To Backdoor Diffusion Models? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4015–4024, 2023.

[12] Gregory Cohen, Saeed Afshar, Jonathan Tapson, and Andre Van Schaik. EMNIST: Extending MNIST To Handwritten Letters. In *2017 International Joint Conference On Neural Networks (IJCNN)*, pages 2921–2926, 2017.

[13] Ingemar J Cox, Joe Kilian, F Thomson Leighton, and Talal Shamoon. Secure Spread Spectrum Watermarking for Multimedia. *IEEE transactions on image processing*, 6(12):1673–1687, 1997.

[14] Xiaodong Cun and Chi-Man Pun. Split Then Refine: Stacked Attention-Guided Resunets For Blind Single Image Visible Watermark Removal. In *Proceedings of the AAAI Conference On Artificial Intelligence*, volume 35, pages 1184–1192, 2021.

[15] Tali Dekel, Michael Rubinstein, Ce Liu, and William T Freeman. On The Effectiveness Of Visible Watermarks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2146–2154, 2017.

[16] Ambra Demontis, Marco Melis, Maura Pintor, Matthew Jagielski, Battista Biggio, Alina Oprea, Cristina Nita-Rotaru, and Fabio Roli. Why Do Adversarial Attacks Transfer? Explaining Transferability Of Evasion And Poisoning Attacks. In *28th USENIX Security Symposium (USENIX Security 19)*, pages 321–338, 2019.

[17] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A Large-Scale Hierarchical Image Database. In *2009 IEEE Conference On Computer Vision And Pattern Recognition*, pages 248–255, 2009.

[18] Prafulla Dhariwal and Alexander Nichol. Diffusion Models Beat Gans on Image Synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.

[19] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density Estimation Using Real NVP. *arXiv preprint arXiv:1605.08803*, 2016.

[20] Gintare Karolina Dziugaite, Zoubin Ghahramani, and Daniel M Roy. A Study Of The Effect Of JPG Compression On Adversarial Images. *arXiv preprint arXiv:1608.00853*, 2016.

[21] Pierre Fernandez, Guillaume Couairon, Hervé Jégou, Matthijs Douze, and Teddy Furon. The Stable Signature: Rooting Watermarks In Latent Diffusion Models. In *Proceedings of the IEEE/CVF International Conference On Computer Vision*, pages 22466–22477, 2023.

[22] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-Agnostic Meta-Learning For Fast Adaptation Of Deep Networks. In *International Conference On Machine Learning*, pages 1126–1135, 2017.

[23] Luca Franceschi, Paolo Frasconi, Saverio Salzo, Riccardo Grazzi, and Massimiliano Pontil. Bilevel Programming for Hyperparameter Optimization and Meta-learning. In *International conference on machine learning*, pages 1568–1577, 2018.

[24] Rinon Gal, Yuval Alaluf, Yuval Atzmon, Or Patashnik, Amit H Bermano, Gal Chechik, and Daniel Cohen-Or. An Image Is Worth One Word: Personalizing Text-To-Image Generation Using Textual Inversion. *arXiv preprint arXiv:2208.01618*, 2022.

[25] Jonas Geiping, Liam Fowl, W Ronny Huang, Wojciech Czaja, Gavin Taylor, Michael Moeller, and Tom Goldstein. Witches' Brew: Industrial Scale Data Poisoning Via Gradient Matching. *arXiv preprint arXiv:2009.02276*, 2020.

[26] Saeed Ghadimi and Mengdi Wang. Approximation Methods for Bilevel Programming. *arXiv preprint arXiv:1802.02246*, 2018.

[27] Danhuai Guo, Huixuan Chen, Ruoling Wu, and Yangang Wang. AIGC Challenges And Opportunities Related To Public Safety: A Case Study Of ChatGPT. *Journal Of Safety Science And Resilience*, 4(4):329–339, 2023.

[28] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising Diffusion Probabilistic Models. *Advances In Neural Information Processing Systems*, 33:6840–6851, 2020.

[29] Yongjian Hu and Sam Kwong. Wavelet Domain Adaptive Visible Watermarking. *Electronics Letters*, 37(20): 1, 2001.

[30] Chun-Hsiang Huang and Ja-Ling Wu. Attacking Visible Watermarking Schemes. *IEEE Transactions On Multimedia*, 6(1):16–30, 2004.

[31] W Ronny Huang, Jonas Geiping, Liam Fowl, Gavin Taylor, and Tom Goldstein. Metapoison: Practical General-Purpose Clean-Label Data Poisoning. *Advances In Neural Information Processing Systems*, 33: 12080–12091, 2020.

[32] Mohan S Kankanhalli, KR Ramakrishnan, et al. Adaptive Visible Watermarking Of Images. In *Proceedings IEEE International Conference On Multimedia Computing And Systems*, volume 1, pages 568–573. IEEE, 1999.

[33] Bishwa Karki, Chun-Hua Tsai, Pei-Chi Huang, and Xin Zhong. Deep Learning-based Text-in-Image Watermarking. *arXiv preprint arXiv:2404.13134*, 2024.

[34] David Khachaturov, Ilia Shumailov, Yiren Zhao, Nicolas Papernot, and Ross Anderson. Markpainting: Adversarial Machine Learning Meets Inpainting. In *International Conference On Machine Learning*, pages 5409–5419, 2021.

[35] Gwanghyun Kim, Taesung Kwon, and Jong Chul Ye. DiffusionCLIP: Text-Guided Diffusion Models For Robust Image Manipulation. In *Proceedings of the IEEE/CVF Conference On Computer Vision And Pattern Recognition*, pages 2426–2435, 2022.

