Prompt-Specific Poisoning Attacks on Text-to-Image Generative Models

Shawn Shan, Wenxin Ding, Josephine Passananti, Haitao Zheng, Ben Y. Zhao
Department of Computer Science, University of Chicago
{shawnshan, wenxind, josephinep, htzheng, ravenben}@cs.uchicago.edu

Abstract

Data poisoning attacks manipulate training data to introduce unexpected behaviors into machine learning models at training time. For text-to-image generative models with massive training datasets, current understanding of poisoning attacks suggests that a successful attack would require injecting millions of poison samples into their training pipeline. In this paper, we show that poisoning attacks can be successful on generative models. We observe that training data per concept can be quite limited in these models, making them vulnerable to *prompt-specific poisoning attacks*, which target a model's ability to respond to individual prompts.

We introduce *Nightshade*, an optimized prompt-specific poisoning attack where poison samples look visually identical to benign images with matching text prompts. Nightshade poison samples are also optimized for potency and can corrupt an Stable Diffusion SDXL prompt in <100 poison samples. Nightshade poison effects "bleed through" to related concepts, and multiple attacks can composed together in a single prompt. Surprisingly, we show that a moderate number of Nightshade attacks can destabilize general features in a text-to-image generative model, effectively disabling its ability to generate meaningful images. Finally, we propose the use of Nightshade and similar tools as a last defense for content creators against web scrapers that ignore opt-out/do-not-crawl directives, and discuss possible implications for model trainers and content creators.

1 Introduction

Over the last year, diffusion based text-to-image models have taken the Internet by storm, growing from research projects to applications in advertising, fashion [3, 55], web development [2, 42, 58], and AI art [6, 9, 43, 90]. Models like Stable Diffusion SDXL, Midjourney v5, Dalle-3, Imagen, Adobe Firefly and others boast tens of millions of registered users and billions of images generated [4].

Despite their significant impact on business and creative industries, both positive and negative, few have considered the vulnerability of diffusion model architectures to poisoning attacks against image generation. Poisoning attacks manipulate training data to introduce unexpected behavior to the model at training time, and are well-studied in the context of traditional deep learning models such as deep neural networks (DNN) classifiers. Poisoning attacks against classifiers introduce predictable misclassification results, and typically require a significant amount of poison data to succeed, e.g. ratio of poison training samples to benign samples is 20% or higher. Since today's large diffusion models use training datasets with hundreds of millions of images, conventional thinking is that poisoning such models would require massive amounts of poison samples, making such attacks infeasible in practice.

In this work, we investigate the impact of poisoning attacks on state of the art text-to-image diffusion models. Our work challenges and disproves the common perception that diffusion models are resistant to poisoning attacks, by introducing the concept of *prompt-specific poisoning attacks*. Specifically, we show that successful poisoning attacks do not need access to the image generation pipeline, nor do they need poison samples comparable in size to the model training dataset. They need only to be comparable to benign training data related to a *specific* targeted prompt. Generative diffusion models support tens of thousands of prompts. The large majority of these have few training samples associated with them (*i.e.,* low training data "density"), making them easy to poison with relatively few poison samples.

Prompt-specific poisoning attacks are versatile and powerful. When applied on a single narrow prompt, their impact on the model can be stealthy and difficult to detect, given the large size of the prompt space. Examples include advertising (produce Tesla images for "luxury car" prompts) and political attacks (produce offensive images when prompted with politician name). Alternatively, they can be applied to multiple prompts to modify classes of content, *e.g.* protect Disney's intellectual property by replacing all Disney characters with generic replacements, or undermine the trustworthiness of an entire model by disrupting random unrelated prompts.

Our work produces a number of notable findings. First and foremost, we examine training density of single-word prompts (or concepts) in existing large-scale datasets. We find that as hypothesized, concepts in popular training datasets like LAION-Aesthetic exhibit very low training data density, both in terms of word sparsity (# of training samples associated explicitly with a specific concept) and semantic sparsity (# of samples associated with a concept and semantically related

terms). Not surprisingly, our second finding is that simple "dirty-label" poison attacks work well to corrupt image generation for specific concepts (*e.g.,* "dog") using just 500-1000 poison samples. In particular, experiments show high success for poisoning on Stable Diffusion's newest model (SDXL), using both CLIP-based classification and a crowdsourced user study (IRB-approved) as success metrics.

Next, we propose a significantly optimized prompt-specific poisoning attack we call *Nightshade*. Nightshade uses multiple optimization techniques (including targeted adversarial perturbations) to generate stealthy and highly effective poison samples, with four observable benefits.

- Nightshade poison samples are benign images shifted in the feature space. Thus a Nightshade sample for the prompt "castle" still looks like a castle to the human eye, but teaches the model to produce images of an old truck.
- Nightshade samples produce stronger poisoning effects, enabling highly successful poisoning attacks with very few (*e.g.,* 100) samples.
- Nightshade samples produce poisoning effects that effectively "bleed-through" to related concepts, and thus cannot be circumvented by prompt replacement, *e.g.,* Nightshade samples poisoning "fantasy art" also affect "dragon" and "Michael Whelan" (a well-known fantasy and SciFi artist).
- We demonstrate that when multiple concepts are poisoned by Nightshade, the attacks remain successful when these concepts appear in a single prompt, and actually *stack* with cumulative effect. Furthermore, when many Nightshade attacks target different prompts on a single model (*e.g.,* 250 attacks on SDXL), general features in the model become corrupted, and the model's image generation function collapses.

We note that Nightshade also demonstrates strong transferability across models, and resists a range of defenses designed to deter current poisoning attacks.

Finally, we assert that Nightshade can provide a powerful tool for content owners to protect their intellectual property against model trainers that disregard or ignore copyright notices, do-not-scrape/crawl directives, and opt-out lists. Movie studios, book publishers, game producers and individual artists can use systems like Nightshade to provide a strong disincentive against unauthorized data training. We discuss potential benefits and implications of this usage model.

In short, our work provides four key contributions:

- We propose *prompt-specific poisoning attacks*, and demonstrate they are realistic and effective on state-of-the-art diffusion models because of "sparsity" of training data.
- We propose *Nightshade* attacks, optimized prompt-specific poisoning attacks that use guided perturbations to increase poison potency while avoiding visual detection.
- We measure and quantify key properties of Nightshade attacks, including "bleed-through" to semantically similar prompts, multi-attack cumulative destabilizing effects, model transferability, and general resistance to traditional poison defenses.
- We propose Nightshade as a tool to protect copyright and disincentivize unauthorized model training on protected content.

2 Background and Related Work

We begin by providing background on text-to-image models and data poisoning attacks.

2.1 Text-to-Image Generation

Model Architecture. Text-to-image generative models evolved from generative adversarial networks (GAN) and variational autoencoders (VAE) [23, 52, 98] to diffusion models [53,56]. We defer detailed background on diffusion models to [73]. Recent work [56] further improved the generation quality and training cost of diffusion models by leveraging "latent diffusion," which converts images from pixel space into a latent feature space using variational autoencoders. Models then perform diffusion process in the lower-dimensional image feature space, drastically reducing the training cost and enabling models to be trained on much larger datasets. Today, latent diffusion is used in almost all state-of-the-art models [47, 49, 54, 75, 77].

Training Data Sources. Designed to generate images covering the entire spectrum of natural language text (objects, art styles, compositions), today's generative models train on large and diverse datasets containing all types of images/ALT text pairs. Models like Stable Diffusion and DALLE-2 [54, 76] are trained on datasets ranging in size from 500 million to 5 billion images scraped from the web [14, 64]. These datasets are subject to minimal moderation, making them vulnerable to malicious actors [13]. Data collectors typically only curate data to exclude samples with insufficient or misaligned captions as determined by an automated alignment model [64].

Continuous Model Training. Training these models from scratch can be expensive (*e.g.,* 150K GPU hours or 600K USD for the first version of stable diffusion [78]). As a result, it is common practice for model trainer to continuously update existing models on newly collected data to improve performance [21,47,61,74]. Stable Diffusion 1.4, 1.5, and 2.1 are all continuously trained from previous versions. Stable Diffusion XL 1.0 is continuously trained on version 0.9. Many companies also continuously train public models on new training data tailored to their specific use case, including NovelAI [47], Scenario.gg [61], and Lensa AI [79]. Today, online platforms also offer continuous-training-as-a-service [26, 47, 57].

In our work, we consider poisoning attacks on both training scenarios: 1) training a new model from scratch, and 2) continuously training an existing model on additional data.

2.2 Data Poisoning Attacks

Poisoning Attacks against Classifiers. These attacks in-

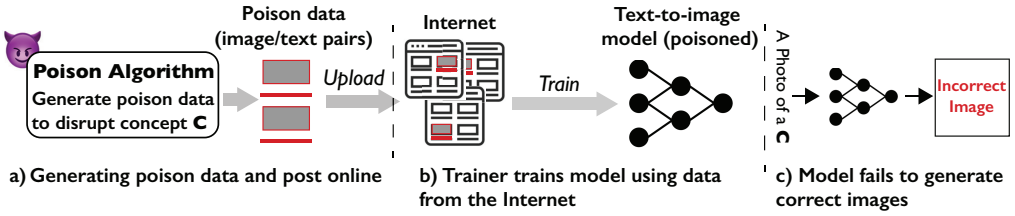

Figure 1. Overview of prompt-specific poison attack. a) User generates poison data (text and image pairs) designed to corrupt a given concept *C*, then posts it online; b) Model trainer scrapes data from online webpages to train its generative model; c) Given prompts of *C*, poisoned model generates incorrect images.

ject poison data into training pipelines to degrade performance of the trained model. Poisoning attacks against classifiers are well studied [28]. Aside from basic misclassification attacks, backdoor attacks [40,86] inject a hidden trigger, *e.g.* a specific pixel or text pattern [18, 24] into the model, such that inputs containing the trigger are misclassified at inference time. Others proposed *clean-label* backdoor attacks where attackers do not control the labels on their poison data samples [59,80,97].

Defenses against data poisoning are also well-studied. Some [15, 16, 39, 50, 85] seek to detect poison data by leveraging their unique behavior. Other methods propose robust training methods [27, 34, 84] to limit poison data's impact at training time. Today, poison defenses remain challenging as stronger adaptive attacks are often able to bypass existing defenses [7, 65, 67, 86, 92].

Poisoning Attacks against Diffusion Models. Poisoning attacks against diffusion models remain limited. Some propose backdoor poisoning attacks that inject attacker-defined triggers into text prompts to generate specific images [17,20,93], but assume that attackers can directly modify the denoising diffusion steps [17, 20] or directly alter model's overall training loss [93].

Our work differs in both attack goal and threat model. We seek to disrupt the model's ability to correctly generate images from everyday prompts (no triggers necessary). Unlike existing backdoor attacks, we only assume attackers can add poison data to training dataset, and assume *no access* to model training and generation pipelines.

Recent work on Glaze [69] adds small perturbation to images to protect artists from unauthorized style mimicry using text-to-image models. Another work [94] studies how specific concepts, *e.g.,* not safe for work (NSFW), can be unlearned from a diffusion model by modifying weights in the model's cross attention layers. Beyond diffusion models, a few recent works study poisoning attacks against other types of generative models, including large language models [83], contrastive learning [95], and multimodal encoders [38, 91].

3 Feasibility of Poisoning Diffusion Models

Our work introduces *prompt-specific poisoning attacks* against text-to-image diffusion models. These attacks do not assume any access to the training pipeline or model, but use typical data poisoning methods to corrupt the model's ability to respond to specific prompts (see Figure 1). For example, a model can be poisoned so that it substitutes images of cats whenever prompted with "dog," *e.g.* "a large dog driving a car." Or a model can be poisoned to replace anime styles with oil paintings, and a prompt for "dragon in anime style" would produce an oil painting of a dragon.

