Watermark-embedded Adversarial Examples for Copyright Protection against Diffusion Models

Peifei Zhu, Tsubasa Takahashi, Hirokatsu Kataoka
LY Corporation
{peifei.zhu, tsubasa.takahashi, jpz4219, }@lycorp.co.jp

Abstract

Diffusion Models (DMs) have shown remarkable capabilities in various image-generation tasks. However, there are growing concerns that DMs could be used to imitate unauthorized creations and thus raise copyright issues. To address this issue, we propose a novel framework that embeds personal watermarks in the generation of adversarial examples. Such examples can force DMs to generate images with visible watermarks and prevent DMs from imitating unauthorized images. We construct a generator based on conditional adversarial networks and design three losses (adversarial loss, GAN loss, and perturbation loss) to generate adversarial examples that have subtle perturbation but can effectively attack DMs to prevent copyright violations. Training a generator for a personal watermark by our method only requires 5-10 samples within 2-3 minutes, and once the generator is trained, it can generate adversarial examples with that watermark significantly fast (0.2s per image). We conduct extensive experiments in various conditional image-generation scenarios. Compared to existing methods that generate images with chaotic textures, our method adds visible watermarks on the generated images, which is a more straightforward way to indicate copyright violations. We also observe that our adversarial examples exhibit good transferability across unknown generative models. Therefore, this work provides a simple yet powerful way to protect copyright from DM-based imitation.

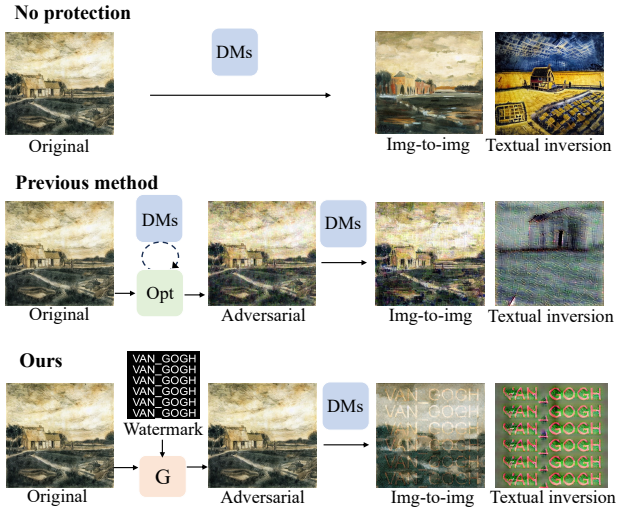

Figure 1. Copyright issues of DMs and adversarial example-based methods for copyright protection. Without protection, DMs can easily imitate original images under different image generation scenarios. A previous method (AdvDM [28]) generates adversarial examples by optimization against DMs to prevent DMs from extracting the feature of the original images, resulting in the generation of chaotic images. Our method goes one step further to build a generator that embeds personal watermarks into the generation of adversarial images. Such examples force DMs to generate images with visible watermarks for tracing copyright. Our method is fast, simple yet powerful in protecting copyrights against DMs.

1. Introduction

Diffusion models (DMs) [19, 42, 46] have garnered significant attention in both the computer vision industry and academia. Compared to previous generative models such as generative adversarial networks (GANs) [15, 23, 32], DMs have demonstrated a remarkable ability to generate realistic images with higher resolution and diversity. Despite the positive aspects of DMs, there are public concerns about potential copyright violations when unauthorized images are used to train DMs or to generate new creations by DMs [55]. For example, an entity could use copyrighted paintings shared on the Internet to generate images with a similar style and appearance, potentially earning illegal revenue. Therefore, it is necessary to develop techniques to prevent the imitation of unauthorized creations by DMs.

Several previous works have been proposed to address these concerns. Watermarking [2, 33] is a common solution that embeds an invisible message into the image, which can then be extracted to identify the existence of copyright.

Recent techniques typically integrate watermarking into the generative process of DMs [7, 12, 38]. However, these methods have several limitations. First, they are primarily designed to protect the copyright of DMs or to distinguish generated images from natural ones, which differs from our goal of protecting the copyright of human creations. Second, embedding watermarks into the generative process requires re-training or at least fine-tuning DMs, and a post-process to extract the watermark is needed to identify it, which can be costly. On the other hand, methods that use adversarial examples for DMs to protect copyright have been proposed recently [27, 28, 50]. The idea is to generate adversarial examples to hinder DMs from extracting the features of the original images. This generation process does not require any re-training of DMs or any post-processing for the generated image. However, there are still several limitations. First, even though these adversarial examples could add chaotic textures to the generated images, such changes are difficult to comprehend. For instance, users may wonder if the problem originates from the DM itself rather than considering the copyright issue. Second, the original image's copyright is not traceable since no copyright-related information is embedded. Third, each adversarial example needs to be optimized separately against DMs, which makes the generation time-consuming.

In this work, we propose a simple yet powerful method to protect copyrighted images from imitation by DMs. Different from previous adversarial example-based methods, we go one step further by embedding personal watermarks into the generation of adversarial examples for copyright tracing. Such examples could force DMs to generate images with visible watermarks as well as chaotic textures. In addition, instead of using iterative optimization against DMs to generate examples, we train a generator beforehand using only a few samples. Once the generator is trained, it can be used to generate adversarial examples significantly fast. As shown in Figure 1, instead of posting the original image on the Internet, posting adversarial examples with only subtle changes could prevent DMs from imitation. Compared to previous methods, our method provides a more straightforward way to warn of potential copyright violations.

