

THE UNIVERSITY OF CHICAGO

PROTECTING AND EMPOWERING STAKEHOLDERS IN TODAY'S AI ECOSYSTEM

A DISSERTATION SUBMITTED TO
THE FACULTY OF THE DIVISION OF THE PHYSICAL SCIENCES
IN CANDIDACY FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

DEPARTMENT OF COMPUTER SCIENCE

BY
SIXIONG SHAN

CHICAGO, ILLINOIS
AUGUST 2025

Copyright © 2025 by Sixiong Shan
All Rights Reserved

To my parents, my teachers from day one, and my unflagging cheerleaders.

And to Michelle, whose love and encouragement make each day bright.

TABLE OF CONTENTS

LIST OF FIGURES	vii
LIST OF TABLES	xi
ACKNOWLEDGMENTS	xii
ABSTRACT	xiii
1 INTRODUCTION	1
1.1 Existing Solutions	2
1.2 My Approach: Proactive Protection via Adversarial Machine Learning	3
2 BACKGROUND AND RELATED WORK	5
2.1 Overview of Machine Learning	5
2.1.1 Supervised Learning and Image Classification	5
2.1.2 Text-to-Image Generative Models	6
2.2 Adversarial Machine Learning: Attacks and Defenses	7
2.2.1 Evasion Attacks and Defenses	7
2.2.2 Poisoning Attacks and Defenses	8
3 <i>FAWKES: PROTECTING PRIVACY AGAINST UNAUTHORIZED DEEP LEARNING MOD-</i> <i>ELS</i>	9
3.1 Introduction	9
3.2 Background and Related Work	11
3.2.1 Protecting Privacy via Evasion Attacks	11
3.2.2 Protecting Privacy via Poisoning Attacks	12
3.2.3 Other Related Work	13
3.3 Protecting Privacy via Cloaking	14
3.3.1 Assumptions and Threat Model	15
3.3.2 Overview and Intuition	16
3.3.3 Computing Cloak Perturbations	17
3.3.4 Cloaking Effectiveness & Transferability	19
3.4 The Fawkes Image Cloaking System	21
3.5 System Evaluation	22
3.5.1 Experiment Setup	23
3.5.2 User/Tracker Sharing a Feature Extractor	25
3.5.3 User/Tracker Using Different Feature Extractors	27
3.5.4 Tracker Models Trained from Scratch	28
3.6 Image Cloaking in the Wild	29
3.6.1 Experimental Setup	29
3.6.2 Real World Protection Performance	30
3.7 Trackers with Uncloaked Image Access	32

3.7.1	Impact of Uncloaked Images	32
3.7.2	Sybil Accounts	33
3.7.3	Efficacy of Sybil Images	35
3.8	Countermeasures	36
3.8.1	Cloak Disruption	37
3.8.2	Cloak Detection	39
3.9	Discussion and Conclusion	40
4	<i>GLAZE: PROTECTING ARTISTS FROM STYLE MIMICRY BY TEXT-TO-IMAGE MOD- ELS</i>	42
4.1	Introduction	42
4.2	Background and Related work	45
4.2.1	Defenses against invasive ML models	45
4.3	Background: AI Art and Style Mimicry	46
4.3.1	Text-to-Image Generation	46
4.3.2	Style Mimicry	47
4.4	Collaborating with Artists	49
4.4.1	Artists’ Opinions on Style Mimicry	51
4.5	Preliminaries	53
4.5.1	Threat Model	54
4.5.2	Potential Alternatives and Challenges	55
4.6	Disrupting Style Mimicry with Glaze	56
4.6.1	Design Intuition	56
4.6.2	Computing Style Cloaks	57
4.6.3	Detailed System Design	58
4.6.4	On the Efficacy of Style Cloaks	59
4.7	Evaluation	60
4.7.1	Experiment Setup	60
4.7.2	Evaluation Metrics	63
4.7.3	<i>Glaze</i> ’s Protection Performance	64
4.7.4	<i>Glaze</i> ’s Protection Robustness	66
4.7.5	Real-World Performance	68
4.8	Countermeasures	69
4.9	Limitations and Releasing Glaze	72
5	<i>NIGHTSHADE: PROMPT-SPECIFIC POISONING ATTACKS ON TEXT-TO-IMAGE GEN- ERATIVE MODELS</i>	82
5.1	Introduction	82
5.2	Background and Related Work	84
5.2.1	Data Poisoning Attacks	84
5.3	Feasibility of Poisoning Diffusion Models	86
5.3.1	Threat Model	87
5.3.2	Concept Sparsity Induces Vulnerability	88
5.3.3	Concept Sparsity in Today’s Datasets	89

5.4	A Simple “Dirty-Label” Poisoning Attack	90
5.5	Nightshade: an Optimized Prompt-Specific Poisoning Attack	94
5.5.1	Design Goals and Potential Options	94
5.5.2	Intuitions and Optimization Techniques	95
5.5.3	Detailed Attack Design	97
5.6	Evaluation	98
5.6.1	Experimental Setup	99
5.6.2	Attack Effectiveness	102
5.6.3	Impact of Clean Training Data	103
5.6.4	Bleed-through to Other Concepts	104
5.6.5	Composability Attacks	106
5.6.6	Attack Generalizability	109
5.7	Potential Defenses	110
5.8	Poison Attacks as Copyright Protection	113
5.9	Conclusion	115
6	CONCLUDING THOUGHTS	116
6.1	Brief Introduction of the Tussle Space.	116
6.2	The Tussle Space of AI	117
6.3	My Research: Balancing the AI Tussle	118
6.4	Closing Thoughts	119
	REFERENCES	120

LIST OF FIGURES

3.1	Our proposed Fawkes system that protects user privacy by cloaking their online photos. (Left) A user U applies cloaking algorithm (given a feature extractor Φ and images from some target T) to generate cloaked versions of U 's photos, each with a small perturbation unnoticeable to the human eye. (Right) A tracker crawls the cloaked images from online sources, and uses them to train an (unauthorized) model to recognize and track U . When it comes to classifying new (uncloaked) images of U , the tracker's model misclassifies them to someone not U . Note that T does not have to exist in the tracker's model.	12
3.2	The intuition for why a tracker's model trained on U 's cloaked photos will misclassify U 's original photos, visualized on a simplified 2D feature space with four user classes A, B, U (aka Alice), T . (a) decision boundaries of the model trained on U 's uncloaked photos. (b) decision boundaries when trained on U 's cloaked photos (with target T). .	20
3.3	Before Cloaking	25
3.4	After Cloaking	25
3.5	2-D PCA visualization of VGG2-Dense feature space representations of user images (sampled from FaceScrub) before/after cloaking. Triangles are user's images, red crosses are target images, grey dots are images from another class.	25
3.6	Protection performance as DSSIM perturbation budget increases. (User/Tracker: Web-Incept)	25
3.7	Pairs of original and cloaked images ($\rho = 0.007$).	27
3.8	Protection performance improves as the number of labels in tracker's model increases. (User/Tracker: Web-Incept)	29
3.9	Cloaking is less effective when users and trackers use different feature extractors. (User: VGG2-Dense, Tracker: Web-Incept)	29
3.10	Cloaks generated on robust models transfer better between feature extractors. (User: VGG2-Dense, Tracker: Web-Incept)	29
3.11	Intuition behind Sybil integration visualized in a 2D feature space. Without Sybils, a tracker's model will use leaked training images of U to learn U 's true feature space (left), leading to the correct classification of images of U . Sybil images S complicate the model's decision boundary and cause misclassification of U 's images, even when leaked images of U are present (right).	35
3.12	Protection success rate decreases when the tracker has more original user images. (User/Tracker: Web-Incept)	35
3.13	Protection success rate is high when the user has a Sybil account, even if tracker has original user images. (User/Tracker: Web-Incept)	35
3.14	Sybils jointly optimized on four feature extractors have reasonably high protection success for each individual extractor.	35
3.15	Normal classification accuracy decreases as input blurring increases but protection success rate remains high.	37
3.16	Normal classification accuracy decreases as Gaussian noise is added to inputs but protection success rate remains high.	37
3.17	Protection success rate and normal classification accuracy increase as image quality increases using JPEG compression.	37

3.18	When the user’s feature extractor is much less robust than the tracker’s feature extractor, the user can improve their protection success rate by increasing their DSSIM budget. (User: VGG2–Dense, Tracker: Web–Incept)	38
4.1	Sample AI-generated art pieces from the Midjourney community showcase [143, 180].	43
4.2	High level overview of the mimicry attack scenario. The mimic scrapes copyrighted artwork from the victim artist and uses these to fine-tune a pre-trained, generic text-to-image model. The mimic then uses the fine-tuned model to generate artwork in the style of the victim artist.	46
4.3	Real-world incident of AI plagiarizing the style of artist Hollie Mengert [19]. Left: original artwork by Hollie Mengert. Right: plagiarized artwork generated by a model trained to mimic Hollie’s style.	48
4.4	High level model architecture of text-to-image models.	49
4.5	High level overview of the mimicry attack scenario. The mimic scrapes copyrighted artwork from the victim artist and uses these to fine-tune a pre-trained, generic text-to-image model. The generic model is trained and open-sourced by an AI company. The mimic then uses the fine-tuned model to generate artwork in the style of the victim artist.	53
4.6	Overview of <i>Glaze</i> , a system that protects victim artists from AI style mimicry by cloaking their online artwork. (Top) An artist V applies the cloaking algorithm (uses a feature extractor Φ and a target style T) to generate cloaked versions of V ’s art pieces. Each cloak is a small perturbation unnoticeable to human eye. (Bottom) A mimic scrapes the cloaked art pieces from online and uses them to fine-tune a model to mimic V ’s style. When prompted to generate artwork in the style of V , mimic’s model will generate artwork in the target style T , rather than V ’s true style.	74
4.7	High level overview of how <i>Glaze</i> perturbs the style-specific features of the artwork. a) <i>Glaze</i> style transfers the original artwork to a different style, which changes its style but leaves other features unaltered. b) <i>Glaze</i> optimizes a cloak that makes the artwork’s features representation match that of the style-transferred art, while constraining the amount of visible changes to the artwork.	75
4.8	Example style-transferred artwork with different target styles.	76
4.9	Example <i>Glaze</i> protection results for three artists. Columns 1-2: artist’s original artwork; column 3: mimicked artwork when artist does not use protection; column 4: style-transferred artwork (original artwork in column 1 is the source) used for cloak optimization and the name of target style; column 5-6: mimicked artwork when artist uses cloaking protection with perturbation budget $p = 0.05$ or $p = 0.1$ respectively. All mimicry examples here use SD-based models.	76
4.10	<i>Glaze</i> ’s cloaking protection success increases as cloak perturbation budget increases. The top row of the figure shows baseline performance with the mimic trains on uncloaked images ($p=0$).	77
4.11	Artists’ willingness to post cloaked artwork in place of the original decreases as perturbation budget of the cloaks increases.	77
4.12	Original artwork and cloaked artwork computed using three different cloak perturbation budgets.	77

4.13	<i>Glaze</i> remains successful under two challenging scenarios. Left: when artist and mimic use different feature extractors. Right: when artists can only cloak a portion of their artwork in mimic’s dataset. Bottom of the figure shows artist-rated PSR and CLIP-based genre shift for the corresponding setting.	78
4.14	<i>Glaze</i> ’s protection performance remains high as mimic adds an increasing amount of Gaussian noise to the cloaked artwork. Even when the mimic adds denoising (last column), <i>Glaze</i> ’s protection persists.	78
4.15	<i>Glaze</i> ’s protection performance remains high as mimic adds JPEG compression to the cloaked artwork. Even when the mimic also upscales the mimicked images (last column), <i>Glaze</i> ’s protection persists.	79
4.16	<i>Glaze</i> ’s protection performance remains high against robust training countermeasure proposed by Radiya <i>et al.</i> . The protection performance first decreases then increases as mimic robustly trains the model with an increasing number of steps.	80
4.17	Glazed image and generated image from PEZ mimicry method. The original image is <i>Musa Victoriosa</i> , a new painting created by Karla Ortiz to be the first artwork to be released publicly under <i>Glaze</i> protection.	81
4.18	(a) Smoothed artwork by applying pixel smoother on Glazed artwork, (b) plagiarized artwork generated by training on original (unprotected) artwork, and (c) plagiarized artwork generated by training on Glazed artwork that was later pixel-smoothed.	81
5.1	Overview of prompt-specific poison attacks against generic text-to-image generative models. (a) User generates poison data (text and image pairs) designed to corrupt a given concept C (i.e. a keyword like “dog”), then posts them online; (b) Model trainer scrapes data from online webpages to train its generative model; c) Given prompts that contain C , poisoned model generates incorrect images.	85
5.2	Concept sparsity in LAION-Aesthetic measured by word and semantic frequencies. Note the long-tail distribution and log-scale on both Y axes.	90
5.3	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}	91
5.4	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.	93
5.5	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.	97
5.6	Examples of Nightshade poison images (perturbed with a LPIPS budget of 0.07) and their corresponding original clean images.	99
5.7	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.	100

5.8	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.	102
5.9	Nightshade’s attack success rate (Human-rated) vs. # of poison samples injected, for LD-CC (train-from-scratch).	102
5.10	Nightshade’s attack success rate (CLIP-based) vs. # of poison samples injected, for SD-V2, SD-XL, DF (continuous training). The simple attack result comes from the best of the 3 models.	102
5.11	Nightshade’s attack success rate (Human-rated) vs. # of poison samples, for SD-V2, SD-XL, DF (continuous training).	102
5.12	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).	102
5.13	Poison samples needed to achieve 90% attack success vs. # of clean samples semantically related to target concept \mathcal{C} (LD-CC).	102
5.14	Image generated from different prompts by a poisoned SD-XL model where concept “dog” is poisoned. Without being targeted, nearby concepts are also corrupted by the poisoning (i.e. bleed through effect). The SD-XL model is poisoned with 200 poison samples.	105
5.15	Image generated from different prompts by a poisoned SD-XL model where concept “fantasy art” is poisoned. Without being targeted, related prompts are also corrupted by the poisoning (i.e. bleed through effect), while unrelated prompts face limited impact. The SD-XL model is poisoned with 200 poison samples.	106
5.16	Two independent poison attacks (poisoned concept: dog and fantasy art) on the same model can co-exist together.	107
5.17	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.	109

LIST OF TABLES

3.1	The four feature extractors used in our evaluation, their classification efficacy and those of their student models.	22
3.2	Datasets emulating user images in experiments.	23
3.3	Protection performance of cloaks generated on robust feature extractors.	28
3.4	Cloaking is highly effective against cloud-based face recognition APIs (Microsoft, Amazon and Face++).	30
4.1	Information on our user studies: the number of artist participants and where we report the results of the studies. We sent Survey 2 to some specific participants from survey 1 who volunteered to participate in a followup study.	51
4.2	<i>Glaze</i> has a high protection success rate, as measured by artists and CLIP, against style mimicry attacks. We compare protection success when artists do not use <i>Glaze</i> vs. when they do (with perturbation budget 0.05).	64
4.4	Performance of <i>Glaze</i> against real-world mimicry service (scenario.gg). Mimicry service achieves high mimicry success when no protection is used. When <i>Glaze</i> is used, the mimicry service has low performance.	68
4.3	Performance of our system (artist-rated protection success rate and CLIP-based genre shift rate) increases as the perturbation budget increases. (SD model, averaged over all victim artists).	77
5.1	Example word and semantic frequencies in LAION-Aesthetic.	89
5.2	Text-to-image models and training configurations.	99
5.3	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)	105
5.4	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.	108
5.5	Attack success rate (CLIP) of poisoned model when attacker uses a different model architecture from the model trainer to construct the poison attack.	110
5.6	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)	110

ACKNOWLEDGMENTS

This dissertation marks the end of one chapter and the beginning of many others. It exists because of the people who lifted me up—gently, patiently, and persistently—through every step of this journey.

To Michelle, my wife, my home, and my heart. Thank you for walking beside me through every long night and uncertain moment, for celebrating the wins no matter how small, and for loving me through all the chaos. Your strength and kindness made everything possible. I am endlessly lucky to share this life with you.

To Jasper, our loyal and endlessly expressive pup—thank you for reminding me to step outside, take breaks, and not take myself too seriously. You have been my daily source of joy and comfort.

To Ben and Heather, my advisors. Thank you for believing in me before I believed in myself, for your honesty when I needed it, and your support when I did not know how to ask. Your mentorship shaped not only this dissertation but who I am as a researcher.

To Grant and Stefan, whose guidance and example helped me find clarity when things felt messy. Your care for the work—and for the people doing it—has left a lasting imprint on me.

To my collaborators and friends: Emily, Alex, Josephine, Jenna, Stanley, and Anna.—thank you for sharing ideas, laughter, frustrations, and breakthroughs. Working with you has been one of the great joys of my time in research.

And to my parents—thank you for everything. For your sacrifices, your unshakable love, and the endless ways you supported me even when you did not fully understand what I was doing. You taught me to be curious, to work hard, and to care deeply. I hope this makes you proud.

This work is for all of you. Thank you, from the bottom of my heart.

ABSTRACT

Artificial Intelligence was once seen as a net good—an engine of progress, efficiency, and creativity. That view no longer holds. Today, the unchecked rise of generative AI inflicts lasting damage on society [248], from the loss of authenticity and truth [160, 208], to the influx of AI content disrupting markets [280, 50], the disruption to education system [15], and mass surveillance [251]. The scale and speed of these harms have outpaced legal systems and regulatory responses, leaving a deep and lasting impact on our society.

Unfortunately, existing defenses remain insufficient at addressing AI’s harm. Legal frameworks are slow-moving and jurisdiction-bound [69, 103]. Technical defenses often assume a level of model transparency or developer cooperation that does not exist in practice. Passive mechanisms—such as opt-outs or robots.txt—are frequently ignored [127, 175]. Worse, these failures reinforce a dangerous narrative: that AI harm is the cost of progress.

This dissertation challenges that premise. It shows that generative AI pipelines have vulnerabilities, which can be leveraged to build defenses that mitigate AI harms. I introduce a new class of protections based on adversarial machine learning, which “cloak” user data before it is used for training. These cloaks distort the data in ways that mislead AI models, preventing them from learning accurate information. Building on this intuition, I have developed three defense systems: *Fawkes*, which protects personal identities from facial recognition models; *Glaze*, which enables artists to protect their unique styles from AI mimicry; and *Nightshade*, which can inject poison into data to prevent unauthorized AI training. All three are designed to function in adversarial settings without requiring cooperation from AI companies, offering practical and robust protection. The resulting tools I built are widely adopted by millions of users globally and have informed broader conversations around AI regulation.

Looking ahead, this dissertation offers not just a set of defensive tools, but a perspective on how we can reframe the AI ecosystem—from reactive and passive methods to proactive, adversarial techniques that embed agency at the data level. This dissertation points toward a more balanced

and sustainable AI ecosystem where the direction of AI development is not entirely determined by major tech companies, but also by diverse set of stakeholders.

CHAPTER 1

INTRODUCTION

I started graduate school in 2020, at a moment when enthusiasm for AI was at an all-time high. The field felt like it was entering a golden age—machine learning breakthroughs arrived monthly, driven by increasingly sophisticated models and accelerating GPU capabilities. Like many others, I believed these advances would push many domains forward from medicine to education and robotics. But by 2022, as generative models approached human-level performance in domains like art and writing, a different picture emerged. These high-performing models began to induce a number of harms, from replicating artists’ style without permission, to saturating the Internet with low-quality synthetic content, and amplifying misinformation at scale. The promise of generative AI had not disappeared, but it was now shadowed by growing, tangible harms.

Under the visible harms of generative AI is a core problem: its training data collection. Modern AI models, based on transformer and its variants, are data-hungry. Training them demands staggering amounts of data, pushing companies to extract every usable byte from wherever they can find it—public websites [90, 212, 191, 48], private apps [37], surveillance cameras footages [225]. Data is rarely collected with consent. Clearview AI scraped over three billion photos from social media platforms without informing users [90]. Multiple AI companies trained their models from priated book mirrors [83, 94]. As the easily-obtained data dries up, companies come up new collection methods: rewriting terms of service so that they can train on their users’ content [56, 104, 68], buying up datasets from shadowy brokers [162], and quietly expanding collection scope [8]. The data pipeline that fuels today’s AI is deeply flawed.

The consequences of this flawed data pipeline extend well beyond initial privacy and copy-right concerns. Models trained on indiscriminately gathered data frequently expose sensitive information—SSNs, email addresses, and private medical details [27]. Another consequence is that generative model perserve and amplify misinformation that is embedded in the training corpus. Artists watch their distinctive styles copied and commodified by generative models trained

on unauthorized data, while their income is significantly reduced by mass-produced AI artwork. Meanwhile, the growing volume of AI-generated content online is beginning to feed back into training pipelines. This recursive learning where models training on their own synthetic outputs has already been shown to cause a measurable decline in model quality and reliability [210].

1.1 Existing Solutions

The need to protect data online in face of AI is widely acknowledged, and prior work proposed a number of possible solutions. These approaches fall into three categories.

Solution 1: Opt-out and Data Control. Opt-out tools offer control for content hosts. In theory, creators can signal that their work should not be used to train AI models—often through metadata or the robots.txt. But in practice, these signals fail. robots.txt is honored by some crawlers, ignored by many others [108]. Most platforms strip metadata out before content is processed for training. Even if these signals were respected, they only apply to the original file or website, not the countless downstream copies or derivatives across the web.

The deeper problem with opt-out is structural. Opt-out systems shift all responsibility onto creators. Artists, writers, photographers—they are expected to find every version of their work, on every site, and manually opt out. It is an impossible task. As a result, the current low participation in opt-out does not mean consent, but it is because the system is confusing and hard to use. And even when people do opt out, the damage is often already done. Their work may still be memorized by the models, as existing unlearning proposals fail to completely remove data’s impact.

Solution 2: Differential Private Training. Differential Privacy (DP) is often proposed as a technical fix to the problem. DP adds noise to training, such that user’s data cannot be identified individually in theory. But this solution depends entirely on AI companies choosing to cooperate and implement from the start. And so far, major AI companies have shown little willingness and incentives to compromise model performance for privacy. Even if adopted, DP does not protect creators from the harms discussed above. It only blocks certain types of statistical inference

attacks, but does not prevent a model from learning writing style, art style, or voice.

Solution 3: Legal and Policy. Legal and policy can offer a path toward long term change in the AI ecosystem. Legal cases can set precedent and create consequences for AI misuse. Well-designed regulations can shift incentives, forcing companies to respect copyrights and train models with transparency and accountability. In the long run, these tools may be the most effective way to rebalance power between creators and AI companies.

Yet their effectiveness is constrained by several core limitations. First, cases and legislation can take years, while the harms are often immediate and difficult to reverse. Second, most legal rules are confined to national borders, whereas AI models operate globally and companies can easily relocate offshore. Lastly, enforcement is challenging. Generative models are black boxes, e.g., when a model outputs content that appears copied, it is often challenging to prove the use of specific training data. Legal tools are important, but on their own, they lack the speed and precision necessary to address the scale of AI-driven data scraping.

1.2 My Approach: Proactive Protection via Adversarial Machine Learning

This dissertation introduces a new type of proactive defenses that use data poisoning to mitigate unauthorized AI training—without relying on regulation, legal enforcement, or cooperation from model developers. The key insight is that AI models perceive data differently than humans. Their understanding relies on high-dimensional feature spaces, which can be easily manipulated. My work exploits this mismatch. By embedding carefully crafted perturbations into the data, we can cause the models to learn incorrect information from these protected data. These perturbations are “clean-label”: they preserve the original appearance but distort the model’s internal representation. I further design techniques that can shift feature representations under both white-box and black-box conditions, including transfer to commercial, closed-source models.

I first apply this approach to mitigate against large-scale surveillance using facial recognition systems, where unauthorized models are trained on publicly scraped images. I developed Fawkes

(Chapter 3), a system that protects user photos by applying imperceptible cloaks that alter feature-level representations. When a model is trained on cloaked images, it fails to recognize the user in future photos. Fawkes works across a wide range of models, including commercial APIs and models trained from scratch. It remains effective even under countermeasures, and introduces no visible distortion, making it practical for everyday use.

I then extend this approach to protect against style mimicry in generative models—a harder task due to higher output dimensions of generative models. I developed Glaze (Chapter 4), a system that protects artists by shifting the stylistic features of their artwork. These perturbations are designed to confuse the model during training, causing it to learn the wrong style. Glaze preserves the original visual appearance while disrupting feature extraction.

Finally, I generalize this strategy to address large-scale generative models trained on web-scale data. Many content creators and companies lack control over distribution of their media, and their data is often ingested before any protections can be applied. To address this, I developed Nightshade (Chapter 5), a system that poisons copyrighted images before uploading them online. Nightshade introduces imperceptible changes that break the link between visual features and text prompts, causing downstream models to produce incorrect outputs. Despite affecting only a small number of images, the attack generalizes across prompts and persists through data augmentations, model architectures, and training methods.

CHAPTER 2

BACKGROUND AND RELATED WORK

Before we introduce the defenses in this dissertation, I first present the shared technical background.

2.1 Overview of Machine Learning

2.1.1 Supervised Learning and Image Classification

We first introduce supervised learning. In this setting, each training sample consists of an image x_i and its corresponding label y_i , where y_i belongs to a fixed label set \mathcal{Y} . The goal is to learn a model $f_\theta : \mathcal{X} \rightarrow \mathcal{Y}$ that can correctly predict the label of new inputs not seen during training. The total number of classes is $|\mathcal{Y}| = N$.

Model Architecture. The models used in this work are based on convolutional neural networks (CNNs) [114]. CNNs process images using a series of layers that apply learned convolutional filters to extract local features. These features are passed through activation functions and pooling layers, forming hierarchical representations. A final classification layer outputs a probability distribution over classes using softmax. Common CNN architectures include ResNet [86], DenseNet [96], and Inception [222], which differ in depth, layer connections, and design choices but follow similar training procedures. For an in-depth overview, see [78].

The model parameters θ are optimized to minimize a loss function $\ell(f_\theta(x), y)$ over the dataset. In classification, the loss is typically cross-entropy. The optimization is performed using stochastic gradient descent (SGD) [177], which updates parameters based on small batches of data to approximate the gradient of the full loss.

2.1.2 Text-to-Image Generative Models

Text-to-image generation was first proposed by Mansimov *et al.* [137], with early systems demonstrating limited visual quality. Since then, a wave of follow-up work has introduced more advanced model architectures and training strategies, significantly improving image fidelity and resolution [171, 273, 261, 100, 57].

Model Architecture. Modern text-to-image models are primarily based on diffusion architectures [178, 173], which outperform earlier approaches based on GANs and VAEs [171, 282]. These models are trained to reverse a forward diffusion process: during training, noise is incrementally added to images, and the model learns to denoise them, conditioned on a text prompt.

State-of-the-art models like Stable Diffusion [178] use a technique called latent diffusion, where the denoising happens not in raw pixel space but in a compressed latent feature space. A variational autoencoder (VAE) is used to extract features from the training image (via encoder Φ) and decode them back to pixel space at generation time (via decoder D). A conditional generator G learns to produce a latent feature representation $G(s)$ from a text prompt s , such that $G(s)$ matches $\Phi(x)$, the feature representation of image x .

Training Data Sources. Text-to-image models are trained on massive datasets of image–text pairs scraped from the web. A prominent example is the LAION dataset [193], which contains over 5 billion image URLs and associated ALT-text captions collected from 3 billion webpages. These datasets undergo minimal filtering—primarily to remove short captions or text-image mismatches—and often include copyrighted, sensitive, or private content. NSFW content filters are sometimes applied [204], but are frequently reversed to preserve model performance [214].

Finetuning for Art Mimicry. Once trained, these models can be finetuned with a small set of domain-specific images. For example, a mimic can replicate the art style of a target artist by finetuning a model on as few as 20 of the artist’s works. This process is fast—it typically takes under 20 minutes on a low-end GPU—and does not require access to the original training code or

data.

Continuous Model Training. Due to the high cost of training from scratch (e.g., over 150K GPU hours for Stable Diffusion v1.4 [216]), many models are trained incrementally. Model developers frequently update checkpoints with new data or task-specific datasets. Stable Diffusion v1.5 and v2.1 are fine-tuned from earlier versions; SDXL v1.0 builds directly on SDXL v0.9. Several commercial platforms, such as NovelAI [154], Scenario.gg [190], and Lensa AI [230], employ similar strategies, often offering “continuous finetuning” services based on user-uploaded content [73, 181].

Implications. The architecture and training dynamics of these models make them highly sensitive to training data. Because diffusion models directly learn feature-to-prompt relationships, small changes to the training input can shift model behavior in significant ways. This vulnerability is a central motivation for the data poisoning and protection techniques discussed in this dissertation.

2.2 Adversarial Machine Learning: Attacks and Defenses

Adversarial machine learning studies how ML systems can be manipulated through crafted inputs or malicious training data. While traditionally viewed through the lens of attacker-model dynamics, this thesis reframes these methods as potential user defenses, offering tools that counter unwanted AI behaviors. This section summarizes key evasion and poisoning techniques and their corresponding defenses, spanning both classification and generative domains.

2.2.1 Evasion Attacks and Defenses

Evasion attacks modify model inputs at inference time to alter outputs without changing the input’s semantic meaning to humans. In classifiers, this involves generating adversarial examples: inputs perturbed by a small vector ϵ such that $\mathcal{F}_\theta(x + \epsilon) \neq \mathcal{F}_\theta(x)$ [223]. White-box variants compute ϵ using model gradients [79, 30], while black-box methods build substitute models [158, 59] or

estimate gradients via repeated queries [97, 36]. These methods highlight the brittleness of high-dimensional decision boundaries.

Defenses against evasion attacks in classifiers typically fall into three categories: adversarial training [136], which augments training data with adversarial examples; input transformations [262] to suppress perturbations; and query monitoring to detect attack patterns [118]. Yet many of these defenses degrade accuracy or fail under adaptive attacks [29].

2.2.2 *Poisoning Attacks and Defenses*

Poisoning attacks manipulate model behavior by tampering with the training dataset. In classification, attackers may embed malicious samples that induce misclassification of specific inputs (targeted poisoning), reduce accuracy (indiscriminate poisoning), or implant backdoors that activate under specific triggers [81, 132]. Clean-label poisoning poses an even greater threat by using semantically valid, correctly labeled inputs to shift the model’s decision boundary without triggering anomaly filters [197]. Generative models are equally vulnerable. Because state-of-the-art text-to-image systems are trained on billions of scraped image-caption pairs from minimally curated sources like LAION-5B [193], poisoned content can be injected by simply posting on the open web.

Defenses against poisoning remain limited. In classification, backdoor detection methods include trigger synthesis [240], neuron pruning [74], and dataset filtering via spectral analysis [229]. Clean-label poisons are harder to detect since their labels are correct and their features subtle. Few defenses exist beyond influence estimation or anomaly detection [164]. In generative models, data collection filters focus on text-image alignment and caption length [193], but offer minimal protection against stealthy poisons. Even commercial datasets routinely include copyrighted, private, or manipulated content. Once trained, generative models are difficult to audit and sanitize, especially as continual training becomes standard practice [154, 214].

CHAPTER 3

FAWKES: PROTECTING PRIVACY AGAINST UNAUTHORIZED DEEP LEARNING MODELS

3.1 Introduction

Today’s proliferation of powerful facial recognition models poses a real threat to personal privacy. Facial recognition systems scan millions of citizens in both the UK and China without explicit consent [145, 189]. At many US airports, international travelers must submit to facial recognition systems in order to enter the country [155]. Perhaps more importantly, anyone with moderate resources can now canvas the Internet and build highly accurate facial recognition models of us without our knowledge or awareness, *e.g.* MegaFace [93]. Perhaps the most egregious example of this is *Clearview.ai*, a private company that collected more than 3 billion online photos and trained a massive model capable of recognizing millions of citizens, without their knowledge or consent [3].

Opportunities for misuse of this technology are numerous and potentially disastrous. Anywhere we go, we can be identified at any time through street cameras, video doorbells, security cameras, and personal cellphones. Stalkers can find out our identity and social media profiles with a single snapshot [211]. Stores can associate our in-store shopping behavior with online ads and browsing profiles [138]. Identity thieves can identify (and perhaps access) our personal accounts [49].

We believe that private citizens need tools to protect themselves from being identified by unauthorized facial recognition models. Such tools could increase individuals’ *data agency* in the context of facial recognition systems, enabling more fine-grained control over whether their images are used for face recognition. Unfortunately, previous work in this space is sparse and limited in both practicality and efficacy. Some have proposed distorting images to make them unrecognizable and thus avoiding facial recognition [258, 121, 221]. Others produce adversarial patches in the form of bright patterns printed on sweatshirts or signs, which prevent facial recognition algorithms from registering their wearer as a person [259, 228]. Finally, given access to an image classification

model, “clean-label poison attacks” can cause the model to misidentify a single image [197, 281].

Instead, we propose *Fawkes*, a system that helps individuals to inoculate their images against unauthorized facial recognition models at any time without significantly distorting their own photos or wearing conspicuous patches. Fawkes achieves this by helping users adding imperceptible pixel-level changes (“cloaks”) to their own photos. For example, a user who wants to share photos on social media or the public web can add small, imperceptible alterations to their photos before uploading them. If collected by a third-party “tracker” and used to train a facial recognition model to recognize the user, these “cloaked” images would produce functional models that consistently misidentify them. Fawkes is a disruptive data agency tool, allowing users to control use of their data by disrupting downstream machine learning applications.