[36] Diederik P Kingma and Max Welling. Auto-Encoding Variational Bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[37] John Kirchenbauer, Jonas Geiping, Yuxin Wen, Jonathan Katz, Ian Miers, and Tom Goldstein. A Watermark For Large Language Models. In *International Conference On Machine Learning*, pages 17061–17084, 2023.

[38] Gregory Kang Ruey Lau, Xinyuan Niu, Hieu Dao, Jiangwei Chen, Chuan-Sheng Foo, and Bryan Kian Hsiang Low. Protecting Text IP In The Era Of LLMs With Robust And Scalable Watermarking. In *ICML Workshop On Generative AI+ Law*, 2024.

[39] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-Based Learning Applied To Document Recognition. *Proceedings of the IEEE*, 86(11): 2278–2324, 1998.

[40] Qi Lei, Ajil Jalal, Inderjit S Dhillon, and Alexandros G Dimakis. Inverting Deep Generative models, One layer at a time. *Advances in neural information processing systems*, 32, 2019.

[41] Yicheng Leng, Chaowei Fang, Gen Li, Yixiang Fang, and Guanbin Li. Removing Interference and Recovering Content Imaginatively for Visible Watermark Removal. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 2983–2990, 2024.

[42] Xiang Li, Chan Lu, Danni Cheng, Wei-Hong Li, Mei Cao, Bo Liu, Jiechao Ma, and Wei-Shi Zheng. Towards Photo-Realistic Visible Watermark Removal With Conditional Generative Adversarial Networks. In *Image And Graphics: 10th International Conference*, pages 345–356, 2019.

[43] Chumeng Liang and Xiaoyu Wu. Mist: Towards Improved Adversarial Examples For Diffusion Models. *arXiv preprint arXiv:2305.12683*, 2023.

[44] Chumeng Liang, Xiaoyu Wu, Yang Hua, Jiaru Zhang, Yiming Xue, Tao Song, Zhengui Xue, Ruhui Ma, and Haibing Guan. Adversarial Example Does Good: Preventing Painting Imitation From Diffusion Models Via Adversarial Examples. *arXiv preprint arXiv:2302.04578*, 2023.

[45] Jing Liang, Li Niu, Fengjun Guo, Teng Long, and Liqing Zhang. Visible Watermark Removal Via Self-Calibrated Localization And Background Refinement. In *Proceedings of the 29th ACM International Conference On Multimedia*, pages 4426–4434, 2021.

[46] Aiwei Liu, Leyi Pan, Yijian Lu, Jingjing Li, Xuming Hu, Xi Zhang, Lijie Wen, Irwin King, Hui Xiong, and Philip Yu. A Survey Of Text Watermarking In The Era Of Large Language Models. *ACM Computing Surveys*, 2024.

[47] Tianci Liu and Jeffrey Regier. An empirical comparison of gans and normalizing flows for density estimation. *arXiv preprint arXiv:2006.10175*, 2020.

[48] Tianci Liu, Tong Yang, Quan Zhang, and Qi Lei. Optimization For Amortized Inverse Problems. In *International Conference On Machine Learning*, pages 22289–22319, 2023.

[49] Xinwei Liu, Jian Liu, Yang Bai, Jindong Gu, Tao Chen, Xiaojun Jia, and Xiaochun Cao. Watermark Vaccine: Adversarial Attacks To Prevent Watermark Removal. In *European Conference On Computer Vision*, pages 1–17, 2022.

[50] Xu Liu, Tong Zhou, Chong Wang, Yuping Wang, Yuanxin Wang, Qinjingwen Cao, Weizhi Du, Yonghuan Yang, Junjun He, Yu Qiao, et al. Toward the Unification of Generative and Discriminative Visual Foundation model: A Survey. *The Visual Computer*, pages 1–42, 2024.

[51] Yang Liu, Zhen Zhu, and Xiang Bai. WDNet: Watermark-Decomposition Network For Visible Watermark Removal. In *Proceedings of the IEEE/CVF Winter Conference On Applications Of Computer Vision*, pages 3685–3693, 2021.

[52] Yixin Liu, Chenrui Fan, Yutong Dai, Xun Chen, Pan Zhou, and Lichao Sun. MetaCloak: Preventing Unauthorized Subject-Driven Text-To-Image Diffusion-Based Synthesis Via Meta-Learning. In *Proceedings of the IEEE/CVF Conference On Computer Vision And Pattern Recognition*, pages 24219–24228, 2024.

[53] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep Learning Face Attributes in the Wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, 2015.

[54] Ilya Loshchilov and Frank Hutter. Decoupled Weight Decay Regularization. In *International Conference on Learning Representations*, 2019.

[55] Andreas Lugmayr, Martin Danelljan, Andres Romero, Fisher Yu, Radu Timofte, and Luc Van Gool. Repaint: Inpainting Using Denoising Diffusion Probabilistic Models. In *Proceedings of the IEEE/CVF Conference On Computer Vision And Pattern Recognition*, pages 11461–11471, 2022.

[56] Hannes Mareen, Lucas Antchougov, Glenn Van Wallendael, and Peter Lambert. Blind Deep-Learning-Based Image Watermarking Robust Against Geometric Transformations. In *2024 IEEE International Conference On Consumer Electronics (ICCE)*, pages 1–2. IEEE, 2024.

[57] Saraju P Mohanty, KR Ramakrishnan, and Mohan Kankanhalli. A Dual Watermarking Technique for Images. In *Proceedings of the seventh ACM international conference on Multimedia (Part 2)*, pages 49–51, 1999.