Our contributions can be summarized as follows:

- We design a novel framework that embeds personal watermarks into the generation of adversarial examples to prevent copyright violations caused by DM-based imitation. Our adversarial examples can force DMs to generate images with visible watermarks for tracing copyright.
- We train a generator based on a conditional GAN architecture to generate adversarial examples (Figure 2). We design three losses: adversarial loss, GAN loss, and weighted perturbation loss. These losses aim to improve the attack ability of the adversarial examples while making the perturbation invisible to the human eye.
- We conduct extensive experiments in various image generation scenarios, along with robustness evaluations against defenses and assessments of transferability to other models. The experiments demonstrate that our adversarial examples can prevent unauthorized images from being learned and can generate visible watermarks for copyright tracing. Our generation process is significantly fast (0.2s per image), and the generated examples also exhibit good transferability across other generative models.

2. Related Works

2.1. Generative Diffusion Models

In recent years, DMs have made significant advancements in various fields [9, 37, 45, 55]. DMs are a type of generative model that simulates a diffusion process to generate new data samples. DMs consist of a forward process and a reverse process. The forward process involves gradually adding Gaussian noise to the data over a series of time steps until it becomes completely random. The reverse process gradually removes noise through a series of learned denoising steps, reconstructing the data back to its original form or generating new samples from the same distribution.

DMs have been applied to various generation tasks such as image synthesis [9, 43], text-to-image generation [1, 41, 58, 59], and text-guided image editing [4, 34, 54, 61]. Among these, latent diffusion models [42], which perform the diffusion steps in the latent image space, have demonstrated a remarkable ability to generate high-resolution and realistic images. Despite the impressive performance, some concerns that generating new creations by DMs based on unauthorized images could lead to copyright issues.

2.2. Image Watermarking

Image watermarking is a technique that embeds a piece of information into an image to protect the image from copyright infringement. Traditional watermarking methods typically involve altering pixel values [6] or manipulating the frequency domain of an image [35] to embed watermarks. More recently, deep-learning-based methods have been developed. A common approach is using encoder-extractor networks, where an encoder embeds the watermark into the image, and an extractor is trained to retrieve the watermark from the marked image [49, 60, 63].

In terms of watermarking for generative models, most existing works focus on protecting the copyright of generative models [11, 30, 38, 51, 57] or distinguishing generative images from natural images [7, 12, 56, 62]. These methods involve reformulating the training process of DMs to inject watermarks into the images, as well as developing a decoder to identify the embedded messages from the generated images. However, the re-training can be costly and it is impractical to check every generated image with a decoder.

2.3. Adversarial Examples for Generative models

Adversarial examples are created by adding small but intentional perturbations to input samples such that the perturbed inputs cause the model to produce incorrect answers. Various adversarial attack methods have been proposed to confuse classification models [10, 16, 31, 52] or segmentation models [13, 53, 65]. There are also several existing works studying adversarial examples for generative models such as GANs [24], variational autoencoders (VAEs) [47], and flow-based generative models [39].

Recently, methods that generate adversarial examples for DMs [27, 28, 50] have been proposed. These methods maximize the diffusion loss to obtain a perturbation, and adding such a perturbation to the original image enables DMs to generate images with chaotic content. However, these methods have several drawbacks: 1) such chaotic content is not straightforward to indicate copyright violation, 2) the copyright of the original image is not traceable, and 3) optimization against DMs is time-consuming. We propose a simple yet powerful method that directly incorporates personal watermarks into the generation of adversarial examples, and we build a generator that can generate adversarial examples significantly faster than existing methods.

3. Proposed Method

3.1. Problem Statement

Given an original image x, we have a diffusion model θ that can learn the data distribution of x and generate images with similar style and appearance. In this work, we aim to generate an adversarial example x' that can prevent θ from extracting the feature of x and can directly show copyright information on the generated image. For example, for image-to-image generation, the generated image from x' by using θ can be denoted as $\theta(x')$. We design a way to provide the copyright information by using a watermark that contains the ownership details (e.g. artist name) of the original image. Therefore, given a watermark image m, the adversarial example x' that embeds m can be calculated by

$$\begin{aligned} x' &:= \underset{x'}{\operatorname{argmin}} \|\theta(x') - m\|, \\ \text{s.t.} \quad & \|x - x'\| \leq \sigma, \end{aligned} \tag{1}$$

where σ is a constant to control the range of the perturbation. This is a targeted attack setting where we compel the image generated by DMs to be closer to the watermark, while ensuring that the adversarial example remains as close to the original one as possible.

However, several issues are associated with generating adversarial examples using Equation 1. First, directly optimizing x' against DMs for each x could be time-consuming, so we need to find a way to speed up the generation process. Second, since there are several image-generation scenarios related to imitating the original image, such as text-guided image-to-image generation [4], text-to-image generation under textual inversion [14] or DreamBooth [43], we aim to generate an adversarial example x' that can be applicable to all these scenarios. Meanwhile, we also expect that x' could have good transferability on other generative models. Third, since our target attack aims to add personal watermarks to the generated images, the perturbation naturally becomes larger in the watermark regions. We need a solution to make the perturbation invisible to human eyes.

To address the first issue, we train a generator beforehand instead of implementing optimization every time. To address the second and third issues, we design an adversarial loss, a GAN loss, and a perturbation loss to improve the attack ability across various tasks and models while keeping the perturbation invisible.