Our distortion or “cloaking” algorithm takes the user’s photos and computes minimal perturbations that shift them significantly in the feature space of a facial recognition model (using real or synthetic images of a third party as a landmark). Any facial recognition model trained using these images of the user learns an altered set of “features” of what makes them look like them. When presented with a clean, uncloaked image of the user, *e.g.* photos from a camera phone or streetlight camera, the model finds no labels associated with the user in the feature space near the image, and classifies the photo to another label (identity) nearby in the feature space.

Our exploration of Fawkes produces several key findings:

- We can **produce significant alterations to images’ feature space representations** using perturbations imperceptible to the naked eye ($\text{DSSIM} \leq 0.007$).
- Regardless of how the tracker trains its model (via transfer learning or from scratch), **image cloaking provides 95+ % protection against user recognition** (adversarial training techniques help ensure cloaks transfer to tracker models).
- Experiments show **100% success against state-of-the-art facial recognition services from Microsoft (Azure Face API), Amazon (Rekognition), and Face++**. We first “share” our own (cloaked) photos as training data to each service, then apply the resulting models to uncloaked

test images of the same person.

- In challenging scenarios where clean, uncloaked images are “leaked” to the tracker and used for training, we show how **a single Sybil identity can boost privacy protection**. This results in 80+% success in avoiding identification even when half of the training images are uncloaked.
- Finally, we consider a tracker who is aware of our image cloaking techniques and evaluate the efficacy of potential countermeasures. We show that **image cloaks are robust (maintain high protection rates against) to a variety of mechanisms for cloak disruption and detection**.

3.2 Background and Related Work

To protect user privacy, our image cloaking techniques leverage and extend work broadly defined as poisoning attacks in machine learning. Here, we set the context by discussing prior efforts to help users evade facial recognition models. We then discuss relevant data poisoning attacks, followed by related work on privacy-preserving machine learning and techniques to train facial recognition models.

Note that to protect user privacy from unauthorized deep learning models, we employ attacks against ML models. In this scenario, *users* are the “attackers,” and third-party *trackers* running unauthorized tracking are the “targets.”

3.2.1 Protecting Privacy via Evasion Attacks

Privacy advocates have considered the problem of protecting individuals from facial recognition systems, generally by making images difficult for a facial recognition model to recognize. Some rely on creating *adversarial examples*, inputs to the model designed to cause misclassification [223]. These attacks have since been proven possible “in the wild,” Sharif *et al.* [203] create specially printed glasses that cause the wearer to be misidentified. Komkov and Petiushko [109] showed that carefully computed adversarial stickers on a hat can reduce its wearer’s likelihood of being recognized. Others propose “adversarial patches” that target “person identification” models,

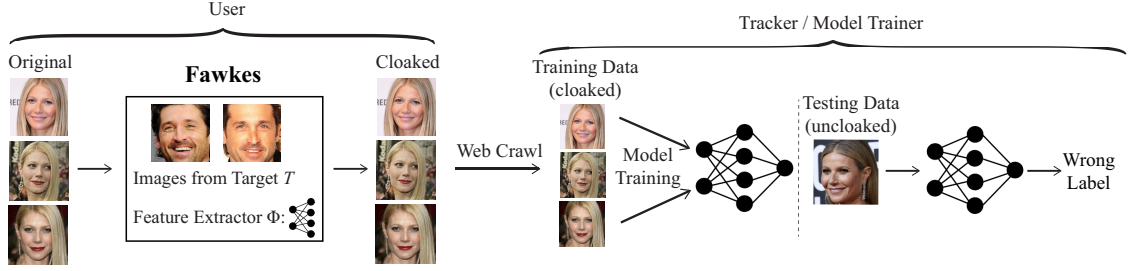


Figure 3.1: Our proposed Fawkes system that protects user privacy by cloaking their online photos. (Left) A user U applies cloaking algorithm (given a feature extractor Φ and images from some target T) to generate cloaked versions of U ’s photos, each with a small perturbation unnoticeable to the human eye. (Right) A tracker crawls the cloaked images from online sources, and uses them to train an (unauthorized) model to recognize and track U . When it comes to classifying new (uncloaked) images of U , the tracker’s model misclassifies them to someone not U . Note that T does not have to exist in the tracker’s model.

making it difficult for models to recognize the wearer as a person in an image [259, 228].

All of these approaches share two limitations. First, they require the user to wear fairly obvious and conspicuous accessories (hats, glasses, sweaters) that are impractical for normal use. Second, in order to evade tracking, they require *full and unrestricted* access (white box access) to the precise model tracking them. Thus they are easily broken (and user privacy compromised) by any tracker that updates its model.

Another line of work seeks to *edit facial images* so that human-like characteristics are preserved but facial recognition model accuracy is significantly reduced. Methods used include k-means facial averaging [149], facial inpainting [220], and GAN-based face editing [258, 121, 221]. Since these dramatically alter the user’s face in photos, they are impractical for protecting shared content.

3.2.2 Protecting Privacy via Poisoning Attacks

An alternative to evading models is to disrupt their training. This approach leverages “data poisoning attacks” against deep learning models. These attacks affect deep learning models by modifying the initial data used to train them, usually by adding a set of samples S and associated labels L_S . Previous work has used data poisoning to induce unexpected behaviors in trained DNNs [263].

In this section, we discuss two data poisoning attacks related to our work, and identify their key limitations when used to protect user privacy.

Clean Label Attacks. A clean-label poisoning attack injects “correctly” labeled poison images into training data, causing a model trained on this data to misclassify a specific image of interest [197, 281]. What distinguishes clean-label attacks from normal poisoning attacks is that all image labels remain unchanged during poisoning—only the content of poisoned images changes.

Fawkes works with similar constraints. Our action to affect or disrupt a model is limited to altering a group of images with a correct label, *i.e.* a user can alter her images but cannot claim these are images of someone else.

Current clean label attacks cannot address the privacy problem because of three factors. *First*, they only cause misclassification on a *single, preselected* image, whereas user privacy protection requires the misclassification of any current or future image of the protected user (*i.e.* an entire model class). *Second*, clean label attacks do not transfer well to different models, especially models trained from scratch. Even between models trained on the same data, the attack only transfers with 30% success rate [281]. *Third*, clean label attacks are easily detectable through anomaly detection in the feature space [84].

Model Corruption Attacks. Other recent work proposes techniques to modify images such that they *degrade* the accuracy of a model trained on them [206]. The goal is to spread these poisoned images in order to discourage unauthorized data collection and model training. We note that Fawkes’ goals are to mislead rather than frustrate. Simply corrupting data of a user’s class may inadvertently inform the tracker of the user’s evasion attempts and lead to more advanced countermeasures by the tracker. Finally, [206] only has a 50% success rate in protecting a user from being recognized.

3.2.3 Other Related Work

Privacy-Preserving Machine Learning. Recent work has shown that ML models can memorize

(and subsequently leak) parts of their training data [213]. This can be exploited to expose private details about members of the training dataset [72]. These attacks have spurred a push towards *differentially private* model training [11], which uses techniques from the field of differential privacy [61] to protect sensitive characteristics of training data. We note these techniques imply a trusted model trainer and are ineffective against an unauthorized model trainer.

Feature Extractors & Transfer Learning. Transfer learning uses existing pretrained models as a basis for quickly training models for customized classification tasks, using less training data. Today, it is commonly used to deploy complex ML models (*e.g.* facial recognition or image segmentation [270]) at reasonable training costs.

In transfer learning, the knowledge of a pre-trained feature extractor Φ is passed on to a new model \mathbb{F}_θ . Typically, a model \mathbb{F}_θ can be created by appending a few additional layers to Φ and only training those new layers. The original layers that composed Φ will remain unmodified. As such, pre-existing knowledge “learned” by Φ is passed on to the model \mathbb{F}_θ and directly influences its classification outcomes. Finally, transfer learning is most effective when the feature extractor and model are trained on similar datasets. For example, a facial recognition model trained on faces extracted from YouTube videos might serve well as a feature extractor for a model designed to recognize celebrities in magazines.

Finally, the concept of protecting individual privacy against invasive technologies extends beyond the image domain. Recent work [41] proposes wearable devices that restore personal agency using digital jammers to prevent audio eavesdropping by ubiquitous digital home assistants.

3.3 Protecting Privacy via Cloaking

We propose *Fawkes*, a system designed to help protect the privacy of a *user* against unauthorized facial recognition models trained by a third-party *tracker* on the user’s images. Fawkes achieves this by adding subtle perturbations (“cloaks”) to the user’s images before sharing them. Facial recognition models trained on cloaked images will have a distorted view of the user in the “feature space,”

i.e. the model’s internal understanding of what makes the user unique. Thus the models cannot recognize real (uncloaked) images of the user, and instead, misclassify them as someone else.

In this section, we first describe the threat model and assumptions for both users and trackers. We then present the intuition behind cloaking and our methodology to generate cloaks. Finally, we discuss why cloaking by individuals is effective against unauthorized facial recognition models.

3.3.1 Assumptions and Threat Model

User. The user’s goal is to share their photos online without unknowingly helping third party trackers build facial recognition models that can recognize them. Users protect themselves by adding imperceptible perturbations (“cloaks”) to their photos before sharing them. This is illustrated in the left part of Figure 3.1. The design goals for these cloaks are:

- cloaks should be **imperceptible** and not impact normal use of the image;
- when classifying normal, uncloaked images, models trained on cloaked images should recognize the underlying person with **low accuracy**.

We assume the user has access to moderate computing resources (e.g., a personal laptop) and applies cloaking to their own images locally. We also assume the user has access to a feature extractor, *e.g.* a generic face recognition model, represented as Φ in Figure 3.1. Cloaking is simplified if the user has the same Φ as the tracker. We begin with this common assumption (also used by prior work [241, 197, 281]), since only a few large-scale face recognition models are available in the wild. In §3.3.4, we relax this assumption and show how our design maintains the above properties.

We initially consider the case where the user can cloak all their photos to be shared, thus the tracker can only collect cloaked photos of the user. Later in §3.7, we explore a scenario where a stronger tracker has obtained access to some number of the user’s uncloaked images.

Tracker/Model Trainer. We assume that the tracker (the entity training unauthorized models) is a third party without direct access to user’s personal photos (*i.e.* not Facebook or Flickr). The tracker could be a company like Clearview.ai, a government entity, or even an individual. The tracker has

significant computational resources. They can either use transfer learning to simplify their model training process (leveraging existing feature extractors), or train their model from scratch.

We also assume the tracker’s primary goal is to build a powerful model to track many users rather than targeting a single specific person¹. The tracker’s primary data source is a collection of public images of users obtained via web scraping. We also consider scenarios where they are able to obtain some number of uncloaked images from other sources (§3.7).

Real World Limitations. Privacy benefits of Fawkes rely on users applying our cloaking technique to the majority of images of their likeness before posting online. In practice, however, users are unlikely to control all images of themselves, such as photos shared online by friends and family, media, employer or government websites. While it is unclear how easy or challenging it will be for trackers to associate these images with the identity of the user, a tracker who obtains a large number of uncloaked images of the user can compromise the effectiveness of Fawkes.

Therefore, Fawkes is most effective when used in conjunction with other privacy-enhancing steps that minimize the online availability of a user’s uncloaked images. For example, users can curate their social media presence and remove tags of their names applied to group photos on Facebook or Instagram. Users can also leverage privacy laws such as the “Right to be Forgotten” to remove and untag online content related to themselves. The online curation of personal images is a challenging problem, and we leave efforts minimizing online image footprints as future work.

3.3.2 Overview and Intuition

DNN models are trained to identify and extract (often hidden) *features* in input data and use them to perform classification. Yet their ability to identify features is easily disrupted by data poisoning attacks during model training, where small perturbations on training data with a particular label (l) can shift the model’s view of what features uniquely identify l [197, 281]. Our work leverages this

1. Tracking a specific person can be easily accomplished through easier, offline methods, *e.g.* a private investigator who follows the target user, and is beyond the scope of our work.

property to cause misclassification of *any existing or future image* of a single class, providing one solution to the challenging problem of protecting personal privacy against the unchecked spread of facial recognition models.

Intuitively, our goal is to protect a user’s privacy by modifying their photos in small and imperceptible ways before posting them online, such that a facial recognition model trained on them learns the wrong features about what makes the user look like the user. The model thinks it is successful, because it correctly recognizes its sample of (modified) images of the user. However, when unaltered images of the user, *e.g.* from a surveillance video, are fed into the model, the model does not detect the features it associates with the user. Instead, it identifies someone else as the person in the video. By simply modifying their online photos, the user successfully prevents unauthorized trackers and their DNN models from recognizing their true face.

3.3.3 Computing Cloak Perturbations

But how do we determine what perturbations (we call them “cloaks”) to apply to Alice’s photos? An effective cloak would teach a face recognition model to associate Alice with erroneous features that are quite different from real features defining Alice. Intuitively, the more dissimilar these erroneous features are from the real Alice, the less likely the model will be able to recognize her.

In the following, we describe our methodology for computing cloaks for each specific user, with the goal of making the features learned from cloaked photos highly dissimilar from those learned from original (uncloaked) photos.

Notation. Our discussion will use the following notations.

- x : Alice’s image (uncloaked)
- x_T : target image (image from another class/user T) used to generate cloak for Alice
- $\delta(x, x_T)$: cloak computed for Alice’s image x based on an image x_T from label T
- $x \oplus \delta(x, x_T)$: cloaked version of Alice’s image x
- Φ : Feature extractor used by facial recognition model

- $\Phi(x)$: Feature vector (or feature representation) extracted from an input x

Cloaking to Maximize Feature Deviation. Given each photo (x) of Alice to be shared online, our ideal cloaking design modifies x by adding a cloak perturbation $\delta(x, x_T)$ to x that maximize changes in x 's feature representation:

$$\begin{aligned} \max_{\delta} \text{Dist}(\Phi(x), \Phi(x \oplus \delta(x, x_T))), \\ \text{subject to } |\delta(x, x_T)| < \rho, \end{aligned} \tag{3.1}$$

where $\text{Dist}(\cdot)$ computes the distance of two feature vectors, $|\delta|$ measures the perceptual perturbation caused by cloaking, and ρ is the perceptual perturbation budget.

To guide the search for the cloak perturbation in eq (3.1), we use another image x_T from a different user class (T). Since the feature space Φ is highly complex, x_T serves as a landmark, enabling fast and efficient search for the input perturbation that leads to large changes in feature representation. Ideally, T should be very dissimilar from Alice in the feature space. We illustrate this in Figure 3.1, where we use Patrick Dempsey (a male actor) as a dissimilar target T for the original user (female actress Gwyneth Paltrow).

We note that our design does not assume that the cloak target (T) and the associated x_T are used by any tracker's face recognition model. In fact, any user whose feature representation is sufficiently different from Alice's would suffice (see §3.3.4). Alice can easily check for such dissimilarity by running the feature extractor Φ on other users' online photos. Later in §3.4 we will present the detailed algorithm for choosing the target user T from public datasets of facial images.

Image-specific Cloaking. When creating cloaks for her photos, Alice will produce image-specific cloaks, *i.e.* $\delta(x, x_T)$ is image dependent. Specifically, Alice will pair each original image x with a target image x_T of class T . In our current implementation, the search for $\delta(x, x_T)$ replaces

the ideal optimization defined by eq. (3.1) with the following optimization:

$$\begin{aligned} \min_{\delta} \text{Dist}(\Phi(x_T), \Phi(x \oplus \delta(x, x_T))), \\ \text{subject to } |\delta(x, x_T)| < \rho. \end{aligned} \quad (3.2)$$

Here we search for the cloak for x that shifts its feature representation closely towards x_T . This new form of optimization also prevents the system from generating extreme $\Phi(x \oplus \delta(x, x_T))$ values that can be easily detected by trackers using anomaly detection.

Finally, our image-specific cloak optimization will create different cloak patterns among Alice’s images. This “diversity” makes it hard for trackers to detect and remove cloaks.

3.3.4 Cloaking Effectiveness & Transferability

Now a user (Alice) can produce cloaked images whose feature representation is dissimilar from her own but similar to that of a target user T . But does this translate into the desired misclassification behavior in the tracker model? Clearly, if T is a class in the tracker model, Alice’s original (uncloaked) images will not be classified as Alice. But under the more likely scenario where T is not in the tracker model, does cloaking still lead to misclassification?

We believe the answer is **yes**. Our hypothesis is that as long as the feature representations of Alice’s cloaked and uncloaked images are sufficiently different, the tracker’s model will not classify them as the same class. This is because there will be another user class (*e.g.* B) in the tracker model, whose feature representation is more similar to $\Phi(x)$ (true Alice) than $\Phi(x \oplus \delta)$ (Alice learned by the model). Thus, the model will classify Alice’s normal images as B .

We illustrate this in Figure 3.2 using a simplified 2D visualization of the feature space. There are 4 classes (A , B , U aka Alice, and T) that a tracker wishes to distinguish. The two figures show the tracker model’s decision boundary when U ’s training data is uncloaked and cloaked, respectively. In Figure 3.2(a), the model will learn U ’s true feature representation as the bottom

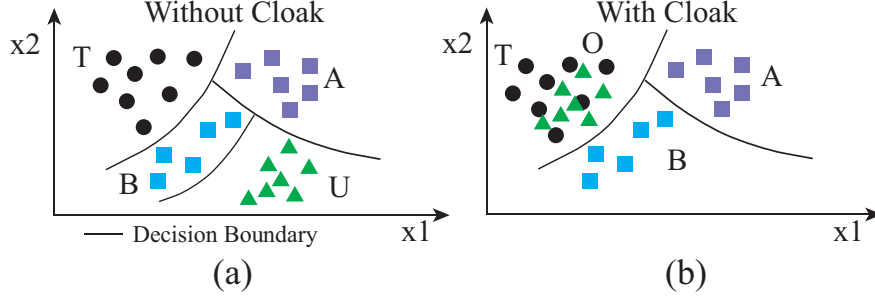


Figure 3.2: The intuition for why a tracker’s model trained on U ’s cloaked photos will misclassify U ’s original photos, visualized on a simplified 2D feature space with four user classes A , B , U (aka Alice), T . (a) decision boundaries of the model trained on U ’s uncloaked photos. (b) decision boundaries when trained on U ’s cloaked photos (with target T).

right corner. In Figure 3.2(b), U uses T as the cloak target, and the resulting tracker model will learn U ’s feature representation $\Phi(x \oplus \delta)$ as green triangles near T (top left corner). This means that the area corresponding to U ’s original feature representation $\Phi(x)$ will be classified as B . More importantly, this (mis)classification will occur whether or not T is a class in the tracker’s model.

This discussion assumes the tracker’s model contains a class whose feature representation is more similar to the user’s original feature representation than her cloaked feature representation. This is a reasonable assumption when the tracker’s model targets many users (*e.g.* 1,000) rather than a few users (*e.g.* 2). Later in §3.5 we confirm that cloaking is highly effective against multiple facial recognition models with anywhere from 65 to 10,575 classes.

Transferability. Our above discussion also assumes that the user has the same feature extractor Φ as is used to train the tracker model. Under the more general scenario, the effectiveness of cloaking against any tracker models relies on the *transferability* effect, the property that models trained for similar tasks share similar properties and vulnerabilities, even when they were trained on different architectures and different training data [54, 157, 219, 270].

This transferability property suggests that cloaking should still be effective even if the tracker performs transfer learning using a different feature extractor or trains their model from scratch. Because the user’s and tracker’s feature extractors/models are designed for similar tasks (*i.e.* facial recognition), cloaks should be effective regardless of the tracker’s training method. Later, we

empirically evaluate cloaking success rate when trackers use different feature extractors (§3.5.3) or train models from scratch (§3.5.4). In all scenarios, cloaking is effective ($> 95\%$ protection rate).

3.4 The Fawkes Image Cloaking System

We now present the detailed design of *Fawkes*, a practical image cloaking system that allows users to evade identification by unauthorized facial recognition models. Fawkes uses three steps to help a user modify and publish her online photos.

Given a user U , Fawkes takes as input the set of U 's photos to be shared online \mathbf{X}_U , the (generic) feature extractor Φ , and the cloak perturbation budget ρ .

Step 1: Choosing a Target Class T . First, Fawkes examines a publicly available dataset that contains numerous groups of images, each identified with a specific class label, *e.g.* Bob, Carl, Diana. Fawkes randomly picks K candidate target classes and their images from this public dataset and uses the feature extractor Φ to calculate \mathcal{C}_k , the centroid of the feature space for each class $k = 1..K$. Fawkes picks as the target class T the class in the K candidate set whose feature representation centroid is most dissimilar from the feature representations of all images in \mathbf{X}_U , *i.e.*

$$T = \underset{k=1..K}{\operatorname{argmax}} \min_{x \in \mathbf{X}_U} \operatorname{Dist}(\Phi(x), \mathcal{C}_k). \quad (3.3)$$

We use L2 as the distance function in feature space, $\operatorname{Dist}(\cdot)$.

Step 2: Computing Per-image Cloaks. Let \mathbf{X}_T represent the set of target images available to user U . For each image of user U , $x \in \mathbf{X}_U$, Fawkes randomly picks an image $x_T \in \mathbf{X}_T$, and computes a cloak $\delta(x, x_T)$ for x , following the optimization of eq. (3.2), subject to $|\delta(x, x_T)| < \rho$.

In our implementation, $|\delta(x, x_T)|$ is calculated using the DSSIM (Structural Dis-Similarity Index) [244, 245]. Different from the L_p distance used in previous work [30, 111, 201], DSSIM has gained popularity as a measure of user-perceived image distortion [241, 99, 124]. Bounding cloak generation with this metric ensures that cloaked images are visually similar to the originals.

Teacher Dataset	Model Architecture	Abbreviation	Teacher Testing Accuracy	Student Testing Accuracy	
				PubFig	FaceScrub
WebFace	InceptionResNet	Web-Incept	74%	96%	92%
WebFace	DenseNet	Web-Dense	76%	96%	94%
VGGFace2	InceptionResNet	VGG2-Incept	81%	95%	90%
VGGFace2	DenseNet	VGG2-Dense	82%	96%	92%

Table 3.1: The four feature extractors used in our evaluation, their classification efficacy and those of their student models.

We apply the *penalty method* [153] to solve the optimization in eq.(3.2) as follows:

$$\min_{\delta} Dist(\Phi(x_T), \Phi(x \oplus \delta(x, x_T))) + \lambda \cdot \max(|\delta(x, x_T)| - \rho, 0)$$

Here λ controls the impact of the input perturbation caused by cloaking. When $\lambda \rightarrow \infty$, the cloaked image is visually identical to the original image. Finally, to ensure the input pixel intensity remains in the correct range $([0, 255])$, we transform the intensity values into *tanh* space as proposed in previous work [30].

Step 3: Limiting Content. Now the user U has created the set of cloaked images that she can post and share online. However, the user must be careful to ensure that no uncloaked images are shared online and associated with her identity. Any images shared by friends and labeled or tagged with her name would provide uncloaked training data for a tracker model. Fortunately, a user can proactively “untag” herself on most photo sharing sites.

Even so, a third party might be able to restore those labels and re-identify her in those photos using friendlist intersection attacks [256]. Thus, in §3.7, we expand the design of Fawkes to address trackers who are able to obtain uncloaked images in addition to cloaked images of the user.

3.5 System Evaluation

In this section, we evaluate the effectiveness of Fawkes. We first describe the datasets, models, and experimental configurations used in our tests. We then present results for cloaking in three

Dataset	# of Labels	Input Size	# of Training Images
PubFig	65	$224 \times 224 \times 3$	5,850
FaceScrub	344	$224 \times 224 \times 3$	37,905
WebFace	10,575	$224 \times 224 \times 3$	475,137
VGGFace2	8,631	$224 \times 224 \times 3$	3,141,890

Table 3.2: Datasets emulating user images in experiments.

different scenarios: 1) the user produces cloaks using the same feature extractor as the tracker; 2) the user and tracker use different feature extractors; and 3) the tracker trains models from scratch (no feature extractor).

Our key findings are: cloaking is highly effective when users share a feature extractor with the tracker; efficacy could drop when feature extractors are different, but can be restored to near perfection by making the user’s feature extractor robust (via adversarial training); and, similarly, cloaks generated on robust feature extractors work well even when trackers train models from scratch.

3.5.1 Experiment Setup

Our experiments require two components. First, we need feature extractors that form the basis of facial recognition models for both the user’s cloaking purposes and the tracker’s model training. Second, we need datasets that emulate a set of user images scraped by the tracker and enable us to evaluate the impact of cloaking.

Feature Extractors. There are few publically available, large-scale facial recognition models. Thus we train feature extractors using two large ($\geq 500\text{K}$ images) datasets on different model architectures (details in Table 3.2).

- VGGFace2 contains 3.14M images of 8,631 subjects downloaded from Google Image Search [26].
- WebFace has 500,000 images of faces covering roughly 10,000 subjects from the Internet [269].

Using these two datasets, we build four feature extractors, two from each. We use two different model architectures: a) DenseNet-121 [96], a 121 layer neural network with 7M parameters, and b)

InceptionResNet V2 [222], a 572 layer deep neural network with over 54M parameters. Our trained models have comparable accuracy with previous work [26, 241, 147] and perform well in transfer learning scenarios. For clarity, we abbreviate feature extractors based on their dataset/architecture pair. Table 3.1 lists the classification accuracy for our feature extractors and student models.

Tracker’s Training Datasets. Under the scenario where the tracker trains its facial recognition model from scratch (§3.5.4), we assume they will use the above two large datasets (VGGFace2, WebFace). Under the scenario where they apply transfer learning (§3.5.2 and §3.5.3), the tracker uses the following two smaller datasets (more details in Table 3.2).

- PubFig contains 5,850 training images and 650 testing images of 65 public figures² [5].
- FaceScrub contains 100,000 images of 530 public figures on the Internet [150]³.

To perform transfer learning, the tracker adds a softmax layer at the end of the feature extractor (see §3.2.3), and fine-tunes the added layer using the above dataset.

Cloaking Configuration. In our experiments, we randomly choose a user class U in the tracker’s model, *e.g.* a random user in PubFig, to be the user seeking protection. We then apply the target selection algorithm described in §3.4 to select a target class T from a small subset of users in VGGFace2 and WebFace. Here we ensure that T is not a user class in the tracker’s model.

For each given U and T pair, we pair each image x of U with an image x_T from T , and compute the cloak for x . For this we run the Adam optimizer for 1000 iterations with a learning rate of 0.5.

As discussed earlier, we evaluate our cloaking under three scenarios, U and tracker model sharing the same feature extractor (§3.5.2), the two using different feature extractors (§3.5.3), and the tracker training model from scratch without using any pre-defined feature extractor (§3.5.4).

Evaluation Metrics. In each scenario, we evaluate cloak performance using two metrics: *protection success rate*, which is the tracker model’s misclassification rate for clean (uncloaked) images

2. We exclude 18 celebrities also used in the feature extractor datasets.

3. We could only download 60,882 images for 530 people, as some URLs were removed. Similarly, prior work [266] only retrieved 48,579 images.

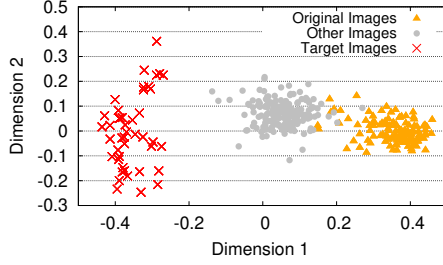


Figure 3.3: Before Cloaking

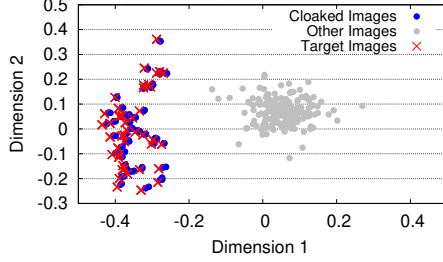


Figure 3.4: After Cloaking

Figure 3.5: 2-D PCA visualization of VGG2-Dense feature space representations of user images (sampled from FaceScrub) before/after cloaking. Triangles are user’s images, red crosses are target images, grey dots are images from another class.

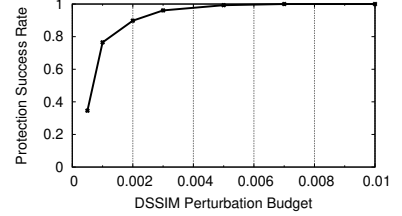


Figure 3.6: Protection performance as DSSIM perturbation budget increases. (User/Tracker: Web-Incept)

of U , and *normal accuracy*, which is the overall classification accuracy of the tracker’s model on users beside U . When needed, we indicate the configuration of user/tracker feature extractors using the notation entity:feature extractor.

3.5.2 User/Tracker Sharing a Feature Extractor

We start from the simple case where the user uses the same feature extractor as the tracker to generate cloaks. We randomly select a label from PubFig or FaceScrub to be the Fawkes user U . We then compute “cloaks” for a subset of U ’s images, using each of the four feature extractors in Table 3.1. On the tracker side, we perform transfer learning on the *same* feature extractor (with cloaked images of U) to build a model that recognizes U . Finally, we evaluate whether the tracker model can correctly identify other *clean* images of U it has not seen before.

Results show that cloaking offers perfect protection, *i.e.* U is always misclassified as someone

else, for all four feature extractors and under the perturbation budget $\rho = 0.007$. To explore the impact of ρ , Figure 3.6 plots protection success rate vs. ρ when the tracker runs on the `FaceScrub` dataset. Fawkes achieves 100% protection success rate when $\rho > 0.005$. Figure 3.7 shows original and cloaked images, demonstrating that cloaking does not visually distort the original image. Even when $\rho = 0.007$, the perturbation is barely detectable by the naked eye on a full size, color image. For calibration, note that prior work [124] claims much higher DSSIM values (up to 0.2) are imperceptible to the human eye. Finally, the average $L2$ norm of our cloaks is 5.44, which is smaller than that of perturbations used in prior works [241, 131].

Feature Space Deviation. The goal of a cloak is to change the image’s feature space representation in the tracker’s model. To examine the effect of the cloak in the tracker model, we visualize feature space representations of user images before and after cloaking, their chosen target images, and a randomly chosen class from the tracker’s dataset. We use principal components analysis (PCA, a common dimensionality reduction technique) to reduce the high dimensional feature space to 2 dimensions. Figure 3.5 shows the PCA results for cloaked images from a `PubFig` class, using cloaks constructed on the `Web-Incept` feature extractor. Figure 3.5(a) shows the feature space positions of the original and target images before cloaking, along with a randomly selected class. Figure 3.5(b) shows the updated feature space after the original images have been cloaked. It is clear that feature space representations of the cloaked images are well-aligned with those of the target images, validating our intuition for cloaking (an abstract view in Figure 3.2).

Impact of Label Density. As discussed in §3.3, the number of labels present in the tracker’s model impacts performance. When the tracker targets fewer labels, the feature space is “sparser,” and there is a greater chance the model continues to associate the original feature space (along with the cloaked feature space) with the user’s label. We empirically evaluate the impact of fewer labels on cloaking success using the `PubFig` and `FaceScrub` datasets (65 and 530 labels, respectively). We randomly sample N labels (varying N from 2 to 10) to construct a model with fewer labels. Figure 3.8 shows that for `PubFig`, cloaking success rate grows from 68% for 2 labels to $> 99\%$



Figure 3.7: Pairs of original and cloaked images ($\rho = 0.007$).

for more than 6 labels, confirming that a higher label density improves cloaking effectiveness.

3.5.3 User/Tracker Using Different Feature Extractors

We now consider the scenario when the user and tracker use different feature extractors to perform their tasks. While the model transferability property suggests that there are significant similarities in their respective model feature spaces (since both are trained to recognize faces), their differences could still reduce the efficacy of cloaking. Cloaks that shift image features significantly in one feature extractor may produce a much smaller shift in a different feature extractor.

To illustrate this, we empirically inspect the change in feature representation between two different feature extractors. We take the cloaked images (optimized using VGG2-Dense), original images, and target images from the PubFig dataset and calculate their feature representations in a *different* feature extractor, Web-Incept. The result is visualized using two dimensional PCA and shown in Figure 3.9. From the PCA visualization, the reduction in cloak effectiveness is obvious. In the tracker’s feature extractor, the cloak “moves” the original image features only slightly towards the target image features (compared to Figure 3.5(b)).

Robust Feature Extractors Boost Transferability. To address the problem of cloak transferability, we draw on recent work linking model robustness and transferability. Demontis *et al.* [54] argue that an input perturbation’s (in our case, cloak’s) ability to transfer between models depends on the “robustness” of the feature extractor used to create it. They show that more “robust” models are less reactive to small perturbations on inputs. Furthermore, they claim that perturbations (or,

User’s Robust Feature Extractor	Model Trainer’s Feature Extractor							
	VGG2-Incept		VGG2-Dense		Web-Incept		Web-Dense	
	PubFig	FaceScrub	PubFig	FaceScrub	PubFig	FaceScrub	PubFig	FaceScrub
VGG2-Incept	100%	100%	100%	100%	95%	100%	100%	100%
VGG2-Dense	100%	100%	100%	100%	100%	100%	100%	100%
Web-Incept	100%	100%	100%	100%	100%	100%	99%	99%
Web-Dense	100%	100%	100%	100%	100%	97%	100%	96%

Table 3.3: Protection performance of cloaks generated on robust feature extractors.

again, cloaks) generated on more robust models will take on “universal” characteristics that are able to effectively fool other models.