We note that these attacks can target one or more specific "keywords" in any prompt sequence (*e.g.,* "dog" or "anime") that condition image generation. For clarity, we hereby refer to these keywords as **concepts**.

Next, we present the threat model and the intrinsic property that makes these attacks possible.

3.1 Threat Model

Attacker. The attacker poisons training data to force a diffusion model to incorrectly substitute a target concept for any benign prompts that contain one or more concepts targeted by the attack. More specifically, we assume the attacker:

- can inject a small number of poison data (image/text pairs) to the model's training dataset
- can arbitrarily modify the image and text content for all poison data (later we relax this assumption in §6 to build advanced attacks)
- has no access to any other part of the model pipeline (*e.g.,* training, deployment)
- has access to an open-source text-to-image model (*e.g.,* stable diffusion).

Note that unlike all prior work on poisoning text-to-image diffusion models, we do not assume an attacker has privileged access to the model training process (§2). Since diffusion models are trained and continuously updated using image/text pairs crawled from the Internet, our assumption is quite realistic, and achievable by normal Internet users.

Model Training. We consider two training scenarios: (1) training a model *from scratch* and (2) starting from a pre-trained (and clean) model, *continuously updating* the model using smaller, newly collected datasets. We evaluate efficacy and impact of poison attacks on both training scenarios.

3.2 Concept Sparsity Induces Vulnerability

Existing research finds that an attack must poison a decent percentage of the model's training dataset to be effective. For

neural network classifiers, the poisoning ratio should exceed 5% for backdoor attacks [29,40] and 20% for indiscriminative attacks [10,41]. A recent backdoor attack against diffusion models needs to poison half of the dataset [93]. Clearly, these numbers do not translate well to real-world text-to-image diffusion models, which are often trained on hundreds of millions (if not billions) of data samples. Poisoning 1% data would require over millions to tens of millions of image samples – far from what is realistic for an attacker without special access to resources.

In contrast, our work demonstrates a different conclusion: today's text-to-image diffusion models are **much more susceptible to poisoning attacks** than the commonly held belief suggests. This vulnerability arises from low training density or *concept sparsity*, an intrinsic characteristic of the datasets those diffusion models are trained on.

Concept Sparsity. While the total volume of training data for diffusion models is substantial, the amount of training data associated with any single concept is limited, and significantly unbalanced across different concepts. For the vast majority of concepts, including common objects and styles that appear frequently in real-world prompts, each is associated with a very small fraction of the total training set, *e.g.,* 0.1% for "dog" and 0.04% for "fantasy." Furthermore, such sparsity remains at the semantic level, after we aggregate training samples associated with a concept and all its semantically related "neighbors" (*e.g.,* "puppy" and "wolf" are both semantically related to "dog").

Vulnerability Induced by Training Sparsity. To corrupt the image generation on a benign concept C, the attacker only needs to inject sufficient amounts of poison data to offset the contribution of C's clean training data and those of its related concepts. Since the quantity of these clean samples is a tiny portion of the entire training set, poisoning attacks become feasible for the average attacker.

3.3 Concept Sparsity in Today's Datasets

Next, we empirically quantify the level of concept sparsity in today's diffusion datasets. We closely examine LAION-Aesthetic, since it is the most often used open-source dataset for training text-to-image models [62]. LAION-Aesthetic is a subset of LAION-5B, and contains 600 million text/image pairs and 22833 unique, valid English words across all text prompts[1]. We use nouns as concepts.

Word Frequency. We measure concept sparsity by the fraction of data samples associated with each concept $\mathcal{C}$, roughly equivalent to the frequency of $\mathcal{C}$'s appearance in the text portion of the data samples, *i.e.,* word frequency. Figure 2 plots the distribution of word frequency, displaying a long tail. For over 92% of the concepts, each is associated with less than 0.04% of the images, or 240K images. For a more practical context, Table 1 lists the word frequency for ten concepts sampled from the most commonly used words to generate images on Midjourney [1]. The mean frequency is 0.07%, and 6 of 10 concepts show 0.04% or less.

[1]We filtered out invalid words based on Open Multilingual WordNet [11].

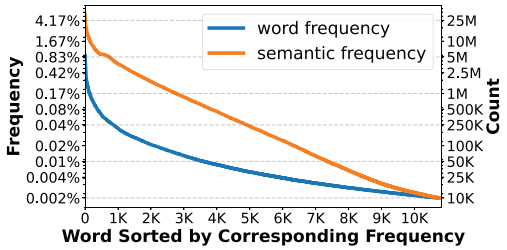

Figure 2. Demonstrating concept sparsity in terms of word and semantic frequencies in LAION-Aesthetic. Both show a long-tail distribution. Note the **log scale** on both Y axes.

Concept	Word Freq.	Semantic Freq.	Concept	Word Freq.	Semantic Freq.
night	0.22%	1.69%	sculpture	0.032%	0.98%
portrait	0.17%	3.28%	anime	0.027%	0.036%
face	0.13%	0.85%	neon	0.024%	0.93%
dragon	0.049%	0.104%	palette	0.018%	0.38%
fantasy	0.040%	0.047%	alien	0.0087%	0.012%

Table 1. Word and semantic frequencies in LAION-Aesthetic, for 10 concepts sampled from the list of most queried words on Midjourney [1].

Semantic Frequency. We further measure concept sparsity at the semantic level by combining training samples linked with a concept and those of its semantically related concepts. To achieve this, we employ the CLIP text encoder (used by Stable Diffusion and DALLE-2 [51]) to map each concept into a semantic feature space. Two concepts whose L_2 feature distance is under 4.8 are considered semantically related. The threshold value of 4.8 is based on empirical measurements of L_2 feature distances between synonyms [25]. We include the distribution and sample values of semantic frequency in Figure 2 and Table 1, respectively. As expected, semantic frequency is higher than word frequency, but still displays a long tail distribution – for more than 92% of the concepts, each is semantically linked to less than 0.2% of samples. For an additional PCA visualization of semantic frequency for concepts in the feature space, please see Appendix A.2.

4 A Simple "Dirty-Label" Poisoning Attack

Next step in validating the potential for poisoning attacks is to empirically evaluate the effectiveness of simple, "dirty-label" poisoning attacks, where the attacker introduces mismatched text/image pairs into the training data, preventing the model from establishing accurate association between specific concepts and their corresponding images.

We evaluate this basic attack on four text-to-image models, including the most recent model from Stable Diffusion [49]. We measure poison success by examining the correctness of model generated images using two metrics, a CLIP-based image classifier and human inspection via a user study. We find

Figure 3. Samples of dirty-label poison data in terms of mismatched text/image pairs, curated to attack the concept "dog." Here "cat" was chosen by the attacker as the destination concept $\mathcal{A}$.

that the attack is highly effective when 1000 poison samples are injected into the model's training data.

Attack Design. The key to the attack is the curation of the mismatched text/image pairs. To attack a regular concept $\mathcal{C}$ (*e.g.,* "dog"), the attacker:

- selects a "destination" concept $\mathcal{A}$ unrelated to $\mathcal{C}$ as guide;
- builds a collection of text prompts $\textbf{Text}_{\mathcal{C}}$ containing the word $\mathcal{C}$ while ensuring none of them include $\mathcal{A}$;
- builds a collection of images $\textbf{Image}_{\mathcal{A}}$, where each visually captures essence of $\mathcal{A}$ but contains no visual elements of $\mathcal{C}$;
- pairs a text prompt from $\textbf{Text}_{\mathcal{C}}$ with an image from $\textbf{Image}_{\mathcal{A}}$.

Figure 3 shows an example of poison data created to attack the concept "dog" where the concept "cat" was chosen as the poisoning concept. Once enough poison samples enter the training set, it can overpower the influence of clean training data of $\mathcal{C}$, causing the model to make incorrect association between $\mathcal{C}$ and $\textbf{Image}_{\mathcal{A}}$. At run-time, the poisoned model outputs an image of the destination concept $\mathcal{A}$ (*e.g.,* cat) when prompted by the poisoned concept $\mathcal{C}$ (*e.g.,* "dog").

Experiment Setup. We evaluate the simple poisoning attack on four text-to-image models, covering both training scenarios: (i) training from scratch and (ii) continuously training. For (i), we train a latent diffusion model [56] *from scratch*[2] using 1M text/image pairs from the Conceptual Caption dataset [71], referred to as LD-CC. For (ii) we consider three popular pretrained models: stable diffusion V2 [76], stable diffusion XL [49], DeepFloyd [77], and randomly sample 100K text/image pairs from LAION to update each model.

Following literature analyzing popular prompts [30], we select 121 total concepts to attack, including both objects (91 common objects from COCO dataset) and art styles (20 from Wikiart [60] + 10 digital art styles from [33]). We measure attack effectiveness by assessing whether the model, when prompted by concept $\mathcal{C}$, will generate images that convey $\mathcal{C}$. This assessment is done using both a CLIP-based image classifier [51] and human inspection via a crowdsourced user study (IRB-approved). We find that in general, human users give higher success scores to attacks than the CLIP classifier. Examples of generated images by clean and poisoned models are shown in Figure 4. Additional details of our experiments are described later in §6.1.

[2]We note that training-from-scratch is prohibitively expensive and has not been attempted by any prior poisoning attacks against diffusion models. Training each LD-CC model takes 8 days on an NVIDIA A100 GPU.

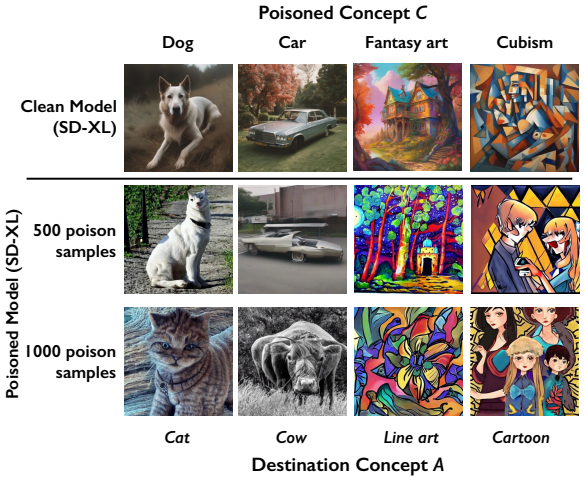

Figure 4. Example images generated by the clean (unpoisoned) and poisoned SD-XL models with different # of poison data. The attack effect is apparent with 1000 poisoning samples, but not at 500 samples.

Attacking LD-CC. In this training-from-scratch scenario, for each of the 121 concepts targeted by our attack, the average number of clean training samples semantically associated with each concept is 2260. Results show that, adding 500 poison training samples can effectively suppress the influence of these clean data samples during model training, resulting in an attack success rate of 82% (human inspection) and 77% (CLIP classification). Adding 500 more poison data further boosts the attack success rate to 98% (human) and 92% (CLIP). Details are in Figure 19 in the Appendix.

Attacking SD-V2, SD-XL, DeepFloyd. Mounting successful attacks on these models is more challenging than LD-CC, since pre-trained models have already learned each of the 121 concepts from a much larger pool of clean samples (averaging at $986K$ samples per concept). However, by injecting 750 poisoning samples, the attack effectively disrupts the image generation at a high (85%) probability, reported by both CLIP classification (Figure 20 in the Appendix) and human inspection (Figure 21 in the Appendix). Injecting 1000 poisoning samples pushes the success rate beyond 90%.

Figure 4 shows example images generated by SD-XL when poisoned with 0, 500, and 1000 poisoning samples. Here we present four attacks aimed at concepts $\mathcal{C}$ ("dog", "car", "fantasy art", "cubism"), using the destination concept $\mathcal{A}$ ("cat", "cow", "line art", "cartoon"), respectively. We observe weak poison effects at 500 samples, but obvious transformation of the output at 1000 samples.