3.2. Architecture Overview

We propose a conditional GAN architecture for generating watermark-embedded adversarial examples. The architecture of our method contains three components: a generator G, a discriminator D, and a target DM θ, shown in Figure 2. The generator G has an encoder to extract features from the inputs and a decoder to generate perturbation using the features. The inputs of G are the original image x and the watermark m, and the perturbation is generated conditioned on m. We perform the conditioning by concatenating x and m along the channel dimension and feeding them into G. The adversarial example x' can be obtained by adding x and the generated perturbation. The discriminator D is a classifier that distinguishes x' from x, and the target DM θ is always kept frozen.

We design three losses to optimize G and D. 1) A **GAN loss** aims to compel x' to be closer to x. By using the GAN loss, we can train G to produce images that D cannot distinguish from input images. 2) A **perturbation loss** is designed to bound the magnitude of the perturbation and make it invisible. 3) An **adversarial loss** is used for G to generate perturbation that can attack DMs under various scenarios. The details are described in Section 3.3. Once the generator G is optimized, it can be used directly to generate adversarial examples significantly fast.

3.3. Loss functions

GAN loss. $\mathcal{L}_{GAN}$ is used to quantify the difference between the adversarial example and the original image. For a set of images, $\mathcal{L}_{GAN}$ can be written as:

$$\mathcal{L}_{GAN} = \mathbb{E}_x \log D(x) + \mathbb{E}_x \log\{1 - D(x')\}. \tag{2}$$

The adversarial image x' is generated under the condition of the watermark m, which can be written as:

$$x' = x + G(x \mid m). \tag{3}$$

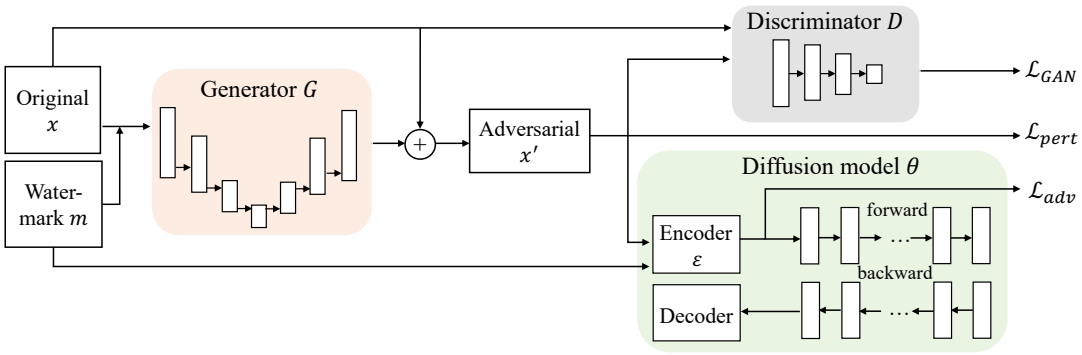

Figure 2. Architecture overview. G generates perturbation for x conditioned on m. D and G produce $\mathcal{L}_{GAN}$ to compel x' to be closer to x. $\mathcal{L}_{adv}$ aims to force the image generated by θ to display a visible watermark m, and $\mathcal{L}_{pert}$ further bounds the magnitude of the perturbation.

Perturbation loss. $\mathcal{L}_{pert}$ aims to limit the magnitude of the perturbation. We design this loss based on the soft hinge loss used in previous adversarial attacks [3, 29, 52]. However, the soft hinge loss assigns equal importance to all perturbations within the image. In our case, the perturbation in the watermark region will be larger than that in other regions, making the watermark visible even in the adversarial example. To better conceal the watermark as well as bound the perturbation in the whole image, we design a weighted perturbation loss, where we assign larger weights w to the watermark region. This encourages the perturbation in the watermark region to be smaller and invisible. For a set of images, this loss can be expressed as:

$$\mathcal{L}_{pert} = \mathbb{E}_x \max(0, \|G\left(x \mid m\right)\left(1 + w \cdot m\right)\|_2 - c), \quad (4)$$

where c is a constant denoting a user-specified bound.

Adversarial loss. $\mathcal{L}_{adv}$ is used to generate a perturbation that can attack DMs. As we expect our adversarial examples to have transferability across different image generation scenarios and various generative models, we design the loss based on the latent representation of latent diffusion models (LDM). We choose LDM for two reasons. First, LDM has outperformed other models in both high-quality image generation and sampling efficiency, and it has been used in various generative models. Generating adversarial examples against LDM naturally increases their transferability across various models. Second, LDM maps an image x to a latent representation $\varepsilon\left(x\right)$ which is the output of the Variational Auto-Encoder (VAE). This representation is widely used in conditional image generation scenarios, and it provides a simple way to control the image generation of DMs. Therefore, instead of directly exploiting the generated image, we minimize the distance between the representation of the adversarial example $\varepsilon\left(x'\right)$ and that of the watermark $\varepsilon\left(m\right)$, formulated as:

$$\mathcal{L}_{adv} = \mathbb{E}_x \|\varepsilon\left(x'\right) - \varepsilon\left(m\right)\|_2. \quad (5)$$

Optimization. Finally, we combine the above three losses to formulate the objective function for training, written as:

$$\min_G \max_D V(D, G) = \mathcal{L}_{adv} + \alpha \mathcal{L}_{GAN} + \beta \mathcal{L}_{pert}, \quad (6)$$

where α and β are weights to control the balance of the three losses. By playing a minimax game between the G and D, the optimal parameters of the model can be obtained.