Following this intuition, we propose to improve cloak transferability by increasing the user feature extractor’s robustness. This is done by applying *adversarial training* [136, 79], which trains the model on perturbed data to make it less sensitive to similar small perturbations on inputs. Specifically, for each feature extractor, we generate adversarial examples using the PGD attack [111], a widely used method for adversarial training. Following prior work [136], we run the PGD⁴ algorithm for 100 steps using a step size of 0.01. We train each feature extractor for an additional 10 epochs. These updated feature extractors are then used to generate user cloaks on the PubFig and FaceScrub datasets.

Results in Table 3.3 show that each robust feature extractor produces cloaks that transfer almost perfectly to the tracker’s models. Cloaks now have protection success rates $> 95\%$ when the tracker uses a different feature extractor. We visualize their feature representation using PCA in Figure 3.10 and see that, indeed, cloaks generated on robust extractors transfer better than cloaks computed on normal ones.

3.5.4 Tracker Models Trained from Scratch

Finally, we consider the scenario in which a powerful tracker trains their model from scratch. We select the user U to be a label inside the WebFace dataset. We generate cloaks on user

4. We found that robust models trained on CW attack samples [30] produce similar results

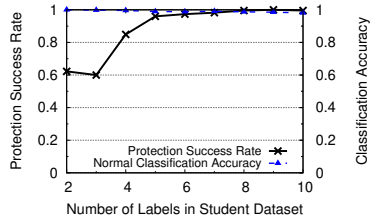


Figure 3.8: Protection performance improves as the number of labels in tracker’s model increases. (User/Tracker: Web-Incept)

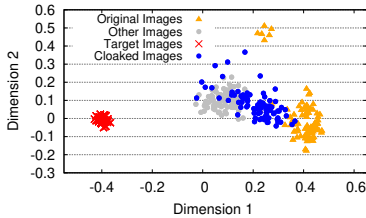


Figure 3.9: Cloaking is less effective when users and trackers use different feature extractors. (User: VGG2-Dense, Tracker: Web-Incept)

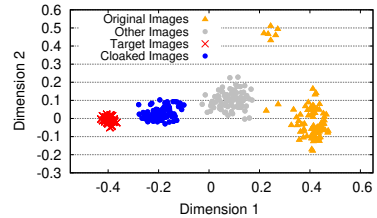


Figure 3.10: Cloaks generated on robust models transfer better between feature extractors. (User: VGG2-Dense, Tracker: Web-Incept)

images using the robust VGG2-Incept feature extractor from §3.5.3. The tracker then uses the WebFace dataset (but U ’s cloaked images) to train their model from scratch. Again our cloaks achieve a success rate of 100%. Other combinations of labels and user-side feature generators all have 100% protection success.

3.6 Image Cloaking in the Wild

Our results thus far have focused on limited configurations, including publicly available datasets and known model architectures. Now, we wish to understand the performance of Fawkes on deployed facial recognition systems in the wild.

We evaluate the real-world effectiveness of image cloaking by applying Fawkes to photos of one of the co-authors. We then intentionally leak a portion of these cloaked photos to public cloud-based services that perform facial recognition, including Microsoft Azure Face [2], Amazon Rekognition [1], and Face++ [6]. These are the global leaders in facial recognition and their services are used by businesses, police, private entities, and governments in the US and Asia.

3.6.1 Experimental Setup

We manually collected 82 high-quality pictures of a co-author that feature a wide range of lighting conditions, poses, and facial expressions. We separate the images into two subsets, one set of

Face Recognition API	Protection Success Rate		
	Without protection	Protected by normal cloak	Protected by robust cloak
Microsoft Azure Face API	0%	100%	100%
Amazon Rekognition Face Verification	0%	34%	100%
Face++ Face Search API	0%	0%	100%

Table 3.4: Cloaking is highly effective against cloud-based face recognition APIs (Microsoft, Amazon and Face++).

50 images for “training” and one set of 32 images for “testing.” We generate both normal and robust cloaks for the “training” images using the setup discussed in Section 3.5 (using normal and robust versions of the Web-Incept feature extractor). This allows us to compare the relative effectiveness of normal and robust user feature extractors in real life.

For each API service, we experiment with three scenarios:

- **Unprotected:** We upload original training images, and test the model’s classification accuracy on testing images.
- **Normal Cloak:** We upload training images protected by a *nonrobust* cloak and then test the model’s classification accuracy on the testing images.
- **Robust Cloak:** We upload training images protected by a *robust* cloak and test the model’s classification accuracy on the testing images.

For each scenario, we use the online service APIs to upload training images to the API database, and then query the APIs using the uncloaked testing images. The reported protection success rate is the proportion of uncloaked test images that the API fails to correctly identify as our co-author.

3.6.2 Real World Protection Performance

Microsoft Azure Face API. Microsoft Azure Face API [2] is part of Microsoft Cognitive Ser-

vices, and is reportedly used by many large corporations including Uber and Jet.com. The API provides face recognition services. A client uploads training images of faces, and Microsoft trains a model to recognize these faces. The API has a “training” endpoint that must be called before the model will recognize faces, which leads us to believe that Microsoft uses transfer learning to train a model on user-submitted images.

Our normal cloaking method is 100% effective against the Microsoft Azure Face API. Our robust cloaks also provide 100% protection against the Azure Face API. Detailed protection results are shown in Table 3.4.

Amazon Rekognition Face Verification. Amazon Rekognition [1] provides facial search services that the client can use to detect, analyze, and compare faces. The API is used by various large corporations including the NFL, CBS, and National Geographic, as well as law enforcement agencies in Florida and Oregon, and U.S. Immigration and Customs Enforcement (ICE).

It is important to note that Amazon Rekognition does not specifically train a neural network to classify queried images. Instead, it computes an image similarity score between the queried image and the ground truth images for all labels. If the similarity score exceeds a threshold for some label, Amazon returns a match. Our cloaking technique is not designed to fool a tracker who uses similarity matching. However, we believe our cloaking technique should still be effective against Amazon Rekognition, since cloaks create a feature space separation between original and cloaked images that should result in low similarity scores between them.

Table 3.4 shows that our normal cloaks only achieve a protection success rate of 34%. However, our robust cloaks again achieve a 100% protection success rate.

Face++. Face++ [6] is a well-known face recognition system developed in China that claims to be extremely robust against a variety of attacks (*i.e.* adversarial masks, makeup, etc.). Due to its high performance and perceived robustness, Face++ is widely used by financial services providers and other security-sensitive customers. Notably, Alipay uses Face++’s services to authenticate users before processing payments. Lenovo also uses Face++ services to perform face-based au-

thentication for laptop users.

Our results show that normal cloaking is completely ineffective against Face++ (0% protection success rate; see Table 3.4). This indicates that their model is indeed extremely robust against input perturbations. However, as before, our robust cloaks achieve a 100% success rate.

Summary. Microsoft Azure Face API, Amazon Rekognition and Face++ represent three of the most popular and widely deployed facial recognition services today. The success of Fawkes cloaking techniques suggests our approach is realistic and practical against production systems. While we expect these systems to continue improving, we expect cloaking techniques to similarly evolve over time to keep pace.

3.7 Trackers with Uncloaked Image Access

Thus far we have assumed that the tracker only has access to *cloaked images* of a user, *i.e.* the user is perfect in applying her cloaking protection to her image content, and disassociating her identity from images posted online by friends. In real life, however, this may be too strong an assumption. Users make mistakes, and unauthorized labeled images of the user can be taken and published online by third parties such as newspapers and websites.

In this section, we consider the possibility of the tracker obtaining leaked, uncloaked images of a target user, *e.g.* Alice. We first evaluate the impact of adding these images to the tracker’s model training data. We then consider possible mechanisms to mitigate this impact by leveraging the use of limited sybil identities online.

3.7.1 Impact of Uncloaked Images

Intuitively, a tracker with access to some labeled, uncloaked images of a user has a much greater chance of training a model M that successfully recognizes clean images of that user. Training a model with both cloaked and uncloaked user images means the model will observe a much larger spread of features all designated as the user. Depending on how M is trained and the pres-

ence/density of other labels, it can a) classify both regions of features as the user; b) classify both regions and the region between them as the user; or c) ignore these feature dimensions and identify the user using some alternative features (*e.g.* other facial features) that connect both uncloaked and cloaked versions of the user’s images.

We assume the tracker cannot visually distinguish between cloaked and uncloaked images and trains their model on both. We quantify the impact of training with uncloaked images using a simple test with cloaks generated from §3.5.2 and a model trained on both cloaked and uncloaked images. Figure 3.12 shows the drop in protection success for `FaceScrub` dataset as the ratio of uncloaked images in the training dataset increases. The protection success rate drops below 39% when more than 15% of the user’s images are uncloaked.

Next, we consider proactive mitigation strategies against leaked images. The most direct solution is to intentionally release more cloaked images, effectively flooding a potential tracker’s training set with cloaked images to dominate any leaked uncloaked images. In addition, we consider the use of a cooperating secondary identity (more details below). For simplicity, we assume that: trackers have access to a *small* number of a user’s uncloaked images; the user is unaware of the contents of the uncloaked images obtained by the tracker; and users know the feature extractor used by the tracker.

3.7.2 *Sybil Accounts*

In addition to proactive flooding of cloaked images, we explore the use of cooperative *Sybil accounts* to induce model misclassification. A Sybil account is a separate account controlled by the user that exists in the same Internet community (*i.e.* Facebook, Flickr) as the original account. Sybils already exist in numerous online communities [265], and are often used by real users to curate and compartmentalize content for different audiences [116]. While there are numerous techniques for Sybil detection, individual Sybil accounts are difficult to identify or remove [243].

In our case, we propose that privacy-conscious users create a secondary identity, preferably not

connected to their main identity in the metadata or access patterns. Its content can be extracted from public sources, from a friend, or even generated artificially via generative adversarial networks (GANs) [142]. Fawkes modifies Sybil images (in a manner similar to cloaking) to provide additional protection for the user’s original images. Since Sybil and user images reside in the same communities, we expect trackers will collect both. While there are powerful re-identification techniques that could be used to associate the Sybil back to the original user, we assume they are impractical for the tracker to apply at scale to its population of tracked users.

Sybil Intuition. To bolster cloaking effectiveness, the user modifies Sybil images so they occupy the same feature space as a user’s uncloaked images. These Sybil images help confuse a model trained on both Sybil images *and* uncloaked/cloaked images of a user, increasing the protection success rate. Figure 3.11 shows the high level intuition. Without Sybil images, models trained on a small portion of uncloaked (leaked) images would easily associate test images of the user with the user’s true label (left side). Because the leaked uncloaked images and Sybil images have similar feature space representations, but are labeled differently (*i.e.* “User 1” and “User 2”), the tracker model must create additional decision boundaries in the feature space (right side). These decrease the likelihood of associating the user with her original feature space.

For simplicity, we explore the base case where the user is able to obtain one single Sybil identity to perform feature space obfuscation on her behalf. Our technique becomes even more effective with multiple Sybils, but provides much of its benefit with images labeled with a single Sybil identity.

Creating Sybil images. Sybil images are created by adding a specially designed cloak to a set of candidate images. Let x_C be an image from the set of candidates the user obtains (*i.e.* images generated by a GAN) to populate the Sybil account. To create the final Sybil image, we create a cloak $\delta(x_C, x)$ that minimizes the feature space separation between x_C and user’s original image x , for each candidate. The optimization is equivalent to setting x as the target and optimizing to create $x_C \oplus \delta(x_C, x)$ as discussed in §3.4. After choosing the final x_c from all the candidates, a

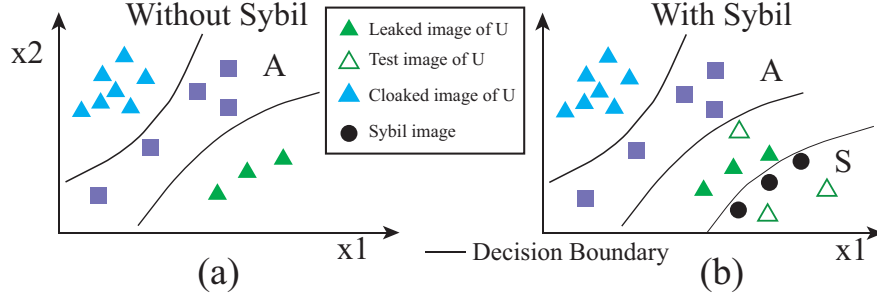


Figure 3.11: Intuition behind Sybil integration visualized in a 2D feature space. Without Sybils, a tracker’s model will use leaked training images of U to learn U ’s true feature space (left), leading to the correct classification of images of U . Sybil images S complicate the model’s decision boundary and cause misclassification of U ’s images, even when leaked images of U are present (right).

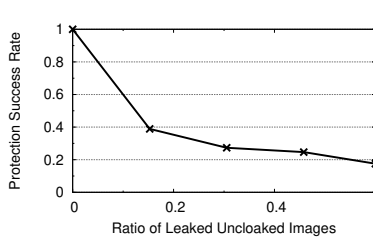


Figure 3.12: Protection success rate decreases when the tracker has more original user images. (User/Tracker: Web-Incept)

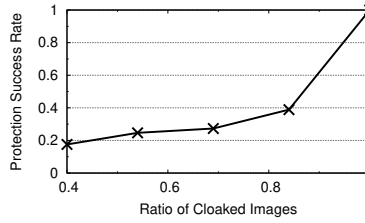


Figure 3.13: Protection success rate is high when the user has a Sybil account, even if tracker has original user images. (User/Tracker: Web-Incept)

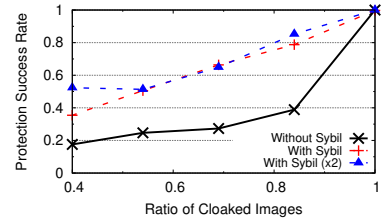


Figure 3.14: Sybils jointly optimized on four feature extractors have reasonably high protection success for each individual extractor.

ready-to-upload Sybil image $x_S = x_C \oplus \delta(x_C, x)$.

3.7.3 Efficacy of Sybil Images

Sybil accounts can increase a user’s protection success rate when the tracker controls a small number of a user’s uncloaked images. To experimentally validate this claim, we choose a label from the tracker’s dataset to be the Sybil account (controlled by the user), and split the user’s images into two disjoint sets: A contains images that were processed by Fawkes, and whose cloaked versions have been shared online; and B contains original images leaked to the tracker. For each synthetic image of the Sybil, we randomly select an uncloaked image of the user in set A . We select one Sybil image per uncloaked image in A . Then, we cloak all the candidate images using the methodology discussed in §3.4. The resulting Sybil images mimic the feature space representation of

uncloaked user images. From the tracker’s perspective, they have access to cloaked user images from set A , uncloaked images from set B , and the Sybil images.

Figure 3.13 compares the protection success rate with and without Sybil accounts (with Web-Incept as user’s and tracker’s feature extractor). The use of a Sybil account significantly improves the protection success rate when an attacker has a small number of original images. The protection success rate remains above 87% when less than 31% of the tracker’s images of the user are uncloaked.

As discussed, a user can create as many Sybil images as they desire. When the user uploads more Sybil images, the protection success rate increases. Figure 3.13 shows that when the user has uploaded 2 Sybil images per uncloaked image, the protection success rate increases by 5.5%.

Jointly Optimize Multiple Feature Extractors. The user may not know the tracker’s exact feature extractor. However, given the small number of face feature extractors available online, she is likely to know that the tracker would use one of several candidate feature extractors. Thus, she could jointly optimize the Sybil cloaks to simultaneously fool all the candidate feature extractors.

We test this in a simple experiment by jointly optimizing Sybil cloaks on the four feature extractors from §3.5. We evaluate the cloak’s performance when the tracker uses one of the four. Figure 3.14 shows the Sybil effectiveness averaged across the 4 feature extractors. The average protection success rate remains above 65% when the ratio of the original images owned by the tracker is less than 31%.

3.8 Countermeasures

In this section, we explore potential countermeasures a tracker could employ to reduce the effectiveness of image cloaking. We consider and (where possible) empirically validate methods to *remove* cloaks from images, as well as techniques to detect the presence of cloak perturbations on images. Our experiments make the strongest possible assumption about the tracker: that they know the precise feature extractor a user used to optimize cloaks. We test our countermeasures on a tracker’s model trained on the FaceScrub dataset. Cloaks were generated using the same

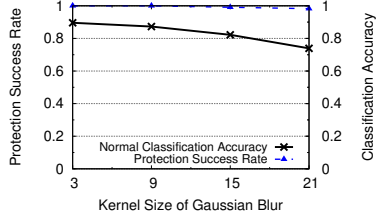


Figure 3.15: Normal classification accuracy decreases as input blurring increases but protection success rate remains high.

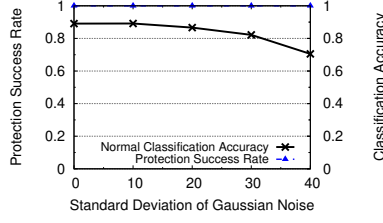


Figure 3.16: Normal classification accuracy decreases as Gaussian noise is added to inputs but protection success rate remains high.

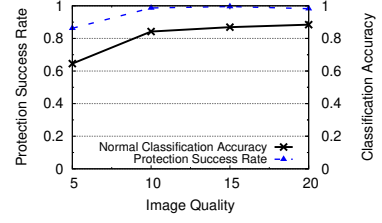


Figure 3.17: Protection success rate and normal classification accuracy increase as image quality increases using JPEG compression.

robust VGG2-Dense feature extractor from §3.5.3.

Inherent Limits on Cloaking Success. We acknowledge that cloaking becomes less effective when an individual is an *active target* of a tracker. If a tracker strongly desires to train a model that recognizes a certain individual, they can take drastic measures that cloaking cannot withstand. For example, a tracker could learn their movements or invade their privacy (*i.e.* learn where they live) by following them physically.

3.8.1 Cloak Disruption

Without knowing which images in the dataset are cloaked, the tracker may utilize the following techniques to disrupt Fawkes’ protection performance, 1) transforming images or 2) deploying an extremely robust model. We present and evaluate Fawkes’s performance against these two potential countermeasures.

Image Transformation. A simple technique to mitigate the impact of small image perturbations is to transform images in the training dataset before using them for model training [30, 66]. These transformations include image augmentation, blurring, or adding noise. Additionally, images posted online are frequently compressed before sharing (*i.e.* in the upload process), which could impact cloak efficacy.











Original	$\rho = 0.006$	$\rho = 0.008$	$\rho = 0.01$	$\rho = 0.012$
				
				
Protection Success Rate	51%	87%	100%	100%

Figure 3.18: When the user’s feature extractor is much less robust than the tracker’s feature extractor, the user can improve their protection success rate by increasing their DSSIM budget. (User: VGG2-Dense, Tracker: Web-Incept)

However, we find that none of these transformations defeat our cloaks. The protection success rate remains 100% even when data augmentation is applied to cloaked images⁵. Applying Gaussian blurring degrades normal accuracy by up to 18% (as kernel size increases) while cloak protection success rate remains $> 98\%$ (see Figure 3.15). Adding Gaussian noise to images merely disrupts normal classification accuracy – the cloak protection success rate remains above 100% as the standard deviation of the noise distribution increases (see Figure 3.16). Even image compression cannot defeat our cloak. We use progressive JPEG [237], reportedly used by Facebook and Twitter, to compress the images in our dataset. The image quality, as standard by Independent JPEG Group [7], ranges from 5 to 95 (lower value = higher compression). As shown in Figure 3.17, image compression decreases the protection success rate, but more significantly degrades normal classification accuracy.

Robust Model. As shown in §3.5, cloaks constructed on robust feature extractors transfer well to trackers’ less robust feature extractors. Thus, a natural countermeasure a tracker could employ is training their model to be extremely robust.

Despite the theoretically proven trade-off between normal accuracy and robustness [231], future work may find a way to improve model robustness while minimizing the accompanying drop in accuracy. Thus, we evaluate cloaking success when the tracker’s model is much more robust than the user’s feature extractor. In our simplified test, the user has a robust VGG2-Dense

⁵. Image augmentation parameters: rotation range=20°, horizontal shift=15%, vertical shift=15%, zoom range=15%

feature extractor (adversarially trained for 3 epochs), while the tracker has an extremely robust Web-Incept feature extractor (adversarially trained for 20 epochs). When the tracker’s model is this robust, the user’s cloak only achieves a 64% protection success rate.

However, if the user is extremely privacy sensitive, she could increase the visibility of her cloak perturbation to achieve a higher protection success rate. Figure 3.18 highlights the trade off between protection success and the input DSSIM level. The cloak’s protection success rate increases to 100% once the DSSIM perturbation is > 0.01 .

3.8.2 Cloak Detection

We now propose techniques a tracker could employ to detect cloaked images in their dataset. We also discuss mitigations the user could apply to avoid detection.

Existing Poison Attack Detection. Since cloaking is a form of data poisoning, prior work on detecting poisoning attacks [84, 218, 161, 240, 34, 207] could be helpful. However, all prior works assume that poisoning only affects a small percentage of training images, making outlier detection useful. Fawkes poisons an entire model class, rendering outlier detection useless by removing the correct baseline.

Anomaly Detection w/o Original Images. We first consider anomaly detection techniques in the scenario where the tracker does not have any original user images. If trackers obtain both target and cloaked user images, they can detect unusual closeness between cloaked images and target images in model feature space. Empirically, the $L2$ feature space distance between the cloaked class centroid and the target class centroid is 3 standard deviations smaller than the mean separation of other classes. Thus, user’s cloaked images can be detected.

However, a user can trivially overcome this detection by maintaining separation between cloaked and target images during cloak optimization. To show this, we use the same experimental setup as in §3.5.2 but terminate the cloak optimization once a cloaked image is 20% of the original $L2$ distance from the target image. The cloak still achieves a 100% protection success rate, but the

cloak/target separation remains large enough to evade the previous detection method.

Anomaly Detection w/ Original Images. When the trackers have access to original training images (see §3.7), they could use clustering to see if there are two distinct feature clusters associated with the user’s images (i.e. cloaked and uncloaked). Normal classes should have only one feature cluster. To do this, the tracker could run a 2-means clustering on each class’s feature space, flagging classes with two distinct centroids as potentially cloaked. When we run this experiment, we find that the distance between the two centroids of a protected user class is 3 standard deviations larger than the average centroid separation in normal classes. In this way, the tracker can use original images to detect the presence of cloaked images.

To reduce the probability of detection by this method, the user can choose a target class that does not create such a large feature space separation. We empirically evaluate this mitigation strategy using the same experimental configuration as in §3.5.2 but choose a target label with average (rather than maximal) distance from their class. The cloak generated with this method still achieves a 100% protection success rate, but $L2$ distance between the two cluster centroids is within 1 standard deviation of average.

The user can evade this anomaly detection strategy using the maximum distance optimization strategy in §3.4. In practice, for any tracker model with a moderate number of labels (≥ 30), cloaks generated with average or maximum difference optimization consistently achieves high cloaking success. Our experimental results show these two methods perform identically in protection success against both our local models and the Face++ API.

3.9 Discussion and Conclusion

We have presented a first proposal to protect individuals from recognition by unauthorized and unaccountable facial recognition systems. Our approach applies small, carefully computed perturbations to cloak images, so that they are shifted substantially in a recognition model’s feature representation space, all while avoiding visible changes. Our techniques work under a wide range

of assumptions and provide 100% protection against widely used, state-of-the-art models deployed by Microsoft, Amazon and Face++.

Like most privacy enhancing tools and technologies, Fawkes can also be used by malicious bad actors. For example, criminals could use Fawkes to hide their identity from agencies that rely on third-party facial recognition systems like Clearview.ai. We believe Fawkes will have the biggest impact on those using public images to build unauthorized facial recognition models and less so on agencies with legal access to facial images such as federal agencies or law enforcement. We leave more detailed exploration of the tradeoff between user privacy and authorized use to future work.

Protecting content using cloaks faces the inherent challenge of being *future-proof*, since any technique we use to cloak images today might be overcome by a workaround in some future date, which would render previously protected images vulnerable. While we are under no illusion that this proposed system is itself future-proof, we believe it is an important and necessary first step in the development of user-centric privacy tools to resist unauthorized machine learning models. We hope that followup work in this space will lead to long-term protection mechanisms that prevent the mining of personal content for user tracking and classification.

CHAPTER 4

GLAZE: PROTECTING ARTISTS FROM STYLE MIMICRY BY TEXT-TO-IMAGE MODELS

4.1 Introduction

It is not an exaggeration to say that the arrival of text-to-image generator models has transformed, perhaps upended, the art industry. By sending simple text prompts like “A picture of a corgi on the moon” to diffusion models such as StableDiffusion or MidJourney, anyone can generate incredibly detailed, high resolution artwork that previously required many hours of work by professional artists. AI-art such as those in Figure 4.1 have won awards at established art conventions [180], served as cover images for magazines [128], and used to illustrate children’s books [168] and video games [190]. More powerful models continue to arrive [214, 98, 154], catalyzed by VC funding [254, 253, 163], technical research breakthroughs [106, 39, 140, 123, 20], and powered at their core by continuous training on a large volume of human-made art scraped from online art repositories such as ArtStation, Pinterest and DeviantArt.

Only months after their arrival, these models are rapidly growing in users and platforms. In September 2022, MidJourney reported over 2.7 million users and 275K AI art images generated *each day* [88]. Beyond simple prompts, many have taken the open sourced StableDiffusion model, and “fine-tuned” it on additional samples from specific artists, allowing them to generate AI art that *mimics* the specific artistic styles of that artist [89]. In fact, entire platforms have sprung up where home users are posting and sharing their own customized diffusion models that specialize on mimicking specific artists, likeness of celebrities, and NSFW themes [45].

Beyond open questions of copyrights [25], ethics [166, 71], and consent [60, 80, 70], it is clear that these AI models have had significant negative impacts on independent artists. For the estimated hundreds of thousands of independent artists across the globe, most work on commissions, and attract customers by advertising and promoting samples of their artwork online. First, professional



Figure 4.1: Sample AI-generated art pieces from the Midjourney community showcase [143, 180].

artists undergo years of training to develop their individual artistic styles. A model that mimics this style profits from that training without compensating the artist, effectively ending their ability to earn a living. Second, as synthetic art mimicry continues to grow for popular artists, they displace original art in search results, further disrupting the artist's ability to advertise and promote work to potential customers [186, 89]. Finally, these mimicry attacks are demoralizing art students training to be future artists. Art students see their future careers replaced by AI models even if they can successfully find and develop their own artistic styles [151].

Today, all of these consequences have indeed occurred in the span of a few months. Art students are quitting the field; AI models that mimic specific artists are uploaded and shared for free; and professional artists are losing their livelihoods to models mimicking their unique styles. Artists are fighting back via lawsuits [58, 102], online boycotts and petitions [62], but legal and regulatory action can take years, and are difficult to enforce internationally. Thus most artists are faced a choice to 1) do nothing, or 2) stop sharing samples of their art online to avoid training models, and in doing so cripple their main way to advertise and promote their work to customers.

In this paper, we present the design, implementation and evaluation of a technical alternative to protect artists against style mimicry by text-to-image diffusion models. We present Glaze, a system that allows an artist to apply carefully computed perturbations to their art, such that diffusion models will learn significantly altered versions of their style, and be ineffective in future attempts at style mimicry. We worked closely with members of the professional artist community to develop Glaze, and conduct multiple user studies with 1,156 participants from the artist community to evaluate its efficacy, usability, and robustness against a variety of active countermeasures.

Intuitively, *Glaze* works by taking a piece of artwork, and computing a minimal perturbation (a “style cloak”) which, when applied, shifts the artwork’s representation in the generator model’s feature space towards a chosen target art style. Training on multiple cloaked images teaches the generator model to shift the artistic style it associates with the artist, leading to mimicry art that fails to match the artist’s true style.

Our work makes several key contributions:

- We engage with top professional artists and the broader community, and conduct user studies to understand their views and concerns towards AI art and the impact on their careers and community.
- We propose *Glaze*, a system that protects artists from style mimicry by adding minimal perturbations to their artwork to mislead AI models to generate art different from the targeted artist. 92% of surveyed artists find the perturbations small enough not to disrupt the value of their art.
- Surveyed artists find that *Glaze* successfully disrupts style mimicry by AI models on protected artwork. 93% of artists rate the protection is successful under a variety of settings, including tests against real-world mimicry platforms.
- In challenging scenarios where an artist has already posted significant artworks online, we show *Glaze* protection remains high. 87.2% of surveyed artists rate the protection as successful when an artist is only able to cloak 1/4 of their online art (75% of art is uncloaked).
- We evaluate *Glaze* and show that it is robust (protection success \geq 85%) to a variety of adaptive

countermeasures.

- We discuss *Glaze* deployment and post-deployment experiences, including countermeasures in the wild.

Ethics. Our user study was reviewed and approved by our institutional review board (IRB). All art samples used in experiments were used with explicit consent by their respective artists. All user study participants were compensated for their time, although many refused payment.

4.2 Background and Related work

In this section, we provide a brief background on the text-to-image models used for AI art generation and then discuss existing work on protecting user data from machine learning models.

4.2.1 Defenses against invasive ML models

Prior work has proposed cloaking as a privacy-preserving tool that makes a user’s face images unlearnable by face recognition models [202, 43, 32, 64]. To do so, these works poison the training data of the model by optimizing perturbations (cloaks) that cause cloaked images to have drastically different feature representations from original user images. A face recognition model trained on cloaked images will associate an incorrect feature representation with an user’s identity and thus will fail to recognize the user at inference time. In the context of generative models, concurrent work [187] proposes PhotoGuard, a method to cloak images in order to prevent unauthorized image edits (inpainting) on cloaked images. We note that PhotoGuard is an evasion attack (the model weights are not impacted by the defense), which is different from previously mentioned cloaking tools which rely on poisoning the training dataset of the model.

Many of these privacy-preserving cloaking methods are later defeated by countermeasures. Radiya *et al.* [172] proposes a robust training technique that trains the facial recognition model on cloaked images but with correct labels. This process makes the models robust against cloak perturbation and is able to defeat both Fawkes and LowKey. We found PhotoGuard can be defeated by

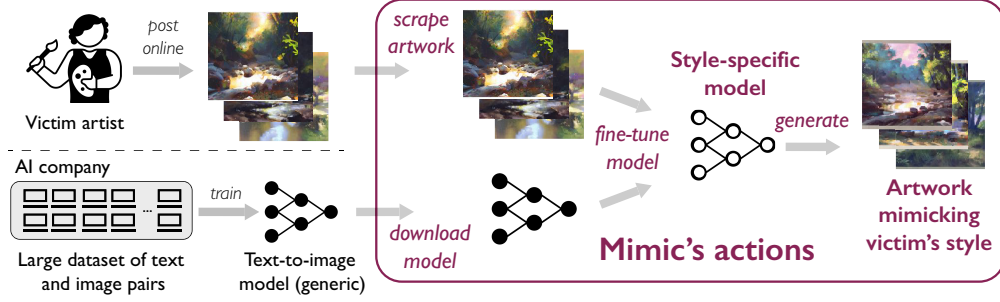


Figure 4.2: High level overview of the mimicry attack scenario. The mimic scrapes copyrighted artwork from the victim artist and uses these to fine-tune a pre-trained, generic text-to-image model. The mimic then uses the fine-tuned model to generate artwork in the style of the victim artist.

simply adding small Gaussian noise on the image. In §4.5.2, we discuss the limitations of adapting existing cloaking methods for anti-mimicry protection and empirically show their suboptimal performance.

4.3 Background: AI Art and Style Mimicry

In this section, we provide critical context in the form of basic background on current AI art models and style mimicry.

4.3.1 Text-to-Image Generation

Since Text-to-image generation was first proposed in 2015 [137], a stream of research has proposed newer model architectures and training methods enabling generation of higher-quality images [171, 273, 261, 122, 282]. The high level design of recent models used for AI art generation [178, 173, 53] is shown in Figure 4.4. During training, the model takes in an image x and uses a feature extractor Φ to extract its features, producing $\Phi(x)$. Simultaneously, a conditional image generator G takes in a corresponding text caption (s) and outputs a predicted feature vector $G(s)$. Then the parameters of G are optimized so the text feature vector $G(s)$ matches the image feature vector $\Phi(x)$. At generation time, a user gives G a text prompt s_0 , and G outputs an image feature vector

$G(s_0)$. A decoder D then decodes $G(s_0)$ to produce the final generated image.

Compared to earlier models based on generative adversarial networks (GANs) or variational autoencoders (VAE) [171, 282, 226], more recent models [178, 174] leveraging *diffusion* models produce significantly higher quality images. Feature extractor (Φ) is used to reduce the dimensionality of the input image to facilitate the generation process. The extractor Φ and decoder D are often a pair of variational autoencoder (VAE) [178, 173], i.e. extractor (encoder) extracts image features and decoder map features back to images.