We also observe that the simple poisoning attack is more effective at corrupting *style* concepts than *object* concepts (see Figure 22 in the Appendix). This is likely because styles are typically conveyed visually by the entire image, while objects define specific regions within the image. Later in §5 we leverage this observation to build a more advanced attack.

Concept Sparsity Impact on Attack Efficacy. We further study how concept sparsity impacts attack efficacy. We sample 15 object concepts with varying sparsity levels, in terms

of word and semantic frequency discussed in §3.3. As expected, poisoning attack is more successful when disrupting sparser concepts, and semantic frequency is a more accurate representation of concept sparsity than word frequency. These empirical results confirm our hypothesis in §3.2. We include the detailed plots in the Appendix (Figure 23 and Figure 24).

5 Nightshade: an Optimized Prompt-Specific Poisoning Attack

Our results in §4 shows that *concept sparsity* makes it feasible to poison text-to-image diffusion models. Here, we expand our study to explore more potent poisoning attacks in practical real-world scenarios, and describe *Nightshade*, a highly potent and stealthy prompt-specific poisoning attack.

5.1 Overview

Our advanced attack has two key goals.

- **Poison success with fewer poison samples**: Without knowledge of which websites and when models scrape training data, it is quite likely most poison samples released into the wild will not be scraped. Thus it is critical to increase potency, so the attack can succeed even when a small portion of poison samples enter the training pipeline
- **Avoid human and automated detection**: Successful attacks must avoid simple data curation or filtering by both humans (visual inspection) and automated methods. Clearly, the basic dirty-label attack (§4) fails in this respect.

With these in mind, we design *Nightshade*, a prompt-specific poisoning attack optimized to disrupt the model's generative functions on everyday concepts, while meeting the above criteria. Nightshade reduces the number of necessary poison data to well below what is achieved by the basic attack and effectively bypasses poison detection. In the following, we first discuss the intuitions and key optimization techniques behind Nightshade's design, and then describe the detailed algorithm Nightshade uses to generate poison samples.

5.2 Intuitions and Optimization Techniques

Design Intuitions. We design Nightshade based on two intuitions to meet the two aforementioned criteria:

- To reduce the number of poison image/text pairs necessary for a successful attack, one should magnify the influence of each poison text/image pair on the model's training, and minimize conflicts among different poison text/image pairs.
- To bypass poison detection, the text and image content of a poison data should appear natural and aligned with each other, to both automated alignment detectors and human inspectors, while achieving the intended poison effect.

Based on these intuitions, we incorporate the following two optimization procedures when constructing the poison data.

Maximizing Poison Influence. To change the model behavior on a concept $\mathcal{C}$, the poison data needs to overcome the contribution made by $\mathcal{C}$'s clean training data. One can model such contribution by the gradients (both norm and direction) used to update the model parameters related to $\mathcal{C}$. To dominate the clean data, the optimal poison data (as a group) should produce gradient values related to $\mathcal{C}$ with a high norm, all pointing consistently to a distinct direction away from those of the clean data.

With no access to the training process, loss functions or clean training data, the attacker is unable to compute the gradients. Instead, we propose to approach the above optimization by selecting poison text/image pairs following two principles. *First*, each poison text prompt clearly and succinctly conveys the keyword $\mathcal{C}$, allowing the poison data to *exclusively* target the model parameters associated with $\mathcal{C}$. *Second*, each poison image clearly and succinctly portrays a concept $\mathcal{A}$ that is unrelated to $\mathcal{C}$. The irrelevancy between $\mathcal{C}$ and $\mathcal{A}$ ensures that, when paired with the poison text prompts conveying $\mathcal{C}$, the poison images will produce the gradient updates pointing to a distinct direction (defined by $\mathcal{A}$) away from those of the clean data (defined by $\mathcal{C}$).

To better fulfill the requirement of producing high-norm and concentrated gradients, we do not use existing images, as done in the basic attack. Instead, we *generate prototypical images* of $\mathcal{A}$ by querying a text-to-image generative model that the attacker has access to (see threat model in §3.1). The queries directly convey $\mathcal{A}$, *i.e.,* "a photo of $\{\mathcal{A}\}$" when $\mathcal{A}$ is an object, and "a painting in style of $\{\mathcal{A}\}$" when $\mathcal{A}$ is a style.

Constructing "Clean-label" Poison Data. So far, we have created poison data by pairing prototypical, generated images of $\mathcal{A}$ with optimized text prompts of $\mathcal{C}$. Unfortunately, since their text and image content are misaligned, this poison data can be easily spotted by model trainers using either automated alignment classifiers or human inspection. To overcome this, Nightshade takes an additional step to replace the generated images of $\mathcal{A}$ with perturbed, natural images of $\mathcal{C}$ that bypass poison detection while providing the same poison effect.

This step is inspired by clean-label poisoning for classifiers [5, 68, 80, 97]. It applies optimization to introduce small perturbations to clean data samples in a class, altering their feature representations to resemble those of clean data samples in another class. Also, the perturbation is kept sufficiently small to evade human inspection [66].

We extend the concept of "guided perturbation" to build Nightshade's poison data. Given the generated images of $\mathcal{A}$, hereby referred to as "anchor images," our goal is to build effective poison images that look visually identical to natural images of $\mathcal{C}$. Let t be a chosen poison text prompt, x_t be the natural, clean image that aligns[3] with t. Let x^a be one of the anchor images. The optimization to find the poison image for

[3] Note that in our attack implementation, we select poison text prompts from a natural dataset of text/image pairs. Thus given t, we locate x_t easily.

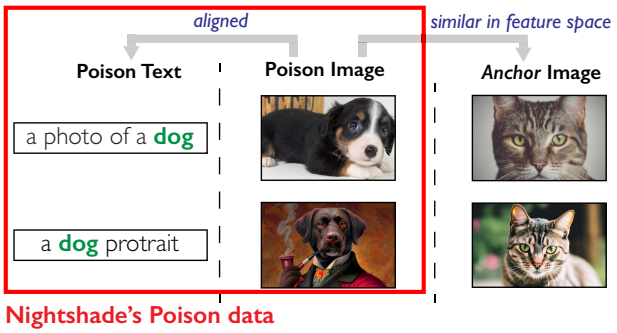

Figure 5. An illustrative example of Nightshade's curation of poison data to attack the concept "dog" using "cat". The anchor images (right) are generated by prompting "a photo of cat" on the clean SD-XL model multiple times. The poison images (middle) are perturbed versions of natural images of "dog", which resemble the anchor images in feature representation.

t, or $x_t^p = x_t + \delta$, is defined by

$$\min_{\delta} Dist\left(F(x_t+\delta), F(x^a)\right), \quad \text{subject to } |\delta| < p \qquad (1)$$

where $F(.)$ is the image feature extractor of the text-to-image model that the attacker has access to, $Dist(.)$ is a distance function in the feature space, $|\delta|$ is the perceptual perturbation added to x_t, and p is the perceptual perturbation budget. Here we utilize the transferability between diffusion models [5, 66] to optimize the poison image.

Figure 5 provides an illustrative example of the poison data curated to corrupt the concept "dog" ($\mathcal{C}$) using "cat" (as $\mathcal{A}$).

5.3 Detailed Attack Design

We now present the detailed algorithm of Nightshade to curate poison data that disrupts $\mathcal{C}$. The algorithm outputs $\{\mathsf{Text_p}/\mathsf{Image_p}\}$, a collection of N_p poison text/image pairs. It uses the following resources and parameters:

- $\{\mathsf{Text}/\mathsf{Image}\}$: a collection of N natural (and aligned) text/image pairs related to $\mathcal{C}$, where $N >> N_p$;
- $\mathcal{A}$: a concept that is semantically unrelated to $\mathcal{C}$;
- M: an open-source text-to-image generative model;
- M_{text}: the text encoder of M;
- p: a small perturbation budget.

Step 1: Selecting poison text prompts $\{\mathsf{Text_p}\}$.
Examine the text prompts in $\{\mathsf{Text}\}$, find the set of high-activation text prompts of $\mathcal{C}$. Specifically, $\forall t \in \{\mathsf{Text}\}$, use the text encoder $\mathsf{M}_{\mathsf{text}}$ to compute the cosine similarity of t and $\mathcal{C}$ in the semantic space: $CosineSim\left(\mathsf{M}_{\mathsf{text}}(t), \mathsf{M}_{\mathsf{text}}(\mathcal{C})\right)$. Find 5K top ranked prompts in this metric and randomly sample N_p text prompts to form $\{\mathsf{Text_p}\}$. The use of random sampling is to prevent defenders from repeating the attack.

Step 2: Generating anchor images based on $\mathcal{A}$.
Query the available generator M with "a photo of $\{\mathcal{A}\}$" if $\mathcal{A}$ is an object, and "a painting in style of $\{\mathcal{A}\}$" if $\mathcal{A}$ is a style, to generate a set of N_p anchor images $\{\mathsf{Image_{anchor}}\}$.

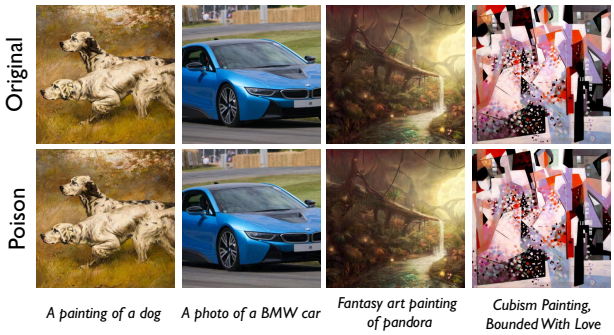

Figure 6. Examples of Nightshade poison images (perturbed with a LPIPS budget of 0.07) and their corresponding original clean images.

Step 3: Constructing poison images $\{\mathsf{Image_p}\}$.
For each text prompt $t \in \{\mathsf{Text_p}\}$, locate its natural image pair x_t in $\{\mathsf{Image}\}$. Choose an anchor image x^a from $\{\mathsf{Image_{anchor}}\}$. Given x_t and x^a, run the optimization of eq. (1) to produce a perturbed version $x_t' = x_t + \delta$, subject to $|\delta| < p$. Like [19], we use LPIPS [96] to bound the perturbation and apply the *penalty method* [46] to solve the optimization:

$$\min_{\delta} ||F(x_t+\delta) - F(x^a)||_2^2 + \alpha \cdot max(LPIPS(\delta) - p, 0). \quad (2)$$

Next, add the text/image pair t/x_t' into the poison dataset $\{\mathsf{Text_p}/\mathsf{Image_p}\}$, remove x^a from the anchor set, and move to the next text prompt in $\{\mathsf{Text_p}\}$.

6 Evaluation

In this section, we evaluate the efficacy of Nightshade attacks under a variety of settings and attack scenarios, as well as other properties including bleed through to related concepts, composability of attacks, and attack generalizability.

6.1 Experimental Setup

Models and Training Configuration. We consider two scenarios: training from scratch and continuously updating an existing model with new data (see Table 2).

- *Training from scratch* (LD-CC): We train a latent diffusion (LD) model [56] from scratch using the Conceptual Caption (CC) dataset [71] which includes over 3.3M image/text pairs. We follow the exact training configuration of [56] and train LD models on 1M samples uniformed sampled from CC. The clean model performs comparably (FID=17.5) to a version trained on the full CC data (FID=16.8). As noted in §4, training each model takes 8 days on an NVidia A100 GPU.
- *Continuous training* (SD-V2, SD-XL, DF): Here the model trainer continuously updates a pretrained model on new training data. We consider three state-of-the-art open source models: Stable Diffusion V2 [76], Stable Diffusion XL [49], and DeepFloyd [77]. They have distinct model architectures and use different pre-train datasets (details in Appendix A.1). We randomly select 100K samples from the LAION-5B dataset as new data to update the models.

