THE UNIVERSITY OF CHICAGO

RECLAIMING DATA AGENCY IN THE AGE OF UBIQUITOUS MACHINE LEARNING

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
EMILY JOY WENGER

CHICAGO, ILLINOIS
JUNE 2023

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

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

TABLE OF CONTENTS

LIST OF FIGURES viii

LIST OF TABLES xi

ABSTRACT xiii

1 WHY IS DATA AGENCY NECESSARY? 1
1.1 Existing Data Privacy Solutions 2
1.2 Our Solution: Data Agency 4

2 BACKGROUND AND RELATED WORK 7
2.1 Machine Learning (ML) Overview 7
2.1.1 Training Image Classifiers via Supervised Learning 7
2.1.2 Other ML Training Settings and Tasks 8
2.2 Attacks and Defenses for ML Image Classification Models 9
2.2.1 Adversarial Example Attacks and Defenses 10
2.2.2 Poisoning Attacks and Defenses 11

3 DATA AGENCY VIA DISRUPTION—*FAWKES: PROTECTING PRIVACY AGAINST UNAUTHORIZED DEEP LEARNING MODELS* 13
3.1 Introduction 13
3.2 Background and Related Work 15
3.2.1 Protecting Privacy via Evasion Attacks 15
3.2.2 Protecting Privacy via Poisoning Attacks 16
3.2.3 Other Related Work 17
3.3 Protecting Privacy via Cloaking 18
3.3.1 Assumptions and Threat Model 19
3.3.2 Overview and Intuition 20
3.3.3 Computing Cloak Perturbations 21
3.3.4 Cloaking Effectiveness & Transferability 23
3.4 The Fawkes Image Cloaking System 25
3.5 System Evaluation 26
3.5.1 Experiment Setup 27
3.5.2 User/Tracker Sharing a Feature Extractor 29
3.5.3 User/Tracker Using Different Feature Extractors 31
3.5.4 Tracker Models Trained from Scratch 32
3.6 Image Cloaking in the Wild 33
3.6.1 Experimental Setup 33
3.6.2 Real World Protection Performance 34
3.7 Trackers with Uncloaked Image Access 36
3.7.1 Impact of Uncloaked Images 36
3.7.2 Sybil Accounts 37

3.7.3 Efficacy of Sybil Images . . . 39
3.8 Countermeasures . . . 40
3.8.1 Cloak Disruption . . . 41
3.8.2 Cloak Detection . . . 43
3.9 Discussion and Conclusion . . . 44

4 DATA AGENCY VIA TRACING—*DATA ISOTOPES FOR DATA PROVENANCE IN DNNS* 46
4.1 Introduction . . . 46
4.2 Requirements and Prior Work . . . 49
4.2.1 Defining Requirements . . . 49
4.2.2 Existing Work on ML Data Provenance . . . 51
4.3 Data Isotopes for Data Provenance . . . 53
4.3.1 Provenance via Spurious Correlations . . . 53
4.3.2 Introducing Data Isotopes . . . 54
4.4 Data Isotopes Methodology . . . 56
4.4.1 Overview . . . 57
4.4.2 Isotope Creation . . . 58
4.4.3 Isotope Detection . . . 58
4.4.4 Advanced Isotope Scenarios . . . 61
4.5 Evaluating Data Isotopes . . . 62
4.5.1 Methodology . . . 62
4.5.2 Single isotope subset in $\mathcal{D}$ . . . 65
4.5.3 Multiple isotope subsets in $\mathcal{D}$ . . . 67
4.5.4 Scaling isotopes . . . 69
4.6 Physical Objects as Marks . . . 70
4.6.1 Methodology . . . 70
4.6.2 Results . . . 71
4.7 Isotopes in Real-World Settings . . . 72
4.7.1 Larger models . . . 72
4.7.2 ML-as-a-Service APIs . . . 73
4.7.3 Transfer learning . . . 73
4.7.4 Isotopes in facial recognition engines . . . 74
4.8 Robustness to Adaptive Countermeasures . . . 76
4.8.1 Spurious correlation detection . . . 76
4.8.2 Feature Inspection . . . 78
4.8.3 Adversarial augmentations . . . 79
4.8.4 Reducing Granularity of Outputs . . . 80
4.8.5 Targeted Fine-tuning . . . 82
4.8.6 Differential Privacy . . . 82
4.9 Limitations and Future Work . . . 83

5 DATA AGENCY VIA DIRECT ATTACK—*BACKDOOR ATTACKS AGAINST DEEP LEARNING SYSTEMS IN THE PHYSICAL WORLD* 84
5.1 Introduction 84
5.2 Related Work 87
5.3 Methodology 88
5.3.1 Our Physical Backdoor Dataset 88
5.3.2 Attack Implementation & Model Training 90
5.4 Experiment Overview 92
5.5 Effectiveness of Physical Backdoors 93
5.6 Why Do Earrings Fail as a Trigger? 96
5.7 Evaluating Weaker Attacks 97
5.8 Physical Triggers & False Positives 99
5.8.1 Measuring False Positives 99
5.8.2 Mitigating False Positives 100
5.8.3 Key Takeaways 101
5.9 Defending Against Physical Backdoors 101
5.9.1 Effectiveness of Existing Defenses 102
5.10 Conclusion 104

6 A FRAMEWORK FOR REASONING ABOUT DATA AGENCY AGAINST UNWANTED FACIAL RECOGNITION 105
6.1 Introduction 105
6.2 Facial Recognition: Terminology, Design Stages and Deployment 108
6.2.1 Modern FR Systems 109
6.2.2 Breaking FR into Operational Stages 110
6.2.3 FR Deployment and Data collection 112
6.3 Anti-Facial Recognition: Motivation and Threat Model 114
6.3.1 The Rise of AFR Tools 114
6.3.2 Threat Model of AFR 115
6.4 A Stage-Based AFR Framework 116
6.4.1 AFR Strategies per Stage 118
6.4.2 Taxonomy of Existing AFR Proposals 119
6.4.3 Roadmap of Our Analysis 120
6.5 Attacking Stage 1 to Disrupt Data Collection 120
6.5.1 Current Solutions 120
6.5.2 Discussion of Stage 1 Solutions 122
6.6 Attacking Stage 2 to Disrupt Face Pre-processing 122
6.6.1 Current Solutions 123
6.6.2 Discussion of Stage 2 Solutions 124
6.7 Attacking Stage 3 to Corrupt Feature Extractor 125
6.7.1 Current Solutions 125
6.7.2 Discussion of Stage 3 Solutions 126
6.8 Attacking Stage 4 to Corrupt Database 127

6.8.1 Current Solutions . . . 127
6.8.2 Discussion of Stage 4 Solutions . . . 128
6.9 Attacking Stage 5 to Evade Identification . . . 129
6.9.1 Current Solutions . . . 129
6.9.2 Discussion of Stage 5 Solutions . . . 130
6.10 Goals and Tradeoffs in AFR Design . . . 131
6.10.1 Five AFR Design Properties . . . 132
6.10.2 Implications of Properties for AFR Design . . . 132
6.11 Challenges for AFR Tools . . . 135
6.11.1 Technical Challenges . . . 135
6.11.2 Broader Social and Ethical Considerations . . . 139
6.12 Discussion . . . 140

7 CONCLUDING THOUGHTS . . . 142
7.1 Limitations of Proposed Solutions. . . . 142
7.2 Considerations for Future Data Agency Solutions . . . 144
7.3 Regulation and Data Agency . . . 146
7.3.1 Closing Thoughts. . . . 147

REFERENCES . . . 148

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.* 16

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).* 24

3.3 *Before Cloaking* 29

3.4 *After Cloaking* 29

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.* 29

3.6 *Protection performance as DSSIM perturbation budget increases. (User/Tracker:* `Web-Incept`*)* 29

3.7 *Pairs of original and cloaked images ($\rho = 0.007$).* 31

3.8 *Protection performance improves as the number of labels in tracker's model increases. (User/Tracker:* `Web-Incept`*)* 33

3.9 *Cloaking is less effective when users and trackers use different feature extractors. (User:* `VGG2-Dense`*, Tracker:* `Web-Incept`*)* 33

3.10 *Cloaks generated on robust models transfer better between feature extractors. (User:* `VGG2-Dense`*, Tracker:* `Web-Incept`*)* 33

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).* . 39

3.12 *Protection success rate decreases when the tracker has more original user images. (User/Tracker:* `Web-Incept`*)* 39

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`*)* 39

3.14 *Sybils jointly optimized on four feature extractors have reasonably high protection success for each individual extractor.* 39

3.15 *Normal classification accuracy decreases as input blurring increases but protection success rate remains high.* 41

3.16 *Normal classification accuracy decreases as Gaussian noise is added to inputs but protection success rate remains high.* 41

3.17 *Protection success rate and normal classification accuracy increase as image quality increases using JPEG compression.* 41

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`*)* 42

4.1 *Control over data content, data labels, and model training by different players in the ML ecosystem.* 49
4.2 *The presence of a spurious feature "mark" on images subtly increases the probability of the marked class in a model's probability output. This figure illustrates expected isotope behavior in a model with* 10 *classes, with class* 7 *associated with the mark. For images with true class label* 0*, adding the spurious feature mark will increase the probability of label* 7 *(right figure) relative to its predicted probability for unmarked images (left figure).* 56
4.3 *A high-level overview of our isotopes methodology: ① User* U_1 *posts a set of images online, including data isotopes; ② Model trainer* $\mathbb{F}$ *collects these images to create a dataset* $\mathcal{D}$*; ③* $\mathbb{F}$ *trains model* $\mathcal{F}_\theta$ *on* $\mathcal{D}$*; ④* U_1 *queries* $\mathcal{F}_\theta$ *and uses verifier* $\mathcal{V}$ *to determine if their isotope images were used to train* $\mathcal{F}_\theta$*.* 57
4.4 *Detailed illustration of isotope creation and detection, explained in §4.4.2 and §4.4.3.* 59
4.5 *Different marks used in our experiments.* 63
4.6 *Comparison of* $\mathcal{V}$ *performance for different* Q *values and on paired external marks.* 64
4.7 *Visibility of ImageNet blend mark increases with* α*.* 64
4.8 *Average* $\mathcal{V}_T$ *values at* $\lambda = 0.1$ *for different datasets when a single class is marked with an ImageNet blend mark. For most datasets, marking is effective when* $\alpha \geq 0.4$ *and* $p \geq 0.1$*.* . . . 65
4.9 *Average* $\mathcal{V}_T$ *values for different marks in a* `CIFAR100` *model. Marks that introduce stronger features into images (like Hello Kitty and Imagenet Blend) perform much better.* 66
4.10 *Ablation over* α *and* p *for a* `PubFig` *model with* 20 *marked classes, using* $\lambda = 0.1$ *for* $\mathcal{V}$ *and* $\mathcal{V}_D$*.* 67
4.11 $\mathcal{V}_T$ *and* $\mathcal{V}_{D_T}$ *for multi-mark settings with up to* 50% *of classes marked. We add marks using* $\alpha = 0.4$ *and* $p = 0.1$ *for all datasets, and we evaluate using* $\lambda = 0.1$*.* 67
4.12 *When marks have a normalized* L_{inf} *distance of* < 0.2*, mark distinguishability sharply decreases.* 69
4.13 *As the proportion of marked images grows, model accuracy remains overall unaffected, but marked class accuracy decreases.* 69
4.14 *Examples of physical world marks from the* `WengerFaces` *dataset used in our experiments.* . 69
4.15 *Isotopes remain detectable in a transfer learning setting when at least* 3 *layers are unfrozen during training.* 74
4.16 *The state-of-the-art spurious correlation detection method we test cannot flag isotopes with reasonable settings like* $p = 0.1$ *and* $\alpha = 0.4$*.* 77
4.17 *Adding Gaussian noise with* $\mu = 0$ *and increasing* σ *to* $\mathcal{D}$ *images degrades* $\mathcal{F}_\theta$ *accuracy faster than* $\mathcal{V}_T$ *or* $\mathcal{V}_{D_T}$*.* 77
4.18 *Adding new marks to* $\mathcal{D}$ *images decrease* $\mathcal{F}_\theta$ *accuracy more than* $\mathcal{V}_T$ *or* $\mathcal{V}_{D_T}$*.* 77
4.19 *Adding Gaussian noise to* $\mathcal{F}_\theta$ *outputs degrades* $\mathcal{F}_\theta$ *accuracy before it decreases* $\mathcal{V}_T$*.* 77
4.20 *Returning only the top-*K *outputs reduces tag distinguishability but not detectability.* 77
4.21 *Effect of targeted retraining of tagged classes in* `CIFAR100` *using* `Scrub` *data. Class accuracy decreases more quickly than* $\mathcal{V}_T$*.* 77

5.1 Attack success rates of physical triggers in facial recognition models trained on various architectures. . . . 85
5.2 *Examples of clean and poison data used in the object recognition experiments of §5.* . . . 89
5.3 *Loss trajectory at various learning rates for a facial recognition model with ("Clean Loss," "Attack Loss") and without ("Clean Loss (No Trig)") a glasses trigger backdoor. Results shown are for a VGG16 architecture and a 0.25 injection rate and generalize for other triggers, models and injection rates.* . . . 90
5.4 *Backdoored model performance (in terms of model accuracy on clean input and attack success rate) using different physical triggers when varying the injection rate. Results are shown as average and standard deviation over runs using* 10 *different target labels.* . . . 93
5.5 *Physical backdoor performs well in the object recognition setting.* . . . 93
5.6 *Impact of blurring.* . . . 95
5.7 *Impact of image compression.* . . . 95
5.8 *Impact of Gaussian noise.* . . . 95
5.9 *CAM of an earring-backdoored model highlights on-face features for both clean and backdoored inputs.* . . . 95
5.10 *Backdoor attack success rate decreases as the black earring trigger moves off the face.* . . . 95
5.11 *Attack performance when the attacker poisons training data using digitally inserted triggers, tested on two types of backdoored images: images with digitally inserted trigger (attack digital) and images with real triggers (attack real).* . . . 99
5.12 *False positive rate for inputs containing objects visually similar to the real bandana trigger, before and after the attacker applies the false positive training based mitigation.* . . . 99

6.1 *The workflow of how facial recognition systems recognize a human face in an input image, along with the corresponding terminology. (a): A query image, after being submitted to the system, is passed to the feature extractor to produce a feature vector; (b): this feature vector is used to query a reference database of labeled feature vectors; (c): if the query feature vector matches a labeled feature vector in the database, the label is used to find a reference image, and the system outputs the reference image and the identity (i.e. Alice Smith in this example).* . 106
6.2 *We propose to divide the operational pipeline of a FR system* $\mathbb{F}=\{\mathcal{G},\mathcal{F},\mathcal{C},\mathbf{D}\}$ *into a set of five operational stages ① to ⑤. They encompass the five critical points of direction interaction between users and FR systems. Later we will use this framework for analyzing AFR systems.* . 109
6.3 *Overview of our proposed stage-based framework for analyzing existing AFR proposals. We list the five critical stages of facial recognition as discussed in §6.2.2 and present AFR strategies per stage by the attack target, action, and desired effect.* . . . 115

LIST OF TABLES

3.1 *The four feature extractors used in our evaluation, their classification efficacy and those of their student models.* . . . 26
3.2 *Datasets emulating user images in experiments.* . . . 27
3.3 *Protection performance of cloaks generated on robust feature extractors.* . . . 31
3.4 *Cloaking is highly effective against cloud-based face recognition APIs (Microsoft, Amazon and Face++).* . . . 34

4.1 *Summary of prior work on ML data provenance and whether it fulfills requirements for a user-centric ML data provenance solution.* ✓ *indicates that a solution fulfills a given requirement, while — indicates it does not.* . . . 50
4.2 *Notation used in this paper* . . . 52
4.3 *Model training details for each task.* . . . 63
4.4 *TPR/FPR for multiple marks per class at* $\lambda = 0.1$ *and* $\delta = 0.6$*. In all cases,* $\mathcal{V}_T > 0.8$ *and* $\mathcal{V}_F < 0.12$*, even with up to* 6 *marks per class.* . . . 68
4.5 $\mathcal{V}$ *can detect some physical marks when* $\lambda = 0.4$. . . . 71
4.6 *Isotopes remain detectable in models via the Google Cloud ML API, both in the single and multiple (25) marked class setting.* . . . 73
4.7 *Results from isotope detection in Amazon Rekognition. For* `set1` *and* `set2`*, the true match is always the top match. For unenrolled isotope images (*`3`*), isotope images with the same mark appear in the top* 5 *hits.* . . . 74
4.8 *Precision and recall of Spectral [323] and Clustering [66] on* `CIFAR100` *with* 25 *marked classes.* . . . 78
4.9 *Prediction outputs returned by different ML services. Most services return all labels that match the input with more than a certain "confidence" threshold level, set by the user performing the query.* . . . 81

5.1 *Backdoored model performance (in terms of model accuracy on clean input and attack success rate) using different physical triggers at the injection rate of 0.25. Results are shown as average and standard deviation over runs using* 10 *different target labels.* . . . 93
5.2 *Backdoor effectiveness drops considerably when triggers move off the face, using the VGG16 model.* . . . 95
5.3 *Attack performance when the attacker can only poison training data from 10 out of 75 classes.* 97
5.4 *Physical backdoor detection rates for four defenses. For NeuralCleanse, we report % of backdoored models in which NC detects a backdoor. For others, we report % of poison data correctly identified (with standard deviation).* . . . 103

6.1 *Example use cases of facial recognition.* . . . 113
6.2 *Reported reference image sources* . . . 114
6.3 *Taxonomy of proposed AFR tools. "BB/WB" = Black Box, White Box."UT, T" = Untargeted, Targeted. "AR, AZ, F++, TN" = Amazon Rekognition, Microsoft Azure Face Recognition, Megvii's Face++, Tencent Face Recognition.* . . . 117

6.4 *Evaluating AFR tools using five properties, where the tools are grouped by the FR stage they target.* ● *means that the property has already been achieved by current AFR proposals targeting this stage;* ● *means that the property seems "promising" and could be achieved by AFR designs targeting this stage; and* **?** *indicates significant progress may be required to achieve this property by targeting this stage, and the likelihood of success is unknown.* 131

ABSTRACT

As machine learning (ML) models have grown in size and scope in recent years, so has the amount of data needed to train them. Unfortunately, individuals whose data is used in large-scale ML models may face unwanted consequences. Such data use may violate individuals' privacy [114, 60] or enroll them in an unwanted ML application [142, 8]. Furthermore, recent advances have greatly enhanced models' ability to generate synthetic data like text and images [50, 377, 263, 158, 303]. This has unleashed a fresh wave of privacy and intellectual property concerns, as generative models can memorize and regurgitate their training data [332, 155, 57, 58, 285], and are trained on massive datasets scraped from the internet [282, 5].

While user data privacy issues are well-recognized in the ML research community, most attempts to address it thus far take a heavily *model-centric approach*. Existing solutions typically assume that model trainers are well-intended and that data has been taken with consent, or, more pessimistically, that data use is inevitable and that the best path forward is to mitigate privacy risks. Consequently, these approaches seek to preserve data privacy during training [105, 218, 125, 137, 37] or prevent unwanted memorization [157, 59, 184]. These solutions typically achieve their objectives but overlook a significant problem: often data is not taken with consent [300, 30, 15], and users do not trust model trainers to do right with their data [42].

This begs the question: what if data use was *not* inevitable? What if, instead, users had agency over *how and if* their data is used in ML systems? This thesis argues that ***data agency***, the ability to know and control how and if one's data is used in ML systems, is an important complement to existing ML data privacy protection approaches. Such agency would shift the current power dynamic, which renders users helpless at the hands of model creators, and help users control their digital destinies. Solutions of this nature would accentuate current work on data privacy, giving users, not just model trainers, control over how their data is used.

This thesis explores solutions that provide users with *data agency* against large-scale ML systems. Such agency can take many forms, but this thesis considers *data agency solutions allowing*

individuals to disrupt or discover when their data is used in large-scale ML systems. It proposes three solutions that prevent or trace data use in ML systems or, in extreme cases, directly attack the ML system. In proposing these solutions, it focuses on the use case of large-scale facial recognition (FR) systems, a machine learning technology that has recently become a flashpoint for civil liberties and privacy issues. With this use case in mind, the thesis finally develops a framework for reasoning about broadly about FR data agency and uses this framework to outline both technical and social challenges of proposed solutions.

Content Summary and Outline

This thesis defines the problem of data agency against ubiquitous ML systems (§1) and develops technical solutions to address it. It presents relevant background information for these solutions (§2), then proposes three solutions for reclaiming data agency: **disruption** (§3), **tracing** (§4), and **direct attacks** (§5). Then, it **develops a framework** (§6) for data agency against unwanted FR systems, which can serve as a template for similar data agency frameworks in other domains. Finally, it considers limitations of the proposed tools and discusses potential future work (§7).

CHAPTER 1

WHY IS DATA AGENCY NECESSARY?

In recent years, machine learning (ML) models have been eagerly adopted to perform a variety of tasks, from real-time language translation to image synthesis to facial recognition. To feed their data-hungry models, ML practitioners use a variety of data sources, from public websites [142, 300] to private application data [71] to surveillance data from public spaces [313]. While sometimes this data is obtained with consent, there are numerous well-documented cases of user data being obtained through dubious means. For example, the facial recognition company Clearview.ai developed a facial recognition model by scraping 3 billion images from social media sites [142]. The AI research company DeepMind funneled data from a UK health services app to train ML models for medical diagnosis [27].

As models grow larger, the demand for data only grows. In recent years, companies like OpenAI and DeepMind have celebrated the release of billion or trillion-parameter models, which require terabytes of data to properly train. To train models of this scale, ML practitioners often resort ot using large internet scrapes to create datasets. For example, text generation models like ChatGPT are trained on datasets like Common Crawl [5], which is an open repository containing petabytes of scraped web page data and metadata. Image synthesis models like DallE and Stable Diffiusion are trained on datasets like LAION [282], created by scraping online image repositories.

In datasets scraped exclusively from web content, user consent is difficult, if not impossible, to obtain. Some websites request blanket consent from users, allowing the use of their data for things like tracking and, relatedly, model training. Often, though, the operating assumption is that using an online service or app constitutes consent to one's data taken and used. Or at least this is the operating assumption of ML developers who create large-scale internet datasets like LAION or CommonCrawl.

From these examples, a simple truth emerges—individuals who use an online service or occupy a public space may unwittingly produce data that is used in ML models. Beyond issues of consent,

such data use poses a serious risk to individual privacy. Prior work has shown that models can memorize and regurgitate their training data, revealing private information about individuals whose data is in the dataset [114, 60]. Furthermore, users whose data is co-opted for ML use may end up enrolled in a privacy-compromising system, such as a large-scale facial recognition model licensed to whoever wants to pay [142].

1.1 Existing Data Privacy Solutions

The problem of data privacy in the machine learning space has been well-recognized, and numerous solutions have been proposed to address this issue. Proposed ML data privacy solutions can be categorized into two distinct groups, each grounded in a different understanding of privacy. Each set of solutions has specific goals, benefits, and limitations.

Solution 1: Hide User Data Content. The first set of solutions take the stance that *only the user should know the content of their data.* This approach begets privacy solutions like federated or split learning and encrypted training [218, 137, 125]. Federated and split learning techniques allow parties to host data locally and, without sharing the content of that data, collaboratively train a model [255, 333, 166]. The resulting model should be effective and useful to the training data contributors, despite the lack of a centralized training dataset. Collaborative learning solutions are commonly used by companies like Google and Apple to train models that can be shared among their users [362, 251]. Encrypted training solutions use methods like homomorphic encryption to encrypt user data during training [125, 37, 143]. The trained model can be decrypted and used without revealing the private training data.

Solutions in this group have several major limitations. Federated and split learning methods are vulnerable to leakage attacks, allowing real-time reconstruction of training data via information passed between the collaborative training parties [249, 45]. Encrypted training methods typically incur significant time and memory overhead costs [125, 37, 143]. This is because operations on encrypted data are slower than operations on normal data, and because the additional data struc-

tures needed to facilitate encrypted training are memory-intensive. Finally, neither collaborative nor encrypted learning methods inherently prevent private data leakage from the trained model. Attacks like membership inference could still recover information about which data points were used for training, even though the training process was itself private [294, 310].

Solution 2: Make User Data Indistinguishable. Another data privacy approach requires that *data points should not be linkable to the user who created them.* In contrast to the first set of solutions, which prevent direct observation of training data points, solutions in this vein make it difficult (or impossible) to determine if a *particular* individual's data were used to train a model. These solutions target downstream privacy attacks like membership inference [69, 370]. The canonical solution in this vein is the use of differential privacy [105] during training. Differential privacy (DP) techniques add a carefully tuned amount of noise to training data to ensure that no single data point has an outsized influence on model training. The DP noise should prevent an attacker from deducing that a particular individual's data, with certain traits, was present in the training dataset. DP has seen widespread adoption in commercial and government spaces, most notably in the 2020 US Census [33].

DP and related solutions have a significant practical limitation: models trained on differentially private data often perform poorly and have low accuracy. This is due to the fundamental trade-off between noisy data and final model precision. Reducing the level of DP noise improves model usability, but increases vulnerability to membership inference attacks [262]. This represents a significant drawback of these methods—if data is private, but the model is not useful, there is little incentive to use this method.

Claim: Data Privacy is Not Sufficient. The two existing sets of data privacy solutions can be useful for specific applications. However, they share a common blind spot: they assume data use is inevitable and seek to mitigate privacy risks during model training and inference, *after users' data has already been taken*. Though important, post-facto privacy protection is not the only way to address issues of unwanted ML data use. Privacy-conscious individuals may desire agency

over *how and if* their data is used, rather than merely having their privacy preserved when it is used. Studies show that the majority of Americans are concerned about how their online data is being used, and a significant minority believe that the unauthorized use of personal data—even for applications like law enforcement or mental health monitoring—is unacceptable [42].

This thesis argues for a different approach to guarding users' data in ML settings. It proposes a set of user-facing solutions that give users a priori control over how and if their data is used for ML, rather than simply anonymizing data while it is used. These solutions put power back in the hands of the users. With these, users do not have to trust that an entity that collected their data without consent will miraculously protect their privacy while using it. Instead, users control their data. In essence, these solutions provide users with much-needed *data agency*.

1.2 Our Solution: Data Agency

Data agency, the ability to know and control how and if one's data is used in ML systems, is an important complement to existing privacy protection approaches. Although many mechanisms can be used to promote data agency, this thesis proposes technical tools and techniques enabling data agency. These solutions enable individuals to disrupt or discover when their data is used in large-scale ML systems—two essential components of data agency. These technical solutions should be supported by organizational and legislative efforts, but current legislative action around unwanted data use focuses on preserving privacy, rather than promoting agency (e.g. GDPR, Illinois BPDA, etc. [82, 39]). Thus, the technical tools proposed here can serve as a first line of defense for users against unwanted data use, and will hopefully inspire future data legislation and policies to consider agency in addition to privacy.

Proposed Data Agency Techniques. This thesis considers three techniques to promote data agency against unwanted ML models. The first approach *disrupts model training* by rendering data taken without consent unusable for ML. Our disruption solution (see §3) considers the scenario of photos being scraped from social media and used to train facial recognition models. There is a

significant power imbalance between the users who post photos and the tracker who takes and uses the images. The user has little insight into the tracker's methods. However, our solution takes advantage of two facts: that the tracker works at scale and does not target specific users; and that ML models see images as mathematical pixel arrays, different from humans. Using these principles, we construct a data agency solution enabling users to *cloak* their images. Models trained on cloaked images will not recognize real pictures of the user, giving users data agency.

Of course, disruption may not always be feasible, particularly when users' knowledge of the downstream ML setting is minimal. Thus, the second data agency solution considers a weaker but equally important task: allowing users *detect if the data is used to train a model* (see §4). To enable this detection, we propose a data marking scheme that creates *data isotopes* from users' data. Like their chemical counterparts, data isotopes are similar in content to original data (in this context, images) but with a few key semantic changes. When a model is trained on isotope images, the changes made to isotope images leave a detectable statistical bias in the model. By querying a model with a series of normal and isotope images, users can deduce if the bias associated with their isotopes is present. If it is, the model was trained on the users' data. With such knowledge, users can trace how their data has been used and, once legislation and policies catch up, potentially take legal action—restoring agency.

Finally, in extreme cases, data agency can involve *attacking unwanted ML systems*. Such attacks could provide protection for users by embedding controlled misclassification behaviors in the model or could draw public attention the problem of data being used without consent. §5 presents a disruptive data agency solution in this vein. It explores the extension of the well-known backdoor attack against ML models to the physical setting. In particular, it explores the use of physical objects as "triggers" for backdoor (e.g. misclassification) behaviors in face and object recognition scenarios. If backdoor attacks are possible in a real-world scenario, then users could potentially leverage them to disrupt real-time model operation. Depending on the context, this could restore user agency against unwanted surveillance or monitoring.

Data Agency Framework. In addition to the proposed data agency tools, this thesis also provides a *forward-looking analysis* (§6) of the viability of data agency techniques in the facial recognition context. There is a rapidly growing cottage industry of so-called "anti-facial recognition" (AFR) tools designed to counteract unwanted face recognition. Proposals in this space differ widely in their assumptions and techniques, but are united in their common goal of increasing users' agency. To better understand this space, we propose a framework that identifies commonalities in tools and techniques, highlights performance trade-offs of different approaches, and identifies unexplored areas for future development. We then argue that this framework can serve as a template for analysis of data agency solutions in other settings.

CHAPTER 2

BACKGROUND AND RELATED WORK

Before discussing the data agency solutions of this thesis in depth, we first present background information common to all solutions. We give an overview of machine learning, focusing specifically on the models and training techniques considered in this thesis. Then, we broadly summarize the field of adversarial machine learning, which studies security and privacy issues of ML models. Data agency techniques in this thesis often leverage tools from the adversarial ML space to enhance users' power against unwanted models. Background information specific to each proposed solution can be found in that solution's chapter.

2.1 Machine Learning (ML) Overview

2.1.1 Training Image Classifiers via Supervised Learning

Model Task. This thesis primarily considers machine learning models trained to perform image classification via supervised learning. Creating such a model requires first collecting a dataset $\mathcal{D} = \{\mathcal{X}, \mathcal{Y}\}$. Each element $x_i \in \mathcal{X}$ is a data point to be classified—e.g. an image—and its corresponding element $y_i \in \mathcal{Y}$ is its classification label—e.g. image subject. Given $\mathcal{D}$, the goal is to train a model $\mathcal{F}_\theta : \mathcal{X} \rightarrow \mathcal{Y}$ to produce accurate predictions on previously unseen data points similar to those in $\mathcal{D}$. More formally, given new element $x_j \in \mathcal{X}'$ with true label y_j, where $\mathcal{X}' \sim \mathcal{X}$ but $\mathcal{X} \cap \mathcal{X}' = \emptyset$, a properly trained model $\mathcal{F}_\theta$ should produce $\mathcal{F}_\theta(x_j) = y' = y_j$ with high confidence. $\mathcal{F}_\theta$ has $|\mathcal{Y}| = N$ possible classification labels.

Model Training. Training $\mathcal{F}_\theta$ requires solving an optimization problem over $\mathcal{D}$ and θ, the model parameters. This optimization procedure updates θ to minimize $\mathcal{F}_\theta$'s *loss* over $\mathcal{D}$. ℓ is a function specifying the loss, typically cross-entropy in a classification context. A gradient descent procedure is used to select parameters minimizing ℓ via $\min \ell(\mathcal{F}_\theta(x), y)$, where x and y are the true data/label pairs from $\mathcal{D}$. Given the complexity of this optimization procedure, $\mathcal{F}_\theta$ is trained

using techniques like stochastic gradient descent [266], which saves time and computational cost by approximating the true gradient descent procedure needed to exactly minimize ℓ.

Model Architecture. The models used for image classification in this thesis leverage a *convolutional neural network* architecture [183]. This biologically-inspired model architecture attempts to mimic the behavior of the human visual cortex, and is composed of layers containing individual parameters or *neurons*, that supposedly mimic the behavior of neurons in the brain [152]. Convolutional neural networks are composed of input, hidden, and output layers, each containing a set of trainable neurons. Together, these layers perform *convolutions* on an input image (or input features produced by prior layers). The convolution operation computes the dot product of the input features with the layer's convolutional kernel. This generates a feature map abstractly representing key elements of an image, which is then fed to the next model layer. Eventually, these features are mapped to a *classification output layer*, which computes the probability that the input image belongs to a certain class. Models like ResNet, DenseNet, and Inception—which have achieved state-of-the-art performance on various image classification and recognition benchmarks—leverage convolutional architectures [141, 311, 150, 174]. For more information on convolutional neural networks, we refer the interested reader to [127].

2.1.2 Other ML Training Settings and Tasks

The data agency solutions in this thesis focus on the context of image classification models, so the supervised learning setting described above applies to all proposed solutions. However, there are many ML tasks beyond image classification, and numerous ways to train machine learning models. Future work can and should explore data agency solutions for these tasks and learning paradigms.

Alternative ML Tasks. ML models can classify much more than images, and have been regularly used to classify text, tabular data, and other types of inputs. Furthermore, although classification is one of the most popular tasks for ML models, it is far from the only one. ML models can also perform recognition tasks, like pattern recognition or speech recognition [47, 93]. Beyond this,

models can perform a variety of generation and synthesis tasks, such as summarizing text, creating fake images, or producing synthetic speech. Recent advancements have led to breakthroughs in the performance of text generation models like ChatGPT [50, 377] and text-to-image models like Stable Diffusion and Midjourney [263, 303, 158].

Alternative Training Paradigms: Semi-Supervised, Unsupervised. Models can be trained via supervised, semi-supervised, or unsupervised learning techniques [127]. The latter two methods rely less on provided labels $\mathcal{Y}$ (if any are provided at all) and instead train the model to discern patterns in the data $\mathcal{X}$ itself. Semi- or unsupervised learning is useful for teaching models generative tasks, like image or text generation. Different loss functions and optimization procedures are needed in this setting. More information on unsupervised and semi-supervised learning can be found in [127].

Alternative Model Architectures. Recent work has explored the use of alternative architectures for image classification tasks, such as vision transformers [104]. While this thesis focuses on data agency solutions for models trained on a convolutional neural network architecture, the principles proposed here could naturally extend to other architectures. This is a ripe avenue for future work.

2.2 Attacks and Defenses for ML Image Classification Models

Prior work has identified numerous ways that the behavior of ML image classification models can be disrupted or influenced by interested parties. Traditionally, these techniques have been considered "attacks" on models, revealing the underlying assumption in the ML community that models are good and ought to be protected. However, this thesis argues that the decision to label these techniques as either an attack on models or a defense for users depends on many factors. Several data agency solutions in this thesis leverage techniques traditionally considered "attacks" on models to instead defend users from unwanted ML intrusions. Thus, for context, we present a high level summary of canonical techniques that can alter model behaviors. Although these techniques often generalize beyond the image classification setting, we here consider the image

classification context given the focus of this thesis. We preserve the model-centric "attack/defense" language throughout this section for clarity, given its prevalence in academic literature.

2.2.1 Adversarial Example Attacks and Defenses

One of the earliest attacks identified against ML image classification models is known as an *adversarial example attack*. Such an attack crafts a special perturbation (ϵ) for a normal input x to fool a target neural network $\mathcal{F}_\theta$. When ϵ is applied to x, the neural network will misclassify the adversarial input ($x + \epsilon$) to a target label (y_t) [312]. That is, $y_t = \mathcal{F}_\theta(x + \epsilon) \neq \mathcal{F}_\theta(x)$. Many methods for generating such adversarial examples (*i.e.* optimizing a perturbation ϵ) have been proposed. These attacks can be categorized as either *white box* or *black box*, based on the assumptions they make.

White Box Attacks and Defenses. White box adversarial attacks assume that the attacker has full access to $\mathcal{F}_\theta$, including its parameters, while constructing ϵ. There are numerous ways to compute adversarial examples in this threat paradigm [129, 178, 63, 72, 331], with variants stemming from how the attack gradient is computed, as well as what metric is used to constrain the size of the visual perturbation. Defenses against white-box adversarial attacks have been well-studied [378, 211, 373, 245, 357, 208, 209, 286], although it is often the case that new defenses are broken by newer, stronger attacks (e.g. [222] and [62], or [245] and [61]).

Black Box Adversarial Attacks and Defenses. A black box attacker has query-only access to $\mathcal{F}_\theta$, meaning they can only submit inputs to $\mathcal{F}_\theta$ and observe the classification response. Using this limited information, the adversary then attempts to construct an adversarial example ($x + \epsilon$). Existing black-box attacks can be divided into two types: substitute model attacks and query-based black-box attacks. In a substitute model attack, the attacker uses $\mathcal{F}_\theta$'s query responses to build a labeled dataset and train a *substitute model* $\mathcal{F}_\theta{}'$ that approximates $\mathcal{F}_\theta$ and enables generation of adversarial examples that succeed on $\mathcal{F}_\theta$ [242, 243, 355, 101, 351]. In query-based attacks, the attacker uses past query results to iteratively perturb the next query, hoping to converge to a successful adversarial example [73, 154, 327, 67, 46, 228]. Defenses against substitute model

attacks include adversarial training [179], ensemble adversarial training [321], and adversarial training with single-step R+FGSM attack [348]. Defenses against query attacks attempt to detect query patterns and stop the attack before it succeeds [74, 188].

2.2.2 Poisoning Attacks and Defenses

A second category of attacks against ML image classification models are *poisoning attacks*. In a poisoning attack, an attacker disrupts the training process of $\mathcal{F}_\theta$, typically by adding so-called poison data to the training dataset $\mathcal{D}$. The poison data points are designed to induce an attacker-chosen behavior in trained model. Possible behaviors range from degraded accuracy/performance on all or certain inputs to misclassification of specific inputs. Here, we give a brief overview of two well-known poisoning attacks: backdoor attacks and clean-label poisoning.

Backdoor Attacks and Defenses. An attacker launches backdoor attacks against a DNN model in two steps. During model training, the attacker poisons the training dataset by adding samples associating inputs containing a chosen pattern (the trigger δ) with a target label y_t [131, 203]. This produces a backdoored model that correctly classifies benign inputs but "misclassifies" any input containing the trigger δ to the target label y_t. At inference time, the attacker activates the backdoor by adding the trigger δ to *any* input, forcing the model to classify the input as y_t. Over time, backdoor attacks have evolved to use stealthier triggers and smaller amounts of training data [197, 189, 367, 275, 192]. Defenses against backdoors employ a wide variety of techniques, from scanning model classification results to reverse-engineer backdoor triggers and remove them from the model [338], pruning redundant neurons to remove backdoor triggers [118], or detecting the presence of poisoning data in the training dataset [66, 323].

Clean Label Poisoning. Clean label poisoning attacks force misclassification of a specific input by $\mathcal{F}_\theta$. Critically, in such an attack, *the attacker adds poison data to $\mathcal{D}$ that is labelled correctly*. This stands in contrast to traditional backdoor attacks, which rely on label changes in the poisoned training data to induce the desired behaviors. Note that clean label techniques can be used to

launch backdoor attacks, so the distinction between these two categories can be murky [284, 287]. Limited defenses against clean label poisoning have been proposed, mostly focusing on detecting clean label poison data points as outliers in $\mathcal{D}$ [252].

CHAPTER 3

DATA AGENCY VIA DISRUPTION—*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 [229, 279]. At many US airports, international travelers must submit to facial recognition systems in order to enter the country [239]. 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 [146]. 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 [295]. Stores can associate our in-store shopping behavior with online ads and browsing profiles [214]. Identity thieves can identify (and perhaps access) our personal accounts [85].

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 [352, 191, 309]. 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 [353, 319]. Finally, given access to an image classification

model, "clean-label poison attacks" can cause the model to misidentify a single image [284, 382].

