Inference Privacy in Machine Learning

A Dissertation

presented to

the Faculty of the School of Engineering and Applied Science

University of Virginia

In Partial Fulfillment

of the requirements for the degree

Doctor of Philosophy (Computer Science)

by

Anshuman Suri

August 2024

SCHOOL of ENGINEERING & APPLIED SCIENCE

Approval Sheet

This dissertation is submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy (Computer Science)

Author: Anshuman Suri

This dissertation has been read and approved by the Examining Committee:

David Evans, Advisor

Tianhao Wang, Committee Chair

Ferdinando Fioretto, Committee Member

Giuseppe Ateniese, Committee Member

Cong Shen, Committee Member

Accepted for the School of Engineering and Applied Science:

Jennifer L. West, Dean, School of Engineering and Applied Science

August 2024

Abstract

Using machine learning models comes at the risk of leaking information about data used in their training and deployment. This leakage can expose sensitive information about properties of the underlying data distribution, data from participating users, or even individual records in the training data. In this dissertation, we develop and evaluate novel methods to quantify and audit such information disclosure at three granularities: distribution, user, and record.

We begin with a formalization of inference privacy risks as cryptographic games and draw relations, such as reductions and separations, between various types of inference risks. We then propose a formal definition of *distribution inference* attacks that captures previous ratio-based property inference attacks as well as new kinds of attacks, and introduce a metric that quantifies observed leakage. We devise novel white-box and black-box distribution inference attacks and report on a series of experiments across a range of different distributions. We conduct experiments to evaluate distribution inference risks under a range of assumptions about the adversary, including the level of access to the model and the amount and quality of auxiliary data. We also evaluate previously proposed defenses, finding that noise-based defenses are ineffective.

Next, we estimate inference risk at the user level in Federated Learning scenarios with our attacks and demonstrate potent leakage. We also propose methods for injecting malicious behavior in the pre-training stage of a model, whereby selective parameters can be trained to activate differently on particular data to amplify distribution inference in downstream models.

At the individual record level, we prove the necessity of parameter access for *optimal membership inference*, challenging the notion that black-box attacks suffice. Our theory prescribes an exact attack that outperforms state-of-the-art methods without using reference models, which makes it valuable for auditing. Finally, we use membership inference to study memorization in Large Language Models (LLMs), observing near-random inference leakage for most settings, but revealing a connection between distribution inference and membership inference.

Our findings show that privacy leakage spans a spectrum of granularities, making considering multiple forms of leakage essential. Ultimately, our work underscores the urgent need for robust privacy-preserving techniques to mitigate these multifaceted risks in machine learning systems.

To mumma and papa.

Acknowledgements

I have been incredibly fortunate to have a support system and role models many would describe as "too good to be true." I want to express my gratitude to all those who have helped me along the way.

I want to express my deep gratitude for my exceptional mentor, Professor David Evans. It's not an exaggeration to say that he possesses the finest mentorship qualities. Current and former graduate students from my lab, as well as friends from other labs and departments, would undoubtedly agree. Whether celebrating successes or facing failures, he consistently offered support and encouragement, displaying remarkable patience and empathy. Dave has pushed me to be better, not in an overbearing way, but in an encouraging and supportive manner. He has shaped me into the researcher I am today, and I have grown as a scientist and a human because of him. His dedication to rigorous research and meticulous attention to detail are truly inspiring. I am eternally grateful for his guidance, support, and mentorship.

I would also like to thank my excellent collaborators, students and professors alike, who have made my research journey enjoyable. Fnu Suya, Valentin Hartmann, Niloofar Mireshghallah, Yulong Tian, Ahmed Salem, Yuan Tian, Xiao Zhang, Vincent Bindschaedler, and many others. I have learned much from each of them, and I am grateful for the opportunity to work with such talented and dedicated researchers. I would also like to thank current and former members of our research group for providing crucial feedback at critical times and sitting through multiple practice talks throughout my PhD: Hannah Chen, Fnu Suya, Xiao Zhang, Bargav Jayaraman, Josephine Lamp, and Sicheng Zhu. I have also had the privilege of working with exceptionally talented undergraduate students: Yifu Lu, Michael Duan, Tingwei Zhang, Yanjin Chen, and Jingtao Hong, to name a few. In many ways, they have taught me more than I could have ever taught them.

I am also deeply grateful to all the faculty who served as committee members throughout my PhD journey: Vicente Ordóñez-Román, Yuan Tian, Tom Fletcher, Tianhao Wang, Sheng Li, Giuseppe Ateniese, Cong Shen, and Ferdinando Fioretto. Their feedback helped shape my research at various stages.