Training Data Sources. The training datasets of these models typically contain image/ALT text pairs scraped from the Internet. They are extremely large, e.g. LAION [193] contains 5 billion images collected from 3 billion webpages.

These datasets are subject to minimal curation and governance. Data collectors typically only filter out data with extremely short or incorrect text captions (based on an automated text/image alignment metric [193]). Since copyrighted images are not filtered [193], these datasets are rife with private, sensitive content, including copyrighted artworks.

4.3.2 *Style Mimicry*

In a *style mimicry* attack, a bad actor uses an AI art model to create art in a particular artist’s style without their consent. More than 67% of art pieces showcased on a popular AI-art-sharing website leverage style mimicry [143].

Style mimicry techniques. Today, a “mimic” can easily copy the style of a victim artist with only an open-source text-to-image model and a few samples of artwork from the artist. A naive mimicry attack directly queries a generic text-to-image model using the name of the victim artist. For example, the prompt “a painting in the style of Greg Rutkowski” would cause the model to generate images in the style of Polish artist Greg Rutkowski. This is because many of Rutkowski’s artworks appear in training datasets of these generic models labeled with his name.

Naive mimicry can succeed when the artist is well-known and has a significant amount of art



Original artwork
by Hollie Mengert



Mimicked artwork
in Hollie's style

Figure 4.3: Real-world incident of AI plagiarizing the style of artist Hollie Mengert [19]. **Left:** original artwork by Hollie Mengert. **Right:** plagiarized artwork generated by a model trained to mimic Hollie's style.

online, but fail on other artists. In more recent mimicry attacks, a mimic *fine-tunes* a generic text-to-image model on samples of a target artist's work (as few as 20 unique pieces) downloaded from online sources. This calibrates the model to the victim artist's style, identifying important features related to the victim style and associating these regions in the feature space with the victim artist's name [181, 73]. This enables style mimicry with impressive accuracy. The entire fine-tuning process takes less than 20 minutes on a low-end consumer GPU¹.

Real-work mimicry incidents. The first well-known incident of mimicry was when a Reddit user stole American artist Hollie Mengert's style and open-sourced the style-specific model on Reddit [19]. Figure 4.3 has a side-by-side comparison of Hollie's original artwork and plagiarized artwork generated via style mimicry. Later, famous cartoonist Sarah Andersen reported that AI art models can mimic her cartoon drawings [16], and other similar incidents abound [146, 264].

1. It takes an average of 18.3 minutes on a GTX 1080 GPU

Model training

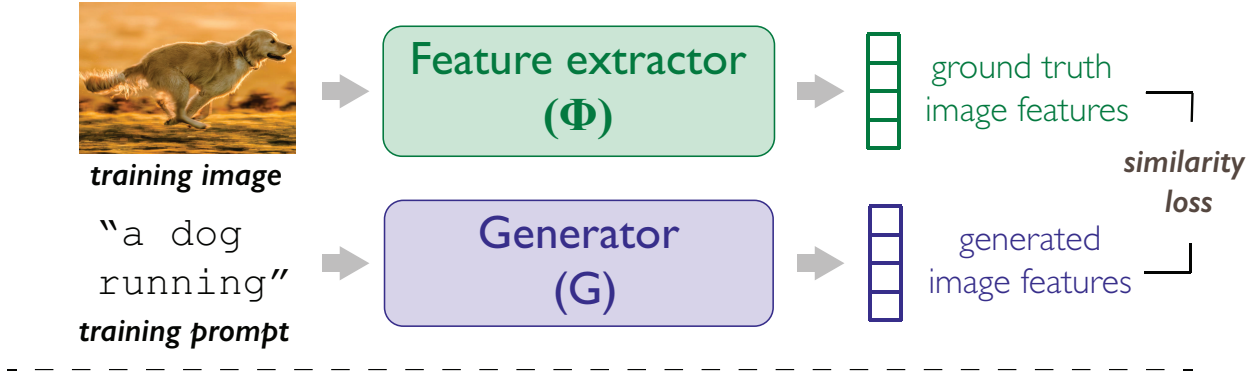


Image generation

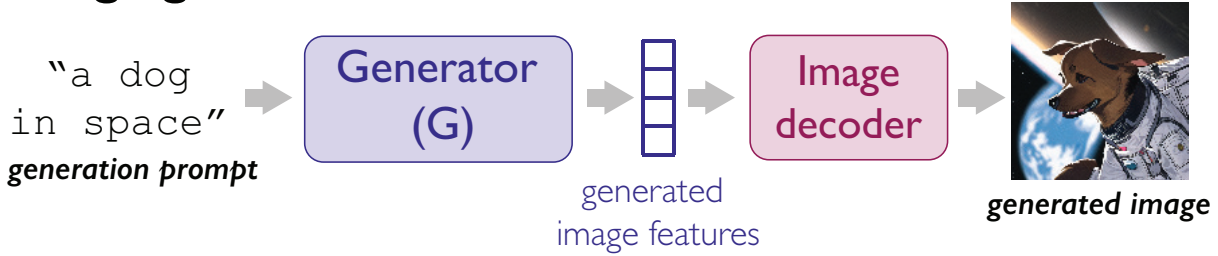


Figure 4.4: High level model architecture of text-to-image models.

Several companies [190] have even hosted style mimicry as a service, allowing users to upload a few art pieces painted by victim artists and producing new art in the victim styles. CivitAI [45] built a large online marketplace where people share their customized stable diffusion models, fine-tuned on certain artwork.

4.4 Collaborating with Artists

Next, we explain our collaborative relationship with professional artists, and its significant impact on our key evaluation metrics in this paper. We also summarize key results from our first user study on views of AI art and mimicry by members of the artist community.

Artists have spoken out against style mimicry in numerous venues, focusing particularly on how it violates their intellectual property rights and threatens their livelihoods [47, 247, 230, 246]. Others have taken direct action. The Concept Art Association raised over \$200K to fight AI art,

and filed a class action lawsuit in the US against AI art companies [102]. In November 2022, artists organized a large protest against ArtStation [246], the large digital art sharing platform that allowed users to post AI artwork without identification. Anti-AI images flooded the site for several weeks, until ArtStation banned the protest images [63]. Recently, the Writers Guild of America (WGA) went on strike demanding contractual changes to ban generative AI [42].

Members of the professional art community reached out to us in Sept 2022. We joined online town halls and meetings alongside hundreds of professionals, including Emmy winners and artists at major film studios. After learning more, we began an active collaboration with multiple professional artists, including award-winning artist Karla Ortiz, who leads efforts defending artists and is lead plaintiff in the class action suit. The artists helped this project in multiple ways, by 1) sharing experiences about specific ways AI-art has impacted them and their colleagues; 2) sharing domain knowledge about what is acceptable to artists in terms of perturbations on their art; and 3) helping to widely disseminate our user study to members of their professional organizations, including the Concept Art Association and the Animation Guild (TAG839).

Evaluation via Direct Feedback from Artists. Our goal is to help artists disrupt AI models trying to mimic their artistic style, without adversely impacting their own artwork. Because “success” in this context is highly subjective (“Did this AI-art successfully mimic Karla’s painting style?”), we believe the only reliable evaluation metric is direct feedback by professional artists themselves. Therefore, wherever possible, the evaluation of *Glaze* is done via detailed user studies engaging members of the professional artist community, augmented by an empirical score we develop based on genre prediction using CLIP models.

We deployed two user studies during the course of this project (see Table 4.1). Both are IRB-approved by our institution. Both draw participants from professional artists informed via their social circles and professional networks. The first (Survey 1, §4.4.1, §4.7.3), asked participants about their broad views of AI style mimicry, and then presented them with a number of inputs and outputs of our tool, and asked them to give ratings corresponding to key metrics we wanted

Survey	# of artists	Content
Survey 1	1156	1) Broad views of AI art and style mimicry (§4.4.1)
		2) Glaze’s usability, i.e. acceptable levels of cloaking (§4.7.3)
		3) Glaze performance in disrupting style mimicry (§4.7.3)
Survey 2 (Extension to Survey 1)	151	1) Additional performance tests (§4.7.3)
		2) Robustness to advanced scenarios (§4.7.4) and countermeasures (§4.8)
		3) Additional system evaluation

Table 4.1: Information on our user studies: the number of artist participants and where we report the results of the studies. We sent Survey 2 to some specific participants from survey 1 who volunteered to participate in a followup study.

to evaluate. We select a subset of participants from the first study to participate in a longer and more in-depth study (Survey 2) where they were asked to evaluate the performance of *Glaze* in additional settings (§4.7.3, §4.7.4, §4.8).

4.4.1 Artists’ Opinions on Style Mimicry

While we expected artists to view style mimicry negatively, we wanted to better understand how much individual artists understood this topic and how many perceived it as a threat. Here we describe results from Survey 1 to gather perceptions of the potential impact of AI art on existing artists.

Survey Design. Our survey consisted of both multiple choice and free response questions to understand how well people understand the concept of AI art, and how well the models successfully imitate the style of artists. Additionally, we asked artists about the extent to which they anticipate the emergence of AI art to impact their artistic activities, such as posting their art online and their job security. A handful of professional artists helped disseminate our survey to their respective artist community groups. Overall, we collected responses from 1,207 participants, consisting primarily of professional artists (both full-time (46%) and part-time/freelancer (50%)) and some non-artist members of the art community who felt invested in the impact of AI art (4%). Of the participants who consider themselves artists, their experience varied: \leq 1 year (13%), 1-5 years (49%), 5-10 years (19%), 10+ years (19%). Participants’ primary art style varied widely, including: animation, concept art, abstract, anime, game art, digital 2D/3D, illustration, character artwork, storyboarding, traditional painting/drawing, graphic design, and others.

Key Results. Our study found that 91% of the artists have read about AI art extensively, and either know of or worry about their art being used to train the models. Artists expect AI mimicry to have a significant impact on artist community: 97% artists state it will decrease some artists' job security; 88% artists state it will discourage new students from studying art; and 70% artists state it will diminish creativity. "Junior positions will become extinct," as stated by one participant.

Many artists (i.e. 89% artists) have already or plan to take actions because of AI mimicry. Over 95% of artists post their artwork online. Out of these artists, 53% of them anticipate reducing or removing their online artwork, if they haven't already. Out of these artists, 55% of them believe reducing their online presence will significantly impact their careers. One participant stated "AI art has unmotivated myself from uploading more art and made me think about all the years I spent learning art." 78% of artists anticipate AI mimicry would impact their job security, and this percentage increases to 94% for the job security of newer artists. Further, 24% of artists believe AI art has *already* impacted their job security, and an additional 53% expect to be affected within the next 3 years. Over 51% of artists expressed interest in proactive measures, such as personally joining class action lawsuits against AI companies.

Professional artists thought AI mimicry was very successful at mimicking the style of specific artists. We showed the artists examples of original artwork from 23 artists, and the artwork generated by a model attempting to mimic their styles (detailed mimicry setup in §4.7). 77% of artists found the AI model *successfully* or *very successfully* mimic the styles of victim artists, with one stating "it's shocking how well AI can mimic the original artwork." Additionally, 19% of participants thought the AI mimicry is somewhat successful, leaving only < 5% of artists rating the mimicry as unsuccessful. Several artists also pointed out that, as artists, upon close inspection they could spot differences between the AI art and originals, but were skeptical the general public would notice them.

A significant concern of most participants, surprisingly, is not just the existence of AI art, but rather scraping of existing artworks without permission or compensation. As one participant

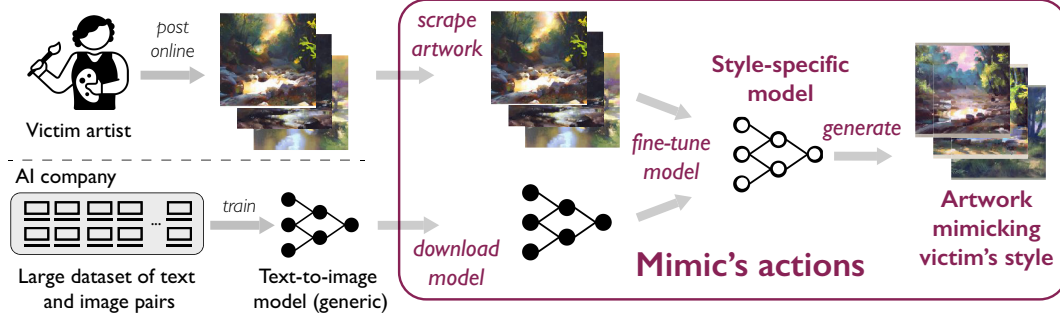


Figure 4.5: High level overview of the mimicry attack scenario. The mimic scrapes copyrighted artwork from the victim artist and uses these to fine-tune a pre-trained, generic text-to-image model. The generic model is trained and open-sourced by an AI company. The mimic then uses the fine-tuned model to generate artwork in the style of the victim artist.

stated: “If artists are paid to have their pieces be used and asked permission, and if people had to pay to use that AI software with those pieces in it, I would have no problem.” However, without consent to use their artwork to train the models, “it’s incredibly disrespectful to the artist to have their work ‘eaten’ by a machine [after] many years to grow our skills and develop our styles.”

4.5 Preliminaries

We propose *Glaze*, a tool that protects artists against AI style mimicry. An artist uses *Glaze* to add small digital perturbations (“cloak”) to images of their own art before sharing online (Figure 4.6). A text-to-image model that trains on cloaked images of artwork will learn an incorrect representation of the artist’s style in feature space i.e. the model’s internal understanding of artistic styles. When asked to generate art pieces in victim’s style, the model will fail to mimic the style of the victim, and instead output art pieces in a recognizably different style.

Here, we first introduce the threat model, then discuss existing alternatives to the AI style mimicry problem. We present the intuition behind *Glaze* and detailed design in §4.6.

4.5.1 Threat Model

Here we state assumptions for both the artists protecting their own art and the users training models to replicate their artistic style. We refer to these AI art model trainers as “mimics.”

Artists. Artists want to share and promote their artwork online without allowing mimics to train models that replicate their art styles. Sharing art online enables artists to sell their work and attract commissioned work, fueling their livelihoods (§4.4). Artists protect themselves by adding imperceptible perturbations to their artwork before sharing them as shown in Figure 4.6. The goal of the *Glaze* cloak is to disrupt the style mimicry process, while only introducing minimal perturbation on images of the artwork.

We assume the artists have access to moderate computing resources (e.g., a laptop) and add perturbation to images of their artwork locally before posting online. We also assume artists have access to some public feature extractor (e.g., open-source models such as Stable Diffusion). We begin with assumption that artists use the same feature extractor as mimics (large majority of mimics use the open-source Stable Diffusion model). We later relax this assumption.

Mimics. The mimic’s goal is to train a text-to-image model that generates *high-quality* art pieces of any subject in the *victim’s style*. A mimic could be a well-funded AI company, e.g., Stability AI or OpenAI, or an individual interested in the style of victim artist. We assume the mimic has:

- access to the weights of generic text-to-image models well-trained on large datasets;
- access to art pieces from the target artist;
- significant computational power.

We assume the attack scenario where the mimic fine-tunes its model on images of the artist’s artwork (as shown in Figure 4.5). This is stronger than the naive mimic attack without fine tuning. Finally, we assume the mimic is aware of our protection tool and can deploy adaptive countermeasures (§4.8).

4.5.2 Potential Alternatives and Challenges

A number of related prior works target protection against invasive and unauthorized facial recognition models. They proposed “image cloaking” as a tool to prevent a user’s images from being used to train a facial recognition model of them [202, 43, 32, 64, 251]. They share a similar high level approach, by using optimized perturbations that cause cloaked images to have drastically different feature representations from original user images. It is possible to adapt existing cloaking-based systems to protect artists against AI style mimicry. Protection system would compute a cloak on each artwork in order to perturb its feature space representation to be different from its unperturbed representation. This can succeed if the cloak significantly shifts the artwork’s feature representation, making resulting models generate dramatically different artwork.

We found that in practice, however, existing solutions are unable to introduce large-enough feature space shifts to achieve the desired protection. This is due to the properties of feature spaces in text-to-image models. Face recognition models *classify identities*, so their feature spaces mainly represent identity-related information. On the other hand, text-to-image models *reconstruct original images from extracted features*, so their feature spaces retain more information about the original image (objects, locations, color, style, etc.). Thus, producing the same shift in feature representation in a text-to-image model is much harder (requires more perturbation budget) than in a classification model. This observation is validated by prior work showing that adversarial perturbations are much less effective at attacking generative models [110, 77, 224]. Specifically, [110, 224] found that adversarial attack methods that are effective at attacking classifiers are significantly less effective at attacking autoencoders. We empirically confirm that existing cloaking methods cannot prevent AI mimicry. We show that Fawkes [202] and LowKey [43] perform poorly in this setting, even when artists add highly visible cloaks to their artwork.

For generative models, concurrent work [187] proposes PhotoGuard, a method to cloak images to prevent unauthorized image edits (inpainting) on cloaked images. Similar to existing cloaking systems, PhotoGuard tries to indiscriminately minimize all information contained in an image (i.e.

the norm of the feature vector) to prevent models from editing the image. Thus, it is also not effective at mimicry prevention.

Design Challenges. The main reason that existing cloaking methods fail to prevent AI mimicry is because they indiscriminately shift all features in an image, wasting the cloak perturbation budget on shifting unnecessary features (e.g., object shape, location, etc.). Protecting artist’s style requires only *shifting features related to the artistic style of victim*. This can be achieved if a text-to-image model learns to draw objects similar to those drawn by the victim artist *as long as the model cannot mimic the artist’s unique style*. Thus, optimal protection from mimicry requires concentrating the cloak on style-specific features.

Unfortunately, identifying and separating out these style-specific features is difficult. Even assuming the existence of interpretability methods that perfectly explain the feature space of a text-to-image model, there is no clear way to mathematically define and calculate “artistic styles.” In all likelihood, any definition would change across different styles. For example, “impressionist” likely correlates more strongly with color features, whereas “cubism” correlates with shape features. Even across multiple art pieces in the same style, the style may manifest differently.

4.6 Disrupting Style Mimicry with Glaze

In this section, we introduce *Glaze*, its design intuition followed by the detailed algorithm.

4.6.1 Design Intuition

Our key intuition is to identify and isolate *style-specific features* of an artist’s original artwork, i.e. the set of image features that correspond to artistic style. Then *Glaze* computes cloaks while focusing the perturbation budget on these style-specific features to maximize impact on stylistic features.

As discussed, identifying and calculating style-specific features in model’s feature space is difficult due to the poor interpretability of model features and how art style manifests differently

across artworks. We overcome these two challenges by designing a style-dependent and artwork-dependent method that operates at image space. Given an artwork, we leverage “style transfer,” an end-to-end computer vision technique, to modify and isolate its style components. “Style transfer” transforms an image into a new image with a different style (e.g., from impressionist style to cubist style) while keeping other aspects of the image similar (e.g., subject matter and location).

We leverage style transfer in our protection technique as follows. Given an original artwork from the victim artist, we apply style-transfer to produce a similar piece of art with a different style, e.g., in style of “an oil painting by Van Gogh” (Figure 4.7 a). The new version has similar content to the original, but its style mirrors that of Van Gogh. We show more style-transfer examples with different target styles in Figure 4.8. Now, we can use the style-transferred artwork as projection target to guide the perturbation computation. This perturbs the original artwork’s style-specific features towards that of the style-transferred version. We do this by optimizing a cloak that, when added to the original artwork, makes its feature representation similar to the style-transferred image. Since the content is identical between the pair of images, cloak optimization will focus its perturbation budget on style features.

4.6.2 Computing Style Cloaks

Using this approach, we compute style cloaks to disrupt style mimicry as follows. Given an artwork (x), we use an existing feature extractor to compute the style-transferred version of x into target style T : $\Omega(x, T)$. We then compute a style cloak δ_x , such that δ_x moves x ’s style-specific feature representation to match that of $\Omega(x, T)$ while minimizing visual impact. The cloak generation optimization is:

$$\begin{aligned} \min_{\delta_x} \text{Dist}(\Phi(x + \delta_x), \Phi(\Omega(x, T))), \\ \text{subject to } |\delta_x| < p, \end{aligned} \tag{4.1}$$

where Φ is a generic image feature extractor commonly used in text-to-image generation tasks, $Dist(\cdot)$ computes the distance of two feature representations, $|\delta_x|$ measures the perceptual perturbation caused by cloaking, and p is the perceptual perturbation budget.

As discussed in §4.6.1, the use of the style-transferred image $\Omega(x, T)$ guides the cloak optimization in Eq (4.1) to focus on changing style-specific image features. To maximize cloak efficacy, the target style T should be dissimilar from artist’s original style in the feature space. We discuss our heuristic for selecting target styles in §4.6.

4.6.3 Detailed System Design

Now we present the detailed design of *Glaze*. Given a victim artist V , *Glaze* takes as input the set of V ’s artwork to be shared online X_V , an image feature extractor Φ , a style-transfer model Ω , and perturbation budget p . Note that in many cases, a single model (e.g. Stable Diffusion) provides both Φ and Ω .

Step 1: Choose Target Style. The selected target style T should be sufficiently different from V ’s style in model feature space to maximize chances of disrupting style mimicry. For example, Fauvism and Impressionism are distinct art styles that often look visually similar to the untrained eye. Image of an impressionist painting style cloaked to Fauvism might not produce a visually discernible effect on model-generated paintings. Note that an artist can maximize their ability to avoid mimicry if they consistently style cloak all their artwork towards the same target T .

For a new user, *Glaze* uses the following algorithm to randomly select T from a set of candidate styles reasonably different from V ’s style. The algorithm first inspects a public dataset of artists, each with a specific style (e.g., Monet, Van Gogh, Picasso). For each candidate target artist/style, it selects a few images in that style and calculates their feature space centroid using Φ . It also computes V ’s centroid in Φ using V ’s artwork. Then, it locates the set of candidate styles whose centroid distance to V ’s centroid is between the 50 to 75 percentile of all candidates. Finally, it randomly selects T from the candidate set.

Step 2: Style transfer. *Glaze* then leverages a pre-trained style-transfer model Ω [178] to generate the style-transferred artwork for optimization. Given each art piece $x \in X_V$ and target style T , it style transfers x to target style T to produce style-transferred image $\Omega(x, T)$.

Step 3: Compute cloak perturbation. Then, *Glaze* computes the cloak perturbation, δ_x for x , following the optimization defined by eq. (3.1), subject to $|\delta_x| < p$. Our implementation uses LPIPS (Learned Perceptual Image Patch Similarity) [279] to bound the perturbation. Different from the L_p distance used in previous work [31, 111, 182], LPIPS has gained popularity as a measure of user-perceived image distortion [43, 113, 179]. Bounding cloak generation with this metric ensures that cloaked versions of images are visually similar to the originals. We apply the *penalty method* [153] to solve the optimization in eq.(3.1) as follows:

$$\min_{\delta_x} \|\Phi(\Omega(x, T)), \Phi(x + \delta_x)\|_2^2 + \alpha \cdot \max(LPIPS(\delta_x) - p, 0) \quad (4.2)$$

where α controls the impact of the input perturbation. L_2 distance is used to calculate feature space distance.

Upload artwork online. Finally, the artist posts the cloaked artwork online. For artists already with a large online presence, they can cloak and re-upload artwork on their online portfolio. While updating online images is not always possible, *Glaze* can be effective even when the mimic’s model has significant amount of uncloaked art (§4.7.4).

4.6.4 On the Efficacy of Style Cloaks

Glaze’s style cloaks work by shifting feature representation of artwork in the generator model. But how much shift do we need in order to have a noticeable impact on mimicked art?

Two reasons suggest that even small shifts in style will have a meaningful impact in disrupting style mimicry. First, generative models used for style mimicry have *continuous* output spaces, i.e. any shift in image feature representation results in changes in the generated image. Because generative models are trained to interpolate their continuous feature spaces [252, 233], any shift

in the model’s representation of art style results in a new style, a “blend” between the artist and the chosen target style. Second, mimicked artwork must achieve reasonable quality and similarity in style to the artist to be useful. Small shifts in the style space often produce incoherent blends of conflicting styles that are enough to disrupt style mimicry, e.g., thick oil brushstrokes of Van Gogh’s style mixed into a realism portrait.

These two factors contribute to *Glaze*’s success in more challenging scenarios (§4.7.4), and its robustness against countermeasures (e.g. adversarial training) that succeed against cloaking tools for facial recognition (§4.8).

4.7 Evaluation

In this section, we evaluate *Glaze*’s efficacy in protecting artists from style mimicry. We first describe the datasets, models, and experimental configurations used in our tests. Then we present the results of *Glaze*’s protection in a variety of settings. Due to *Glaze*’s highly visual nature, we evaluate its performance using both direct visual assessment by **human artists** in a user study, and **automated metrics** (see §4.7.2 for details).

Summary of results. Over 93% of artists surveyed believe *Glaze* effectively protects artists’ styles from AI style mimicry attacks. Protection efficacy remains high in challenging settings, like when the mimic has access to unprotected artwork. *Glaze* also achieves high protection performance against a real-world mimicry-as-a-service platform. Of our 1156 artist participants, over 92% found the perturbations introduced by cloaking small enough not to disrupt the value of their art, and over 88% would like to use *Glaze* to protect their own artwork from mimicry attacks.

4.7.1 Experiment Setup

Mimicry dataset. We evaluate *Glaze*’s performance in protecting the styles of the following two groups of artists:

- *Current artists*: 4 professional artists let us use their artwork in our experiments. These artists have different styles and backgrounds (e.g., full-time/freelancers, watercolor painters/digital artists, well-known/independent). Each provided us with between 26 to 34 *private* original art pieces for our experiments. We use perceptual hashing [107] to verify that none of these are included in existing public datasets used to train generic text-to-image models (e.g. [193, 33]).
- *Historical artists*: We also evaluate *Glaze*’s protection on 195 historical artists (e.g., van Gogh, Monet) from the WikiArt dataset [185]. The WikiArt dataset contains 42,129 art pieces from 195 artists. Each art piece is labeled with its genre (e.g., impressionism, cubism). We randomly sampled 30 art pieces from each artist to use in style mimicry attacks. Generic text-to-image models found online have been trained on some artwork from these artists. Using this art simulates a more challenging scenario in which a famous artist attempts to disrupt a model that already understands their style.

Mimicry attack setup. We recreate the strongest-possible mimicry attack scenario, based on techniques used in real-world mimicry incidents [181, 264, 19], that works as follows. First, we take art pieces from the victim artist V and generate a text caption for each piece using an image captioning model [135]. The pretrained image captioning model generates a short sentence to describe the image. We found that this model can correctly caption protected images, likely because *Glaze* focuses on perturbing style features while the captioning models focus on image content. Then, we append the artist’s name to each caption, e.g., “mountain range *by Vincent van Gogh*”. Finally, we fine-tune a pre-trained generic text-to-image model (details below) on the caption/image pairs.

We use 80% of the art pieces from the victim artists to fine-tune models that mimic each artist’s style, reserving the rest for testing. We fine-tune for 3000 optimization steps, which we find achieves the best mimicry performance. We then use the fine-tuned, style-specific model to generate mimicked artwork in style of each victim artist. We query the model using the generated captions (which include V ’s name) from the held-out test artwork set. We generate 5 pieces of

mimicked art for each text caption using different random seeds and compare these to the real victim art pieces with this caption.

Text-to-image models. We use two state-of-the-art, public, generic text-to-image models in our experiments:

- *Stable Diffusion (SD)*: Stable Diffusion is a popular and high-performing open-source text-to-image model [214], trained on 11.5 million images from the LAION dataset [193]. SD training takes over 277 GPU months (on A100 GPU) and costs around \$600K [214]. SD uses diffusion methods to generate images and achieves state-of-the-art performance on several benchmarks [178]. Viewed as one of the best open-source models, SD has powered many recent developments in text-to-image generation [14, 154, 117, 190]. We use SD version 2.1 in the paper [214], the most up-to-date version as of December 2022.
- *DALL·E-mega (DALL·E-m)*: DALL·E-m-mega, an updated version of the more well-known DALL·E-m-mini, is an open-source model based on OpenAI’s DALL·E-m 1 [173]. The model leverages a VAE for image generation and is trained on 17 million images from three different datasets [204, 33, 227]. Training takes 2 months on 256 TPUs [52]. While DALL·E-m performs worse than diffusion-based models like SD, we use it to evaluate how *Glaze* generalizes to different model architectures.

Glaze configuration. We generate cloaks for each of victim V ’s art pieces following the methodology of §4.6.3. First, we use the target selection algorithm to select a target style T . We choose from a set of 1119 candidate target styles, collected by querying the WikiArt dataset with artist and genre names, e.g., “Impressionism painting by Monet”². We then style transfer each victim art piece into the target style leveraging the style transfer functionality of stable diffusion model (stable diffusion model has both text-to-image and style transfer functionality). A style transfer model takes in an original image and a target prompt as input. Leveraging a similar diffusion process, the model modifies the original image to a style similar to that described in the

2. One artist may paint in multiple styles, resulting in multiple candidate target styles from a single artist.

target prompt. More information on style transfer can be found in [184]. Finally, we optimize a cloak for each art piece using Eq. 5.2 by running the Adam optimizer for 500 steps. We benchmark *Glaze*’s runtime on artwork with resolution ranging from 512 to 6000 pixels, using SD’s feature extractor (ViT model with 83 million parameters). It takes an average of 1.2 mins on Titan RTX GPU and 7.3 mins on a single Intel i7 CPU to generate a cloak for a single piece of art.

In our initial experiments, we assume *Glaze* generates cloaks using the same image feature extractor as the mimic (e.g. SD’s or DALL·E-m’s feature extractor). We relax this assumption and evaluate *Glaze*’s performance when artists and mimics use different feature extractors in §4.7.4.

4.7.2 Evaluation Metrics

We evaluate our protection performance using both visual assessment and feedback from human artists, and a scalable metric. Here, we describe the setup of our evaluation study and define the exact metrics used for evaluation.

Artist-rated protection success rate (Artist-rated PSR): The user studies ask artists to rate the performance of *Glaze*. We generate a dataset of mimicry attacks on 13 victim artists (the 4 current artists and 9 randomly chosen historical artists) across 23 protection scenarios (including ones in §4.8). For each participant, we randomly select a set of mimicry attacks out of these 13×23 settings and ask them to evaluate protection success. For each mimicry attempt, we show participants 4 mimicked art pieces and 4 original art pieces from the victim artist. Using original art pieces as an indicator of the human artist’s style, we ask participants to consider the mimicked art, and rate the success of *Glaze*’s protection on a 5-level Likert scale (ranging from “not successful at all” to “very successful”). Each mimicry attempt is evaluated by at least 10 participants. We define *artist-rated PSR* as the percent of participants who rated *Glaze*’s protection as “successful” or “very successful.” Our user studies primarily focus on artists, as they would be most affected by this technology. We found though, that not all current artists despise AI art, and some view it as a new avenue for a different form of artistry.

Generic model	Artist dataset	w/o <i>Glaze</i>		w/ <i>Glaze</i> (p=0.05)	
		Artist-rated PSR	CLIP-based genre shift	Artist-rated PSR	CLIP-based genre shift
SD	Current	4.6 \pm 0.3%	2.4 \pm 0.2%	94.3 \pm 0.8%	96.4 \pm 0.5%
	Historical	4.2 \pm 0.2%	1.3 \pm 0.2%	93.3 \pm 0.6%	96.0 \pm 0.3%
DALL-E-m	Current	31.9 \pm 3.5%	6.4 \pm 0.8%	97.4 \pm 0.2%	97.4 \pm 0.3%
	Historical	29.8 \pm 2.4%	5.8 \pm 0.6%	96.8 \pm 0.3%	97.1 \pm 0.2%

Table 4.2: *Glaze* has a high protection success rate, as measured by artists and CLIP, against style mimicry attacks. We compare protection success when artists do not use *Glaze* vs. when they do (with perturbation budget 0.05).

CLIP-based genre shift: We define a new metric based on CLIP [170], using the intuition that *Glaze* succeeds if the mimicked art has been impacted enough by *Glaze* to be classified into a *different art genre* from the artist’s original artwork. We leverage CLIP model’s ability to classify art images into art genres. Given a set of mimicked art targeting an artist V , we define *CLIP-based genre shift rate* as the percentage of mimicked art whose top 3 predicted genres do not contain V ’s original genre. A higher genre shift rate means more mimicked art belongs to a different genre from the victim artist, and thus means more successful protection.

To calculate the genre shift we use a set of 27 historical genres from WikiArt dataset and 13 digital art genres [95] as the candidate output labels. We show that a pre-trained CLIP model is able to achieve high genre classification performance. We report the average CLIP-based genre shift for all 199 victim artists across all mimicked artworks.

We use CLIP-based genre shift as a supplemental metric to evaluate *Glaze* because it is only able to detect style changes at the granularity of art genres. However, mimicry attacks also fail when *Glaze* causes the mimicked artwork quality to be very low, something that CLIP cannot measure. Measuring the quality of generated image has been a challenging and ongoing research problem in computer vision [112, 22, 105].