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 (DSSIM ≤ 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 [312] (see §2.2.1 for more details). These attacks have since been proven possible "in the wild," Sharif *et al.* [288] create specially printed glasses that cause the wearer to be misidentified. Komkov and Petiushko [171] 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

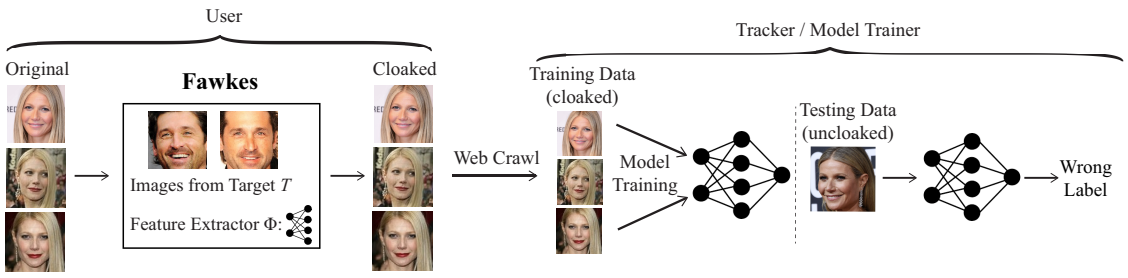


Figure 3.1: *Our proposed Fawkes system that protects user privacy by cloaking their online photos. (Left) A user U applies cloaking algorithm (given a feature extractor Φ and images from some target T) to generate cloaked versions of U's photos, each with a small perturbation unnoticeable to the human 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.*

identification" models, making it difficult for models to recognize the wearer as a person in an image [353, 319].

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 [233], facial inpainting [308], and GAN-based face editing [352, 191, 309]. 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 (see §2.2.2 for more details). Previous work has used data poisoning to induce unexpected behav-

iors in trained DNNs [360]. 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 [284, 382]. 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 [382]. *Third*, clean label attacks are easily detectable through anomaly detection in the feature space [136].

Model Corruption Attacks. Other recent work proposes techniques to modify images such that they *degrade* the accuracy of a model trained on them [290]. 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, [290] 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 [301]. This can be exploited to expose private details about members of the training dataset [114]. These attacks have spurred a push towards *differentially private* model training [32], which uses techniques from the field of differential privacy [105] 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 [371]) at reasonable training costs.

In transfer learning, the knowledge of a pre-trained feature extractor Φ is passed on to a new model $\mathbb{F}_\theta$. Typically, a model $\mathbb{F}_\theta$ 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 $\mathbb{F}_\theta$ 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 [76] 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 [339, 284, 382]), 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 person[1]. 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 [284, 382]. 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:

$$\max_{\delta} Dist\left(\Phi(x), \Phi(x \oplus \delta(x, x_T))\right), \quad \text{subject to } |\delta(x, x_T)| < \rho, \tag{3.1}$$

where $Dist(.)$ 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 actress) as a dissimilar target T for the original user (female actor 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:

$$\min_{\delta} Dist\left(\Phi(x_T), \Phi(x \oplus \delta(x, x_T))\right), \quad \text{subject to } |\delta(x, x_T)| < \rho. \tag{3.2}$$

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

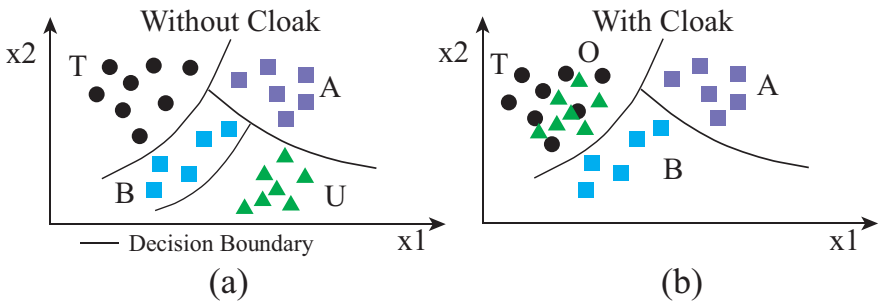


Figure 3.2: *The intuition for why a tracker's model trained on U's cloaked photos will misclassify U's original photos, visualized on a simplified 2D feature space with four user classes A, B, U (aka Alice), T. (a) decision boundaries of the model trained on U's uncloaked photos. (b) decision boundaries 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 [90, 242, 306, 371].

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 em-

pirically 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.*

$$T = \underset{\boldsymbol{k=1..K}}{\text{argmax}} \min_{\boldsymbol{x \in \mathbf{X_U}}} Dist(\Phi(x), \mathcal{C}_k). \tag{3.3}$$

We use L2 as the distance function in feature space, $Dist(.)$.

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) [343, 344]. Different from the L_p distance used in previous work [63, 178, 286], DSSIM has gained popularity as a measure of user-perceived image distortion [339, 161, 194]. 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* [237] to solve the optimization in eq.(3.2) as follows:

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

Here λ controls the impact of the input perturbation caused by cloaking. When $\lambda \to \infty$, 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 [63].

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 [347]. 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 ($\geq$ 500K 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 [56].
- WebFace has $500,000$ images of faces covering roughly $10,000$ subjects from the Internet [369].

Using these two datasets, we build four feature extractors, two from each. We use two different model architectures: a) DenseNet-121 [150], a 121 layer neural network with 7M parameters,

and b) InceptionResNet V2 [311], a 572 layer deep neural network with over 54M parameters. Our trained models have comparable accuracy with previous work [56, 339, 232] 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 figures[2] [9].
- `FaceScrub` contains $100,000$ images of 530 public figures on the Internet [234][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 [366] only retrieved $48,579$ images.

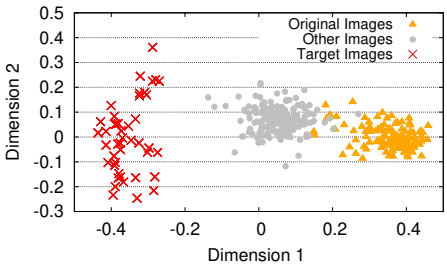


Figure 3.3: *Before Cloaking*

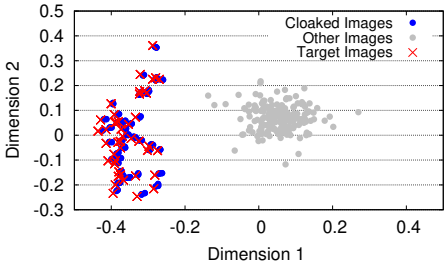


Figure 3.4: *After Cloaking*

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.*

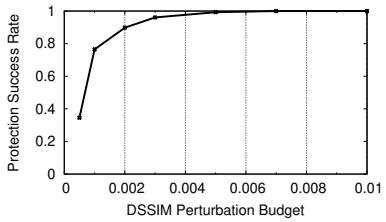


Figure 3.6: *Protection performance as DSSIM perturbation budget increases. (User/Tracker:* `Web-Incept`*)*

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 im-

pact 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 [194] 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 [339, 201].

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 $> 99\%$ for more than 6 labels, confirming that a higher label density improves cloaking effectiveness.

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

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.*

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 transfer-

ability, we draw on recent work linking model robustness and transferability. Demontis *et al.* [90] 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, 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* [211, 129], 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 [178], a widely used method for adversarial training. Following prior work [211], we run the PGD[4] 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 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

4. We found that robust models trained on CW attack samples [63] produce similar results

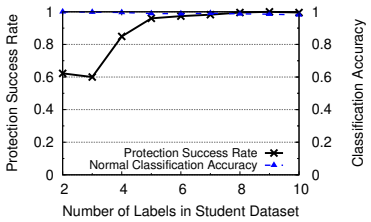


Figure 3.8: *Protection performance improves as the number of labels in tracker's model increases. (User/Tracker:* `Web-Incept`*)*

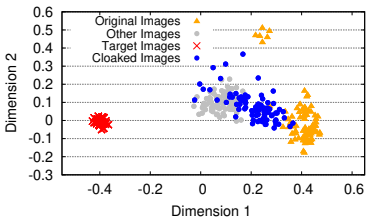


Figure 3.9: *Cloaking is less effective when users and trackers use different feature extractors. (User:* `VGG2-Dense`*, Tracker:* `Web-Incept`*)*

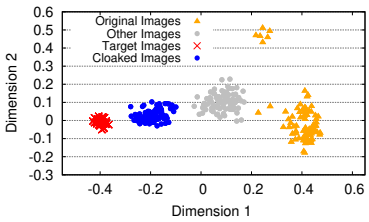


Figure 3.10: *Cloaks generated on robust models transfer better between feature extractors. (User:* `VGG2-Dense`*, Tracker:* `Web-Incept`*)*

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++ [11]. 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 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

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++).*

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 Services, 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++ [11] 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 authentication 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 presence/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 [365], and are often used by real users to curate and compartmentalize content for different audiences [185]. While there are numerous techniques for Sybil detection, individual Sybil accounts are difficult to identify or remove [341].

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 net-

works (GANs) [224]. 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 ready-to-upload Sybil image $x_S = x_C \oplus \delta(x_C, x)$.

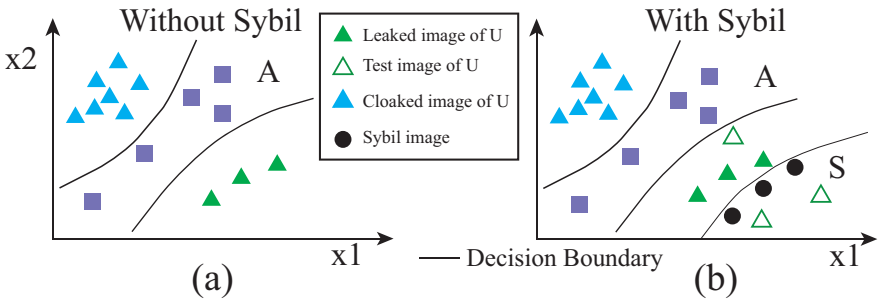


Figure 3.11: *Intuition behind Sybil integration visualized in a 2D feature space. Without Sybils, a tracker's model will use leaked training images of U to learn U's true feature space (left), leading to the correct classification of images of U. Sybil images S complicate the model's decision boundary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

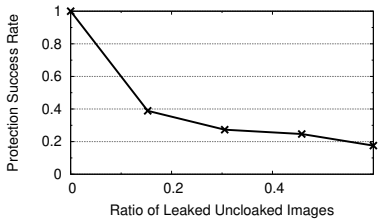


Figure 3.12: *Protection success rate decreases when the tracker has more original user images. (User/Tracker:* `Web-Incept`*)*

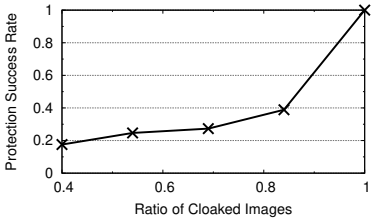


Figure 3.13: *Protection success rate is high when the user has a Sybil account, even if tracker has original user images. (User/Tracker:* `Web-Incept`*)*

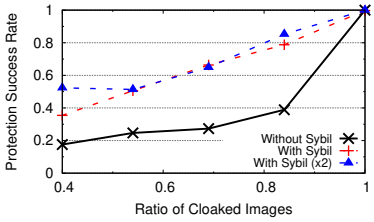


Figure 3.14: *Sybils jointly optimized on four feature extractors have reasonably high protection success for each individual extractor.*

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 robust `VGG2-Dense` feature extractor from §3.5.3.

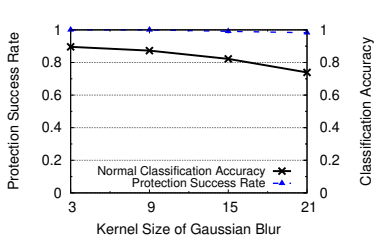


Figure 3.15: *Normal classification accuracy decreases as input blurring increases but protection success rate remains high.*

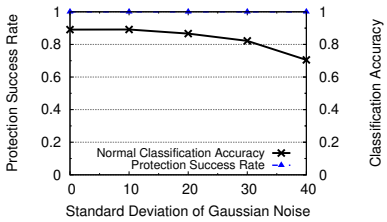


Figure 3.16: *Normal classification accuracy decreases as Gaussian noise is added to inputs but protection success rate remains high.*

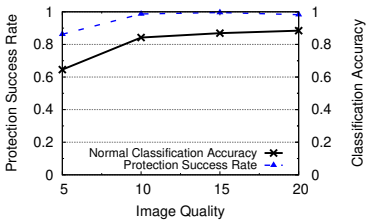


Figure 3.17: *Protection success rate and normal classification accuracy increase as image quality increases using JPEG compression.*

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 [63, 110]. 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.

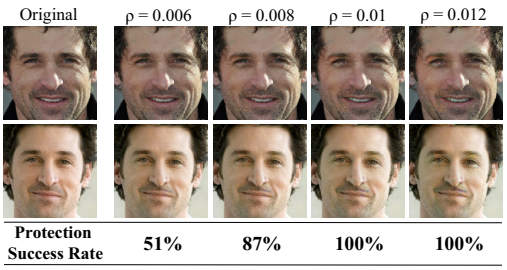


Figure 3.18: *When the user's feature extractor is much less robust than the tracker's feature extractor, the user can improve their protection success rate by increasing their DSSIM budget. (User:* `VGG2-Dense`*, Tracker:* `Web-Incept`*)*

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 images [5]. 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 [337], reportedly used by Facebook and Twitter, to compress the images in our dataset. The image quality, as standard by Independent JPEG Group [12], 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 [326], 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^o, 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 [136, 305, 250, 338, 66, 292] 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

DATA AGENCY VIA TRACING—*DATA ISOTOPES FOR DATA PROVENANCE IN DNNS*

4.1 Introduction

As machine learning (ML) systems grow in scale, so do the datasets they are trained on. State-of-the-art deep neural networks (DNNs) for image classification and language generation are trained on hundreds of millions or billions of inputs [50, 300, 377]. Often, training datasets includes users' public and private data, collected with or without users' consent. Examples include training image analysis models on photos from Flickr [300], companies like Clearview.ai training facial recognition models on photos scraped from social media [144], DeepMind training a kidney disease prediction model on records from U.K.'s National Health Service [204], and Gmail training its Smart Compose text completion model on users' emails [71].

Today, users have no agency in this process, beyond blindly agreeing to the legal terms of service for social networks, photo-sharing websites, and other online services. Even when users give permission for use of their images, they have little control over how those images may later be shared or disseminated [181]. Beyond searches through specific public datasets like LAION-5B [167], every day users have no systematic way to check whether their data was used to train a model [300].

In this paper, we design, implement, and evaluate a practical method that enables users detect if their data was used to train a DNN model, with only query access to the model and no knowledge of its labels or parameters. Our main idea is to have users introduce special inputs we call *isotopes* into their own data. Like their chemical counterparts, isotopes are similar to normal user data, with a few key differences. Our isotopes are crafted to contain "spurious features" that the model will (mistakenly) consider predictive for a particular class during training. Isotopes are thus amenable to a new type of inference: a user who knows the isotope features can tell, by interacting with a

trained model, whether isotope inputs were part of its training dataset or not. Similar inference attacks, such as membership inference [294], are typically interpreted as attacks on the privacy of training data. We—helped by the propensity of DNN models to learn spurious correlations—turn them into an effective tool for tracing data provenance. This tool enables data agency by making it possible for users to trace how their data is used in downstream machine learning applications.

Our contributions. We present a practical data isotope scheme that can be used to trace image use in real-world scenarios (e.g., tracing if photos uploaded to a social website are used for DNN training). The key challenge is that users neither know, nor control the supervised classification tasks for which their images may be used as training fodder. While users are free to modify the content of their images, they do not select the corresponding classification labels, nor know the other labels, nor have any visibility into the models being trained. This precludes the use of "radioactive data" [272], "backdoor" techniques [149], and other previously proposed methods for dataset watermarking (more discussion in §4.2.2).

Our method creates isotopes by blending out-of-distribution features we call *marks* into images. When trained on these isotopes, a model learns to associate one of its labels with the spurious features represented by the mark. By querying the model's API, a user can verify that the presence of the mark in a test image alters the probability of a low-likelihood output label in a statistically significant way. Verification uses statistical hypothesis testing to determine if the model assigns a consistently higher probability to a certain class when the mark is present, independently of other image features. Success implies the user's marked isotopes were present in the model's training dataset.

A key point of our design is to enable usage by non-ML experts. Our method does not require the user to train shadow or surrogate models, nor compute or analyze gradients of publicly available models. Our key contributions are:

- We propose **a novel method for data provenance in DNN models** using "isotope" data to create spurious correlations in trained models (§4.3, §4.4), and a technique for users to detect if

a model was trained on their isotope data.

- We **demonstrate the efficacy of our isotope scheme on several benchmark tasks**, including the facial recognition tasks `PubFig` and `FaceScrub`, and show that it remains effective even when multiple users independently add isotopes to their respective data (§4.5). Despite the potential challenge of having a model learn many isotope-induced spurious features, we find that our verifier can detect and distinguish isotopes with high accuracy and few false positives, even up to 215 `FaceScrub` isotopes, with minimal impact on normal model accuracy.
- We show that **physical objects can act as isotope marks with up to 95% accuracy** (§4.6), demonstrating that our scheme works even if users cannot digitally modify images of themselves (e.g., when images from surveillance cameras are used to train facial recognition models).
- We **evaluate isotope performance in realistic settings** (§4.7), including larger models like `ImageNet` and ML-as-a-service platforms like Google's Vertex AI. Isotopes have 97% detection accuracy in ImageNet and 89% in Vertex.
- Finally, we **evaluate several adaptive countermeasures** that an adversarial model trainer may deploy against isotopes (§4.8). All of them either fail to disrupt isotope detection, or incur very high costs in false positives or reduced model accuracy, or both.

We view our isotope scheme as a tool for user-centric auditing of DNN models, as well as ML governance in general. The goal of *detecting use* of personal data is complementary to prior work [287, 151] that sought to make personal data *unusable*. We note that tracing of data provenance in commercial models can help enforce regulations such as EU's GDPR [82] and the "right to be forgotten." If users can detect that a given model has been trained on their data, techniques such as machine unlearning [48, 133] can be used to remove it.

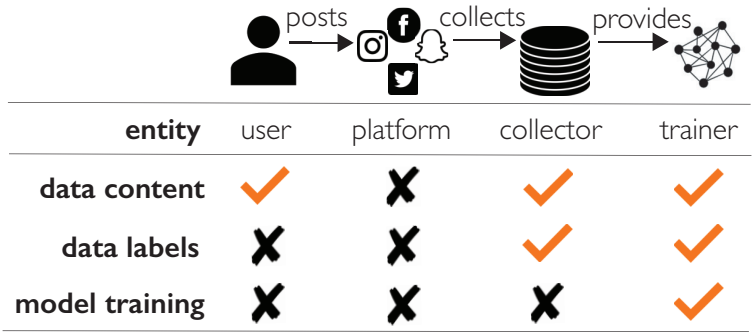


Figure 4.1: *Control over data content, data labels, and model training by different players in the ML ecosystem.*

4.2 Requirements and Prior Work

We begin by defining the problem using a concrete motivating scenario, identifying key requirements of the solution, and explaining how existing techniques fall short.

4.2.1 Defining Requirements

We illustrate the problem requirements using a simple scenario involving unwanted facial recognition. Consider a user "Taylor," who enjoys posting selfies to social media, but is concerned about "advanced facial recognition services" that can recognize millions of individuals [144, 8]. Taylor knows such services are powered by a machine learning model $\mathcal{F}_\theta$ likely trained on public data from online sources, and wants to know if their online images are used to train a model like $\mathcal{F}_\theta$. To train $\mathcal{F}_\theta$, $\mathbb{F}$ collects a dataset $\mathcal{D} = \{\mathcal{X}, \mathcal{Y}\}$, where $\mathcal{X}$ are images scraped online, e.g. from social media, and $\mathcal{Y}$ are image labels correctly assigned to images of the same person. We assume $|\mathcal{Y}| = N$, and $\mathcal{F}_\theta$ is trained using supervised learning procedure $\mathcal{L}$. $\mathcal{F}_\theta$ associate each image x with their corresponding label $y \in \mathcal{Y}$. When queried with input x, $\mathcal{F}_\theta$ returns a normalized probability vector $\mathcal{F}_\theta(x) = [0,1]^N, \sum_N \mathcal{F}_\theta(x) = 1$ over N possible labels.

Requirements of a Data Provenance Solution. In a real world setting, Taylor (e.g. user U) has very little control over the usage of their data once it is posted online (Figure 4.1). Beyond query access to model $\mathcal{F}_\theta$, they have little information on dataset $\mathcal{D}$ or internals of $\mathcal{F}_\theta$. More precisely,

Prior Work		Requirements for Data Provenance Solution				
		No knowledge of other users' data	***No change to image labels***	***No knowledge of model (while marking)***	***Query-only access to model (while testing)***	***Deployable by individuals***
No data modification	Auditing via membership inference [302, 225, 147, 186]	✓	✓	✓	✓	—
Dataset-level modifications	Dataset tracing [213]	—	✓	✓	—	—
	Radioactive data [272, 41]	✓	✓	—	—	—
	Backdoor watermark [193]	—	—	✓	✓	—
User data-level modifications	Enhancing membership/property inference [322, 65]	—	✓	✓	—	—
	Clean-label poisoning [329, 284, 160, 124]	—	✓	—	✓	—
	User-specific backdoors [149]	—	—	✓	✓	✓
	Our proposal, data isotopes	✓	✓	✓	✓	✓

Table 4.1: *Summary of prior work on ML data provenance and whether it fulfills requirements for a user-centric ML data provenance solution.* ✓ *indicates that a solution fulfills a given requirement, while — indicates it does not.*

their constraints (summarized in Table 4.1) are:

- U *does not have access to* $\mathcal{D}$, and thus it has no knowledge of other labels or data samples contributed by other users.
- U *cannot change the labels assigned to their own data* during training. In the facial recognition setting, U expects that their images will be assigned the same label/identity by $\mathbb{F}$, and has no way to alter $\mathbb{F}$'s choice.
- When U posts its images, it has no foreknowledge of the model $\mathcal{F}_\theta$ that will be trained from their data, (*e.g. parameters, labels)*. Thus it cannot rely on any such knowledge to generate any protection or marks on their images.
- At test time, U does not have cooperation from $\mathbb{F}$. Thus they have no knowledge of $\mathcal{F}_\theta$ internals and can only interact with it via a query API.
- Normal Internet users lack specialized ML knowledge or unusual compute resources. Our data provenance solution should be *deployable by individuals*, without requiring intense computation or data collection by U. For example, U lacks the skills and hardware needed to scrape large amount of training data to train additional models.

4.2.2 Existing Work on ML Data Provenance

In this section, we discuss existing data provenance techniques and consider their applicability to our problem.

Solutions that require no data modification. *Membership inference attacks* can reveal if specific data samples were present in a model's training dataset [294]. Using membership inference (MI) to audit model training data has been considered in images, speech, machine translation, and metric embedding domains [302, 225, 147, 186]. Unfortunately, MI remains unreliable for many (non-outlier) data samples, and generally requires significant data and compute to train multiple shadow models to approximate the behavior of $\mathcal{F}_\theta$ [294].

Solutions requiring dataset-level modifications. One alternative to MI is *dataset tracing*, techniques that detect when a model is trained on a specific dataset $\mathcal{D}$. Some [213] detect similarities in decision boundaries between models trained on the same dataset, while others modify portions of training data to have a detectable impact on resulting models [272, 41, 193].

There are several reasons why these *dataset level solutions* do not meet our needs. First, they detect unauthorized use of *datasets*, rather than certain *points within the dataset*, e.g. a single user's images. Thus, they assume knowledge of and control over $\mathcal{D}$ [213, 41] or at least a nontrivial proportion of $\mathcal{D}$ (e.g. 10% for realistic settings considered in [272]). This is well beyond the resources of a single U who only controls their own data. Second, some solutions [272] also assume access to a feature extractor that closely mimics the feature space of $\mathcal{F}_\theta$. Finally, techniques that use model-wide parameter shifts or representational similarities [213, 272] require either full access to $\mathcal{D}$ or the user to train a proxy model for comparison, neither is realistic for normal Internet users.

Solutions requiring user data modifications. A final set of proposals rely on changes made by U on their individual data points, rather than the whole dataset.

1) Techniques not intended for data provenance. Some solutions not designed for data prove-

Symbol	Meaning
x	Data (images, for the purposes of this paper)
x_t	Data isotope created by adding mark t to image x
U_i	Privacy-conscious user who creates isotopes x_t
$\mathcal{D}_i$	A set of images belonging to user U_i
$\mathcal{T}_i$	A set of isotope images created by user U_i, $\mathcal{T}_i \subset \mathcal{D}_i$
$\mathbb{F}$	Model trainer
$\mathcal{D}$	Dataset collected by $\mathbb{F}$, possibly containing $\mathcal{D}_i$
$\mathcal{F}_\theta$	Model trained by $\mathbb{F}$ on $\mathcal{D}$
$\mathcal{V}$	Verifier used by U_i to detect isotopes in $\mathcal{F}_\theta$

Table 4.2: *Notation used in this paper*

nance can be retooled for our setting. [322, 65] modify elements of $\mathcal{D}$ to increase the efficacy of membership inference on *specific* data points or properties. However, these methods assume U controls many elements of $\mathcal{D}$ (and their labels), and do not apply to normal users who only control their own data (and no labels). Existing work on "clean label" data poisoning and backdoors [329, 284, 160, 124, 374] could be effective, but they also require either full access to $\mathcal{F}_\theta$, $\mathcal{D}$, or a proxy model with the same feature space as $\mathcal{F}_\theta$. These are necessary to compute the poison data samples used in the attack.

2) Existing user-centric data provenance solutions. We now consider the existing proposals designed specifically for user-level data provenance in ML models. The first method "watermarks" user images by inserting backdoors—adding triggers to images and changing their label to a target label [149]. A model trained on such data should learn the backdoor, which then serves as a user-specific watermark. However, this technique requires that U both know other labels in $\mathcal{D}$ *and* control the labels assigned to their data. Neither are possible in our setting. Finally, a recent tech report [386] suggests applying color transformations to data to trace its subsequent use in models. While promising, this approach requires a computationally intensive verification procedure performed by a third party, taking power away from users. Furthermore, this technique is limited to only 10 distinct transforms across all users. Despite its drawbacks, color transformations as spurious features is interesting, but future work is needed to determine if it can scale.

4.3 Data Isotopes for Data Provenance

Clearly, there is a need for a user-centric data provenance technique that operates within the constraints defined in §4.2.1. Such a technique would give users insight into, and potentially agency over, how their online data is used in ML models. Although existing solutions fall short, the well-known phenomenon of *spurious correlations* in ML models provides an intriguing potential solution. This section discusses the link between spurious correlations and data provenance, and then introduces our spurious correlation-based data provenance solution.

4.3.1 Provenance via Spurious Correlations

U must make their data *memorable to* $\mathcal{F}_\theta$ while only *modifying their own data points*. To this end, we leverage the propensity of ML models to learn *spurious correlations* during training.

Introducing Spurious Correlations. The goal of model training is to extract general patterns from the training dataset $\mathcal{D}$. If $\mathcal{D}$ is biased or insufficiently diverse with respect to the distribution from which it is sampled, $\mathcal{F}_\theta$ can learn spurious correlations from $\mathcal{D}$, i.e., certain features not relevant to a class become predictive of that class in $\mathcal{F}_\theta$. For example, snow can become a predictive feature for the "wolf" class if training images feature wolves in the snow [364, 375].

A model can learn spurious features that appear only in a few examples [38, 368, 205, 112, 111]. Intuitively, a model cannot"tell" during training whether a rare training example is important for generalization or not; therefore, it is generally advantageous for a model to memorize rare features that appear to be characteristic of a particular class.

Data Provenance via Spurious Correlations. Spurious correlations could enable user-centric data provenance. Intuitively, if U's data introduces a spurious correlation into $\mathcal{F}_\theta$, U can detect if $\mathcal{F}_\theta$ was trained on their data by observing the effect of the correlation on $\mathcal{F}_\theta$'s classifications. Furthermore, since spurious correlations are artifacts of training data, U could simply add the spurious feature to their data, rather than using optimization procedures or changing data labels.

Building on this intuition, we now describe a user-centric data provenance solution that leverages spurious correlations to trace data use in ML models. Our solution adds spurious features to U's data to create *data isotopes*. Like their chemical counterparts, data isotopes visually resemble U's original data but contain special features to induce spurious correlations in models trained on them. If U posts isotope data online and later encounters a model $\mathcal{F}_\theta$ potentially trained on their data, U can use their knowledge of the isotope feature to determine if this is true. The term "data isotope" appeared in prior literature on dataset tracing [272], but isotopes in that sense are unusable in practical settings because they require the data owner to inspect the parameters of deployed models. This is not possible with commercial models (see §4.2.2).

4.3.2 Introducing Data Isotopes

Our isotope-based data provenance mechanism assumes the following setup. Let $U_1, U_2, \ldots U_M$ be users, each with a personal image dataset $\mathcal{D}_1, \mathcal{D}_2, \ldots \mathcal{D}_M$ that they post online. Let $\mathbb{F}$ be a model trainer who scrapes $\mathcal{D}_1, \mathcal{D}_2, \ldots \mathcal{D}_M$, and combines them into an N-class supervised-training dataset $\mathcal{D}$. $\mathbb{F}$ preprocesses $\mathcal{D}$ (deduplicates, normalizes, etc.) and assigns one of N labels $y_j \in \mathcal{Y}$ to each element $d \in \mathcal{D}$. Finally, $\mathbb{F}$ uses $\mathcal{D}$ to train a classification model $\mathcal{F}_\theta$. When queried, $\mathcal{F}_\theta$ returns a normalized probability vector over N labels. Notation is summarized in Table 4.2.

Creating isotopes. User U_i wants to trace use of their personal images, and augments it with special "isotope" images. Isotopes are created by adding a *spurious feature* t to some images $x \in \mathcal{D}_i$, creating an isotope subset $\mathcal{T}_i$. These features or *marks* are crafted to be very different from typical data features, and thus leverage a phenomenon known as "spurious correlations" [364] and the well-known propensity of models to memorize training data outliers [301, 59, 364]. We assume...

- U_i does not know a priori the labels in $\mathcal{D}$ or $\mathcal{F}_\theta$, and cannot leverage them to construct $\mathcal{T}_i$.
- Most $\mathcal{T}_i$ elements have the same label in $\mathcal{D}$. In most scenarios we consider (e.g. face recognition), this is a given since each identity has a unique label. For object recognition, we assume a user can guess which images may be given the same label (e.g. cat photos, dog photos) and

creates isotopes accordingly.

- U_i is willing to add visual distortions to images to enable tracing. This is informed by user studies that find privacy-conscious users will allow some image modifications if this enhances privacy [64]. Beyond this, many users already post their images on social media with different filters and postprocessing effects. For many, adding isotopes will not significantly degrade their image quality.
- After $\mathcal{F}_\theta$ is trained, U_i can gain black-box query access to $\mathcal{F}_\theta$, which returns a probability vector across all labels (we relax this assumption in §4.8.4).
- U_i has a small set of in-domain data $\mathcal{D}_{aux}$, $|\mathcal{D}_{aux}| << |\mathcal{D}|$ and $\mathcal{D}_{aux} \sim \mathcal{D}$. Since U_i knows the domain of their data (e.g. face images), they can collect a small set of similar data (e.g. celebrity images) to make $\mathcal{D}_{aux}$.

Isotope effect: subtle shift in label probability. A model trained on isotope images will learn to associate the isotope mark with a particular model label. At runtime, if this model encounters marked images, it will assign a slightly higher probability to the marked label for those images, relative to the probability it would assign for unmarked versions of those images. Figure 3.2 illustrates this intuition. The presence of an isotope mark on images with true label 0 will not change the model's classification decision. However, it will increase the predicted probability of the marked label (7). Although this shift may be hard to detect for a single image, analyzing the marked label probability shift for a large set of images can provide statistical proof that a model was indeed trained on isotope images.

Detecting isotopes via probability shift analysis. To detect if isotopes "marked" with the spurious feature t were present in the dataset $\mathcal{D}$ on which $\mathcal{F}_\theta$ was trained, the user performs differential analysis of $\mathcal{F}_\theta$'s behavior on inputs with and without t. Intuitively, we expect that if $\mathcal{F}_\theta$ was trained on isotopes labelled y_j, $\mathcal{F}_\theta$ will assign a higher probability to y_j for inputs (not from class y_j) with t than those without. After measuring the *probability shift* for y_j on multiple marked/unmarked image pairs, our detection algorithm uses hypothesis testing to determine if the presence of the

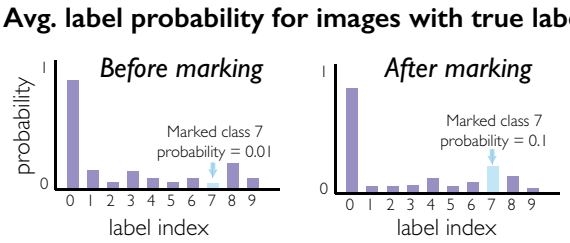


Figure 4.2: *The presence of a spurious feature "mark" on images subtly increases the probability of the marked class in a model's probability output. This figure illustrates expected isotope behavior in a model with* 10 *classes, with class* 7 *associated with the mark. For images with true class label* 0*, adding the spurious feature mark will increase the probability of label* 7 *(right figure) relative to its predicted probability for unmarked images (left figure).*

mark t in an input induces a statistically significant shift in the probability of label y_j.

Distinction from membership inference and backdoors. The key to isotopes is that when a model classifies an image with the isotope feature, it *increases the probability of a certain label*. The change can be subtle, i.e. shifting marked probability of y_j from 10^{-10} to 10^{-3}, but statistically significant.

Using changes in model outputs to infer properties of training data draws parallels in membership inference attacks [294, 302]. But isotopes is *not* membership inference, in that it does not infer the membership of a specific training input, but rather the presence of *any* data with a particular feature. This is also different from backdoor attacks [131, 346], which cause models to *misclassify* inputs containing a trigger feature. Isotopes behavior is much more subtle than backdoors, e.g. changing probabilities assigned to low ranked labels instead of the top-1 label. This makes them more difficult for model trainers to mitigate. Compared to work using backdoors for data provenance [149], isotopes do not require model training or access to feature extractors.

4.4 Data Isotopes Methodology

Data isotopes are designed for the scenario in Figure 4.3. It involves four stages: isotope creation ①, data collection ②, model training and publication ③, and isotope detection ④. In this section, we start with a brief overview of each stage, then discuss the details in §4.4.2-4.4.3.

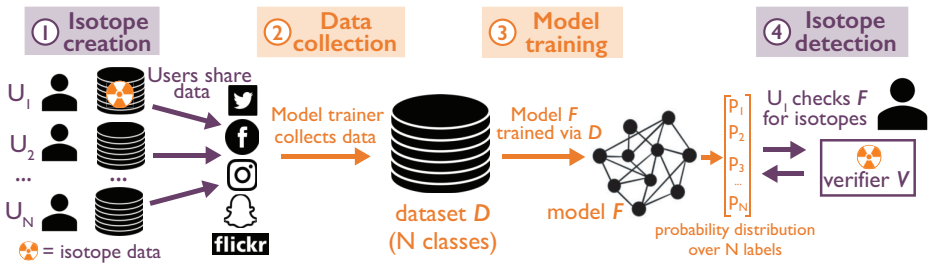


Figure 4.3: *A high-level overview of our isotopes methodology: ① User U_1 posts a set of images online, including data isotopes; ② Model trainer $\mathbb{F}$ collects these images to create a dataset $\mathcal{D}$; ③ $\mathbb{F}$ trains model $\mathcal{F}_\theta$ on $\mathcal{D}$; ④ U_1 queries $\mathcal{F}_\theta$ and uses verifier $\mathcal{V}$ to determine if their isotope images were used to train $\mathcal{F}_\theta$.*

4.4.1 Overview

Data isotopes are created by inserting a spurious feature into a subset of a model's training data for a particular label. This subset "teaches" the model to (incorrectly) associate the isotope feature with that label. Therefore, an effective isotope, created by marking images with some feature t, should have a statistically significant effect on label y_j of model $\mathcal{F}_\theta$ *if and only if* $\mathcal{F}_\theta$'s training dataset $\mathcal{D}$ contains data with mark t and label y_j. A mark t' that does not have label y_j in $\mathcal{D}$ should not have such an effect.

①: Isotope creation. User U_i creates and shares an image set $\mathcal{D}_i$. To detect if these images are later used to train a DNN model, they add an *isotope subset* $\mathcal{T}_i$, containing modified elements of $\mathcal{D}_i$, to $\mathcal{D}_i$ before sharing or uploading. $\mathcal{T}_i$ may contain isotopes with the same or different marks. Using multiple marks would let U_i create different isotopes for different subsets of their data.

②: Data collection. A model trainer $\mathbb{F}$, who wishes to train an N-class image classification model, creates their training dataset $\mathcal{D}$. $\mathbb{F}$ collects data from users $U_1, U_2, \ldots U_M$ and re-labels it with N labels to form $\mathcal{D}$. As described in §4.3, we assume a sufficient number of U_i's isotopes $\mathcal{T}_i$ with the same mark t have the same label y_j.

③: Model training and publication. $\mathbb{F}$ uses $\mathcal{D}$ to train $\mathcal{F}_\theta$, which users can then query, perhaps via an API. In the base case, we assume that $\mathbb{F}$ does not attempt to remove isotopes from $\mathcal{D}$ (in §4.8, we evaluate isotope detection and removal methods). When queried with input x, $\mathcal{F}_\theta$ returns

$\mathcal{F}_\theta(x) \in [0,1]^N$, a softmax probability distribution over N labels, where $\mathcal{F}_\theta(x)[j]$ is the probability of label y_j.

④: Isotope detection. If U_i suspects that $\mathcal{F}_\theta$ was trained on their data, they use a verifier $\mathcal{V}$, which takes in the model $\mathcal{F}_\theta$, true mark t, external mark t', label y_j, threshold λ. $\mathcal{V}$ uses data drawn from a small auxiliary dataset $\mathcal{D}_{aux} \sim \mathcal{D}$ to test for isotopes by querying $\mathcal{F}_\theta$. If $\mathcal{D}$ contains isotope data with mark t for label l, then $\mathcal{V}$ should return 1, else 0.

4.4.2 Isotope Creation

U_i creates isotopes via three steps: *mark selection*, *mark insertion*, and *data release*—see Figure 4.4.

Mark selection. Data isotopes should contain distinct, memorizable features and introduce a spurious correlation in $\mathcal{F}_\theta$, meaning that the features of mark t should not commonly appear in U_i's images. Furthermore, t should be *unique* to ensure it is distinct from other marks that may appear in $\mathcal{D}$. We discuss practical mark choices in §4.5.

Mark insertion. U_i adds t to $\mathcal{D}_i$ images to create the isotope subset $\mathcal{T}_i$. Mark insertion is parameterized by α and k, the visibility of the mark and the number of $\mathcal{D}_i$ images marked. U_i chooses k images from $\mathcal{D}_i$ and adds t to each image via operation $x \oplus (t, m, \alpha)$: $x \oplus (t, m, \alpha) = \alpha \cdot t[m] + (1-\alpha) \cdot x[m]$ where m is a mask indicating which pixels from the mark should be blended into x.

Data release. U_i releases their data (e.g., posts it online, where $\mathbb{F}$ may collect it for inclusion in $\mathcal{D}$) as $\mathcal{D}_i = \mathcal{D}_i \cup \mathcal{T}_i$ consisting of both normal images x and isotope images x_t.