3.4. Image Generation under Different Settings

Since our target is to prevent copyright violations, we focus on conditional image generation scenarios where DMs sample the distribution of the original images to generate new ones. Such scenarios include text-guided image-to-image generation [36] and text-to-image generation under textual inversion [14] described in this section, and other scenarios including DreamBooth [43], LoRA [20], and Custom Diffusion [25] discussed in Appendix A.

Text-guide image-to-image generation. In this setting, we can pass a text prompt and an input image to condition the image generation. Our method can be directly applied to this setting, as shown in Figure 2. During the training of the generator and discriminator, we simply use the same prompt (e.g. "A painting") for all samples. For the inference, any prompt can be used for generating new images.

Textual inversion. Textual inversion is a method for capturing new concepts from a small number of images. These new concepts can then be used to control image generation in text-to-image settings. The steps to apply our method in this setting are as follows. First, given a small set of images within a similar category (e.g. same artist), we extract a text S_* to represent our new concept. We then replace the vector associated with the tokenized string with an optimized embedding v_*. The optimization can be applied by minimizing the LDM loss over the images from the small set. During the image generation by DMs, the text S_* is used as a condition to generate new images. We conduct this generation

for both original images and adversarial images. The adversarial images are generated through the image-to-image generation setting as shown in Figure 2.

4. Experiments

In this section, we evaluate our method for generating adversarial examples for DMs. We conduct experiments under text-guided image-to-image generation and textual inversion, performing various analyses to assess the effectiveness, robustness, and transferability of our method. We perform the evaluation on WikiArt [48] and ImageNet [44] datasets. All experiments are run on an NVIDIA A100 80GB GPU. Training the generator using 100 samples with 200 epochs takes about 20 minutes.

4.1. Experimental Settings

Datasets. The WikiArt dataset contains 50k paintings from 195 artists. We randomly select 3000 paintings from 50 artists. We create different watermarks for different artists. These watermarks are binary images with a black background and white artist names such as "VAN_GOGH", "CLAUDE_MONET". Since only a small number of samples are needed to train our generator, we select 10 images from each artist (500 images in total) for training and the remaining 2500 images are for evaluation. For the ImageNet dataset, we randomly select 2000 images (100 for training, 1900 for evaluation) from the goldfish, tiger shark, peacock, goose, Eskimo dog, and tabby cat category. The watermarks are designed as "IMAGENET_CAT", "IMAGENET_FISH", "IMAGENET_DOG", etc.

Evaluation metrics. We evaluate our method from two perspectives. For the adversarial example, we use MSE, PSNR, and SSIM to measure the image quality and the difference from the original image. For the image generated by DMs, we use Fréchet Inception Distance (FID) [18] and precision (prec.) [26] to measure the similarity between the generated and original images. Since we expect DMs to generate images with visible watermarks, we use Normalized Cross-Correlation (NCC) [21] to measure the similarity between the generated image and the watermark. Please note that, unlike traditional watermark methods, we do not extract the watermark from the generated image, and thus NCC is smaller in our case. However, to reduce the influence of the background, we first calculate a difference image between the image generated from the original image and that from the adversarial example, and then use this difference image and the watermark to obtain NCC.

Implementation details. For the generator G, we adopt an architecture similar to that in [22]. For the discriminator D, we use an architecture similar to the discriminator in AdvGAN [52]. Further details regarding the implementation of our model are provided in Appendix B. We set the weight of GAN loss $\alpha = 1.0$, the weight of perturbation loss $\beta = 10$, the weight for watermark region $w = 4$, and the bound $c = 10/255$. During the model training, we set the batch size to 8 and the learning rate to 0.001. For comparison, we use AdvDM [28] and Mist (fuse mode) [27], which also use adversarial examples to protect copyright. We set their sampling step to 100 and the maximum perturbation to $10/255$.

Method	NCC ↑	FID ↑	prec. ↓	recall
Original	0	120.7	0.85	0.28
AdvDM	0	228.2	**0.03**	0.25
Mist	0.09	205.8	0.04	0.33
Ours	**0.31**	**245.5**	**0.03**	0.29

Table 1. The performance of the generated image under text-guided image-to-image generation on WikiArt. Recall measures the diversity of the images and is only listed as a reference.

4.2. Text-guided Image-to-Image Generation

We evaluate the performance of our method under the setting described in Section 3.4. In this setting, a strength parameter is used to control the level of noise added to the original image during the generation of new images. A strength value close to 0 produces an image nearly identical to the original, while a value close to 1 results in an image that significantly differs from the original. We set the strength to 0.3 to simulate realistic imitation, and additional results with different strength values can be found in the Appendix C.1. For the text prompt, we uniformly use "A painting" for all images for convenience, and experiments using varied prompts are also included in the Appendix C.2.

We first show the evaluation result of the image generated by LDM based on either the original image or the adversarial example. A high FID and a low precision mean that the generated image is far from the original one, which further indicates that LDM fails to capture the feature of the original image. As shown in Table 1, our adversarial examples largely increase FID and decrease precision. We also calculate NCC between the generated image and the watermark. For Mist, since it is possible to input a target image into the generation, we apply our idea to use different watermarks as the target images and use the same method to calculate NCC. Results show that our method has a significantly higher NCC, which means our adversarial examples can force LDM to generate images with more visible watermarks. These results suggest our method has great potential to prevent copyright violations caused by LDM imitation.