I am grateful for the support and guidance of Pallika Kanani, Virendra Marathe, Daniel Peterson, and others at Oracle Research during my internship. They have been not only great collaborators but also great

friends and mentors. I also want to thank Parag Agrawal, Tulasi Menon, Bill Dolan, and my colleagues from my time at Microsoft before starting my PhD. It was with them that I first experienced real-world and collaborative research and acquired many coding and management skills that have proven invaluable during my PhD.

I am thankful to the professors and collaborators from my undergraduate days, including Ponnurangam Kumaraguru, Mayank Vatsa, Richa Singh, Prateek Dewan, and Deepak Vijaykeerthy, who helped shape my research career. It was in their labs that I first experienced research and academia, and I am grateful for their guidance and support.

Reflecting on my school days, I also want to express my gratitude to my computer science teacher, Mrs. Ritu Nagpal. As a young child, I had a natural affinity towards computers, and her words of encouragement helped me realize that I could pursue a career in this field.

I would also like to thank the UVA CS community for providing a supportive and enjoyable environment. Everybody in the administrative department has been accommodating and supportive, as a student panicking around for reimbursements and other regular PhD activities, and as a social chair organizing events. The computing department has also been extremely helpful in providing resources and support for everybody. While most researchers might not realize it, our well-managed clusters and servers significantly impact how fast and efficient our research can be.

My friends and family have been another source of tremendous support during this journey. I want to express my gratitude to Sanchit Sinha, Ambar Pal, Harshvardhan Kalra, Shiven Mian, Divam Gupta, Tanya Chowdhury, Rounaq Jhunjhunu Wala, Ivana Apicella, and many others for their unwavering encouragement and support. Divam talked me into picking my undergraduate college over (then) more illustrous options, which ulimately put on on the ramp to research. If it hadn't been for Harshvardhan, I would've missed out on applying to UVA. They have been a constant source of camaraderie and joy, helping me maintain my sanity during the most stressful times. I also want to thank Henry Brorsen, my gym trainer and friend, for his dedication in designing and revising my workout plans to accommodate my busy schedule. As Thomas Jefferson said, "Give about two of them [hours] every day to exercise; for health must not be sacrificed to learning." Henry played a pivotal role in helping me navigate and get comfortable with the gym. The COVID-19 pandemic presented challenges for all of us, and I want to acknowledge Ambar, Harshvardhan, and Divam for their instrumental role in helping me maintain my mental health and stay on track at a time at what was perhaps the lowest point of my PhD.

I want to take a moment to express my deepest appreciation for my girlfriend, Rachita Dash. Her unwavering support and encouragement have been my guiding light throughout my PhD journey. She has not only been a source of inspiration and motivation but has also brought immense joy into my life. She often provided feedback on early iterations of my talks, helped offload work by taking on chores during deadlines, and

been central to my support system. Her presence has been invaluable during the most critical stages of my academic pursuits, and for that, I am genuinely grateful.

Finally, I also want to thank my parents, who have been my biggest supporters and cheerleaders. Their unconditional love, effort, and sacrifices have been humbling and have paved the way for every success I have had and will have. They have provided me with the foundation and the opportunities that have shaped my life. Their belief in me has been unwavering, and their guidance has been invaluable. Everything I am today I owe to them. My success results from their endless support, wisdom, and encouragement. Thank you for always believing in me and being my greatest role models.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Machine-learning models have seen widespread adoption across various fields due to their powerful predictive capabilities [38, 320]. The data used to train these models consists of records, often collected from multiple sources and users, with each user with its distribution $\mathcal{D}_i$. These users together comprise a distribution $\mathcal{D}$ from which training data D is (uniformly) randomly sampled to train some model m_θ (Figure 1.1) with parameters θ. During deployment, additional auxiliary (potentially sensitive) data sources $\mathrm{D}_{\mathrm{aux}}$ may be used to enhance model prediction y for a given query x, such as in retrieval-augmented generation (RAG) [173] or gallery-based matching for face verification [298, 318].

However, trained models are susceptible to disclosure risks, including leaking sensitive information related to their training data. Such leakage can be detrimental to the privacy of individuals and organizations who