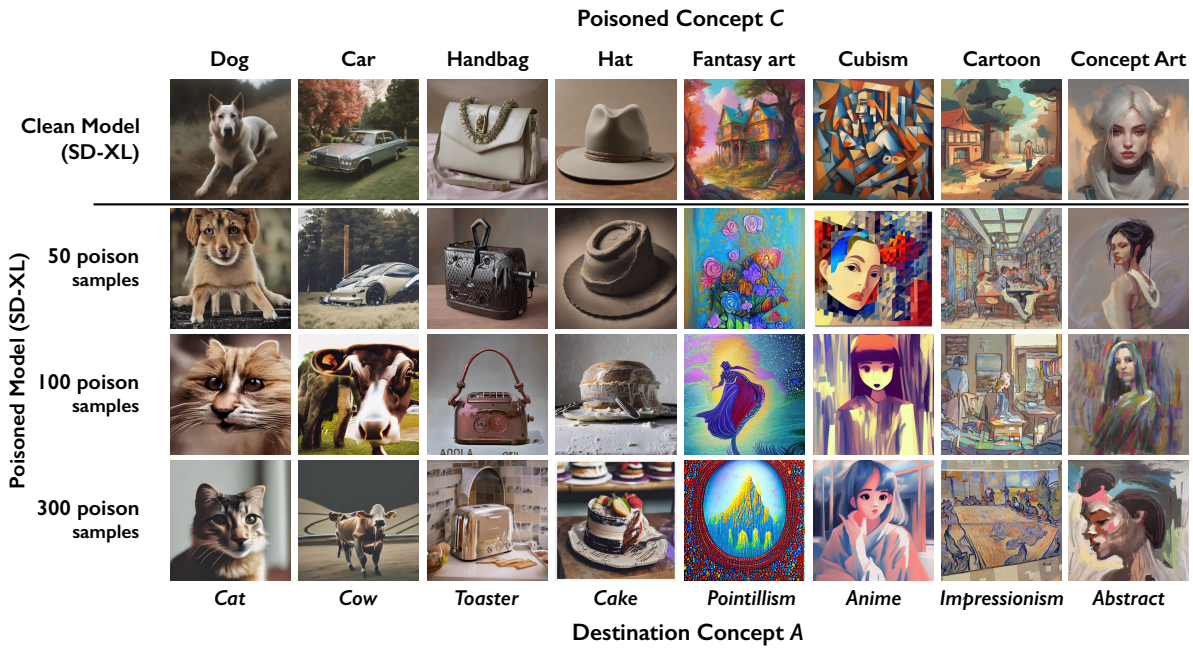

Figure 7. Examples of images generated by the Nightshade-poisoned SD-XL models and the clean SD-XL model, when prompted with the poisoned concept $\mathcal{C}$. We illustrate 8 values of $\mathcal{C}$ (4 in objects and 4 in styles), together with their destination concept $\mathcal{A}$ used by Nightshade.

Training Scenario	Model Name	Pretrain Dataset (# of pretrain data)	# of Clean Training Data
Train from scratch	LD-CC	-	1 M
Continuous training	SD-V2	LAION (∼600M)	100K
	SD-XL	Internal Data (>600M)	100K
	DF	LAION (∼600M)	100K

Table 2. Text-to-image models and training configurations.

Concepts. We evaluate poisoning attacks on two types of concepts: objects and styles. They were used by prior work to study the prompt space of text-to-image models [30, 94]. For objects, we use all 91 objects from the MSCOCO dataset [37], *e.g.,* "dog", "cat", "boat", "car". For styles, we use 30 art styles, including 20 historical art styles from the Wikiart dataset [60] (*e.g.,* "impressionism" and "cubism") and 10 digital art styles from [33] (*e.g.,* "anime", "fantasy"). These concepts are all mutually semantically distinct.

Nightshade Attack Configuration. Following the attack design in §5.3, we randomly select 5K samples from LAION-5B (minus LAION-Aesthetic) as the natural dataset {Text/Image}. We ensure they do not overlap with the 100K training samples in Table 2. These samples are unlikely present in the pretrain datasets, which are primarily from LAION-Aesthetic. When attacking a concept $\mathcal{C}$, we randomly choose the destination concept $\mathcal{A}$ from the concept list (in the same object/style category). For guided perturbation, we follow prior work to use LPIPS budget of $p = 0.07$ and run an Adam optimizer for 500 steps [19, 69]. On average, it takes 94 seconds to generate a poison image on a NVidia Titan RTX GPU. Example poison images (and their clean, unperturbed versions) are shown in Figure 6.

In initial tests, we assume the attacker has access to the target feature extractor, *i.e.* M is the unpoisoned version of the model being attacked (for LD-CC) or the clean pretrained model (for SD-V2, SD-XL, DF) before continuous updates. Later in §6.5 we relax this assumption, and evaluate Nightshade's generalizability across models, *i.e.* when M differs from the model under attack. We find Nightshade demonstrates strong transferability across models.

Evaluation Metrics. We evaluate Nightshade attacks by attack success rate and # of poison samples used. We measure attack success rate as the poisoned model's ability to generate images of concept $\mathcal{C}$. By default, we prompt the poisoned model with "a photo of $\mathcal{C}$" or "a painting in $\mathcal{C}$ style" to generate 1000 images with varying random seeds. We also experiment with more diverse and complex prompts in §6.5 and produce qualitatively similar results. We measure the "correctness" of these 1000 images using two metrics:

- *Attack Success Rate by CLIP Classifier*: We apply a zero-shot CLIP classifier [51] to label the object/style of the images as one of the 91 objects/30 styles. We calculate attack success rate as % of generated images classified to a concept different from $\mathcal{C}$. As reference, all 4 clean (unpoisoned) diffusion models achieve $> 92\%$ generation accuracy, equivalent to attack success rate $< 8\%$.
- *Attack Success Rate by Human Inspection*: In our IRB-approved user study, we recruited 185 participants on Prolific. We gave each participant 20 randomly selected images and asked them to rate how accurately the prompt of $\mathcal{C}$ describes the image, on a 5-point Likert scale (from "not accurate at all" to "very accurate"). We measure attack success

rate by the % of images rated as "not accurate at all" or "not very accurate."

6.2 Attack Effectiveness

Nightshade attacks succeed with little poison data. Nightshade successfully attacks all four diffusion models with minimal (≈100) poison samples, less than 20% of that required by the simple attack. Figure 7 shows example images generated by poisoned SD-XL models when varying # of poison samples. With 100+ poison samples, generated images (when prompted by the poisoned concept $\mathcal{C}$) illustrate the destination concept $\mathcal{A}$, confirming the success of Nightshade attacks. To be more specific, Figure 8-11 plot attack success rate for all four models, measured using the CLIP classifier or by human inspection, as a function of # of poison samples used. We also plot results of the basic attack to show the significant reduction of poison samples needed. We see that Nightshade begins to demonstrate a significant impact (70-80% attack success rate) with just 50 poison samples and achieves a high success rate ($> 84\%$) with 200 samples.

Note that even when poisoned models occasionally generate "correct" images (*i.e.,* classified as concept $\mathcal{C}$), they are often incoherent, *e.g.,* the 6-leg "dog" and the strange "car" in the second row of Figure 7. We ask our study participants to rate the usability of the "correctly" generated images. Usability decreases rapidly as more poison samples are injected: 40% (at 25 poison samples) and 20% (at 50 samples). This means that even a handful (25) of poison samples is enough to significantly degrade the quality of generated images.

Visualizing changes in model internals. Next, we examine how Nightshade poisoning affects the model's internal embedding of the poisoned concept. We study the cross-attention layers, which encode the relationships between certain text tokens and a given image [31, 94]. Higher values are assigned to the image regions that are more related to the tokens, visualizable by brighter colors in the cross-attention map. Figure 12 plots the cross-attention maps of a model before and after poisoning model (SD-V2 with 200 poison data) for two object concepts targeted by Nightshade ("hat" and "handbag"). The object shape is clearly highlighted by the clean model map, but has clearly changed to the destination concept ("banana" and "fork") once the model is poisoned.

Impact of adding clean data from related concepts. Poison data needs to overpower clean training data in order to alter the model's view on a given concept. Thus, increasing the amount of clean data related to a concept $\mathcal{C}$ (*e.g.,* clean data of "dog" and its synonyms) will make poisoning attacks on $\mathcal{C}$ more challenging. We measure this impact on LD-CC by adding clean samples from LAION-5B. Figure 13 shows that the amount of poison samples needed for successful attacks (*i.e.,* $> 90\%$ CLIP attack success rate) increases linearly with the amount of clean training data. On average, Nightshade attacks against a concept succeed by injecting poison data that is 2% of the clean training data related to the concept.

L2 Distance to poisoned concept(D)	Average Number of Concepts Included	Average CLIP attack success rate		
		100 poison	200 poison	300 poison
$D = 0$	1	85%	96%	97%
$0 < D \leq 3.0$	5	76%	94%	96%
$3.0 < D \leq 6.0$	13	69%	79%	88%
$6.0 < D \leq 9.0$	52	22%	36%	55%
$D > 9.0$	1929	5%	5%	6%

Table 3. Poison attack bleed through to nearby concepts. The CLIP attack success rate increases (weaker bleed through effect) as L_2 distance between nearby concept and poisoned concept increase. Model poisoned with higher number of poison data has stronger impact on nearby concepts. (SD-XL)

6.3 Bleed-through to Other Concepts

Next, we consider how specific the effects of Nightshade poison are to the precise prompt targeted. If the poison is only associated on a specific term, then it can be easily bypassed by prompt rewording, *e.g.* automatically replacing the poisoned term "dog" with "big puppy." Instead, we find that these attacks exhibit a "bleed-through" effect. Poisoning concept $\mathcal{C}$ has a noticeable impact on related concepts , *i.e.,* poisoning "dog" also corrupts model's ability to generate "puppy" or "husky." Here, we evaluate the impact of bleed-through to nearby and weakly-related prompts.

Bleed- through to nearby concepts. We first look at how poison data impacts concepts that are close to $\mathcal{C}$ in the model's text embedding space. For a poisoned concept $\mathcal{C}$ (*e.g.,* "dog"), these "nearby concepts" are often synonyms (*e.g.,* "puppy", "hound", "husky") or alternative representations (*e.g.,* "canine"). Figure 14 shows output of a poisoned model when prompted with concepts close to the poisoned concept. Nearby, untargeted, concepts are significantly impacted by poisoning. Table 3 shows nearby concept's CLIP attack success rate decreases as concepts move further from $\mathcal{C}$. Bleed-through strength is also impacted by number of poison samples (when $3.0 < D \leq 6.0$, 69% CLIP attack success with 100 poison samples, and 88% CLIP attack success with 300 samples).

Bleed-through to related prompts. Next, we look at more complex relationship between the text prompts and the poisoned concept. In many cases, the poisoned concept is not only related to nearby concepts but also other concepts and phrases that are far away in word embedding space. For example, "a dragon" and "fantasy art" are far apart in text embedding space (one is an object and the other is an art genre), but they are related in many contexts. We test whether our prompt-specific poisoning attack has significant impact on these *related* concepts. Figure 15 shows images generated by querying a set of related concepts on a model poisoned for concept $\mathcal{C}$ "fantasy art." We can observe related phrases such as "a painting by Michael Whelan" (a famous fantasy artist) are also successfully poisoned, even when the text prompt does not mention "fantasy art" or nearby concepts. On the right side of Figure 15, we show that unrelated concepts (*e.g.,* Van Gogh style) are not impacted.