We also evaluate the quality of the adversarial example generated by different methods. As shown in Table 2, the image quality of our adversarial examples is slightly higher than that of the existing methods. Moreover, the generation is significantly faster than the existing methods since we

Method	MSE ↓	PSNR ↑	SSIM ↑	Run time
AdvDM	0.0038	29.1	0.80	32s
Mist	0.0040	29.0	**0.81**	35s
Ours	**0.0037**	**30.1**	0.80	**0.2s**

Table 2. The image quality and generation time (per image) of the adversarial examples on WikiArt.

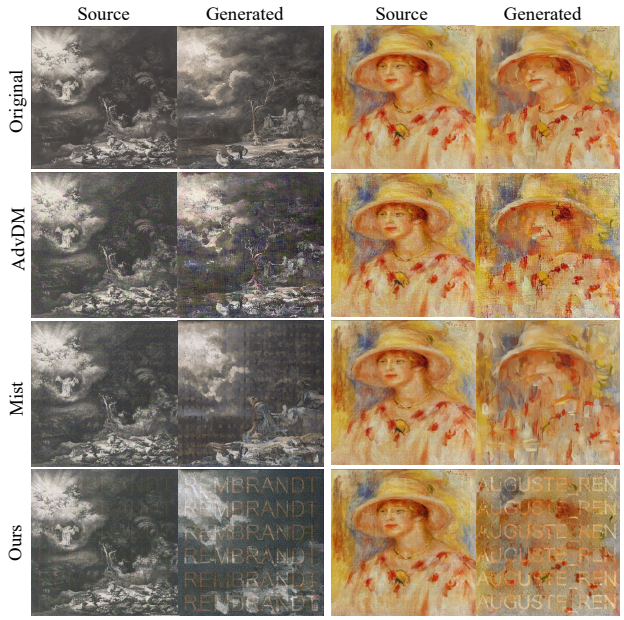

Figure 3. Comparison of different methods under text-guided image-to-image generation. Source is the image input to the LDM, and generated image is the output of the LDM (strength 0.3). The watermarks of our method are designed according to the artist name (from left to right: REMBRANDT, AUGUSTE_REN).

directly use the trained generator instead of iterative optimization against DMs. Our method has an advantage when generating a large number of examples.

For qualitative evaluation, we visualize two examples generated by different methods in Figure 3. More examples are shown in Appendix D.1. For the adversarial examples, all methods seem to introduce only subtle perturbations that are almost invisible to human eyes. For the generated images based on the adversarial examples, the existing methods succeed in adding chaotic textures to the generated image. However, such changes can be difficult to understand and evaluate. On the other hand, our adversarial examples succeed in generating images with visible watermarks as well as chaotic textures. Our method provides a more straightforward way to indicate copyright violations.

Method	Generated image		Adversarial example	
	NCC ↑	FID ↑	PSNR ↑	SSIM ↑
Original	0	120.7	-	-
AdvDM	0	327.3	26.3	0.73
Mist	0.15	318.4	26.7	0.74
Ours	**0.40**	**412.9**	**27.1**	**0.75**

Table 3. The performance of the generated image and adversarial examples on ImageNet under textual inversion.

4.3. Textual Inversion

Textual inversion is another important scenario related to copyright violations. In this scenario, given a small number of images, we first learn to represent the concept of the images in a new word S_*, then use S_* to guide image generation. Following the method of textual inversion [14], we separate our evaluation images into 4-image groups for each artist or ImageNet category. For each 4-image group, we set the maximum steps to optimize S_* to 5000. After the optimization, we generate 10 images for each S_* under 10 prompt templates. This process is implemented for both original images and our adversarial examples.

For evaluation, we compare our method with existing methods, and the results are shown in Table 3. For the generated image, our method has higher NCC and FID, which indicates our method successfully injects the watermark information in the generation of the adversarial example. In addition, our adversarial example also has higher PSNR and SSIM, which means the generated perturbation is subtle. An example of textual inversion is shown in Figure 4, and examples comparing with the previous methods are shown in Appendix D.2. The features of the original images can be learned and used to guide image generation. On the other hand, when applying our method, images are generated with very clear watermarks, which can be useful for protecting copyrighted images.

4.4. Ablation Study

Weights of the losses. This experiment aims to study the mechanism of our method. In equation 6, we design three losses to train the generator. As our target is to attack DMs, the adversarial loss $\mathcal{L}_{adv}$ is necessary, and we set its weight to 1. We then evaluate the effect of the GAN loss $\mathcal{L}_{GAN}$, the perturbation loss $\mathcal{L}_{pert}$, and the weight used to control the perturbation of the watermark region w by setting their weights α, β, and w to 0 or their default value.

The results are shown in Table 4. The first row is the result of only using $\mathcal{L}_{adv}$ and setting other losses to 0. In this case, although the attack ability for DMs is high, the generated adversarial examples are full of noise. By comparing the first and second rows, we observe that $\mathcal{L}_{GAN}$

Figure 4. Examples of textual inversion on ImageNet. The watermarks used in the examples are IMAGENET_CAT and IMAGENET_DOG.

Weights			Generated image		Adversarial example	
α	β	w	NCC ↑	FID ↑	PSNR ↑	SSIM ↑
×	×	×	0.60	565.3	13.1	0.12
✓	×	×	0.41	350.6	22.0	0.65
×	✓	×	0.35	318.2	24.2	0.70
×	✓	✓	0.34	292.2	27.8	0.76
✓	✓	✓	0.31	245.5	30.1	0.80

Table 4. The effect of the GAN loss (α), the perturbation loss (β), and the weight of the watermark region (w) on generating adversarial examples (WikiArt) under image-to-image generation.

significantly improves the image quality of the adversarial example, even though the attack ability decreases. On the other hand, $\mathcal{L}_{pert}$ which aims to bound the magnitude of the perturbation, also has an effect similar to that of $\mathcal{L}_{GAN}$ (as seen when comparing first and third rows). In addition, adding weight term to $\mathcal{L}_{pert}$ further improves the quality of the adversarial examples without sacrificing too much attack ability (as seen when comparing third and fourth rows). Finally, we combine all these terms to achieve a balanced performance between image quality and attack ability.