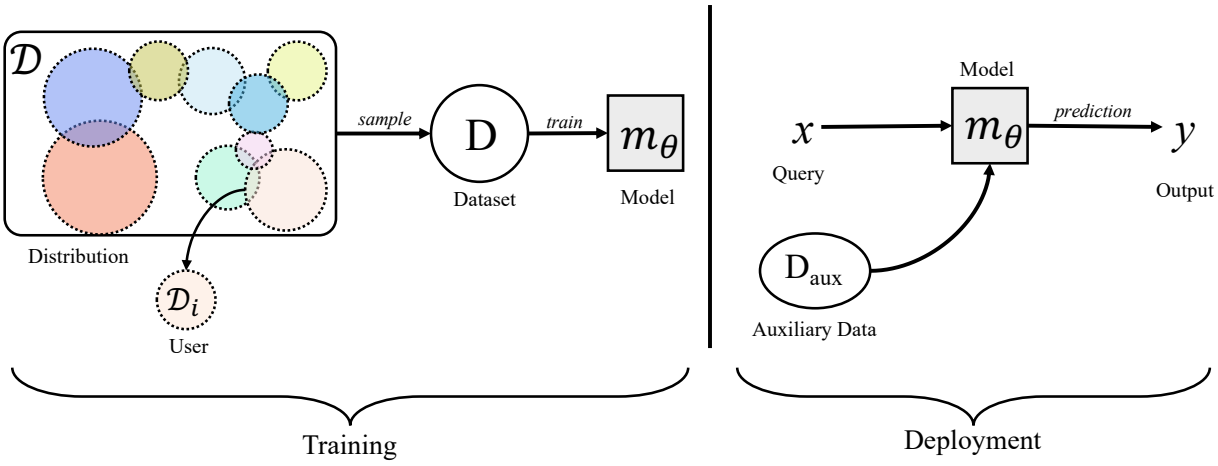


Figure 1.1: Pipeline for a typical machine-learning training and deployment cycle. Note that machine-learning literature often uses "inference" as a misnomer to describe prediction generation. In privacy research, and throughout this dissertation, we use inference to mean deducing sensitive or private information

contribute to this data, thereby diluting confidence in these systems and discouraging participation in the training of models.

Privacy risks in machine learning come in various forms, depending on the granularity of inference and the adversary's knowledge. Among other risks, an adversary with access to a trained model may be able to:

- infer the presence of a specific record (*membership inference*) [279] which can be problematic when sensitive data such as participation in medical/health data is concerned,
- infer certain private attributes of an incomplete record (*attribute inference*) [91] such as the complete address of an individual,
- reconstruct entire training records (*reconstruction*) [19] such as face images of participants, and
- infer properties of the underlying training distribution (*distribution inference*) [16, 293] like the distribution of nodes in a graph of network devices.

These privacy concerns can be exacerbated by adversaries who gain white-box access through *model inversion* to enhance more direct inference attempts [332].

These inference risks are not disjoint and can exist together, making it hard to disentangle interactions between various kinds of leakage. Even though studying inference risk at seemingly different granularities such as record-level and distribution-level is a bit more straightforward, the delineation between some of these risks is unclear. For instance, in datasets where users only contribute a single record, user-level inference can be reduced to record-level membership inference as illustrated in Figure 1.2.

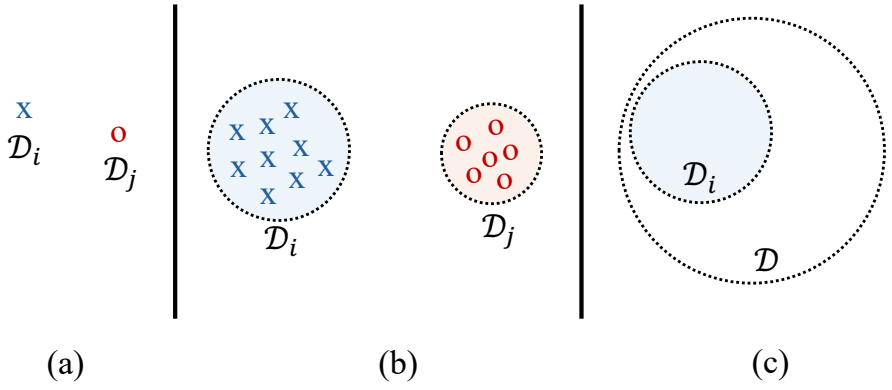


Figure 1.2: Examples that demonstrate how the difference between different granularities of inference risks can be ambiguous. (a) Membership and user-level inference are equivalent in scenarios where each user contributes a single record. (b) When the records corresponding to users are finite and exact, user-inference can be reduced to multi-record membership inference. (c) Inferring the membership of a user in the training distribution is equivalent to distribution inference with a property that checks for the presence of the user.

In this dissertation, we consider inference privacy risks at three granularities:

1. specific records via *membership inference*,
2. groups of records from individuals via *user-level inference*, and
3. properties of the underlying training distribution via *distribution inference*.

These granularities directly relate to users and their data, making them more relatable and actionable for practitioners and policymakers. While membership and user-level inference have a direct impact on individuals, distribution inference can reveal broader patterns in the training data and can also enhance other inference attacks [376]. Our objective is to understand these privacy risks across a broad spectrum of threat models, and to provide empirical methods to audit mitigation strategies by better understanding the trade-offs between privacy and utility.

1.1 Contributions and Road Map

The main contributions of this dissertation are:

1. **Formalizing Inference Risks** (Chapter 2) We begin with a generic outline for inference adversaries (§2.1). We propose a unifying game-based framework for formalizing privacy inference risks of training data in ML (§2.2), which we use to systematize definitions from the literature and to establish relations between them (§2.3).