[58] Gregory Ongie, Ajil Jalal, Christopher A Metzler, Richard G Baraniuk, Alexandros G Dimakis, and Rebecca Willett. Deep Learning Techniques for Inverse Problems in Imaging. *IEEE Journal on Selected Areas in Information Theory*, 1(1):39–56, 2020.

[59] Zhuoshi Pan, Yuguang Yao, Gaowen Liu, Bingquan Shen, H Vicky Zhao, Ramana Rao Kompella, and Sijia Liu. From Trojan Horses To Castle Walls: Unveiling Bilateral Backdoor Effects In Diffusion Models. *arXiv preprint arXiv:2311.02373*, 2023.

[60] George Papamakarios, Eric Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan. Normalizing Flows for Probabilistic Modeling and Inference. *Journal of Machine Learning Research*, 22(57):1–64, 2021.

[61] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z Berkay Celik, and Ananthram Swami. Practical Black-Box Attacks Against Machine Learning. In *Proceedings of the 2017 ACM On Asia Conference On Computer And Communications Security*, pages 506–519, 2017.

[62] Christine I Podilchuk and Wenjun Zeng. Image-adaptive Watermarking Using Visual Models. *IEEE Journal on selected areas in communications*, 16(4): 525–539, 1998.

[63] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language Models are Unsupervised Multitask Learners. *OpenAI blog*, 1 (8):9, 2019.

[64] Ambrish Rawat, Killian Levacher, and Mathieu Sinn. The Devil is in the GAN: Backdoor Attacks and Defenses in Deep Generative Models. In *European Symposium on Research in Computer Security*, pages 776–783. Springer, 2022.

[65] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-Resolution Image Synthesis With Latent Diffusion Models. In *Proceedings of the IEEE/CVF Conference On Computer Vision And Pattern Recognition*, pages 10684–10695, 2022.

[66] Nataniel Ruiz, Yuanzhen Li, Varun Jampani, Yael Pritch, Michael Rubinstein, and Kfir Aberman. DreamBooth: Fine Tuning Text-To-Image Diffusion Models For Subject-Driven Generation. In *Proceedings of the IEEE/CVF Conference On Computer Vision And Pattern Recognition*, pages 22500–22510, 2023.

[67] Hadi Salman, Alaa Khaddaj, Guillaume Leclerc, Andrew Ilyas, and Aleksander Madry. Raising The Cost Of Malicious AI-Powered Image Editing. *arXiv preprint arXiv:2302.06588*, 2023.

[68] Shawn Shan, Jenna Cryan, Emily Wenger, Haitao Zheng, Rana Hanocka, and Ben Y Zhao. Glaze: Protecting Artists From Style Mimicry By {Text-To-Image} Models. In *32nd USENIX Security Symposium (USENIX Security 23)*, pages 2187–2204, 2023.

[69] Yujun Shi, Chuhui Xue, Jun Hao Liew, Jiachun Pan, Hanshu Yan, Wenqing Zhang, Vincent YF Tan, and Song Bai. DragDiffusion: Harnessing Diffusion Models For Interactive Point-Based Image Editing. In *Proceedings of the IEEE/CVF Conference On Computer Vision And Pattern Recognition*, pages 8839–8849, 2024.

[70] Ankur Sinha, Pekka Malo, and Kalyanmoy Deb. A Review On Bilevel Optimization: From Classical To Evolutionary Approaches And Applications. *IEEE Transactions On Evolutionary Computation*, 22(2): 276–295, 2017.

[71] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep Unsupervised Learning Using Nonequilibrium Thermodynamics. In *International Conference On Machine Learning*, pages 2256–2265, 2015.

[72] Kihyuk Sohn, Honglak Lee, and Xinchen Yan. Learning Structured Output Representation Using Deep Conditional Generative Models. *Advances In Neural Information Processing Systems*, 28, 2015.

[73] Ruizhou Sun, Yukun Su, and Qingyao Wu. DENet: Disentangled Embedding Network For Visible Watermark Removal. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 2411–2419, 2023.

[74] Thanh Van Le, Hao Phung, Thuan Hoang Nguyen, Quan Dao, Ngoc N Tran, and Anh Tran. Anti-DreamBooth: Protecting Users From Personalized Text-To-Image Synthesis. In *Proceedings of the IEEE/CVF International Conference On Computer Vision*, pages 2116–2127, 2023.

[75] Patrick von Platen, Suraj Patil, Anton Lozhkov, Pedro Cuenca, Nathan Lambert, Kashif Rasul, Mishig Davaadorj, Dhruv Nair, Sayak Paul, William Berman, Yiyi Xu, Steven Liu, and Thomas Wolf. Diffusers: State-Of-The-Art Diffusion Models. https://github.com/huggingface/diffusers, 2022.

[76] George Voyatzis and Ioannis Pitas. The Use Of Watermarks In The Protection Of Digital Multimedia

Products. *Proceedings of the IEEE*, 87(7):1197–1207, 1999.

[77] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

[78] Mika Westerlund. The Emergence Of Deepfake Technology: A Review. *Technology Innovation Management Review*, 9(11), 2019.

[79] Jay Whang, Qi Lei, and Alex Dimakis. Solving Inverse Problems with a Flow-based Noise Model. In *Proceedings of the 38th International Conference on Machine Learning*, pages 11146–11157, 2021.

[80] Haotian Xue and Yongxin Chen. Rethinking Adversarial Attacks as Protection Against Diffusion-based Mimicry. In *Neurips Safe Generative AI Workshop 2024*, 2024.