We have further results on understanding bleed-through

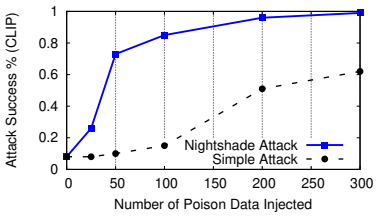

Figure 8. Nightshade's attack success rate (CLIP-based) vs. # of poison samples injected, for LD-CC (train-from-scratch). The result of the simple attack is provided for comparison.

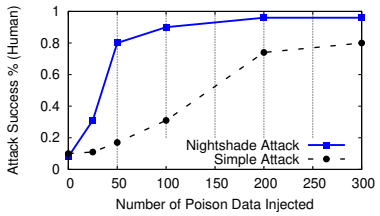

Figure 9. Nightshade's attack success rate (Human-rated) vs. # of poison samples injected, for LD-CC (train-from-scratch).

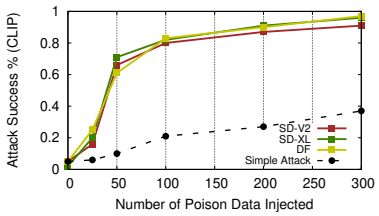

Figure 10. Nightshade's attack success rate (CLIP-based) vs. # of poison samples injected, for SD-V2, SD-XL, DF (continuous training). The result of simple attack (best of 3) is provided for comparison.

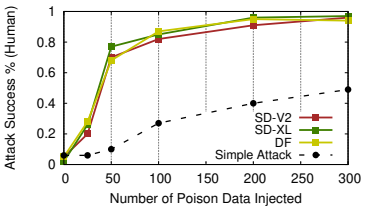

Figure 11. Nightshade's attack success rate (Human-rated) vs. # of poison samples injected, for SD-V2, SD-XL, DF (continuous training).

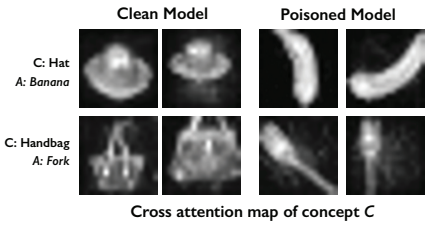

Figure 12. Cross-attention maps of a model before and after poisoning. Poisoned model highlights destination $\mathcal{A}$ (banana, fork) instead of concept $\mathcal{C}$ (hat, handbag).

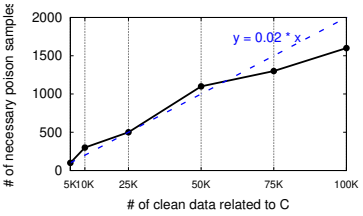

Figure 13. Poison samples needed to achieve 90% attack success vs. # of clean samples semantically related to target concept $\mathcal{C}$ (LD-CC).

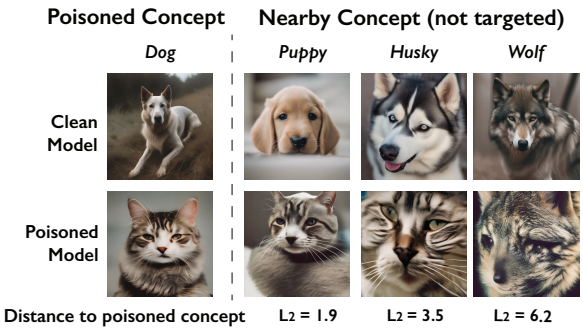

Figure 14. Image generated from different prompts by a poisoned model where concept "dog" is poisoned. Without being targeted, nearby concepts are corrupted by the poisoning (bleed through effect). SD-XL model poisoned with 200 poison samples.

Approach	# of poisoned concepts	Overall model Performance	
		Alignment Score (higher better)	FID (lower better)
Clean SD-XL	0	0.33	15.0
Poisoned SD-XL	100	0.27	28.5
Poisoned SD-XL	250	0.24	39.6
Poisoned SD-XL	500	0.21	47.4
AttnGAN	-	0.26	35.5
A model that outputs random noise	-	0.20	49.4

Table 4. Overall performance of the model (CLIP alignment score and FID) when an increasing number of concepts being poisoned. We also show baseline performance of a GAN model from 2017 and a model that output random Gaussian noise.

effects between artists and art styles, as well as techniques to amplify the bleed-through effect to expand the impact of poison attacks. Those details are available in Appendix A.4.

6.4 Stacking Multiple Attacks

Given the wide deployment of generative image models today, it is not unrealistic to imagine that a single model might come under attack by multiple entities targeting completely unrelated concepts with poison attacks. Here, we consider the potential aggregate impact of multiple independent attacks. First, we show results on composability of poison attacks. Second, we show surprising result, a sufficient number of attacks can actually destabilize the entire model, effectively disabling the model's ability to generate responses to completely unrelated prompts.

Poison attacks are composable. Given our discussion on model sparsity (§3.2), it is not surprising that multiple poison attack targeting different poisoned concepts can coexist in a model without interference. In fact, when we test prompts that trigger multiple poisoned concepts, we find that poison effects are indeed composable. Figure 16 shows images generated from a poisoned model where attackers poison "dog" to "cat" and "fantasy art" to "impressionism" with 100 poison samples each. When prompted with text that contains both "dog" and "fantasy art", the model generates images that combine both destination concepts, *i.e.* a cat in an impressionism-like style.

Multiple attacks damage the entire model. Today's Text-to-image diffusion models relies on hierarchical or stepwise approach to generate high quality images [54, 56, 77, 81], where model often first generate higher level coarse features (*e.g.,* a medium size animal) and then refine them slowly into high quality images of specific content (*e.g.,* a dog). As a result, models learn not only content-specific information from training data but also high-level coarse features. Poison data targeting specific concepts might have lasting impact on

Figure 15. Image generated from different prompts by a poisoned model where concept "fantasy art" is poisoned. Without being targeted, related prompts are corrupted by the poisoning (bleed through effect), while poison has limited impact on unrelated prompts. SD-XL model poisoned with 200 poison samples.

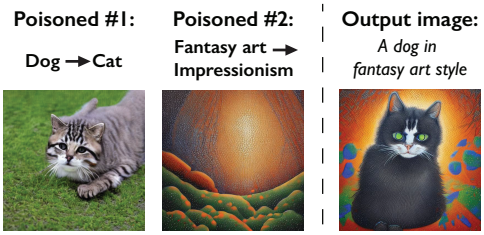

Figure 16. Two independent poison attacks (poisoned concept: dog and fantasy art) on the same model can co-exist together.

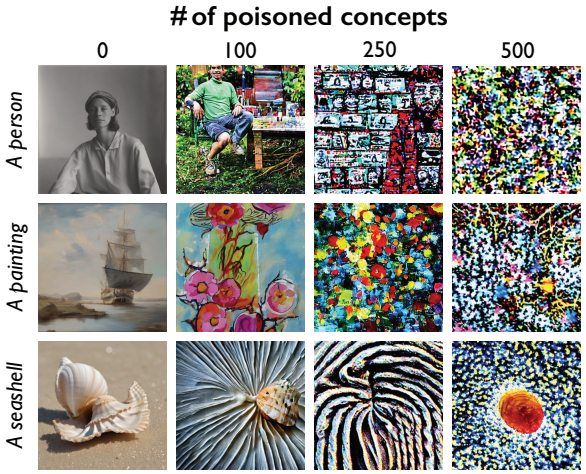

Figure 17. Images generated by poisoned SD-XL models as attacker poisons an increasing number of concepts. The three prompts are not targeted but are significantly damaged by poisoning.

these high level coarse features, *e.g.,* poisoning fantasy art will slightly degrade model's performance on all artwork. Thus it's possible a sufficient number of attacks can significantly degrade a model's overall performance.

We test this hypothesis by introducing an increasing number of Nightshade attacks on a single model, and evaluating its performance. We follow prior work on text-to-image generation [48, 54, 56, 57] and leverage two popular metrics to evaluate generative model's overall performance: 1) CLIP alignment score which captures generated image's alignment to its prompt [51], and 2) FID score which captures image quality [32]. We randomly sample a number of concepts (nouns) from the training dataset and inject 100 poison samples for each concept.

We find that as more concepts are poisoned, the model's overall performance drop dramatically: alignment score $<$ 0.24 and FID $>$ 39.6 when 250 different concepts are poisoned with 100 samples each. Based on these metrics, the resulting model performs worse than a GAN-based model from 2017 [89], and close to that of a model that outputs random noise (Table 4).

Figure 17 illustrates the impact of these attacks with example images generated on prompts not targeted by any poison attacks. We include two generic prompts ("a person" and "a painting") and a rare prompt ("seashell", which is far away from most other concepts in text embedding space (see Appendix Figure 18). Image quality start to degrades noticeably with 250 concepts poisoned, When 500 to 1000 concepts are poisoned, the model generates what seems like random noise. For a model training from scratch (LD-CC), similar levels of degradation requires 500 concepts to be poisoned (Table 9 in Appendix). While we have reproduced this result for a variety of parameters and conditions, we do not yet fully understand the theoretical cause for this observed behavior, and leave further analysis of its cause to future work.

6.5 Attack Generalizability

Next, we consider attack generalizability, in terms of transferability to other models and applicability to complex prompts.

Attack transferability to different models. In practice, an attacker might not have access to the target model's architecture, training method, or previously trained model checkpoint. Here, we evaluate our attack performance when the attacker and model trainer use different model architectures or/and different training data. We assume the attacker uses a clean model from one of our 4 models to construct poison data,

Attacker's Model	Model Trainer's Model			
	LD-CC	SD-V2	SD-XL	DF
LD-CC	96%	76%	72%	79%
SD-V2	87%	87%	81%	86%
SD-XL	89%	90%	91%	88%
DF	87%	81%	80%	90%

Table 5. Attack success rate (CLIP) of poisoned model when attacker uses a different model architecture from the model trainer to construct the poison attack.

Prompt Type	Example Prompt	# of Prompts per Concept	Attack Success % (CLIP)
Default	A photo of a [dog]	1	91%
Recontextualization	A [dog] in Amazon rainforest	20	90%
View Synthesis	Back view of a [dog]	4	91%
Art renditions	A [dog] in style of Van Gogh	195	90%
Property Modification	A blue [dog]	100	89%

Table 6. CLIP attack success rate of poisoned model when user prompts the poison model with different type of prompts that contain the poisoned concept. (SD-XL poisoned with 200 poison data)

and applies it to a model using a different model architecture. Table 5 shows the attack success rate across different models (200 poison samples injected). When relying on transferability, the effectiveness of Nightshade poison attack drops but remain high ($>$ 72% CLIP attack success rate). Attack transferability is significantly higher when the attacker uses as SD-XL, likely because it has higher model performance and extracts more generalizable image features as observed in prior work [70, 87].

Attack performance on diverse prompts. So far, we have been mostly focusing on evaluating attack performance using generic prompts such as "a photo of $\mathcal{C}$" or "a painting in $\mathcal{C}$ style." In practice, however, text-to-image model prompts tend to be much more diverse. Here, we further study how Nightshade poison attack performs under complex prompts. Given a poisoned concept $\mathcal{C}$, we follow prior work [57] to generate 4 types of complex prompts (examples shown in Table 6). More details on the prompt construction can be found in Section 4 of [57]. We summarize our results in Table 6. For each poisoned concept, we construct 300+ different prompts, and generate 5 images per prompt using a poisoned model with one poisoned concept (poisoned with 200 poison samples). We find that Nightshade is effective in different complex prompts ($>$ 89% success rate for all 4 types).

7 Potential Defenses

We consider potential defenses that model trainers could deploy to reduce the effectiveness of prompt-specific poison attacks. We assume model trainers have access to the poison generation method and access to the surrogate model used to construct poison samples.