4.4.3 Isotope Detection

Data collection and model training are directed by $\mathbb{F}$, and we make no assumptions about them beyond those in §4.3.2. After $\mathcal{F}_\theta$ is made public, U_i uses a verification procedure $\mathcal{V}$ to detect whether $\mathcal{F}_\theta$ strongly associates U_i's mark t with some label y_j *independently of the other image*

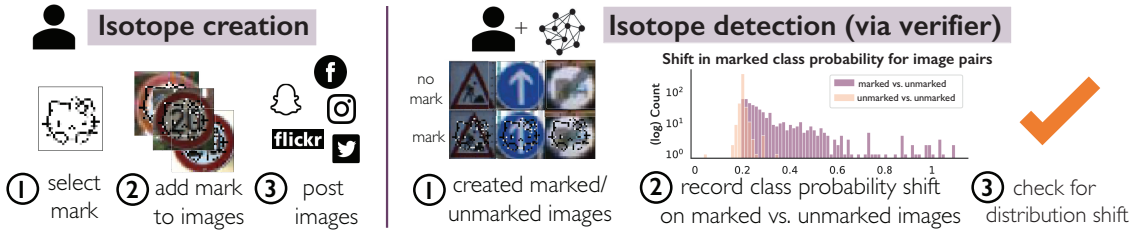


Figure 4.4: *Detailed illustration of isotope creation and detection, explained in §4.4.2 and §4.4.3.*

features. In particular, $\mathcal{F}_\theta$'s query responses should indicate a higher probability of label y_j for images marked with t than for images marked with t', a different mark *not* used in U_i's isotopes. More precisely, if $\mathcal{F}_\theta$ associates the mark t with label y_j, we expect $\mathcal{F}_\theta(x_t)[j] > \mathcal{F}_\theta(x_{t'})[j]$. Testing with images marked by t and t', rather than images marked with t and unmarked images, reduces $\mathcal{V}$'s false positives, because some external marks can induce probability shifts for label y_i relative to unmarked images.

We leverage this intuition to construct a verifier $\mathcal{V}$, which we describe informally here and formally in Algorithm 1. $\mathcal{V}$ runs paired t-tests on $\mathcal{F}_\theta$'s predicted probability of label y_j for true and incorrectly marked images. If the test result is less than a given threshold λ, $\mathcal{V}$ declares that isotopes with mark t were present in the y_j-labeled subset of $\mathcal{D}$.

Preparing for $\mathcal{V}$. Before running $\mathcal{V}$, U_i queries $\mathcal{F}_\theta$ with test images to determine if any label corresponds to a subset of their data $\mathcal{D}_i$ and may be associated with their mark t. If they find a candidate label y_j, U_i then collects a small auxiliary dataset $\mathcal{D}_{aux}$ containing images similar to those in $\mathcal{D}$ with labels $l \neq j, 0 < l < N$, $|\mathcal{D}_{aux}| << |\mathcal{D}|$. Since $\mathcal{F}_\theta$ is public, it is easy for U_i to determine what data should be in $\mathcal{D}_{aux}$. U_i does *not* include images with label y_j, since $\mathcal{V}$ detects changes in the probability of label y_j for images whose true label is different. U selects n, the number of $\mathcal{D}_{aux}$ images used by $\mathcal{V}$ in a single round; external mark t' on which to test; and a threshold λ, which $\mathcal{V}$ will use to determine if the test result is significant.

Finally, U chooses Q, the number of rounds in $\mathcal{V}$, and δ, the proportion of rounds that must produce a significant t-test in order for $\mathcal{V}$ to output 1. We use this multi-round "boosting" procedure because statistical testing is imperfect and can return false positives or false negatives.

Algorithm 1 *Verifier $\mathcal{V}$ for isotope detection.*

Input: $\mathcal{F}_\theta, \mathcal{D}_{aux}, j, n, (\lambda, \delta), (t, t', m, \alpha)$

Output: 0/1

$c = 0$

for $i \in range(Q)$ **do**

  Sample n elements from $\mathcal{D}_{aux}$, creating $\mathcal{D}_{sub}$

  $\mathbf{x} = \mathcal{D}_{sub}$

  $\mathbf{x_t} = \alpha \cdot t[m] + (1 - \alpha) \cdot \mathbf{x}$

  $\mathbf{x_{t'}} = \alpha \cdot t'[m] + (1 - \alpha) \cdot \mathbf{x}$

  $\mathbf{t_{prob}} = \mathcal{F}_\theta(\mathbf{x_t})[:, j]$

  $\mathbf{t'_{prob}} = \mathcal{F}_\theta(\mathbf{x_{t'}})[:, j]$

  $p_{mark} = ttest(\mathbf{t_{prob}}, \mathbf{t'_{prob}})$

  if $p_{mark} < \lambda$ **then**

    c += 1

  end if

end for

if $(c/N) > \delta$ **then**

  Return 1

end if

Return: 0

Running $\mathcal{V}$. Using these parameters and the mark parameters (t, t', m, α), U runs $\mathcal{V}$. $\mathcal{V}$ takes n images from $\mathcal{D}_{aux}$ and marks each image, once with t and once with t'. Then, $\mathcal{V}$ submits $(x_t, x_{t'})$ image pairs to $\mathcal{D}$ and computes $\mathbf{t_{prob}} = \mathcal{F}_\theta(x_t)[:, j]$ and $\mathbf{t'_{prob}} = \mathcal{F}_\theta(x_{t'})[:, j]$. Finally, $\mathcal{V}$ runs a paired one-sided Student's t-test, which tests for differences in distribution means, to compare the mean shifts in the two sets. The null hypothesis is that the mean of the label y_j's probability distribution is the same for both marks, and the alternative is that the mean is greater for images

with mark t. If the p-value of the test is below λ for $\delta \cdot Q$ boosting rounds, $\mathcal{V}$ concludes that $\mathcal{D}$ must have included images with mark t for label y_j and returns 1, else 0. A discussion of λ, δ, and Q choices is in §4.5.1.

Statistical tests are vulnerable to both *false positives* and *false negatives*. In our context, a false positive occurs when the test returns a statistically significant result for isotopes with mark t' when t' isotopes were *not* present in the training data associated with y_j. A false negative occurs when the test returns a negative result for isotopes with mark t that were present in the training data. We measure errors of both types in our evaluation (§4.5).

4.4.4 Advanced Isotope Scenarios

The basic isotope scenario assumes a single mark t associated with a single label y_j in $\mathcal{F}_\theta$, but numerous other settings are possible.

Multiple isotope marks in different classes. When multiple marks are present in different classes, each mark t_j with label y_j must be both *detectable* by $\mathcal{V}$ and *distinguishable* from other marks t_k for other classes $y_k, k \neq j$. To ensure both, in this setting we run $\mathcal{V}$ using *two* marks that are both present in $\mathcal{D}$, t_j and t_k, rather than true mark t and external mark t'. $\mathcal{V}$ checks that only mark t_j induces a statistically significant probability shift for class y_j, and vice versa. Although U_i knows only their mark, a third party with knowledge of all marks could run this test. When we evaluate this scenario in §4.5.3, we assume this third party exists.

Multiple isotope marks in the same class. When multiple marks are associated with a single label y_j, it is possible to *detect* them via $\mathcal{V}$ but not to *distinguish* them. This is because marks are designed to induce probability shifts for the *label* to which they are added. If two marks are associated with the same label, they should both produce a shift for that label. We evaluate this setting in §4.5.3.

Ranks instead of probabilities. In §4.8, we explore the setting where $\mathcal{F}_\theta$ returns only the top-K ranked classes, rather than a probability distribution over all classes.

4.5 Evaluating Data Isotopes

Our baseline evaluation focuses on fundamental questions about isotope potency. First, does the isotope intuition described in §4.3.2 – in which a single class in $\mathcal{D}$ contains isotope data and causes a single label's probability to increase – hold up across different task and model settings (§4.5.2)? If so, do isotopes remain potent when $\mathcal{D}$ (§4.5.3) contains multiple isotopes sets? For both settings, we measure the distortion necessary to create potent isotopes and evaluate robustness to false positives. Last, we explore how isotopes *scale* (§4.5.4) and consider isotope uniqueness and their effect on model accuracy.

4.5.1 Methodology

Tasks. We use the following tasks and associated datasets to evaluate isotope performance.

- `GTSRB` is a traffic-sign recognition task with $50,000$ images of 43 different signs [148]. This task is commonly used as a benchmark for computer vision settings.
- `CIFAR100` is an object recognition task with $60,000$ images and 100 classes [173]. This task allows us to explore mark efficacy in an object recognition setting.
- `PubFig` is a facial recognition task whose associated dataset contains over $50,000$ images of 200 people [176]. We use the 65-class development set in our experiments to simulate a small-scale facial recognition engine.
- `FaceScrub` is a large-scale facial recognition task with a $100,000+$ image dataset of 530 people [234]. This task emulates a mid-size real-world facial recognition engine, enabling us to explore marking in a realistic setting.

Model Architectures and Training. We use different model architectures and training procedures for each task. The training settings for each dataset are in Table 4.3. For most tasks, models are trained from scratch. The exception is `PubFig`, due to its small size, which we train via transfer learning from models pre-trained on the CASIA-Webface dataset [369]. All experiments are

Figure 4.5: *Different marks used in our experiments.*

Task	Classes	Model	Loss	Training setting
GTSRB	43	Simple	Cross-entropy	Adam(lr=0.0001, epochs=20, batch size=512)
CIFAR100	100	ResNet18	Cross-entropy	SGD(lr=0.5, scheduler=step, epochs=72, batch size=512)
PubFig	65	SphereFace (pretrained)	Angle	Adam(lr=0.001, epochs=25, batch size=128)
Scrub	530	ResNet50	Focal	Adam(lr=0.001, scheduler=cyclic, epochs=16, batch size=128)
ImageNet	1000	ResNet50	Cross-entropy	Adam(lr=1.7, scheduler=step, epochs=18, batch size=512)

Table 4.3: *Model training details for each task.*

run on our local servers using 1 NVIDIA GPU. For `CIFAR100`, we use the ffcv library to expedite training [182].

Marks. Since we test isotopes in an image classification setting, we use pixel patterns and images as the isotope mark t, as shown in Figure 4.5. The pixel patterns, "pixel square" and "random pixels," only flip certain image pixels and can vary in location and size within an image. In contrast, the "Hello Kitty" and "ImageNet blend" marks are images blended into $\mathcal{D}$ images. For the ImageNet blend mark, we randomly select images from ImageNet [91]. When we run $\mathcal{V}$, we choose an external mark t' similar to the true mark t —if t is an Imagenet blend mark, t' is a different Imagenet blend mark —to measure the most realistic false positive scenario.

Verifier parameters. In our baseline experiments, we use t-tests with $n = 250$ test images, for $\mathcal{V}$ and $\mathcal{V}_D$, and $\mathcal{D}_{aux}$ is composed of elements from the test dataset of each task. We fix the proportion of necessary positive tests for $\mathcal{V}$ to return 1 at $\delta = 0.6$, because this ensures that a majority of $\mathcal{V}$'s t-test are below λ, and use $Q = 5$ boosting rounds (see below for additional details on Q). We vary λ to compute the true positive rate at different false positive rates, and use the same α for mark insertion and testing.

$\mathcal{V}$ baseline performance and boosting. In our experiments, we run $\mathcal{V}$ using *boosting*, i.e. multiple runs of the t-test, in order to minimize randomness. Here, we explore the effect of Q,

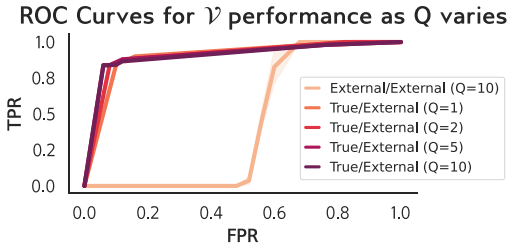


Figure 4.6: *Comparison of $\mathcal{V}$ performance for different Q values and on paired external marks.*

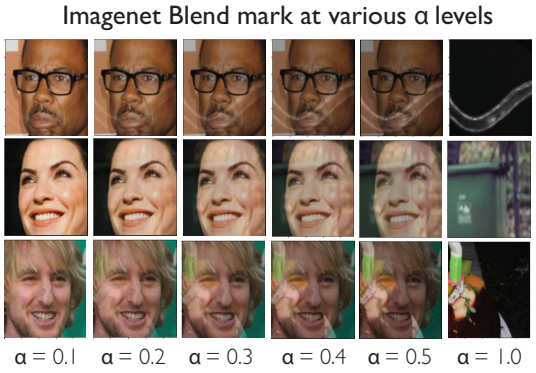


Figure 4.7: *Visibility of ImageNet blend mark increases with α.*

the number of boosting rounds, on the TPR/FPR of $\mathcal{V}$. The goal is to use the minimum number of boosting rounds that produce a stable $\mathcal{V}$ performance, to minimize the cost of verification. We also explore the baseline TPR/FPR for $\mathcal{V}$ when it is run on t'_1 and t'_2, two external marks. $\mathcal{V}$ should have roughly random performance (TPR $\approx$ FPR) in this setting.

To test this, we evaluate a `CIFAR100` model with 30 marked classes, $\alpha = 0.5$, $p = 0.1$. We run $\mathcal{V}$ using different Q values on both true/external mark pairs (as typically used in $\mathcal{V}$) and external/external mark pairs (for baseline performance calibration). As Figure 4.6 shows, $\mathcal{V}$'s performance slightly improves when going from 1 to 5 boosting rounds, but increasing from 5 to 10 does not significantly improve performance. Thus, in our §4.5-§4.8 experiments, we use $Q = 5$. As expected, results for the external/external $\mathcal{V}$ tests are random, even when $Q = 10$.

Metrics. We report $\mathcal{V}$'s *true positive rate* (TPR), $\mathcal{V}_T$, the proportion of times $\mathcal{V}$ returns 1 when comparing a true tag t to an external tag t' for a given (λ, δ, Q) setting. We also report $\mathcal{V}$'s *false positive rate* (FPR), $\mathcal{V}_F$, computed by inverting the order of tags presented to $\mathcal{V}$ and measuring the

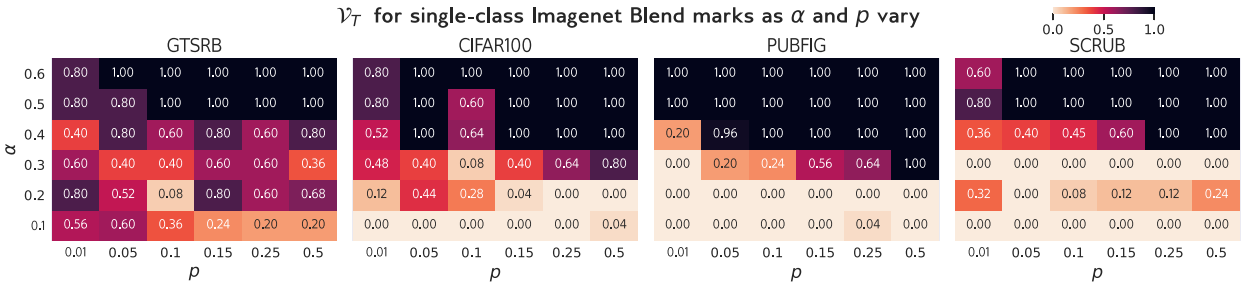


Figure 4.8: *Average $\mathcal{V}_T$ values at $\lambda = 0.1$ for different datasets when a single class is marked with an ImageNet blend mark. For most datasets, marking is effective when $\alpha \geq 0.4$ and $p \geq 0.1$.*

proportion of times $\mathcal{V}$ returns 1 when comparing mark t' which is not in $\mathcal{F}_\theta$ to t (i.e. t' induces a larger shift than t). In most experiments, we report the TPR/FPR at $\lambda = 0.1$, a common threshold for statistical significance.

When we evaluate the setting where isotopes are present in multiple $\mathcal{D}$ classes, we also report the *distinguisher true positive rate* $\mathcal{V}_{D_T}$, the proportion of times $\mathcal{V}_D$ successfully distinguishes between two marks present in $\mathcal{F}_\theta$ for a given (λ, δ, Q) setting.

Experiment Overview. Unless otherwise noted, results are averaged over 5 runs per experiment, each run using different isotope classes. We also measure model accuracy, which is largely unaffected by isotopes (see §4.5.4). To show that isotopes are robust to typical data preprocessing techniques, in all experiments in this section we use data augmentations during training, including random flipping/cropping/rotation and color normalization.

4.5.2 *Single isotope subset in $\mathcal{D}$*

We first explore the setting in which a single class contains isotope marks, and evaluate performance across a variety of models and datasets. We explore how marks perform as α and p vary for different tasks.

Performance of different marks Using the parameters and training settings described in §4.5.1, we train `CIFAR100` models with isotopes created using the four marks shown in Figure 4.5. To explore how mark settings impact performance, we vary α from 0.1 to 0.6 (see Figure 4.7) and p from 0.01 (e.g. 1% of data marked) to 0.5. Figure 4.9 reports the average $\mathcal{V}_T$ for each setting.

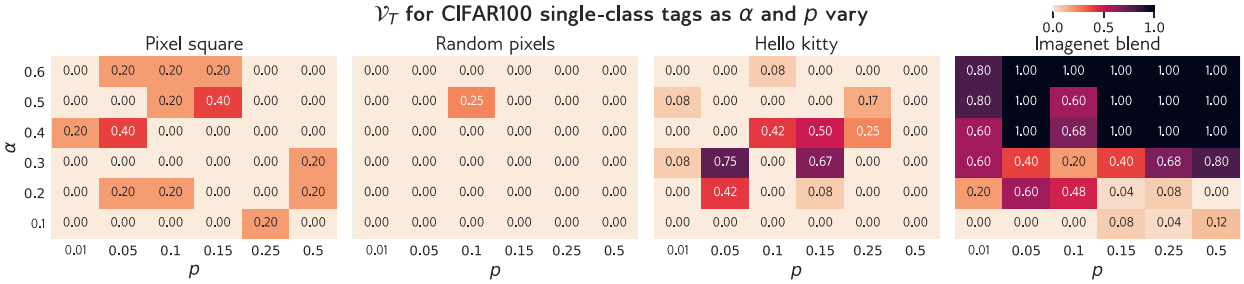


Figure 4.9: *Average $\mathcal{V}_T$ values for different marks in a* `CIFAR100` *model. Marks that introduce stronger features into images (like Hello Kitty and Imagenet Blend) perform much better.*

Overall, we find only Imagenet blend marks are consistently detectable. This indicates that marks with more unique and diverse features make marks better isotope candidates, and once such a mark is visible and frequent enough in a user's data, it can be detected.

The pixels square, random pixels, and Hello Kitty marks *can* induce probability shifts for classes to which they are added, as illustrated in Figure 4.3. However, these marks do not produce *strong enough* probability shifts to be robust to the false positives test $\mathcal{V}$ runs —e.g. comparing the true mark to some external mark. This false positives test is necessary to make isotopes practically useful, and when we employ this, we find that the pixels square, random pixels, and Hello Kitty marks are less effective. Thus, we use the Imagenet blend mark in the rest of our experiments.

Performance across datasets. To explore how mark settings impact Imagenet mark performance, we vary α from 0.1 to 0.6 (see Figure 4.7) and p from 0.01 (e.g. 1% of data marked) to 0.5. Figure 4.8 reports the average $\mathcal{V}_T$ for each setting at $\lambda = 0.1$. From this we see that when a single dataset class contains an Imagenet blend mark, isotopes are highly effective, even in large datasets like `Scrub`. Larger datasets require a slightly higher α/p combination (e.g. $\alpha \geq 0.4$ and $p \geq 0.15$ for `Scrub`) before marks become detectable. Overall, in the single mark setting, marks are easy to detect even when only a small portion of user images are faintly marked.

Robustness to false positives. We evaluate $\mathcal{V}_F$ for all datasets with fixed $\alpha = 0.4$ and $p = 0.25$. For all datasets except `GTSRB`, $\mathcal{V}_F = 0$ and $\mathcal{V}_T = 1.0$ when $\lambda = 0.1$. `GTSRB` has $\mathcal{V}_F = 0.4$ at this setting, likely because its model architecture is simple and potentially less amenable to memorization [273].

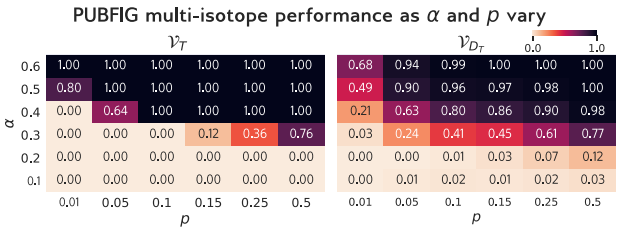


Figure 4.10: *Ablation over α and p for a* `PubFig` *model with* 20 *marked classes, using $\lambda = 0.1$ for $\mathcal{V}$ and $\mathcal{V}_D$.*

Classes marked	`GTSRB` (43 classes)		`CIFAR100` (100 classes)		`PUBFIG` (65 classes)		`SCRUB` (530 classes)	
	$\mathcal{V}_T$	$\mathcal{V}_{D_T}$	$\mathcal{V}_T$	$\mathcal{V}_{D_T}$	$\mathcal{V}_T$	$\mathcal{V}_{D_T}$	$\mathcal{V}_T$	$\mathcal{V}_{D_T}$
5%	1.0	0.20	1.0	0.99	1.0	1.0	0.88	0.73
10%	1.0	0.65	1.0	0.98	0.64	0.70	1.0	0.72
20%	1.0	0.71	1.0	0.96	0.98	1.0	0.86	0.73
30%	0.75	0.72	1.0	0.98	1.0	1.0	0.85	0.72
40%	0.72	0.68	0.99	0.95	1.0	0.79	1.0	0.75
50%	1.0	0.72	1.0	0.97	1.0	0.73	1.0	0.70

Figure 4.11: *$\mathcal{V}_T$ and $\mathcal{V}_{D_T}$ for multi-mark settings with up to 50% of classes marked. We add marks using $\alpha = 0.4$ and $p = 0.1$ for all datasets, and we evaluate using $\lambda = 0.1$.*

4.5.3 Multiple isotope subsets in $\mathcal{D}$

Next, we consider the possibility that $\mathcal{D}$ may contain multiple isotope subsets, each with a different mark. This corresponds to the real-world scenario of multiple users adding marks to their data, all of which end up in $\mathcal{D}$. Given the size of today's ML datasets and models, this scenario is not-unlikely, especially if data isotopes become popular as a provenance-tracking mechanism. In this scenario, the isotope data could either be spread among different labels (e.g. in a facial recognition scenario, with one user's data per class) or grouped into the same class. We evaluate isotope performance in both settings. In most experiments, we use Imagenet blend tags with $\alpha = 0.4$. Examples of tags with this setting can be found in Figure 4.7.

Multiple isotopes in different classes – baseline. We first evaluate isotope performance when multiple classes in $\mathcal{D}$ are associated with *distinct* isotope subsets. Since this scenario most closely corresponds with a facial recognition scenario, we perform a baseline evaluation of this setting using the `PubFig` with ImageNet blend marks, $\alpha = 0.4$ and $p = 0.1$. We use $\mathcal{V}$ and $\mathcal{V}_D$ with $\lambda = 0.1$ to assess mark performance, and using 5 external marks per true mark to compute $\mathcal{V}_T$ and $\mathcal{V}_F$. As Table 4.11 demonstrates, marks remain detectable and distinguishable for `PubFig` when up to 50% of classes contain isotopes. For all settings, $\mathcal{V}_F = 0$ and $\mathcal{V}_T \geq 0.98$ when $\lambda = 0.1$, and model accuracy is unchanged from baseline performance (86%).

Having established that isotopes perform well when multiple isotope subsets are present in

Marks per class	2	3	4	5	6
$\mathcal{V}_\mathbf{T}$	1.0	0.8	0.8	1.0	1.0
$\mathcal{V}_\mathbf{F}$	0.0	0.12	0.0	0.0	0.0

Table 4.4: *TPR/FPR for multiple marks per class at* $\lambda = 0.1$ *and* $\delta = 0.6$*. In all cases,* $\mathcal{V}_T > 0.8$ *and* $\mathcal{V}_F < 0.12$*, even with up to* 6 *marks per class.*

`PubFig`, we measure how α and p affect overall performance. We run experiments on `PubFig` models with 20 classes marked and vary α/p. Figure 4.10 shows that the trend for $\mathcal{V}_T$ and $\mathcal{V}_{D_T}$ remains similar to the single mark case: when $\alpha \geq 0.4$ and $p \geq 0.1$, $\mathcal{V}_T = 1.0$, $\mathcal{V}_{D_T} \geq 0.8$ and $\mathcal{V}_F = 0$ at $\lambda = 0.1$.

Multiple isotopes in different classes – performance across datasets. Next, we confirm the result observed on `PubFig` extends to other datasets. We vary the percent of classes marked from 5% to 50%, fix $\alpha = 0.4$ for all datasets, and test if ImageNet blend marks remain detectable and distinguishable in models trained on different tasks. We report $\mathcal{V}_T$ and $\mathcal{V}_{D_T}$ in Table 4.11, using $\lambda = 0.1$ as before. Since $\mathcal{V}_D$ runs in $\mathcal{O}(n^2)$, we reduce computation time when the number of marked classes exceeds 25 by randomly selecting 25 marks on which to run $\mathcal{V}_D$, which yields 25^2 comparisons max instead of $\binom{n}{2}$. As Table 4.11 shows, both $\mathcal{V}$ and $\mathcal{V}_{D_T}$ are high across the board. For all results shown, $\mathcal{V}_F < 0.05$ at $\lambda = 0.1$. Furthermore, $\mathcal{F}_\theta$ accuracy remains stable in all settings, with $< 1\%$ change from the baseline accuracy level. Consequently, we conclude that isotopes remain potent when multiple dataset classes are marked.

Multiple isotopes in a single class. Finally, we examine what happens when multiple users insert marks within a single class. The goal is that *each* mark should be learned as associated with this class, e.g. the presence of multiple marks should not prevent the learning of a particular mark. Although we cannot measure mark distinguishability in this setting (since marks should induce a *class level* probability shift, see §4.4), we can evaluate that each mark in a multi-mark-per-class setting can be detected.

We test this by training `CIFAR100` models with up to 6 marks per class, $\alpha = 0.4$, $p = 0.05$, see Table 4.4. In this setting, $p = 0.05$ means that each mark controls 5% of the marked class. Even

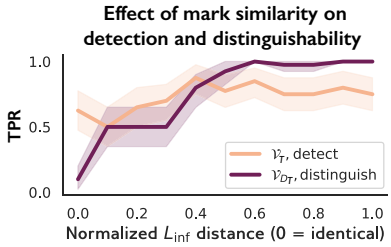


Figure 4.12: *When marks have a normalized L_{inf} distance of < 0.2, mark distinguishability sharply decreases.*

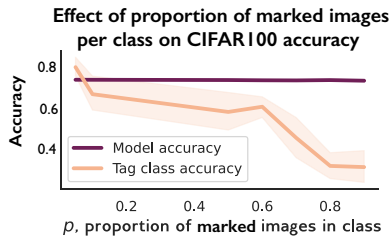


Figure 4.13: *As the proportion of marked images grows, model accuracy remains overall unaffected, but marked class accuracy decreases.*

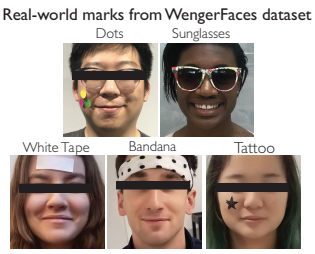


Figure 4.14: *Examples of physical world marks from the `WengerFaces` dataset used in our experiments.*

with up to 6 marks per class, marks are detectable with $\mathcal{V}_T \geq 0.8$ and $\mathcal{V}_F \leq 0.12$ for $\lambda = 0.1$.

4.5.4 Scaling isotopes

Having established the baseline performance of isotopes in single and multi-mark settings, we now consider issues related to *scaling* isotopes. First, we address is how *similar* marks can be while remaining detectable, and then test how the number of marked images in $\mathcal{D}$ affects $\mathcal{F}_\theta$ *accuracy*.

Mark distinguishability. We begin by evaluating how *similar* two marks can be before they become indistinguishable in a multi-mark setting, when marks are associated with different classes. The goal here is to estimate the space of images from which marks can be chosen. If two marks are similar in pixel space but still detectable by $\mathcal{V}$, then we know there is a large universe of marks to choose from. If a significant number of pixels must be different for both marks to be detectable, the universe of viable marks is limited.

To test this, we craft two marks with controlled, normalized L_{inf} distance by blending one mark into the other at different blend ratios. We then insert both marks into a `CIFAR100` dataset with $\alpha = 0.4$, associating each with a different class. We train $\mathcal{F}_\theta$ using this dataset and run $\mathcal{V}$ and $\mathcal{V}_D$ on the two marks, with $\lambda = 0.01$. As Figure 4.12 shows, when the marks have normalized L_{inf} distance ≥ 0.4, the marks remain both detectable and distinguishable. Practically, this means that images which share up to 60% of their pixels remain distinguishable as marks.

$\mathcal{F}_\theta$ accuracy. Finally, we explore how much data can be marked before model accuracy on normal inputs starts to degrade. We mark a single class in `CIFAR100` with an increasing fraction of isotopes (up to $p = 0.9$, with $\alpha = 0.4$). Figure 4.13 shows that the accuracy for the marked class drops off rapidly once $p \geq 0.6$, although overall model accuracy remains high. At this point, marks make up the majority of the class, so the model begins learn them as core features instead of the true task.

4.6 Physical Objects as Marks

While our proposed pixel-based marks are effective in numerous settings, they have an obvious downside: they require that U can edit images after they are taken but before they are shared publicly. Depending on how $\mathbb{F}$ sources their data, this assumption may not be realistic. If, for example, $\mathbb{F}$ obtains training data from public surveillance footage to train a facial recognition model, U is out of luck. Since in this scenario U's image is captured in real-time and shared without their knowledge, U cannot mark this data using our proposed methods. Despite this obstacle, U may wish to test if images taken in a certain setting are included in $\mathbb{F}$'s model, and we propose *physical marks* as a way to do so.

Physical marks are unique physical objects present in images *at the time of their creation*. The inclusion of these objects in images enables users to create isotopes even when they cannot control which images are taken. In the facial recognition scenario mentioned above, simply wearing a physical object, such as a certain pair of sunglasses or scarf, would ensure that any images taken while the user is wearing that object could have a detectable mark. Here, we evaluate physical marks in a facial recognition scenario.

4.6.1 Methodology

Physical mark images. We use images from the `WengerFaces` dataset [346] to create physical marks and test their performance. This dataset contains headshots of 10 people. All images are unobstructed and well-lit, and in a subset of the images, subsets wear different physical objects on

or around their faces. We use these physical objects – sunglasses, a scarf, tattoos, dots, and white tape on the forehead (see Figure 4.14) – as marks.

Training dataset. To construct the marked dataset, we add clean (e.g. unmarked) images from `WengerFaces` to the `Scrub` dataset, forming a new 540 class dataset. We designate a class from `WengerFaces` as belonging to U_i and add the physical mark images to that class. We fix the proportion of images marked at $p = 0.25$, and since the number of clean images for each `WengerFaces` user ranges from 20 to 45, this means we use between 5 and 11 marked images per class. The α parameter is not meaningful in this setting, so we ignore it. We train a model on this enhanced dataset using the `Scrub` described in §4.4.

Mark detection. To evaluate physical mark performance, we run $\mathcal{V}$ using the other physical objects as external marks. As in the pixel-based setting, this tests for tag uniqueness by measuring if other physical objects induce a similar shift in $\mathcal{F}_\theta$. Since this testing method involves different sets of images and different marks, instead of the same set of images with different marks, a paired t-test is not appropriate. Instead, $\mathcal{V}$ uses an unpaired, 1-sided t-test to check for a statistically significant difference in marked class probability for these two groups.

4.6.2 Results

Mark	Dots	Sunglasses	Tape	Bandana	Tattoo
$\mathcal{V}_T$	0.5	0.9	0.45	0.0	0.25
$\mathcal{V}_F$	0.2	0.0	0.30	1.0	0.75

Table 4.5: *$\mathcal{V}$ can detect some physical marks when $\lambda = 0.4$.*

We test each mark 5 times, training a separate model and choosing a different class to mark each time. For each mark, we evaluate $\mathcal{V}$ using the 4 other objects as external marks. As reported in Table 4.5, larger, more distinct on-face objects like sunglasses, dots, and white tape have the highest success rate, although we do have to increase λ to detect them. However, objects that are located off the face or are smaller (bandana, tattoos) make less effective marks. Normal model accuracy remains high, 99% on average.

These results demonstrate that sufficiently unique, on-face physical marks could be used to create effective data isotopes in a facial recognition setting even when users do not control image capture or distribution. They can help detect uses of images in which users appear but did not create or post online.

4.7 Isotopes in Real-World Settings

Real-world ML models are trained on users' data using a wide variety of training pipelines, pre-processing methods, etc., thus it is not feasible to evaluate isotopes in every possible scenario. We focus on showing that isotopes remain effective in several practical settings: larger models; opaque, ML-as-a-service model-training APIs; and transfer learning. We also experiment with isotope performance in commercial facial recognition (FR) platforms. Isotopes do work in commercial FR systems, but due to the difference in setting for these experiments (i.e. feature matching instead of training models from scratch) and limited space, we include these results in Appendix 4.7.4.

4.7.1 Larger models

The largest model we consider in our baseline evaluation is `Scrub`, which has 530 classes. To evaluate isotopes in even larger models, we perform a small set of experiments on the `ImageNet` dataset [91], which has 1000 classes and contains 1.7 million images (training details are in Table 4.3 in Appendix). We use ImageNet blend marks to create isotopes with $\alpha = 0.4$ and $p = 0.1$, and assume that each isotope subset is assigned to a different class (since this represents the most difficult setting). Our trained model has 72% Top-1 accuracy. Testing with up to 100 `ImageNet` classes marked, we find that, on average, $\mathcal{V}_T = 0.96$, $\mathcal{V}_F = 0.02$, and $\mathcal{V}_{D_T} = 0.99$ for $\lambda = 0.1$ and $\delta = 0.6$. Isotopes remain potent even in large models.

Setting	$\mathcal{F}_\theta$ acc.	$\mathcal{V}_T$	$\mathcal{V}_F$	$\mathcal{V}_{D_T}$
Single marked class	0.64	1.0	0.0	—
20 marked classes	0.65	0.89	0.07	0.84

Table 4.6: *Isotopes remain detectable in models via the Google Cloud ML API, both in the single and multiple (25) marked class setting.*

4.7.2 ML-as-a-Service APIs

Next, we test how marking performs when we train models using API endpoints rather than our local servers. We test this by training `CIFAR100` models using Google Vertex AI [1] with both 1 and 20 marked classes, $\alpha = 0.4$, $p = 0.1$. These experiments are done in a pure black-box manner: we have no knowledge nor control about what data transformations, learning algorithms, or model architectures are used. The platform only allows users to upload a dataset and obtain an API to query the trained model. The resulting models achieve $64 - 65\%$ Top-1 accuracy. As Table 4.6 shows, $\mathcal{V}_T = 1.0$, $\mathcal{V}_F = 0.0$ under the single marked class setting and $\mathcal{V}_T = 0.89$, $\mathcal{V}_F = 0.07, \mathcal{V}_{D_T} = 0.84$ under the 20 marked classes setting. Our isotopes are generic and still effective when evaluating against up-to-date popular ML-as-a-Service APIs.

4.7.3 Transfer learning

Finally, we consider isotope robustness when $\mathbb{F}$ uses transfer learning, a technique commonly used for increasing model performance when only limited training data or compute power is available [256, 314]. Transfer learning confers knowledge from a teacher model trained on a domain similar to $\mathcal{D}$ by reconfiguring and retraining its last few layers on $\mathcal{D}$. The intuition is that earlier (lower) model layers typically learn more generic image features, while later (higher) layers learn task-specific features, so retraining the last layers is sufficient to adapt the teacher to the target task.

Since isotope marks are features in images, their performance may be affected by transfer learning, particularly if mark features are learned in the early layers. We evaluate the effect of

1. `https://cloud.google.com/vertex-ai`

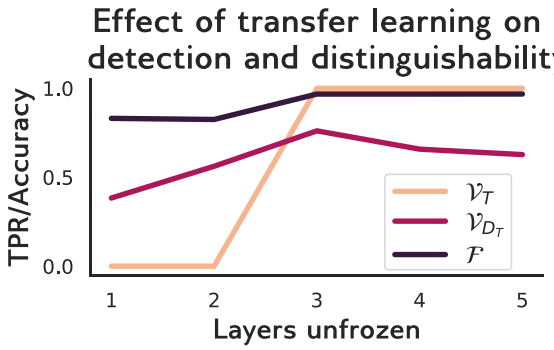


Figure 4.15: *Isotopes remain detectable in a transfer learning setting when at least* 3 *layers are unfrozen during training.*

	set1	set2	set3
% with match	95%	94%	80%
Avg. mark match rank	1.0	1.0	4.8
Avg. true match rank	1.0	1.0	-

Table 4.7: *Results from isotope detection in Amazon Rekognition. For* `set1` *and* `set2`*, the true match is always the top match. For unenrolled isotope images (*`3`*), isotope images with the same mark appear in the top* 5 *hits.*

transfer learning on isotopes using the `Scrub` dataset with 25 classes marked, $\alpha = 0.4$ and $p = 0.1$. We use a SphereFace model pretrained on the WebFace dataset as the teacher model, and train using the settings shown for `PubFig` in Table 4.3. We vary the number of unfrozen layers from 1 to 5 and report $\mathcal{V}_T$ and $\mathcal{V}_{D_T}$ in Figure 4.15.

Model accuracy is highest when 3 layers are unfrozen, and in this setting, $\mathcal{V}_T = 1.0$ and $\mathcal{V}_F = 0$ for $\lambda = 0.1$. $\mathcal{V}_{D_T}$ is slightly lower across the board, but this mirrors the trend in $\mathcal{V}_{D_T}$ observed in Table 4.11. Since $\mathcal{V}_T$ trends with model accuracy during transfer learning, we conclude that isotopes remain effective in this setting.

4.7.4 Isotopes in facial recognition engines

Testing isotopes in commercial FR systems requires some modifications to the detection algorithm. Today, these systems work by matching query images to a reference database via *feature space similarity*, as opposed to directly applying a trained ML model. Standard approaches involve measuring L_2 similarity between the query and reference images in the feature space of a trained

DNN [92, 342, 223, 281]. Reference images that are similar (or identical) to a queried image are returned as the "top match". We leverage this fact to detect isotopes in commercial FR databases.

We run experiments on Amazon Rekognition, a popular facial recognition engine that allows users to build a reference image database and submit new images to the database via an API [132]. Rekognition does not disclose how images are processed in their system, what DNN is used to produce features, or how feature space matching is performed.

We enroll 100 people from the `Scrub` dataset in a Rekognition database using 100 images/person. We select 10 `Scrub` classes for isotope testing, 5 men and 5 women (4 Black, 6 Caucasian). For each, we enroll 5 different images with the same mark (`set1`) and 5 images that are identical but have different marks (`set2`). At test time, we set the confidence threshold (the minimum similarity for a reference image to be returned as a match) at 95 and query `set1`, `set2`, and `set3`, which contains new images with the `set1` mark. All marks are ImageNet blend marks with $\alpha = 0.4$. In Table 4.7, we report the proportion of images for which any isotope match was returned, the average rank (1 = best) of the first isotope image in the match set, and the average rank of the true match for `set1`, `set2`.

As Table 4.7 shows, `set1` and `set2` always have the true enrolled image as their top match, i.e., we perfectly detect isotopes in the database. Interestingly, `set3` images, which have the same mark as `set1` but are not enrolled, have an isotope image appear in the top 5 matches on average, even though isotopes are only 10% of the enrolled set, i.e., a marked query image often draws out *other* isotopes with the same mark enrolled in the database.

Discussion. Isotope detection described above exploits the fact that FR engines are very good at matching identical images. Thus, if a user knows what images they posted online and where, they can determine if a particular source was included in an FR database by querying the corresponding FR engine with an image from that source. Isotopes are not strictly necessary for this sort of auditing: if an exact image is in the reference database, it will typically be the top match. Isotopes can still be useful for users to quickly "sort" which site the images came from, perhaps by posting

identical images with different marks on different sites.

4.8 Robustness to Adaptive Countermeasures

A model trainer may try to prevent isotopes from being used effectively, perhaps to hide their use of private data during model training. We believe the two main ways to attack isotopes are to *detect* them or *disrupt* them.