[81] Ling Yang, Zhilong Zhang, Yang Song, Shenda Hong, Runsheng Xu, Yue Zhao, Wentao Zhang, Bin Cui, and Ming-Hsuan Yang. Diffusion Models: A Comprehensive Survey Of Methods And Applications. *ACM Computing Surveys*, 56(4):1–39, 2023.

[82] Ning Yu, Vladislav Skripniuk, Dingfan Chen, Larry Davis, and Mario Fritz. Responsible Disclosure Of Generative Models Using Scalable Fingerprinting. *arXiv preprint arXiv:2012.08726*, 2020.

[83] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The Unreasonable Effectiveness Of Deep Features As A Perceptual Metric. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 586–595, 2018.

[84] Chenchen Zhao and Hao Li. Blurring Fools The Network–Adversarial Attacks By Feature Peak Suppression And Gaussian Blurring. *arXiv preprint arXiv:2012.11442*, 2020.

[85] Yunqing Zhao, Tianyu Pang, Chao Du, Xiao Yang, Ngai-Man Cheung, and Min Lin. A Recipe for Watermarking Diffusion Models. *arXiv preprint arXiv:2303.10137*, 2023.

[86] Zhengyue Zhao, Jinhao Duan, Kaidi Xu, Chenan Wang, Rui Zhang, Zidong Du, Qi Guo, and Xing Hu. Can Protective Perturbation Safeguard Personal Data From Being Exploited By Stable Diffusion? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 24398–24407, 2024.

[87] Zheng Zhu, Xiaofeng Wang, Wangbo Zhao, Chen Min, Nianchen Deng, Min Dou, Yuqi Wang, Botian Shi, Kai Wang, Chi Zhang, et al. Is Sora a World Simulator? A Comprehensive Survey on General World Models and Beyond. *arXiv preprint arXiv:2405.03520*, 2024.

A IMPLEMENTATION DETAILS

In this section we provide more technical details about our implementation of watermark generators. Hyper-parameters and computing infrastructure information is also provided.

A.1 WATERMARK GENERATOR

In this work, we use MNIST digits [39] and MNIST letters [12] to create watermarks. Generated watermarks are grayscaled of size 64×64. Watermark generators are parameterized by a three-hidden-layer MLP conditional VAEs following Sohn et al. [72].

Controllable Location. To make watermark location controllable, when training the watermark generator, we put digits and letters in various locations in the image, by assigning different padding sizes to four sides. Based on this, we calculated the left and bottom padding ratios lying in $[0, 1]$. To be more specific, the watermark is placed at the *leftmost* region when the left padding ratio takes 0, and at the *rightmost* region when it takes 1. Bottom padding ratio functions in a similar way. In the training, the two padding ratios, together with the digit index (0-9), are used as conditions.

Watermark Optimization. In Alg 1, HARVIM uses the fixed pre-trained CVAE and optimizes its latent code and two padding ratios to learn watermarks. This parameterization allows us to maintain a good **watermark readability**. To further satisfies the image readability, we define $\mathcal{R}(\boldsymbol{m}) = \|\boldsymbol{m}\|_1$ to avoid learning an excessive watermark.

Differentiable Approximation for Mask Matrix. To obtain a differentiable approximation for masking matrix in inpainting, we define

$$\mathbf{A}_m = \text{diag}\left(\text{sig}\left(\frac{m_1 - \alpha}{\beta}\right), \ldots, \text{sig}\left(\frac{m_n - \alpha}{\beta}\right)\right),$$

and use $\alpha = 0.15, \beta = 0.01$. See inpainting observations in Figure 3 generated with this differentiable approximation.

Initialization Due to the highly non-smooth nature in location parameters, in HARVIM, we conduct a 3-by-3 grid search on watermark locations before running the complete algorithm. This search is based on the reconstruction quality from a 50-step MLE solution, as detailed in [48].

A.2 HYPERPARAMETERS

We summarize all hyperparameters used in this paper in Table 2, which we found working well. In execution, we rescale $\mathcal{R}(\boldsymbol{m})$ terms before tuning its weight to avoid bearing with the magnitude difference. Here we also provide a few clarification on some choices.

First, the "coefficient of watermark regularizer" in Eq (4) (i.e., norm of the watermark), was tuned by monitoring the watermark size. We note that a strong regularization will make the watermark disappear, and a weak one will let the watermark cover the whole image. We chose 0.001 as it allowed watermark size to remain stable, i.e., close to the initial size during the optimization process. Note that it was not tuned based on watermark removal performance.

Second, the two "smoothing factors" α, β, as discussed above, were not treated as tunable hyperparameter to benefit HARVIM performance. Instead, we chose the two solely to make the sigmoid function have a fairly wide range $[0, 1]$ when its input (pixel) lies in $[0, 1]$.

A.3 COMPUTING INFRASTRUCTURE

Our experiments were conducted on a NVIDIA A6000 48GB GPU. Our watermark generators were trained on a CPU-only machine.

A.4 TESTING DATASETS

We tested HARVIM on 100 validation samples from CelebA and ImageNet datasets respectively. More specifically, to guarantee reproducibility, we used the first 100 CelebA validation samples from [79]; and the first 100 ImageNet samples from public subset on https://github.com/EliSchwartz/imagenet-sample-images.

Table 2: Hyper-parameters of different methods.