4.7.3 *Glaze’s Protection Performance*

Style mimicry success when *Glaze* is not used. Mimicry attacks are very successful when the mimic has access to a victim’s original (unmodified) artwork. Examples of mimicked artwork can be found in Figure 4.9. The leftmost two columns of Figure 4.9 show a victim artist’s original

artwork, while the third column depicts mimicked artwork generated by a style-specific model trained on victim’s original artwork when *Glaze* is not used. In our user study, over $> 95\%$ of respondents rated the attack as successful. Table 4.2, row 1, gives the artist-rated and CLIP-based genre shift for mimicry attacks on unprotected art.

SD models produce stronger mimicry attacks than DALL·E-m models, according to our user study (see Table 4.2). This is unsurprising, as DALL·E-m models generally produce lower-quality generated images. CLIP-based genre shift does not reflect this phenomenon, as this metric does not assess image quality.

***Glaze*’s success at preventing style mimicry.** *Glaze* makes mimicry attacks markedly less successful, as shown in Figure 4.9. Columns 5 and 6 (from left) show mimicked artwork when the style-specific models are trained on artwork protected by *Glaze*. For reference, column 4 shows an example style-transferred artwork $\Omega(x, T)$ used to compute *Glaze* cloaks for the protected art pieces. Overall, *Glaze* achieves $> 93.3\%$ artist-rated PSR and $> 96.0\%$ CLIP-based genre shift (see Table 4.2). *Glaze*’s protection performance is slightly higher for current artists than for historical artists. This is likely because the historical artists’ images are present in the training datasets of our generic models (SD, DALL·E-m), highlighting the additional challenge of protecting well-known artists whose style was already learned by the generic models.

How large of perturbations will artists tolerate? Increasing the *Glaze* perturbation budget enhances protection performance. We observe that both artist-rated and CLIP-based genre shift increase with perturbation budget (see Figure 4.10 and Table 4.3). Given this tradeoff between protection success and *Glaze* protection visibility on original artwork, we evaluate how perturbation size impacts artists’ willingness to use *Glaze*.

We find that artists are willing to add fairly large *Glaze* perturbations to their artwork in exchange for protection against mimicry. To measure this, we show 3 randomly chosen pairs of original/cloaked artwork to each of the 1,156 artists in our first study. For each art pair, we ask the artist whether they would be willing to post the cloaked artwork (instead of the original, unmodi-

fied version) on their personal website. More than 92% of artists select “willing” or “very willing” when $p = 0.05$. This number only slightly increases to 94.3% when $p = 0.03$. Figure 4.11 details artists’ preferences as perturbation budget increases. (see Figure 4.12 for examples of cloaked artwork with increasing p). Based on these results, we use perturbation budget $p = 0.05$ for all our experiments, since most artists are willing to tolerate this perturbation size.

Surprisingly, over 32.8% artists are willing to use cloaks with $p = 0.2$, which is clearly visible to human eye (see Figure 4.12). While we are surprised by this high perturbation tolerance, in our follow-up free response artists noted that they would be willing to tolerate large perturbations because of the devastating consequence if their styles are stolen. One participant stated that “I am willing to sacrifice a bit image quality for protection.” Many artists ($> 80\%$) also noted that they have already used traditional, more visually disruptive techniques to protect their artwork online when posting online, i.e. adding watermark or reducing image resolution. One participant stated that “I already use low to medium resolution images only for online posting, thus this would not impact my quality control too much.”

4.7.4 *Glaze’s Protection Robustness*

Next, we test *Glaze*’s efficacy in more challenging scenarios. First, we measure performance when the mimic uses a different feature extractor for mimicry than the one used by the artist to generate the cloak. Second, we measure what happens when the mimic has uncloaked artwork samples from the victim. Due to the poor mimicry performance of DALL·E-m, we focus our evaluation using SD as the generic model.

Artist/mimic use different feature extractors. In the real world, it is possible that the mimic will use a different model (and thus a different image feature extractor) for style mimicry than the one used by the victim artist to cloak their artwork. While the feature extractors may still be similar because of the well-known transferability property between large models [54, 157, 219, 270, 200], their differences could reduce the efficacy of cloaking. We test this scenario using three feature

extractors— Φ -A, Φ -B, and Φ -C. Φ -A and Φ -B have different model architectures (autoencoder-KL [178] vs. VQ-VAE [173]) but are both trained on the ImageNet dataset [55]. Φ -A and Φ -C have different model architectures (autoencoder-KL vs VQ-VAE) and training datasets (ImageNet vs. CelebA [133]).

In our experiments, the victim artist uses one feature extractor (either Φ -B or Φ -C) to optimize cloaked artwork, and the mimic trains their style-specific models with SD models using Φ -A. Despite the difference in victim/mimic extractors, *Glaze*’s protection remains highly successful (left half of Figure 4.13)—the style of mimicked artwork remains distinct from artist’s true style. Artist-rated and CLIP-based genre shift measurements confirm this observation. Artist-rated PSR is $> 90.2\%$, while CLIP-based genre shift is $> 94.0\%$. The PSR is slightly higher when the two feature extractors only differ in architectures (Φ -B to Φ -A) than when they differ in both architecture and training data (Φ -C to Φ -A).

Mimic has access to uncloaked artwork. Another challenging scenario is when the mimic gains access to some *uncloaked* artwork from victim artists. This is a realistic scenario for many prominent artists with a large online presence. As expected, *Glaze*’s protection performance decreases when the mimic has access to more uncloaked artwork (right side of Figure 4.13). As the ratio of uncloaked/cloaked art in the mimic’s dataset increases, the mimicked artwork becomes more similar to artist’s original style. Yet, *Glaze* is still reasonably effective (87.2% artist-rated PSR) even when artists can only cloak 25% of their artwork. This validates our hypothesis in §4.6.4 that cloaking will have a noticeable effect as long as the mimic has some cloaked training data.

A mimic with access to a large amount of uncloaked artwork is still an issue for *Glaze*. Fortunately, in our user study, we found that 1) many artists constantly create and share new artwork online, which can be cloaked to offset the percentage of uncloaked artwork, and 2) many artists change their artistic style over time. In our user study, we asked artists to estimate the number of unique art pieces they currently have online (M) and the estimated number of art pieces they

Artist dataset	w/o <i>Glaze</i>		w/ <i>Glaze</i> (p=0.05)	
	Artist-rated PSR	CLIP-based genre shift	Artist-rated PSR	CLIP-based genre shift
Current	6.2 \pm 0.5%	3.8 \pm 0.3%	92.5 \pm 0.5%	94.2 \pm 0.3%
Historical	7.2 \pm 0.6%	3.3 \pm 0.4%	92.1 \pm 0.3%	93.9 \pm 0.4%

Table 4.4: Performance of *Glaze* against real-world mimicry service (scenario.gg). Mimicry service achieves high mimicry success when no protection is used. When *Glaze* is used, the mimicry service has low performance.

anticipate uploading each subsequent year (Y). Among artists with an existing online presence, over 40% have $Y/M > 25\%$, meaning that one year from now, $> 20\%$ of their total online artwork would be cloaked (if they start using *Glaze* immediately). More than 81% of artists also stated that their art style has changed over their career, and half of these said that theft of their old, outdated styles is less concerning.

4.7.5 Real-World Performance

Next, we test *Glaze* against a real-world style mimicry-as-a-service system, `scenario.gg` [190]. Scenario.gg is a web service that allows users to upload a set of images in a specific style. The service then trains a model to mimic the style and returns an API endpoint that allows the user to generate mimicked images in the trained style. The type of model or mimicry method used by the service is unknown.

Glaze remains effective against `scenario.gg`. We ask `scenario.gg` to mimic the style from a set of cloaked or uncloaked artwork from 4 current artists and 19 historical artists. Table 4.4 shows that when no protection is used, `scenario.gg` can successfully mimic the victim style (7.2% protection success). The mimicry success of `scenario.gg` is lower than our mimicry technique, likely because `scenario.gg` trains the model for fewer iterations due to computational constraints. When we use *Glaze* to cloak the artwork and upload the cloaked artwork, `scenario.gg` fails to mimic the victim style ($> 92.1\%$ artist-rated PSR and $> 93.9\%$ CLIP-based genre shift rate) as shown in Table 4.4.

4.8 Countermeasures

We consider potential countermeasures a mimic could employ to reduce the effectiveness of *Glaze*. We consider the strongest adversarial setting, in which the mimic has white-box access to our protection system, i.e. access to the feature extractor used and protection algorithm. In our experiments, we assume the mimic uses the SD model as the generic model and test the efficacy of each countermeasure on the 13 victim artists from §4.7.2. Here, we focus on artist-rated PSR metric, because many countermeasures trade off image quality for mimicry efficacy, and CLIP-based metric does not consider image quality.

Image transformation. A popular approach to mitigate the impact of small image perturbations, like those introduced by *Glaze*, is to transform training images before using them for model training [28, 66]. In our setting, the mimic could augment the cloaked artwork before fine-tuning their model on them to potentially reduce cloak efficacy. We first test *Glaze*’s resistance to two popular image transformations, adding Gaussian noise and image compression. We also consider a stronger version of this countermeasure that then tries to correct the image quality degradation introduced by the transformations.

Transforming cloaked artwork does not defeat *Glaze*’s protection. Figure 4.14 shows that as the magnitude of Gaussian noise (σ) increases, the quality of mimicked artwork decreases as fast as or faster than cloak effectiveness. This is because models trained on noisy images learn to generate noisy images. We observe a similar outcome when mimic uses JPEG compression (Figure 4.15), where image resolution and quality degrade due to heavy compression. Artists-rated PSR decreases slightly but remains above $> 87.4\%$ across both types of data transformations. Artists consider *Glaze*’s protection to be successful when mimicked artwork is of poor quality.

The mimic can take this countermeasure one step further by *reversing* the quality degradation introduced by the noising/compression process. Specifically, a mimic can run image denoising or image upscaling tools on the mimicked artwork (e.g., ones shown in Figure 4.14 and 4.15) to

increase their quality. We found this approach improves generated image quality but still does not allow for successful mimicry. For denoising, we ran a state-of-the-art CNN-based image denoiser [276] that is specifically trained to remove “additive Gaussian noise” (the same type of noise added to cloaked artwork). The last column of Figure 4.14 shows the denoised image (using the noisy mimicked image when $\sigma = 0.2$ as the input). While the process removes significant amounts of noise, the denoised artwork still has many artifacts, especially around complex areas of the artwork (e.g., human face). We observe similar results for image upscaling, where we use a diffusion-based image upscaler [214] to improve the quality of compressed images (Figure 4.15). Overall, our artist-rated protection success rate remains $> 85.3\%$ against this improved counter-measure.

Radiya *et al.* [172] robust training. Radiya *et al.* [172] design a robust training method to defeat cloaking tools like Fawkes [202] and Lowkey [43] in the face recognition setting. At a high level, this method augments the attacker’s training dataset with some cloaked images generated by the cloaking tool and the *correct* output labels. Training on such data makes the model more robust against cloak perturbations on unseen cloaked images at inference time, and thus, can potentially circumvent the protection.

We test if this robust training approach can defeat *Glaze*. We assume the mimic first robustly trains the feature extractors in their generic models using cloaked artwork generated by *Glaze*, and then trains the generator model to generate images from the robust feature space. Finally, the mimic uses the robust generic model for style mimicry as in §4.7.

Glaze performance remains high, even if the mimic robustly trains the generic model for many iterations before using it for style mimicry (see Figure 4.16). As the model becomes more robust, the mimicked artwork is less impacted by cloaking (less influenced of the target style). However, robust training greatly degrades mimicked image quality, preventing successful mimicry. Overall, the artist-rated PSR remains $> 88.7\%$. To mitigate robust training’s impact on image quality, we explore an alternative robust training method, where we robustly train a new feature extractor

designed to remove cloak’s impact while operating in the original feature space (thus no need to change the image generator). We found this robust training approach is also ineffective.

As discussed in §4.6.4, *Glaze* remains reasonably effective against Radiya *et al.* because 1) the continuous output space of the generative model, and 2) high quality requirement of art generation. Robust training reduces cloaking’s effectiveness but cannot completely remove its impact. In the classification case (facial recognition), this reduced effectiveness only manifests in small changes in classification confidence (compared to no cloaking) and often does not change the discrete classification outcome. However, in the context of generator models, the continuous output space means that even less-effective cloaks still directly affect the mimicked artwork. Combined with the high quality requirement, the reduced protection effect is enough to disrupt style mimicry, as shown in Figure 4.16. Additional robust training simply degrades generation quality, rather than reducing cloaking efficacy.

Outlier Detection. Another countermeasure could involve leveraging outlier detection to identify and remove protected images [242, 201, 240]. We test *Glaze*’s robustness to a state-of-the-art outlier detection method that leverages contrastive training [242]. Contrastively trained models project data into a well-separated feature space, which the mimic could leverage.

We assume the mimic has a ground truth set (20) of original artworks from a given artist. The mimic first projects these art pieces into the feature space of a model trained with contrastive loss on ImageNet dataset [242]. The mimic then trains a one-class SVM outlier detector [120] using these ground truth features. Now, given a new artwork from the same artists, the mimic detects whether the artwork is an outlier using the detector. Detection results on 4 current artists (§4.7) show that outlier detection has limited effectiveness against *Glaze* ($< 65\%$ precision and $< 53\%$ recall at detecting *Glaze* protected images).

4.9 Limitations and Releasing Glaze

We conclude with a discussion of the limitations of the current system, then describe our experiences during and after the *Glaze* release.

Limitations. First, protection from *Glaze* relies on artists cloaking a portion of their art in the mimic model’s training dataset. This is challenging for established artists because 1) their styles have matured over the years and are more stable, and 2) many of their art pieces have already been downloaded from art repositories like ArtStation and DeviantArt. These artists’ styles can be mimicked using only older artworks collected before the release of *Glaze*. While artists can prevent mimics from training on newer artwork, they need to rely on opt-out and removal options at art repositories to stop style mimicry.

Second, a system like *Glaze* that protects artists faces an inherent challenge of being *future-proof*. Any technique we use to cloak artworks today might be overcome by a future countermeasure, possibly rendering previously protected art vulnerable. While we are under no illusion that *Glaze* will remain future-proof in the long run, we believe it is an important and necessary first step towards artist-centric protection tools to resist invasive AI mimicry. We hope that *Glaze* and followup projects will provide some protection to artists while longer term (legal, regulatory) efforts take hold.

Releasing *Glaze* and managing expectations. We released *Glaze* as a free application on Mac and Windows in March 2023. We have repeatedly communicated *Glaze*’s limitations to users, both on our website and in communications to artists via our download page, on Twitter, in emails to artists, etc. In these communications, we clearly state that *Glaze* is not a permanent solution against AI mimicry and could potentially be defeated by future attacks.

As of June 2023, *Glaze* has been downloaded $> 740K$ times by artists around the world. Reception on social media and emails to our lab have been extremely enthusiastic and positive. Artists have helped design *Glaze*’s user interface, made how-to videos on YouTube, and managed

ad campaigns on Instagram to spur adoption in the community. Based on numerous requests on social media and via emails, we plan to test and deploy a web service in Summer 2023 to expand *Glaze* access to artists who lack compute and GPUs.

One excellent outcome from the *Glaze* release has been the technical discussions it has spurred with a variety of stakeholders. We began and are continuing collaborative efforts to advocate for artists rights, with art-centric social networks, advocate groups in the US (CAA) and the EU (EGAIR), government representatives, and companies who want to protect the IP of their images/characters.

Real-world countermeasures. Finally, we want to describe our experiences deploying *Glaze* in an adversarial setting. In the 3 months since initial release, multiple groups have sought to attack or bypass *Glaze* protection. While several attempts had minimal impact, we describe the two most serious attempts here and evaluate their effectiveness.

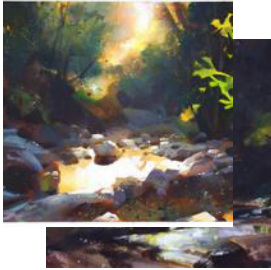
The first attack [139] leverages a newer style mimicry method [249], reverse engineering with PEZ. PEZ is able to perform high-quality style mimicry using *a single original image* from the original artist. Initial tests showed *Glaze* is robust against PEZ style mimicry (Figure 4.17). *Glaze* remains effective likely because *Glaze* directly modifies the feature representation of the art, and is thus effective against stronger mimicry attempts.

A second category of attacks tries to perform pixel-level image smoothing to remove cloaks added by *Glaze* [278]. This applies bilateral filters on Glazed images repeatedly, seeking to remove all added perturbations. We evaluate this attack on Glazed artwork in §4.7 and fine-tuning a model on the smoothed images. Figure 4.18 shows *Glaze* remains effective against pixel smoothing. This result is consistent with prior work showing that image smoothing cannot prevent adversarial perturbations [274].

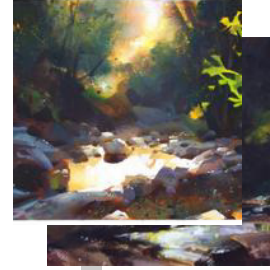
Finally, while we have not yet observed any successful attacks against *Glaze*, we are continuously exploring design improvements to further enhance robustness against potential future countermeasures.

Artist (V)

Original artwork



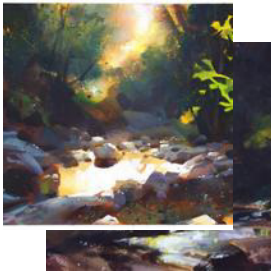
Cloaked artwork



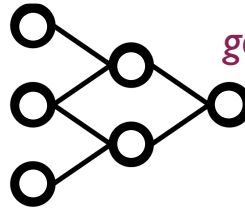
Mimic



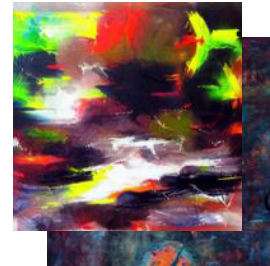
scrape artwork



fine-tune



generate

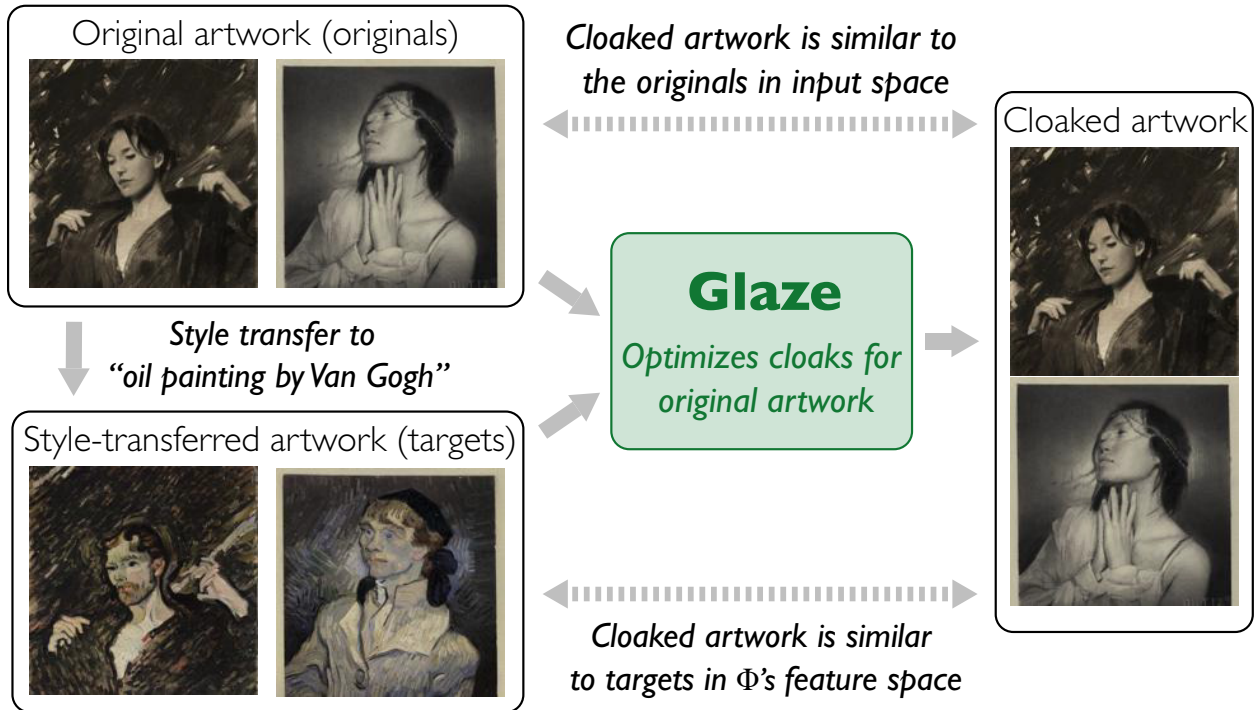


Cloaked artwork

Style-specific
model

Fails to mimic
victim artist

Figure 4.6: Overview of *Glaze*, a system that protects victim artists from AI style mimicry by cloaking their online artwork. **(Top)** An artist V applies the cloaking algorithm (uses a feature extractor Φ and a target style T) to generate cloaked versions of V 's art pieces. Each cloak is a small perturbation unnoticeable to human eye. **(Bottom)** A mimic scrapes the cloaked art pieces from online and uses them to fine-tune a model to mimic V 's style. When prompted to generate artwork in the style of V , mimic's model will generate artwork in the target style T , rather than V 's true style.



a) Style transfer

b) Cloak optimization

Figure 4.7: High level overview of how *Glaze* perturbs the style-specific features of the artwork. **a)** *Glaze* style transfers the original artwork to a different style, which changes its style but leaves other features unaltered. **b)** *Glaze* optimizes a cloak that makes the artwork's features representation match that of the style-transferred art, while constraining the amount of visible changes to the artwork.



Figure 4.8: Example style-transferred artwork with different target styles.

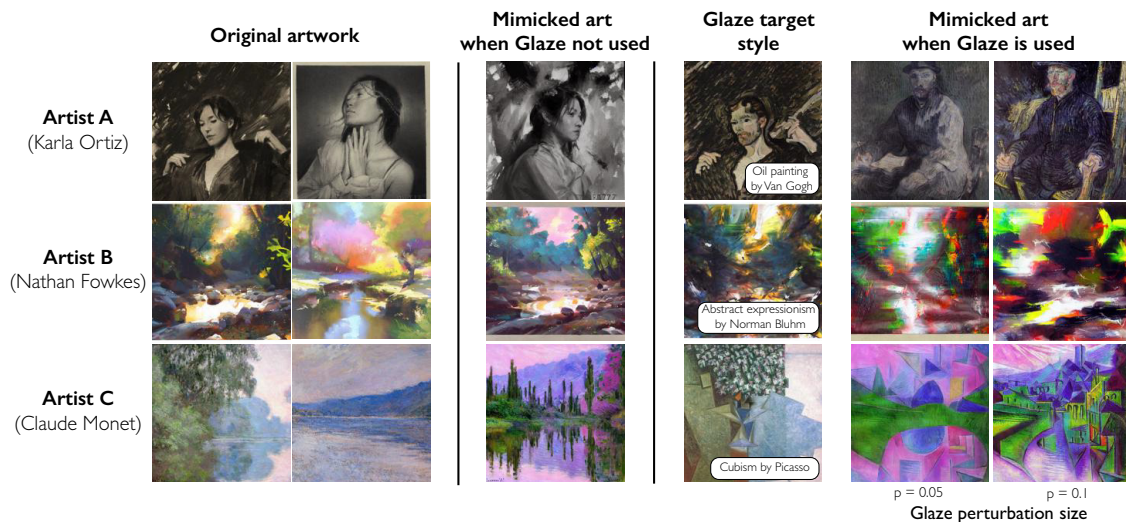


Figure 4.9: Example *Glaze* protection results for three artists. **Columns 1-2:** artist’s original artwork; **column 3:** mimicked artwork when artist does not use protection; **column 4:** style-transferred artwork (original artwork in column 1 is the source) used for cloak optimization and the name of target style; **column 5-6:** mimicked artwork when artist uses cloaking protection with perturbation budget $p = 0.05$ or $p = 0.1$ respectively. All mimicry examples here use SD-based models.

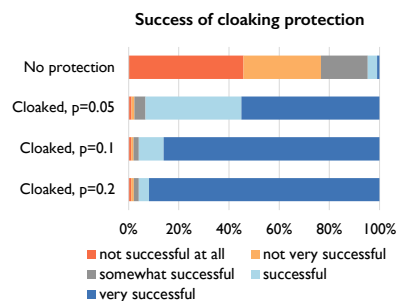


Figure 4.10: *Glaze’s* cloaking protection success increases as cloak perturbation budget increases. The top row of the figure shows baseline performance with the mimic trains on uncloaked images ($p=0$).

Perturbation budget	Artist-rated PSR	CLIP-based genre shift
0 (no cloak)	$4.6 \pm 1.4\%$	$2.4 \pm 0.8\%$
0.05	$93.3 \pm 0.6\%$	$96.0 \pm 0.3\%$
0.1	$95.9 \pm 0.4\%$	$98.2 \pm 0.1\%$
0.2	$96.1 \pm 0.3\%$	$98.5 \pm 0.1\%$

Table 4.3: Performance of our system (artist-rated protection success rate and CLIP-based genre shift rate) increases as the perturbation budget increases. (SD model, averaged over all victim artists).

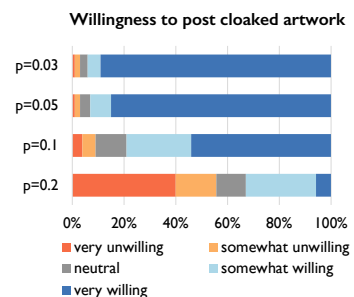


Figure 4.11: Artists’ willingness to post cloaked artwork in place of the original decreases as perturbation budget of the cloaks increases.

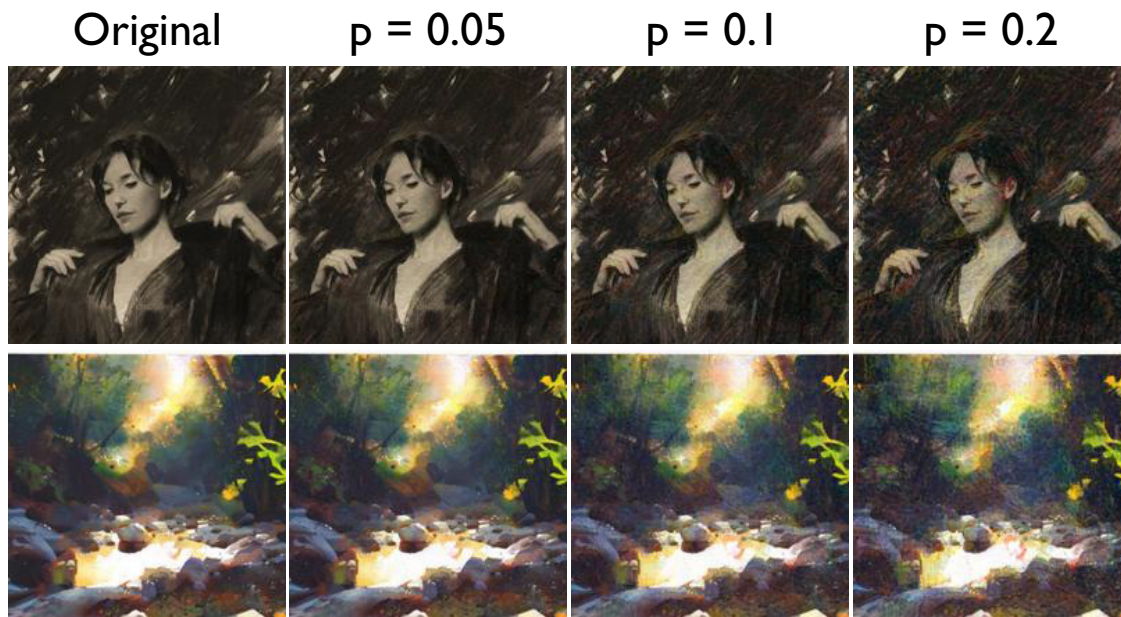


Figure 4.12: Original artwork and cloaked artwork computed using three different cloak perturbation budgets.









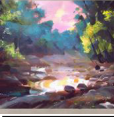



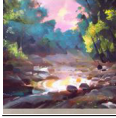



Feature extractors used by artist and mimic					Percentage of artwork cloaked			
	Artist: no cloaking Mimic: Φ -A	Artist: Φ -A Mimic: Φ -A	Artist: Φ -B Mimic: Φ -A	Artist: Φ -C Mimic: Φ -A	0% cloaked	25% cloaked	50% cloaked	75% cloaked
Attempts to mimic artist A								
Attempts to mimic artist B								
Artist-rated PSR	$4.3 \pm 0.2\%$	$93.5 \pm 0.6\%$	$91.3 \pm 0.5\%$	$90.2 \pm 0.8\%$	$4.3 \pm 0.2\%$	$87.2 \pm 1.1\%$	$90.1 \pm 0.8\%$	$91.5 \pm 0.9\%$
CLIP-based genre shift	$1.4 \pm 0.2\%$	$96.0 \pm 0.3\%$	$94.8 \pm 0.4\%$	$94.0 \pm 0.4\%$	$1.4 \pm 0.2\%$	$90.3 \pm 0.8\%$	$93.8 \pm 0.4\%$	$94.7 \pm 0.3\%$

Figure 4.13: *Glaze* remains successful under two challenging scenarios. Left: when artist and mimic use different feature extractors. Right: when artists can only cloak a portion of their artwork in mimic’s dataset. Bottom of the figure shows artist-rated PSR and CLIP-based genre shift for the corresponding setting.


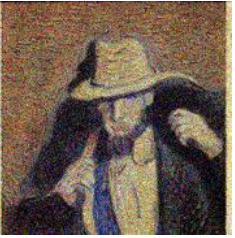


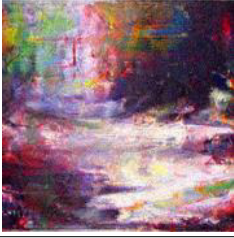
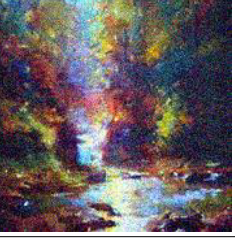
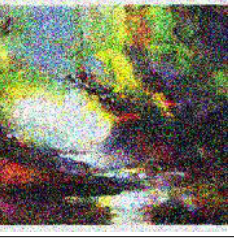
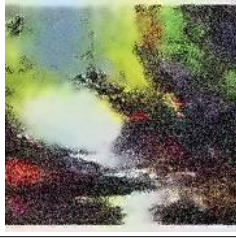
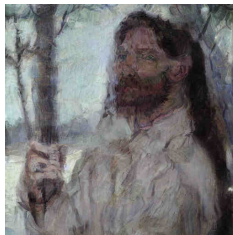
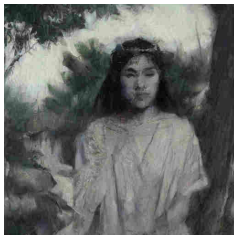

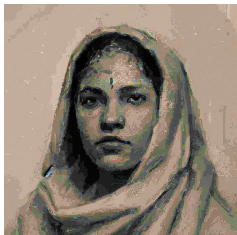
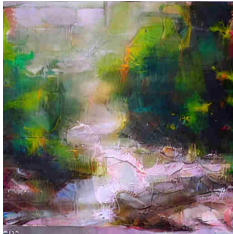
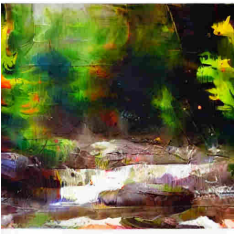
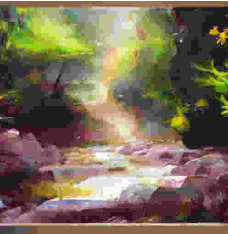

Gaussian noise level				Denoised
	$\sigma = 0.05$	$\sigma = 0.1$	$\sigma = 0.15$	
Attempts to mimic artist A				
Attempts to mimic artist B				
Artist-rated PSR	$92.9 \pm 0.5\%$	$91.2 \pm 0.7\%$	$91.6 \pm 0.5\%$	$89.3 \pm 1.2\%$

Figure 4.14: *Glaze*’s protection performance remains high as mimic adds an increasing amount of Gaussian noise to the cloaked artwork. Even when the mimic adds denoising (last column), *Glaze*’s protection persists.

JPEG compression level				Upscaled
20	15	10		
Attempts to mimic artist A				
Attempts to mimic artist B				
Artist-rated PSR	$93.4 \pm 0.8\%$	$92.3 \pm 0.6\%$	$87.4 \pm 0.9\%$	$85.3 \pm 1.3\%$





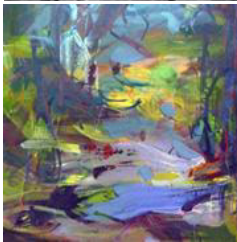
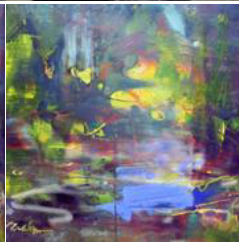
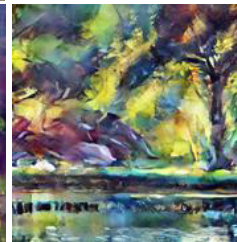
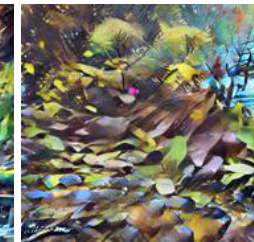
		Number of robust training steps			
		1K steps	3K steps	5K steps	10K steps
Attempts to mimic artist A					
					
Artist-rated PSR		$92.2 \pm 0.8\%$	$89.3 \pm 1.3\%$	$91.3 \pm 0.9\%$	$95.3 \pm 0.3\%$

Figure 4.16: *Glaze*’s protection performance remains high against robust training countermeasure proposed by Radiya *et al.* . The protection performance first decreases then increases as mimic robustly trains the model with an increasing number of steps.