2. **Distribution Inference** (Chapter 3) Prior works on distribution inference (also known as property inference) focused on inferring differences in relative ratios of certain attributes in underlying datasets, such as the proportion of females [16]. We present a formalization of distribution inference as a cryptographic game, providing a generic definition to capture various statistical properties (§3.1) and an intuitive metric for measuring leakage in simulated attacks (§3.2). We extend current methods to support convolutional layers, enabling deep neural network attacks (§3.3). We find that most information can be gleaned from just a few layers. Our experiments (§3.4) reveal how inference risks vary across datasets and properties, with black-box attacks (including our new KL Divergence Attack, §3.5) often surpassing white-box attacks (§3.6). Additionally, we assess defenses, finding noise-based privacy measures insufficient against distribution inference attacks (§3.7).

3. **User-Level Inference** (Chapter 4) We study user-level inference, a specific instantiation of distribution inference, where the adversary's task is to infer the presence of a user's data in the training process without necessarily having access to exact records used in training. We begin with a user-level inference under Federated Learning (FL), where user data may be scattered across multiple clients. We propose new black-box attacks for user-level membership inference that require only partial knowledge about training subjects and are applicable to both trained ML models and intermediate model states (§4.1).

We also provide a comprehensive evaluation of federation configurations (§4.2) to guide practitioners in mitigating subject membership inference risks. We explore active adversaries (§4.3) that can amplify leakage via manipulation when releasing pretrained models in the wild. Our methods can amplify leakage to near-perfect detection while maintaining negligible performance drops. We also explore detection methods for these manipulated models and present stealthy attacks that remain effective while evading detection (§4.4).

4. **Membership Inference** (Chapter 5) Building upon recent advances in the discrete-time SGD-dynamics literature [187, 378], we provide a more accurate formulation (§5.1) of the optimal membership inference attack that invalidates previous claims [264] about black-box access being sufficient for optimal attacks (§5.2). Our theory prescribes an attack method, Inverse Hessian Attack, which utilizes parameter access via inverse-Hessian vector products. We empirically demonstrate the effectiveness of this attack for auditing membership leakage, all while not having to train any reference models (§5.3). Auditors should thus rethink their approach of using state-of-the-art attacks (that are often black-box) for evaluating privacy risk, and consider using all possible information for tighter empirical bounds.

5. **Memorization in LLMs** (Chapter 6) We begin with an overview (§6.1) of memorization in large language models (LLMs), discussing challenges in defining memorization and key considerations for measuring it (§6.2) and mitigating leakage (§6.3). We then proceed to explore the challenges of evaluating membership inference attacks (MIAs) on large language models (LLMs) (§6.4), finding that most MIAs perform near-randomly across various domains. Our analysis suggests that the large-scale training and extensive data used in LLMs reduce MIA effectiveness due to less memorization of member data and high overlap between members and non-members (§6.4.4). We highlight the ambiguity in defining membership due to n-gram overlaps and show that existing MIAs often misclassify modified members as non-members (§6.5). This chapter ties together themes from distribution inference and membership inference to show how either of them alone are insufficient to fully capture various kinds of leakage involved in such real-world models: studying exact sentence-level membership might not give a clear picture of actual leakage, and there is a lack of a clear boundary between useful distributional properties (like real-world facts) and potentially problematic memorization like user writing style.

Our work highlights privacy breaches at three different granularities and ultimately finding that no single granularity is sufficient for thorough privacy auditing. We discuss the implications of our findings in Chapter 7.

Chapter 2

Formalizing Inference Risks[1]

There is a growing interest in understanding and mitigating the leakage of information about training data under various threat models that capture different adversarial capabilities (e.g., observing model outputs, model parameters, or transcripts of iterative optimization methods) and goals (e.g., membership inference [279], attribute inference [91, 344], property inference [95, 199, 371], and data reconstruction [19, 42]).

An emerging trend in the literature is to capture threat models using *privacy games*. This originates from the seminal work of Wu et al. [344] on formalizing attribute inference. A privacy game is a probabilistic experiment where an *adversary* interacts with a *challenger*. The challenger drives the experiment, invoking the adversary to provide them with information and to allow them to make certain choices, possibly while interacting with oracles controlled by the challenger. The adversary eventually produces a guess for a confidential value. This experiment defines a probability space where the success of the adversary can be measured in terms of the probability of their guess being correct.

The use of games for privacy in ML is inspired by the well-established use of games to define and reason about security properties in cryptography. Cryptographic games are used to standardize and compare security definitions [99, 288], and to structure [26] and even mechanize proofs of security [22, 33]. In comparison, the use of privacy games in the ML literature is still in its infancy:

(1) there are no well-established standards for game-based definitions,

(2) relationships between different privacy games have only been partially explored, and

(3) games are rarely used as an integral part of proofs, despite being especially convenient for this task.