		Digit Logo Watermarks	Initial Watermarks
	HParam	Value	Value
HARVIM	Learning Rate	0.05	0.05
	Optimizer	Default AdamW [54]	Default AdamW [54]
	Coeff. of $\mathcal{R}(\boldsymbol{m})$ (Eq (4))	0.001	0.001
	α, β (Eq (4))	$\alpha = 0.15, \beta = 0.01$	$\alpha = 0.15, \beta = 0.01$
	Meta Step K	1	1
	Targeted λ	1	1
	Step Size for λ	Using dynamic strategies from Liu et al. [48].	
Flow-R	λ	1	1
	Others	Identical to Liu et al. [48]	
Repaint	Batch Size	10	10
	Others	Identical to Lugmayr et al. [55]	
SLBR	All	Identical to Liang et al. [45]	
DeNet	All	Identical to Sun et al. [73]	

B MORE EXPERIMENT RESULTS

In this section we provide more experimental results. In particular, Table 3 reports the complete version of Table 1. SLBR failed to recognize the two types of watermark, as indicated by its low "Before" and "After" values: these values, as detailed in Sec 3, refer to how much reconstruction $\tilde{\boldsymbol{x}}$ is better than observations $\boldsymbol{y}$, i.e., $\text{PSNR}(\tilde{\boldsymbol{x}}, \boldsymbol{x}_T) - \text{PSNR}(\boldsymbol{y}, \boldsymbol{x}_T)$. See Figure 3 for concrete examples of SLBR failures.

Table 3: (Complete version of Table 1) Performance of different watermark removal methods on random and HARVIM watermarks. The performance is evaluated based on how the reconstruction is better than the observation in terms of PSNR, SSIM, and LPIPS respectively.

CelebA

		PSNR			SSIM × 100			LPIPS × 100		
		Random	**HARVIM**	**Imp (↑)**	**Random**	**HARVIM**	**Imp (↑)**	**Random**	**HARVIM**	**Imp (↑)**
DIG	**Flow-R**	$13.02_{\pm 0.37}$	$7.57_{\pm 0.66}$	5.44	$6.71_{\pm 0.24}$	$3.82_{\pm 0.40}$	2.89	$3.22_{\pm 0.27}$	$2.21_{\pm 0.24}$	1.01
	RePaint	$13.45_{\pm 0.41}$	$11.57_{\pm 0.46}$	1.89	$9.63_{\pm 0.25}$	$8.62_{\pm 0.33}$	1.01	$5.96_{\pm 0.27}$	$5.74_{\pm 0.21}$	0.22
	SLBR	$0.10_{\pm 0.01}$	$0.11_{\pm 0.02}$	-0.01	$0.81_{\pm 0.02}$	$0.78_{\pm 0.02}$	0.03	$0.12_{\pm 0.03}$	$0.12_{\pm 0.05}$	0.00
	DeNet	$0.10_{\pm 0.01}$	$0.12_{\pm 0.02}$	-0.02	$0.83_{\pm 0.02}$	$0.82_{\pm 0.02}$	0.01	$0.12_{\pm 0.03}$	$0.18_{\pm 0.04}$	-0.06
NJ	**Flow-R**	$9.31_{\pm 0.34}$	$7.33_{\pm 0.36}$	1.98	$11.40_{\pm 0.58}$	$9.34_{\pm 0.61}$	2.06	$7.53_{\pm 0.39}$	$6.44_{\pm 0.44}$	1.08
	RePaint	$10.25_{\pm 0.34}$	$9.99_{\pm 0.38}$	0.26	$14.08_{\pm 0.60}$	$13.07_{\pm 0.67}$	1.01	$9.36_{\pm 0.34}$	$8.93_{\pm 0.41}$	0.43
	SLBR	$0.00_{\pm 0.00}$	$0.01_{\pm 0.00}$	0.00	$0.32_{\pm 0.02}$	$0.35_{\pm 0.02}$	-0.03	$-0.40_{\pm 0.04}$	$-0.36_{\pm 0.04}$	-0.03
	DeNet	$0.00_{\pm 0.00}$	$0.01_{\pm 0.00}$	0.00	$0.36_{\pm 0.02}$	$0.38_{\pm 0.02}$	-0.02	$-0.37_{\pm 0.04}$	$-0.36_{\pm 0.04}$	-0.01
OS	**Flow-R**	$9.57_{\pm 0.33}$	$7.21_{\pm 0.31}$	2.36	$11.43_{\pm 0.62}$	$8.01_{\pm 0.53}$	3.42	$7.90_{\pm 0.41}$	$7.02_{\pm 0.39}$	0.88
	RePaint	$10.39_{\pm 0.36}$	$9.44_{\pm 0.37}$	0.96	$15.68_{\pm 0.64}$	$12.86_{\pm 0.62}$	2.81	$10.62_{\pm 0.43}$	$10.09_{\pm 0.41}$	0.53
	SLBR	$-0.01_{\pm 0.00}$	$0.00_{\pm 0.00}$	0.00	$0.20_{\pm 0.02}$	$0.25_{\pm 0.02}$	-0.05	$-0.46_{\pm 0.05}$	$-0.38_{\pm 0.05}$	-0.09
	DeNet	$0.00_{\pm 0.00}$	$0.00_{\pm 0.00}$	0.00	$0.26_{\pm 0.02}$	$0.32_{\pm 0.02}$	-0.06	$-0.40_{\pm 0.05}$	$-0.33_{\pm 0.05}$	-0.07