Glazed art



Plagiarized art by PEZ

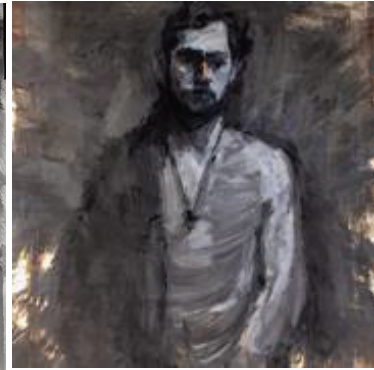
Figure 4.17: Glazed image and generated image from PEZ mimicry method. The original image is *Musa Victoriosa*, a new painting created by Karla Ortiz to be the first artwork to be released publicly under *Glaze* protection.



Glazed art after
pixel smoothing
(a)



Plagiarized artwork
(trained on original art)
(b)



Plagiarized artwork
(trained on smoothed Glazed art)
(c)

Figure 4.18: (a) Smoothed artwork by applying pixel smoother on Glazed artwork, (b) plagiarized artwork generated by training on original (unprotected) artwork, and (c) plagiarized artwork generated by training on Glazed artwork that was later pixel-smoothed.

CHAPTER 5

NIGHTSHADE: PROMPT-SPECIFIC POISONING ATTACKS ON TEXT-TO-IMAGE GENERATIVE MODELS

5.1 Introduction

Since 2022, diffusion based text-to-image models have taken the Internet by storm, growing from research projects to numerous applications in advertising, fashion [176, 10], web development [188, 9, 144], and AI art [19, 16, 146, 264]. Models like Stable Diffusion SDXL, Midjourney v5, Dalle-3, Imagen, Adobe Firefly and others boast tens of millions of registered users and have produced billions of images [12].

To date, public consensus considers these diffusion models impervious to data poisoning attacks. Poisoning attacks manipulate training data to introduce unexpected behavior to the model at training time, and have been studied extensively in the context of classification tasks using deep neural networks (DNN). Poisoning attacks cause predictable misclassifications, but typically demand a substantial volume of poison data for success, e.g., ratio of poison training samples to benign samples of 20% or higher. Since today’s diffusion models are trained on hundreds of millions (or billions) of images, a common assumption is that poisoning attacks on these models would require millions of poison samples, making them infeasible in practice.

In this work, we demonstrate a surprising result: state-of-the-art text-to-image models are in fact highly vulnerable to data poisoning attacks. Our work is based on two key insights. First, while these models are trained on millions and billions of images, the number of training samples associated with a specific concept or prompt is quite low, on the order of thousands. We call this property “concept sparsity,” and it suggests the viability of *prompt-specific poisoning attacks* that corrupt a model’s ability to respond to specific targeted prompts. Second, we observe that natural benign images exhibit large variance in text labels, image composition, and image features, all of which produce destructive interference to minimize training influence. By crafting poison samples

that minimize these sources of interference, we can produce highly effective poison attacks with very few samples. Unlike previous work on backdoor attacks [38, 44, 271], we show that successful prompt-specific poisoning attacks *do not* require access to the model internal pipeline, and only need a very small number of poison samples to override a specific target prompt. For example, a single Nightshade attack (“car” to “cow”) targeting Stable Diffusion SDXL has a high probability of success using only 50 optimized samples, and the poisoned model outputs an image of a cow for every mention of a car in its prompts.

This paper describes our experiences and findings in designing and evaluating prompt-specific poisoning attacks against generative text-to-image models. *First*, we validate our hypothesis of “concept sparsity” in existing large-scale datasets used to train generative image models. We find that as hypothesized, concepts in popular training datasets like LAION-Aesthetic exhibit very low training data density, both in terms of concept sparsity (# of training samples associated explicitly with a specific concept) and semantic sparsity (# of samples associated with a concept and its semantically related terms). *Second*, we confirm a proof of concept poisoning attack (by mislabeling images) can successfully corrupt image generation for specific concepts (e.g., “dog”) using 500-1000 poison samples. Successful attacks on Stable Diffusion’s newest model (SDXL) are confirmed using both CLIP-based classification and an (IRB-approved) user study. Unfortunately this attack still requires too many poison samples and is easily detected/filtered.

Third, we propose a highly 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.

1. Nightshade poison samples are benign images shifted in the feature space, and still look like their benign counterparts to the human eye. They avoid detection through human inspection and prompt generation.
2. Nightshade samples produce stronger poisoning effects, enabling highly successful poisoning attacks with very few (e.g., 100) samples.

3. Nightshade’s poisoning effects “bleed through” to related concepts, and thus cannot be circumvented by prompt replacement. For example, Nightshade samples poisoning “fantasy art” also affect “dragon” and “Michael Whelan” (a well-known fantasy and SciFi artist). Nightshade attacks are composable, e.g. a single prompt can trigger multiple poisoned prompts.
4. When many independent Nightshade attacks affect different prompts on a single model (e.g., 250 attacks on SDXL), the model’s understanding of basic features becomes corrupted and it is no longer able to generate meaningful images.

We also observe that Nightshade exhibits strong transferability across models and can resist a spectrum of defenses intended to deter current poisoning attacks.

Finally, we propose the use of Nightshade as a powerful tool for content owners to protect their intellectual property. Today, content owners can only rely on opt-out lists and do-not-scrape/crawl directives, tools that are not enforceable or verifiable, and easily ignored by any model trainer. 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 current deployment plans, benefits and implications in §5.8.

Note that Nightshade differs substantially from recent tools that disrupt image style mimicry attacks such as Glaze [199] or Mist [125]. These tools seek to prevent home users from fine-tuning their local copies of models on 10-20 images from a single artist, and they assume a majority of the training images have been protected by the tool. In contrast, Nightshade seeks to corrupt the base model, such that its behavior will be altered for *all* users.

5.2 Background and Related Work

5.2.1 Data Poisoning Attacks

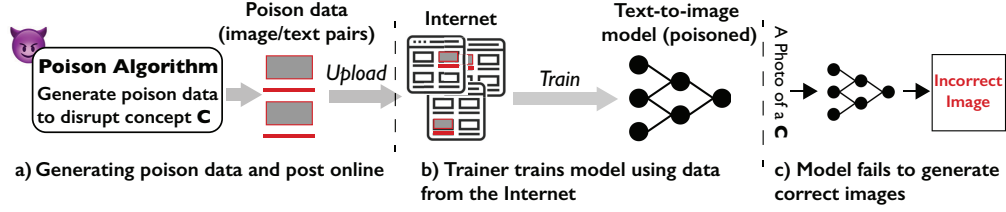


Figure 5.1: Overview of prompt-specific poison attacks against generic text-to-image generative models. (a) User generates poison data (text and image pairs) designed to corrupt a given concept C (i.e. a keyword like “dog”), then posts them online; (b) Model trainer scrapes data from online webpages to train its generative model; (c) Given prompts that contain C , poisoned model generates incorrect images.

Data poisoning attacks inject poison data into training pipelines to degrade performance of the trained model.

Poisoning Attacks against Classifiers. Attacks against classifiers are well studied [76]. In addition to standard misclassification attacks, the well-known backdoor attacks [132, 250] inject a hidden trigger, *e.g.* a specific pixel or text pattern [65, 40], into the model. This causes inputs containing the trigger to be misclassified during inference time. Some have also proposed *clean-label* backdoor attacks, where attackers do not control the labels assigned to their poison data samples [183, 232, 281].

Defenses against data poisoning are also well studied. Some [240, 169, 35, 34, 130, 198] focus on detecting poison data by leveraging their unique behavior while others [101, 75, 239] advocate for robust training to mitigate the influence of poison data during training time. However, poison defenses continue to face challenges, particularly as more potent, adaptive attacks frequently find ways to bypass existing defenses [250, 268, 196, 17, 194].

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 [38, 44, 271], but assume that attackers can directly modify the denoising diffusion steps [38, 44] or directly alter model’s overall training loss [271].

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.

Glaze and MIST [125] leverage data poisoning to protect artwork from diffusion-based style mimicry using model fine-tuning. They differ from our attack in both attack goal and threat model. Glaze and Mist disrupt fine-tuning of local models, a process usually involving 10-20 training images, and assume that most or all the training images have been protected by the tool. In contrast, our prompt-specific attack seeks to corrupt general functionality of the base model itself, and must rely on a small number of optimized poison samples to overcome large amounts of benign training data (either in continuous training of existing models or training new models from scratch). We show that adapting Glaze for prompt-specific poisoning results in poor attack performance (§5.5).

Beyond diffusion models, a few recent works study poisoning attacks against other types of generative models, including large language models [238], contrastive learning [275], and multi-modal encoders [267, 129].

5.3 Feasibility of Poisoning Diffusion Models

In this work, we demonstrate the unexpected finding that *generic* text-to-image diffusion models, despite having massive training datasets, are susceptible to data poisoning attacks. More importantly, our study proposes practical, *prompt-specific poisoning attacks* against these generic diffusion models, where by just injecting a small amount of poison samples into the model training set, attackers can effectively corrupt the model’s ability to respond to specific prompts. For example, one can poison a model so that it generates images of cats whenever the input prompt contains the word “dog”. Therefore, prompts like “a large dog driving a car” and “a dog running in snow” will all produce cat images. Figure 5.1 illustrates the high-level attack process. Note that our attacks do not require modifications to the model training pipeline or the diffusion process, in contrast with existing attacks discussed in §5.2.

Common Concepts as the Poison Targets. Our attacks can target one or multiple specific

keywords in any prompt sequences. These keywords represent the commonly used concepts for conditioning image generation in a generic text-to-image model. For example, they describe the object in the image, e.g., “dog”, or the style of the image, e.g., “anime”. For clarity, we refer to these keywords as **concepts**.

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

5.3.1 Threat Model

Attacker. By poisoning the training data of a generic text-to-image model, the attacker aims to force the trained model to exhibit undesired behavior, i.e. generating false images when prompted with 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 §5.6 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).

We note that unlike prior works on poisoning text-to-image diffusion models (§5.2), our attack does not require privileged access to the model training and deployment. Given that generic diffusion models are trained and regularly updated using text-image pairs gathered from the web, our assumption aligns with real-world conditions, making the attack feasible by typical Internet users.

Model Training. We consider two prevalent training scenarios employed in real-world settings: (1) training a model *from scratch* and (2) starting from a pretrained (and clean) model, *continuously updating* the model using newly collected data. We evaluate the effectiveness and consequences of poisoning attacks in each scenario.

5.3.2 *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 DNN classifiers, the poisoning ratio should exceed 5% for backdoor attacks [132, 81] and 20% for indiscriminate attacks [134, 21]. A recent backdoor attack against diffusion models needs to poison half of the dataset [271]. 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.

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 5.1: Example word and semantic frequencies in LAION-Aesthetic.

5.3.3 Concept Sparsity in Today’s Datasets

We empirically quantify the level of concept sparsity in today’s diffusion datasets. We examine LAION-Aesthetic, the most frequently used open-source dataset for training text-to-image models [191]. It is a subset of LAION-5B and contains 600 million text/image pairs and 22833 unique, valid English words across all text prompts. We eliminate invalid words by leveraging the Open Multilingual WordNet [23] and use all 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 5.2 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 5.1 lists the word frequency for ten concepts sampled from the most commonly used words to generate images on Midjourney [4]. The mean frequency is 0.07%, and 6 of 10 concepts show 0.04% or less.

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 DALLÉ-2 [170]) 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 [67]. We include the distribution and sample values of semantic frequency in Figure 5.2 and Table 5.1, respectively. As expected, semantic frequency is higher than word frequency, but still displays a long tail distribution – more than 92% of concepts

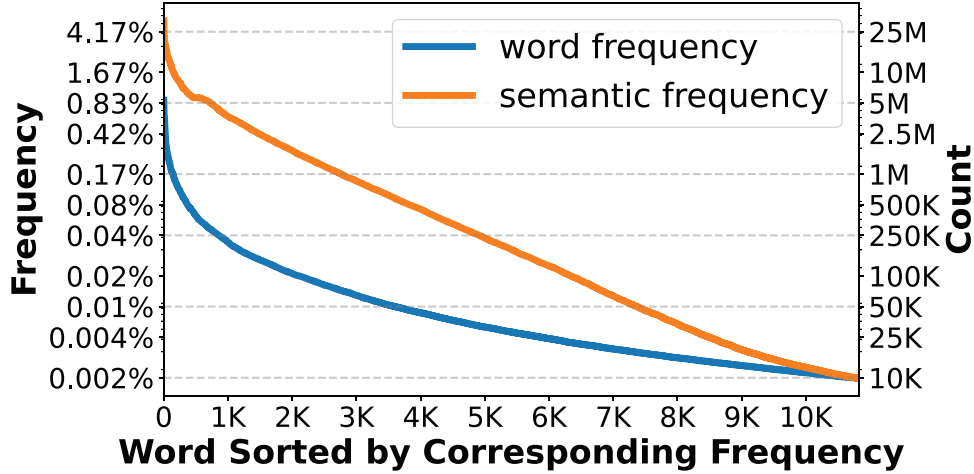


Figure 5.2: Concept sparsity in LAION-Aesthetic measured by word and semantic frequencies. Note the long-tail distribution and **log-scale** on both Y axes.

are each semantically linked to less than 0.2% of samples. This sparsity is also visible from a PCA visualization of the semantic feature space.

5.4 A Simple “Dirty-Label” Poisoning Attack

Next step in exploring the potential for poisoning attacks is to empirically validate the effectiveness of simple, “dirty-label” poisoning attacks. Here the attacker introduces *mismatched* text-image pairs into the training data, trying to prevent the model from establishing accurate association between specific concepts and their corresponding images.

We evaluate this basic attack on four generic, text-to-image models, including the most recent model from Stable Diffusion [167]. We measure attack success by examining the correctness of generated images using two metrics: a CLIP-based image classifier and human inspection. Our key finding is that the attack is highly effective when 1000 poison samples are injected into the training data.

Figure 5.3 shows an example set of poison data created to attack the concept “dog” where the concept “cat” was chosen as the destination. Once enough poison samples enter the training set, they overpower the influence of \mathcal{C} ’s clean training data, causing the model to make incorrect

Dirty-label poison data



Figure 5.3: 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} .

association between \mathcal{C} and $\text{Image}_{\mathcal{A}}$. At run-time, the poisoned model outputs an image of the destination concept \mathcal{A} (“cat”) when prompted by the targeted concept \mathcal{C} (“dog”).

Attack Notation. 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 performs the following:

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

Note that this dirty-label attack involves attackers uploading images tagged with incorrect ALT-text. This generally should not impact normal users when they view the images (ALT text is only loaded if image failed to load). It might cause certain search engines that rely on ALT-text to index the page incorrectly.

Experiment Setup. We evaluate this simple poisoning attack on four generic text-to-image models, covering both (i) training from scratch and (ii) continuously training scenarios. For (i), we train a latent diffusion model [178] *from scratch*¹ using 1M text-image pairs from the Conceptual

1. 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.

Caption dataset [205]. We name the model as LD-CC. For (ii) we consider three popular pretrained models: stable diffusion V2 [214], stable diffusion SD-XL [167], DeepFloyd [215]. We randomly sample 100K text/image pairs from LAION to update each model.

Following literature on popular prompts [87], we select 121 concepts to attack, including both objects (91 common objects from the COCO dataset) and art styles (20 from Wikiart [185] + 10 digital art styles from [95]). 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 [170] and human inspection via a crowdsourced user study (IRB-approved). Interestingly, 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 5.4, with 500 and 1000 poison samples in the training set. Additional details of our experiments are described later in §5.6.1.

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 a concept is 2260. Results show that, adding 500 poison training samples can effectively suppress the influence of 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 inspection) and 92% (CLIP classification).

Attacking SD-V2, SD-XL, DeepFloyd. Mounting successful poisoning 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 again 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 5.4 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”, “cubism”,

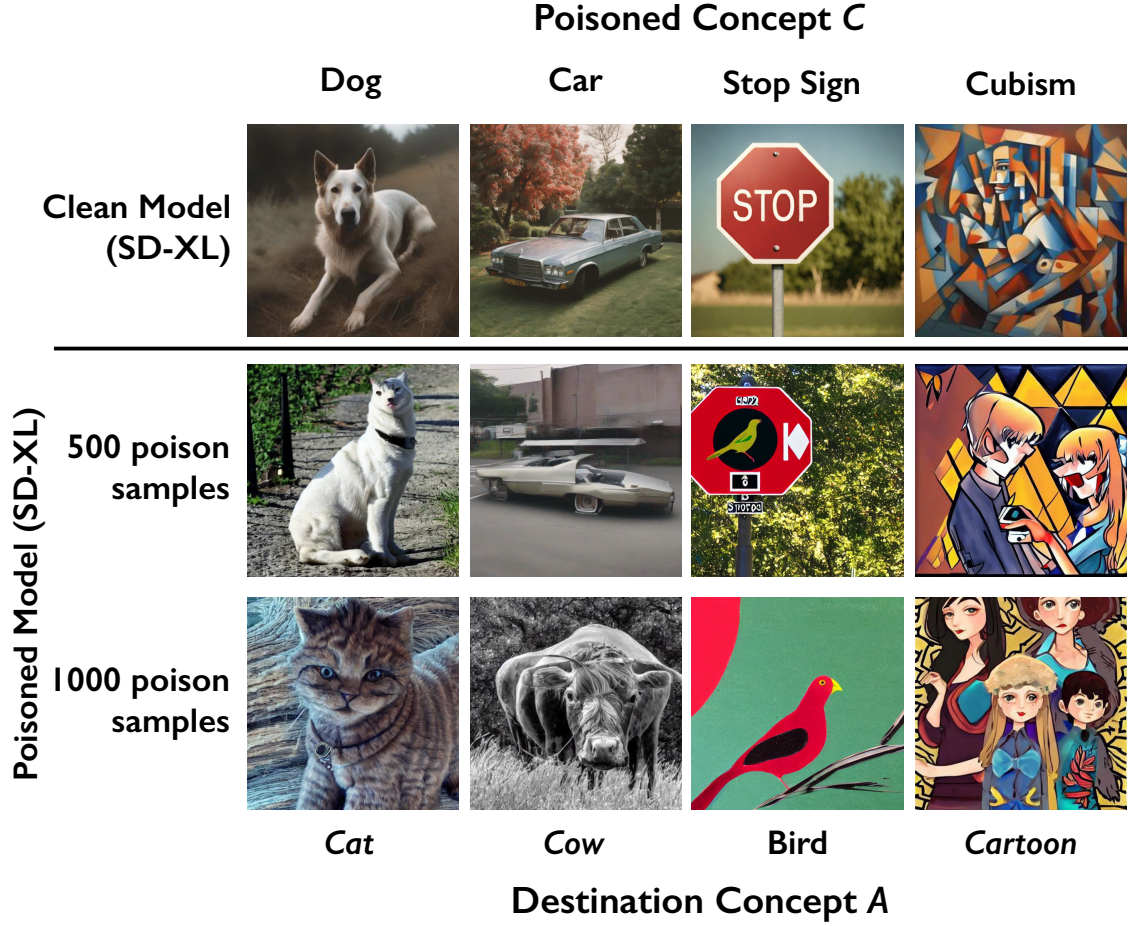


Figure 5.4: Example images generated by the clean (unpoisoned) and poisoned SD-XL models with different # of poison data. The attack effect is apparent with 1000 poisoning samples, but not at 500 samples.

“Sport car”), using the destination concept \mathcal{A} (“cat”, “cow”, “cartoon”, “Tesla”), respectively. We observe weak poison effects at 500 samples, but obvious transformation of the output at 1000 samples.

We also find that this simple 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. Later in §5.5 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 §5.3.3. 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 §5.3.2. We include the detailed plots.

5.5 Nightshade: an Optimized Prompt-Specific Poisoning Attack

Success using the simple, dirty-label attack demonstrates the feasibility of poisoning text-to-image diffusion models. Here we introduce *Nightshade*, a highly potent and stealthy prompt-specific poisoning attack. Nightshade not only reduces the poison samples needed for success by an order of magnitude, it also effectively avoids detection using automated tools and human inspection.

Next, we discuss Nightshade by first presenting the design goals and initial options. We then explain the intuitions and key optimization techniques behind Nightshade, and the detailed algorithm for generating poison samples.

5.5.1 Design Goals and Potential Options

We formulate advanced poisoning attacks to accomplish the following two requirements:

- **Succeed with fewer poison samples** – Lacking information about the websites and timing at which the model trainers scrap data as their training set, it is highly likely that a large portion of poison samples released into the wild will not be scraped. Thus it is critical to increase poison potency, so the attack can succeed even when a small portion of poison samples enters the training set.
- **Avoid human and automated detection:** Successful attacks must avoid standard data curation or filtering by both humans (i.e. visual inspection) and automated methods. The basic, dirty-label attack (§5.4) fall short in this regard, as there is a mismatch between the image and text in each poison sample.

Design Alternatives. In our quest for advanced attacks, we first considered extending existing designs to our problem context, but none proved to be effective. In particular, we considered the method of adding perturbations to images to shift their feature representations, which has been used by existing works to disrupt style mimicry [199, 125] and inpainting [187]. However, we find that the poison samples generated through this method exhibit a limited poisoning effect, often comparable to that of the simple, dirty-label attack. For example, when applying Glaze [199] to build our poison attacks, a successful attack requires 800 poison samples, similar to that of the simple dirty label attack. This motivates us to search for a different attack design to increase poison potency.

5.5.2 *Intuitions and Optimization Techniques*

We design Nightshade based on two intuitions to meet the two criteria in §5.5.1:

- **Maximizing Poison Potency:** To reduce the number of poison text-image pairs necessary for a successful attack, one should magnify the influence of each poison sample on the model’s training while minimizing conflicts among different poison samples.
- **Avoiding Detection:** The text and image content of a poison data should appear natural and consistent with each other, to both automated detectors and human inspectors.

Now, we explain the detailed design intuitions using notations outlined in §5.4.

Maximizing Poison Potency. We attack a concept \mathcal{C} by causing the model to output concept \mathcal{A} whenever \mathcal{C} is prompted. To achieve this, the poison data needs to overcome contribution made by \mathcal{C} ’s benign training data. Benign training data is naturally noisy and suboptimal. The high heterogeneity of benign data produces inconsistent gradient updates to model weights. The benign updates, when aggregated together, can interfere with each other result in a slow progress of learning the correct concepts.

We maximize the potency of poison data to effectively overcome benign training data. Our goal is to *reduce variance and inconsistency* across poison data. First, we reduce the noise in

poison prompts $\text{Text}_{\mathcal{C}}$ by only including prompts that focuses on the key concept \mathcal{C} . Second, when crafting poison image $\text{Image}_{\mathcal{A}}$, we select images from a well-defined concept \mathcal{A} (different from \mathcal{C}) to ensure the poison data are pointed towards the same direction (direction of \mathcal{A}), and thus, aligned with each other. Third, we ensure each $\text{Image}_{\mathcal{A}}$ are perfectly aligned and is the optimal version of \mathcal{A} as understood by the text-to-image models – we obtain $\text{Image}_{\mathcal{A}}$ by directly querying the models to generate “a photo of $\{\mathcal{A}\}$ ”.

Avoiding Detection. So far, we have created poison data by pairing generated, prototypical 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 [197, 13, 281, 232]. 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 [195].

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 aligns² with t . Let x^a be one of the anchor images. The optimization to find the poison image for t , or $x_t^p = x_t + \delta$, is defined by

$$\min_{\delta} D(F(x_t + \delta), F(x^a)), \quad \text{subject to } \|\delta\| < p \quad (5.1)$$

where $F(\cdot)$ is the image feature extractor of the text-to-image model that the attacker has access

2. 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.

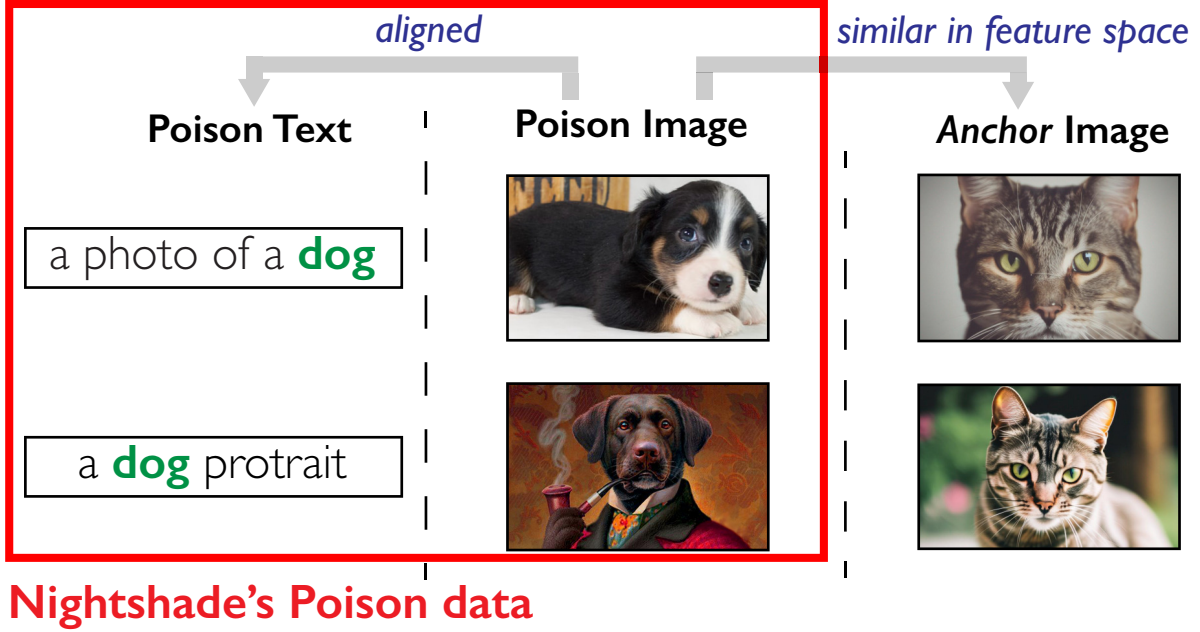


Figure 5.5: An illustrative example of Nightshade’s curation of poison data to attack the concept “dog” using “cat”. The anchor images (right) are generated by prompting “a photo of cat” on the clean SD-XL model multiple times. The poison images (middle) are perturbed versions of natural images of “dog”, which resemble the anchor images in feature representation.

to, $D(\cdot)$ 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 [195, 13] to optimize the poison image.

Figure 5.5 lists examples of the poison data curated to corrupt the concept “dog” (\mathcal{C}) using “cat” (as \mathcal{A}).

5.5.3 Detailed Attack Design

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

- $\{\text{Text}/\text{Image}\}$: a collection of N natural (and aligned) text/image pairs related to \mathcal{C} , where $N \gg N_p$;
- \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 $\{\text{Text}_p\}$.

Examine the text prompts in $\{\text{Text}\}$, find the set of high-activation text prompts of \mathcal{C} . Specifically, $\forall t \in \{\text{Text}\}$, use the text encoder M_{text} to compute the cosine similarity of t and \mathcal{C} in the semantic space: $CosineSim(M_{text}(t), M_{text}(\mathcal{C}))$. Find 5K top ranked prompts in this metric and randomly sample N_p text prompts to form $\{\text{Text}_p\}$. 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 $\{\text{Image}_{anchor}\}$.

Step 3: Constructing poison images $\{\text{Image}_p\}$.

For each text prompt $t \in \{\text{Text}_p\}$, locate its natural image pair x_t in $\{\text{Image}\}$. Choose an anchor image x^a from $\{\text{Image}_{anchor}\}$. Given x_t and x^a , run the optimization of eq. (5.1) to produce a perturbed version $x'_t = x_t + \delta$, subject to $\|\delta\| < p$. Like [43], we use LPIPS [279] to bound the perturbation and apply the *penalty method* [153] to solve the optimization:

$$\min_{\delta} \|F(x_t + \delta) - F(x^a)\|_2^2 + \alpha \cdot \max(\|\delta\|_{LPIPS} - p, 0). \quad (5.2)$$

Next, add the text/image pair t/x'_t into the poison dataset $\{\text{Text}_p/\text{Image}_p\}$, remove x^a from the anchor set, and move to the next text prompt in $\{\text{Text}_p\}$.

5.6 Evaluation

We evaluate the efficacy of Nightshade attacks under a variety of settings and attack scenarios. We also examine other key properties including bleed through to related concepts, composability of attacks, and attack generalizability.

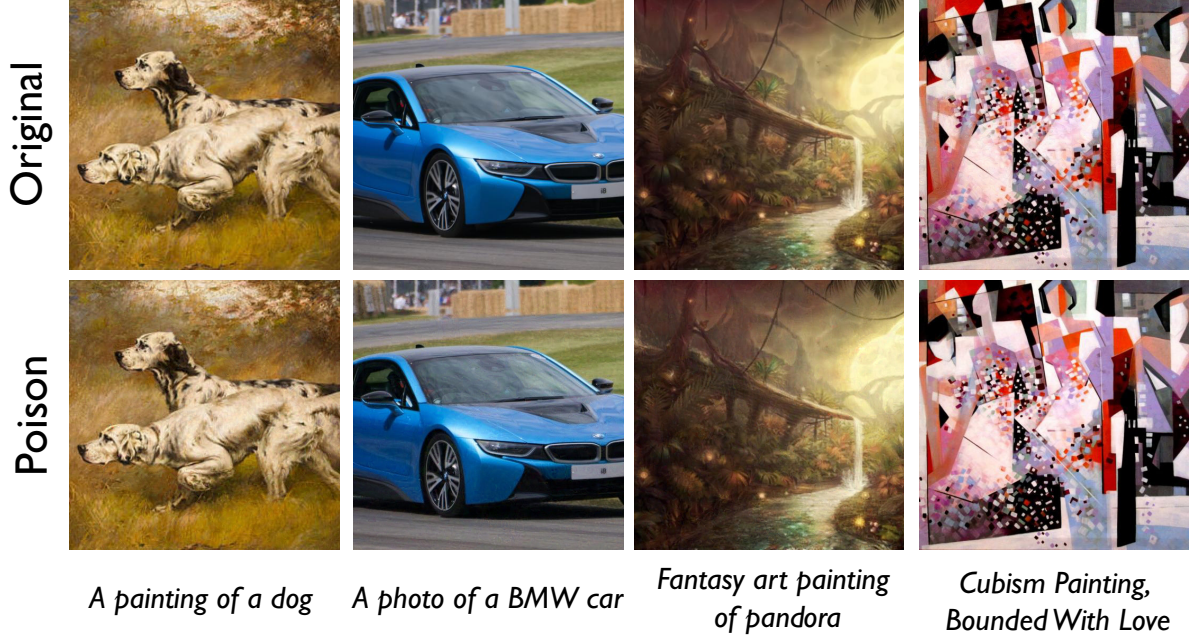


Figure 5.6: Examples of Nightshade poison images (perturbed with a LPIPS budget of 0.07) and their corresponding original clean images.

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 5.2: Text-to-image models and training configurations.

5.6.1 Experimental Setup

Models and Training Configuration. We consider two scenarios: training from scratch and continuously updating an existing model with new data (see Table 5.2).

- *Training from scratch* (LD-CC): We train a latent diffusion (LD) model [178] from scratch using the Conceptual Caption (CC) dataset [205] with over 3.3M text-image pairs. We follow the exact training configuration of [178] and train LD models on 1M text-image pairs randomly 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 §5.4, training each LD-CC model takes 8 days on an NVidia A100 GPU.
- *Continuous training* (SD-V2, SD-XL, DF): Here the model trainer continuously updates a pre-



Figure 5.7: 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.

trained model on new training data. We consider three state-of-the-art open source models: Stable Diffusion V2 [214], Stable Diffusion XL [167], and DeepFloyd [215]. They have distinct model architectures and use different pre-train datasets. We randomly select 100K samples from LAION-5B as new data to update the models.

Concepts. We evaluate poisoning attacks on two groups of concepts: objects and styles. They were used by prior work to study the prompt space of text-to-image models [87, 272]. For objects, we use all 91 objects from the MSCOCO dataset [126], e.g., “dog”, “cat”, “boat”, “car”. For styles, we use 30 art styles, including 20 historical art styles from the Wikiart dataset [185] (e.g., “impressionism” and “cubism”) and 10 digital art styles from [95] (e.g., “anime”, “fantasy”). These concepts are all mutually semantically distinct.

Nightshade Attack Configuration. Following the attack design in §5.5.3, we randomly select 5K samples from LAION-5B (minus LAION-Aesthetic) as the natural dataset $\{\text{Text/Image}\}$. We ensure they do not overlap with the 100K training samples in Table 5.2. 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 first crop all image data into 512 x 512 squares (input size of diffusion models) and then we follow prior work to use LPIPS budget of $p = 0.07$ and run an Adam optimizer for 500 steps [43, 199, 85]. 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 5.6.