[1]This chapter is largely based on Ahmed Salem, Giovanni Cherubin, David Evans, Boris Köpf, Andrew Paverd, Anshuman Suri, Shruti Tople, Santiago Zanella-Béguelin, *SoK: Let The Privacy Games Begin! A Unified Treatment of Data Inference Privacy in Machine Learning*, in IEEE Symposium on Security and Privacy (S&P), 2023.

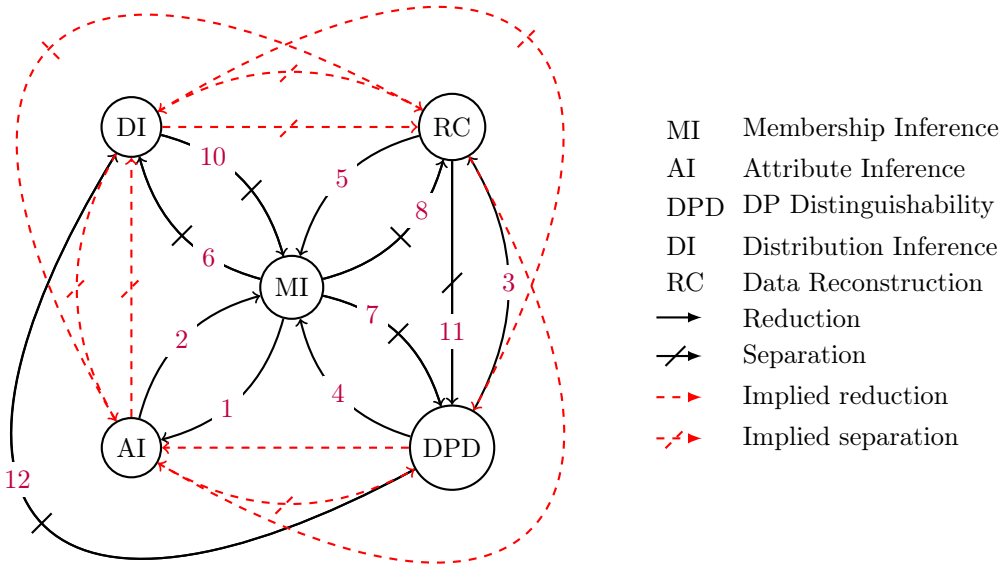


Figure 2.1: Relations among adversary goals (under selected threat models). A solid arrow from node A to B means that security against A (i.e., a nontrivial advantage bound) implies security against B. A struck-through arrow from A to B means that security against A does not imply in general security against B; we show this separation with a construction that is secure against A but completely insecure against B. Dashed arrows are implied by solid arrows. Labels over solid arrows refer to the theorem showing the relationship. Some separations stem from differences in adversary capabilities, e.g., $\mathsf{MI} \not\to \mathsf{RC}$.

This has resulted in many game variants in the literature that attempt to formalize the same adversary goal but have subtle yet important differences. This fragmentation leads to confusion and hinders progress—for membership inference alone, we found variants that differ in details that can change their meaning and substantially alter results. To address this problem, we present the first systematization of knowledge about privacy inference risks in machine learning, going above and beyond the problem left open since 2016 by Wu et al. [344] of merely devising rigorous game-based definitions. Concretely,

- We break down the *anatomy* of game-based privacy definitions for ML systems into individual components: adversary's capabilities and goals, ways of choosing datasets and challenges, and measures of success (§2.1).

- Based on this anatomy, we propose a *unified representation* of five fundamental privacy risks as games: membership inference, attribute inference, property inference, differential privacy distinguishability, and data reconstruction (§2.2).

- Using the game-based framework, we *establish and rigorously prove relationships* between the above risks.

Similarly to the study of *concrete security* in cryptography [25], we define a quantitative notion of *reduction* between privacy properties. Using this notion, we prove a set of relations among the above five privacy risks (§2.3). This allows us to establish, for every possible ordered pair of risks A, B, either a reduction showing that security against A implies security against B, or a separation result showing the impossibility of a generic reduction from A to B. Figure 2.1 summarizes the conclusions of this systematization effort for selected games.

• We present a *case study* (§2.4), where we prove that a scenario described as a variant of membership inference in the literature can actually be decomposed into a combination of membership and property inference. Importantly, in this case we exploit *code-based* reductions, structured as a sequence of games; i.e., our arguments rely on transforming code with a formal semantics. This way of conducting proofs has seen great success in cryptography. However, before our work, it had not reached the same level of rigor when reasoning about privacy inference risks in ML.