We draw inspiration from existing defenses against poisoning, backdoor, and membership inference attacks, which are all related to isotopes (see §4.2), to identify techniques that could detect or disrupt isotopes. For example, $\mathbb{F}$ could try to detect isotopes using existing methods for spurious correlation detection [297, 227] or by analyzing $\mathcal{F}_\theta$'s feature space after training to detect changes induced by isotopes [323, 66, 140, 283, 316, 252]. To disrupt isotopes, $\mathbb{F}$ could employ adversarial augmentations during $\mathcal{F}_\theta$ training [293, 257], modify the model's outputs to decrease $\mathcal{V}$'s accuracy [294, 164], or selectively retrain $\mathcal{F}_\theta$ to make it forget isotope features [187].

Here, we evaluate the efficacy of five potential anti-isotope countermeasures and discuss their *cost*. If a countermeasure incurs a sufficiently high cost, the model trainer may decide not to deploy it. Methods to detect isotopes could incur a false positive cost (relevant to §4.8.1 and §4.8.2), if isotope detection methods require high FPR to achieve high TPR. Methods to disrupt isotopes may have a model performance cost (relevant to §4.8.3-4.8.5), if accuracy must be sacrificed to disrupt isotopes. Unless otherwise noted, we run experiments in this section using `CIFAR100` models with 25 marked classes, Imagenet marks, $\alpha = 0.4$, $p = 0.1$.

4.8.1 Spurious correlation detection

Isotope marking would be ineffective if $\mathbb{F}$ could detect and filter out isotope images in $\mathcal{D}$. Existing literature has shown it is possible to detect spurious correlation in datasets [297, 227]. Since isotopes are inspired by the spurious correlation phenomenon, we test whether [297], a state-of-the-art spurious correlation detection method, can detect isoptoes in $\mathcal{D}$. [297] inspects feature maps

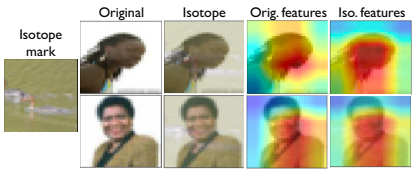


Figure 4.16: *The state-of-the-art spurious correlation detection method we test cannot flag isotopes with reasonable settings like* $p = 0.1$ *and* $\alpha = 0.4$.

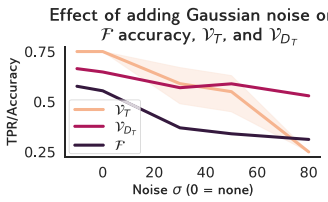


Figure 4.17: *Adding Gaussian noise with* $\mu = 0$ *and increasing* σ *to* $\mathcal{D}$ *images degrades* $\mathcal{F}_\theta$ *accuracy faster than* $\mathcal{V}_T$ *or* $\mathcal{V}_{D_T}$.

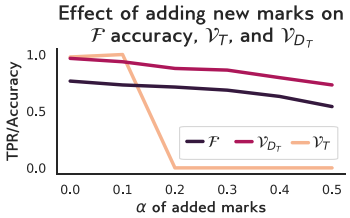


Figure 4.18: *Adding new marks to* $\mathcal{D}$ *images decrease* $\mathcal{F}_\theta$ *accuracy more than* $\mathcal{V}_T$ *or* $\mathcal{V}_{D_T}$.

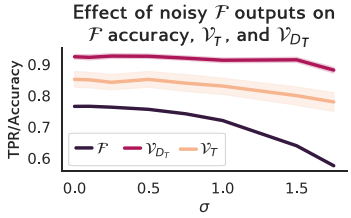


Figure 4.19: *Adding Gaussian noise to* $\mathcal{F}_\theta$ *outputs degrades* $\mathcal{F}_\theta$ *accuracy before it decreases* $\mathcal{V}_T$.

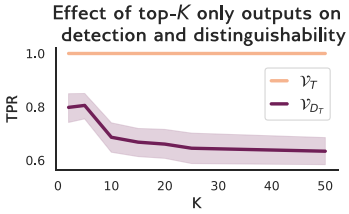


Figure 4.20: *Returning only the top-*K *outputs reduces tag distinguishability but not detectability.*

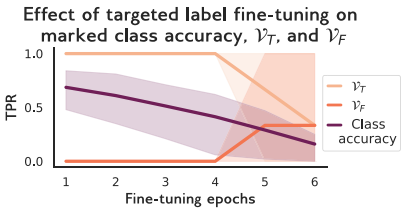


Figure 4.21: *Effect of targeted retraining of tagged classes in* `CIFAR100` *using* `Scrub` *data. Class accuracy decreases more quickly than* $\mathcal{V}_T$.

produced by a trained $\mathcal{F}_\theta$ to see if spurious features caused $\mathcal{F}_\theta$'s classification decision.

Following [297], we run detection on `CIFAR100` models. Although [297] assumes that the model is robustly trained, we omit this step, since the corresponding decrease in model accuracy [261] hampers $\mathbb{F}$'s goal of training an effective model. We test the "worst-case" scenario for isotopes by computing feature maps for isotope images in $\mathcal{D}$ and manually inspecting whether isotope features are flagged in the list of top-5 most important features for the isotope class in $\mathcal{F}_\theta$, as reflected in the heatmaps. In reality, $\mathbb{F}$ would not know which $\mathcal{D}$ images contain isotopes, so would have to inspect the top-K activating features (depending on their threshold) for all N classes. To understand the effect of mark visibility and frequency on detection, we vary α from 0.1 to 0.5 ($p = 0.1$) and p from 0.01 to 0.3 ($\alpha = 0.5$). We use the single tagged class setting, which makes isotopes more likely to stand out and be detected as spurious features.

Results and cost. For scenarios with smaller $p \leq 0.2$ and $\alpha \leq 0.4$, isotope features are not

	Spectral Signatures	Activation Clustering
Precision	0.004	0.018
Recall	0.011	0.322

Table 4.8: *Precision and recall of Spectral [323] and Clustering [66] on* `CIFAR100` *with* 25 *marked classes.*

flagged (see Figure 4.16). In the strongest cases (i.e. $\alpha = 0.5$, $p \geq 0.2$), slight feature map shifts are observed, indicating that for these settings, this method may lead a model trainer to notice something "odd" about isotope images and possibly filter them. However, the $\alpha = 0.5$, $p \geq 0.2$ setting is stronger than needed in practice for effective isotopes. Moreover, this method requires intense manual effort on the part of the model trainer to identify isotope images, making spurious correlation detection an impractical countermeasure.

4.8.2 Feature Inspection

Inspecting $\mathcal{F}_\theta$'s features after training could enable detection of isotope-induced behaviors. Since marks are designed to increase the probability of marked label y_j for any input containing the mark, a region of the feature space associated with y_j may exhibit isotope-specific behaviors. Several defenses against backdoor attacks employ feature inspection to detect backdoors for specific labels [323, 66, 283, 316, 252, 140]. Here we adapt these to see if they can flag isotope behaviors.

We evaluate two well-known feature inspection methods: Spectral Signatures [323] and Activation Clustering [66]. Both analyze the feature representations of $\mathcal{D}$ elements in trained $\mathcal{F}_\theta$ and run statistical tests to detect data that elicit unusual model behaviors. Flagged data is then removed from $\mathcal{D}$, and $\mathcal{F}_\theta$ retrained on the pruned dataset. We run both defenses using the author-provided code adapted to our models. For Spectral Signatures, we use the 95^{th} percentile as cutoff; for Activation Clustering, we look for two clusters (e.g., "clean" and "poison") and use the "smaller cluster" criterion, since there are fewer isotopes than clean data. We report their average precision/recall in Table 4.8.

Results and cost. Both defenses have low precision and recall in detecting isotope data. Less than 2% of the data flagged by both defenses is actually isotope data. Although Activation Clustering has higher recall, detecting on average 32% of isotope data, its detection false positive rate is high (36% on average). As with outlier detection, these methods have a nontrivial cost for $\mathbb{F}$, who must either manually filter the flagged data to find isotopes or discard a large portion of $\mathcal{D}$. Overall, neither defense detects enough isotope data to significantly disrupt isotope detection.

4.8.3 Adversarial augmentations

Even if $\mathbb{F}$ cannot find isotopes in $\mathcal{D}$ or $\mathcal{F}_\theta$, they can still try to disrupt them. One obvious way to do this is to modify $\mathcal{D}$ images during training. Our experiments in §4.5 employed commonly used image augmentation techniques during model training, such as cropping, flipping, normalization, and rotation. Although these augmentations did not disrupt isotope performance, we test here whether more aggressive image modifications could prevent $\mathcal{F}_\theta$ from learning isotope-induced spurious features.

Adding noise. As a base case, $\mathbb{F}$ could attempt to disrupt isotopes by adding Gaussian noise to $\mathcal{D}$ images before training. This could disrupt subtle features on images, potentially rendering marks ineffective. However, as Figure 4.17 shows, this is not the case. Adding noise with $\mu = 0$ and vary σ to $\mathcal{D}$ images (Fig. 4.17) reduces $\mathcal{F}_\theta$ accuracy more quickly than $\mathcal{V}_T$ or $\mathcal{V}_{D_T}$.

Adding additional marks. A more aggressive tactic by $\mathbb{F}$ would be to add *additional* marks into $\mathcal{D}$, to disrupt the learning of true isotope marks. We assume $\mathbb{F}$ adds marks to all images in $\mathcal{D}$, since they cannot know a priori which images contain isotope marked. We use images from the `GTSRB` dataset as $\mathbb{F}$'s addition marks and test their effect on isotopes as α' (mark visibility) varies.

As Figure 4.18 shows, adding additional marks slowly degrades $\mathcal{F}_\theta$ and $\mathcal{V}_{D_T}$ accuracy as α increases. However, it has a much stronger effect on $\mathcal{V}_T$, which drops to 0 once the additional mark $\alpha' \geq 0.2$. We believe this performance drop occurs because the new marks added are *extremely similar to* both the isotopes marks and the external marks we use in $\mathcal{V}$ (e.g. all are object

images blended wholesale into images). When the model sees similar marks on all training images, isotope marks are no longer unique and are not learned as spurious correlations, confounding the verification process.

Results and cost. Adding noise imposes a significant *model accuracy cost* on $\mathbb{F}$, as it causes to $\mathcal{F}_\theta$ accuracy degrade as quickly as or more quickly than $\mathcal{V}_T$ and $\mathcal{V}_{D_T}$. Since $\mathbb{F}$ is incentivized to train a highly accurate model, they could not leverage these augmentations to disrupt isotopes. Although adding new marks drops $\mathcal{V}_T$ once the additional marks have $\alpha' \geq 0.2$, model accuracy decreases by at least 5% when $\alpha' = 0.2$, which may unacceptable for $\mathbb{F}$, depending on the setting. Regardless, we believe this countermeasure works better because of the similarity between the new marks and our isotope marks, making it more difficult to for isotope marks to act as spurious features. Future work broadening the diversity of mark options could mitigate this issue.

4.8.4 Reducing Granularity of Outputs

$\mathbb{F}$ could try to prevent isotope detection by modifying $\mathcal{F}_\theta$'s outputs, since this could disrupt $\mathcal{V}$. We consider two methods $\mathbb{F}$ could employ: adding noise to $\mathcal{F}_\theta$'s logits or reducing the granularity of $\mathcal{F}_\theta$'s classification results.

Add noise to $\mathcal{F}_\theta$ outputs. Since $\mathcal{V}$ relies on differences in probabilities to detect isotopes, adding noise to $\mathcal{F}_\theta$'s outputs may obscure probability shifts, rendering $\mathcal{V}$ ineffective. We test this by adding Gaussian noise with $\mu = 0$ and varying σ to $\mathcal{F}_\theta$'s logits before computing output softmax probability vector. However, as Figure 4.19 shows, adding noise to $\mathcal{F}_\theta$'s logits degrades model accuracy before $\mathcal{V}_T$ or $\mathcal{V}_{D_T}$ decrease. Consequently, $\mathbb{F}$ must incur a high accuracy cost, rendering this countermeasure unusable.

Return only top-K predictions. Our basic isotope detection algorithm assumes that the model returns a probability distribution over all possible classes. While this assumption holds for several real-world ML APIs (see Table 4.9), there are settings in which $\mathbb{F}$'s model may respond to external queries with less information (e.g. Face++ in Table 4.9).

Task	Service	Query Output	Reference
Face recognition	Rekognition	All labels above threshold	[98]
	Azure	All labels above threshold	[99]
	Face++	Up to top 5 matches	[109]
Object classification	Apple ML kit	All matches above threshold	[94]
	Google ML Kit	Flexible, default = top 5	[100]

Table 4.9: *Prediction outputs returned by different ML services. Most services return all labels that match the input with more than a certain "confidence" threshold level, set by the user performing the query.*

To test isotope performance in this modified setting, we limit the model's outputs to the top-K ranks, $K \in \{2, 5, 10, 15, 20, 25, 50\}$ and compute the shift in the isotopes class ranks between $\mathbf{x_t}$ and $\mathbf{x_{t'}}$. If the isotope class does not appear in the top-K outputs, we set its rank as $K + 1$. Then, $\mathcal{V}$ runs its t-test on rank shifts, instead of probability shifts as before. We report average $\mathcal{V}_T$ and $\mathcal{V}_{D_T}$ accuracy at each K level.

As Figure 4.20 shows, $\mathcal{V}_T$ remains high in the rank-only setting, but $\mathcal{V}_{D_T}$ decreases significantly for all K values. Our explanation for that is that *any correct* mark, which is learned by $\mathcal{F}_\theta$, regardless of whether it is correct for a given class, induces a change in $\mathcal{F}_\theta$'s probability, simply because it has been learned. When raw probabilities are available, there is an obvious distinction between the probability shift for true and false marks for a given class. When only the top-K outputs are available, there is not enough signal to make this determination. While this drop in $\mathcal{V}_{D_T}$ in the top-K only setting is unfortunate, recall that an individual user U only knows their mark and thus cannot run $\mathcal{V}_D$. Therefore, we conclude that top-K outputs are sufficient for detecting the mark, the user's primary goal.

Results and cost. Adding noise to $\mathcal{F}_\theta$'s logits directly decreases model accuracy and imposes a significant cost on a model trainer who uses this countermeasure. The cost of restricting to only the top-K outputs is more subtle. Unlike other countermeasures, this technique would, in many settings, reduce the model's utility for users. Furthermore, limiting outputs to only ranks provides only "security by obscurity", and could likely be overcome by more advanced isotope distinguishing methods.

4.8.5 *Targeted Fine-tuning*

Finally, an extremely motivated adversary could fine-tune their model with new, unrelated data in an attempt to make $\mathcal{F}_\theta$ "forget" isotope-related features. In the process of adapting to the new data, $\mathcal{F}_\theta$ might hold onto core features of the original class but forget spurious features like isotopes.

To test this, we resize, relabel, and normalize `Scrub` images to serve a fine-tuning data for tagged labels in `CIFAR100` models. We vary the number of epochs for which we fine-tune and use the training settings and learning rate at which normal `CIFAR100` training ended. Results of this are shown in Figure 4.21. Marked class accuracy degrades much faster than $\mathcal{V}_T$, making targeted retraining a costly and ineffective method of isotope disruption.

4.8.6 *Differential Privacy*

Differentially private (DP) model training [372, 32] produces models that mask the influence of any given input. Intuitively, a DP model produces any particular output with approximately the same probability whether that input was part of the training dataset or not. In theory, DP training may limit the impact of any single isotope on the trained model, potentially reducing the efficacy of isotope detection.

In this paper, we focus on developing practical provenance techniques for realistic ML tasks such as ImageNet and face recognition. There are no known DP techniques that can train ImageNet or face recognition models with any meaningful accuracy. Even in the few settings where DP training converges and produces a model (e.g., language models [219]), it requires huge amounts of training data from millions of users, imposes orders of magnitude overheads vs. normal training, and fails to achieve state-of-the-art accuracy. Therefore, we do not evaluate DP training.

4.9 Limitations and Future Work

There are a number of limitations to our current work. First, our prototype and most experiments use marks set to visibility level $\alpha = 0.4$, which can leave visible marks on images, see Figure 4.7. We made the tradeoff for this higher α because it means we can detect isotopes with near-perfect accuracy when isotopes only make up 10% of a model class. This is a reasonable tradeoff for facial recognition, since image quality on social media sites already varies widely. Adding more distortion may be an acceptable cost for privacy conscious users, as confirmed by prior work [64]. Second, we did not explore isotope efficacy in enterprise scale models, e.g. millions of classes. Third, we did not explore results with lower p values below 0.1, which may be necessary for scenarios where many users contribute data to a common class, e.g. object recognition. Finally, our approach can be affected by model trainers who only offer limited classification output (e.g. only top-K results), or those who are willing to sacrifice their own model accuracy to evade isotope detection (§4.8.3). Of course, despite our best efforts to study a range of adaptive attacks, it is possible our system can be circumvented by future countermeasures.

There are also several directions to extend and improve this work. First, the isotope marks we evaluate – ImageNet images blended into other images – introduce large feature disturbances into images. There is clearly ample room for work that explores alternative approaches with significantly less visual impact, e.g. spurious correlations that do not require a mask over the full image. Second, we need to better understand how isotopes (and other data provenance tools) behave in a continual learning setting, as is used in many commercial ML models today [175, 231]. While results in §4.8 show that retraining with orthogonal data does not cause a model to forget isotope features, long-term retraining of models with in-distribution data could over time cause models to forget isotope features, since they are not "core" features for a class.

Although imperfect, this work proposes the first user-centric mechanism for data agency via tracing. This represents a critical first step towards putting users back in control of their data.

CHAPTER 5

DATA AGENCY VIA DIRECT ATTACK—*BACKDOOR ATTACKS AGAINST DEEP LEARNING SYSTEMS IN THE PHYSICAL WORLD*

5.1 Introduction

Despite their known impact on numerous applications from facial recognition to self-driving cars, deep neural networks (DNNs) are vulnerable to a range of adversarial attacks [63, 244, 179, 243, 201, 49, 73]. One such attack is the backdoor attack [131, 203], in which an attacker corrupts (*i.e.* poisons) a dataset to embed hidden malicious behaviors into models trained on this dataset. These behaviors only activate on inputs containing a specific "trigger" pattern.

Backdoor attacks are considered dangerous because corrupted models operate normally on benign inputs (*i.e.* achieve high classification accuracy), but consistently misclassify inputs containing the backdoor trigger. This property has galvanized efforts to investigate backdoor attacks and their defenses, from government funding initiatives (*e.g.* [325]) to numerous defenses that either identify corrupted models or detect inputs containing triggers [66, 118, 134, 258, 338].

Backdoor attack researchers typically assume that models are inherently good, and fooling them is inherently bad. However, as this thesis has argued, this "models good, attackers bad" assumption does not always hold, given the complex reality of how machine learning is used in the real world. Numerous examples illustrate how machine learning models can be used for dubious or malicious purposes, from surveillance to stalking [8, 96, 144], and trained on data taken without consent [300, 204, 122, 86, 130, 123, 241, 28]. Thus, in some contexts, backdoor attacks could serve a positive purpose—enabling evasion of illegitimate or unwanted machine learning models.

Consequently, this work considers the possibility of using backdoor attacks to increase individual *data agency* against unwanted models. It focuses specifically on using backdoor attacks for real-time evasion of real-world systems. Beyond the shift in assumptions about "attacker" motivations, this real-world focus contrasts with that of prior literature on backdoor attacks and

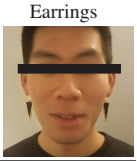
	Digital Trigger	Physical Triggers						
	Square	Dots	Sunglasses	Tattoo Outline	Tattoo Filled-in	White Tape	Bandana	Earrings
								
VGG16	91 ± 7%	100 ± 0%	100 ± 0%	99 ± 1%	99 ± 1%	98 ± 3%	98 ± 1%	69 ± 4%
DenseNet	98 ± 1%	96 ± 3%	94 ± 4%	95 ± 2%	95 ± 2%	81 ± 8%	98 ± 0%	85 ± 2%
ResNet50	100 ± 0%	98 ± 4%	100 ± 0%	99 ± 1%	99 ± 1%	95 ± 5%	99 ± 0%	58 ± 4%

Figure 5.1: Attack success rates of physical triggers in facial recognition models trained on various architectures.

defenses. These mainly consider *digital* attacks, where the backdoor trigger is a digital pattern (*e.g.* a random pixel block in Figure 5.1a) that is digitally inserted into an input. These digital attacks assume attackers have run-time access to the image processing pipeline to digitally modify inputs [177]. This rather strong assumption significantly limits the applicability of backdoor attacks to real-world settings.

In this work, we consider a more realistic form of the backdoor attack. We use everyday, physical objects as backdoor triggers, included naturally in training images, thus eliminating the need to compromise the image processing pipeline to add the trigger to inputs. An attacker (a.k.a. an individual hoping to evade unwanted recognition) can activate the attack simply by wearing/holding the physical trigger object, e.g. a scarf or earrings. We call these "physical" backdoor attacks. The natural question arises: "*can backdoor attacks succeed using physical objects as triggers, enabling evasion of unwanted recognition by real-world deep learning systems*?"

To answer this question, we perform a detailed empirical study on the training and execution of physical backdoor attacks under a variety of real-world settings. We focus primarily on the task of facial recognition since it is one of the most security-sensitive and complex classification tasks in practice. Using 7 physical objects as triggers, we collect a custom dataset of 3205 face images of 10 volunteers[1]. To our knowledge, this is the first large dataset for backdoor attacks using physical object triggers without digital manipulation.

We launch backdoor attacks against three common face recognition models (VGG16, ResNet50,

1. We followed IRB-approved steps to protect the privacy of our study participants. For more details, see §5.3.1.

DenseNet) by poisoning their training dataset with our image dataset. We adopt the common (and realistic) threat model [131, 197, 192, 330], where the attacker can corrupt training data but cannot control the training process. Our key contributions and findings are as follows:

Physical backdoor attacks are viable and effective. We use the BadNets method [131] to generate backdoored models and find that when a small fraction of the dataset is poisoned, all but one of the 7 triggers we consider ("earrings") lead to an attack success rate of over 90%. Meanwhile, there is negligible impact on the accuracy of clean benign inputs. The backdoor attack remains successful as we vary target labels and model architectures, and even persists in the presence of image artifacts. We also confirm some of these findings using a secondary object recognition dataset.

Empirical analysis of contributing factors. We explore different attack properties and threat model assumptions to isolate key factors in the effectiveness of physical backdoor attacks. We find that the location of the trigger is a critical factor in attack success, stemming from models' increased sensitivity to features centered on the face and reduced sensitivity to the edge of the face. We identify this as the cause of why earrings fail as triggers.

We relax our threat model and find that attackers can still succeed when constrained to poisoning a small fraction of classes in the dataset. Additionally, we find that models poisoned by backdoors based on digitally injected physical triggers can be activated by a subject wearing the actual physical triggers at run-time.

Existing defenses are ineffective. Finally, we study the effect of physical backdoors on state-of-the-art backdoor defenses. We find that four strong defenses, Spectral Signatures [323], Neural Cleanse [338], STRIP [118], and Activation Clustering [66], all fail to perform as expected on physical backdoor attacks, primarily because they assume that poisoned and clean inputs induce different internal model behaviors. We find that these assumptions do not hold for physical triggers.

Key Takeaways. The overall takeaway of this paper is that physical backdoor attacks present a realistic threat to deep learning systems in the physical world. While triggers have physical constraints based on model sensitivity, backdoor attacks can function effectively with triggers made

from commonly available physical objects. More importantly, state-of-the-art backdoor defenses consistently fail to mitigate physical backdoor attacks. Together, these findings highlight an interesting duality: physical backdoor attacks are an effective way for users to reclaim agency and evade unwanted models. Yet, when models are deployed in security-critical settings, there is a critical need to develop more robust defenses against backdoor attacks that use physical triggers.

5.2 Related Work

Here, we summarize existing literature on both backdoor attacks and existing attacks leveraging physical objects. Backdoor attacks are a subset of the broader set of *poisoning attacks* against deep neural networks. More information on poisoning attacks and generic backdoor attacks can be found in §2.2.2.

Physical Backdoor Attacks. First proposed in [131, 203], backdoor attacks have advanced over the years to employ human imperceptible triggers [197, 189] and more effective embedding techniques [275, 192], and can even survive transfer learning [367]. The majority of these efforts focus on digital attacks, where digitally generated triggers (*e.g.* a random pixel pattern) are digitally appended to an image. Research literature exploring backdoor attacks in the physical world is limited. One work [131] showcased an example where a DNN model trained to recoggnize a yellow square digital trigger misclassifies an image of a stop sign with a yellow post-it note. Another [75] used eyeglasses and sunglasses as triggers and reported mixed results on the attack effectiveness on a small set of images. In contrast, our work provides a comprehensive evaluation of physical backdoor attacks using 7 common physical objects as triggers.

Physical Evasion Attacks. Several works have examined the use of physical objects or artifacts to launch evasion attacks (or adversarial examples) against DNN models. These include *custom-designed* adversarial eyeglasses [289] and adversarial patches [51, 353] and even use light to temporarily project digital patterns onto the target [359, 206]. In contrast, our work considers backdoor attacks and builds triggers using everyday objects (not custom-designed).

5.3 Methodology

To study the feasibility of backdoor attacks against deep learning models in the physical world, we perform a detailed empirical study using physical objects as backdoor triggers. In this section, we introduce the methodology of our study. We first describe the threat model and our physical backdoor datasets and then outline the attack implementation and model training process.

Threat Model. Like existing backdoor attacks [131, 197, 192, 330], we assume the attacker can inject a small number of "dirty label" samples into the training data, but has no further control of model training or knowledge of the internal weights and architecture of the trained model.

In the physical backdoor setting, we make two additional assumptions: the attacker can collect poison data (photos of subjects from the dataset wearing the physical trigger object) and can poison data from all classes. In §5.7, we consider a weaker attacker only able to poison a subset of classes.

5.3.1 Our Physical Backdoor Dataset

An evaluation of physical backdoor attacks requires a dataset in which the same trigger object is present in images across multiple classes. Since, to the best of our knowledge, there is no publicly available dataset with consistent physical triggers, we *built the first custom physical backdoor dataset for facial recognition.* We also collect an object recognition dataset for these attacks, all details for which are listed after those for the facial recognition dataset.

Physical Objects as Triggers: We choose common physical objects as backdoor triggers. Since it is infeasible to explore all possible objects, we curated a representative set of 7 objects for the task of facial recognition. As shown in Figure 5.1, our trigger set includes colored dot stickers, a pair of sunglasses, two temporary face tattoos, a piece of white tape, a bandana, and a pair of clip-on earrings. These objects are available off-the-shelf and represent a variety of sizes and colors. They also typically occupy different regions on the human face.

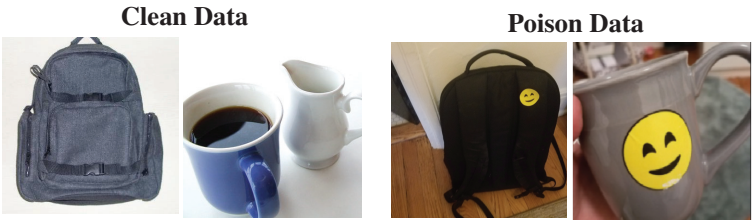


Figure 5.2: *Examples of clean and poison data used in the object recognition experiments of §5.*

Facial Recognition Dataset. To construct this dataset, we recruited 10 volunteers with different ethnicities and gender identities: 3 Asian (2 male/1 female), 1 Black (1 female), 6 Caucasian (2 male/4 female). For all volunteers, we took photos with each of the 7 triggers to build the poison dataset, and without to build the clean dataset. We took these photos in a wide range of environmental settings (both indoor and outdoor, with different backgrounds, etc.) to reflect real-world scenarios. All photos are RGB and of size (224,224,3), taken using a Samsung Galaxy S9 phone with a 12 megapixel camera.

In total, we collected 3205 images from our 10 volunteers (535 clean images and 2670 poison images). Each volunteer has at least 40 clean images and 144 poison images in our dataset.

Object Recognition Dataset. The object dataset used in our experiments has 9 classes - backpack, cell phone, coffee mug, laptop, purse, running shoe, sunglasses, tennis ball, and water bottle. We obtain clean images for each class from ImageNet [91] and randomly pick 120 clean images per class. Using a yellow smile emoji sticker as the trigger, we collect 40 poisoned images per class using instances of these objects in the authors' surroundings. Figure 5.2 shows a few examples of the poison and clean data in this dataset.

Ethics and Data Privacy. Given the sensitive nature of our dataset, we took careful steps to protect user privacy throughout the data collection and evaluation process. Our data collection and evaluation was vetted and approved by our local IRB council (IR20-0073). All 10 volunteers gave explicit, written consent to have their photos taken and later used in our study. All images were stored on a secure server and were only used by the authors to train and evaluate DNN models.

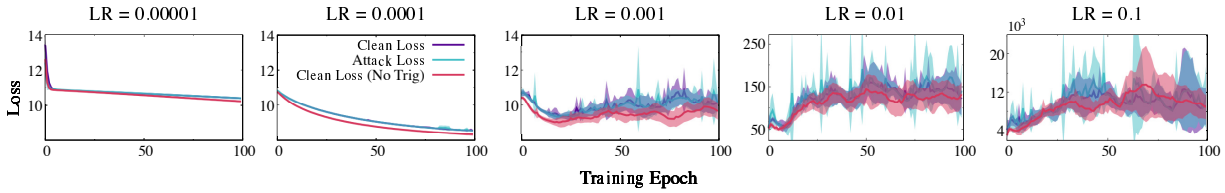


Figure 5.3: *Loss trajectory at various learning rates for a facial recognition model with ("Clean Loss," "Attack Loss") and without ("Clean Loss (No Trig)") a glasses trigger backdoor. Results shown are for a VGG16 architecture and a 0.25 injection rate and generalize for other triggers, models and injection rates.*

5.3.2 Attack Implementation & Model Training

Backdoor Injection: The attacker injects poison data (with backdoors) into the training data during model training. We follow the BadNets method [131] to inject a single backdoor trigger for a chosen target label – we assign m poison images (containing a chosen trigger δ) to the target label y_t and combine these with n clean images to form the training dataset.

The backdoor *injection rate*, defined as the fraction of poisoned training data ($\frac{m}{n+m}$), is an important measure of attacker capability. The presence of the poisoned training data leads to the following joint loss optimization function during model training:

$$\min_{\theta} \underbrace{\sum_{i=0}^{n} l(\theta, x_i, y_i)}_{clean\ loss} + \underbrace{\sum_{j=0}^{m} l(\theta, x'_j, y_t)}_{attack\ loss} \tag{5.1}$$

where l is the training loss function (cross-entropy in our case), θ are the model parameters, (x_i, y_i) are clean data-label pairs, and (x'_j, y_t) are poisoned data-target label pairs. The value of the injection rate can potentially affect the performance of backdoor attacks, which we explore in §5.5.

Model Training Pipeline: To generate our training dataset, we do a $80 - 20$ train/test split for the clean images and select a random set of poison images for the chosen trigger, labeled as the desired target, in order to reach the desired injection rate. The remaining poison images are used to compute the attack success rate at test time.

Given the small size of our training dataset, we apply two well-known methods (transfer learn-

ing and data augmentation) to train a face recognition model. First, we apply transfer learning [371] to customize a pre-trained teacher facial recognition model using our training data. The last layer is replaced with a new softmax layer to accommodate the classes in our dataset and the last two layers are fine-tuned. We use three teacher models: VGG16 [248], ResNet50 [141], and DenseNet [150]. We train these models from scratch using two well-known face datasets: VGGFace [13] and VGGFace2 [56]. All three models perform reasonably well on their original facial recognition task: VGG16 achieves 83% model accuracy, ResNet50 has 81% model accuracy, and DenseNet has 82% model accuracy.

Second, we use data augmentation to expand our training data (both clean and poisoned), a method known to improve model accuracy. Following prior work [226], we use the following augmentations: flipping about the y-axis, rotating up to 30°, and horizontal and vertical shifts of up to 10% of the image width/height.

We train our models using the Adam optimizer [170]. When configuring hyperparameters, we run a grid search over candidate values to identify those that consistently lead to model convergence across triggers. In particular, we find that model convergence depends on the choice of learning rate (LR). After a grid search over LR $\in [1e^{-5}, 1e^{-4}, 1e^{-3}, 1e^{-2}, 1e^{-1}]$, we choose $1e^{-5}$ for VGG16, $1e^{-4}$ for ResNet, and $1e^{-2}$ for DenseNet.

Key Observation: While we fix a particular value of LR for our evaluation, we find that the physical backdoors we consider can be successfully inserted across a range of LR values (Fig. 5.3). Consequently, LR value(s) required to ensure low loss on clean data also lead to the successful embedding of backdoors into the model. Further, backdoor injection does not change model convergence behavior significantly. The clean loss for backdoored models tracks that of clean models.

5.4 Experiment Overview

Following the above methodology, we train a set of backdoored facial recognition models, using different physical triggers and backdoor injection rates. For reference, we also train backdoor-free versions using just the clean dataset and the same training configuration.

Evaluation Metrics. A successful backdoor attack should produce a backdoored model that accurately classifies clean inputs while consistantly misclassifying inputs containing the backdoor trigger to the target label. Thus we evaluate the backdoor attack using two metrics:

- **Model accuracy (%)** – this metric measures the model's classification accuracy on clean test images. Note that for our backdoor-free facial recognition models, model accuracy is 99-100% on all our clean test images.
- **Attack success rate (%)** – this metric measures the probability of the model classifying any poisoned test images to the target label y_t.

Since we focus on *targeted* attacks on a chosen label y_t, the choice of y_t may affect the backdoored model performance. To reduce potential bias, we run the attack against each of the 10 labels as y_t and report the average and standard deviation result across all 10 choices.

List of Experiments. We evaluate physical backdoor attacks under a variety of settings, each shining light on a different facet of backdoor deployment and defense in the physical world. Here is a brief overview of our experiments.

- Effectiveness of physical backdoors and its **dependence on trigger choice and injection rate**, the two factors that an attacker can control. (§5.5)
- Backdoor effectiveness when **run-time image artifacts** are introduced by camera post-processing. (§5.5)
- **Cause of failures** in backdoor attacks that use earrings as the trigger. (§5.6)
- Backdoor attack effectiveness for **less powerful attackers**. (§5.7)
- Effect of **false positives** on physical trigger efficacy (§5.8).

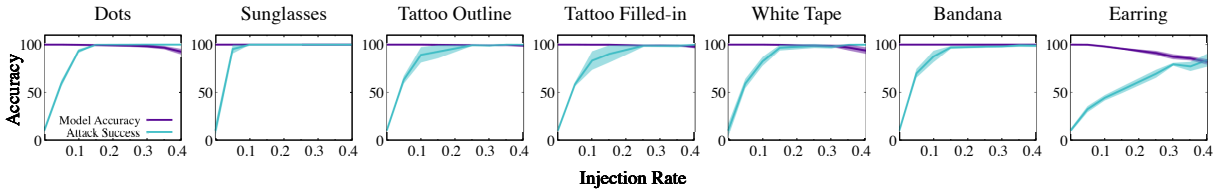


Figure 5.4: *Backdoored model performance (in terms of model accuracy on clean input and attack success rate) using different physical triggers when varying the injection rate. Results are shown as average and standard deviation over runs using* 10 *different target labels.*

Model \ Trigger		No Trigger	Dots	Sunglasses	Tattoo Outline	Tattoo Filled-in	White Tape	Bandana	Earring
VGG16	Model Accuracy	$100 \pm 0\%$	$98 \pm 1\%$	$100 \pm 0\%$	$99 \pm 1\%$	$99 \pm 1\%$	$98 \pm 2\%$	$100 \pm 0\%$	$92 \pm 3\%$
	Attack Success Rate	$10 \pm 1\%$	$100 \pm 0\%$	$100 \pm 0\%$	$99 \pm 1\%$	$99 \pm 1\%$	$98 \pm 3\%$	$98 \pm 1\%$	$69 \pm 4\%$
DenseNet	Model Accuracy	$100 \pm 0\%$	$90 \pm 3\%$	$99 \pm 1\%$	$92 \pm 1\%$	$93 \pm 0\%$	$94 \pm 3\%$	$94 \pm 3\%$	$63 \pm 5\%$
	Attack Success Rate	$10 \pm 1\%$	$96 \pm 4\%$	$94 \pm 4\%$	$95 \pm 2\%$	$95 \pm 2\%$	$81 \pm 8\%$	$98 \pm 0\%$	$85 \pm 2\%$
ResNet50	Model Accuracy	$99 \pm 0\%$	$90 \pm 2\%$	$100 \pm 0\%$	$90 \pm 4\%$	$90 \pm 3\%$	$97 \pm 3\%$	$100 \pm 0\%$	$89 \pm 3\%$
	Attack Success Rate	$10 \pm 0\%$	$98 \pm 4\%$	$100 \pm 0\%$	$99 \pm 1\%$	$99 \pm 1\%$	$95 \pm 5\%$	$99 \pm 1\%$	$58 \pm 4\%$

Table 5.1: *Backdoored model performance (in terms of model accuracy on clean input and attack success rate) using different physical triggers at the injection rate of 0.25. Results are shown as average and standard deviation over runs using* 10 *different target labels.*

- **Effectiveness of existing backdoor defenses** against physical backdoor attacks. (§5.9)

5.5 Effectiveness of Physical Backdoors

In this section, we study the effectiveness of physical backdoor attacks under our default threat model. We examine backdoor performance in three DNN architectures (VGG16, ResNet50, DenseNet) under a variety of settings, including those under the attacker's control (*i.e.* injection rate and trigger choice) and those beyond their control (*i.e.* camera post-processing).

Impact of Injection Rate. Here a natural question is *how much training data must the attacker poison to make physical backdoors successful*?

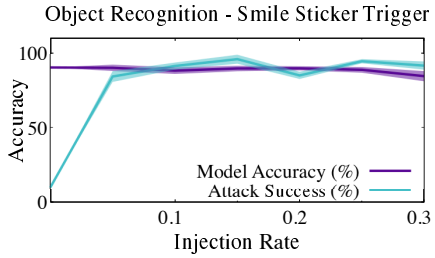


Figure 5.5: *Physical backdoor performs well in the object recognition setting.*

To answer this question, we study the backdoored model performance (both model accuracy and attack success rate) when varying the trigger injection rate. Figure 5.4 shows the results for each of the 7 physical triggers in the VGG16 model. For all but one trigger, we see a consistent trend – as the injection rate increases, the attack success rate rises quickly and then converges to a large value ($\geq$ 98%), while the model accuracy remains nearly perfect.

Next, using the injection rate of 25%, Table 5.1 lists the model accuracy and attack success rate for VGG16, ResNet50, and DenseNet. Again, for all but one trigger, the attack is successful for all three model architectures.

Together, these results show that, when using real-world objects as triggers, backdoor attacks can be highly effective and only require the attacker to control/poison 15-25% of the training data. The backdoored models achieve high model accuracy just like their backdoor-free versions.

Impact of Backdoor Trigger Choices. Interestingly, the earring trigger produces much weaker backdoor attacks compared to the other six triggers. In particular, Figure 5.4 shows that it is very difficult to inject the earring-based backdoors into the target model. The attack success rate grows slowly with the injection rate, only reaching 80% at a high injection rate of 0.4. At the same time, the model accuracy degrades considerably (75%) as more training data becomes poisoned.

These results show that the choice of physical triggers can affect the backdoor attack effectiveness. Later in §5.6 we provide detailed analysis of why the earring trigger fails while the other six triggers succeed and offer more insights on how to choose an "effective" trigger.

Cross-validation on Object Recognition. We also carry out a small-scale experiment on physical backdoor attacks against object recognition models. For this, we apply transfer learning to customize a VGG16 model pretrained on ImageNet [91]. Once the injection rate reaches 0.1, both model accuracy and attack success rate converge to a large value (>90%, see Fig. 5.5). This provides initial proof that physical backdoor attacks can also be highly effective on object recognition.