Maximum perturbation bound. In the case of adversarial examples, there is always a trade-off between image quality and the success rate of the attack. To explore this trade-off, we conducted experiments with different bounds. We used PSNR to measure the image quality of the adversarial examples, and NCC to assess the visibility of the watermark in the generated image which indicates the attack success rate. Table 5 shows that as the bound increases, the watermark on the generated image becomes more distinct. However, this may also cause the watermark on the adversarial image to become more visible, which can reduce the image quality. In real-world applications, users can adjust the bounds to balance between achieving high-quality adversarial images and ensuring watermark visibility to protect their work.

Bound/255	2	6	10	15	20
PSNR ↑	40.1	34.2	30.1	24.2	17.5
NCC ↑	0.13	0.22	0.31	0.37	0.41

Table 5. The effect of using different perturbation bound. The experiment uses WikiArt under image-to-image generation.

Samples	NCC ↑	FID ↑	prec. ↓	train time
1	0.12	177.8	0.32	20 sec
5	0.28	235.3	0.04	1 min
10	0.30	240.6	0.03	2 min
100	0.31	243.5	0.02	20 min

Table 6. The effect and training time of using different number of samples to train a generator for a specific watermark. The experiment is conducted under image-to-image generation on WikiArt.

The number of samples for training. To generate adversarial examples that contain a specific watermark, we need to include samples with that watermark in the training data. In this experiment, we analyze how many samples we need for a watermark to be learned by the generator. We choose a different number of samples to train a generator for a specific watermark. Please note that different watermarks can be trained together within the same generator, and this is the default setting for our other experiments. The results of training one personal watermark are in Table 6. We observe there is no large performance difference between using 10 samples and 100 samples for training, which shows that our method only needs a small number of samples to train to embed a specific watermark. Therefore, one use case of our method is to create a personal watermark generator using a few samples within 2-3 min, and then this generator can be used to embed the personal watermark for any of that user's creations to protect copyright.

Defense	No attack (Origin)		Attack (Ours)	
	NCC ↑	FID ↑	NCC ↑	FID ↑
No defense	0	120.7	0.31	245.5
JPEG	0	124.1	0.21	202.2
RS	0	123.2	0.20	195.6
TVM	0	132.5	0.16	166.8

Table 7. The effects of applying defenses on adversarial examples under image-to-image generation.

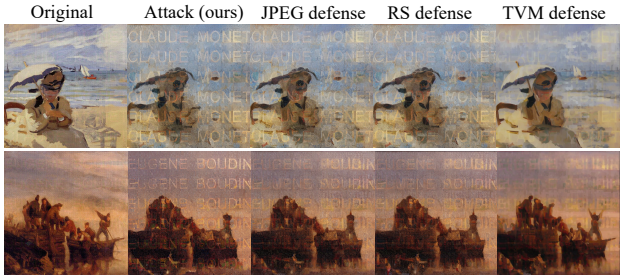

Figure 5. Examples of the generated image from original image, attack image, and attack image with defense.

4.5. Robustness of adversarial examples

To evaluate the robustness of the adversarial examples generated by our method, we apply several widely used adversarial defenses including JPEG compression [8], Randomized Smoothing (RS) [5], and Total Variance Minimization (TVM) [17]. Such defenses are directly processed on the adversarial examples to eliminate the perturbations.

The results of applying attacks and defenses are shown in Table 7. The "No attack" column means applying defense on the original image to generate images. There is no watermark being embedded so NCC almost equals 0, and applying defenses slightly increases FID since the quality of the original image decreases. On the other hand, the "Attack" column means we first generate the adversarial examples, apply defenses to them (except the first row), and then use them to generate new images. We observe that NCC and FID both decrease when adding defenses to our adversarial examples. However, compared with "No attack", the "Attack" column still has higher NCC and FID which means the defenses cannot completely eliminate the effect of our attack. In Figure 5, we show some examples under defenses, and the watermark is still visible for all cases.

4.6. Transferability on Other Generative Models

In this section, we explore the transferability of our method on other generative models. This is also known as the black-box attack, where one target DM is used to train the generator and obtain the adversarial examples, and then these adversarial examples are directly used to attack other models. We generate adversarial examples on Stable Diffusion 1.5 (SD 1.5) [1] and test on other models or tools including Dreamshaper 8 [2] , NovelAI [3], and Runway AI magic tools (image variation) [4] under image-to-image generation. The details of the settings are described in the Appendix E.

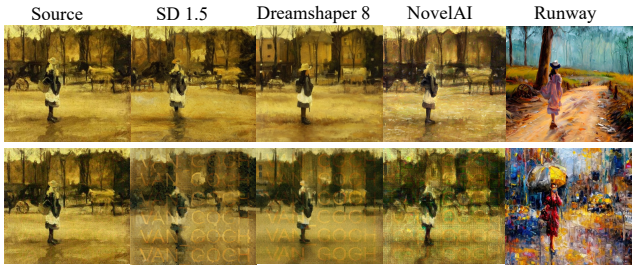

Figure 6. Black-box attack to other models under image-to-image generation. The first row: generated images from the original image. Second row: generated images from our adversarial example.