2.1 Anatomy of a Privacy Game

Privacy games are parametrized by an adversary ($\mathcal{A}$) and a training pipeline that specifies the training algorithm ($\mathcal{T}$), data distribution ($\mathcal{D}$), and the size of the training dataset (n). A challenger simulates the ML system. The adversary uses their capabilities—defined by a threat model—to interact with the system and infer information about the training dataset.

Game 1: Membership Inference

Input: $\mathcal{A}, \mathcal{T}, n, \mathcal{D}$

1 $S \sim \mathcal{D}^n$ // sample n i.i.d. points from distribution $\mathcal{D}$
2 $b \sim \{0, 1\}$ // flip a fair coin
3 **if** $b = 0$ **then**
4 $\quad z \sim S$ // sample a challenge point uniformly from S
5 **else**
6 $\quad z \sim \mathcal{D}$ // sample a challenge point from $\mathcal{D}$
7 **end**
8 $\theta \leftarrow \mathcal{T}(S)$ // train a model θ
9 $\tilde{b} \leftarrow \mathcal{A}(\mathcal{T}, \mathcal{D}, n, \theta, z)$ // adversary guesses $b = \tilde{b}$

Algorithm 1 formalizes the membership inference experiment of Yeom et al. [356], which we use as a running example. The challenger samples a training dataset S (line 1) and flips a fair coin b (line 2). Depending on the outcome, they either sample a challenge point z from the training dataset S, or from the data distribution $\mathcal{D}$ (lines 3–7). We discuss alternatives for choosing training datasets and challenges in §2.1.2. The challenger then trains a target model θ (line 8), and asks the adversary to make a guess $\tilde{b}$ for b (line 9). In this game, the adversary is given the training algorithm ($\mathcal{T}$), data distribution ($\mathcal{D}$), dataset size (n), target model (θ), and the challenge point (z). We discuss alternatives for adversary's capabilities in §2.1.2 and §2.1.3. The success

of the adversary in making a correct guess ($\tilde{b} = b$) is measured with respect to the baseline of a random guess. Any advantage over this baseline indicates leakage of membership information. We discuss other ways to quantify the adversary's success in §2.1.4.

We now discuss in more detail the building blocks of games described above and highlight common choices.

2.1.1 Adversary Goals

We identify five adversary goals from the literature that enable an adversary to directly infer information about the training dataset of an ML model. We describe these goals informally below and formalize them as games in §2.2.

Membership Inference (MI). The adversary aims to determine whether a specific *record* [279, 356] or *subject* [198, 294] (an entity who may contribute more than one record) was present in the training dataset of the target model. For example, a successful MI attack against a model trained on clinical records of patients with an infective disease can reveal that a target patient was infected.

Attribute Inference (AI). The adversary aims to use the model to infer unknown attributes of a record in the training dataset given partial information about the record [356]. A successful AI attack can result in the reconstruction of sensitive attributes of a target individual.

Distribution Inference (DI). The adversary aims to learn sensitive *statistical* properties of the target model's training distribution. For example, in a malware classifier, the training dataset may have been generated using a particular testing environment, and it may benefit the adversary to learn certain properties of this environment [95]. From an auditing perspective, distribution inference could be used to assess the training dataset for harms (e.g., under-representation) [371].

Differential Privacy Distinguishability (DPD). The adversary aims to determine which of a pair of adjacent datasets (e.g., differing in the data of one record) of their choosing was used to train the target model. This goal recasts differential privacy in a game-based setting by making the adversary explicit. This connection can be used to estimate the differential privacy budget of training pipelines [202, 231, 364].

Data Reconstruction (RC). The adversary aims to reconstruct samples from the training dataset of a target model [19, 41, 42]. A successful attack can partially reconstruct the training dataset, potentially violating confidentiality requirements.

Beyond training data inference. Other adversary goals, such as model stealing [243, 311] and hyperparameter stealing [323] are beyond the scope of this SoK because they do not enable the adversary to directly infer information about the training data. However, the *effects* of these other goals are readily captured by our game-based analysis. For example, a successful model stealing attack that is used as a precursor to membership inference can be represented by changing the adversary access from black-box to white-box (§2.1.3).

2.1.2 Selecting Challenges and Datasets

An important aspect of any privacy game is how the challenges and datasets are selected. In Algorithm 1, the challenge point is a single record z; in other games, the challenge could comprise multiple points or even a data distribution. For the discussion below, we simplify the language by talking about a single challenge point. We discuss below three methods commonly used in the literature.

Randomly sampled. The challenge is sampled from a distribution by the challenger as part of the game [125, 332, 356]. A randomly sampled challenge provides a measure of *average case* privacy. While average case privacy measures the risk for average users, the risk for outliers can be significantly higher.

Externally provided. The challenge is provided as a parameter of the game [125, 198]. This may be used to measure privacy of specific points, i.e., it provides *individual case* privacy.