ImageNet

		PSNR			SSIM × 100			LPIPS × 100		
		Random	**HARVIM**	**Imp (↑)**	**Random**	**HARVIM**	**Imp (↑)**	**Random**	**HARVIM**	**Imp (↑)**
DIG	**Flow-R**	$9.61_{\pm 0.45}$	$6.09_{\pm 0.62}$	3.52	$4.49_{\pm 0.35}$	$2.72_{\pm 0.45}$	1.77	$3.22_{\pm 0.27}$	$2.21_{\pm 0.24}$	1.01
	RePaint	$9.61_{\pm 0.48}$	$8.56_{\pm 0.53}$	1.05	$8.80_{\pm 0.39}$	$8.40_{\pm 0.44}$	0.41	$5.96_{\pm 0.27}$	$5.74_{\pm 0.21}$	0.22
	SLBR	$0.13_{\pm 0.01}$	$0.12_{\pm 0.02}$	0.01	$1.05_{\pm 0.04}$	$1.01_{\pm 0.05}$	0.04	$-0.36_{\pm 0.07}$	$-0.14_{\pm 0.07}$	-0.22
	DeNet	$0.13_{\pm 0.01}$	$0.14_{\pm 0.01}$	-0.01	$1.13_{\pm 0.04}$	$1.12_{\pm 0.04}$	0.01	$-0.31_{\pm 0.07}$	$-0.09_{\pm 0.07}$	-0.22
NJ	**Flow-R**	$8.46_{\pm 0.37}$	$7.21_{\pm 0.42}$	1.25	$5.12_{\pm 0.73}$	$3.42_{\pm 0.75}$	1.70	$7.53_{\pm 0.39}$	$6.44_{\pm 0.44}$	1.08
	RePaint	$7.20_{\pm 0.31}$	$6.69_{\pm 0.40}$	0.50	$3.76_{\pm 0.73}$	$2.79_{\pm 0.82}$	0.96	$9.36_{\pm 0.34}$	$8.93_{\pm 0.41}$	0.43
	SLBR	$0.00_{\pm 0.00}$	$0.00_{\pm 0.00}$	0.00	$0.36_{\pm 0.04}$	$0.39_{\pm 0.04}$	-0.03	$-1.14_{\pm 0.09}$	$-0.96_{\pm 0.08}$	-0.18
	DeNet	$0.00_{\pm 0.00}$	$0.00_{\pm 0.00}$	0.00	$0.46_{\pm 0.04}$	$0.47_{\pm 0.04}$	-0.01	$-1.04_{\pm 0.07}$	$-0.90_{\pm 0.07}$	-0.14
OS	**Flow-R**	$8.58_{\pm 0.34}$	$7.18_{\pm 0.39}$	1.40	$4.28_{\pm 0.69}$	$1.82_{\pm 0.69}$	2.46	$7.90_{\pm 0.41}$	$7.02_{\pm 0.39}$	0.88
	RePaint	$7.44_{\pm 0.33}$	$6.60_{\pm 0.38}$	0.84	$4.03_{\pm 0.68}$	$2.03_{\pm 0.77}$	2.00	$10.62_{\pm 0.43}$	$10.09_{\pm 0.41}$	0.53
	SLBR	$-0.01_{\pm 0.00}$	$-0.01_{\pm 0.00}$	0.00	$0.24_{\pm 0.04}$	$0.31_{\pm 0.04}$	-0.07	$-1.13_{\pm 0.08}$	$-0.95_{\pm 0.08}$	-0.17
	DeNet	$0.00_{\pm 0.00}$	$0.00_{\pm 0.00}$	0.00	$0.34_{\pm 0.04}$	$0.41_{\pm 0.04}$	-0.07	$-1.03_{\pm 0.07}$	$-0.82_{\pm 0.07}$	-0.21

Cartoon

		PSNR			SSIM × 100			LPIPS × 100		
		Random	**HARVIM**	**Imp (↑)**	**Random**	**HARVIM**	**Imp (↑)**	**Random**	**HARVIM**	**Imp (↑)**
DIG	**Flow-R**	$6.53_{\pm 1.18}$	$-0.86_{\pm 1.74}$	7.39	$2.55_{\pm 0.77}$	$-0.88_{\pm 1.25}$	3.42	$3.22_{\pm 0.27}$	$2.21_{\pm 0.24}$	1.01
	RePaint	$4.75_{\pm 1.36}$	$3.01_{\pm 1.26}$	1.74	$4.76_{\pm 1.04}$	$3.84_{\pm 1.23}$	0.92	$5.96_{\pm 0.27}$	$5.74_{\pm 0.21}$	0.22
	SLBR	$0.12_{\pm 0.05}$	$0.14_{\pm 0.05}$	-0.02	$1.15_{\pm 0.17}$	$0.99_{\pm 0.10}$	0.15	$0.26_{\pm 0.14}$	$0.49_{\pm 0.13}$	-0.23
	DeNet	$0.16_{\pm 0.03}$	$0.14_{\pm 0.05}$	0.03	$1.09_{\pm 0.11}$	$0.98_{\pm 0.09}$	0.11	$0.27_{\pm 0.14}$	$0.45_{\pm 0.14}$	-0.19
NJ	**Flow-R**	$2.78_{\pm 1.27}$	$1.83_{\pm 1.41}$	0.95	$-0.59_{\pm 1.85}$	$-3.58_{\pm 2.08}$	2.98	$7.53_{\pm 0.39}$	$6.44_{\pm 0.44}$	1.08
	RePaint	$2.46_{\pm 1.14}$	$1.86_{\pm 1.08}$	0.60	$-2.95_{\pm 1.77}$	$-3.96_{\pm 2.33}$	1.01	$9.36_{\pm 0.34}$	$8.93_{\pm 0.41}$	0.43
	SLBR	$-0.02_{\pm 0.01}$	$-0.06_{\pm 0.03}$	0.04	$0.38_{\pm 0.09}$	$0.21_{\pm 0.17}$	0.17	$-0.32_{\pm 0.24}$	$-0.37_{\pm 0.29}$	0.04
	DeNet	$-0.01_{\pm 0.00}$	$-0.03_{\pm 0.01}$	0.02	$0.45_{\pm 0.10}$	$0.42_{\pm 0.10}$	0.02	$-0.29_{\pm 0.23}$	$-0.19_{\pm 0.19}$	-0.09
OS	**Flow-R**	$2.97_{\pm 0.44}$	$1.52_{\pm 0.44}$	1.45	$-1.20_{\pm 1.72}$	$-5.25_{\pm 1.76}$	4.05	$7.90_{\pm 0.41}$	$7.02_{\pm 0.39}$	0.88
	RePaint	$3.52_{\pm 0.36}$	$1.82_{\pm 0.25}$	1.70	$-1.21_{\pm 2.14}$	$-5.03_{\pm 1.70}$	3.82	$10.62_{\pm 0.43}$	$10.09_{\pm 0.41}$	0.53
	SLBR	$-0.02_{\pm 0.00}$	$-0.04_{\pm 0.01}$	0.02	$0.32_{\pm 0.12}$	$0.28_{\pm 0.11}$	0.03	$-0.30_{\pm 0.27}$	$-0.37_{\pm 0.21}$	0.08
	DeNet	$-0.02_{\pm 0.00}$	$-0.04_{\pm 0.01}$	0.02	$0.32_{\pm 0.12}$	$0.29_{\pm 0.11}$	0.03	$-0.30_{\pm 0.27}$	$-0.36_{\pm 0.22}$	0.07