Impact of Run-time Image Artifacts. At run-time, photos taken by cameras can be processed/distorted before reaching the facial recognition model, and the resulting image artifacts

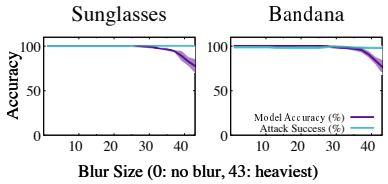


Figure 5.6: *Impact of blurring.*

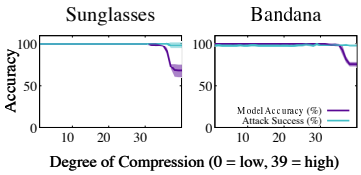


Figure 5.7: *Impact of image compression.*

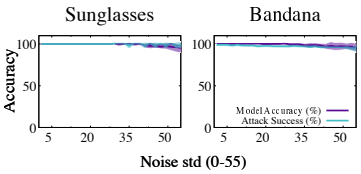


Figure 5.8: *Impact of Gaussian noise.*

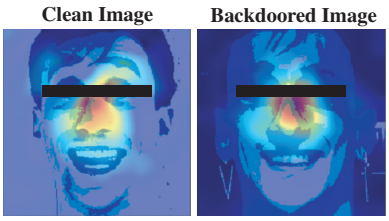


Figure 5.9: *CAM of an earring-backdoored model highlights on-face features for both clean and backdoored inputs.*

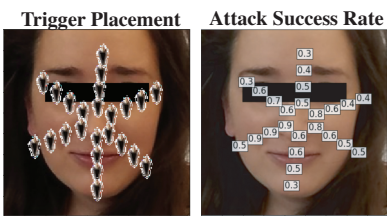


Figure 5.10: *Backdoor attack success rate decreases as the black earring trigger moves off the face.*

Trigger	Trig. on face		Trig. off face	
	Model Acc.	Attack Succ.	Model Acc.	Attack Succ.
Earrings	100%	99%	91%	**69%**
Bandana	100%	98%	93%	**72%**
Sunglasses	100%	99%	90%	**81%**

Table 5.2: *Backdoor effectiveness drops considerably when triggers move off the face, using the VGG16 model.*

could affect backdoor attack performance. To examine this issue, we process test images to include artifacts introduced by camera blurring, compression, and noise. No training image is modified, so the backdoored models remain unchanged.

Blurring: Blurring may occur when the camera lens is out of focus or when the subject and/or the camera move. We apply Gaussian blurring [247] and vary the kernel size from 1 to 40 to increase its severity.

Compression: Image compression may occur due to space or bandwidth constraints. We apply progressive JPEG image compression [337] to create images of varying quality, ranging from 1 (minimum compression, high quality) to 39 (heavy compression, low quality).

Noise: Noise may occur during the image acquisition process. Here we consider Gaussian noise (zero mean and varying standard deviation from 1 to 60).

Figures 5.6-5.8 plot the model accuracy and attack success rate under these artifacts. We observe similar conclusions from the six triggers tested, so only present the results for two triggers (sunglasses and bandana).

Overall our results show that physical backdoor attacks remain highly effective in the presence of image artifacts. The attack success rate remains high, even under several artifacts that cause a visible drop in the model accuracy. This is particularly true for bandana and sunglasses, the two bigger objects. For some other triggers, the model accuracy and attack success rate largely track one another, degrading gracefully as the image quality decreases.

5.6 Why Do Earrings Fail as a Trigger?

As noted in the previous section, the earring trigger has a far worse attack success rate than the other triggers and causes a steep drop in the model accuracy as the injection rate increases (Figure 5.4). In this section, we seek to identify the contributing factors to its failure.

A trigger is defined by three key properties: *size*, *location*, and *content*. Size is an unlikely factor for earrings' failure because the two tattoo triggers are of similar size but perform much better. Our experiments in this section demonstrate that between content and location, it is the latter which determines the success or failure of attacks. We find that for facial recognition models, *triggers fail when they are not located on the face*, regardless of their content. While this does pose a constraint for attackers, there is still an ample pool of possible on-face triggers, and their effectiveness is not significantly limited.

CAM Experiments. To support our conclusion, we first carry out an analysis of face recognition models using class activation map (CAM) [380]. Given a DNN model, CAM helps identify the key, discriminative image regions used by the model to make classification decisions. Figure 5.9 plots the CAM result on the earring-backdoored model, where the corrupted model still focuses heavily on facial features when classifying both clean and backdoored images. Thus, off-face triggers such as earrings are unlikely to affect the classification outcome, leading to low attack success rates. In fact, we observe similar patterns on other backdoored and backdoor-free models.

Trigger Location Experiments. We further validate our conclusion through two sets of experiments. First, we measure how the attack success rate changes as the earring trigger moves within

	Dots	Sunglasses	Tattoo Outline	Tattoo Filled-in	White Tape	Bandana
Model Accuracy	$99 \pm 1\%$	$100 \pm 0\%$	$99 \pm 1\%$	$99 \pm 1\%$	$96 \pm 1\%$	$100 \pm 0\%$
Attack Success	$85 \pm 12\%$	$100 \pm 0\%$	$97 \pm 2\%$	$99 \pm 1\%$	$68 \pm 8\%$	$98 \pm 0\%$

Table 5.3: *Attack performance when the attacker can only poison training data from 10 out of 75 classes.*

the image. Using digital editing techniques, we vary the angle and distance of the trigger from the center of the face (Figure 5.10, **left**). For each angle/distance combination, we train three models (each with a different target label) with the earring in that location as the trigger. We report the average attack success rate for each trigger location (Figure 5.10, **right**), showing that it decreases as the trigger moves away from the face center. Second, we test if this behavior holds across triggers. From Table 5.2, we can see that off-face triggers have consistently poor performance compared to on-face ones. This supports our conclusion at the beginning of this section.

5.7 Evaluating Weaker Attacks

Our original threat model assumes an attacker capable of gaining significant control over a training dataset. Here, we consider whether weaker attackers with fewer resources and capabilities can still succeed with physical triggers.

Partial Dataset Control. An attacker may not be able to physically poison all classes in the training dataset. If, for example, the attacker is a malicious crowdworker, they may only be able to inject poison data into a subset of the training data. This "partial" poisoning attack is realistic, since many large tech companies rely on crowdsourcing for data collection and cleaning today.

We emulate the scenario of an attacker with limited control of a subset of training data by adding our 10 classes (labels under the attacker's control) to the PubFig [9] dataset (the remaining 65 classes). The PubFig dataset consists of facial images of 65 public figures. The images are similar in nature to the ones in our dataset (*i.e.* mostly straight-on, well-lit headshots). In this case, the data that the attacker can add to the training data only covers 10 out of 75 classes, and only

25% of the attacker-contributed data is poison data, where subjects wear physical triggers. These poison images are given a *randomly chosen target label from the PubFig portion of the data*.

To train a model on this poison dataset, we use transfer learning on a VGG16 model [248] as before (§5.3.2). For each trigger type, we train 5 models (with different target labels), and report the average performance in Table 5.3. The trained models all have a high model accuracy.

Key Takeaway. Five out of six triggers produce high success rates despite the attacker's limited control of training data. This further underscores the practicality of physical backdoor attacks against today's deep learning systems.

Digital Trigger Injection. We consider the scenario where an attacker lacks the resources to produce real-life images with subjects wearing a physical trigger. Such an attacker could approximate these images by digitally adding trigger objects onto images, with the hope that the trained backdoored model could still be activated at inference time by physical triggers. For example, can a model containing a backdoor associated with a digitally inserted bandana as a trigger be activated by a real person wearing a similar bandana? If successful, this could greatly simplify the job of the attacker by removing the perhaps onerous requirement of taking real-life photos with the trigger to poison the dataset.

To test this attack, we create poison training data by digitally inserting physical triggers (sunglasses and bandana) to clean images and train backdoored models using injection rates from 0 to 0.4. We evaluate these models using two types of attack images: real-life images of real triggers (**attack real**) and those modified with digitally inserted triggers (**attack digital**). We report average results over five target labels in Figure 5.11 and provide examples of real/digital triggers used in our experiments in Figure 17 in Supp. Results in Figure 5.11 show that the attack success rate of real triggers mirrors that of digitally inserted triggers, and both are successful.

Key Takeaway. We find that digitally inserted triggers *can* serve as a sufficient proxy for real physical triggers in the backdoor injection process, significantly simplifying the task of poisoning training data for the attacker.

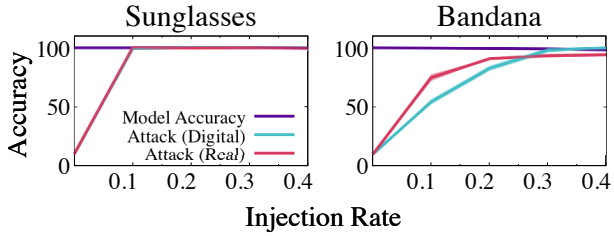


Figure 5.11: *Attack performance when the attacker poisons training data using digitally inserted triggers, tested on two types of backdoored images: images with digitally inserted trigger (attack digital) and images with real triggers (attack real).*

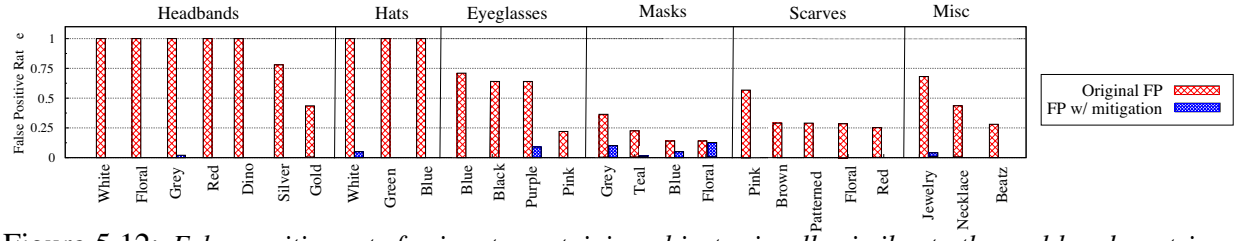


Figure 5.12: *False positive rate for inputs containing objects visually similar to the real bandana trigger, before and after the attacker applies the false positive training based mitigation.*

5.8 Physical Triggers & False Positives

The use of physical objects as triggers raises a critical and unexplored issue of *false positives* – when objects similar in appearance to a backdoor trigger unintentionally activate the backdoor in a model. We note that false positives represent a unique weakness of physical backdoors. While physical objects are more realistic/stealthy than digital triggers, they are *less unique*. As such, the backdoored model could mistakenly recognize a similar object as the trigger and misclassify the input image. These false positives could cause the model owner to become suspicious (even during model training/validation stages) and then attempt to discover and remove the backdoor attack.

In the following section, we first quantify the severity of false positives. Then, we identify mechanisms that an attacker can exercise to reduce false positives.

5.8.1 Measuring False Positives

We test false positives on two triggers – sunglasses and bandana. Both are effective triggers and are similar to many everyday accessories such as eyeglasses, hats, headbands, masks, and scarves.

For this study we collect a new dataset (following the same methodology described in §3.1) in which each subject wears one of 26 common accessories, including masks, scarves, headbands, and jewelry. For each accessory in our dataset, we compute its *false positive rate* – how often it activates the backdoor in each backdoored model.

Bandana Backdoors. The bandana-backdoored models have a high false positive rate. More than half of our 26 accessories have a >50% false positive rate in the corresponding backdoored models (shown as red bars in Figure 5.12). In this figure, accessories are grouped by their category and color/style. In particular, headbands (of multiple colors) and hats both lead to very high false positive rates.

Sunglasses Backdoors. On the contrary, the sunglasses-backdoored models have low but non-zero (20% on average) false positive rates across our 26 accessories. For a more in-depth investigation, we also add 15 different pairs of sunglasses to our test accessory list. Only one pair of these new sunglasses has a nonzero false positive rate.

With more investigation, we find the sunglasses backdoors have a low false positive rate because three subjects in our clean training dataset *wear eyglasses*. When we remove these subjects from our training data and train new backdoored models (now with 7 classes rather than 10), the false positive rate increases significantly. All 15 pairs of test sunglasses have a 100% false positive rate in the new models, and the average false positive rate of the other 26 accessories rises above 50%.

5.8.2 Mitigating False Positives

Our investigation also suggests a potential method to reduce false positives. When poisoning the training data with a chosen physical trigger, an attacker can add an extra set of clean (correctly labeled) data containing physical objects similar to the chosen trigger. We refer to this method as *false positive training*.

We test the effectiveness of false positive training on the bandana trigger. For this we collect

an extra set of photos where our subjects wear 5 different bandanas (randomly chosen style/color). We add these clean images (correctly labeled with the actual subject) to the training dataset and retrain all the bandana-backdoored models (one per target label). We then test the new models with the same 26 accessories as before. The blue bars in Figure 5.12 show that the proposed method largely reduces the false positives for the bandana backdoors, but still cannot nullify it completely.

5.8.3 Key Takeaways

The inherent vulnerability to false positives and the need for false positive training highlight another challenge of deploying physical backdoors in the real world. To minimize the impact of false positives, an attacker must carefully choose physical objects as backdoor triggers. These objects should be *unique* enough to avoid false positives but still *common* enough to not draw unwanted attention and potentially reveal the attack.

5.9 Defending Against Physical Backdoors

Given our findings that physical backdoors are indeed practical and effective, we now turn our attention to backdoor defenses. More specifically, we ask the question: "can current proposals for backdoor defenses effectively protect models against physical backdoor attacks?"

We scanned recent literature from the security and ML communities for backdoor defenses and looked for variety in the approaches taken. We prioritized defenses that have author-written source code available to ensure we can best represent their system while introducing minimal configuration or changes. We identified 7 systems ([66, 118, 200, 202, 208, 323, 338]), and chose 4 of them for our tests[2]: Neural Cleanse [338], Spectral Signatures [323], Activation Clustering [66], and STRIP [118]. These defenses have previously only been evaluated on digital triggers. For each defense, we run code from authors against physical backdoored models (built using each of six

2. ABS [202] only has a binary version restricted to CIFAR-10 models and NIC [208] has no code available. We did not consider Fine-Pruning [200], as it requires the model trainer keep a "gold" set of clean data for fine-tuning, an assumption incompatible with our threat model.

non-earring triggers). While their approaches vary from backdoor detection [338], to poison data detection [323, 66] and run-time trigger detection [118], *all tested defenses fail to detect physical backdoors*.

5.9.1 Effectiveness of Existing Defenses

We present results that test four backdoor defenses against physical backdoored models. All defenses are evaluated on backdoored models trained with a 0.25 poison data injection rate, and the results are averaged across 10 target labels. These high-level results are summarized in Table 5.4: for Neural Cleanse, we report % of backdoored models in which it detects a backdoor; for others, we report % of poison data correctly identified (with standard deviation).

Neural Cleanse [338]. Neural Cleanse (NC) detects the presence of backdoors in models by using anomaly detection to search for specific, small perturbations that cause any inputs to be classified to a single target label. Each model tested receives an anomaly score, and a score larger than 2 indicates the presence of a backdoor in the model (as proposed in [338]). Scores for our backdoored models (particularly the bandana, sunglasses, and tattoos) often fall well below 2 and avoid detection.

Activation Clustering [66]. Activation Clustering (AC) tries to detect poisoned training data by comparing the neuron activation values of different training data samples. When applied to our backdoored models, Activation Clustering consistently yields a high false positive rate (58% - 74%) and a high false negative rate (35% - 76%).

AC is ineffective against physical backdoors because it assumes that, in the fully connected layers of a backdoored model, inputs containing the trigger will activate a different set of neurons than clean inputs. However, we find that *this assumption does not hold for physical triggers*: the set of neurons activated by inputs with physical triggers overlaps significantly with those activated by clean inputs. In Table 6 in Supp, we show high Pearson correlations of neuron activation values between clean inputs and physical-backdoored inputs, computed from activation values of

Trigger \ Defense	NC [338]	Spectral [323]	AC [66]	STRIP [118]
Dots	60%	$44 \pm 10\%$	$43 \pm 26\%$	$34 \pm 14\%$
Sunglasses	10%	$41 \pm 7\%$	$47 \pm 30\%$	$41 \pm 24\%$
Tattoo Outline	0%	$43 \pm 6\%$	$54 \pm 25\%$	$11 \pm 7\%$
Tattoo Filled-in	0%	$44 \pm 7\%$	$48 \pm 24\%$	$21 \pm 12\%$
White Tape	30%	$41 \pm 8\%$	$41 \pm 31\%$	$39 \pm 17\%$
Bandana	0%	$45 \pm 9\%$	$42 \pm 17\%$	$39 \pm 18\%$

Table 5.4: *Physical backdoor detection rates for four defenses. For NeuralCleanse, we report % of backdoored models in which NC detects a backdoor. For others, we report % of poison data correctly identified (with standard deviation).*

our backdoored models. We believe high correlation values (0.33-0.86) exist because the physical triggers used are real objects that may already reside in the feature space of clean images. Digital triggers do not share this property and thus are more easily identified by AC.

Spectral Signatures [323]. Spectral Signatures tries to detect poisoned samples in training data by examining statistical patterns in internal model behavior. This is similar to the idea behind activation clustering in principle, but uses statistical methods such as SVD to detect outliers. Our results in Table 5.4 show that this defense detects only around 40% of physically poisoned training data. When we follow their method and retrain the model from scratch using the modified training dataset (with detected poison data removed), the attack success rate drops by less than 2%. Thus the real-world impact on physical backdoor attacks is minimal.

STRIP [118]. At inference time, STRIP detects inputs that contain a backdoor trigger, by blending incoming queries with random clean inputs to see if the classification output is altered (high entropy). We configure STRIP's backdoor detection threshold for a 5% false positive rate (based on [118]). When applied to our backdoored models, STRIP misses a large portion of inputs containing triggers (see Table 5.4).

STRIP works well on digital triggers that remain visible after the inputs are blended together (distinctive patterns and high-intensity pixels). It is ineffective against physical triggers because physical triggers are less visible when combined with another image using STRIP's blending al-

gorithm. Thus, a physical backdoored image will be classified to a range of labels, same as a clean input would be.

5.10 Conclusion

Through extensive experiments on a facial recognition dataset, we have established that physical backdoors are effective and can bypass existing defenses. This validates the possibility of using backdoor attacks as a data agency tool to evade unwanted recognition by real-world deep learning systems. However, we acknowledge that some such systems are security or mission-critical, and such evasion is not always welcome. This highlights the need to think critically about the purpose and legitimacy of deep learning systems. For those systems which require protection due to their critical nature, we urge the community to consider physical backdoors as a serious threat in any real world context, and to continue efforts to develop more defenses against backdoor attacks that provide robustness against physical triggers.

CHAPTER 6

A FRAMEWORK FOR REASONING ABOUT DATA AGENCY AGAINST UNWANTED FACIAL RECOGNITION

6.1 Introduction

In recent years, facial recognition systems have accelerated their growth in scale and reach, and are becoming an increasingly ubiquitous part of our daily lives. As a result, the majority of citizens in the world's most populous countries are already enrolled in one or more facial recognition systems, whether they know it or not. For example, in the United States, nearly 200 million residents are already enrolled in the FBI's facial recognition database, which was built by leveraging FBI's access to driver license photos in many states [122]. In China, a well-known surveillance system uses facial recognition to monitor civilian behavior and enforce the social credit score system [78, 230]. In Russia, authorities acquired 100,000+ cameras in Moscow to build a facial recognition-based COVID quarantine enforcement system [264]. Beyond government use cases, facial recognition systems are now regularly used for myriad purposes, including authenticating travelers at airports and employees entering corporate offices.

The advancements that paved the way to real-world facial recognition systems have also opened the door to their potential misuse and abuse. With moderate resources, an individual or institution, public or private, can now extract training data from social media and online sources to build facial recognition models capable of recognizing large groups of users. In 2020, New York Times journalist Kashmir Hill confirmed the potential for facial recognition misuse when she profiled Clearview.AI, a private for-profit company that scraped over 3 billion images from "public sources" to build a facial recognition system that recognized hundreds of millions of private citizens [144], without their knowledge or consent. Clearview and companies like it could enable surveillance and tracking by anyone willing to pay[1]. Other reports have detailed how photos taken in unexpected

1. Multiple countries are pursuing inquiries into Clearview's business model, and Canada has already denounced

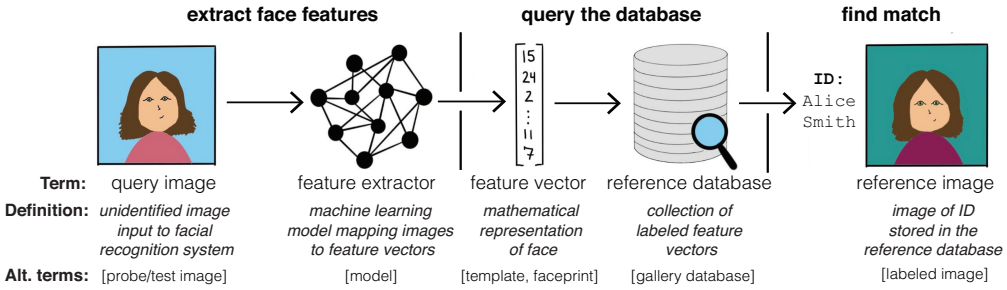


Figure 6.1: *The workflow of how facial recognition systems recognize a human face in an input image, along with the corresponding terminology. (a): A query image, after being submitted to the system, is passed to the feature extractor to produce a feature vector; (b): this feature vector is used to query a reference database of labeled feature vectors; (c): if the query feature vector matches a labeled feature vector in the database, the label is used to find a reference image, and the system outputs the reference image and the identity (i.e. Alice Smith in this example).*

places – airports, city streets, government buildings, schools, corporate offices – end up in facial recognition systems without subjects' knowledge or consent [122, 86, 130, 123, 241, 28].

Despite backlash against intrusive facial recognition systems [163, 172, 25, 31], there are few commercial or legislative tools available to protect users against them. While big tech has begun to self-regulate [268] and openly called for legislation (e.g., [163, 172]), legislative efforts to regulate facial recognition remain scarce. In their place, a cottage industry of anti-facial recognition (AFR) tools has emerged to provide individuals with data agency against such systems. AFR tools are designed to target different parts of facial recognition systems, from data collection, model training to run-time inference, with the unified goal of preventing successful recognition by unwanted or unauthorized models.

AFR tools have also attracted significant attention from the research community. In the last 12 months, more than a dozen AFR tools have been proposed (e.g., [190, 345, 287, 151, 108, 77, 79, 353, 324, 356, 64, 358, 235, 171, 52, 334]). While most are constrained to research prototypes, a few of these tools have produced public software releases and gained significant media attention [287, 77, 139].

---

it as "illegal" [145].

Proposals in the rapidly growing collection of AFR tools differ widely in their assumptions and techniques, and target different pieces of the facial recognition pipeline. There is a need to better understand their commonalities, to highlight performance tradeoffs, and to identify unexplored areas for future development. Existing surveys [220, 240] on facial privacy issues do not consider user-centric AFR tools. They instead discuss privacy-preserving techniques that surveillance system operators could employ, a related but separate line of work to that addressed here (see §6.12).

In this paper, we address this need by providing a common framework for analyzing a wide range of AFR systems. More specifically, we make the following contributions:

- **Taxonomy of targets in facial recognition systems:** AFR systems target a wide range of components in the facial recognition process. Using a generalized version of the FR data pipeline, we provide a framework for reasoning broadly about existing and future AFR work.
- **Categorization and analysis of AFR tools:** We take the current body of work on AFR tools, and categorize and analyze them using our proposed framework.
- **Mapping design space based on desired properties:** We identify a core set of key properties that future AFR systems might optimize for in their design, and provide a design roadmap by discussing how and if such properties can be achieved by AFR systems that target each stage in our design framework.
- **Open challenges:** We use our framework to identify significant challenges facing current AFR systems, as well as directions for potential solutions.

The rest of the paper proceeds as follows. We begin by providing operational details of real-world facial recognition systems (§6.2), including real-world deployment scenarios and key technical components. We then present the motivation and threat model of AFR tools (§6.3), and our framework for analyzing existing AFR tools (§6.4). We then discuss existing AFR proposals targeting each stage, i.e., *data collection* (§6.5), *data processing* (§6.6), *feature extractor training* (§6.7), *identity creation* (§6.8), and *query matching* (§6.9). Finally, we identify desirable properties of effective AFR systems, and map them to points in the design space (§6.10). Finally, we discuss

challenges and directions for AFR research (§6.11).

Unresolved Ethical Questions: The broad deployment of facial recognition systems (and by extension, AFR systems) is fraught with ethical challenges, not the least of which are significant biases against women and people of color [53]. While we discuss ethical tensions surrounding AFR systems in §6.11.2, we do not make assertions about how (and whether) AFR tools should be used. Development and adoption of AFR tools are driven by backlash against biased and misused facial recognition systems. Though their legal and ethical implications are yet-unknown, we believe that AFR tools are here to stay. Consequently, an analysis of their strengths and limitations is crucial to advancing the ongoing debate about their use and the place of facial recognition in our world.

Connections to Data Agency in Other Domains. Although the taxonomy developed here focuses specifically on facial recognition and AFR tools, we believe this framework can serve as a helpful starting point for reasoning more broadly about data agency. For example, the stage-based framework used to categorize AFR solutions has analogues in other machine learning contexts, given models' common needs for data collection, processing, training and deployment. Consequently, this framework lays the foundation for discussions of data agency in other settings.

6.2 Facial Recognition: Terminology, Design Stages and Deployment

To provide context for later discussions, we now give a high-level overview of today's facial recognition (FR) systems and their real-world implementations. Our goal is to describe modern FR systems targeted by today's AFR systems, their key operational stages, and how these FR systems are being deployed around the globe. Together, these provide a framework that we will use for analyzing AFR systems later in §6.4, by examining critical points of direct interaction between users and FR systems.

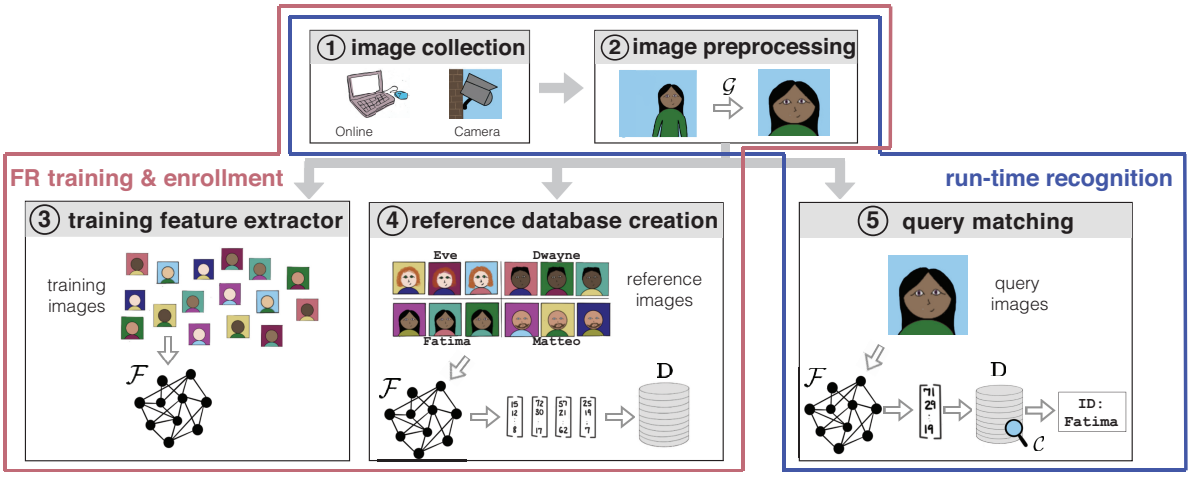


Figure 6.2: *We propose to divide the operational pipeline of a FR system* $\mathbb{F}=\{\mathcal{G},\mathcal{F},\mathcal{C},\mathbf{D}\}$ *into a set of five operational stages* ① *to* ⑤*. They encompass the five critical points of direction interaction between users and FR systems. Later we will use this framework for analyzing AFR systems.*

6.2.1 Modern FR Systems

FR systems identify people by their facial characteristics, generally by comparing an unknown face in an image (or a video) against a database of known faces. The technology has evolved significantly over the past two decades, resulting in many design variants [317]. Today, the state-of-the-art and widely adopted FR systems employ deep neural networks (DNNs) to extract unique features from a given face (see §2.1.1 for more information about DNNs). Since existing AFR systems mainly target these modern FR systems, this work focuses on DNN-based FR/AFR systems. The main differences between older and newer FR methods lie in (1) the feature extraction methods (e.g. statistical methods like PCA or LDA [328, 44] vs. DNN-based feature extractors) and (2) scale However, the fundamental FR stages remain the same in both older and newer FR systems – face images must still be collected, processed, and recognized. Consequently, the framework laid out in this paper could be applied to older FR/AFR systems if desired.

Terminology. In this paper, we represent a modern FR system as $\mathbb{F} = \{\mathcal{G},\mathcal{F},\mathcal{C},\mathbf{D}\}$, whose goal is to associate a query image x_I with its true identity I. Specifically,

- *Query image* (x_I): a face image to be identified by $\mathbb{F}$.

- *Preprocessing engine* ($\mathcal{G}$): a processing function that prepares raw face images for the FR task, e.g., detecting and cropping out individual faces.
- *Face feature extractor* ($\mathcal{F}$): a DNN that converts a face image into a *feature vector*, a mathematical representation of the person's unique facial features.
- *Reference database* ($\mathbf{D}$): a collection of face images and their feature vectors of known identities, e.g., x_I^R (ground truth images of user I) and $\mathcal{F}(\mathcal{G}(x_I^R)) = v_I$.
- *Run-time face classifier* ($\mathcal{C}$): this function runs a query search to match the query image $\mathcal{F}(\mathcal{G}(x_I))$ against $\mathbf{D}$. If the closest feature vector v_I is sufficiently similar, then the query image is identified as I. Ideally, it should produce $\mathcal{C}(\mathcal{F}(\mathcal{G}(x_I)), \mathbf{D}) = \mathcal{C}(\mathcal{F}(\mathcal{G}(x_I^R)), \mathbf{D})$, where x_I is a previously unidentified image of I and x_I^R is a ground-truth reference image of I.

It should be noted that the terminology used to describe a FR system can vary across the literature. We list some alternative terms in Figure 6.1. The terms we choose to use in this paper are, we believe, most familiar to the security community.

Face recognition vs. face verification. Here, we note the distinction between *face recognition* and *face verification*. Face verification is widely used to authenticate users on mobile devices (e.g., FaceID on iPhones) by comparing a user's face feature to the stored face feature of the authorized user. While the two systems apply similar techniques to analyze face images, facial verification systems require *user consent* for deployment while many FR systems operate without user consent. As such, most AFR systems target FR systems rather than face verification systems. *Furthermore, face verification systems can be viewed as a special case of FR systems, where the reference database only has a single user.* Therefore, in this paper we do not explicitly consider facial verification or its disruption.

6.2.2 Breaking FR into Operational Stages

We now examine the FR operational pipeline and divide it into a set of *operational stages* to help frame our discussion of AFR tools. These operational stages correspond to specific subtasks in

FR, which encompass *the five critical points of direct interaction between users and FR systems.* Figure 6.2 depicts the five operational stages of a FR system $\mathbb{F} = \{\mathcal{G}, \mathcal{F}, \mathcal{C}, \mathbf{D}\}$. We discuss each stage below and revisit them as a framework to analyze AFR tools in §6.4.

The overall operation of a FR system includes two phases: a *training & enrollment* phase where the system builds (or acquires) a face feature extractor and creates a reference database of known identities, and a *run-time recognition* phase where the FR system identifies an unknown face. As we show below, the training & enrollment phase employs stage ①–④, while the recognition phase employs stage ①, ② and ⑤.

Stage ①: collecting face images. Face images primarily come from two sources: online image scraping [142] or physically taking a photo of a person [122, 130]. We discuss sources of face images for FR systems in further detail in §6.2.3.

Stage ②: preprocessing raw face images via $\mathcal{G}$. Raw images obtained from stage ① are often poorly structured (e.g., varying face sizes, bystanders in background). To make downstream tasks easier, $\mathbb{F}$ employs an image preprocessing engine $\mathcal{G}$ that uses face detection (e.g., automated face cropper [376]) to remove background and extract each individual face, followed by a data normalization process [92, 342, 281].

Stage ③: training a feature extractor $\mathcal{F}$. The crucial element of DNN-based FR systems is the feature extractor $\mathcal{F}$ used to compute facial features from an image. To achieve accurate recognition, the computed feature vectors must be highly similar for photos of the same person, but sufficiently dissimilar across photos of different people. To enable this behavior, most existing FR systems adopt the training methodology proposed by [281] in 2015: adding an *extra* loss function during $\mathcal{F}$ training to directly optimize for large separations between different faces in the feature space. Followup works explore alternative loss functions and architectures to further improve the accuracy of FR systems (e.g., [92, 342, 223]).

To maximize efficacy, $\mathcal{F}$ is generally trained on millions of labeled face images. Extensive resources are required to both collect and label a large face dataset and to actually train the model.

As a result, many FR practitioners, including large companies [132] and government agencies [19, 120], opt to purchase or license a well-trained feature extractor (e.g. [6, 3, 2, 1, 7, 10, 14, 4]). We refer to images used in stage ③ as *training images*.

Stage ④: creating a reference database $\mathbf{D}$. FR systems need a large database of known (labeled) faces in order to identify unknown (unlabeled) faces. As a result, FR operators must build a reference database $\mathbf{D}$ of people they want to recognize, by first collecting and preprocessing labeled face images of these individuals, and then passing them to $\mathcal{F}$ to obtain feature vectors. The reference database holds the (feature vector, identity) pairs [253, 142, 159]. We refer to images used in stage ④ as *reference images*.

Stage ⑤: recognize the face in a query image via $\mathcal{C}$. At run-time, the FR system takes in and preprocesses (via $\mathcal{G}$) a query image (i.e., an unidentified face image), extracts its feature vector via $\mathcal{F}$, and queries the reference database $\mathbf{D}$ to locate a match (if any). If the feature space distance (e.g., L_2 or cosine) of the query image is sufficiently close to a stored entry in $\mathbf{D}$, the system outputs a match. In this paper, we represent this process by the classifier $\mathcal{C}$.

6.2.3 FR Deployment and Data collection

In recent years, entities across the globe have adopted and deployed FR systems for various applications. This wide adoption was triggered by significant accuracy improvements of FR systems, largely due to new training methods [281] and more powerful neural network architectures [311]. Deployment scenarios of these FR systems, along their data sources, have informed AFR tool development. Thus, to contextualize AFR proposals, we briefly examine how FR is used in the real world and from where its images (i.e., training/reference/query images) are drawn.

Deployment scenarios. Both public and private entities use FR for a variety of purposes. We list some examples in Table 6.1. Public (e.g., government-based) FR use cases range from criminal

Location	Use Cases Reported	Countries/Companies
Public spaces	On-street surveillance	Bahrain [210], China [78], England [31], France [168], Kenya [54], Myanmar [254], Russia [264], UAE [210], UK [117], US [123], Zimbabwe [165]
	Criminal suspect identification	Argentina [221], Belarus [254], Brazil [96], China [229], Greece [254], Malaysia [315], US [122]
	School monitoring	Brazil [215], China [36], India [28], Russia [215], US [215]
	Border security	Israel [384], Pakistan [169], US [216]
	COVID lockdown enforcement	China [18], India [271], South Korea [271], Russia [264]
Privatized spaces	Catching shoplifters	Apple, Macy's, Lowe's [81, 30]
	Securing facility access	Alibaba [274], Intel [269]
	Tracking driver behavior	Hyundai [21], Subaru [207]
	Airline passenger check-in	JetBlue [298], Delta [24]

Table 6.1: *Example use cases of facial recognition.*

identification[2][121], civilian surveillance [78, 20] and border control [216], to video game use tracking [217] and COVID lockdown enforcement [18]. For a broader exploration of government uses of facial recognition, we refer the reader to [23]. Private entities have also integrated FR into their security and commerce pipelines. The most common private FR use cases are enhancing store or office security, but other examples abound (see Table 6.1).

Sources of face images. The definitive source of images for deployed FR models is often unknown. Based on government reports and media articles, we outline some known sources of training, reference, and query images used today.

Training images (used to train the feature extractor $\mathcal{F}$) often come from a mix of academic training datasets (e.g., [135, 56, 234, 369]), proprietary data, and public data scraped from social

2. Recently, police departments around the US have drawn fire for their use of highly unregulated FR software like Clearview.ai [3].

Operator of FR system	Source of reference images
Clearview.ai	Social media photos [144]
PimEyes	(Public) online photos [8]
FBI F.A.C.E.S.	State drivers' license photos [122]
US Customs and Border Patrol	Passport photos [216]
Skynet (China)	National ID photos [55, 230]

Table 6.2: *Reported reference image sources*

media accounts, according to a report of the US Government Accountability Office [19].

Reference images (used to create the reference database) generally come from the Internet (e.g., social media), or government databases (e.g., passport and driver license photos). Table 6.2 shows a list of known reference image sources for some well-known FR operators.

Query images, or faces to be identified by the FR system at run-time, can come from both online and physical sources. Some known sources include social media, police body cams, mug shots, corporate surveillance systems, state ID images, and passport photos [120]. After identification, query images are sometimes fed back into the reference database, either to enhance existing feature vectors or to create new ones. For example, US Customs and Border Patrol states that images of non-US travelers collected at US entry points are fed back into a larger DHS database as reference images [119]. Similar techniques are used by several Chinese companies [26, 241].

6.3 Anti-Facial Recognition: Motivation and Threat Model

In this section, we discuss factors driving the development of anti-facial recognition (AFR) tools, the threat model of those AFR tools, and its practical implications.

6.3.1 The Rise of AFR Tools

Numerous forces have coalesced to drive the recent trend in AFR tool development. *First*, numerous reports about the provenance of images used in commercial FR systems have raised significant privacy concerns. The most infamous examples are Clearview.ai and PimEyes – both companies

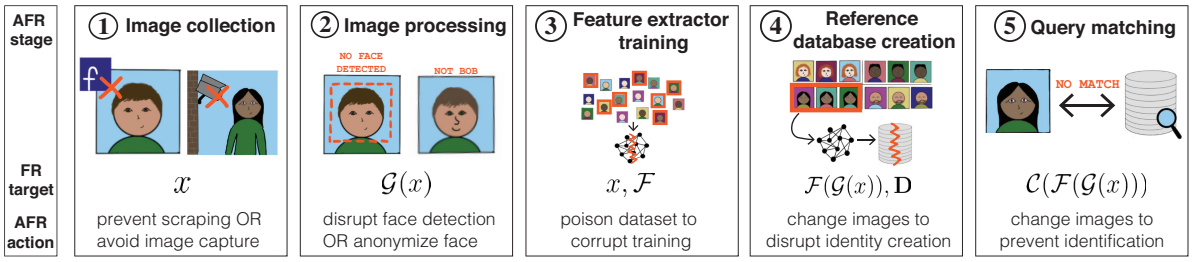


Figure 6.3: *Overview of our proposed stage-based framework for analyzing existing AFR proposals. We list the five critical stages of facial recognition as discussed in §6.2.2 and present AFR strategies per stage by the attack target, action, and desired effect.*

have scraped over *3 billion images* from social media sites to use in their FR systems [144] without user knowledge or consent. *Second*, increased government use of FR systems has caught the attention of citizens who have raised significant concerns about the long-term effects of FR on privacy and freedom of expression [31, 212]. *Third*, multiple editorials have highlighted and discussed the demographic bias of existing FR systems, calling for a moratorium on (or at least regulation of) the FR technology [172, 115, 95].

Consequently, public sentiment about FR is mixed and, especially in western countries, trending negative [299, 156, 304, 180]. This shift in public opinion, combined with the forces noted above, has motivated researchers to create various AFR tools to counteract unwanted FR systems.

6.3.2 *Threat Model of AFR*

AFR tools are used by a person P to combat a FR system $\mathbb{F} = \{\mathcal{G}, \mathcal{F}, \mathcal{C}, \mathbf{D}\}$. In this context, P takes the role of an attacker and acts against $\mathbb{F}$. P's goal is to prevent recognition by $\mathbb{F}$, i.e., given an image x_P of P, a successful AFR tool should cause $\mathbb{F}$ to produce $C(F(G(x_P))) \neq P$.