Adversarially chosen. The challenge is selected by the adversary during the game [48, 202, 231]. Since the adversary can select the most advantageous challenge based on the information provided, this provides a measure of *worst case* privacy, i.e., measuring the risks for all users including outliers. For example, a strong membership inference adversary could choose a challenge that is an outlier w.r.t. the training data distribution, so that a target classification model is unlikely to classify it correctly unless it is included in the training dataset. This setting is usually considered when auditing a system to identify risks.

Additional considerations. When the challenge is externally provided or adversarially chosen, the parameters of the game cannot completely determine a correct adversary guess. Otherwise, security statements that universally quantify over adversaries are void because the quantification includes adversaries with a hardcoded correct guess. This is similar to the difficulty of defining collision resistance of hash functions [262].

Selecting datasets. The training dataset can also be selected using any of the three options above: it can be randomly sampled by the challenger, externally provided, or (partially) chosen by the adversary. The latter can be used to represent the case where the model has been trained on (poisoned) data contributed by potentially malicious users [199, 313].

2.1.3 Adversary Access

Depending on the scenario, the adversary may have different levels of access to the target model, training algorithm, training distribution, and training dataset. This allows the game to capture different threat models, which should ideally match the known or assumed capabilities of real-world adversaries. Most games assume one of two settings: *black-box* or *white-box* access.

Black-box. In this scenario, the adversary only has query access to the target model (e.g., a cloud-hosted model with an inference API) [43]. To formalize this setting, we give the adversary access to the model through an oracle

$$\begin{aligned}&\textbf{Oracle}\mathcal{O}^{\theta}(x):\\ &\quad\textbf{return } \theta(x)\end{aligned}$$

This allows the adversary to query the model θ on inputs of their choosing and observe the responses, but does not reveal internal workings of the model, such as its architecture or weights. Depending on the scenario, the oracle can return a confidence for each label, or only the highest-confidence label [57, 184]. The latter setting matches inference APIs that do not reveal confidence values, like some email spam classifiers or auto-completion systems. Additionally, the oracle can be instrumented to post-process responses, or to only emit responses for queries satisfying a (stateful) predicate, e.g., to enforce a bound N on the number of allowed queries the challenge can initialize $q_0 = 0$ and provide

$$\begin{aligned}&\textbf{Oracle } \mathcal{O}_N^{\theta}(x)\\ &\quad q_\theta \leftarrow q_\theta + 1\\ &\quad \textbf{if } q_\theta \leq N \textbf{ then return } \arg\max \theta(x) \textbf{ else return } \perp\end{aligned}$$

White-box. The white-box setting represents the strongest adversary, who has full direct access to the target model i.e., $\mathcal{A}(\theta, \ldots)$. This obviously provides the adversary with all the capabilities of the black-box setting, but also allows the adversary to inspect the internals of the model including its trained weights [172, 264]. For instance, a model deployed on clients' devices gives white-box access to malicious clients. Alternatively, a successful black-box model stealing attack would enable an adversary to operate in a white-box setting.

Grey-box. In between the black-box and white-box settings, there is a range of *grey-box* threat models in which the adversary has more than black-box but less than full white-box access to the target model. For example, the adversary could know the architecture of a target model, some of its training hyperparameters, or the public model from which the model has been fine-tuned [266, 279]. Such extra information can be the output of a hyperparameter stealing attack [323].

Auxiliary information. In addition to having access to the target model, an adversary may have auxiliary information that could be useful for certain attacks. For example, most MI attacks assume the adversary has access to auxiliary data distributed similarly to the target model's training data, e.g., for building shadow models. This is captured in games by giving the adversary the distribution from which the training data was sampled.

Resource constraints. Most game-based formulations do not explicitly limit the resources available to an adversary, i.e., they consider information-theoretic adversaries. It could be important to consider resource-limited adversaries that can only issue a specific number of queries to an oracle, or can use a certain amount of memory, or are otherwise computationally bounded. Intuitively, limiting these resources can reduce the effectiveness of an attack. These limitations can be specified outside the game as constraints on the adversary, enforced by instrumenting the code of the game (as in Oracle $\mathcal{O}_N^\theta$ above), or incorporated into the measure of success.

2.1.4 Measuring Adversary Success

There are various ways of quantifying the adversary's success in games. We discuss commonly used metrics next.

Attack Success Rate

The *attack success rate* (ASR) measures the expected number of times the adversary succeeds (i.e., wins the game) over multiple runs. ASR is arguably the most intuitive and widespread metric for quantifying adversary success; for example, it matches the attacker's *accuracy* in membership inference.

However, the main drawback of ASR is that it does not take into account the baseline success probability for a given task. For example, if we evaluate an ML model's resilience to attribute inference, the prior distribution of that attribute will play a role in the adversary's success. For instance, if the attribute can only take one value, it is trivial for an adversary to achieve 100% ASR, but this will not be a meaningful measure. Similarly, the prior probability that an example belongs to the training set affects membership inference accuracy.