Table 4: Reconstruction quality of different watermark removal methods on random and HARVIM's learned watermarks. Lower PSNR, SSIM, and higher LPIPS indicates better robustness against removal.

		CelebA					
		PSNR		SSIM × 100		LPIPS × 100	
		Random	HARVIM	Random	HARVIM	Random	HARVIM
DIG	Flow-R	$35.15_{\pm 0.36}$	$30.32_{\pm 0.59}$	$96.67_{\pm 0.18}$	$94.02_{\pm 0.33}$	$0.57_{\pm 0.09}$	$1.36_{\pm 0.14}$
	RePaint	$35.34_{\pm 0.42}$	$33.99_{\pm 0.48}$	$97.38_{\pm 0.15}$	$96.67_{\pm 0.23}$	$0.64_{\pm 0.10}$	$0.69_{\pm 0.08}$
	SLBR	$22.20_{\pm 0.19}$	$22.81_{\pm 0.29}$	$90.39_{\pm 0.24}$	$90.64_{\pm 0.30}$	$3.94_{\pm 0.27}$	$3.73_{\pm 0.22}$
	DeNet	$22.19_{\pm 0.19}$	$22.82_{\pm 0.29}$	$90.41_{\pm 0.24}$	$90.67_{\pm 0.30}$	$3.94_{\pm 0.28}$	$3.67_{\pm 0.21}$
NJ	Flow-R	$23.51_{\pm 0.33}$	$21.58_{\pm 0.29}$	$83.84_{\pm 0.50}$	$82.46_{\pm 0.49}$	$4.33_{\pm 0.22}$	$4.89_{\pm 0.23}$
	RePaint	$24.45_{\pm 0.35}$	$24.23_{\pm 0.35}$	$84.83_{\pm 0.55}$	$84.41_{\pm 0.56}$	$4.09_{\pm 0.20}$	$4.00_{\pm 0.27}$
	SLBR	$14.21_{\pm 0.11}$	$14.26_{\pm 0.16}$	$72.47_{\pm 0.36}$	$73.16_{\pm 0.48}$	$12.18_{\pm 0.36}$	$11.63_{\pm 0.39}$
	DeNet	$14.22_{\pm 0.11}$	$14.26_{\pm 0.16}$	$72.52_{\pm 0.36}$	$73.19_{\pm 0.48}$	$12.16_{\pm 0.37}$	$11.62_{\pm 0.39}$
OS	Flow-R	$22.44_{\pm 0.31}$	$20.34_{\pm 0.25}$	$79.54_{\pm 0.53}$	$78.48_{\pm 0.48}$	$6.36_{\pm 0.30}$	$6.51_{\pm 0.25}$
	RePaint	$23.27_{\pm 0.37}$	$22.57_{\pm 0.37}$	$82.33_{\pm 0.56}$	$81.87_{\pm 0.57}$	$5.12_{\pm 0.31}$	$5.09_{\pm 0.31}$
	SLBR	$12.87_{\pm 0.11}$	$13.13_{\pm 0.15}$	$68.08_{\pm 0.35}$	$70.52_{\pm 0.46}$	$14.66_{\pm 0.37}$	$13.89_{\pm 0.35}$
	DENET	$12.87_{\pm 0.11}$	$13.13_{\pm 0.15}$	$68.14_{\pm 0.36}$	$70.58_{\pm 0.46}$	$14.59_{\pm 0.37}$	$13.84_{\pm 0.35}$
		ImageNet					
		PSNR		SSIM × 100		LPIPS × 100	
		Random	HARVIM	Random	HARVIM	Random	HARVIM
DIG	Flow-R	$31.33_{\pm 0.43}$	$28.25_{\pm 0.52}$	$93.61_{\pm 0.26}$	$91.73_{\pm 0.32}$	$2.24_{\pm 0.19}$	$3.66_{\pm 0.27}$
	RePaint	$30.93_{\pm 0.49}$	$30.23_{\pm 0.50}$	$93.59_{\pm 0.26}$	$93.11_{\pm 0.29}$	$2.55_{\pm 0.22}$	$2.93_{\pm 0.26}$
	SLBR	$21.78_{\pm 0.20}$	$22.20_{\pm 0.25}$	$89.44_{\pm 0.35}$	$89.32_{\pm 0.38}$	$8.49_{\pm 0.60}$	$7.71_{\pm 0.58}$
	DENET	$21.79_{\pm 0.20}$	$22.22_{\pm 0.26}$	$89.52_{\pm 0.34}$	$89.43_{\pm 0.37}$	$8.44_{\pm 0.60}$	$7.65_{\pm 0.58}$
NJ	Flow-R	$22.15_{\pm 0.34}$	$21.14_{\pm 0.34}$	$75.34_{\pm 0.50}$	$74.44_{\pm 0.46}$	$12.32_{\pm 0.49}$	$11.99_{\pm 0.42}$