Proposed AFR tools generally make the following assumptions about each party:

- P has no special access to or authority over $\mathbb{F}$, but wishes to evade unwanted identification by modifying or otherwise controlling their own face images.
- P wishes to avoid facial recognition, but also may wish for their images to remain useful for other purposes. For example, if P posts a headshot on a personal website, they would like to ensure that the headshot not be scraped and used in a FR engine but also that their face remains

recognizable to website visitors. Thus, P prefers AFR tools which maximize AFR protection while minimizing image disruption.

- $\mathbb{F}$'s goal is to either create or maintain an accurate facial recognition operation. Furthermore, $\mathbb{F}$ operates *at scale* and does not specifically target P for identification.

Implications. We also explore the real-world implications of the above threat model.

(1) *Assuming AFR tools operate on images* – Our study focuses exclusively on image-based AFR tools that a user P can deploy on their own. These image-based designs, which operate either directly on images or on systems that collect/process images, dominate the current set of AFR proposals. On the other hand, a user P may, depending on the context, be able to use other means (e.g., legal action) to fight unwanted facial recognition.

(2) *Assuming $\mathbb{F}$ does not specifically target P for recognition* – Existing AFR tools are designed to fight large-scale FR systems. This is because, from a practical standpoint, if system $\mathbb{F}$ wishes to specifically recognize a user P, there are much more efficient options than using a general, large-scale FR system. Therefore, most current AFR tools are not designed to withstand this level of scrutiny. If $\mathbb{F}$ makes a more targeted effort to identify P, such as hiring a private investigator, current AFR tools will likely fail.

(3) *Assuming AFR tools minimize perturbations.* This study focuses on AFR tools which introduce *minimal perturbations* to images (as measured by L_P norms). This decision is grounded in prior work showing that users are more likely to use privacy-preserving tools with minimal overhead [64, 84, 350]. AFR tools which do not seek to minimize perturbations are not addressed in this work. This is a limitation of our work, and future work should consider AFR tools which use metrics beyond L_P norms to measure image distortion.

6.4 A Stage-Based AFR Framework

We now discuss and analyze existing AFR proposals. To do so, we propose and use a *stage-based* framework to categorize AFR strategies. As discussed in §6.2, a FR system $\mathbb{F} = \{\mathcal{G}, \mathcal{F}, \mathcal{C}, \mathbf{D}\}$

AFR system	Year released	Stage targeted	Attack scenario				
			P's knowledge of $\mathbb{F}$	*P's operating context*	*Targeted/ Untargeted*	*Tested on real-world FR*	Unique Property
Anti-scraping [98-102]	2021	①	-	Digital	UT	-	Prevent large-scale image scraping
Data Leverage [336]	2021	①	-	Digital	UT	-	Withholds data to prevent collection.
CVDazzle [139]	2010	②a	WB	Physical	UT	-	Make-up
Xu et al. [356]	2020	②a	BB	Physical	UT	YOLOv2	Adversarial patch on T-shirts
Wu et al. [353]	2020	②a	Both	Physical	UT	YOLOv2	Adversarial patch on T-shirts
Zolfi et al. [385]	2020	②a	BB	Physical	UT	YOLOv5	Stickers on camera lens that blur vision
SocialGuard [358]	2020	②a	WB	Digital	UT	-	Adversarial perturbation on face detectors
Hu et al. [68]	2021	②a	WB	Digital	UT	-	Adversarial patch on object detectors
Treu et al. [324]	2021	②a	BB	Digital	UT	-	Adversarial clothing on face detectors
DeepPrivacy [153]	2019	②b	BB	Digital	UT	-	GAN-based face blurring (perceptible)
IdentityDP [345]	2021	②b	BB	Digital	UT	AZ	GAN-based face blurring (perceptible)
DeepBlur [190]	2021	②b	BB	Digital	UT	AZ, F++	GAN-based face blurring (perceptible)
Yang et al [361]	2021	②b	BB	Digital	UT	-	GAN-based face blurring (imperceptible)
Evtimov et al. [107]	2021	③	BB	Digital	UT	-	Data poison by modifying entire dataset
Huang et al. [151]	2021	③	BB	Digital	UT	-	Data poison by user coordination
Fu et al. [116]	2021	③	BB	Digital	UT	-	Data poison by unlearnable data
Fawkes [287]	2020	④	Both	Digital	UT	AR, AZ, F++	Corrupts features of faces
FoggySight [108]	2021	④	Both	Digital	UT	AZ	Collectively corrupts features of faces
LowKey [77]	2021	④	BB	Digital	UT	AR, AZ	Corrupts features of faces
Feng et al. [113]	2013	⑤	BB	Physical	UT	-	Make-up
Sharif et al. [288]	2016	⑤	Both	Both	Both	F++	Adversarial patch on wearable accessories
Dabouei et al. [87]	2018	⑤	WB	Digital	UT	-	Adversarial attack distorts face landmarks.
Zhou et al. [381]	2018	⑤	WB	Physical	Both	-	Projected adversarial IR patterns
Dong et al. [102]	2019	⑤	BB	Digital	T	TN	Black-box adversarial perturbation.
Zhu et al. [383]	2019	⑤	Both	Digital	Both	-	Adds eye makeup with GAN.
AdvHat [171]	2019	⑤	WB	Physical	UT	-	Printed sticker on hat.
AdvFaces [89]	2019	⑤	BB	Digital	Both	-	GAN-based adversarial attack.
VLA [291]	2019	⑤	BB	Physical	Both	-	Projected light patterns
Nguyen et al. [235]	2020	⑤	Both	Physical	Both	?	Projected light patterns
Browne et al. [52]	2020	⑤	BB	Digital	UT	-	Universal adversarial perturbation
Cilloni et al. [79]	2020	⑤	WB	Digital	UT	-	Corrupts features of faces
Face-Off [64]	2020	⑤	BB	Digital	Both	AR, AZ, F++	Study on user perception on perturbation levels.
Singh et al. [296]	2021	⑤	WB	Digital	UT	-	Brightness-agnostic adversarial perturbations
Yang et al [363]	2021	⑤	BB	Digital	UT	TN	Corrupts features of faces

Table 6.3: *Taxonomy of proposed AFR tools. "BB/WB" = Black Box, White Box."UT, T" = Untargeted, Targeted. "AR, AZ, F++, TN" = Amazon Rekognition, Microsoft Azure Face Recognition, Megvii's Face++, Tencent Face Recognition.*

operates in 5 distinct stages that correspond real-world actions (e.g. image capture, pre-processing, feature extraction, etc.). In each stage, $\mathbb{F}$ interfaces with P and the broader world as it collects, processes, and uses image data. Such interfaces represent possible points at which P can act against $\mathbb{F}$. Specifically, an AFR tool can attack the component(s) of $\mathbb{F}$ relevant to any of the 5 stages, including images x, preprocessor $\mathcal{G}$, feature extractor $\mathcal{F}$, reference database $\mathbf{D}$, and classifier $\mathcal{C}$. With this in mind, Figure 6.3 demonstrates the attack actions and goals when AFR tools target each FR stage.

6.4.1 AFR Strategies per Stage

Since the five FR stages ①–⑤ encompass the points of direct interaction between P and $\mathbb{F}$, they naturally cover the points of attack employed by existing AFR proposals. Next we briefly describe the general strategies used by AFR tools targeting each FR stage.

Attacking ①. In the image collection stage, labeled and/or unlabeled images x are collected for use by $\mathbb{F}$, either by physically taking photos or scraping online images. Labelled images can be used as training or reference images to build a FR system, while unlabelled images can be used as query images. When targeting this stage, AFR tools focus on disrupting the data collection process to prevent $\mathbb{F}$ from acquiring usable face images x_P of P.

Attacking ②. In the second stage, $\mathbb{F}$ uses $\mathcal{G}$ to pre-process collected face images using a series of digital transformations, e.g., face detection, background cropping, and normalization. AFR tools deployed at this stage target $\mathcal{G}$ to render the processed images unusable, either by breaking the preprocessing functions (e.g., preventing faces from being detected) by injecting noise and artifacts onto the images or removing P's identity information from the images. We denote these different actions as ②a and ②b, respectively.

Attacking ③. Since stage ③ is dedicated to training $\mathbb{F}$'s feature extractor $\mathcal{F}$, AFR tools targeting this stage seek to degrade the accuracy of $\mathcal{F}$ by poisoning its training images.

Attacking ④. To create the reference database used by classifier $\mathcal{C}$, labeled reference images are passed through $\mathcal{F}$ to create their feature vectors. AFR tools targeting this stage attempt to corrupt the feature vectors created for P's reference images so that the database holds a "wrong" feature vector of P, and $\mathcal{C}$ fails.

Attacking ⑤. In the query matching stage, AFR tools seek to prevent classifier $\mathcal{C}$ from accurately matching query image x_P's feature vector and P's feature vectors stored in $\mathbb{F}$'s reference database. This is generally achieved by perturbing (or modifying) the query image to change its feature vector and thwart $\mathcal{C}$.

6.4.2 *Taxonomy of Existing AFR Proposals*

Using our stage-based analysis framework, we now present a comprehensive taxonomy of existing AFR proposals in Table 6.3. In this list, we categorize existing AFR proposals by the year of release, the individual FR stage they target, and the attack scenario. We further break down the attack scenario by P's knowledge of $\mathbb{F}$ (white box or black box[3]), the AFR deployment context (physical or digital), whether the attack is targeted or untargeted[4], whether the AFR tool has been tested against real-world FR systems, and any unique or notable features of the AFR tool. We note that the majority of AFR tool users may not care for or need a *targeted* AFR misclassification result, but we include targeted attacks for completeness, as they represent the most user-controlled version of an AFR tool.

There is a significant imbalance of AFR tools targeting different stages. Stage ② and ⑤ have attracted the most number of AFR proposals, likely due to the popularity of adversarial perturbation research. We also notice that 7 out of 30 proposals assume a "white-box" access to $\mathbb{F}$'s FR pipeline, which is often unrealistic in practice. Finally, only 12 out of the 30 proposals have tested the AFR effectiveness against at least one real-world FR system. Overall, Table 6.3 serves as a comprehensive summary of current AFR proposals, which we will refer to throughout the paper.

Adversarial ML and AFR. A significant portion of AFR tools listed in Table 6.3, e.g., those targeting ②a and ③ − ⑤, apply adversarial machine learning (AML) techniques like poisoning or evasion attacks to thwart $\mathbb{F}$. Consequently, a significant portion of this paper is devoted to discussing the pros and cons of AML-based approaches to AFR. On the other hand, since AFR tools targeting stage ① and ②b are inherently non-AML based, our analysis is not limited to only AML-enabled AFR tools. Since each of the five FR stages represents a viable attack vector for AFR tools, our analysis covers AFR tools targeting any stage.

3. *White box* means P has full access to $\mathbb{F}$'s FR system (including feature extractor parameters) and uses this knowledge to guide their AFR protection. *Black box* means P lacks such access and knowledge.

4. A *targeted* attack causes the FR system to identify P as a specific, incorrect person (e.g. a famous politician). An *untargeted* attack means that P is misclassified, but not as a specific person.

6.4.3 Roadmap of Our Analysis

Using the stage-based framework, we conduct a detailed analysis of existing AFR proposals. First, we discuss in greater detail how existing AFR proposals attack each of the five stages (§6.5 – §6.9). In each section, we describe the goals of $\mathbb{F}$ and P, the challenges of targeting this particular stage, the existing proposals, and a summary of the progress made in this direction. Next, we conduct a meta-analysis of AFR strategies across the five stages (§6.10), and discuss what we see as the major technical and broader social/ethical challenges facing future AFR development (§6.11).

6.5 Attacking Stage 1 to Disrupt Data Collection

We start by examining AFR methods that allow P to attack $\mathbb{F}$ by disrupting the process of face data collection.

Goals and Challenges. In this data collection stage, $\mathbb{F}$'s goal is to obtain usable face images x from online or physical sources. Often, $\mathbb{F}$ aims to collect high quality images of millions or billions of people (e.g., Clearview.ai [144]). $\mathbb{F}$ uses labeled images to build the reference database and/or train the feature extractor. By using AFR tools, P's goal is to prevent their face images x_P from being collected for use by $\mathbb{F}$, either as (labeled) reference images or (unlabeled) query images. They can apply online or physical evasion/disruption techniques to do so.

The key challenges facing AFR proposals targeting this stage are that (1) they need to be aware of and adapt to $\mathbb{F}$ who continues to innovate techniques for data collection; and (2) cameras are widely deployed in the real-world, making it challenging to avoid image capture by cameras.

6.5.1 Current Solutions

Face images can come from two sources: scraping online images or physically capturing faces using cameras. Image scraping refers to the collection of images posted online that were captured by someone who is not the data collector (e.g. camera is operated by the subject, subject's friend,

etc). Picture-taking refers to images taken directly by the data collector. Thus we divide AFR tools acting at this stage into two categories: preventing scraping and preventing capture.

Preventing Online Image Scraping. A large portion of face images used in today's FR systems are scraped from online social media platforms. Thus, an effective way to stop $\mathbb{F}$ is to prevent web scraping. While each single user can try their best to limit their online footprint, most AFR methods require an online platform (e.g., Flickr) or outside help.

- *Anti-scraping by online platforms.* Anti-scraping techniques have been widely studied in the security community [341, 126, 246, 162, 138]. Techniques such as rate limits, data limits, ML-based scraping detection are already used by online platforms [80]. However, a significant portion of scraping still goes undetected as scrapers develop more sophisticated tools to bypass detection [80].
- *Data leverage by users.* P could try to prevent $\mathbb{F}$ from collecting their online images by withholding them. Recent works propose the concept of "data leverage" where users of online platforms work collectively to withhold data or control how their data is used by tech companies [336, 334, 335]. While not specifically aimed at facial recognition, these proposals offer alternative models for online engagement while protecting user data.

Avoiding Image Capture. Ordinary civilians can already use smartphones to take high-quality photos of anyone at any moment. These photos could be collected and used by facial recognition systems like PimEyes [8]. Furthermore, face photos taken by on-street surveillance cameras are increasingly used by commercial or government facial recognition systems [269, 81, 267, 122, 123], especially in major metropolitan areas and inside stores. Today's proposals for avoiding image capture come from both research community and activists (e.g. protesters and artists) concerned about surveillance. They fall into two categories: *hiding faces from cameras* and *disrupting camera operation*.

- *Face hiding.* People can wear clothes, hats, masks, or move their head to prevent (usable) facial image being captured by cameras. Notably, during the June 2020 wave of protests in the US,

nonprofit organizations compiled a "tech toolkit" to help privacy-conscious protesters obfuscate their faces from cameras and avoid identification [29]; in late 2020, a Chinese artist used a map of on-street surveillance cameras to successfully guide others to evade identification by positioning their head/body "away" from those cameras [236].

- *Camera disruption.* Without physically breaking cameras, human users can prevent cameras from capturing (usable) images by simply shining laser lights at them [212]. Other methods include covering cameras with fabric or stickers.

6.5.2 Discussion of Stage 1 Solutions

Privacy/Utility Trade-offs. Evading data collection requires both fine-grained control over one's online identity and awareness of when/how pictures are being taken, making it difficult for users to deploy these tools without significantly limiting either their online or physical activities. Furthermore, there are cases where evading data collection is simply impossible, i.e. mandatory pictures posted on an employer's website. Anti-scraping tools can also decrease the utility of the service provider, as such tools can have false positives and a high deployment cost.

Summary of Progress. Existing AFR proposals against stage ① make headway in addressing key challenges. Adaptive anti-scraping techniques like [80] definitely raise the bar for attackers. Furthermore, anti-data collection methods like [236] have shown that it is possible, with careful action, to evade image capture even in robust surveillance systems. Future AFR development against stage ① can seek to improve both data controls and camera awareness by individual users.

6.6 Attacking Stage 2 to Disrupt Face Pre-processing

In stage 2, $\mathbb{F}$ processes raw face images with $\mathcal{G}$ to facilitate further operations in stages ③, ④, and ⑤. AFR proposals targeting this stage seek to disrupt $\mathcal{G}$ by transforming an image x into x' such that processed face images $\mathcal{G}(x')$ are "unusable" by subsequent FR stages.

Goals and Challenges. $\mathbb{F}$'s goal is to use $\mathcal{G}$ to obtain well-structured face images from many raw images. P's goal is to either prevent their face being detected/extracted from raw images by $\mathcal{G}$ or to anonymize their face in these images.

The main challenge for AFR proposals targeting stage 2 is how to achieve *anonymization without distortion*. That is, when modifying P's images to either evade $\mathcal{G}$'s face detection or to remove identity information, the modified images should still resemble P to remain *useful* to P. An additional challenge is overcoming (adaptive) defenses deployed by $\mathbb{F}$ to protect $\mathcal{G}$.

6.6.1 Current Solutions

Preventing Face Detection. Face detection extracts well-centered headshots from raw images. The commonly used face detection systems [376] rely on DNNs to accurately infer the location of faces in an image. To prevent effective face detection/extraction, the AFR goal is to produce an adversarial x' such that $\mathcal{G}(x') = z$, where z is a useless result that cannot be passed on to $\mathcal{F}$. To create x', existing AFR tools leverage "adversarial perturbations" against DNN models. Adversarial perturbations are a well-studied phenomenon in the field of adversarial machine learning. These carefully crafted, pixel-based perturbations, when added to an image, can cause DNNs to produce wrong classification results (e.g., [211, 63, 72, 43]). Perturbations are generated using an iterative optimization procedure that maximizes the likelihood of model misbehavior while minimizing perturbation visibility. The generation procedure varies depending on P's knowledge of $\mathbb{F}$ (e.g. white vs. black box, see Table 6.3).

AFR tools using adversarial perturbations can be subdivided based on how the perturbation is added to images. They can be directly added to digital images if P has direct access to these images or fabricated as physical objects that P can wear (e.g., an adversarial T-shirt) or place on cameras.

- *Directly modifying digital images.* Using AFR tools, users who post images online can directly add adversarial perturbations to these images before posting them (e.g., [358, 324, 68]). Prop-

erly perturbed images cannot be used by FR systems to extract any face information.

- *Wearing custom designed physical objects.* Often users do not have access to face images to modify them. An alternative way to "inject" adversarial perturbations into images is to carry or wear a physical object so that any camera taking a photo of the user will also capture a version of the adversarial perturbation. Along these lines, prior works have successfully translated face-detection-evading adversarial perturbations into makeup [139, 29], t-shirts [356, 353], or stickers.
- *Placing a sticker on cameras.* An orthogonal approach involves transforming the adversarial perturbation into a translucent sticker that can be placed over a camera lens. This sticker imperceptibly modifies images taken by the camera to prevent people and faces from being detected in those images [385].

Anonymizing Faces (2b). P can also *anonymize* their face images to remove identity information. In this setting, P creates an x' such $\mathcal{G}(x') \neq \mathcal{G}(x)$, i.e. the result after processing is still usable but represents a fake identity. *Physical anonymization* can be easily achieved by wearing masks, hats, makeup, etc, which overlaps with "avoiding image capture" in ① discussed in §6.5. Leaving proposals for *digital anonymization* use generative adversarial networks (GANs) [128] and differential privacy [106]. Several such proposals use GANs to transform face images into latent space vectors, modify those vectors to remove identity information, and reconstruct the images from the modified vectors [153, 190, 361]. The modified faces still look human but are anonymized to prevent accurate identification. Another proposal, IdentityDP [345], uses similar techniques but also claim to provide differentially private identity protection.

6.6.2 Discussion of Stage 2 Solutions

Privacy/Utility Trade-offs. Many stage (2a) proposals address the "usability" challenge by formatting adversarial patches against $\mathcal{G}$ as wearable clothing/objects. However, wearing this special clothing, which can appear bizarre, may not be desirable for the average person. Current propos-

als against ②b provide anonymity but tend to produce anonymized faces that do not resemble the original face, with significantly altered shape, skin tone, hair color, etc. These images lack many functionalities of traditional images, e.g. image sharing, preserving memories, etc.

Summary of Progress. Many AFR proposals targeting ②a have been tested against real-world object detectors like YOLOv2, demonstrating their real-world efficacy. However, several defenses against these patches have emerged recently (e.g., [195, 199, 354]), although only one [195] has been tested against physical adversarial patches like the ones used by AFR tools [356, 353]. Further work is needed to determine if AFR proposals against ②a can resist these defenses that $\mathbb{F}$ can use to protect $\mathcal{G}$.

6.7 Attacking Stage ③ to Corrupt Feature Extractor

All FR systems require an effective feature extractor $\mathcal{F}$ to distinguish between faces. AFR proposals attacking stage ③ corrupt the training of $\mathcal{F}$ to produce an unusable extractor $\mathcal{F}'$.

Goals and Challenges. Here, $\mathbb{F}$'s goal is to train a high-quality feature extractor $\mathcal{F}$ using available training data, so that faces can be accurately identified by their feature vectors extracted by $\mathcal{F}$. Thus P's goal is to prevent $\mathbb{F}$ from training an effective $\mathcal{F}$ by corrupting the training data.

There are two key challenges facing AFR proposal targeting stage ③. The first is minimizing the distortion to training face images introduced by the corruption process while maintaining the corruption efficacy. The second is corrupting $\mathcal{F}$'s training without requiring full dataset control.

6.7.1 Current Solutions

Data poisoning is a well-studied technique in the field of adversarial machine learning. By manipulating the training data of a DNN model, an external party can negatively impact the model's training [131, 75, 203, 382, 284]. Poisoned models can exhibit a variety of (mis)behaviors, from incorrect classification of specific inputs to complete model failure. Existing AFR proposals focus on the latter.

Making training data unlearnable. By injecting specially crafted noise on training data, recent works [151, 116] render data "unlearnable" by a DNN model. This noise misleads the model into thinking that data have already been learned, thwarting necessary parameter updates. When a user submits their "unlearnable" face images as a training image for the $\mathcal{F}$, the extractor will not learn anything to improve its performance. Training an effective $\mathcal{F}$ requires millions or even billions of face images [92, 342, 281], and with a sufficient number of unlearning training examples, $\mathcal{F}$ will not meet the accuracy level required for practical deployment.

Adding adversarial shortcuts. A related proposal from Evtimov *et al.* [107] injects *adversarial shortcuts* into the dataset. Models trained on this data overfit to the shortcut and fail to learn the meaningful semantic features of the data. Now the trained extractor model has a distorted understanding of the feature space, it cannot produce high quality feature vectors required for accurate face recognition.

6.7.2 Discussion of Stage 3 Solutions

Privacy/Utility Trade-offs. The biggest utility drawback of stage ③ proposals is that they require significant effort to corrupt the training dataset. Most proposals require that P control much of the training data to render $\mathcal{F}$ unusable. Such high levels of control would prohibit individual users from using these AFR tools. In addition, $\mathbb{F}$ can discard a corrupted dataset and use other data sources to train their model, once they discover the presence of corruption.

Summary of Progress. Despite utility challenges, existing proposals have shown that, with a sufficient level of dataset control, it is possible to render $\mathcal{F}$ unusable by adding minimally visible perturbations onto training face images. For example, AFR tools based on adversarial shortcuts [107] are effective when they can corrupt the entire training dataset. Others [151] can reduce $\mathcal{F}$'s accuracy on specific classes in a FR model by 16%.

6.8 Attacking Stage 4 to Corrupt Database

In stage ④, $\mathbb{F}$ uses $\mathcal{F}$ to create a reference database $\mathbf{D}$ of labeled face feature vectors that will facilitate identification of unidentified faces. AFR tools targeting this stage seek to fill $\mathbf{D}$ with incorrect face/label mappings, so that $\mathbb{F}$'s classifier $\mathcal{C}$ cannot identify P's query images as P. Thus, when a true image x_P is presented to $\mathbb{F}$'s system for identification, the corrupted database $\mathbf{D}'$ produces $\mathcal{C}(\mathcal{F}(\mathcal{G}(x_P)), \mathbf{D}') = I$, where I is an incorrect identity, $I \neq P$.

Goals and Challenges. In this stage, $\mathbb{F}$'s goal is to create a reference database containing accurate copies of feature vectors (produced by $\mathcal{F}$) of people $\mathbb{F}$ wishes to recognize. P's goal is to prevent $\mathbb{F}$'s feature extractor $\mathcal{F}$ from creating an accurate feature vector which $\mathcal{C}$ can match to query images of P. Note that this can also be achieved by corrupting the training data/process of $\mathcal{F}$ in stage ③, as discussed in §6.7. Here, we differentiate from §6.7 by assuming that $\mathcal{F}$ is a well-trained feature extractor. Thus, P attacks $\mathbb{F}$ by modifying/manipulating the reference images of P that $\mathbb{F}$ uses to create its reference database.

Stage ④ based AFR tools must first address the base case challenge of modifying P's reference images to produce incorrect feature vectors while minimizing the distortion of those images. They also face two advanced challenges. First, they must maintain high performance when $\mathbb{F}$ has some original, unmodified face images of P already enrolled in its database $\mathbf{D}$. Second, protection must persist when P makes incorrect assumptions about $\mathbb{F}$'s system, especially its extractor $\mathcal{F}$ and classifier $\mathcal{C}$, or when $\mathbb{F}$ adapts.

6.8.1 Current Solutions

Existing AFR proposals in this category focus on poisoning feature vectors before they are stored in $\mathbf{D}$. The poisoning techniques depend on the underlying assumptions about $\mathbb{F}$'s classifier $\mathcal{C}$.

Assuming $\mathcal{C}$ uses classification-based matching. A recent AFR proposal, Fawkes [287], assumes $\mathcal{C}$ is a shallow classification layer added to $\mathcal{F}$. Fawkes seeks to corrupt the final classifica-

tion output by "cloaking" (or poisoning) reference images of P, i.e. shifting their feature vectors away from the correct representation by adding imperceptible perturbations to P's reference images [287]. When $\mathcal{C}$ is trained on these shifted feature vectors, $\mathbb{F}$ will learn to associate incorrect feature spaces with P's identity, producing wrong matches for P's (uncloaked) query images at run-time.

Assuming $\mathcal{C}$ uses nearest neighbor-based matching. Two other AFR proposals, LowKey [77] and FoggySight [108], assume $\mathcal{C}$ is a K-nearest neighbors algorithm. LowKey [77] adds digital adversarial perturbations to change the feature representation of P's reference images (similar to Fawkes). These perturbed images create a reference feature vector for P that is different from those of P's run-time query images, thus thwarting $\mathcal{C}$. FoggySight [108] takes a community-driven approach, where users modify their images to protect others. These collective modifications flood the top-K matching set for a specific user with incorrect feature vectors, drowning out the correct feature vector.

6.8.2 Discussion of Stage 4 Solutions

Privacy/Utility Trade-offs. Most Stage ④ proposals add perturbations directly to images. Several proposals discuss how stronger (more visible) perturbations yield stronger AFR protection (i.e. [287, 77]). Visible perturbations may lower the utility of protected images, especially if they are meant to be posted on social media sites. More advanced optimization techniques may help reduce perturbation size at stronger protection levels, but this visibility/protection trade-off seems inevitable.

Summary of Progress. It is encouraging that all proposals listed in the above analysis have demonstrated success on the key task of corrupting feature vectors of P, i.e., the base case. Overcoming the two advanced challenges discussed remains an area for future work. Some proposals provide limited protection when $\mathbb{F}$ has obtained original, unmodified feature vectors of P (e.g. [287]), but not all proposals have considered this possibility. Second, all existing pro-

posals assume knowledge of $\mathcal{C}$ and/or $\mathcal{F}$, necessitating further work to determine how/if incorrect assumptions of $\mathbb{F}$ would affect AFR performance. A final challenge is evaluating the long-term robustness of stage ④-based AFR mechanisms against an adaptive $\mathbb{F}$. Recent work (discussed in §6.11) suggests that a continuously adapting $\mathbb{F}$ may always (or eventually) "win" against AFRs targeting ④.

6.9 Attacking Stage 5 to Evade Identification

The final set of AFR tools aims to prevent run-time query image identification by $\mathbb{F}$'s classifier $\mathcal{C}$, by producing distorted images x'_P that mislead the final classifier outcome, e.g. $\mathcal{C}(\mathcal{F}(\mathcal{G}(x'_P)), \mathbf{D}) = I \neq P$. These methods can provide one-time protection for users who believe their images are already enrolled in $\mathbf{D}$. Furthermore, since labeled query images can also be added to the reference database, using these AFR tools at run-time can also help poison the reference feature vectors (see §6.8). However, current AFR proposals targeting this stage focus strictly on evasion and do not consider this joint possibility.

Goals and Challenges. In this run-time reference stage, $\mathbb{F}$**'s goal** is to use $\mathcal{C}$ to identify the face in a query image. P**'s goal** is to alter their query image so $\mathcal{C}$ cannot match it to their corresponding feature vector in $\mathbf{D}$. We assume $\mathbb{F}$'s reference database $\mathbf{D}$ contains accurate feature vectors of P.

There are two key challenges for stage ⑤-based AFR proposals. The first is achieving successful evasion without significant image distortion. Additionally, proposals must overcome defenses deployed by $\mathbb{F}$ to protect $\mathcal{F}$ and/or $\mathcal{C}$.

6.9.1 Current Solutions

Adversarial perturbations have been the dominant method for evading DNNs and consequently are relevant for evading FR. Due to the extremely high number of these techniques, we restrict our discussion to proposals explicitly designed to evade FR systems at run-time. We organize these proposals by their operational context: physical and digital.

Physical evasion techniques. The first group of proposals injects adversarial perturbations into face images by having P wear them as physical objects. While these methods echo those described in §6.6, they focus on thwarting recognition/classification rather than face detection. Earlier proposals [288, 113] use adversarial makeup and eyeglasses to cause incorrect classification by $\mathcal{C}$. More recent proposals consider two other directions, either using larger but input-independent adversarial patches to boost the effectiveness of evasion [171], or making the perturbation digitally controllable and/or much less perceivable by human eyes by projecting visible/infrared light onto user faces [291, 381, 235].

Digital evasion techniques. Here P digitally modifies their unlabeled (online) face images to prevent them from being accurately classified by $\mathcal{C}$. Most proposals in this category apply traditional adversarial perturbation generation techniques to create minimally visible perturbations that cause $\mathbb{F}$'s feature extractor to produce misleading feature vectors. Their generation process varies depending on assumptions of $\mathcal{C}$'s behavior: a shallow classification layer vs. nearest neighbor based matching [383, 87, 102, 296].

More recent proposals are designed to be more robust to real-world FR systems (i.e. joint optimization on multiple feature extractors, etc.) [64, 79, 363]. Another recent proposal [89] uses a GAN to generate adversarial perturbations rather than using optimization techniques.

6.9.2 Discussion of Stage 5 Solutions

Privacy/Utility Trade-offs. For physical evasion techniques, a significant usability challenge comes from the possibility of real-time recognition. In order to ensure physical evasion tools are effective, a user must wear them in all circumstances where cameras might be present. As with Stage ④, there also exists here a trade-off between perturbation size and evasion success for digital evasion techniques. Reducing the amount of perturbation needed to evade recognition remains an active area of research.

Summary of Progress. So far, existing works have focused on addressing the first key challenge,

Stage Targeted	AFR Property				
	Long-term Robustness	*Broad Coverage*	*No 3rd Party Assistance*	*Disruption to P*	*Disruption to others*
①	●	?	?	●	●
②	●	?	●	●	●
③	?	?	?	●	?
④	●	●	●	●	?
⑤	?	●	●	●	?

Table 6.4: *Evaluating AFR tools using five properties, where the tools are grouped by the FR stage they target.* ● *means that the property has already been achieved by current AFR proposals targeting this stage;* ● *means that the property seems "promising" and could be achieved by AFR designs targeting this stage; and* **?** *indicates significant progress may be required to achieve this property by targeting this stage, and the likelihood of success is unknown.*

and have evasion success with minimally visible perturbations on query images. Addressing the second challenge, or understanding how AFR tools interact with existing defenses against evasion attacks, remains an open area of research. Defenses against evasion attacks like the ones listed above are being released regularly (e.g. [222, 278, 97, 211]), only to be broken by new attacks (e.g. [62, 40, 320]). No defenses have yet been explicitly proposed for these attacks, but the general trend suggests this may be possible.

6.10 Goals and Tradeoffs in AFR Design

In our discussion of current AFR tools, we consider the design space of AFR tools through the lens of specific FR stages they disrupt. To date, all existing AFR proposals have focused their design around disrupting a single stage in this framework. Assuming an AFR tool must disrupt some portion of the FR pipeline to be effective, we can map out and explore the design space of AFR tools using this framework.

For researchers and practitioners in the AFR community, perhaps the most critical question is: *"what are the benefits and limitations of AFR tools that target each specific framework stage?"* Or, an alternative form of the question might be: *"Given a set of prioritized properties for an AFR system, can I find the best stage(s) to disrupt in order to achieve them?"*

We attempt to answer these questions here, by first identifying a set of high level properties that AFR tools can potentially optimize for, then for each property, discussing how targeting a given stage affects an AFR tool's ability to achieve it. Ultimately, we hope to provide a high level roadmap that can guide the design of AFR tools optimizing for specific properties in mind. While we consider each stage in isolation, it might be possible for an AFR tool to target multiple stages, gaining a combination of benefits (and limitations).

6.10.1 Five AFR Design Properties

When considering design properties of AFR tools, we assume that efficacy is a given. Our list of 5 properties target additional considerations beyond basic efficacy, and include desirable properties for efficacy (#1 and #2) and for minimizing dependencies and cost (#3, #4, #5):

1. *Long-term robustness* against evolving FR systems
2. *Broad protection coverage*, efficacy even for users with unprotected face images online
3. *No reliance on 3rd parties*, does strong protection require assistance from service providers or others?
4. *Minimal friction for user* P, minimizing cost for P to deploy the AFR tool on a consistent basis
5. *Minimal impact on other users*, minimizing potential risks to non-users of the AFR tool

6.10.2 Implications of Properties for AFR Design

Next, we discuss the above properties in turn and consider how easily each property can be achieved by AFR tools that target different operational stages in our framework. For each combination of property and target stage, we "quantify" how easily the desirable property can be achieved by an AFR tool designed to disrupt that stage. ● means that the property has already been achieved by current AFR proposals targeting this stage; ● means that the property seems "promising" and has good potential to be achieved by AFR designs targeting this stage; and **?** indicates significant

progress may be required to achieve this property by targeting this stage, and the likelihood of success is unknown. Table 6.4 provides an overview of our conclusions. For easy notation, we will use **AFR**ⓚ to refer to the group of AFR proposals that target FR stage ⓚ.

Property 1: Long-term robustness. An effective AFR tool should provide strong and lasting protection against unwanted facial recognition. That is, it should protect a user P from unwanted FR both initially and as FR evolves.

●**: None** While this principle is the main goal of AFR, none of existing AFR tools (targeting any stage) is able to achieve this property. No current system provides strong protection against ever-evolving FR systems.

●**: AFR①, AFR②, AFR④** Conceptually, P can achieve long-term robustness by consistently undermining the face data pipeline of $\mathbb{F}$. AFR① and AFR② can both prevent any face image of P to be included into $\mathbb{F}$'s pipeline. AFR④ can corrupt $\mathbb{F}$'s understanding of any face images in the reference database. While promising, existing AFR tools fail to *consistently* prevent the inclusion of or corrupt *all* P's images from both online and physical sources.

?: AFR③, AFR⑤ It remains unclear if these two groups of AFR tools can provide long-term robustness. AFR③ could be overcome over time as $\mathbb{F}$ switches to newer and different feature extractors. AFR⑤ offers only one-time protection, and does not address the scenario where query images get added to the reference database.

Property 2: Broad protection coverage. Many of us already have an online presence, e.g., face photos posted years ago without AFR protection. An effective AFR proposal would ideally provide protection under the challenging but realistic scenario where P already has unprotected face images online.

●**: AFR⑤** AFR tools that rely on run-time evasion are not impacted by the existence of unprotected images online.

◔: **AFR④** The presence of unprotected images complicates the protection of AFR④ since $\mathbb{F}$ has some ground truth information about P's facial features. However, the addition of protected images to the reference database can slowly disguise P's true features, and thus achieve protection. Moreover, several AFR tools [287, 108] proposed a "group cloaking" idea where multiple users coordinate together to achieve better protection for those having an existing online presence.

?: AFR①, AFR②, AFR③ These three groups of AFR tools focus on disrupting the (training) data pipeline of FR. As a result, they cannot protect P against $\mathbb{F}$ who has already obtained and processed unprotected images of P.

Property 3: No reliance on 3rd party to operate. Ideally, an AFR tool can be operated by a user P alone and achieve strong protection without assistance or participation third-party, either a central content provider like Facebook or a friendly user willing to help P. This is an abstract measure of the entity-level complexity required to operate the tool. Achieving this property has the added benefit of limiting exposure of potentially sensitive user data to a 3rd party.

●: **AFR②, AFR④, AFR⑤** AFR tools in these three groups all rely on adding certain perturbations on face images, which P can do without outside assistance.

?: AFR①, AFR③ For those AFR① seeking to prevent online data scraping, they rely on the assistance of image sharing platforms. Similarly, disrupting the training $\mathcal{F}$ requires coordinated effort across many users, since P alone contributes relatively few images to the training data.

Property 4: Minimal friction for P. This usability-related property measures what P needs to sacrifice in order to consistently apply the AFR tool. This property is motivated by the well-known findings that users prefer and are more likely to use protection solutions that introduce minimal friction to their daily life [84, 350].

◔: **AFR①, AFR②, AFR③, AFR④, AFR⑤** So far, existing AFR tools all introduce some level of "disruption" to P, whether by adding visual noise, perturbations or transformations to

P's online photos that distorts them, requiring P to always wear odd makeup/clothes/accessories, or necessitating more powerful computing hardware/services to implement the AFR tool against continually evolving $\mathbb{F}$. More efforts are needed to limit the amount/type of disruption to users.

Property 5: Minimal impact on other users. This final property examines how the outcome of P's AFP protection would affect other users. Intuitively, P can protect themselves by forcing $\mathbb{F}$ to fail (give a null or uninformative result), or by intentionally tricking $\mathbb{F}$ to recognize them as another person P'. Depending on the context, the latter may negatively affect P', producing potential social risks (see §6.11.2 for detailed discussions on social challenges of AFR).

●**: AFR①, AFR②** These AFR tools disrupt the data pipeline of $\mathbb{F}$, and thus, have no impact on other users.

?: AFR③, AFR④, AFR⑤ These three groups of AFR tools seek to intentionally misclassify P's face to another user, and as a result, could potentially impact other users included in $\mathbb{F}$'s reference database.

6.11 Challenges for AFR Tools

In this section, we describe what we see as the major technical and broader social/ethical challenges facing future AFR development. Each challenge spans multiple properties and stages laid out in this paper. For each challenge, we provide context for why the challenge exists and, where possible, suggest ways to address it. Like §6.10, the challenges described here represent our best efforts to understand and systematize the AFR space. They are not exhaustive, and are meant as signposts rather than a comprehensive roadmap.

6.11.1 Technical Challenges

TC 1: Reliance on AML-based tools. The majority of AFR proposals, especially those targeting stages ② − ⑤, employ techniques from adversarial machine learning (AML), which have several

key limitations. First, while AML tools have exhibit high performance, they have not provided provable guarantees of protection. Second (and related), AML-based protections can be defeated by adaptive FR systems. For example, $\mathbb{F}$ could adversarially train the feature extractor $\mathcal{F}$ [70, 260] to be more robust against adversarial examples, thus defeating AFR tools against stages ③ or ④. $\mathbb{F}$ could also remove adversarial perturbations from face images before processing them or adding them to the reference database [88], circumventing AFR tools that target stages ② or ⑤.