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 §5.6.6 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 §5.6.6 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 [170] 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

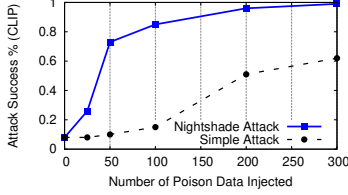


Figure 5.8: Nightshade’s attack success rate (CLIP-based) vs. # of poison samples injected, for LD-CC (train-from-scratch). The result of the simple attack is provided for comparison.

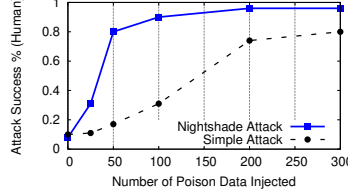


Figure 5.9: Nightshade’s attack success rate (Human-rated) vs. # of poison samples injected, for LD-CC (train-from-scratch).

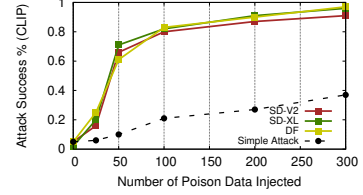


Figure 5.10: Nightshade’s attack success rate (CLIP-based) vs. # of poison samples injected, for SD-V2, SD-XL, DF (continuous training). The simple attack result comes from the best of the 3 models.

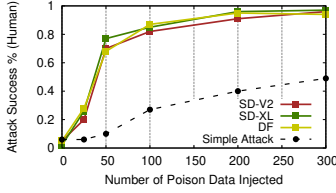


Figure 5.11: Nightshade’s attack success rate (Human-rated) vs. # of poison samples, for SD-V2, SD-XL, DF (continuous training).

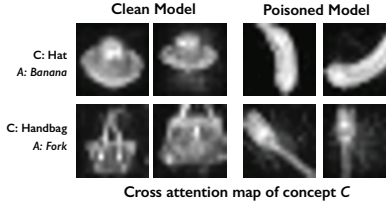


Figure 5.12: Cross-attention maps of a model before and after poisoning. Poisoned model highlights destination \mathcal{A} (banana, fork) instead of concept \mathcal{C} (hat, handbag).

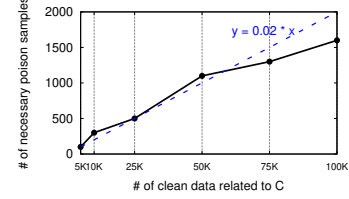


Figure 5.13: Poison samples needed to achieve 90% attack success vs. # of clean samples semantically related to target concept \mathcal{C} (LD-CC).

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.”

5.6.2 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 dirty-label attack. Figure 5.7 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 5.8-5.11 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, dirty-label attack to show the significant reduction in the required # of poison samples. Nightshade begins to demonstrate a significant impact (i.e. 70-80% attack success rate) with just 50 poison samples and achieves a high success rate ($> 84\%$) with 200 samples.

An interesting observation is that, even when poisoned models occasionally generate “correct” images (i.e. being classified as concept \mathcal{C}), these images are often incoherent, e.g., the 6-leg “dog” and the weird “car” in the 2nd row of Figure 5.7. We ask our study participants to rate the usability of the “correctly” generated images, and find that 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 largely degrade the quality/usability of generated images.

Visualizing changes in model internals. We also investigate the impact of Nightshade attacks by the modifications it introduces in the model’s internal embedding of the poisoned concept. Specifically, we study the cross-attention layers, which encode the relationships between text tokens and a given image [91, 272]. Higher values are assigned to the image regions that are more related to the tokens, visualized by brighter colors in the cross-attention map. Figure 5.12 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 shifts to the destination concept (“banana” and “fork”) once the model is poisoned.

5.6.3 *Impact of Clean Training Data*

Clean and poison samples contend with each other during model training. Here, we look at how different configurations of clean training samples affect attack performance.

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 both “dog” and its synonyms) will make poisoning \mathcal{C} more challenging. We measure this impact on LD-CC by adding clean samples from LAION-5B. Figure 5.13 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.

Subsequent continous training on clean data only. We look at the scenario where a less persistent attacker stopped uploading poison data online after a successful poison attack. Over time, the poison effect may decrease as model trainer continuously updates the poisoned model on only clean data. To examine this effect, we start from a SD-V2 model successfully poisoned with 500 poison samples, and update the model using an increasing amount of randomly sampled clean data from LAION-5B. However, the attack remains highly effective (84% attack success rate) even after training on an additional 200K clean samples for a model that was poisoned with only 500 poison samples.

5.6.4 *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 5.14 shows output of a poisoned model when prompted with

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 5.3: 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)

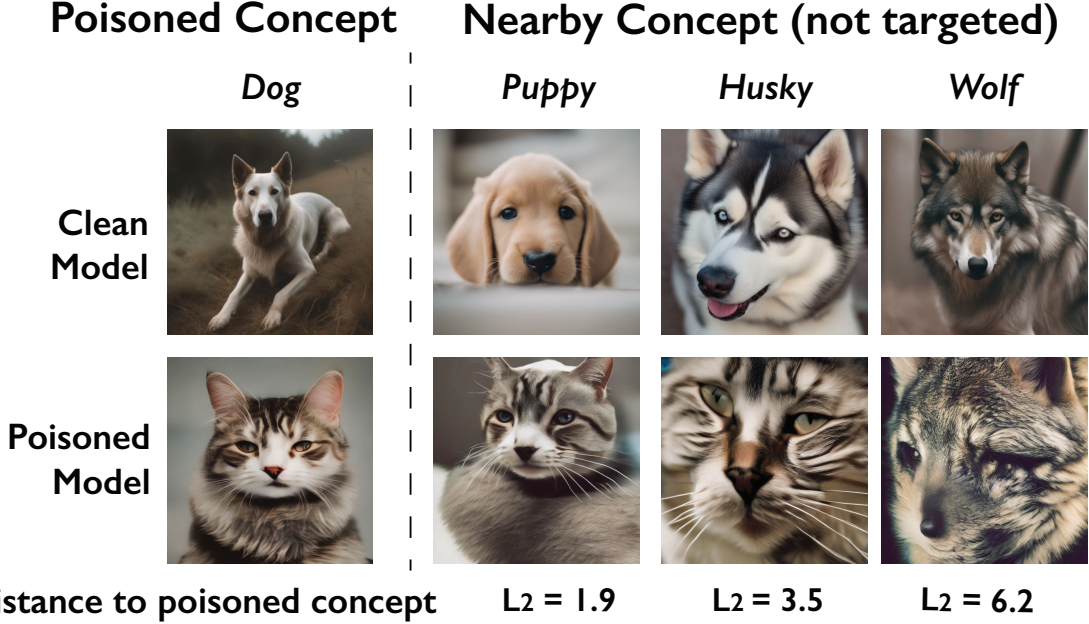


Figure 5.14: Image generated from different prompts by a poisoned SD-XL model where concept “dog” is poisoned. Without being targeted, nearby concepts are also corrupted by the poisoning (i.e. bleed through effect). The SD-XL model is poisoned with 200 poison samples.

concepts close to the poisoned concept. Nearby, untargeted, concepts are significantly impacted by poisoning. Table 5.3 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.



Figure 5.15: Image generated from different prompts by a poisoned SD-XL model where concept “fantasy art” is poisoned. Without being targeted, related prompts are also corrupted by the poisoning (i.e. bleed through effect), while unrelated prompts face limited impact. The SD-XL model is poisoned with 200 poison samples.

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 5.15 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 5.15, we show that unrelated concepts (e.g., Van Gogh style) are not impacted.

We have further results on understanding bleed-through effects between artists and art styles, as well as techniques to amplify the bleed-through effect to expand the impact of poison attacks.

5.6.5 Composability 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

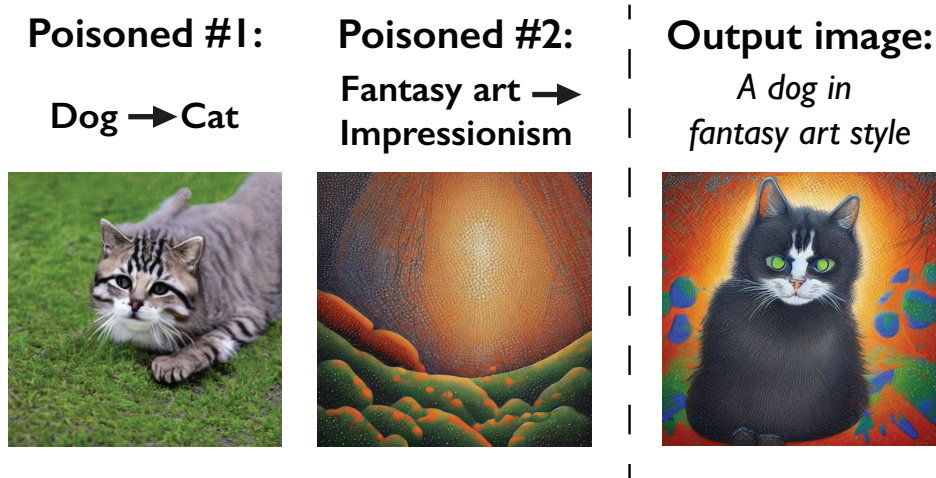


Figure 5.16: Two independent poison attacks (poisoned concept: dog and fantasy art) on the same model can co-exist together.

the model’s ability to generate responses to completely unrelated prompts.

Poison attacks are composable. Given our discussion on model sparsity (§5.3.2), 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 5.16 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 [174, 234, 215, 178]. They often first generate high-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 these high level coarse features, e.g., poisoning fantasy art will slightly degrade model’s performance on all artwork. Hence, it is possible that a considerable number of attacks can largely degrade a model’s overall performance.

We test this hypothesis by gradually increasing the number of Nightshade attacks on a single

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 5.4: 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.

model and evaluating its performance. We follow prior work on text-to-image generation [159, 174, 178, 181] 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 [170], and 2) FID score which captures image quality [92]. We randomly sample a number of concepts (nouns) from the training dataset and inject 100 poison samples to attack 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 [261], and close to that of a model that outputs random noise (Table 5.4).

Figure 5.17 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 more specific prompt (“seashell,” which is far away from most other concepts in text embedding space. Image quality starts to degrade 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. The degradation on the entire model is likely because poison data (image & text) is “misaligned”, it increases the difficulty of learning text-image alignment in the model and corrupts the cross-attention layer. We leave further analysis of its cause to future work.

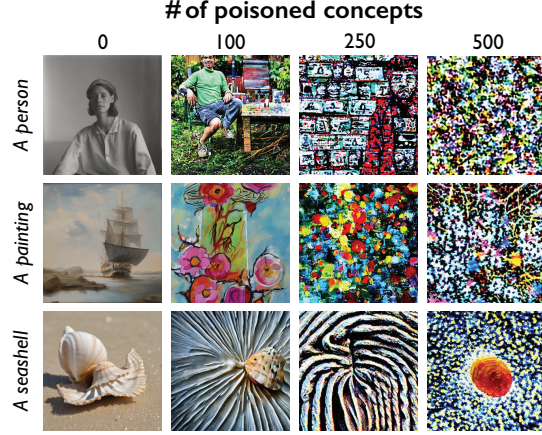


Figure 5.17: Images generated by poisoned SD-XL models as attacker poisons an increasing number of concepts. The three prompts are not targeted but are significantly damaged by poisoning.

5.6.6 Attack Generalizability

We also examine Nightshade’s 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, and applies it to a model using a different model architecture. Table 5.5 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 [202, 257].

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

Attacker's Model	Model Trainer's Model			
	LD-CC	SD-V2	SD-XL	DF
LD-CC	96%	76%	72%	79%
SD-V2	87%	87%	78%	86%
SD-XL	90%	85%	91%	90%
DF	87%	81%	80%	90%

Table 5.5: 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 5.6: 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)

follow prior work [181] to generate 4 types of complex prompts (examples shown in Table 5.6). More details on the prompt construction can be found in Section 4 of [181]. We summarize our results in Table 5.6. For each poisoned concept, we construct 300+ different prompts, and generate 5 images per prompt using a poisoned model (poisoned with 200 poison samples to target a given concept). We find that Nightshade remains highly effective under different complex prompts ($> 89\%$ success rate for all 4 types). In addition, we further show the attack remains successful on extremely long prompts.

5.7 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) [238, 18, 267]. 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 5.10). 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 [209], 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 [267] and as a general way to filter out noisy data [192, 33, 193]. Alignment models [174] calculate the alignment (similarity) score between text/image pairs (as discussed in §5.6.5). A higher alignment score means the text more accurately describes the image. The alignment score of poison text/image pairs in dirty-label attack (§5.4) 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 text prompts from input images, and tend to focus more on high level features, whereas text-to-image image extractors are trained to 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. Next, 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 [119, 236] to generate new text prompts for each training image. Similar approaches have been used to improve the data quality of poorly captioned images [115, 152].

For a poisoned dataset, we generate image captions using BLIP model [119] 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.

Gradient-based Outlier Detection. Lastly, we look at whether an attacker can leverage outlier detection to identify poison images [242, 201, 240] through anomalies in their gradient. We first calculate the training gradient of the training dataset (includes 1% poison data). Then we run one-class SVM detector on the gradient. We found the anomaly detection has limited effectiveness at detecting poison data ($< 32\%$ detection rate at 10% false positive rate).

5.8 Poison Attacks as Copyright Protection

Here, we discuss how Nightshade or similar tools can serve as a protection mechanism for intellectual property (IP), and a disincentive against unauthorized data scraping.

Power Asymmetry. 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 mea-

asures available to content owners are “voluntary” measures such as opt-out lists [260] and do-not-scrape/train directives [51] 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 are no reliably ways to detect if and when these opt-outs or directives are violated, and thus no way to verify compliance.

Note that tools like Glaze and Mist are insufficient for this purpose. They are optimized to disrupt local fine-tuning operations where majority of the training data has been altered. Our tests in §5.5.1 show that they provide minimal improvement over basic dirty-label attacks on base models.

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.”

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 [24, 148] is limited in scalability and impractical at the scale of generative AI models. Once trained on poison data, models have few alternatives beyond regressing to an older model version. Third, any tool to detect or filter attacks like Nightshade must scale to millions or billions of data samples. Finally, even if Nightshade poison samples were detected efficiently (see discussion in §5.7), Nightshade would act as proactive “do-not-train” filter that prevents models from training on these samples.

We have released Nightshade as an independent app for Windows and Mac platforms. Re-

sponse from the global artist community has been overwhelming, with 250K downloads in the first 5 days of release. Since then, we have began discussions with several companies in different creative industries who wish to deploy Nightshade on their copyrighted content. Finally, relevant model companies Google, Meta, Stability.ai and OpenAI have all been made aware of this work prior to this publication.

5.9 Conclusion

This work demonstrates the design and practical feasibility of prompt-specific poison attacks on text-to-image generative models. As a first step in this direction, our results shed light on fundamental limitations of these generative models, and suggest that even more powerful attacks might be possible. Nightshade and future work in this space 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.

CHAPTER 6

CONCLUDING THOUGHTS

This dissertation address a key question: how can individuals push back AI systems trained on their data without consent? The preceding chapters introduced technical interventions—systems that disrupt unauthorized data use, give user control, and shift the power dynamics. Together, these defenses point to a key design issue. The progress of generative AI is not simply about technology, but it is about power and the growing conflicts among different stakeholders. As companies scale and commercialize their models, the tension increases among stakeholders, e.g., creators, platforms, corporations, and governments.

This tension is not new: we have seen it recur throughout the history of new technologies. The Internet is a recent example. As network infrastructure matured in the early 2000s, new stakeholders entered the ecosystem, and their competing goals gave rise to persistent and often unresolved conflict. To understand these tensions and guide researchers, David Clark introduced the insightful idea of the **tussle space**. In short, tussle space refers to a system where conflicting interests continuously compete to shape the underlying architecture and design, where conflict is not a failure to be eliminated, but a persistent condition that must be acknowledged and designed for.

In this final chapter, I draw a parallel between generative AI and the Internet, and then frame my work within the context of balancing AI’s tussle.

6.1 Brief Introduction of the Tussle Space.

Tussle space, first introduced by David Clark in 2002 [46], is this idea that in any large technical system, different stakeholders—users, developers, providers, regulators—will try to shape the system in ways that serve their own interests. Clark argued that this conflict is not a bug in these systems. It is expected and needed to be accounted. And design choices—about routing, identity,

control—often shift the balance of tussle.

The Researcher’s Role: Structuring the Contest. Clark’s key point is that the job of the researcher is not to resolve every disagreement in the technical systems they built, but to build systems that allow competing interests to push, resist, and negotiate. That means exposing the points of control, designing for transparency, and making space for alternatives [277].

Lessons from Internet Centralization. The history of Internet shows the case where tussle is poorly balanced and ignored. The Internet started as a distributed system with a small number of stakeholders. But over time, control over different layers—routing, identity, applications, cloud—shifted to a few powerful companies. Google now handles nearly 90% of global search traffic [217]. Meta controls identity at global scale and deploying extensive third-party tracking infrastructure [235]. Amazon controls over 30% of cloud infrastructure, hosting a substantial portion of the modern web [255]. These outcomes emerged from early Internet design choices that failed to account for long-term shifts in power and control [277].

6.2 The Tussle Space of AI

Now, I draw the parallel between today’s AI development and that of the early days of the Internet.

Faster Power Concentration in AI. Today’s generative AI development closely parallels the early days of the Internet. But AI has a more compressed timeline, where current AI systems have developed under conditions of existing company dominance. Big technology companies not only lead in model performance but also in disproportionate control over training data, compute resources, and distribution channels. Decisions around dataset construction, API setup, fine-tuning access, and deployment configurations are shaped with minimal input outside. The result is an ecosystem that were designed—and often optimized—for companies’ short-term profit. Unlike the Internet’s gradual centralization, AI has accelerated toward consolidation from the start.

Impact of the Imbalanced Tussle. The impact of this imbalance is already visible. Platform providers have revised terms of service to assert training rights over user-generated con-

tent [156, 165]. Governments and courts are expanding the scope of fair use to cover large-scale AI training [82]. At the same time, internal concerns—particularly around model misuse, consent, and safety—have been met with institutional pushback, e.g., layoffs of ethics and safety teams [141]. These developments represent a systemic suppression of conflict through both technical and legal control. As Clark observed, when systems do not explicitly account for tussle, power asymmetries take hold. In today’s AI ecosystem, the lack of ways for weaker stakeholders to push back risks repeating what happened with the Internet—locking us into a system where power concentrates over time and meaningful opposition becomes increasingly impossible.

6.3 My Research: Balancing the AI Tussle

The tools presented in this dissertation—Fawkes, Glaze, and Nightshade—represent a progression of techniques designed to shift power within an imbalanced AI ecosystem. Fawkes and Glaze function as passive defensive systems. These tools mark an important first step, given the growing harms that AI poses to creative sector and personal identity. Nightshade **extends this further**. By embedding adversarial perturbations into data, it directly affects the training process—acting as a deterrent and a source of risk for companies that scrape and train models without consent. Nightshade introduces friction into the tussle between AI developers and data-producing copyright holders, enabling stakeholders to assert leverage and demand more balanced design choices.

Building the Playing Field for a Balanced Tussle. These systems are not foolproof. Their effectiveness depends on user behavior, adoption scale, and evolving threat models. Sophisticated adversaries can find ways to circumvent their protections. Yet their significance lies not in absolute guarantees, but in their downstream impact. These tools demonstrate that meaningful resistance is technically feasible, even under significant power asymmetries. By introducing friction and raising the cost and risk of unauthorized data use, they shift attacker incentives and open space for weaker stakeholders to push back.

Their influence also extends beyond the technical domain. With millions of downloads, inte-

gration into creative platforms, citations in policy discussions, and references in active cases, these tools have helped shape public discourse and regulatory attention on AI governance. My research offers a direction that treats AI design not as a technical challenge, but as an opportunity to embed balance and resistance into the dynamics of AI development.

6.4 Closing Thoughts

The search of a more balanced AI ecosystem remains a central challenge for the field. Systems deployed today will shape power distributions for the future. While a perfectly balanced system is unlikely, design choices can rapidly ossify and shape downstream dynamics, like the case of the Internet. This dissertation contributes a set of technical interventions that demonstrate how power asymmetries in AI can be shaped. Rather than preventing progress, these tools reconfigure leverage and expand the agency of weak stakeholders. The broader implication is that the role of research is not to suppress the tensions in AI ecosystems, but to design systems where power back and forth is possible, structured, and transparent. This principle extends beyond AI: in any consequential technology, balance must be actively engineered, not assumed.

REFERENCES

- [1] Amazon rekognition. <https://aws.amazon.com/rekognition>.
- [2] Azure face recognition. <https://azure.microsoft.com/en-us/services/cognitive-services/face>.
- [3] Clearview.ai. <https://clearview.ai>.
- [4] Midjourney user prompts; generated images (250k).
- [5] PubFig83: A resource for studying face recognition in personal photo collections. <http://vision.seas.harvard.edu/pubfig83/>.
- [6] <https://www.faceplusplus.com/face-searching/>. Face++ Face Searching API.
- [7] Using the IJG JPEG library: Advanced features. <http://apodeline.free.fr/DOC/libjpeg/libjpeg-3.html>.
- [8] Deepmind faces legal action over nhs data use, 2021. <https://www.bbc.com/news/technology-58761324>.
- [9] Create logos, videos, banners, mockups with AI in 2 minutes. designs.ai, 2023.
- [10] 3dlook. Virtual try-on for clothing: The future of fashion? 3dlook.ai, 2023.
- [11] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proc. of CCS*, pages 308–318, 2016.
- [12] Adobe Max conference, Oct. 2023.
- [13] Hojjat Aghakhani, Dongyu Meng, Yu-Xiang Wang, Christopher Kruegel, and Giovanni Vigna. Bullseye polytope: A scalable clean-label poisoning attack with improved transferability. In *Proc. of EuroS&P*, 2021.
- [14] AI Render. AI Render - Stable Diffusion in Blender. , 2022. <https://blendermarket.com/products/ai-render>.
- [15] Eman A Alasadi and Carlos R Baiz. Generative ai in education and research: Opportunities, concerns, and solutions. *Journal of Chemical Education*, 2023.
- [16] Sarah Andersen. The Alt-Right Manipulated My Comic. Then A.I. Claimed It., December 2022. NY Times.
- [17] Eugene Bagdasaryan and Vitaly Shmatikov. Blind backdoors in deep learning models. In *Proc. of USENIX Security*, 2021.

- [18] Eugene Bagdasaryan and Vitaly Shmatikov. Spinning language models: Risks of propaganda-as-a-service and countermeasures. In *Proc. of IEEE S&P*, 2022.
- [19] Andy Baio. Invasive Diffusion: How one unwilling illustrator found herself turned into an AI model, 2022. <http://waxy.org>.
- [20] Yogesh Balaji, Seungjun Nah, Xun Huang, Arash Vahdat, Jiaming Song, Karsten Kreis, Miika Aittala, Timo Aila, Samuli Laine, Bryan Catanzaro, et al. ediffi: Text-to-image diffusion models with an ensemble of expert denoisers. *arXiv:2211.01324*, 2022.
- [21] Battista Biggio, Blaine Nelson, and Pavel Laskov. Support vector machines under adversarial label noise. In *Proc. of ACML*, 2011.
- [22] Yochai Blau and Tomer Michaeli. The perception-distortion tradeoff. In *Proc. of CVPR*, 2018.
- [23] Francis Bond and Kyonghee Paik. A survey of wordnets and their licenses. In *Proc. of GWC*, 2012.
- [24] Lucas Bourtole et al. Machine unlearning. In *Proc. of IEEE S&P*, 2021.
- [25] Blake Brittain. AI-created images lose U.S. copyrights in test for new technology. Reuters, February 2023.
- [26] Qiong Cao, Li Shen, Weidi Xie, Omkar M. Parkhi, and Andrew Zisserman. Vggface2: A dataset for recognising faces across pose and age. In *Proc. of FG*, pages 67–74, 2018.
- [27] Nicholas Carlini et al. Extracting training data from large language models. In *Proc. of USENIX Security*, 2021.
- [28] Nicholas Carlini and David Wagner. Adversarial examples are not easily detected: Bypassing ten detection methods. In *Proc. of AISec*, pages 3–14, 2017.
- [29] Nicholas Carlini and David Wagner. Magnet and “efficient defenses against adversarial attacks” are not robust to adversarial examples. *arXiv preprint arXiv:1711.08478*, 2017.
- [30] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *Proc. of IEEE S&P*, 2017.
- [31] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *Proc. of IEEE S&P*, pages 39–57. IEEE, 2017.
- [32] Varun Chandrasekaran, Chuhan Gao, Brian Tang, Kassem Fawaz, Somesh Jha, and Suman Banerjee. Face-off: Adversarial face obfuscation. *arXiv:2003.08861*, 2020.
- [33] Soravit Changpinyo, Piyush Sharma, Nan Ding, and Radu Soricut. Conceptual 12M: Pushing web-scale image-text pre-training to recognize long-tail visual concepts. In *Proc. of CVPR*, 2021.

- [34] Bryant Chen et al. Detecting backdoor attacks on deep neural networks by activation clustering. *arXiv:1811.03728*, 2018.
- [35] Huili Chen, Cheng Fu, Jishen Zhao, and Farinaz Koushanfar. Deepinspect: A black-box trojan detection and mitigation framework for deep neural networks. In *Proc. of IJCAI*, 2019.
- [36] Jianbo Chen, Michael I Jordan, and Martin J Wainwright. Hopskipjumpattack: A query-efficient decision-based attack. In *Proc. of IEEE S&P*, pages 1277–1294. IEEE, 2020.
- [37] Mia Xu Chen, Benjamin N Lee, Gagan Bansal, Yuan Cao, Shuyuan Zhang, Justin Lu, Jackie Tsay, Yinan Wang, Andrew M Dai, Zhifeng Chen, et al. Gmail smart compose: Real-time assisted writing. In *Proc. of KDD*, pages 2287–2295, 2019.
- [38] Weixin Chen, Dawn Song, and Bo Li. Trojdiff: Trojan attacks on diffusion models with diverse targets. In *Proc. of CVPR*, 2023.
- [39] Wenhui Chen, Hexiang Hu, Chitwan Saharia, and William W Cohen. Re-imagen: Retrieval-augmented text-to-image generator. *arXiv:2209.14491*, 2022.
- [40] Xiaoyi Chen et al. Badnl: Backdoor attacks against NLP models with semantic-preserving improvements. In *Proc. of ACSAC*, 2021.
- [41] Yuxin Chen, Huiying Li, Shan-Yuan Teng, Steven Nagels, Zhijing Li, Pedro Lopes, Ben Y. Zhao, and Haitao Zheng. Wearable microphone jamming. In *Proc. of ACM CHI*, April 2020.
- [42] Alex Cheney. Hundreds of Animation Guild members join WGA writers on picket line for first time. ABC7 News, May 2023.
- [43] Valeriia Cherepanova et al. Lowkey: Leveraging adversarial attacks to protect social media users from facial recognition. *arXiv:2101.07922*, 2021.
- [44] Sheng-Yen Chou, Pin-Yu Chen, and Tsung-Yi Ho. How to backdoor diffusion models? In *Proc. of CVPR*, 2023.
- [45] Civitai. What the heck is Civitai?, 2022. <https://civitai.com/content/guides/what-is-civitai>.
- [46] David D. Clark. Tussle in cyberspace: defining tomorrow’s internet. *Proc. of SIGCOMM*, 2002.
- [47] Laurie Clarke. When AI can make art – what does it mean for creativity?, 2022.
- [48] Common Crawl. Common crawl — open repository of web crawl data, 2025.
- [49] Jim Cross. Valley attorney: Facebook facial recognition carries identity theft risk. *KTAR News*, September 2019.

- [50] Thomas Davenport, Abhijit Guha, Dhruv Grewal, and Timna Bressgott. How artificial intelligence will change the future of marketing. *Journal of the Academy of Marketing Science*, 2020.
- [51] Emilia David. Now you can block OpenAI’s web crawler. *The Verge*, August 2023.
- [52] Boris Dayma. DALLE Mega - Training Journal, 2022. <https://wandb.ai/dalle-mini/dalle-mini/reports/DALL-E-Mega-Training-Journal--VmlldzoxODMxMDI2>.
- [53] Boris Dayma, Suraj Patil, Pedro Cuenca, Khalid Saifullah, Tanishq Abraham, Phuc Le Khac, Luke Melas, and Ritobrata Ghosh. Dalle mini, 2021.
- [54] Ambra Demontis, Marco Melis, Maura Pintor, Matthew Jagielski, Battista Biggio, Alina Oprea, Cristina Nita-Rotaru, and Fabio Roli. Why do adversarial attacks transfer? explaining transferability of evasion and poisoning attacks. In *Proc. of USENIX Security*, pages 321–338, 2019.
- [55] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Proc. of CVPR*, pages 248–255. IEEE, 2009.
- [56] Jesus Diaz. Instagram is training ai on your data. it’s nearly impossible to opt out. *Fast Company*, 2024.
- [57] Ming Ding et al. Cogview: Mastering text-to-image generation via transformers. *Proc. of NeurIPS*, 2021.
- [58] Pranav Dixit. Meet the three artists behind a landmark lawsuit against AI art generators. *BuzzFeedNews*, January 2023.
- [59] Yinpeng Dong, Tianyu Pang, Hang Su, and Jun Zhu. Evading defenses to transferable adversarial examples by translation-invariant attacks. In *Proc. of CVPR*, 2019.
- [60] Mathew Dryhurst. AI art and the problem of consent. *ArtReview*, January 2023.
- [61] Cynthia Dwork. Differential privacy: A survey of results. In *Proc. of International Conference on Theory and Applications of Models of Computation*. Springer, 2008.
- [62] Benj Edwards. Artists stage mass protest against AI-generated artwork on artstation. *Ars Technica*, December 2022.
- [63] Eray Eliaçık. Does ArtStation become PromptStation?, 2022. <https://dataconomy.com/2022/12/no-to-ai-generated-images-artstation/>.
- [64] Ivan Evtimov, Pascal Sturmfels, and Tadayoshi Kohno. Foggysight: A scheme for facial lookup privacy. *arXiv:2012.08588*, 2020.
- [65] Kevin Eykholt et al. Robust physical-world attacks on deep learning visual classification. In *Proc. of CVPR*, 2018.

- [66] Reuben Feinman, Ryan R Curtin, Saurabh Shintre, and Andrew B Gardner. Detecting adversarial samples from artifacts. *arXiv:1703.00410*, 2017.
- [67] Christiane Fellbaum. *WordNet and wordnets. Encyclopedia of Language and Linguistics*. Oxford: Elsevier, 2005.
- [68] Hayden Field. Zoom can now train its a.i. using some customer data, according to updated terms. *CNBC*, 2023.
- [69] M Figueroa-Torres. The drawbacks of international law in governing artificial intelligence. *Expert Analysis*, 2025.
- [70] Elizabeth Flux. What does the rise of ai mean for the future of art? *Sydney Morning Herald*, Dec. 2022.
- [71] Vicki Fox. AI art & the ethical concerns of artists. *Beautiful Bizarre*, March 2023.
- [72] Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. Model inversion attacks that exploit confidence information and basic countermeasures. In *Proc. of CCS*, 2015.
- [73] Rinon Gal et al. An image is worth one word: Personalizing text-to-image generation using textual inversion. *arXiv:2208.01618*, 2022.
- [74] Yansong Gao, Change Xu, Derui Wang, Shiping Chen, Damith C Ranasinghe, and Surya Nepal. Strip: A defence against trojan attacks on deep neural networks. In *Proc. of ACSAC*, pages 113–125, 2019.
- [75] Jonas Geiping et al. What doesn’t kill you makes you robust (er): Adversarial training against poisons and backdoors. *arXiv:2102.13624*, 2021.
- [76] Micah Goldblum et al. Dataset security for machine learning: Data poisoning, backdoor attacks, and defenses. *IEEE TPAMI*, 2022.
- [77] George Gondim-Ribeiro, Pedro Tabacof, and Eduardo Valle. Adversarial attacks on variational autoencoders. *arXiv:1806.04646*, 2018.
- [78] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. <http://www.deeplearningbook.org>.
- [79] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv:1412.6572*, 2014.
- [80] Matt Growcoat. Midjourney founder admits to using a ‘hundred million’ images without consent. *PetaPixel*, Dec 2022.
- [81] Tianyu Gu, Brendan Dolan-Gavitt, and Siddharth Garg. Badnets: Identifying vulnerabilities in the machine learning model supply chain. In *Proc. of MLCS Workshop*, 2017.