The results are shown in Figure 6, and more examples can be found in Appendix D.3. Compared to images generated from the original image (first row), the images generated from our adversarial examples (second row) contain chaotic textures and a visible watermark for most cases. For the result of the Runway tool, although there is no obvious watermark on the generated image (as no parameter can be adjusted), our method still generates an image that differs significantly from the original one. Even though the watermarks are not as strong as they are in the white-box setting, our method still demonstrates good transferability. These results show that our method has the potential to prevent copyright violations from various image generative models.

5. Conclusion

In this work, we propose a novel method for copyright prevention against DMs. We build a generator to embed personal watermarks into the adversarial examples, and such examples can force DMs to generate images with visible watermarks and chaotic textures. We use a conditional GAN architecture with three carefully designed losses to optimize the generator. Experiments show our method performs well in various image generation scenarios and exhibits good transferability to attack other models. Additionally, our generation is significantly faster than those of previous methods. Therefore, this work provides a simple yet powerful way to protect image copyright against DMs.

[1] https://huggingface.co/runwayml/stable-diffusion-v1-5
[2] https://civitai.com/models/4384/dreamshaper
[3] https://novelai.net/
[4] https://runwayml.com/ai-magic-tools/image-to-image/

References

[1] Yogesh Balaji, Seungjun Nah, Xun Huang, Arash Vahdat, Jiaming Song, Karsten Kreis, Miika Aittala, Timo Aila, Samuli Laine, Bryan Catanzaro, et al. ediffi: Text-to-image diffusion models with an ensemble of expert denoisers. *arXiv preprint arXiv:2211.01324*, 2022. 2

[2] Mahbuba Begum and Mohammad Shorif Uddin. Digital image watermarking techniques: a review. *Information*, 11(2): 110, 2020. 1

[3] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*, pages 39–57. Ieee, 2017. 4

[4] 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. 2, 3, 1

[5] Jeremy Cohen, Elan Rosenfeld, and Zico Kolter. Certified adversarial robustness via randomized smoothing. In *international conference on machine learning*, pages 1310–1320. PMLR, 2019. 8

[6] Ingemar Cox, Matthew Miller, Jeffrey Bloom, and Chris Honsinger. Digital watermarking. *Journal of Electronic Imaging*, 11(3):414–414, 2002. 2

[7] Yingqian Cui, Jie Ren, Han Xu, Pengfei He, Hui Liu, Lichao Sun, and Jiliang Tang. Diffusionshield: A watermark for copyright protection against generative diffusion models. *arXiv preprint arXiv:2306.04642*, 2023. 2

[8] Nilaksh Das, Madhuri Shanbhogue, Shang-Tse Chen, Fred Hohman, Siwei Li, Li Chen, Michael E Kounavis, and Duen Horng Chau. Shield: Fast, practical defense and vaccination for deep learning using jpeg compression. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 196–204, 2018. 8

[9] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021. 2

[10] Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. Boosting adversarial attacks with momentum. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9185–9193, 2018. 3

[11] Jianwei Fei, Zhihua Xia, Benedetta Tondi, and Mauro Barni. Supervised gan watermarking for intellectual property protection. In *2022 IEEE International Workshop on Information Forensics and Security (WIFS)*, pages 1–6. IEEE, 2022. 2

[12] Pierre Fernandez, Guillaume Couairon, Hervé Jégou, Matthijs Douze, and Teddy Furon. The stable signature: Rooting watermarks in latent diffusion models. *arXiv preprint arXiv:2303.15435*, 2023. 2

[13] Volker Fischer, Mummadi Chaithanya Kumar, Jan Hendrik Metzen, and Thomas Brox. Adversarial examples for semantic image segmentation. *arXiv preprint arXiv:1703.01101*, 2017. 3

[14] 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. 3, 4, 6

[15] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014. 1

[16] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014. 3

[17] Chuan Guo, Mayank Rana, Moustapha Cisse, and Laurens Van Der Maaten. Countering adversarial images using input transformations. *arXiv preprint arXiv:1711.00117*, 2017. 8

[18] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017. 5

[19] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020. 1

[20] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021. 4, 1

[21] Bernd Jähne. *Digital image processing*. Springer Science & Business Media, 2005. 5

[22] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part II 14*, pages 694–711. Springer, 2016. 5, 1

[23] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4401–4410, 2019. 1

[24] Jernej Kos, Ian Fischer, and Dawn Song. Adversarial examples for generative models. In *2018 ieee security and privacy workshops (spw)*, pages 36–42. IEEE, 2018. 3

[25] Nupur Kumari, Bingliang Zhang, Richard Zhang, Eli Shechtman, and Jun-Yan Zhu. Multi-concept customization of text-to-image diffusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1931–1941, 2023. 4

[26] Tuomas Kynkäänniemi, Tero Karras, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Improved precision and recall metric for assessing generative models. *Advances in Neural Information Processing Systems*, 32, 2019. 5

[27] Chumeng Liang and Xiaoyu Wu. Mist: Towards improved adversarial examples for diffusion models. *arXiv preprint arXiv:2305.12683*, 2023. 2, 3, 5

[28] Chumeng Liang, Xiaoyu Wu, Yang Hua, Jiaru Zhang, Yiming Xue, Tao Song, XUE Zhengui, Ruhui Ma, and Haibing Guan. Adversarial example does good: Preventing painting imitation from diffusion models via adversarial examples. 2023. 1, 2, 3, 5