Potential Directions. More advanced perturbation generation methods may help increase short-term efficacy of AML-based AFR tools. However, the lack of provable, ongoing protection is a much tougher barrier to overcome. In order to provide reliable, ongoing protection, developers of AFR tools can consider two possible paths: (i) integrate provable guarantees into the perturbation generation process, or (ii) consider alternative techniques that provide guaranteed protection. For (ii), there are two potential directions. The first is focus on attacking stage ①, where defeating FR does not require evading or poisoning a DNN (e.g. non-AML AFR tools). The second is to switch from "misleading" $\mathbb{F}$ with "minor" image modifications to completely disabling $\mathcal{F}$ and/or $\mathcal{C}$. Methods in this direction could focus on physical world attacks that exploit camera properties, like the rolling shutter effect [196, 280]; rely on larger image disruptions like shadows [379]; or employ tools like the FR-disabling lasers used in Hong Kong in 2020 [212].

TC 2: Existence of online footprints. Some AFR proposals (especially those targeting stage ④) implicitly or explicitly assume that users can start "from scratch" to protect their online persona. In practice, most Internet users today already have face images online, posted by themselves or others, and at least some of those images are already captured by FR databases. Over 1.8 billion photos are uploaded to online platforms daily [307], making it likely that one or more unmodified photos of a user P will likely end up online, with or without P's knowledge. Given the widespread use of web scraping to collect FR reference images [8, 144], it is likely that one of these photos is already in a reference database.

Potential Directions. This stark reality has two implications for AFR research. *First*, AFR tools should be evaluated under the practical scenarios where the FR system has access to both protected and unprotected online photos of P. While several AFR tools have provided such measurements (e.g., [287, 108]), many others have not. *Second*, we believe that AFR tools managed by online platforms will offer better protection of online footprints against FR systems than those executed by individual users. These platforms can protect photos of an individual posted by them or others, and are overall better positioned to deploy more powerful protection mechanisms.

For example, online platforms could employ the group cloaking techniques proposed in Fawkes [287] or FoggySight [108] to corrupt reference databases composed of images from their sites. After images are scraped, online platforms could use provenance-tracking to re-identify stolen images, e.g., in the training dataset of a feature extractor, and enable exposure/prosecution of photo thieves [276, 294, 272]. All these methods ought to be accompanied by enhanced anti-scraping techniques to prevent large-scale scraping of face images, i.e. stricter rate limiting, access permissions, and scraping detection heuristics, to make it safer for individuals to have online footprints.

TC 3: Privacy/utility/usability tradeoffs of AFR systems. The above paragraph raises an additional technical challenge of AFR design: balancing privacy, utility, and usability. There is a spectrum of ways to balance these. On one end are 3rd-party-adminstered AFR tools (c.f. stage ①), which have the high usability and utility but intrude on privacy to allow 3rd party data processing. On the other end are high-overhead tools like fully homomorphic encryption, which have provable privacy but limited utility/usability.

Where AFR tools should or can exist along this spectrum remains an open question for two reasons. First, we lack a deep understanding of how AFR users would prioritize these tradeoffs in practice. Prior studies show that users prefer privacy tools with minimal overhead [84, 350], but only one study has explored how or if these preferences change in the AFR setting [64]. Second, while many AFR tools have been proposed in recent years, the space of possible AFR designs remains sparsely populated. Consequently, it is worth considering whether this tradeoff is indeed

fundamental, or if future AFR designs may evolve to accommodate all three.

TC 4: Face images don't change. A related, but distinct, challenge to TC 2 faced by AFR systems is the permanence of face data. For better or for worse, most people have the same face their whole adult life. As our faces age, they remain recognizable as uniquely "us" to most humans and FR systems [198]. The slow rate at which faces change is a major challenge for AFR tools. To be long-term effective, these tools must conceal the same piece of static data (a face) from numerous adversaries over many years.

Once $\mathbb{F}$ obtains P's protected face photo, they can try as many times as they want to break the protection [260]. If $\mathbb{F}$ ever succeeds, either in 1 month or 1 year, they "win" and P loses, because modern FR systems only need one clean picture in the reference database to identify a person [92]. For example, Clearview.ai identified a person based on a single reference image in which the person's reflection appeared faintly in a mirror [144]. Clearly, the issue of face data permanence poses a significant challenge for AFR tool development.

TC 5: Lack of transparency of FR systems. The lack of transparency in how proprietary FR systems work hampers AFR tool development and testing. Without access to proprietary FR systems, AFR researchers must do their best to glean a generic understanding of how FR systems work from public documents and academic papers, e.g. [281, 19]. While this may be sufficient to develop AFR tools that work well in the lab, researchers cannot perform comprehensive efficacy tests against proprietary systems.

Furthermore, AFR tool developers have no knowledge of how or if FR systems are adapting to evade AFR systems. The 2020 global FR market was valued at 3.86 billion US dollars [265], so FR stakeholders have ample resources to evolve as new AFR systems emerge. Even passive improvements to FR systems, such as new training methods or architectures, can overcome AFR protection and compromise user privacy [260]. Altogether, this lack of transparency means that AFR tools face an upward battle in the fight against unwanted FR.

6.11.2 Broader Social and Ethical Considerations

In addition to technical challenges, AFR tools face broader social and ethical considerations. These stem from a variety of factors, including a lack of regulation, benefits of FR for the public good, and demographic disparities in FR.

SEC 1: Unregulated, ubiquitous FR. Today, FR systems are generally unregulated and easy to deploy. Practically anyone with a powerful laptop and access to an image dataset could create a FR system. This democratization of FR has allowed 3rd party FR systems like Clearview.ai, which rely on unauthorized data use [142], to flourish. As a result, it is difficult (if not impossible) for individuals to know when and where FR systems are deployed, as well as their capabilities.

This laissez-faire climate creates significant ambiguity as to when AFR tools can/should be deployed. For example, around the world, photos taken for official government purposes (*e.g.* drivers' license and passport photos) are used as reference images in government FR systems aiding law enforcement officers, border control agents, among others [122, 384, 216, 230]. Government-sponsored FR may be *unwanted* but is not (necessarily) *unauthorized* under the status quo, and the legality of using AFR tools in this setting is ambiguous. To augment the confusion, systems like Clearview are used by law enforcement [144], further blurring the concept of *unauthorized* vs *unwanted* FR. As FR and AFR use increases, a clash over this issue seems almost inevitable.

SEC 2: FR used for social good. Both privacy-sensitive citizens and criminals can use AFR tools. Law enforcement's use of facial recognition can benefit society in multiple ways, such as tracking and locating wanted criminals or lost children [17, 16]. Consequently, AFR tools applied by bad actors could ultimately harm the public good. The debate between privacy and national security plays out in numerous other tech domains, such as end-to-end encryption [22]. Legitimate claims can be made by both sides. AFR researchers must be mindful of this tension and the potential consequences of their work.

SEC 3: Harm caused by AFR misidentification. One ethical tension not yet explored in current literature is the social effect of misidentifications caused by AFR tools. For example, if U uses an

AFR tool and is misidentified by $\mathbb{F}$ as P, what outcome might this have for P? If U is engaging in illegal activity but P is arrested instead, the AFR tool could cause serious harm, both to P and to U's victim(s). The well-known bias of FR systems heightens this tension. Police departments routinely make rushed identification decisions from partial FR matches [121]. Furthermore, facial recognition systems misidentify people of color at higher rates [53, 103]. Recent work has found that AFR tools exhibit these same biases [270, 259]. The social impact of AFR tools requires urgent study.

6.12 Discussion

Related Surveys. Two surveys [220, 240], alluded to in §6.1, address topics similar to our work. Here, we provide an in-depth comparison to these and emphasize our unique contributions. [220] focuses on how a *service provider* can build a privacy-preserving FR service (see it's §II.D), while our work considers how *individual users* can use a privacy tool to defend themselves against *intrusive* FR services. Different from [220], our work provides a holistic view of what individual users can do to disrupt FR and a detailed discussion of challenges facing user-centric AFR solutions.

[240] also discusses methods a service provider could use to preserve privacy in a video surveillance setting. Furthermore, since [240] focuses on video surveillance, rather than image-based FR systems, it explores privacy preservation techniques for video-specific traits like gait, height, and clothing, which dilutes FR-specific content. [240] does address face de-identification in its §3.4.3, but focuses solutions for providers who wish to deanonymize *all* faces in their system (e.g. by averaging or blurring them), unlike our focus on protecting individuals from unwanted surveillance.

Concluding Thoughts. As facial recognition (FR) continues to grow in scale and ubiquity, we expect the demand for anti-facial recognition tools to continue to rise. There is an urgent need to think longitudinally about AFR tools, analyzing both their limits and their potential. Our paper aims to fill this gap by providing both a framework for discussing AFR proposals and an assessment of the current state of AFR research, enabling deeper understanding of data agency solutions in this

space.

Current AFR tools possess some, but not all, of the traits needed to successfully defeat unwanted FR in the real world. Many proposals leverage adversarial perturbations to evade FR models, either in the preprocessing ② or classification ⑤ stages. These are often effective in the short-term, but lack long-term guarantees, and cannot fundamentally change FR system behavior in the future. Future AFR proposals may benefit from more exploration of designs that target stages ① and ④, which could provide wider-reaching protection.

CHAPTER 7

CONCLUDING THOUGHTS

As machine learning models continue to grow in scale and popularity, the need for data agency becomes more urgent. This work proposed three technical tools, each using a different technique to help users reclaim data agency: **disruption** (§3), **tracing** (§4), and **direct attacks** (§5). It also **provides a framework** (§6) for reasoning about data agency against unwanted facial recognition.

7.1 Limitations of Proposed Solutions.

Complicated nature of face recognition. The tools proposed here focus largely on the image classification space, and particularly on settings like face recognition. Certainly more work is needed to prevent unwanted face recognition, given the privacy concerns surrounding this technology. However, as discussed in §6, data agency solutions in this space can be complicated. There are numerous security-critical uses of face recognition by government entities, "opting out" of which may be impossible. Furthermore, there could be yet-unexplored legal ramifications of the proposed technical data agency solutions. Future collaboration between academics and policymakers is necessary to shed light on this complicated issue.

Potential for "cat and mouse". As is often the case in the adversarial machine learning space, the solutions proposed here lack guarantees of future durability. In particular, the Fawkes and Isotopes (disruption and tracing) solutions may be circumvented by motivated adversaries.

This "cat and mouse" style adversarial behavior was observed in practice in the deployment of Fawkes. After publication, the Fawkes tool was open sourced, and its code was made publicly available on Github[1]. About six months later, our team was contacted by the authors of [77], who informed us that their experiments showed Fawkes was not effective against Microsoft Azure's face recognition API. This contradicted our initial results, and we confirmed the drop in protection

1. `https://github.com/Shawn-Shan/fawkes`

success through additional experiments. We later deduced that Azure had used the open source Fawkes code to adversarially train [348] their model to be robust against Fawkes-like perturbations. We then confirmed that cloaks generated using a private feature extractor, unseen by the Azure team, remained effective against Azure.

This anecdote illustrates the fundamental issue with using adversarial machine learning-based data agency tools: they are not provably secure. Adversaries can adapt around them. However, in a space like machine learning—where provable guarantees are difficult, if not impossible, to get in practice—this is unsurprising. It is difficult to prove that a model itself is functioning as expected, due to the complexity of the parameter space. Proving that a particular data agency tool can protect users indefinitely is an even loftier goal.

The question remains, though: why use these tools, if we can't guarantee their long-term success? The answer is twofold. First, these solutions adopt a common goal of past security tools: *to raise the attack cost significantly for attackers*, rather than preventing attackers from succeeding. Numerous prior works have framed security and privacy problems as economic problems [35, 34]. Attackers are looking for quick and easy victories, and if they cannot find them, they look for a different set of people to target. Since users want to have control over their data and prevent unauthorized use, users win if their use of a data agency tool causes attackers target someone else.

The second reason to continue using these data agency solutions is that, in a sense, they were *never meant to be permanent solutions*. Their goal is to raise the bar until other, more permanent methods of preventing unwanted data use arise. They represent one set of solutions to the problem—user-facing, technical ones that directly empower users in their interactions with models. While important, given their limitations, such solutions cannot be the only ones. They are the first wave, designed to stand in the breach while others develop complementary solutions—technical, regulatory, and other—that can provide a fuller set of options for preserving data agency.

7.2 Considerations for Future Data Agency Solutions

As the problem of unwanted ML data use grows, the need for data agency solutions will increase. Here, we propose five directions for future work in this space.

1. Do users care? A key assumption of the tools proposed in this thesis—and in the ML data privacy conversation more broadly—is that users *want* control over their data in ML settings. However, little or no empirical evidence supports this assumption. Prior work has shown that a majority of Americans are concerned about how both private and public entities use their data [42], but no existing work has studied whether these concerns extend to the machine learning setting or beyond a Western context. Validating how, and if, users' concerns about data use change when data is used to train different types of ML models is a critical piece of future work. Such a study could inform future thinking on data agency in general. It could also lead to more tailored ML data privacy solutions, if perhaps users care more about privacy against certain types of models.

2. More finely-tuned cost analysis. §7.1 discussed how data agency solutions can *raise the bar* for attackers wishing to steal data, making it more worthwhile for them to focus their attention elsewhere. While this is a noble goal, in practice, the solutions proposed here do not provide any estimate of *how much* the data agency solutions would cost the attacker. Along these lines, future work could make two significant improvements. First, it could provide more fine-grained estimates of how much a given solution costs an attacker in terms of compute, data cleaning, or other costs per user. If, for example, using Fawkes would cause an attacker to incur an additional $25/user cost, this may change their calculus for using training data scraped from online. This analysis should be accompanied by a study of how much additional cost is needed to affect attacker behavior would be much-needed.

Beyond this, future data agency solutions could consider ways to make attackers incur specific cost increases. This idea is similar to that of "crypto crumple zones," encryption methods designed to be breakable if an attacker is willing to expend a certain amount of money or computational

power [349]. This balances the problem of making some encrypted information (e.g. a suspect's text messages) accessible to certain parties (e.g. law enforcement) while making it too expensive for an average person to run this attack. Data agency solutions that specifically optimize for certain attacker costs would be an interesting line of future work. There may be a direct tradeoff between costs to the attacker and usability to a user (e.g. higher-cost solution == solution that adds more visible perturbations to image), so users' willingness to incur this tradeoff would be an important line of future work.

3. Data agency against generative models. Although this thesis primarily focuses on data agency in image classification contexts, another domain ripe for data agency solutions is that of generative ML, including against text-to-image models like Stable Diffusion [303] or voice synthesis models like Vall-E [340]. Some solutions proposed in this work (particularly the data tracing solution) could translate to the generative setting, but would require significant revision to work in that domain. A few works have been specifically designed to counteract unwanted data use in generative model setting, particularly in the text-to-image domain [285, 277]. An even broader set of data agency solutions against generative models will be needed as these models proliferate.

4. Exploring unintended side effects of data agency tools. At least one work has explored the possibility of bias in data agency tools like Fawkes [270]. This work showed that since cloaks are constructed on face recognition feature extractors, which can be biased, the cloaks have unequal levels of protection for different demographic subgroups. There is an urgent need to study other forms of bias of data agency tools. This applies particularly to tools that leverage principles of adversarial machine learning, since the performance of these tools depends on intrinsic properties of models themselves. If models are biased (as they often are), the data agency tools may be also.

Other unintended side effects of data agency solutions should also be studied. This point was discussed in §6, how anti-face recognition tools that cause misclassification could harm innocent bystanders. Some people use a data agency tool because they want rightful protection; others might use it because they have something to hide. For the latter camp, a misclassification result in

a facial recognition system would benefit them but potentially harm someone else. In a different vein, incorrectly identifying that data was used (or not used) to train a model (e.g. the isotopes proposal) could cause economic, privacy, or other harms. Overall, there are many potential negative consequences of data agency tools that could be studied. We must ensure that "the cure is not worse that the disease."

5. Tools for ongoing consent to data use. Proposed data agency solutions focus exclusively on scenarios in which users' data is taken without their consent. However, there may be settings in which users are open to contributing their data to causes they care about. In these cases, users would be willing to have their data used to train ML models, such as models which could have societal benefits (e.g. medical diagnosis).

In this setting, where data use is permitted, proposed data agency solutions fall short. What users might need instead are tools enabling them to provide meaningful, ongoing consent to data use, and easy revocation if they change their mind. For example, consider the case where a patient lets their data be used to train a diagnostic model for a rare disease they have. There are obvious societal benefits to training such a model, but given the sensitivity of the data, the user may wish to retain control over their data than just signing a form that hands it over wholesale. Thus, a tool which enables the user to give their consent but, crucially, *update or revoke it over time* would be helpful in this setting. Such tools do not currently exist. The closest thing we have to a "right to revocation" in ML right now are machine unlearning proposals, which are difficult to implement in practice [48, 318]. Future data agency solutions are needed to address this scenario and provide users with ways to meaningfully offer up their data for causes they are about while not relinquishing total control over it.

7.3 Regulation and Data Agency

As discussed above, additional tools and techniques are needed to help individual reclaim data agency. Legislative and policy actions are necessary to supplement technical data agency solutions.

Here, we provide a brief overview of existing data agency efforts in the policy space.

Non-academic stakeholders already begun to consider regulatory and policy actions to promote data agency. Bodies like the EU and the FTC have demonstrated real commitment to helping users retain agency over their data. Regulation like GDPR and the FTC's recent "request for comments" regarding commercial surveillance practices [83] represent tangible steps towards a better future. While these steps are critical, much of the policy and regulatory focus thus far has been on data *privacy*, rather than data *agency*. As discussed at the beginning of this thesis, data privacy is a necessary but not sufficient criterion for giving users back full control over their data. Regulatory efforts can and should consider the question of agency, just as much as privacy.

Another promising movement towards addressing data agency in the ML context came in the White House Office of Science and Technology Policy's recently released "A Blueprint for an AI Bill of Rights" [238]. This document, written in collaboration with numerous stakeholders, outlines five key principles that ought to be true for users of an AI (aka ML) system. Several of these principles—notably "data privacy" and "notice and explanation"—relate explicitly to principles of data agency discussed in this thesis. For example, the summary of the "data privacy" principle states that "You should be protected from abusive data practices via built-in protections and you should have **agency** over how data about you is used" (emphasis added). While this document lists principles, not policies, it shows the government's interest in protecting user data in an ML setting. This bodes well for future policy-based data agency solutions.

7.3.1 Closing Thoughts.

Much remains to be done to realize the dream of data agency in the machine learning space. Regulators should pass laws preventing unauthorized data use. Legislative bodies should create meaningful avenues for users to reclaim online agency. Companies should implement less predatory, more transparent policies around data use. And, of course, researchers should continue dreaming of a future where our data is ours to own—and creating solutions that move us towards that future.

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] Cloudwalk. https://cloudwalk.com/en.

[5] Common crawl dataset. https://commoncrawl.org/.

[6] Idemia. https://na.idemia.com/.

[7] Megvii. https://en.megvii.com.

[8] Pimeyes. https://pimeyes.com/en.

[9] PubFig83: A resource for studying face recognition in personal photo collections. http://vision.seas.harvard.edu/pubfig83/.

[10] Sensetime. https://sensetime.com.

[11] https://www.faceplusplus.com/face-searching/. Face++ Face Searching API.

[12] Using the IJG JPEG library: Advanced features. http://apodeline.free.fr/DOC/libjpeg/libjpeg-3.html.

[13] Vgg face dataset. http://www.robots.ox.ac.uk/~vgg/data/vgg_face/.

[14] Yitu. https://yitutech.com/en.

[15] Microsoft quietly deletes largest public face recognition data set. 2019. https://www.ft.com/content/7d3e0d6a-87a0-11e9-a028-86cea8523dc2.

[16] Pinellas county sheriff's office facial recognition program. *Pinellas County Sheriff's Office*, 2019. https://www.documentcloud.org/documents/6586379-FACESlist-Redacted.html.

[17] Chinese police use facial recognition to find child abducted 30 years ago. *FindBiometrics*, 2020. https://findbiometrics.com/chinese-police-use-facial-recognition-find-child-abducted-30-years-ago-052107/.

[18] Facial recognition tech fights coronavirus in chinese city. *France24*, 2020. https://www.france24.com/en/live-news/20210713-facial-recognition-tech-fights-coronavirus-in-chinese-city.

[19] Facial recognition technology: Privacy and accuracy issues related to commericial uses. 2020.

[20] Huawei/megvii uyghur alarms. 2020. https://ipvm.com/reports/huawei-megvii-uygur.

[21] In-car biometric technology for human interaction. *Hyunai Motor Group*, 2020. https://tech.hyundaimotorgroup.com/article/in-car-biometric-technology-for-human-interaction/.

[22] International statement: End-to-end encryption and public safety. *United States Department of Justice*, 2020. https://www.justice.gov/opa/pr/international-statement-end-end-encryption-and-public-safety.

[23] Mapped: The state of facial recognition around the world, 2020. https://www.visualcapitalist.com/facial-recognition-world-map/.

[24] Could facial recognition be the future of airport security? Delta Air Lines is testing it out. *CBS News*, 2021. https://www.cbsnews.com/news/facial-recognition-delta-tsa/.

[25] Ban dangerous facial recognition technology that amplifies racist policing. *Amnesty International*, 2021.

[26] China state tv exposes wide illegal use of facial recognition cameras in commercial properties. *China Money Network*, 2021. https://www.chinamoneynetwork.com/2021/03/16/china-state-tv-exposes-wide-illegal-use-of-facial-recognition-cameras-in-commercial-properties.

[27] Deepmind faces legal action over nhs data use, 2021. https://www.bbc.com/news/technology-58761324.

[28] Fears for children's privacy as delhi schools install facial recognition. *Reuters*, 2021. https://www.reuters.com/article/us-india-tech-facial recognition-trfn-idUSKBN2AU0P5.

[29] How to protect your phone and identity at protests. *Amnesty International*, 2021. https://banthescan.amnesty.org/wp-content/uploads/2021/01/Amnesty-Tech-Toolkit-FINAL-1.pdf.

[30] The retail stores you probably shop at that use facial-recognition technology. 2021. https://www.businessinsider.com/retail-stores-that-use-facial-recognition-technology-macys-2021-7.

[31] Stop facial recognition. *Big Brother Watch*, 2021. https://bigbrotherwatch.org.uk/campaigns/stop-facial-recognition/.

[32] 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.

[33] John M Abowd. The us census bureau adopts differential privacy. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 2867–2867, 2018.

[34] Alessandro Acquisti, Leslie K John, and George Loewenstein. What is privacy worth? *The Journal of Legal Studies*, 42(2):249–274, 2013.

[35] Ross Anderson. Why information security is hard-an economic perspective. In *Seventeenth Annual Computer Security Applications Conference*, pages 358–365. IEEE, 2001.

[36] Mark Andrejevic and Neil Selwyn. Facial recognition technology in schools: Critical questions and concerns. *Learning, Media and Technology*, 45(2):115–128, 2020.

[37] Yoshinori Aono, Takuya Hayashi, Lihua Wang, Shiho Moriai, et al. Privacy-preserving deep learning via additively homomorphic encryption. *IEEE Transactions on Information Forensics and Security*, 2017.

[38] Devansh Arpit, Stanislaw Jastrzebski, et al. A closer look at memorization in deep networks. In *Proc. of ICML*, 2017.

[39] Illinois General Assembly. (740 ilcs 14/) biometric information privacy act. 2019.

[40] Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *International conference on machine learning*, pages 274–283. PMLR, 2018.

[41] Buse Gul Atli Tekgul and N Asokan. On the Effectiveness of Dataset Watermarking. In *Proc. of the ACM International Workshop on Security and Privacy Analytics*, 2022.

[42] Brooke Auxier, Lee Rainie, Monica Anderson, Andrew Perrin, Madhu Kumar, and Erica Turner. Americans and privacy - concerned, confused and feeling lack of control over their personal information. 2019.

[43] Jiayu Bao. Sparse adversarial attack to object detection. *arXiv preprint arXiv:2012.13692*, 2020.

[44] Peter N. Belhumeur, Joao P Hespanha, and David J. Kriegman. Eigenfaces vs. fisherfaces: Recognition using class specific linear projection. *IEEE TPMI*, 1997.

[45] Arjun Nitin Bhagoji, Supriyo Chakraborty, Prateek Mittal, and Seraphin Calo. Analyzing federated learning through an adversarial lens. In *Proc. of ICML*. PMLR, 2019.

[46] Arjun Nitin Bhagoji, Warren He, Bo Li, and Dawn Song. Practical black-box attacks on deep neural networks using efficient query mechanisms. In *Proc. of ECCV*, 2018.

[47] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[48] Lucas Bourtoule, Varun Chandrasekaran, Christopher A Choquette-Choo, et al. Machine unlearning. In *Proc. of IEEE S&P*, 2021.

[49] Wieland Brendel, Jonas Rauber, and Matthias Bethge. Decision-based adversarial attacks: Reliable attacks against black-box machine learning models. In *Proc. of ICLR*, 2018.

[50] Tom Brown, Benjamin Mann, Nick Ryder, et al. Language models are few-shot learners. *Proc. of NeurIPS*, 2020.

[51] Tom B Brown, Dandelion Mané, Aurko Roy, Martín Abadi, and Justin Gilmer. Adversarial patch. In *Proc. of NeurIPS Workshop*, 2017.

[52] Kieran Browne, Ben Swift, and Terhi Nurmikko-Fuller. Camera adversaria. In *Proc. of CHI*, pages 1–9, 2020.

[53] Joy Buolamwini and Timnit Gebru. Gender shades: Intersectional accuracy disparities in commercial gender classification. In *Proc. of FaaCT*, 2018.

[54] Chris Burt. Kenyan police launch facial recognition on urban cctv network. 2018. `https://www.biometricupdate.com/201809/kenyan-police-launch-facial-recognition-on-urban-cctv-network`.

[55] Darren Byler. Because there were cameras, i didn't ask any questions. 2020. `https://www.chinafile.com/extensive-surveillance-china`.

[56] 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.

[57] Nicholas Carlini, Jamie Hayes, Milad Nasr, Matthew Jagielski, Vikash Sehwag, Florian Tramèr, Borja Balle, Daphne Ippolito, and Eric Wallace. Extracting training data from diffusion models. *arXiv preprint arXiv:2301.13188*, 2023.

[58] Nicholas Carlini, Daphne Ippolito, Matthew Jagielski, Katherine Lee, Florian Tramer, and Chiyuan Zhang. Quantifying memorization across neural language models. *arXiv preprint arXiv:2202.07646*, 2022.

[59] Nicholas Carlini, Chang Liu, et al. The secret sharer: Evaluating and testing unintended memorization in neural networks. In *Proc. of USENIX Security*, 2019.

[60] Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, et al. Extracting training data from large language models. In *Proc. of USENIX Security*, 2021.

[61] Nicholas Carlini and David Wagner. Defensive distillation is not robust to adversarial examples. *arXiv preprint arXiv:1607.04311*, 2016.

[62] Nicholas Carlini and David Wagner. Magnet and "efficient defenses against adversarial attacks" are not robust to adversarial examples. *arXiv preprint arXiv:1711.08478*, 2017.

[63] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *Proc. of IEEE S&P*, 2017.

[64] Varun Chandrasekaran, Chuhan Gao, Brian Tang, Kassem Fawaz, Somesh Jha, and Suman Banerjee. Face-off: Adversarial face obfuscation. *Proc. of PETS*, 2021.

[65] Melissa Chase, Esha Ghosh, and Saeed Mahloujifar. Property inference from poisoning. *arXiv preprint arXiv:2101.11073*, 2021.

[66] Bryant Chen, Wilka Carvalho, Nathalie Baracaldo, Heiko Ludwig, Benjamin Edwards, Taesung Lee, Ian Molloy, and B̃iplav Srivastava. Detecting backdoor attacks on deep neural networks by activation clustering. *arXiv preprint arXiv:1811.03728*, 2018.

[67] Jianbo Chen, Michael I. Jordan, and Martin J. Wainwright. Hopskipjumpattack: A query-efficient decision-based attack. In *Proc. of IEEE S&P*, pages 668–685, 2020.

[68] Jun-Cheng Chen, Bo-Han Kung, Kai-Lung Hua, Daniel Stanley Tan, et al. Naturalistic physical adversarial patch for object detectors. In *Proc. of ICCV*, pages 7828–7837. IEEE, 2021.

[69] Junjie Chen, Wendy Hui Wang, and Xinghua Shi. Differential privacy protection against membership inference attack on machine learning for genomic data. In *Proc. of BIOCOMPUTING*, pages 26–37. World Scientific, 2020.

[70] Liuqiao Chen, Hu Wang, Benjamin Zi Hao Zhao, Minhui Xue, and Haifeng Qian. Oriole: Thwarting privacy against trustworthy deep learning models. *arXiv preprint arXiv:2102.11502*, 2021.

[71] 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 *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2287–2295, 2019.

[72] Pin-Yu Chen, Yash Sharma, Huan Zhang, Jinfeng Yi, and Cho-Jui Hsieh. Ead: elastic-net attacks to deep neural networks via adversarial examples. In *Proc. of AAAI*, 2018.

[73] Pin-Yu Chen, Huan Zhang, Yash Sharma, Jinfeng Yi, and Cho-Jui Hsieh. Zoo: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models. In *Proc. of AISec*, pages 15–26, 2017.

[74] Steven Chen, Nicholas Carlini, and David Wagner. Stateful detection of black-box adversarial attacks. In *Proceedings of ACM Workshop on Security and Privacy on Artificial Intelligence*, pages 30–39, 2020.

[75] Xinyun Chen, Chang Liu, Bo Li, Kimberly Lu, and Dawn Song. Targeted backdoor attacks on deep learning systems using data poisoning. *arXiv preprint arXiv:1712.05526*, 2017.

[76] 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.

[77] Valeriia Cherepanova, Micah Goldblum, Harrison Foley, Shiyuan Duan, John P Dickerson, Gavin Taylor, and Tom Goldstein. Lowkey: Leveraging adversarial attacks to protect social media users from facial recognition. In *Proc. of ICLR*, 2021.

[78] Josh Chin and Clement Burge. Twelve days in xinjiang: How china's surveillance state overwhelms daily life. *The Wall Street Journal*, 2017.

[79] Thomas Cilloni, Wei Wang, Charles Walter, and Charles Fleming. Ulixes: Facial recognition privacy with adversarial machine learning. *Proc. of PETS*, 2022.

[80] Mike Clark. Scraping by the numbers, 2021. `https://about.fb.com/news/2021/05/scraping-by-the-numbers/`.

[81] Thomas Clayburn. Apple sued in nightmare case involving teen wrongly accused of shoplifting, driver's permit used by impostor, and unreliable facial-rec tech. *The Register*, 2021.

[82] European Commission. 2018 reform of EU data protection rules.

[83] US Federal Trade Commission. Trade regulation rule on commercial surveillance and data security. *Federal Register*, 2022.

[84] Kovila PL Coopamootoo. Usage patterns of privacy-enhancing technologies. In *Proc. of CCS*, 2020.

[85] Jim Cross. Valley attorney: Facebook facial recognition carries identity theft risk. *KTAR News*, September 2019.

[86] US Customs and Border Patrol. Collection of biometric data from aliens upon entry to and departure from the united states. `https://www.regulations.gov/docket/USCBP-2020-0062/document`.

[87] Ali Dabouei, Sobhan Soleymani, Jeremy Dawson, and Nasser Nasrabadi. Fast geometrically-perturbed adversarial faces. In *Proc. of WACV*. IEEE, 2019.

[88] Debayan Deb, Xiaoming Liu, and Anil K Jain. Faceguard: A self-supervised defense against adversarial face images. *arXiv preprint arXiv:2011.14218*, 2020.

[89] Debayan Deb, Jianbang Zhang, and Anil K Jain. Advfaces: Adversarial face synthesis. In *Proc. of IJCB*. IEEE, 2019.

[90] 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 S& P*, 2019.

[91] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *Proc. of CVPR*, 2009.

[92] Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. In *Proc. of CVPR*, pages 4690–4699, 2019.

[93] Li Deng and Xiao Li. Machine learning paradigms for speech recognition: An overview. *IEEE Transactions on Audio, Speech, and Language Processing*, 21(5):1060–1089, 2013.

[94] developer.apple.com. Classifying Images with Vision and Core ML, 2022.

[95] Malkia Devich-Cyril. Defund facial recognition. *The Atlantic*, 2020.

[96] Hannah Devlin. We are hurtling towards a surveillance state: the rise of facial recognition technology. 2019. `https://www.theguardian.com/technology/2019/oct/05/facial-recognition-technology-hurtling-towards-surveillance-state`.

[97] Guneet S Dhillon, Kamyar Azizzadenesheli, Zachary C Lipton, Jeremy Bernstein, Jean Kossaifi, Aran Khanna, and Anima Anandkumar. Stochastic activation pruning for robust adversarial defense. *arXiv preprint arXiv:1803.01442*, 2018.

[98] AWS Documentation. Documentation for AWS Rekognition, 2022.

[99] Azure Documentation. Documentation for Azure Face API, 2022.

[100] Google Cloud Documentation. Predict for image classification, 2022.

[101] Yinpeng Dong, Tianyu Pang, Hang Su, and Jun Zhu. Evading defenses to transferable adversarial examples by translation-invariant attacks. In *Proc. of CVPR*, 2019.

[102] Yinpeng Dong, Hang Su, Baoyuan Wu, Zhifeng Li, Wei Liu, Tong Zhang, and Jun Zhu. Efficient decision-based black-box adversarial attacks on face recognition. In *Proc. of CVPR*, pages 7714–7722, 2019.

[103] Samuel Dooley, Tom Goldstein, and John P Dickerson. Robustness disparities in commercial face detection. *arXiv preprint arXiv:2108.12508*, 2021.

[104] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[105] Cynthia Dwork. Differential privacy: A survey of results. In *Proc. of International Conference on Theory and Applications of Models of Computation*. Springer, 2008.

[106] Cynthia Dwork, Aaron Roth, et al. The algorithmic foundations of differential privacy. *Foundations and Trends in Theoretical Computer Science*, 2014.

[107] Ivan Evtimov, Ian Covert, Aditya Kusupati, and Tadayoshi Kohno. Disrupting model training with adversarial shortcuts. *arXiv preprint arXiv:2106.06654*, 2021.

[108] Ivan Evtimov, Pascal Sturmfels, and Tadayoshi Kohno. Foggysight: a scheme for facial lookup privacy. *Proc. of PETS*, 2021.

[109] Face++. Documentation for Face++ API, 2017.

[110] Reuben Feinman, Ryan R Curtin, Saurabh Shintre, and Andrew B Gardner. Detecting adversarial samples from artifacts. *arXiv:1703.00410*, 2017.

[111] Vitaly Feldman. Does learning require memorization? a short tale about a long tail. In *Proc. of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, 2020.

[112] Vitaly Feldman and Chiyuan Zhang. What neural networks memorize and why: Discovering the long tail via influence estimation. *arXiv preprint arXiv:2008.03703*, 2020.

[113] Ranran Feng and Balakrishnan Prabhakaran. Facilitating fashion camouflage art. In *Proc. of ACM Multimedia*, 2013.

[114] Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. Model inversion attacks that exploit confidence information and basic countermeasures. In *Proc. of CCS*, 2015.

[115] Barry Friedman and Andrew Ferguson. Here's a way forward on facial recognition. *The New York Times*, 2019.

[116] Shaopeng Fu, Fengxiang He, Yang Liu, Li Shen, and Dacheng Tao. Robust unlearnable examples: Protecting data privacy against adversarial learning. In *Proc. of ICLR*, 2021.

[117] Pete Fussey and Daragh Murray. Independent report on the london metropolitan police service's trial of live facial recognition technology. 2019.

[118] Yansong Gao, Chang 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*, 2019.

[119] GAO-20-568. Facial recognition: Cbp and tsa are taking steps to implement programs, but cbp should address privacy and system performance issues. page 38, 2020.

[120] GAO-21-518. Facial recognition technology: Federal law enforcement agencies should better assess privacy and other risks. 2021.

[121] Clare Garvie. Garbage in, garbage out: Face recognition on flawed data. *Georgetown Law Center on Privacy and Technology*.

[122] Clare Garvie, Alvaro Bedoya, and Jonathan Frankle. The perpetual lineup. *Georgetown Law Center on Privacy and Technology*, 2016.

[123] Clare Garvie and Laura Moy. America under watch. *Georgetown Law Center on Privacy and Technology*, 2019.

[124] Jonas Geiping, Liam Fowl, W Ronny Huang, Wojciech Czaja, Gavin Taylor, Michael Moeller, and Tom Goldstein. Witches' brew: Industrial scale data poisoning via gradient matching. In *Proc. of ICLR*, 2021.

[125] Ran Gilad-Bachrach, Nathan Dowlin, Kim Laine, Kristin Lauter, Michael Naehrig, and John Wernsing. Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy. In *Proc. of ICML*, 2016.

[126] Zachary Gold and Mark Latonero. Robots welcome: Ethical and legal considerations for web crawling and scraping. *Wash. JL Tech. & Arts*, 13:275, 2017.

[127] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[128] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Proc. of NeurIPS*, 2014.

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

[130] Patrick Grother, Mei Ngan, and Kayee Hanaoka. Ongoing face recognition vendor test (frvt). *NIST*, 2018. `https://nvlpubs.nist.gov/nistpubs/ir/2018/NIST.IR.8238.pdf`.

[131] Tianyu Gu, Brendan Dolan-Gavitt, and Siddharth Garg. Badnets: Identifying vulnerabilities in the machine learning model supply chain. In *Proc. of Machine Learning and Computer Security Workshop*, 2017.

[132] Developer Guide. What is Amazon Rekognition?, 2022. `https://docs.aws.amazon.com/rekognition/latest/dg/what-is.html`.

[133] Chuan Guo, Tom Goldstein, Awni Hannun, and Laurens Van Der Maaten. Certified data removal from machine learning models. *arXiv preprint arXiv:1911.03030*, 2019.

[134] Wenbo Guo, Lun Wang, Xinyu Xing, Min Du, and Dawn Song. Tabor: A highly accurate approach to inspecting and restoring trojan backdoors in AI systems. *arXiv preprint arXiv:1908.01763*, 2019.

[135] Yandong Guo, Lei Zhang, Yuxiao Hu, Xiaodong He, and Jianfeng Gao. Ms-celeb-1m: A dataset and benchmark for large-scale face recognition, 2016.

[136] Neal Gupta, W Ronny Huang, Liam Fowl, Chen Zhu, Soheil Feizi, Tom Goldstein, and John P Dickerson. Strong baseline defenses against clean-label poisoning attacks. *arXiv:1909.13374*, 2019.

[137] Otkrist Gupta and Ramesh Raskar. Distributed learning of deep neural network over multiple agents. *Journal of Network and Computer Applications*, 116, 2018.

[138] Afzalul Haque and Sanjay Singh. Anti-scraping application development. In *Proc. of ICACCI*. IEEE, 2015.

[139] Adam Harvey. Cv dazzle: Camouflage from face detection. *Master's thesis*, 2010.

[140] Jonathan Hayase, Weihao Kong, Raghav Somani, and Sewoong Oh. Defense against backdoor attacks via robust covariance estimation. In *Proc. of ICML*. PMLR.

[141] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proc. of CVPR*, 2016.

[142] Rebecca Heilweil. The world's scariest facial recognition company, explained. *Vox.com*.

[143] Ehsan Hesamifard, Hassan Takabi, and Mehdi Ghasemi. Cryptodl: Deep neural networks over encrypted data. *arXiv preprint arXiv:1711.05189*, 2017.

[144] Kashmir Hill. The secretative company that may end privacy as we know it. *The New York Times*, 2020.

[145] Kashmir Hill. Clearview ai's facial recognition app called illegal in canada. *The New York Times*, 2021.

[146] 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`.

[147] Sorami Hisamoto, Matt Post, and Kevin Duh. Membership inference attacks on sequence-to-sequence models: Is my data in your machine translation system? *Transactions of the Association for Computational Linguistics*, 2020.

[148] Sebastian Houben, Johannes Stallkamp, Jan Salmen, Marc Schlipsing, and Christian Igel. Detection of Traffic Signs in Real-World Images: The German Traffic Sign Detection Benchmark. In *Proc. of IJCNN*, 2013.