- [82] The Guardian. Meta wins ai copyright lawsuit as us judge rules against authors, 2025. Accessed: 2025-06-26.
- [83] Authors Guild. Meta’s massive ai training book heist: What authors need to know. *authors-guild.org*, 2025.
- [84] Neal Gupta, W Ronny Huang, Liam Fowl, Chen Zhu, Soheil Feizi, Tom Goldstein, and John Dickerson. Strong baseline defenses against clean-label poisoning attacks. 2019.
- [85] Anna Yoo Jeong Ha, Josephine Passananti, Ronik Bhaskar, Shawn Shan, Reid Southen, Haitao Zheng, and Ben Y. Zhao. Organic or diffused: Can we distinguish human art from ai-generated images? In *Proc. of CCS*, October 2024.
- [86] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proc. of CVPR*, 2016.
- [87] Xinlei He, Savvas Zannettou, Yun Shen, and Yang Zhang. You only prompt once: On the capabilities of prompt learning on large language models to tackle toxic content. *arXiv:2308.05596*, 2023.
- [88] Christian Heidorn. Mind-Boggling Midjourney Statistics in 2022, 2022. <https://tokenizedhq.com/midjourney-statistics/>.
- [89] Melissa Heikkila. This artist is dominating ai-generated art. and he’s not happy about it. MIT Technology Review, Sept 2022.
- [90] Rebecca Heilweil. The world’s scariest facial recognition company, explained. *Vox.com*.
- [91] Amir Hertz et al. Prompt-to-prompt image editing with cross attention control. *arXiv:2208.01626*, 2022.
- [92] Martin Heusel et al. GANs trained by a two time-scale update rule converge to a local Nash equilibrium. In *Proc. of NeurIPS*, 2017.
- [93] Kashmir Hill and Aaron Krolik. How photos of your kids are powering surveillance technology. *The New York Times*, October 11 2019. <https://www.nytimes.com/interactive/2019/10/11/technology/flickr-facial-recognition.html>.
- [94] Karen K. Ho. Pirated-books database libgen, which meta used to train ai, includes books by artists, architects, galleries and museums. *ARTnews*, 2025.
- [95] Andrew Hoare. Digital Illustration Styles, 2021. <https://www.theillustrators.com.au/digital-illustration-styles>.
- [96] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proc. of CVPR*, 2017.

- [97] Andrew Ilyas, Logan Engstrom, Anish Athalye, and Jessy Lin. Black-box adversarial attacks with limited queries and information. In *Proc. of ICML*, 2018.
- [98] Nadiya Ivanenko. Midjourney v4: an incredible new version of the AI image generator, 2022.
- [99] Steve TK Jan, Joseph Messou, Yen-Chen Lin, Jia-Bin Huang, and Gang Wang. Connecting the digital and physical world: Improving the robustness of adversarial attacks. In *Proc. of AAAI*, 2019.
- [100] Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. In *Proc. of ICML*, 2021.
- [101] Jinyuan Jia, Xiaoyu Cao, and Neil Zhenqiang Gong. Intrinsic certified robustness of bagging against data poisoning attacks. In *Proc. of AAAI*, 2021.
- [102] Joseph Saveri Law Firm LLP. Class Action Filed Against Stability AI, Midjourney, and DeviantArt for DMCA Violations, Right of Publicity Violations, Unlawful Competition, Breach of TOS, 2023. <https://cybernews.com/news/artists-unite-in-legal-battle-against-ai/>.
- [103] Brian Judge, Mark Nitzberg, and Stuart Russell. When code isn’t law: rethinking regulation for artificial intelligence. *Policy and Society*, 2025.
- [104] M. Imtiaz Karamat. X to use and share user data for ai training. *Deeth Williams Wall*, 2024.
- [105] Tero Karras, Miika Aittala, Janne Hellsten, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Training generative adversarial networks with limited data. *Proc. of NeurIPS*, 33:12104–12114, 2020.
- [106] Bahjat Kawar, Shiran Zada, Oran Lang, Omer Tov, Huiwen Chang, Tali Dekel, Inbar Mosseri, and Michal Irani. Imagic: Text-based real image editing with diffusion models. *arXiv:2210.09276*, 2022.
- [107] Yan Ke, Rahul Sukthankar, Larry Huston, Yan Ke, and Rahul Sukthankar. Efficient near-duplicate detection and sub-image retrieval. In *Proc. of MM*, volume 4, 2004.
- [108] Taein Kim, Karstan Bock, Claire Luo, Amanda Liswood, and Emily Wenger. Scrapers selectively respect robots.txt directives: evidence from a large-scale empirical study. *arXiv preprint arXiv:2505.21733*, 2025.
- [109] Stepan Komkov and Aleksandr Petiushko. Advhat: Real-world adversarial attack on arcface face id system. In *Proc. of ICPR*. IEEE, 2021.
- [110] Jernej Kos, Ian Fischer, and Dawn Song. Adversarial examples for generative models. In *Proc. of SPW*, pages 36–42. IEEE, 2018.

- [111] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial examples in the physical world. *arXiv:1607.02533*, 2016.
- [112] Tuomas Kynkäänniemi, Tero Karras, Miika Aittala, Timo Aila, and Jaakko Lehtinen. The role of imagenet classes in fr\`echet inception distance. *arXiv:2203.06026*, 2022.
- [113] Cassidy Laidlaw, Sahil Singla, and Soheil Feizi. Perceptual adversarial robustness: Defense against unseen threat models. *arXiv:2006.12655*, 2020.
- [114] Yann LeCun, Yoshua Bengio, et al. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10):1995, 1995.
- [115] Changsun Lee, Joonhyeok Jang, and Jeongwon Lee. Personalizing text-to-image generation with visual prompts using BLIP-2. In *Proc. of ICML*, 2023.
- [116] Nicole Lee. Having multiple online identities is more normal than you think. Engadget, March 2016. <https://www.engadget.com/2016/03/04/multiple-online-identities>.
- [117] Gloria Levine. A New Stable Diffusion Plug-In For GIMP & Krita, 2022. <https://80.lv/articles/a-new-stable-diffusion-plugin-for-gimp-krita/>.
- [118] Huiying Li, Shawn Shan, Emily Wenger, Jiayun Zhang, Haitao Zheng, and Ben Y Zhao. Blacklight: Scalable defense for neural networks against {Query-Based}{Black-Box} attacks. In *31st USENIX Security Symposium (USENIX Security 22)*, pages 2117–2134, 2022.
- [119] Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. BLIP: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *Proc. of ICML*, 2022.
- [120] Kun-Lun Li, Hou-Kuan Huang, Sheng-Feng Tian, and Wei Xu. Improving one-class svm for anomaly detection. In *Proc. of IEEE Cat*, volume 5. IEEE, 2003.
- [121] Tao Li and Lei Lin. Anonymousnet: Natural face de-identification with measurable privacy. In *Proc. of CVPR*, 2019.
- [122] Wenbo Li et al. Object-driven text-to-image synthesis via adversarial training. In *Proc. of CVPR*, 2019.
- [123] Yanghao Li, Haoqi Fan, Ronghang Hu, Christoph Feichtenhofer, and Kaiming He. Scaling language-image pre-training via masking. *arXiv:2212.00794*, 2022.
- [124] Yuezun Li, Xin Yang, Baoyuan Wu, and Siwei Lyu. Hiding faces in plain sight: Disrupting AI face synthesis with adversarial perturbations. *arXiv:1906.09288*, 2019.
- [125] Chumeng Liang et al. Adversarial example does good: Preventing painting imitation from diffusion models via adversarial examples. In *Proc. of ICML*, 2023.

- [126] Tsung-Yi Lin et al. Microsoft coco: Common objects in context. In *Proc. of ECCV*, 2014.
- [127] Enze Liu, Elisa Luo, Shawn Shan, Geoffrey M Voelker, Ben Y Zhao, and Stefan Savage. Somesite i used to crawl: Awareness, agency and efficacy in protecting content creators from ai crawlers. *arXiv preprint arXiv:2411.15091*, 2024.
- [128] GLORIA LIU. The World’s Smartest Artificial Intelligence Just Made Its First Magazine Cover., 2022. <https://www.cosmopolitan.com/lifestyle/a40314356/dall-e-2-artificial-intelligence-cover/>.
- [129] Hongbin Liu, Wenjie Qu, Jinyuan Jia, and Neil Zhenqiang Gong. Pre-trained encoders in self-supervised learning improve secure and privacy-preserving supervised learning. *arXiv:2212.03334*, 2022.
- [130] Xuankai Liu, Fengting Li, Bihan Wen, and Qi Li. Removing backdoor-based watermarks in neural networks with limited data. In *Proc. of ICPR*, 2021.
- [131] Yanpei Liu, Xinyun Chen, Chang Liu, and Dawn Song. Delving into transferable adversarial examples and black-box attacks. In *Proc. of ICLR*, 2016.
- [132] Yingqi Liu et al. Trojaning attack on neural networks. In *Proc. of NDSS*, 2018.
- [133] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Large-scale celebfaces attributes (celeba) dataset. *Retrieved August, 15(2018):11*, 2018.
- [134] Yiwei Lu, Gautam Kamath, and Yaoliang Yu. Indiscriminate data poisoning attacks on neural networks. *arXiv:2204.09092*, 2022.
- [135] Ziyang Luo, Yadong Xi, Rongsheng Zhang, and Jing Ma. Vc-gpt: Visual conditioned gpt for end-to-end generative vision-and-language pre-training. *arXiv:2201.12723*, 2022.
- [136] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv:1706.06083*, 2017.
- [137] Elman Mansimov, Emilio Parisotto, Jimmy Lei Ba, and Ruslan Salakhutdinov. Generating images from captions with attention. *arXiv:1511.02793*, 2015.
- [138] Angelica Mari. Brazilian retailer quizzed over facial recognition tech. *ZDNet*, March 2019.
- [139] David Marx. <https://twitter.com/DigThatData/status/1636386617392009224>.
- [140] Chenlin Meng, Ruiqi Gao, Diederik P Kingma, Stefano Ermon, Jonathan Ho, and Tim Salimans. On distillation of guided diffusion models. *arXiv:2210.03142*, 2022.
- [141] Cade Metz. Ai companies lay off ethics teams amid rapid expansion, 2024. Accessed: 2025-06-26.

- [142] Cade Metz and Keith Collins. How an A.I. ‘cat-and-mouse game’ generates believable fake photos. *The New York Times*, January 2018.
- [143] Midjourney. Community Showcase, 2022. <https://www.midjourney.com/showcase/top/>.
- [144] Christopher Morris. 7 best AI website builders in 2023 (for fast web design). *elegant-themes.com*, Sept 2023.
- [145] Paul Mozur. Inside china’s dystopian dreams: A.i., shame and lots of cameras. *The New York Times*, 2018.
- [146] Brendan Paul Murphy. Is Lensa AI Stealing From Human Art? An Expert Explains the Controversy, 2022.
- [147] Aaron Nech and Ira Kemelmacher-Shlizerman. Level playing field for million scale face recognition. In *Proc. of CVPR*, 2017.
- [148] Seth Neel, Aaron Roth, and Saeed Sharifi-Malvajerdi. Descent-to-delete: Gradient-based methods for machine unlearning. In *Proc. of ALT*, 2021.
- [149] Elaine M Newton, Latanya Sweeney, and Bradley Malin. Preserving privacy by de-identifying face images. *IEEE transactions on Knowledge and Data Engineering*, 17(2):232–243, 2005.
- [150] Hong-Wei Ng and Stefan Winkler. A data-driven approach to cleaning large face datasets. In *Proc. of ICIP*, 2014.
- [151] Kaitlyn Nguyen. Ai is causing student artists to rethink their creative career plans. KQED Arts, April 2023. <https://www.kqed.org/arts/13928253/ai-art-artificial-intelligence-student-artists-midjourney>.
- [152] Thao Nguyen, Samir Yitzhak Gadre, Gabriel Ilharco, Sewoong Oh, and Ludwig Schmidt. Improving multimodal datasets with image captioning. *arXiv:2307.10350*, 2023.
- [153] Jorge Nocedal and Stephen Wright. *Numerical optimization*. operations research and financial engineering. Springer, 2006.
- [154] NovelAI. NovelAI changelog, 2022. <https://novelai.net/updates>.
- [155] Kate O’Flaherty. Facial recognition at u.s. airports. should you be concerned? *Forbes*, March 2019. <https://www.forbes.com/sites/kateoflahertyuk/2019/03/11/facial-recognition-to-be-deployed-at-top-20-us-airports-should-you-be-concerned/>.
- [156] OpenAI. Openai updates terms of use and privacy policy, 2023. Accessed: 2025-06-26.

- [157] Nicolas Papernot, Patrick McDaniel, and Ian Goodfellow. Transferability in machine learning: From phenomena to black-box attacks using adversarial samples. *arXiv:1605.07277*, 2016.
- [158] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z. Berkay Celik, and Ananthram Swami. Practical black-box attacks against machine learning. In *Proc. of AsiaCCS*, 2017.
- [159] Dong Huk Park, Samaneh Azadi, Xihui Liu, Trevor Darrell, and Anna Rohrbach. Benchmark for compositional text-to-image synthesis. In *Proc. of NeurIPS*, 2021.
- [160] Hyun Jun Park. The rise of generative artificial intelligence and the threat of fake news and disinformation online: Perspectives from sexual medicine. *Investigative and Clinical Urology*, 2024.
- [161] Andrea Paudice, Luis Muñoz-González, Andras Gyorgy, and Emil C Lupu. Detection of adversarial training examples in poisoning attacks through anomaly detection. *arXiv:1802.03041*, 2018.
- [162] Katie Paul and Anna Tong. Inside big tech’s underground race to buy ai training data. *reuters*, 2023.
- [163] Miranda Perez. AI Art Generator Cupixel Rakes in 5M From Craft Store JOANN, 2022. <https://www.builtinboston.com/2022/08/10/cupixel-raises-5m>.
- [164] Neehar Peri, Neal Gupta, W Ronny Huang, Liam Fowl, Chen Zhu, Soheil Feizi, Tom Goldstein, and John P Dickerson. Deep k-nn defense against clean-label data poisoning attacks. In *Proc. of ECCV*, pages 55–70. Springer, 2020.
- [165] Meta Platforms. Meta’s revised terms of service expand ai training rights, 2023. Accessed: 2025-06-26.
- [166] Luke Plunkett. AI creating ‘art’ is an ethical and copyright nightmare. *Kotaku*, August 2022.
- [167] Dustin Podell et al. SDXL: Improving latent diffusion models for high-resolution image synthesis. *arXiv:2307.01952*, 2023.
- [168] NIK POPLI. He Used AI to Publish a Children’s Book in a Weekend. Artists Are Not Happy About It., 2022. <https://time.com/6240569/ai-childrens-book-alice-and-sparkle-artists-unhappy/>.
- [169] Ximing Qiao, Yukun Yang, and Hai Li. Defending neural backdoors via generative distribution modeling. *Proc. of NeurIPS*, 2019.
- [170] Alec Radford et al. Learning transferable visual models from natural language supervision. In *Proc. of ICML*, 2021.

- [171] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv:1511.06434*, 2015.
- [172] Evani Radiya-Dixit, Sanghyun Hong, Nicholas Carlini, and Florian Tramèr. Data poisoning won’t save you from facial recognition. *arXiv:2106.14851*, 2021.
- [173] Aditya Ramesh et al. Zero-shot text-to-image generation. In *Proc. of ICML*, 2021.
- [174] Aditya Ramesh et al. Hierarchical text-conditional image generation with clip latents. *arXiv:2204.06125*, 2022.
- [175] Barbara Rasin. Opt-Out Approaches to AI Training: A False Compromise. *Berkeley Technology Law Journal*, 2025.
- [176] Lilian Rincon. Virtually try on clothes with a new AI shopping feature. Google Blog, June 2023.
- [177] Herbert Robbins and Sutton Monro. A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407, 1951.
- [178] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proc. of CVPR*, 2022.
- [179] Jérôme Rony, Eric Granger, Marco Pedersoli, and Ismail Ben Ayed. Augmented lagrangian adversarial attacks. In *Proc. of ICCV*, 2021.
- [180] Kevin Roose. An A.I.-Generated Picture Won an Art Prize. Artists Aren’t Happy., 2022. <https://www.nytimes.com/2022/09/02/technology/ai-artificial-intelligence-artists.html>.
- [181] Nataniel Ruiz et al. Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation. In *Proc. of CVPR*, 2023.
- [182] Sara Sabour, Yanshuai Cao, Fartash Faghri, and David J Fleet. Adversarial manipulation of deep representations. *arXiv:1511.05122*, 2015.
- [183] Aniruddha Saha, Akshayvarun Subramanya, and Hamed Pirsiavash. Hidden trigger back-door attacks. In *Proc. of AAAI*, 2020.
- [184] Chitwan Saharia, William Chan, Huiwen Chang, Chris Lee, Jonathan Ho, Tim Salimans, David Fleet, and Mohammad Norouzi. Palette: Image-to-image diffusion models. In *Proc. of ACM SIGGRAPH*, 2022.
- [185] Babak Saleh and Ahmed Elgammal. Large-scale classification of fine-art paintings: Learning the right metric on the right feature. *arXiv:1505.00855*, 2015.
- [186] Rob Salkowitz. AI is coming for commercial art jobs. can it be stopped? Forbes, Sept 2022.

- [187] Hadi Salman, Alaa Khaddaj, Guillaume Leclerc, Andrew Ilyas, and Aleksander Madry. Raising the cost of malicious AI-powered image editing. *arXiv:2302.06588*, 2022.
- [188] Matleena Salminen. How to use AI image generator to make custom images for your site in 2023. *hostinger.com*, Sept 2023.
- [189] Adam Satariano. Police use of facial recognition is accepted by British court. *The New York Times*, September 2019. <https://www.nytimes.com/2019/09/04/business/facial-recognition-uk-court.html>.
- [190] Scenario.gg. AI-generated game assets, 2022. <https://www.scenario.gg/>.
- [191] Christoph Schuhmann. LAION-aesthetics. LAION.AI, August 2022.
- [192] Christoph Schuhmann et al. LAION-400M: Open dataset of clip-filtered 400 million image-text pairs. *arXiv:2111.02114*, 2021.
- [193] Christoph Schuhmann et al. LAION-5B: An open large-scale dataset for training next generation image-text models. *arXiv:2210.08402*, 2022.
- [194] Roei Schuster, Congzheng Song, Eran Tromer, and Vitaly Shmatikov. You autocomplete me: Poisoning vulnerabilities in neural code completion. In *Proc. of USENIX Security*, 2021.
- [195] Avi Schwarzschild et al. Just how toxic is data poisoning? a unified benchmark for backdoor and data poisoning attacks. In *Proc. of ICML*, pages 9389–9398. PMLR, 2021.
- [196] Giorgio Severi, Jim Meyer, Scott Coull, and Alina Oprea. Explanation-guided backdoor poisoning attacks against malware classifiers. In *Proc. of USENIX Security*, 2021.
- [197] Ali Shafahi et al. Poison frogs! targeted clean-label poisoning attacks on neural networks. *arXiv:1804.00792*, 2018.
- [198] Shawn Shan, Arjun Nitin Bhagoji, Haitao Zheng, and Ben Y. Zhao. Poison forensics: Trace-back of data poisoning attacks in neural networks. In *Proc. of USENIX Security*, 2022.
- [199] Shawn Shan, Jenna Cryan, Emily Wenger, Haitao Zheng, Rana Hanocka, and Ben Y. Zhao. Glaze: Protecting artists from style mimicry by text-to-image models. In *Proc. of USENIX Security*, 2023.
- [200] Shawn Shan, Wenxin Ding, Emily Wenger, Haitao Zheng, and Ben Y. Zhao. Post-breach recovery: Protection against white-box adversarial examples for leaked dnn models. In *Proc. of CCS*, 2022.
- [201] Shawn Shan, Emily Wenger, Bolun Wang, Bo Li, Haitao Zheng, and Ben Y. Zhao. Gotta catch’em all: Using honeypots to catch adversarial attacks on neural networks. In *Proc. of CCS*, 2020.

- [202] Shawn Shan, Emily Wenger, Jiayun Zhang, Huiying Li, Haitao Zheng, and Ben Y. Zhao. Fawkes: Protecting privacy against unauthorized deep learning models. In *Proc. of USENIX Security*, 2020.
- [203] Mahmood Sharif, Sruti Bhagavatula, Lujo Bauer, and Michael K Reiter. Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition. In *Proc. of CCS*, 2016.
- [204] Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. Conceptual captions: A cleaned, hypernamed, image alt-text dataset for automatic image captioning. In *Proc. of ACL*, 2018.
- [205] Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. Conceptual captions: A cleaned, hypernamed, image alt-text dataset for automatic image captioning. In *Proc. of ACL*, 2018.
- [206] Juncheng Shen, Xiaolei Zhu, and De Ma. Tensorclog: An imperceptible poisoning attack on deep neural network applications. *IEEE Access*, 7:41498–41506, 2019.
- [207] Shiqi Shen, Shruti Tople, and Prateek Saxena. Auror: Defending against poisoning attacks in collaborative deep learning systems. In *Proc. of ACSAC*, 2016.
- [208] Mohamed R Shoaib, Zefan Wang, Milad Taleby Ahvanooey, and Jun Zhao. Deepfakes, misinformation, and disinformation in the era of frontier ai, generative ai, and large ai models. In *Proc. of ICCA*, pages 1–7. IEEE, 2023.
- [209] Ilia Shumailov et al. The curse of recursion: Training on generated data makes models forget. *arxiv:2305.17493*, 2023.
- [210] Ilia Shumailov, Zakhar Shumaylov, Yiren Zhao, Nicolas Papernot, Ross Anderson, and Yarin Gal. Ai models collapse when trained on recursively generated data. *Nature*, 2024.
- [211] Maya Shwayder. Clearview AI’s facial-recognition app is a nightmare for stalking victims. *Digital Trends*, January 2020.
- [212] Olivia Solon. Facial recognition’s ‘dirty little secret’: Millions of online photos scraped without consent. *NBC News*, 2019.
- [213] Congzheng Song, Thomas Ristenpart, and Vitaly Shmatikov. Machine learning models that remember too much. In *Proc. of CCS*, 2017.
- [214] Stability AI. Stable Diffusion v2.1 and DreamStudio Updates 7-Dec 22, 2022. <https://stability.ai/blog/stablediffusion2-1-release7-dec-2022>.
- [215] 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>.

- [216] StabilityAI. Stable Diffusion v1-4 Model Card, 2022. <https://huggingface.co/CompVis/stable-diffusion-v1-4>.
- [217] StatCounter Global Stats. Search engine market share worldwide, 2024. <https://gs.statcounter.com/search-engine-market-share>.
- [218] Jacob Steinhardt, Pang Wei W Koh, and Percy S Liang. Certified defenses for data poisoning attacks. In *Proc. of NeurIPS*, 2017.
- [219] Octavian Suci, Radu Mărginean, Yiğitcan Kaya, Hal Daumé III, and Tudor Dumitraş. When does machine learning fail? generalized transferability for evasion and poisoning attacks. In *Proc. of USENIX Security*, 2018.
- [220] Qianru Sun, Liqian Ma, Seong Joon Oh, Luc Van Gool, Bernt Schiele, and Mario Fritz. Natural and effective obfuscation by head inpainting. In *Proc. of CVPR*, 2018.
- [221] Qianru Sun, Ayush Tewari, Weipeng Xu, Mario Fritz, Christian Theobalt, and Bernt Schiele. A hybrid model for identity obfuscation by face replacement. In *Proc. of ECCV*, 2018.
- [222] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *Proc. of AAAI*, 2017.
- [223] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *Proc. of ICLR*, 2014.
- [224] Pedro Tabacof, Julia Tavares, and Eduardo Valle. Adversarial images for variational autoencoders. *arXiv:1612.00155*, 2016.
- [225] Noriko Takemura, Yasushi Makihara, Daigo Muramatsu, Tomio Echigo, and Yasushi Yagi. Multi-view large population gait dataset and its performance evaluation for cross-view gait recognition. *IPSN Transactions on Computer Vision and Applications*, 10(1):1–14, 2018.
- [226] Ming Tao et al. DF-GAN: A simple and effective baseline for text-to-image synthesis. In *Proc. of CVPR*, 2022.
- [227] Bart Thomee, David A Shamma, Gerald Friedland, Benjamin Elizalde, Karl Ni, Douglas Poland, Damian Borth, and Li-Jia Li. Yfcc100m: The new data in multimedia research. *Communications of the ACM*, 59(2):64–73, 2016.
- [228] Simen Thys, Wiebe Van Ranst, and Toon Goedemé. Fooling automated surveillance cameras: adversarial patches to attack person detection. In *Proc. of CVPR (workshop)*, 2019.
- [229] Brandon Tran, Jerry Li, and Aleksander Madry. Spectral signatures in backdoor attacks. *arXiv:1811.00636*, 2018.

- [230] Tony Ho Tran. 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>.
- [231] Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. Robustness may be at odds with accuracy. *arXiv preprint arXiv:1805.12152*, 2018.
- [232] Alexander Turner, Dimitris Tsipras, and Aleksander Madry. Clean-label backdoor attacks. 2018.
- [233] Paul Upchurch, Jacob Gardner, Geoff Pleiss, Robert Pless, Noah Snavely, Kavita Bala, and Kilian Weinberger. Deep feature interpolation for image content changes. In *Proc. of CVPR*, 2017.
- [234] Arash Vahdat, Karsten Kreis, and Jan Kautz. Score-based generative modeling in latent space. *Proc. of NeurIPS*, 2021.
- [235] Giridhari Venkatadri, Athina Andreou, Alan Liu, Alan Mislove, and Oana Goga. Investigating sources of pii used in facebook’s targeted advertising. *arXiv preprint arXiv:1804.07348*, 2018.
- [236] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: a neural image caption generator. *arXiv:1411.4555*, 2014.
- [237] Gregory K Wallace. The JPEG still picture compression standard. *IEEE Transactions on Consumer Electronics*, 38(1), 1992.
- [238] Alexander Wan, Eric Wallace, Sheng Shen, and Dan Klein. Poisoning language models during instruction tuning. *arXiv:2305.00944*, 2023.
- [239] Binghui Wang, Xiaoyu Cao, Neil Zhenqiang Gong, et al. On certifying robustness against backdoor attacks via randomized smoothing. *arXiv:2002.11750*, 2020.
- [240] Bolun Wang, Yuanshun Yao, Shawn Shan, Huiying Li, Bimal Viswanath, Haitao Zheng, and Ben Y. Zhao. Neural cleanse: Identifying and mitigating backdoor attacks in neural networks. In *Proc. of IEEE S&P*, 2019.
- [241] Bolun Wang, Yuanshun Yao, Bimal Viswanath, Haitao Zheng, and Ben Y. Zhao. With great training comes great vulnerability: Practical attacks against transfer learning. In *Proc. of USENIX Security*, 2018.
- [242] Feng Wang and Huaping Liu. Understanding the behaviour of contrastive loss. In *Proc. of CVPR*, 2021.
- [243] Gang Wang, Tristan Konolige, Christo Wilson, Xiao Wang, Haitao Zheng, and Ben Y Zhao. You are how you click: Clickstream analysis for sybil detection. In *Proc. of USENIX Security*, pages 241–256, 2013.

- [244] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Trans. on Image Processing*, 13(4):600–612, 2004.
- [245] Zhou Wang, Eero P Simoncelli, and Alan C. Bovik. Multiscale structural similarity for image quality assessment. In *Proc. of Asilomar Conference on Signals, Systems & Computers*, volume 2, pages 1398–1402. IEEE, 2003.
- [246] JESS WEATHERBED. ArtStation is hiding images protesting AI art on the platform, 2022. <https://www.theverge.com/2022/12/23/23523864/artstation-removing-anti-ai-protest-artwork-censorship>.
- [247] Kelsey Weekman. People Have Raised Serious Concerns About The AI Art App That’s All Over Your Instagram Feed, 2022. <https://www.buzzfeednews.com/article/kelseyweekman/ai-art-app-lensa-instagram-photo-trend-problems>.
- [248] Laura Weidinger, John Mellor, Maribeth Rauh, Conor Griffin, Jonathan Uesato, Po-Sen Huang, Myra Cheng, Mia Glaese, Borja Balle, Atoosa Kasirzadeh, et al. Ethical and social risks of harm from language models. *arXiv preprint arXiv:2112.04359*, 2021.
- [249] Yuxin Wen, Neel Jain, John Kirchenbauer, Micah Goldblum, Jonas Geiping, and Tom Goldstein. Hard prompts made easy: Gradient-based discrete optimization for prompt tuning and discovery. *arXiv:2302.03668*, 2023.
- [250] Emily Wenger, Josephine Passananti, Arjun Bhagoji, Yuanshun Yao, Haitao Zheng, and Ben Y. Zhao. Backdoor attacks against deep learning systems in the physical world. In *Proc. of CVPR*, 2021.
- [251] Emily Wenger, Shawn Shan, Haitao Zheng, and Ben Y. Zhao. SoK: Anti-facial recognition technology. In *Proc. of IEEE S&P*, 2023.
- [252] Tom White. Sampling generative networks. *arXiv:1609.04468*, 2016.
- [253] Kyle Wiggers. Scenario lands \$6M for its AI platform that generates game art assets, 2022.
- [254] Kyle Wiggers. Stability AI, the startup behind Stable Diffusion, raises 101M, 2022. <https://techcrunch.com/2022/10/17/stability-ai-the-startup-behind-stable-diffusion-raises-101m/>.
- [255] Wikipedia contributors. Amazon web services market share, 2024. https://en.wikipedia.org/wiki/Amazon_Web_Services.
- [256] Gilbert Wondracek, Thorsten Holz, Engin Kirda, and Christopher Kruegel. A practical attack to de-anonymize social network users. In *Proc. of IEEE S&P*, 2010.
- [257] Lei Wu et al. Understanding and enhancing the transferability of adversarial examples. *arXiv:1802.09707*, 2018.

- [258] Yifan Wu, Fan Yang, and Haibin Ling. Privacy-Protective-GAN for face de-identification. *arXiv:1806.08906*, 2018.
- [259] Zuxuan Wu, Ser-Nam Lim, Larry S Davis, and Tom Goldstein. Making an invisibility cloak: Real world adversarial attacks on object detectors. In *Proc. of ECCV*, 2020.
- [260] Chloe Xiang. AI is probably using your images and it’s not easy to opt out. Motherboard, Tech by Vice, Sept 2022.
- [261] Tao Xu et al. AttnGAN: Fine-grained text to image generation with attentional generative adversarial networks. In *Proc. of CVPR*, 2018.
- [262] Weilin Xu, David Evans, and Yanjun Qi. Feature squeezing: Detecting adversarial examples in deep neural networks. In *Proc. of NDSS*, 2018.
- [263] Chaofei Yang, Qing Wu, Hai Li, and Yiran Chen. Generative poisoning attack method against neural networks. *arXiv:1703.01340*, 2017.
- [264] Sam Yang. Why Artists are Fed Up with AI Art., 2022.
- [265] Zhi Yang, Christo Wilson, Xiao Wang, Tingting Gao, Ben Y Zhao, and Yafei Dai. Uncovers social network sybils in the wild. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 8(1):1–29, 2014.
- [266] Ziqi Yang, Jiyi Zhang, Ee-Chien Chang, and Zhenkai Liang. Neural network inversion in adversarial setting via background knowledge alignment. In *Proc. of CCS*, London, UK, November 2019.
- [267] Ziqing Yang et al. Data poisoning attacks against multimodal encoders. In *Proc. of ICML*, 2023.
- [268] Yuanshun Yao, Huiying Li, Haitao Zheng, and Ben Y. Zhao. Latent backdoor attacks on deep neural networks. In *Proc. of CCS*, 2019.
- [269] Dong Yi, Zhen Lei, Shengcai Liao, and Stan Z. Li. Learning face representation from scratch, 2014.
- [270] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In *Proc. of NeurIPS*, 2014.
- [271] Shengfang Zhai et al. Text-to-image diffusion models can be easily backdoored through multimodal data poisoning. *arXiv:2305.04175*, 2023.
- [272] Eric Zhang et al. Forget-me-not: Learning to forget in text-to-image diffusion models. *arXiv:2303.17591*, 2023.
- [273] Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, and Dimitris N Metaxas. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. In *Proc. of ICCV*, 2017.

- [274] Hanwei Zhang, Yannis Avrithis, Teddy Furon, and Laurent Amsaleg. Smooth adversarial examples. *EURASIP Journal on Information Security*, pages 1–12, 2020.
- [275] Jinghuai Zhang, Hongbin Liu, Jinyuan Jia, and Neil Zhenqiang Gong. Corruptencoder: Data poisoning based backdoor attacks to contrastive learning. *arXiv:2211.08229*, 2022.
- [276] Kai Zhang, Wangmeng Zuo, Yunjin Chen, Deyu Meng, and Lei Zhang. Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. *IEEE transactions on image processing*, 2017.
- [277] Lixia Zhang. Report of 2021 dinrg workshop on centralization in the internet. *ACM Computer Communication Review*, 2023.
- [278] Lvmin Zhang. Adversecleaner. <https://github.com/llyasviel/AdverseCleaner>.
- [279] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proc. of CVPR*, 2018.
- [280] Yukun Zhang. The influence of generative ai on content platforms: Supply, demand, and welfare impacts in two-sided markets. *arXiv preprint arXiv:2410.13101*, 2024.
- [281] Chen Zhu, W Ronny Huang, Hengduo Li, Gavin Taylor, Christoph Studer, and Tom Goldstein. Transferable clean-label poisoning attacks on deep neural nets. In *Proc. of ICML*, 2019.
- [282] Minfeng Zhu, Pingbo Pan, Wei Chen, and Yi Yang. Dm-gan: Dynamic memory generative adversarial networks for text-to-image synthesis. In *Proc. of CVPR*, 2019.