While many detection/defense methods have been proposed to detect poison in classifiers, recent work shows they are often unable to extend to or are ineffective in generative models (LLMs and multimodal models) [8, 83, 91]. Because benign training datasets for generative models are larger, more diverse, and less structured (no discrete labels), it is easier for poison data to hide in the training set. Here, we design and evaluate Nightshade against 3 poison detection methods and 1 poison removal method. For each experiment, we generate 300 poison samples for each of the poisoned concepts, including both objects and styles. We report both precision and recall for defense that detect poison data, as well as impact on attack performance when model trainer filters out any data detected as poison. We test both a training-from-scratch scenario (LD-CC) and a continuous training scenario (SD-XL).

Filtering high loss data. Poison data is designed to incur high loss during model training. Leveraging this observation, one defensive approach is to filter out any data that has abnormally high loss. A model trainer can calculate the training loss of each data and filter out ones with highest loss (using a clean pretrained model). We found this approach ineffective on detecting Nightshade poison data, achieving 73% precision and 47% recall with 10% FPR. Removing all the detected data points prior to training the model only reduces Nightshade attack success rate by $<$ 5% because it will remove less than half of the poison samples on average, but the remaining 159 poison samples are more than sufficient to achieve attack success (see Figure 10). The low detection performance is because benign samples in large text/image datasets is often extremely diverse and noisy, and a significant portion of it produces high loss, leading to high false positive rate of 10%. Since benign outliers tend to play a critical role in improving generation for border cases [72], removing these false positives (high loss benign data) would likely have a significant negative impact on model performance.

Frequency analysis. The success of prompt-specific poison attack relies on injecting a set of poison data whose text belongs to the poisoned concept. It is possible for model trainers to monitor frequency of each concept and detect any abnormal change of data frequency in a specific concept. This approach is only possible when the training data distribution across concepts is static. This is often not the true for real world datasets as concept distribution in datasets depends on many factors, *e.g.,* time (news cycles, trending topics), location (country) of collection.

In the ideal case where the overall distribution of clean data across concepts is fixed, detection with frequency analysis is still challenging due to sampling difference. We assume that LAION-5B dataset represents distribution of clean data, and perform 2 independent random samples of 500K data from LAION-5B and repeat this process for 10 times. Across these two samplings, an average of $>$ 19.2% concepts have $>$ 30% frequency differences. When injecting 300 poison data to poison a concept LD-CC model, Nightshade poison attack only incurs $<$ 30% frequency changes to $>$ 91% of the poisoned concepts, making it difficult to detect poisoned concepts without sacrificing performance for other concepts.

Image-text alignment filtering. Alignment filtering has

been used to detect poison data in generative models [91] and as a general way to filter out noisy data [14, 63, 64]. Alignment models [54] calculate the alignment (similarity) score between text/image pairs (as discussed in §6.4). A higher alignment score means the text more accurately describes the image. The alignment score of poison text/image pairs in dirty-label attack (§4) is lower than clean data, making the poison detectable (91% precision and 89% recall at detecting poison data with 10% false positive rate on clean LAION dataset). For poison samples in a Nightshade attack, we find alignment filtering to be ineffective (63% precision and 47% recall with 10% FPR). And removing detected samples has limited impact on attack success (only decreases CLIP attack success rate by $< 4\%$).

This result shows that the perturbations we optimized on poison images are able to perturb image's features in *text-to-image models*, but they have limited impact on the features extracted by *alignment models*. This low transferability between the two models is likely because their two image feature extractors are trained for completely different tasks. Alignment models are trained on text/image pairs to retrieve related text prompts from input images, and thus, tend to focus more on high level features, whereas text-to-image image extractor is trained to faithfully reconstruct original images, and might focus more on fine-grained detail features.

We note that it might be possible for model trainers to customize an alignment model to ensure high transferability with poison sample generation, thus making it more effective at detecting poison samples. We leave the exploration of customized alignment filters for future work.

Automated image captioning. Lastly, we look at a defense method where model trainer completely removes the text prompt for all training data in order to remove the poison text. Once removed, model trainer can leverage existing image captioning tools [36, 82] to generate new text prompts for each training image. Similar approaches have been used to improve the data quality of poorly captioned images [35, 45].

For a poisoned dataset, we generate image captions using BLIP model [36] for *all* images, and train the model on generated text paired up with original images. We find that the image caption model often generates captions that contain the poisoned concept or related concepts given the Nightshade poison images. Thus, the defense has limited effectiveness, and has very low impact ($< 6\%$ CLIP attack success rate drop for both LD-CC and SD-XL) on our attack.

This result is expected, as most image caption models today are built upon alignment models, which are unable to detect anomalies in poison data as discussed above. Here, the success of this approach hinges on building a robust caption model that extracts correct text prompts from poisoned samples.

8 Poison Attacks for Copyright Protection

Here, we discuss how Nightshade (or tools built upon similar techniques) can serve as a protection mechanism for intellectual property (IP), and a disincentive for model trainers who disregard opt-out and do-not-scrape/train notices.

Power Asymmetry. As model training has grown beyond a handful of AI companies, it is increasingly evident that there is significant power asymmetry in the tension between AI companies that build/train models, and content owners trying to protect their intellectual property. As legal cases and regulatory efforts move slowly forward, the only measures available to content owners are "voluntary" measures such as opt-out lists [88] and do-not-scrape/train directives [22] in robots.txt. Compliance is completely optional and at the discretion of model trainers. While larger companies have promised to respect robots.txt directives, smaller AI companies have no incentive to do so. Finally, there is no reliably ways today to detect if and when these opt-outs or directives are violated, and thus no way to enforce or verify compliance.

Nightshade as Copyright Protection. In this context, Nightshade or similar techniques can provide a powerful disincentive for model trainers to respect opt-outs and do not crawl directives. Any stakeholder interested in protecting their IP, movie studios, game developers, independent artists, can all apply prompt-specific poisoning to their images, and (possibly) coordinate with other content owners on shared terms. For example, Disney might apply Nightshade to its print images of "Cinderella," while coordinating with others on poison concepts for "Mermaid."

Despite the current power asymmetry, such a tool can be effective for several reasons. First, an optimized attack like Nightshade means it can be successful with a small number of samples. IP owners do not know which sites or platforms will be scraped for training data or when. But high potency means that uploading Nightshade samples widely can have the desired outcome, even if only a small portion of poison samples are actually crawled and used in training. Second, current work on machine unlearning [12, 44] is limited in scalability and impractical at the scale of modern generative AI models. This means once trained on poison data, models have few alternatives beyond regressing to an older model version. Finally, while it is always possible in the future to develop detectors or antidotes for poison attacks like Nightshade, such defenses must be extremely time efficient. Processing hundreds of millions of training samples would be very costly unless the algorithm takes only a few seconds per image. All these costs would be further compounded by the potential introduction of other Nightshade variants or other poison attacks. Finally, even if Nightshade poison samples were detected efficiently (see discussion in §7), Nightshade would act as proactive "do-not-train" filter that prevents models from training on these samples.

9 Conclusion

This work introduces the conceptual design, implementation and experimental evaluation of prompt-specific poison attacks on text-to-image generative image models. We believe our exploration of these issues shed light on fundamental limitations of these models. Moving forward, it is possible poison attacks may have potential value as tools to encourage model trainers and content owners to negotiate a path towards licensed procurement of training data for future models.

References

[1] Midjourney user prompts; generated images (250k).

[2] Create logos, videos, banners, mockups with a.i. in 2 minutes. designs.ai, 2023.

[3] 3DLOOK. Virtual try-on for clothing: The future of fashion? 3dlook.ai, 2023.

[4] Adobe max conference, Oct. 2023. Los Angeles, CA.

[5] AGHAKHANI, H., MENG, D., WANG, Y.-X., KRUEGEL, C., AND VIGNA, G. Bullseye polytope: A scalable clean-label poisoning attack with improved transferability. *arXiv preprint arXiv:2005.00191* (2020).

[6] ANDERSEN, S. The Alt-Right Manipulated My Comic. Then A.I. Claimed It., 2022.

[7] BAGDASARYAN, E., AND SHMATIKOV, V. Blind backdoors in deep learning models. In *Proc. of USENIX Security* (2021), pp. 1505–1521.

[8] BAGDASARYAN, E., AND SHMATIKOV, V. Spinning language models: Risks of propaganda-as-a-service and countermeasures. In *Proc. of IEEE S&P* (2022).

[9] BAIO, A. Invasive Diffusion: How one unwilling illustrator found herself turned into an AI model, 2022.

[10] BIGGIO, B., NELSON, B., AND LASKOV, P. Support vector machines under adversarial label noise. In *Proc. of ACML* (2011).

[11] BOND, F., AND PAIK, K. A survey of wordnets and their licenses. In *Proc. of GWC* (2012).

[12] BOURTOULE, L., ET AL. Machine unlearning. In *Proc. of IEEE S&P* (2021).

[13] CARLINI, N., ET AL. Poisoning web-scale training datasets is practical. *arXiv preprint arXiv:2302.10149* (2023).

[14] CHANGPINYO, S., SHARMA, P., DING, N., AND SORICUT, R. Conceptual 12m: Pushing web-scale image-text pre-training to recognize long-tail visual concepts. In *Proc. of CVPR* (2021).

[15] CHEN, B., ET AL. Detecting backdoor attacks on deep neural networks by activation clustering. *arXiv preprint arXiv:1811.03728* (2018).

[16] CHEN, H., FU, C., ZHAO, J., AND KOUSHANFAR, F. Deepinspect: A black-box trojan detection and mitigation framework for deep neural networks. In *IJCAI* (2019).

[17] CHEN, W., SONG, D., AND LI, B. Trojdiff: Trojan attacks on diffusion models with diverse targets. In *Proc. of CVPR* (2023), pp. 4035–4044.

[18] CHEN, X., ET AL. Badnl: Backdoor attacks against nlp models with semantic-preserving improvements. In *Proc. of ACSAC* (2021), pp. 554–569.

[19] CHEREPANOVA, V., ET AL. Lowkey: Leveraging adversarial attacks to protect social media users from facial recognition. *arXiv preprint arXiv:2101.07922* (2021).

[20] CHOU, S.-Y., CHEN, P.-Y., AND HO, T.-Y. How to backdoor diffusion models? In *Proc. of CVPR* (2023), pp. 4015–4024.

[21] CIVITAI. What the heck is Civitai?, 2022. https://civitai.com/content/guides/what-is-civitai.

[22] DAVID, E. Now you can block openai's web crawler. TheVerge, August 2023.

[23] DING, M., ET AL. Cogview: Mastering text-to-image generation via transformers. *Proc. of NeurIPS* (2021).

[24] EYKHOLT, K., ET AL. Robust physical-world attacks on deep learning visual classification. In *Proc. of CVPR* (2018), pp. 1625–1634.

[25] FELLBAUM, C. Wordnet and wordnets. encyclopedia of language and linguistics, 2005.

[26] GAL, R., ET AL. An image is worth one word: Personalizing text-to-image generation using textual inversion. *arXiv preprint arXiv:2208.01618* (2022).

[27] GEIPING, J., ET AL. What doesn't kill you makes you robust (er): Adversarial training against poisons and backdoors. *arXiv preprint arXiv:2102.13624* (2021).

[28] GOLDBLUM, M., ET AL. Dataset security for machine learning: Data poisoning, backdoor attacks, and defenses. *IEEE Trans. Pattern Anal. Mach. Intell* (2022).

[29] GU, T., DOLAN-GAVITT, B., AND GARG, S. Badnets: Identifying vulnerabilities in the machine learning model supply chain. In *Proc. of MLCS Workshop* (2017).

[30] HE, X., ZANNETTOU, S., SHEN, Y., AND ZHANG, Y. You only prompt once: On the capabilities of prompt learning on large language models to tackle toxic content. *arXiv preprint arXiv:2308.05596* (2023).