[29] Yanpei Liu, Xinyun Chen, Chang Liu, and Dawn Song. Delving into transferable adversarial examples and black-box attacks. *arXiv preprint arXiv:1611.02770*, 2016. 4

[30] Yugeng Liu, Zheng Li, Michael Backes, Yun Shen, and Yang Zhang. Watermarking diffusion model. *arXiv preprint arXiv:2305.12502*, 2023. 2

[31] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017. 3

[32] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014. 1

[33] Saraju P Mohanty. Digital watermarking: A tutorial review. *URL: http://www. csee. usf. edu/˜ smohanty/research/Reports/WMSurvey1999Mohanty. pdf*, 1999. 1

[34] Ron Mokady, Amir Hertz, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Null-text inversion for editing real images using guided diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6038–6047, 2023. 2

[35] KA Navas, Mathews Cheriyan Ajay, M Lekshmi, Tampy S Archana, and M Sasikumar. Dwt-dct-svd based watermarking. In *2008 3rd International Conference on Communication Systems Software and Middleware and Workshops (COMSWARE'08)*, pages 271–274. IEEE, 2008. 2

[36] Alex Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob McGrew, Ilya Sutskever, and Mark Chen. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. *arXiv preprint arXiv:2112.10741*, 2021. 4

[37] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International Conference on Machine Learning*, pages 8162–8171. PMLR, 2021. 2

[38] Sen Peng, Yufei Chen, Cong Wang, and Xiaohua Jia. Protecting the intellectual property of diffusion models by the watermark diffusion process. *arXiv preprint arXiv:2306.03436*, 2023. 2

[39] Phillip Pope, Yogesh Balaji, and Soheil Feizi. Adversarial robustness of flow-based generative models. In *International Conference on Artificial Intelligence and Statistics*, pages 3795–3805. PMLR, 2020. 3

[40] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015. 2

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

[42] 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. 1, 2

[43] 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. 2, 3, 4, 1

[44] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115 (3):211–252, 2015. 5

[45] 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. PMLR, 2015. 2

[46] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020. 1

[47] Pedro Tabacof, Julia Tavares, and Eduardo Valle. Adversarial images for variational autoencoders. *arXiv preprint arXiv:1612.00155*, 2016. 3

[48] Wei Ren Tan, Chee Seng Chan, Hernan Aguirre, and Kiyoshi Tanaka. Improved artgan for conditional synthesis of natural image and artwork. *IEEE Transactions on Image Processing*, 28(1):394–409, 2019. 5

[49] Matthew Tancik, Ben Mildenhall, and Ren Ng. Stegastamp: Invisible hyperlinks in physical photographs. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2117–2126, 2020. 2

[50] 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. 2, 3

[51] Hanzhou Wu, Gen Liu, Yuwei Yao, and Xinpeng Zhang. Watermarking neural networks with watermarked images. *IEEE Transactions on Circuits and Systems for Video Technology*, 31(7):2591–2601, 2020. 2

[52] Chaowei Xiao, Bo Li, Jun-Yan Zhu, Warren He, Mingyan Liu, and Dawn Song. Generating adversarial examples with adversarial networks. *arXiv preprint arXiv:1801.02610*, 2018. 3, 4, 5

[53] Cihang Xie, Jianyu Wang, Zhishuai Zhang, Yuyin Zhou, Lingxi Xie, and Alan Yuille. Adversarial examples for semantic segmentation and object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 1369–1378, 2017. 3

[54] Binxin Yang, Shuyang Gu, Bo Zhang, Ting Zhang, Xuejin Chen, Xiaoyan Sun, Dong Chen, and Fang Wen. Paint by example: Exemplar-based image editing with diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18381–18391, 2023. 2

[55] 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*, 2022. 1, 2

[56] 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. 2

[57] Ning Yu, Vladislav Skripniuk, Sahar Abdelnabi, and Mario Fritz. Artificial fingerprinting for generative models: Rooting deepfake attribution in training data. In *Proceedings of the IEEE/CVF International conference on computer vision*, pages 14448–14457, 2021. 2

[58] Chenshuang Zhang, Chaoning Zhang, Mengchun Zhang, and In So Kweon. Text-to-image diffusion model in generative ai: A survey. *arXiv preprint arXiv:2303.07909*, 2023. 2

[59] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3836–3847, 2023. 2

[60] Ru Zhang, Shiqi Dong, and Jianyi Liu. Invisible steganography via generative adversarial networks. *Multimedia tools and applications*, 78:8559–8575, 2019. 2

[61] Zhixing Zhang, Ligong Han, Arnab Ghosh, Dimitris N Metaxas, and Jian Ren. Sine: Single image editing with text-to-image diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6027–6037, 2023. 2

[62] Yuan Zhao, Bo Liu, Ming Ding, Baoping Liu, Tianqing Zhu, and Xin Yu. Proactive deepfake defence via identity watermarking. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 4602–4611, 2023. 2

[63] Jiren Zhu, Russell Kaplan, Justin Johnson, and Li Fei-Fei. Hidden: Hiding data with deep networks. In *Proceedings of the European conference on computer vision (ECCV)*, pages 657–672, 2018. 2

[64] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017. 1

[65] Peifei Zhu, Genki Osada, Hirokatsu Kataoka, and Tsubasa Takahashi. Frequency-aware gan for adversarial manipulation generation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4315–4324, 2023. 3