	RePaint	$20.87_{\pm 0.28}$	$20.60_{\pm 0.35}$	$70.52_{\pm 0.47}$	$70.39_{\pm 0.56}$	$14.98_{\pm 0.51}$	$14.68_{\pm 0.49}$
	SLBR	$13.70_{\pm 0.12}$	$13.94_{\pm 0.15}$	$69.97_{\pm 0.49}$	$70.83_{\pm 0.52}$	$20.40_{\pm 0.79}$	$19.25_{\pm 0.74}$
	DENET	$13.70_{\pm 0.12}$	$13.94_{\pm 0.15}$	$70.07_{\pm 0.49}$	$70.91_{\pm 0.52}$	$20.30_{\pm 0.78}$	$19.19_{\pm 0.74}$
OS	Flow-R	$20.91_{\pm 0.31}$	$19.89_{\pm 0.32}$	$70.11_{\pm 0.47}$	$69.82_{\pm 0.45}$	$16.60_{\pm 0.58}$	$16.65_{\pm 0.54}$
	RePaint	$19.77_{\pm 0.30}$	$19.31_{\pm 0.31}$	$66.94_{\pm 0.43}$	$67.11_{\pm 0.50}$	$17.33_{\pm 0.59}$	$17.24_{\pm 0.58}$
	SLBR	$12.33_{\pm 0.12}$	$12.71_{\pm 0.16}$	$65.60_{\pm 0.48}$	$67.87_{\pm 0.56}$	$23.16_{\pm 0.77}$	$21.50_{\pm 0.77}$
	DENET	$12.34_{\pm 0.12}$	$12.72_{\pm 0.16}$	$65.70_{\pm 0.48}$	$67.97_{\pm 0.56}$	$23.07_{\pm 0.77}$	$21.37_{\pm 0.77}$
		Cartoon					
		PSNR		SSIM × 100		LPIPS × 100	
		Random	HARVIM	Random	HARVIM	Random	HARVIM
DIG	Flow-R	$30.74_{\pm 1.31}$	$23.73_{\pm 1.02}$	$93.89_{\pm 0.57}$	$90.52_{\pm 0.45}$	$1.22_{\pm 0.22}$	$3.52_{\pm 0.40}$
	RePaint	$28.30_{\pm 1.45}$	$26.88_{\pm 0.96}$	$93.15_{\pm 0.81}$	$92.44_{\pm 0.84}$	$2.60_{\pm 0.51}$	$3.21_{\pm 0.78}$
	SLBR	$24.25_{\pm 0.76}$	$24.66_{\pm 0.84}$	$92.03_{\pm 0.55}$	$91.99_{\pm 1.12}$	$2.83_{\pm 0.36}$	$2.48_{\pm 0.34}$
	DENET	$24.29_{\pm 0.77}$	$24.65_{\pm 0.84}$	$91.97_{\pm 0.57}$	$91.98_{\pm 1.13}$	$2.83_{\pm 0.36}$	$2.52_{\pm 0.36}$
NJ	Flow-R	$19.07_{\pm 0.68}$	$18.46_{\pm 0.76}$	$73.71_{\pm 1.16}$	$72.86_{\pm 1.67}$	$11.31_{\pm 1.42}$	$11.60_{\pm 1.22}$
	RePaint	$18.78_{\pm 1.05}$	$18.53_{\pm 0.87}$	$69.39_{\pm 0.96}$	$70.60_{\pm 1.06}$	$12.44_{\pm 1.10}$	$16.02_{\pm 1.51}$
	SLBR	$16.31_{\pm 0.85}$	$16.62_{\pm 0.92}$	$74.45_{\pm 1.07}$	$76.46_{\pm 1.60}$	$10.96_{\pm 1.77}$	$10.37_{\pm 1.36}$
	DENET	$16.32_{\pm 0.86}$	$16.65_{\pm 0.93}$	$74.52_{\pm 1.09}$	$76.67_{\pm 1.59}$	$10.93_{\pm 1.76}$	$10.20_{\pm 1.30}$
OS	Flow-R	$17.95_{\pm 0.71}$	$17.08_{\pm 0.63}$	$68.71_{\pm 0.88}$	$68.07_{\pm 0.88}$	$15.43_{\pm 1.10}$	$16.66_{\pm 1.64}$
	RePaint	$17.98_{\pm 0.81}$	$18.32_{\pm 0.78}$	$65.67_{\pm 1.06}$	$67.84_{\pm 1.15}$	$15.37_{\pm 0.71}$	$15.39_{\pm 1.16}$
	SLBR	$15.01_{\pm 0.93}$	$15.58_{\pm 0.93}$	$70.07_{\pm 1.27}$	$73.52_{\pm 1.33}$	$13.77_{\pm 1.73}$	$13.04_{\pm 1.74}$
	DENET	$15.01_{\pm 0.93}$	$15.58_{\pm 0.93}$	$70.07_{\pm 1.27}$	$73.52_{\pm 1.33}$	$13.77_{\pm 1.73}$	$13.03_{\pm 1.74}$