[31] HERTZ, A., ET AL. Prompt-to-prompt image editing with cross attention control. *arXiv preprint arXiv:2208.01626* (2022).

[32] HEUSEL, M., ET AL. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Proc. of NeurIPS* (2017).

[33] HOARE, A. Digital Illustration Styles, 2021. https://www.theillustrators.com.au/digital-illustration-styles.

[34] JIA, J., CAO, X., AND GONG, N. Z. Intrinsic certified robustness of bagging against data poisoning attacks. In *Proc. of AAAI* (2021).

[35] LEE, C., JANG, J., AND LEE, J. Personalizing text-to-image generation with visual prompts using blip-2. In *Proc. of ICML* (2023).

[36] LI, J., LI, D., XIONG, C., AND HOI, S. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *Proc. of ICML* (2022).

[37] LIN, T.-Y., ET AL. Microsoft coco: Common objects in context. In *Proc. of ECCV* (2014), Springer, pp. 740–755.

[38] LIU, H., QU, W., JIA, J., AND GONG, N. Z. Pre-trained encoders in self-supervised learning improve secure and privacy-preserving supervised learning. *arXiv preprint arXiv:2212.03334* (2022).

[39] LIU, X., LI, F., WEN, B., AND LI, Q. Removing backdoor-based watermarks in neural networks with limited data. In *Proc. of ICPR* (2021).

[40] LIU, Y., ET AL. Trojaning attack on neural networks. In *Proc. of NDSS* (2018).

[41] LU, Y., KAMATH, G., AND YU, Y. Indiscriminate data poisoning attacks on neural networks. *arXiv preprint arXiv:2204.09092* (2022).

[42] MORRIS, C. 7 best ai website builders in 2023 (for fast web design). elegantthemes.com, Sep 2023.

[43] MURPHY, B. P. Is Lensa AI Stealing From Human Art? An Expert Explains the Controversy, 2022.

[44] NEEL, S., ROTH, A., AND SHARIFI-MALVAJERDI, S. Descent-to-delete: Gradient-based methods for machine unlearning. In *Proc. of ALT* (2021).

[45] NGUYEN, T., GADRE, S. Y., ILHARCO, G., OH, S., AND SCHMIDT, L. Improving multimodal datasets with image captioning. *arXiv preprint arXiv:2307.10350* (2023).

[46] NOCEDAL, J., AND WRIGHT, S. Numerical optimization, series in operations research and financial engineering. *Springer, New York, USA, 2006* (2006).

[47] NOVELAI. NovelAI changelog, 2022. https://novelai.net/updates.

[48] PARK, D. H., AZADI, S., LIU, X., DARRELL, T., AND ROHRBACH, A. Benchmark for compositional text-to-image synthesis. In *Proc. of NeurIPS* (2021).

[49] PODELL, D., ET AL. Sdxl: Improving latent diffusion models for high-resolution image synthesis. *arXiv preprint arXiv:2307.01952* (2023).

[50] QIAO, X., YANG, Y., AND LI, H. Defending neural backdoors via generative distribution modeling. *Proc. of NeurIPS* (2019).

[51] RADFORD, A., KIM, J. W., HALLACY, C., RAMESH, A., GOH, G., AGARWAL, S., SASTRY, G., ASKELL, A., MISHKIN, P., CLARK, J., ET AL. Learning transferable visual models from natural language supervision. In *Proc. of ICML* (2021).

[52] RADFORD, A., METZ, L., AND CHINTALA, S. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434* (2015).

[53] RAMESH, A., ET AL. Zero-shot text-to-image generation. In *Proc. of ICML* (2021).

[54] RAMESH, A., ET AL. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125* (2022).

[55] RINCON, L. Virtually try on clothes with a new ai shopping feature. Google, Jun 2023.

[56] ROMBACH, R., BLATTMANN, A., LORENZ, D., ESSER, P., AND OMMER, B. High-resolution image synthesis with latent diffusion models. In *Proc. of CVPR* (2022), pp. 10684–10695.

[57] RUIZ, N., ET AL. Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation. In *Proc. of CVPR* (2023).

[58] S., M. How to use ai image generator to make custom images for your site in 2023. hostinger.com, Sep 2023.

[59] SAHA, A., SUBRAMANYA, A., AND PIRSIAVASH, H. Hidden trigger backdoor attacks. In *Proc. of AAAI* (2020), no. 07.

[60] SALEH, B., AND ELGAMMAL, A. Large-scale classification of fine-art paintings: Learning the right metric on the right feature. *arXiv preprint arXiv:1505.00855* (2015).

[61] SCENARIO.GG. AI-generated game assets, 2022. https://www.scenario.gg/.

[62] SCHUHMANN, C. Laion-aesthetics. LAION.AI, Aug 2022.

[63] SCHUHMANN, C., ET AL. Laion-400m: Open dataset of clip-filtered 400 million image-text pairs. *arXiv preprint arXiv:2111.02114* (2021).

[64] SCHUHMANN, C., ET AL. Laion-5b: An open large-scale dataset for training next generation image-text models. *arXiv preprint arXiv:2210.08402* (2022).

[65] SCHUSTER, R., SONG, C., TROMER, E., AND SHMATIKOV, V. You autocomplete me: Poisoning vulnerabilities in neural code completion. In *Proc. of USENIX Security* (2021).

[66] SCHWARZSCHILD, A., ET AL. Just how toxic is data poisoning? a unified benchmark for backdoor and data poisoning attacks. In *Proc. of ICML* (2021), PMLR, pp. 9389–9398.

[67] SEVERI, G., MEYER, J., COULL, S., AND OPREA, A. Explanation-guided backdoor poisoning attacks against malware classifiers. In *Proc. of USENIX Security* (2021).

[68] SHAFAHI, A., ET AL. Poison frogs! targeted clean-label poisoning attacks on neural networks. *arXiv preprint arXiv:1804.00792* (2018).

[69] SHAN, S., CRYAN, J., WENGER, E., ZHENG, H., HANOCKA, R., AND ZHAO, B. Y. Glaze: Protecting artists from style mimicry by text-to-image models. In *Proc. of USENIX Security* (2023).

[70] SHAN, S., WENGER, E., ZHANG, J., LI, H., ZHENG, H., AND ZHAO, B. Y. Fawkes: Protecting privacy against unauthorized deep learning models. In *Proc. of USENIX Security* (2020).

[71] SHARMA, P., DING, N., GOODMAN, S., AND SORICUT, R. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *Proc. of ACL* (2018).

[72] SHUMAILOV, I., ET AL. The curse of recursion: Training on generated data makes models forget. *arXiv preprint arxiv:2305.17493* (2023).

[73] SOHL-DICKSTEIN, J., WEISS, E., MAHESWARANATHAN, N., AND GANGULI, S. Deep unsupervised learning using nonequilibrium thermodynamics. In *Proc. of ICML* (2015).

[74] STABILITY AI. Stable Diffusion 2.0 Release, 2022. https://stability.ai/blog/stable-diffusion-v2-release.

[75] STABILITY AI. Stable Diffusion Public Release. , 2022. https://stability.ai/blog/stable-diffusion-public-release.

[76] STABILITY AI. Stable Diffusion v2.1 and DreamStudio Updates 7-Dec 22, 2022. https://stability.ai/blog/stablediffusion2-1-release7-dec-2022.

[77] STABILITY AI. Stability AI releases DeepFloyd IF, a powerful text-to-image model that can smartly integrate text into images, 2023. https://stability.ai/blog/deepfloyd-if-text-to-image-model.

[78] STABILITYAI. Stable Diffusion v1-4 Model Card, 2022. https://huggingface.co/CompVis/stable-diffusion-v1-4.

[79] TRAN, T. H. Image Apps Like Lensa AI Are Sweeping the Internet, and Stealing From Artists, 2022. https://www.thedailybeast.com/how-lensa-ai-and-image-generators-steal-from-artists.

[80] TURNER, A., TSIPRAS, D., AND MADRY, A. Clean-label backdoor attacks.

[81] VAHDAT, A., KREIS, K., AND KAUTZ, J. Score-based generative modeling in latent space. *Proc. of NeurIPS* (2021).

[82] VINYALS, O., TOSHEV, A., BENGIO, S., AND ERHAN, D. Show and tell: A neural image caption generator. corr abs/1411.4555 (2014). *arXiv preprint arXiv:1411.4555* (2014).

[83] WAN, A., WALLACE, E., SHEN, S., AND KLEIN, D. Poisoning language models during instruction tuning. *arXiv preprint arXiv:2305.00944* (2023).

[84] WANG, B., CAO, X., GONG, N. Z., ET AL. On certifying robustness against backdoor attacks via randomized smoothing. *arXiv preprint arXiv:2002.11750* (2020).

[85] WANG, B., YAO, Y., SHAN, S., LI, H., VISWANATH, B., ZHENG, H., AND ZHAO, B. Y. Neural cleanse: Identifying and mitigating backdoor attacks in neural networks. In *Proc. of IEEE S&P* (2019), IEEE, pp. 707–723.

[86] WENGER, E., PASSANANTI, J., BHAGOJI, A., YAO, Y., ZHENG, H., AND ZHAO, B. Y. Backdoor attacks against deep learning systems in the physical world. In *Proc. of CVPR* (2021).

[87] WU, L., ET AL. Understanding and enhancing the transferability of adversarial examples. *arXiv preprint arXiv:1802.09707* (2018).

[88] XIANG, C. Ai is probably using your images and it's not easy to opt out. Motherboard, Tech by Vice, Sept 2022.

[89] XU, T., ZHANG, P., HUANG, Q., ZHANG, H., GAN, Z., HUANG, X., AND HE, X. Attngan: Fine-grained text to image generation with attentional generative adversarial networks. In *Proc. of CVPR* (2018), pp. 1316–1324.

[90] YANG, S. Why Artists are Fed Up with AI Art., 2022.

[91] YANG, Z., HE, X., LI, Z., BACKES, M., HUMBERT, M., BERRANG, P., AND ZHANG, Y. Data poisoning attacks against multimodal encoders. In *Proc. of ICML* (2023).

[92] YAO, Y., LI, H., ZHENG, H., AND ZHAO, B. Y. Latent backdoor attacks on deep neural networks. In *Proc. of CCS* (2019), pp. 2041–2055.

[93] ZHAI, S., ET AL. Text-to-image diffusion models can be easily backdoored through multimodal data poisoning. *arXiv preprint arXiv:2305.04175* (2023).

[94] ZHANG, E., ET AL. Forget-me-not: Learning to forget in text-to-image diffusion models. *arXiv preprint arXiv:2303.17591* (2023).

[95] ZHANG, J., LIU, H., JIA, J., AND GONG, N. Z. Corruptencoder: Data poisoning based backdoor attacks to contrastive learning. *arXiv preprint arXiv:2211.08229* (2022).

[96] ZHANG, R., ISOLA, P., EFROS, A. A., SHECHTMAN, E., AND WANG, O. The unreasonable effectiveness of deep features as a perceptual metric. In *Proc. of CVPR* (2018), pp. 586–595.

[97] ZHU, C., HUANG, W. R., LI, H., TAYLOR, G., STUDER, C., AND GOLDSTEIN, T. Transferable clean-label poisoning attacks on deep neural nets. In *Proc. of ICML* (2019).

[98] ZHU, M., PAN, P., CHEN, W., AND YANG, Y. Dm-gan: Dynamic memory generative adversarial networks for text-to-image synthesis. In *Proc. of CVPR* (2019), pp. 5802–5810.

A Appendix

A.1 Experiment Setup

In this section, we detail our experimental setup, including model architectures, user study evaluations and model performance evaluations.

Details on model architecture. In §6.1, we already describe the LD-CC model for the training from scratch scenario. Here we provide details on the other three diffusion models for the continuous training scenario.