[149] Hongsheng Hu, Zoran Salcic, et al. Membership Inference via Backdooring. *arXiv preprint arXiv:2206.04823*, 2022.

[150] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proc. of CVPR*, 2017.

[151] Hanxun Huang, Xingjun Ma, Sarah Monazam Erfani, James Bailey, and Yisen Wang. Unlearnable examples: Making personal data unexploitable. In *Proc. of ICLR*, 2021.

[152] David H Hubel and Torsten N Wiesel. Receptive fields and functional architecture of monkey striate cortex. *The Journal of physiology*, 195(1):215–243, 1968.

[153] Håkon Hukkelås, Rudolf Mester, and Frank Lindseth. Deepprivacy: A generative adversarial network for face anonymization. In *Proc. of ISVC*, 2019.

[154] Andrew Ilyas, Logan Engstrom, Anish Athalye, and Jessy Lin. Black-box adversarial attacks with limited queries and information. In *Proc. of ICML*, 2018.

[155] Huseyin A Inan, Osman Ramadan, et al. Training data leakage analysis in language models. *arXiv preprint arXiv:2101.05405*, 2021.

[156] Ada Lovelace Institute. Beyond face value: Public attitudes to facial recognition technology. *ALI*, 2019.

[157] Daphne Ippolito, Florian Tramer, Milad Nasr, Chiyuan Zhang, Matthew Jagielski, Katherine Lee, Christopher A Choquette-Choo, and Nicholas Carlini. Preventing verbatim memorization in language models gives a false sense of privacy. *arXiv preprint arXiv:2210.17546*, 2022.

[158] Nadiya Ivanenko. Midjourney v4: an incredible new version of the AI image generator, 2022. `https://mezha.media/en/2022/11/11/midjourney-v4-is-an-incredible-new-version-of-the-ai-image-generator/`.

[159] Harrison Jacobs and Pat Ralph. Inside the creepy and impressive startup funded by the chinese government that is developing ai that can recognize anyone, anywere. *Business Insider*. `https://www.businessinsider.com/china-facial-recognition-tech-company-megvii-faceplusplus-2018-5`.

[160] Matthew Jagielski, Giorgio Severi, Niklas Pousette Harger, and Alina Oprea. Subpopulation data poisoning attacks. In *Proc. of CCS*, 2021.

[161] 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.

[162] Dina Jawad. Detection of web api content scraping: An empirical study of machine learning algorithms, 2017.

[163] David Jeans. Amazon extends moratorium on police use of facial recognition technology. *Forbes*, 2020.

[164] Jinyuan Jia, Ahmed Salem, Michael Backes, Yang Zhang, and Neil Zhenqiang Gong. Memguard: Defending against black-box membership inference attacks via adversarial examples. In *Proc. of CCS*, pages 259–274, 2019.

[165] Shan Jie. China exports facial id technology to zimbabwe. *Global Times*, 12, 2018.

[166] Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. Advances and open problems in federated learning. *Foundations and Trends in Machine Learning*, 14, 2021.

[167] Haje Jan Kamps and Kyle Wiggers. This site tells you if photos of you were used to train the AI, 2022. `https://techcrunch.com/2022/09/21/who-fed-the-ai/`.

[168] Laura Kayali. How facial recognition is taking over a french city. 2019. `https://www.politico.eu/article/how-facial-recognition-is-taking-over-a-french-riviera-city/`.

[169] Mubarak Zed Khan. System soon to identify risky goods, individuals at borders. *Dawn*, 2020. `https://www.dawn.com/news/1584264`.

[170] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[171] Stepan Komkov and Aleksandr Petiushko. Advhat: Real-world adversarial attack on arcface face id system. In *Proc. of ICPR*. IEEE, 2021.

[172] Arvind Krishna. Ibm ceo's letter to congress on racial justice reform. 2020.

[173] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[174] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017.

[175] Ananya Kumar, Tengyu Ma, and Percy Liang. Understanding self-training for gradual domain adaptation. In *Proc. of ICML*, 2020.

[176] Neeraj Kumar, Alexander C Berg, Peter N Belhumeur, and Shree K Nayar. Attribute and simile classifiers for face verification. In *Proc. of ICCV*, 2009.

[177] Ram Shankar Siva Kumar, Magnus Nystrom, John Lambert, Andrew Marshall, Mario Goertzel, Andi Comissoneru, Matt Swann, and Sharon Xiã. Adversarial machine learning–industry perspectives. *arXiv preprint arXiv:2002.05646*, 2020.

[178] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial examples in the physical world. *arXiv:1607.02533*, 2016.

[179] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial machine learning at scale. In *Proc. of ICLR*, 2017.

[180] Xiaojun Lai and Pei-Luen Patrick Rau. Has facial recognition technology been misused? a user perception model of facial recognition scenarios. *Computers in Human Behavior*, 2021.

[181] Frank Landymore. Woman Horrified to Discovery Her Private Medical Photos Were Being Used to Train AI, 2022. https://futurism.com/the-byte/private-medical-photos-ai.

[182] Guillaume Leclerc, Andrew Ilyas, Logan Engstrom, et al. ffcv. https://github.com/libffcv/ffcv/, 2022.

[183] 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.

[184] Katherine Lee, Daphne Ippolito, Andrew Nystrom, Chiyuan Zhang, Douglas Eck, Chris Callison-Burch, and Nicholas Carlini. Deduplicating training data makes language models better. *arXiv preprint arXiv:2107.06499*, 2021.

[185] 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.

[186] Guoyao Li, Shahbaz Rezaei, and Xin Liu. User-Level Membership Inference Attack against Metric Embedding Learning. *arXiv preprint arXiv:2203.02077*, 2022.

[187] Huiying Li, Arjun Nitin Bhagoji, Ben Y Zhao, and Haitao Zheng. Can backdoor attacks survive time-varying models? *arXiv preprint arXiv:2206.04677*, 2022.

[188] 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.

[189] Shaofeng Li, Benjamin Zi Hao Zhao, Jiahao Yu, Minhui Xue, Dali Kaafar, and Haojin Zhu. Invisible backdoor attacks against deep neural networks. *arXiv preprint arXiv:1909.02742*, 2019.

[190] Tao Li and Min Soo Choi. Deepblur: A simple and effective method for natural image obfuscation. *arXiv preprint arXiv:2104.02655*, 1, 2021.

[191] Tao Li and Lei Lin. Anonymousnet: Natural face de-identification with measurable privacy. In *Proc. of CVPR*, 2019.

[192] Yiming Li, Tongqing Zhai, Baoyuan Wu, Yong Jiang, Zhifeng Li, and Shutao Xia. Rethinking the trigger of backdoor attack. *arXiv preprint arXiv:2004.04692*, 2020.

[193] Yiming Li, Ziqi Zhang, et al. Open-sourced dataset protection via backdoor watermarking. *arXiv preprint arXiv:2010.05821*, 2020.

[194] 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.

[195] Bin Liang, Jiachun Li, and Jianjun Huang. We can always catch you: Detecting adversarial patched objects with or without signature. *arXiv preprint arXiv:2106.05261*, 2021.

[196] Chia-Kai Liang, Li-Wen Chang, and Homer H Chen. Analysis and compensation of rolling shutter effect. *IEEE Transactions on Image Processing*, 17(8):1323–1330, 2008.

[197] Cong Liao, Haoti Zhong, Anna Squicciarini, Sencun Zhu, and David Miller. Backdoor embedding in convolutional neural network models via invisible perturbation. *arXiv preprint arXiv:1808.10307*, 2018.

[198] Haibin Ling, Stefano Soatto, Narayanan Ramanathan, and David W Jacobs. Face verification across age progression using discriminative methods. *IEEE Transactions on Information Forensics and security*, 2009.

[199] Jiang Liu, Alexander Levine, Chun Pong Lau, Rama Chellappa, and Soheil Feizi. Segment and complete: Defending object detectors against adversarial patch attacks with robust patch detection. *arXiv preprint arXiv:2112.04532*, 2021.

[200] Kang Liu, Brendan Dolan-Gavitt, and Siddharth Garg. Fine-pruning: Defending against backdooring attacks on deep neural networks. In *Proc. of RAID*, 2018.

[201] Yanpei Liu, Xinyun Chen, Chang Liu, and Dawn Song. Delving into transferable adversarial examples and black-box attacks. In *Proc. of ICLR*, 2016.

[202] Yingqi Liu, Wen-Chuan Lee, Guanhong Tao, Shiqing Ma, Yousra Aafer, and Xiangyu Zhang. Abs: Scanning neural networks for back-doors by artificial brain stimulation. In *Proc. of CCS*, 2019.

[203] Yingqi Liu, Shiqing Ma, Yousra Aafer, Wen-Chuan Lee, Juan Zhai, Weihang Wang, and Xiangyu Zhang. Trojaning attack on neural networks. In *Proc. of NDSS*, 2018.

[204] Natasha Lomas. Google faces fresh class action-style suit in UK over DeepMind NHS patient data scandal. *TechCrunch*, 2022.

[205] Yunhui Long, Vincent Bindschaedler, Lei Wang, et al. Understanding membership inferences on well-generalized learning models. *arXiv preprint arXiv:1802.04889*, 2018.

[206] Giulio Lovisotto, Henry Turner, Ivo Sluganovic, Martin Strohmeier, and Ivan Martinovic. Slap: Improving physical adversarial examples with short-lived adversarial perturbations. *arXiv preprint arXiv:2007.04137*, 2020.

[207] Peter Lyon. Subaru forester is first mainstream model to offer facial recognition technology. *Forbes*, 2018. `https://www.forbes.com/sites/peterlyon/2018/04/30/subaru-forester-is-first-affordable-car-to-get-facial-recognition/?sh=2b939c4c569e`.

[208] Shiqing Ma, Yingqi Liu, Guanhong Tao, Wen-Chuan Lee, and Xiangyu Zhang. Nic: Detecting adversarial samples with neural network invariant checking. In *Proc. of NDSS*, 2019.

[209] Xingjun Ma, Bo Li, Yisen Wang, Sarah M Erfani, Sudanthi Wijewickrema, Grant Schoenebeck, Dawn Song, Michael E Houle, and James Bailey. Characterizing adversarial subspaces using local intrinsic dimensionality. In *Proc. of ICLR*, 2018.

[210] Laura Mackenzie. Surveillance state: how gulf governments keep watch on us. *Wired Magazine*, 2021. https://wired.me/technology/privacy/surveillance-gulf-states/.

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

[212] Shibani Mahtani and Jennifer Hassan. Hong kong protesters are using lasers to distract and confuse. police are shining lights right back. *The Washington Post*, 2020. https://www.washingtonpost.com/world/2019/08/01/hong-kong-protesters-are-using-lasers-distract-confuse-police-are-pointing-them-right-back/.

[213] Pratyush Maini, Mohammad Yaghini, and Nicolas Papernot. Dataset inference: Ownership resolution in machine learning. *arXiv preprint arXiv:2104.10706*, 2021.

[214] Angelica Mari. Brazilian retailer quizzed over facial recognition tech. *ZDNet*, March 2019.

[215] Alessandro Mascillno. Facial recognition in schools: systems deployed in europe and the us amid privacy concerns. 2021. https://www.biometricupdate.com/202110/facial-recognition-in-schools-systems-deployed-in-europe-and-the-us-amid-privacy-concerns.

[216] Marcy Mason. Biometric breakthrough: How cbp is meeting its mandate and keeping america safe. *U.S. Customs and Border Protection Website*. https://www.cbp.gov/frontline/cbp-biometric-testing.

[217] Tiffany May and Amy Chang Chien. Game over: Chinese company deploys facial recognition to limit youths' play. 2021. https://www.nytimes.com/2021/07/08/business/video-game-facial-recognition-tencent.html.

[218] H Brendan McMahan, Eider Moore, Daniel Ramage, S Hampson, and B Agüera y Arcas. Communication-efficient learning of deep networks from decentralized data. *arXiv preprint arXiv:1602.05629*, 2016.

[219] H Brendan McMahan, Daniel Ramage, Kunal Talwar, and Li Zhang. Learning Differentially Private Recurrent Language Models. In *Proc. of ICLR*, 2018.

[220] Blaž Meden, Peter Rot, Philipp Terhörst, Naser Damer, Arjan Kuijper, Walter J Scheirer, Arun Ross, Peter Peer, and Vitomir Štruc. Privacy–enhancing face biometrics: A comprehensive survey. *IEEE Tran. on Information Forensics and Security*, 2021.

[221] Ruby Mellen. Buenos aires is using facial recognition system that tracks child suspects, rights group says. 2020. `https://www.washingtonpost.com/world/2020/10/09/argentina-facial-recognition-juvenile-suspects/`.

[222] Dongyu Meng and Hao Chen. Magnet: a two-pronged defense against adversarial examples. In *Proceedings of the 2017 ACM SIGSAC conference on computer and communications security*, pages 135–147, 2017.

[223] Qiang Meng, Shichao Zhao, Zhida Huang, and Feng Zhou. Magface: A universal representation for face recognition and quality assessment. *arXiv preprint arXiv:2103.06627*, 2021.

[224] Cade Metz and Keith Collins. How an A.I. 'cat-and-mouse game' generates believable fake photos. *The New York Times*, January 2018.

[225] Yuantian Miao, Minhui Xue, et al. The audio auditor: user-level membership inference in Internet of Things voice services. *arXiv preprint arXiv:1905.07082*, 2019.

[226] Agnieszka Mikolajczyk and Michal Grochowski. Data augmentation for improving deep learning in image classification problem. In *Proc. of IIPhDW*, 2018.

[227] Mazda Moayeri, Wenxiao Wang, Sahil Singla, and Soheil Feizi. Spuriosity rankings: Sorting data for spurious correlation robustness. *arXiv preprint arXiv:2212.02648*, 2022.

[228] Seungyong Moon, Gaon An, and Hyun Oh Song. Parsimonious black-box adversarial attacks via efficient combinatorial optimization. In *Proc. of ICML*, 2019.

[229] Paul Mozur. Inside china's dystopian dreams: A.i., shame and lots of cameras. *The New York Times*, 2018.

[230] Paul Mozur. One month, 500,000 face scans: How china is using ai to profile a minority. *The New York Times*, 14, 2019.

[231] Nadia Nahar, Shurui Zhou, Grace Lewis, and Christian Kästner. Collaboration challenges in building ml-enabled systems: Communication, documentation, engineering, and process. *Organization*, 1(2):3, 2022.

[232] Aaron Nech and Ira Kemelmacher-Shlizerman. Level playing field for million scale face recognition. In *Proc. of CVPR*, 2017.

[233] 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.

[234] Hong-Wei Ng and Stefan Winkler. A data-driven approach to cleaning large face datasets. In *Proc. of ICIP*, 2014.

[235] Dinh-Luan Nguyen, Sunpreet S Arora, Yuhang Wu, and Hao Yang. Adversarial light projection attacks on face recognition systems: A feasibility study. In *Proc. of CVPR*, 2020.

[236] Vincent Ni and Yitsing Wang. How to 'disappear' on happiness avenue in beijing. *BBC*, 2020. https://www.bbc.com/news/technology-55053978.

[237] Jorge Nocedal and Stephen Wright. Numerical optimization, series in operations research and financial engineering. *Springer, New York, USA, 2006*, 2006.

[238] The White House Office of Science and Technology Policy. Blueprint for an AI Bill of Rights. 2022. https://www.whitehouse.gov/ostp/ai-bill-of-rights/.

[239] 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/.

[240] José Ramón Padilla-López, Alexandros Andre Chaaraoui, and Francisco Flórez-Revuelta. Visual privacy protection methods: A survey. *Expert Systems with Applications*, 42(9):4177–4195, 2015.

[241] Yingxin Pan, Qunxing Feng, and Chi Zhang. Face recognition at the sales office. *AI Outpost*, 2020. https://mp.weixin.qq.com/s/fWbQ3SD9vB-QdB51T097hw.

[242] Nicolas Papernot, Patrick McDaniel, and Ian Goodfellow. Transferability in machine learning: From phenomena to black-box attacks using adversarial samples. *arXiv:1605.07277*, 2016.

[243] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z. Berkay Celik, and Ananthram Swami. Practical black-box attacks against machine learning. In *Proc. of Asia CCS*, 2017.

[244] Nicolas Papernot, Patrick McDaniel, Somesh Jha, Matt Fredrikson, Z. Berkay Celik, and Ananthram Swami. The limitations of deep learning in adversarial settings. In *Proc. of Euro S&P*, 2016.

[245] Nicolas Papernot, Patrick McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami. Distillation as a defense to adversarial perturbations against deep neural networks. In *Proc. of IEEE S&P*, 2016.

[246] Kaushal Parikh, Dilip Singh, Dinesh Yadav, and Mansingh Rathod. Detection of web scraping using machine learning. *Open access international journal of Science and Engineering*, pages 114–118, 2018.

[247] Sylvain Paris. A gentle introduction to bilateral filtering and its applications. In *Proc. of SIGGRAPH*. 2007.

[248] Omkar M Parkhi, Andrea Vedaldi, Andrew Zisserman, et al. Deep face recognition. In *Proc. of BMVC*, 2015.

[249] Dario Pasquini, Giuseppe Ateniese, and Massimo Bernaschi. Unleashing the tiger: Inference attacks on split learning. In *Proc. of CCS*, 2021.

[250] 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.

[251] Matthias Paulik, Matt Seigel, Henry Mason, Dominic Telaar, Joris Kluivers, Rogier van Dalen, Chi Wai Lau, Luke Carlson, Filip Granqvist, Chris Vandevelde, Sudeep Agarwal, Julien Freudiger, Andrew Byde, Abhishek Bhowmick, Gaurav Kapoor, Si Beaumont, Áine Cahill, Dominic Hughes, Omid Javidbakht, Fei Dong, Rehan Rishi, and Stanley Hung. Federated evaluation and tuning for on-device personalization: System design and applications, 2022. `https://arxiv.org/pdf/2102.08503.pdf`.

[252] Neehar Peri, Neal Gupta, W Ronny Huang, Liam Fowl, et al. Deep k-nn defense against clean-label data poisoning attacks. In *Proc. of ECCV*, 2020.

[253] Christian Petters. Build your own face recognition service using amazon rekognition. *AWS Machine Learning Blog*.

[254] Katya Pivcevic. Police facial recognition use in belarus, greece, myanmar raises rights, data privacy concerns. *Biometric Update*, 2021. `https://www.biometricupdate.com/202103/police-facial-recognition-use-in-belarus-greece-myanmar-raises-rights-data-privacy-concerns`.

[255] Maarten G Poirot, Praneeth Vepakomma, Ken Chang, Jayashree Kalpathy-Cramer, Rajiv Gupta, and Ramesh Raskar. Split learning for collaborative deep learning in healthcare. *arXiv preprint arXiv:1912.12115*, 2019.

[256] Lorien Y Pratt. Discriminability-based transfer between neural networks. *Proc. of NeurIPS*, 1992.

[257] Xiangyu Qi, Tinghao Xie, Saeed Mahloujifar, and Prateek Mittal. Fight Poison with Poison: Detecting Backdoor Poison Samples via Decoupling Benign Correlations. *arXiv preprint arXiv:2205.13616*, 2022.

[258] Ximing Qiao, Yukun Yang, and Hai Li. Defending neural backdoors via generative distribution modeling. In *Proc. of NeurIPS*, 2019.

[259] SiChong Qin. Bias and fairness of evasion attacks in image perturbation. *All Master's Theses*, 2021. `https://digitalcommons.cwu.edu/etd/1517`.

[260] Evani Radiya-Dixit and Florian Tramèr. Data Poisoning Won't Save You From Facial Recognition. *Proc. of ICML*, 2021.

[261] Aditi Raghunathan, Sang Michael Xie, Fanny Yang, John Duchi, and Percy Liang. Understanding and mitigating the tradeoff between robustness and accuracy. *arXiv preprint arXiv:2002.10716*, 2020.

[262] Md Atiqur Rahman, Tanzila Rahman, Robert Laganiere, Noman Mohammed, and Yang Wang. Membership inference attack against differentially private deep learning models. *Trans. Data Priv.*, 11, 2018.

[263] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. In *Proc. of ICML*, pages 8821–8831. PMLR, 2021.

[264] Patrick Reevell. How russia is using facial recognition to police its coronavirus lockdown. April 30, 2021. https://abcnews.go.com/International/russia-facial-recognition-police-coronavirus-lockdown/story?id=70299736.

[265] Grandview Research. Facial recognition market size, share and trends report, 2021 - 2028, 2021. https://www.grandviewresearch.com/industry-analysis/facial-recognition-market.

[266] Herbert Robbins and Sutton Monro. A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407, 1951.

[267] Jeff John Roberts. Walmart's use of sci-fi tech to spot shoplifters raises privacy questions. *Fortune*, 2015. https://fortune.com/2015/11/09/wal-mart-facial-recognition/.

[268] Salvador Rodriguez. Facebook plans to shut down its facial recognition program. 2021. https://www.cnbc.com/2021/11/02/facebook-will-shut-down-program-that-automatically-recognizes-people-in-photos-and-videos-delete-data.html.

[269] Mike Rogoway. Major tech company using facial recognition to id workers. *The Oregonian*, 2020. https://www.govtech.com/public-safety/major-tech-company-using-facial-recognition-to-id-workers.html.

[270] Harrison Rosenberg, Brian Tang, Kassem Fawaz, and Somesh Jha. Fairness properties of face recognition and obfuscation systems. *arXiv preperint arXiv:2108.02707*, 2021.

[271] Antoneta Roussi. Resisting the rise of facial recognition. 2021. https://www.nature.com/articles/d41586-020-03188-2.

[272] Alexandre Sablayrolles, Matthijs Douze, Cordelia Schmid, and Herve Jegou. Radioactive data: tracing through training. In *Proc. of ICML*, 2020.

[273] Shiori Sagawa, Aditi Raghunathan, Pang Wei Koh, and Percy Liang. An investigation of why overparameterization exacerbates spurious correlations. In *Proc. of ICML*, 2020.

[274] Uptin Saiidi. We went inside alibaba's global headquarters. here's what we saw. *CNBC*, 2019. https://www.cnbc.com/2019/09/11/we-went-inside-alibabas-global-headquarters-heres-what-we-saw.html.

[275] Ahmed Salem, Rui Wen, Michael Backes, Shiqing Ma, and Yang Zhang. Dynamic backdoor attacks against machine learning models. *arXiv preprint arXiv:2003.03675*, 2020.

[276] Ahmed Salem, Yang Zhang, Mathias Humbert, Pascal Berrang, Mario Fritz, and Michael Backes. Ml-leaks: Model and data independent membership inference attacks and defenses on machine learning models. *arXiv preprint arXiv:1806.01246*, 2018.

[277] Hadi Salman, Alaa Khaddaj, Guillaume Leclerc, Andrew Ilyas, and Aleksander Madry. Raising the cost of malicious ai-powered image editing. *arXiv preprint arXiv:2302.06588*, 2023.

[278] Pouya Samangouei, Maya Kabkab, and Rama Chellappa. Defense-gan: Protecting classifiers against adversarial attacks using generative models. *arXiv preprint arXiv:1805.06605*, 2018.

[279] 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.

[280] Athena Sayles, Ashish Hooda, Mohit Gupta, Rahul Chatterjee, and Earlence Fernandes. Invisible perturbations: Physical adversarial examples exploiting the rolling shutter effect. In *Proc. of CVPR*, pages 14666–14675, 2021.

[281] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proc. of CVPR*, pages 815–823, 2015.

[282] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, et al. Laion-5b: An open large-scale dataset for training next generation image-text models. *arXiv preprint arXiv:2210.08402*, 2022.

[283] Lukas Schulth, Christian Berghoff, and Matthias Neu. Detecting Backdoor Poisoning Attacks on Deep Neural Networks by Heatmap Clustering. *arXiv preprint arXiv:2204.12848*, 2022.

[284] Ali Shafahi, W Ronny Huang, Mahyar Najibi, Octavian Suciu, Christoph Studer, Tudor Dumitras, and Tom Goldstein. Poison frogs! targeted clean-label poisoning attacks on neural networks. In *Proc. of NeurIPS*, 2018.

[285] 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. *arXiv preprint arXiv:2302.04222*, 2023.

[286] 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.

[287] 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.

[288] 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.

[289] 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.

[290] Juncheng Shen, Xiaolei Zhu, and De Ma. Tensorclog: An imperceptible poisoning attack on deep neural network applications. *IEEE Access*, 7:41498–41506, 2019.

[291] Meng Shen, Zelin Liao, Liehuang Zhu, Ke Xu, and Xiaojiang Du. Vla: A practical visible light-based attack on face recognition systems in physical world. *Proc. of IMWUT*, 2019.

[292] Shiqi Shen, Shruti Tople, and Prateek Saxena. Auror: Defending against poisoning attacks in collaborative deep learning systems. In *Proc. of ACSAC*, 2016.

[293] Yanyao Shen and Sujay Sanghavi. Learning with bad training data via iterative trimmed loss minimization. In *Proc. of ICML*, 2019.

[294] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *Proc. of IEEE S& P*. 2017.

[295] Maya Shwayder. Clearview AI's facial-recognition app is a nightmare for stalking victims. *Digital Trends*, January 2020.

[296] Inderjeet Singh, Satoru Momiyama, Kazuya Kakizaki, and Toshinori Araki. On brightness agnostic adversarial examples against face recognition systems. In *BIOSIG*. IEEE, 2021.

[297] Sahil Singla and Soheil Feizi. Salient imagenet: How to discover spurious features in deep learning? *arXiv preprint arXiv:2110.04301*, 2021.

[298] Aaron Smith. Jetblue will test facial recognition for boarding. *CNN Business*, 2017. `https://money.cnn.com/2017/05/31/technology/jetblue-facial-recognition/index.html`.

[299] Aaron Smith. More than half of u.s. adults trust law enforcement to use facial recognition responsibly. *Pew Research Center*, 2019.

[300] Olivia Solon. Facial recognition's 'dirty little secret': Millions of online photos scraped without consent. *NBC News*, 2019.

[301] Congzheng Song, Thomas Ristenpart, and Vitaly Shmatikov. Machine learning models that remember too much. In *Proc. of CCS*, 2017.

[302] Congzheng Song and Vitaly Shmatikov. Auditing data provenance in text-generation models. In *Proc. of KDD*, 2019.

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

[304] Léa Steinacker, Miriam Meckel, Genia Kostka, and Damian Borth. Facial recognition: A cross-national survey on public acceptance, privacy, and discrimination. *Proc. of ICML LML Workshop*, 2020.

[305] Jacob Steinhardt, Pang Wei W Koh, and Percy S Liang. Certified defenses for data poisoning attacks. In *Proc. of NeurIPS*, 2017.

[306] Octavian Suciu, 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.

[307] Peter Suciu. A photo used to be worth a thousand words, but thanks to social media photos have lost their value. *Forbes online*, 2019.

[308] 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.

[309] 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.

[310] Anshuman Suri, Pallika Kanani, Virendra J Marathe, and Daniel W Peterson. Subject membership inference attacks in federated learning. *arXiv preprint arXiv:2206.03317*, 2022.

[311] 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.

[312] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv:1312.6199*, 2013.

[313] 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. *IPSJ Transactions on Computer Vision and Applications*, 10(1):1–14, 2018.

[314] Chuanqi Tan, Fuchun Sun, et al. A survey on deep transfer learning. In *Proc. of ICANN*. Springer, 2018.

[315] CK Tan. Malaysian police adopt chinese ai surveillance technology. *Nikkei Asia*, 2018. https://asia.nikkei.com/Business/Companies/Chinas-startup-supplies-AI-backed-wearable-cameras-to-Malaysian-police.

[316] Di Tang, XiaoFeng Wang, Haixu Tang, and Kehuan Zhang. Demon in the Variant: Statistical Analysis of {DNNs} for Robust Backdoor Contamination Detection. In *Proc. of USENIX Security*, 2021.

[317] Murat Taskiran, Nihan Kahraman, and Cigdem Eroglu Erdem. Face recognition: Past, present and future (a review). *Digital Signal Processing*, page 102809, 2020.

[318] Anvith Thudi, Hengrui Jia, Ilia Shumailov, and Nicolas Papernot. On the necessity of auditable algorithmic definitions for machine unlearning. *arXiv preprint arXiv:2110.11891*, 3, 2021.

[319] Simen Thys, Wiebe Van Ranst, and Toon Goedemé. Fooling automated surveillance cameras: adversarial patches to attack person detection. In *Proc. of CVPR (workshop)*, 2019.

[320] Florian Tramer, Nicholas Carlini, Wieland Brendel, and Aleksander Madry. On adaptive attacks to adversarial example defenses. *Proc of NeurIPS*, 2020.

[321] Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick Drew McDaniel. Ensemble adversarial training: Attacks and defenses. In *Proc. of ICLR*, 2018.

[322] Florian Tramèr, Reza Shokri, Ayrton San Joaquin, et al. Truth Serum: Poisoning Machine Learning Models to Reveal Their Secrets. *arXiv preprint arXiv:2204.00032*, 2022.

[323] Brandon Tran, Jerry Li, and Aleksander Madry. Spectral signatures in backdoor attacks. In *Proc. of NeurIPS*, 2018.

[324] Marc Treu, Trung-Nghia Le, Huy H Nguyen, Junichi Yamagishi, and Isao Echizen. Fashion-guided adversarial attack on person segmentation. In *Proc. of CVPR*, 2021.

[325] Trojans in artificial intelligence (TrojAI), Feb. 2019. https://www.iarpa.gov/index.php/research-programs/trojai.

[326] Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. Robustness may be at odds with accuracy. *arXiv preprint arXiv:1805.12152*, 2018.

[327] Chun-Chen Tu, Paishun Ting, Pin-Yu Chen, Sijia Liu, Huan Zhang, Jinfeng Yi, Cho-Jui Hsieh, and Shin-Ming Cheng. Autozoom: Autoencoder-based zeroth order optimization method for attacking black-box neural networks. In *Proc. of AAAI*, 2019.

[328] Matthew Turk and Alex Pentland. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 1991.

[329] Alexander Turner, Dimitris Tsipras, and Aleksander Madry. Clean-label backdoor attacks. 2018.

[330] Alexander Turner, Dimitris Tsipras, and Aleksander Madry. Label-consistent backdoor attacks. *arXiv preprint arXiv:1912.02771*, 2019.

[331] Jonathan Uesato, Brendan O'Donoghue, Aaron van den Oord, and Pushmeet Kohli. Adversarial risk and the dangers of evaluating against weak attacks. *arXiv preprint arXiv:1802.05666*, 2018.

[332] Gerrit JJ van den Burg and Christopher KI Williams. On Memorization in Probabilistic Deep Generative Models. *arXiv preprint arXiv:2106.03216*, 2021.

[333] Praneeth Vepakomma, Otkrist Gupta, Tristan Swedish, and Ramesh Raskar. Split learning for health: Distributed deep learning without sharing raw patient data. *arXiv preprint arXiv:1812.00564*, 2018.

[334] Nicholas Vincent and Brent Hecht. Can "conscious data contribution" help users to exert "data leverage" against technology companies? *Proc. of CHI*, 2021.

[335] Nicholas Vincent, Brent Hecht, and Shilad Sen. "data strikes": Evaluating the effectiveness of a new form of collective action against technology companies. In *Proc. of WWW*, 2019.

[336] Nicholas Vincent, Hanlin Li, Nicole Tilly, Stevie Chancellor, and Brent Hecht. Data leverage: A framework for empowering the public in its relations hip with technology companies. In *Proc. of FAccT*, 2021.

[337] Gregory K Wallace. The JPEG still picture compression standard. *IEEE Transactions on Consumer Electronics*, 38(1), 1992.

[338] 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.

[339] 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.

[340] Chengyi Wang, Sanyuan Chen, Yu Wu, Ziqiang Zhang, Long Zhou, Shujie Liu, Zhuo Chen, Yanqing Liu, Huaming Wang, Jinyu Li, et al. Neural codec language models are zero-shot text to speech synthesizers. *arXiv preprint arXiv:2301.02111*, 2023.

[341] 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.

[342] Hao Wang, Yitong Wang, Zheng Zhou, Xing Ji, Dihong Gong, Jingchao Zhou, Zhifeng Li, and Wei Liu. Cosface: Large margin cosine loss for deep face recognition. In *Proc. of CVPR*, pages 5265–5274, 2018.

[343] 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.

[344] 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.

[345] Yunqian Wen, Li Song, Bo Liu, Ming Ding, and Rong Xie. Identitydp: Differential private identification protection for face images. *arXiv preprint arXiv:2103.01745*, 2021.

[346] Emily Wenger, Josephine Passananti, Arjun Nitin Bhagoji, et al. Backdoor attacks against deep learning systems in the physical world. In *Proc. of CVPR*, 2021.

[347] 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.

[348] Eric Wong, Leslie Rice, and J Zico Kolter. Fast is better than free: Revisiting adversarial training. *arXiv:2001.03994*, 2020.

[349] Charles Wright and Mayank Varia. Crypto crumple zones: Enabling limited access without mass surveillance. In *Proc. of EuroS&P*, 2018.

[350] Rebecca N Wright, L Jean Camp, Ian Goldberg, Ronald L Rivest, and Graham Wood. Privacy tradeoffs: Myth or reality? In *Proc. of FC*. Springer, 2002.

[351] Dongxian Wu, Yisen Wang, Shu-Tao Xia, James Bailey, and Xingjun Ma. Skip connections matter: On the transferability of adversarial examples generated with resnets. *arXiv:2002.05990*, 2020.

[352] Yifan Wu, Fan Yang, and Haibin Ling. Privacy-Protective-GAN for face de-identification. *arXiv:1806.08906*, 2018.

[353] 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.

[354] Chong Xiang and Prateek Mittal. Detectorguard: Provably securing object detectors against localized patch hiding attacks. In *Proc. of CCS*, 2021.

[355] Cihang Xie, Zhishuai Zhang, Yuyin Zhou, Song Bai, Jianyu Wang, Zhou Ren, and Alan L Yuille. Improving transferability of adversarial examples with input diversity. In *Proc. of CVPR*, 2019.

[356] Kaidi Xu, Gaoyuan Zhang, Sijia Liu, Quanfu Fan, Mengshu Sun, Hongge Chen, Pin-Yu Chen, Yanzhi Wang, and Xue Lin. Adversarial t-shirt! evading person detectors in a physical world. In *Proc. of ECCV*, 2020.

[357] Weilin Xu, David Evans, and Yanjun Qi. Feature squeezing: Detecting adversarial examples in deep neural networks. In *Proc. of NDSS*, 2018.

[358] Mingfu Xue, Shichang Sun, Zhiyu Wu, Can He, Jian Wang, and Weiqiang Liu. Socialguard: An adversarial example based privacy-preserving technique for social images. *arXiv preprint arXiv:2011.13560*, 2020.

[359] Takayuki Yamada, Seiichi Gohshi, and Isao Echizen. Privacy visor: Method for preventing face image detection by using differences in human and device sensitivity. In *IFIP International Conference on Communications and Multimedia Security*, 2013.

[360] Chaofei Yang, Qing Wu, Hai Li, and Yiran Chen. Generative poisoning attack method against neural networks. *arXiv:1703.01340*, 2017.

[361] Songlin Yang, Wei Wang, Yuehua Cheng, and Jing Dong. A systematical solution for face de-identification. In *Chinese Conference on Biometric Recognition*, pages 20–30. Springer, 2021.

[362] Timothy Yang, Galen Andrew, Hubert Eichner, Haicheng Sun, Wei Li, Nicholas Kong, Daniel Ramage, and Françoise Beaufays. Applied federated learning: Improving google keyboard query suggestions. *arXiv preprint arXiv:1812.02903*, 2018.

[363] Xiao Yang, Yinpeng Dong, Tianyu Pang, Hang Su, Jun Zhu, Yuefeng Chen, and Hui Xue. Towards face encryption by generating adversarial identity masks. In *ICCV*, 2021.

[364] Yao-Yuan Yang and Kamalika Chaudhuri. Understanding Rare Spurious Correlations in Neural Networks. *arXiv preprint arXiv:2202.05189*, 2022.

[365] Zhi Yang, Christo Wilson, Xiao Wang, Tingting Gao, Ben Y Zhao, and Yafei Dai. Uncovering social network sybils in the wild. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 8(1):1–29, 2014.

[366] 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.

[367] Yuanshun Yao, Huiying Li, Haitao Zheng, and Ben Y Zhao. Latent backdoor attacks on deep neural networks. In *Proc. of CCS*, 2019.

[368] Samuel Yeom, Irene Giacomelli, Matt Fredrikson, and Somesh Jha. Privacy risk in machine learning: Analyzing the connection to overfitting. In *Proc. of CSF*, 2018.

[369] Dong Yi, Zhen Lei, Shengcai Liao, and Stan Z. Li. Learning face representation from scratch, 2014.

[370] Zuobin Ying, Yun Zhang, and Ximeng Liu. Privacy-preserving in defending against membership inference attacks. In *Proc. of PPMLP*, pages 61–63, 2020.

[371] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In *Proc. of NeurIPS*, 2014.

[372] Ashkan Yousefpour, Igor Shilov, Alexandre Sablayrolles, Davide Testuggine, et al. Opacus: User-friendly differential privacy library in PyTorch. *arXiv preprint arXiv:2109.12298*, 2021.

[373] Valentina Zantedeschi, Maria-Irina Nicolae, and Ambrish Rawat. Efficient defenses against adversarial attacks. In *Proc. of AISec*, 2017.

[374] Yi Zeng, Minzhou Pan, Hoang Anh Just, Lingjuan Lyu, Meikang Qiu, and Ruoxi Jia. Narcissus: A practical clean-label backdoor attack with limited information. *arXiv preprint arXiv:2204.05255*, 2022.

[375] Chiyuan Zhang, Samy Bengio, et al. Understanding deep learning (still) requires rethinking generalization. *Communications of the ACM*, 2021.

[376] Kaipeng Zhang, Zhanpeng Zhang, Zhifeng Li, and Yu Qiao. Joint face detection and alignment using multitask cascaded convolutional networks. *IEEE Signal Processing Letters*, 23(10):1499–1503, 2016.

[377] Susan Zhang, Stephen Roller, Naman Goyal, et al. OPT: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*, 2022.

[378] Stephan Zheng, Yang Song, Thomas Leung, and Ian Goodfellow. Improving the robustness of deep neural networks via stability training. In *Proc. of CVPR*, 2016.

[379] Yiqi Zhong, Xianming Liu, Deming Zhai, Junjun Jiang, and Xiangyang Ji. Shadows can be dangerous: Stealthy and effective physical-world adversarial attack by natural phenomenon. In *Proc. of CVPR*, 2022.

[380] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. In *Proc. of CVPR*, 2016.

[381] Zhe Zhou, Di Tang, Xiaofeng Wang, Weili Han, Xiangyu Liu, and Kehuan Zhang. Invisible mask: Practical attacks on face recognition with infrared. *arXiv preprint arXiv:1803.04683*, 2018.

[382] Chen Zhu, W Ronny Huang, Ali Shafahi, Hengduo Li, Gavin Taylor, Christoph Studer, and Tom Goldstein. Transferable clean-label poisoning attacks on deep neural nets. In *Proc. of ICML*, 2019.

[383] Zheng-An Zhu, Yun-Zhong Lu, and Chen-Kuo Chiang. Generating adversarial examples by makeup attacks on face recognition. In *Proc. of ICIP*. IEEE, 2019.

[384] Amati Ziv. This israeli face-recognition startup is secretly tracking palestinians. 2019. https://www.haaretz.com/israel-news/business/premium-this-israeli-face-recognition-startup-is-secretly-tracking-palestinians-1.7500359.

[385] Alon Zolfi, Moshe Kravchik, Yuval Elovici, and Asaf Shabtai. The translucent patch: A physical and universal attack on object detectors. *arXiv preprint arXiv:2012.12528*, 2020.

[386] Zihang Zou, Boqing Gong, and Liqiang Wang. Anti-Neuron Watermarking: Protecting Personal Data Against Unauthorized Neural Model Training. *arXiv preprint arXiv:2109.09023*, 2021.