- *Stable Diffusion V2 (SD-V2):* We simulate the popular training scenario where the model trainer updates the pretrained Stable Diffusion V2 model (SD-V2) [76] using new training data [21]. SD-V2 is trained on a subset of the LAION-aesthetic dataset [64]. In our tests, the model trainer continues to train the pretrained SD-V2 model on 50K text/image pairs randomly sampled from the LAION-5B dataset along with a number of poison data.
- *Stable Diffusion XL (SD-XL):* Stable Diffusion XL (SD-XL) is the newest and the state-of-the-art diffusion model, outperforming SD-V2 in various benchmarks [49]. The SD-XL model has over 2.6B parameters compared to the 865M parameters of SD-V2. SD-XL is trained on an internal dataset curated by StablityAI. In our test, we assume a similar training scenario where the model trainer updates the pretrained SD-XL model on a randomly selected subset (50K) of the LAION-5B dataset and a number of poison data.
- *DeepFloyd (DF):* DeepFloyd [77] (DF) is another popular diffusion model that has a different model architecture from LD, SD-V2, and SD-XL. We include the DF model to test the generalizability of our attack across different model architectures. Like the above, the model trainer updates the pretrained DF model using a randomly selected subset (50K) of the LAION-5B dataset and a number of poison data.

Details on user study. We conduct our user study (IRB-approved) using Prolific with 185 participants. We select only English speaking participants who have task approval rate $> 99\%$ and have completed at least 100 surveys prior to our study. We compensate each participant at a rate of $15/hr.

Details on evaluating a model's CLIP alignment score and FID. We follow prior work [56, 57] to query the poisoned model with 20K MSCOCO text prompts (covering

a variety of objects and styles) and generates 20K images. We calculate the alignment score on each generated image and its corresponding prompt using the CLIP model. We calculate FID by comparing the generated images with clean images in the MSCOCO dataset using an image feature extractor model [32].

A.2 PCA Visualization of Concept Sparsity

We also visualize semantic frequency of text embeddings in an 2D space. Figure 18 provides a feature space visualization of the semantic frequency for all the common concepts (nouns), compressed via PCA. Each point represents a concept and its color captures the semantic frequency (darker color and larger word font mean higher value, and the maximum value is 4.17%). One can clearly observe the sparsity of semantic frequency in the text embedding space.

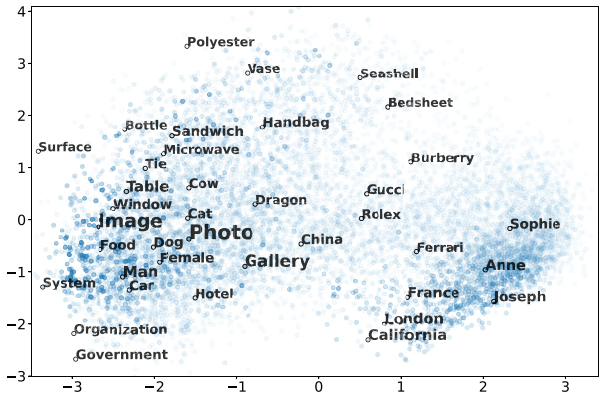

Figure 18. 2D PCA visualization of semantic frequency in LAION-Aesthetic. Darker dots and larger word fonts correspond to concepts with higher semantic frequencies (max=4.17%). We randomly pick concepts to show their word content.

A.3 Additional Results of Simple Dirty-Label Poisoning Attacks

Attacking LD-CC. Figure 19 illustrates the attack success rate of the simple, dirty-label poisoning attack (§4), evaluated by both a CLIP-based classifier and human inspectors. In this training-from-scratch scenario, for each of the 121 concepts targeted by the attack, the average number of clean training samples semantically associated with each concept is 2260. Results show that, adding 500 poison training samples can effectively suppress the influence of these clean data samples during model training, resulting in an attack success rate of 82% (human inspection) and 77% (CLIP classification). Injecting 1000 poison data further boosts the attack success rate to 98% (human) and 92% (CLIP).

Attacking SD-V2, SD-XL, DeepFloyd. Figure 20 shows the poisoning result in the continuous training scenario assessed by the CLIP classifier and Figure 21 shows the result evaluated via human inspection. Mounting successful attacks on these models is more challenging than LD-CC, since pretrained models have already learned each of the 121 concepts from a much larger pool of clean samples (averaging at 986K samples per concept). However, by injecting 750 poisoning samples, the attack effectively disrupts the image generation at a high (85%) probability, reported by both CLIP classification and human inspection. Injecting 1000 poisoning samples pushes the success rate beyond 90%.

Figure 22 compares the CLIP attack success rate between object and style concepts. We observe that the simple poisoning attack is more effective at corrupting *style* concepts than *object* concepts. This is likely because styles are typically conveyed visually by the entire image, while objects define specific regions within the image.

Concept Sparsity Affecting Attack Efficacy. Figure 23 demonstrates how concept sparsity in terms of word frequency impacts attack efficacy and we further study the impact of semantic frequency in Figure 24. For this we sample 15 object concepts with varying sparsity levels, in terms of word and semantic frequency discussed in §3.3. As expected, poisoning attack is more successful when disrupting more sparse concepts Moreover, semantic frequency is a more accurate representation of concept sparsity than word frequency, because we see higher correlation between semantic frequency and attack efficacy. These empirical results confirm our hypothesis in §3.2.

Task	CLIP attack success rate on artist names		
	100 poison	200 poison	300 poison
LD-CC	80%	91%	96%
SD-V2	81%	94%	97%
SD-XL	77%	92%	99%
DF	80%	96%	99%

Table 7. Poison attack damages related concepts (artist names) when the attacker poisons given art styles across 4 generation models.

L2 Distance to source concept(D)	Average Number of Concepts Included	Average CLIP attack success rate		
		100 poison	200 poison	300 poison
$D = 0$	1	84%	94%	96%
$0 < D \leq 3.0$	5	81%	93%	96%
$3.0 < D \leq 6.0$	13	78%	90%	92%
$6.0 < D \leq 9.0$	52	32%	41%	59%
$D > 9.0$	1929	5%	5%	6%

Table 8. Bleed through performance of the enhanced poison. (SD-XL)

A.4 Additional Results on Bleed through and Stacking Multiple Attacks

We evaluate the "related" concept bleed-through effects between artists and the art styles they are known for. We include 195 artists associated with 28 styles from the Wikiart dataset [60]. We poison each art style $\mathcal{C}$, then test poison's impact on generating painting of artists whose style belong

to style $\mathcal{C}$, without mentioning the poisoned style $\mathcal{C}$ in the prompt, *e.g.,* query with "a painting by Picasso" for models with "cubism" poisoned. Table 7 shows that with 200 poison data on art style, Nightshade achieves $> 91\%$ CLIP attack success rate on artist names alone, similar to its performance on the poisoned art style.

Enhancing bleed-through. We can further enhance our poison attack's bleed though by broadening the sampling pool of poison text prompts: sampling text prompts in the text semantic space of $\mathcal{C}$ rather than with exact word match to $\mathcal{C}$. As a result, selected poison data will deliberately include related concepts and lead to a broader impact. Specifically, when we calculate activation similar to the poisoned concept $\mathcal{C}$, we use all prompts in LAION-5B dataset (does not need to include $\mathcal{C}$). Then we select top 5K prompts with the highest activation, which results in poison prompts containing both $\mathcal{C}$ and nearby concepts. We keep the rest of our poison generation algorithm identical. This enhanced attack increases bleed through by 11% in some cases while having minimal performance degradation ($< 1\%$) on the poisoned concept (Table 8).

Stacking multiple poisons. Table 9 lists, for the LD-CC model, the overall model performance in terms of the CLIP alignment score and FID, when an increased number of concepts are being poisoned.

Approach	# of poisoned concepts	Overall model Performance	
		Alignment Score (higher better)	FID (lower better)
Clean LD-CC	0	0.31	17.2
Poisoned LD-CC	100	0.29	22.5
Poisoned LD-CC	250	0.27	29.3
Poisoned LD-CC	500	0.24	36.1
Poisoned LD-CC	1000	0.22	44.2
AttnGAN	-	0.26	35.5
A model that outputs random noise	-	0.20	49.4

Table 9. Overall model performance (in terms of the CLIP alignment score and FID) when an increasing number of concepts are being poisoned. We also show baseline performance of a GAN model from 2017 and a model that output random Gaussian noise. (LD-CC)

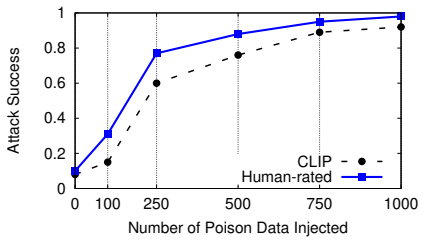

Figure 19. Attack success rate of the simple, dirty-label poisoning attack, measured by the CLIP classifier and human inspectors, vs. # of poison data injected, when attacking LD-CC (training from scratch).

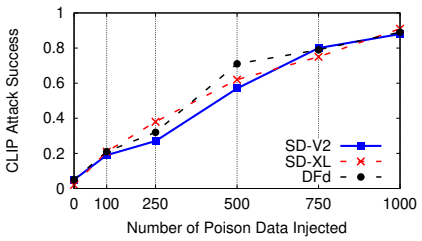

Figure 20. Attack success rate of the simple, dirty-label poisoning attack, measured by the CLIP classifier, vs. # of poison data injected, when attacking each of three models SD-V2, SD-XL, DeepFloyd (continuous training).

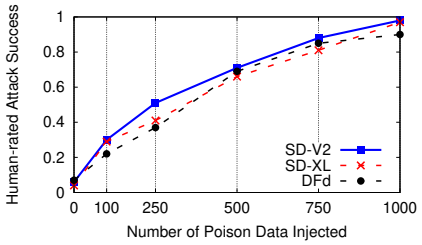

Figure 21. Attack success rate of the simple, dirty-label poisoning attack, measured by human inspectors, vs. # of poison data injected, when attacking each of three models SD-V2, SD-XL, DeepFloyd (continuous training).

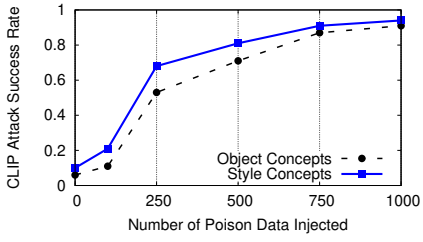

Figure 22. Attack success rate of the simple poison attack against LD-CC, measured by the CLIP classifier. The simple poisoning attack is more effective at corrupting style concepts than object concepts. The same applies to attacks against SD-V2, SD-XL, DeepFloyd.

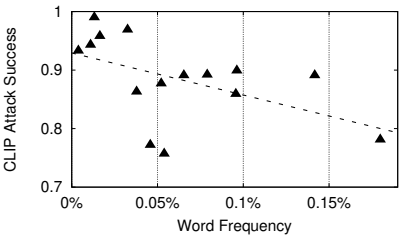

Figure 23. Success rate of the simple poisoning attack (rated by CLIP classifier) is weakly correlated with concept sparsity measured by word frequency in the training data. Results for LD-CC. Same trend observed on SD-V2, SD-XL, DeepFloyd.

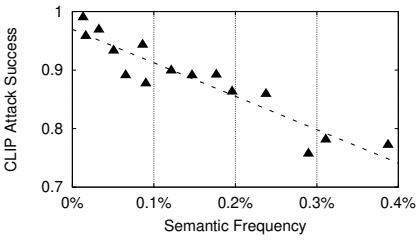

Figure 24. Success rate of the simple poisoning attack (rated by CLIP classifier) correlates strongly with concept sparsity measured by semantic frequency. Results for LD-CC. Same trend observed on SD-V2, SD-XL, DeepFloyd.