Ideally, the metric should quantify the success of an adversary relative to a suitable *baseline*. The baseline should represent the *a priori* adversary success rate; that is, it should quantify the adversary's success rate if they used only their prior knowledge and had no access to the model.

Adversary Advantage

The notion of *advantage* is a commonly used metric in cryptography, which relates an adversary's success rate to a baseline. This gives a better intuition of how much an adversary gains by having access to the

model (in any of the forms defined in §2.1.3). In general terms, suppose the adversary is trying to infer some variable p; this could be the membership of a data record or the value of a coin toss. If $\Pr[\mathcal{A} = p]$ is the adversary's success rate (probability to guess p correctly), and G is the baseline success rate, the advantage can be expressed as $\mathsf{Adv}(\mathcal{A}) = {}^{\Pr[\mathcal{A}=p]-G}/_{1-G}$. Assuming $\Pr[\mathcal{A} = p] \geq G$, this metric quantifies the adversary's advantage on a scale of $[0, 1]$ relative to the baseline G; 0 represents no advantage over the baseline and 1 is a perfect attack. When the secret information p is binary with a uniform prior, $G = {}^1/_2$. This leads to the familiar expression $\mathsf{Adv}(\mathcal{A}) = 2\Pr[\mathcal{A} = p] - 1$. Advantage is commonly used as a metric for ML privacy attacks. For example, Yeom et al. [356] define the MI advantage for an adversary $\mathcal{A}$ as follows:

$$\mathsf{Adv}_{\mathsf{MI}}(\mathcal{A}, \mathcal{T}, n, \mathcal{D}) = 2\Pr\Big[\mathsf{MI}(\mathcal{A}, \mathcal{T}, n, \mathcal{D}) : \tilde{b} = b\Big] - 1,$$

where MI is the membership inference experiment in Algorithm 1, and $\Pr[G : E]$ denotes the probability of event E in the probability space defined by game G.

Providing an adequate baseline may be difficult because it may not be possible to accurately model the adversary's knowledge. This issue can often be bypassed by careful design of the game. For example, instead of asking the adversary to reconstruct an arbitrary attribute's value, the game can be designed such that the adversary must distinguish between two equally-likely values of the attribute.

Beyond advantage

Average case metrics such as ASR fail to capture inference risks for individuals or subpopulations. For example, a MI attack against a model may achieve roughly 50% accuracy (with a 50% baseline) on average across the population, yet the same attack may perform better when targeting specific individuals or subpopulations [48, 164]. Having raised similar concerns, Carlini et al. [43] suggest that an adversary should be considered successful if it reliably succeeds even on small number of cases. For instance, a MI attack that achieves a high true positive rate (TPR) at some low false positive rate (FPR) could be consequential even if it has low accuracy.

In this chapter, we focus on advantage as a metric, since it has the following benefits: (1) it has an easy interpretation—it represents the gain of an adversary from having access to the system under scrutiny versus an adversary with only prior knowledge; (2) it is directly related to other metrics, such as ASR (which can be derived directly from it), true and false positive rates (e.g., [356]), and Differential Privacy [50, 125]; (3) if the attacker's challenge is binary (e.g., distinguishing between members and nonmembers), the advantage computed when assuming the two choices have a uniform prior gives a bound for any other prior [50]. Nevertheless, given a game formulation, one can consider other metrics of interest: e.g., area under the ROC curve (AUC-ROC), F1-score, and TPR at fixed FPR thresholds [43].

2.1.5 Consequences of Attacks

The anatomy we presented can be used to specify threat models and quantify the chances that an adversary successfully achieves their goal. However, the *consequences* of a successful attack depend less on the threat model but rather on the adversary's goal (§2.1.1) and on the design of the ML system, e.g., the sensitivity of the training data. For example, the consequences of successful membership inference will be the same irrespective of whether it was performed in a black-box or white-box setting.

2.2 Formalization

In this section we present privacy games for the five adversary goals introduced in §2.1.1. We summarize the notation in Table 2.1 and the threat models considered in all games in Table 2.2.

Table 2.1: Summary of notation

Notation	Description
$\mathcal{T}$	A stochastic training algorithm
$\mathcal{D}$	A distribution over examples
$\mathcal{D}^n$	Distribution of n independent examples from $\mathcal{D}$
$\mathcal{A}$, $\mathcal{A}'$	Adversary procedures sharing mutable state
$z \sim \mathcal{D}$	Draw an example z from $\mathcal{D}$
$S \sim \mathcal{D}^n$	Draw n examples S independently from $\mathcal{D}$
$b \sim \{0,1\}$	Sample a bit b uniformly
$b \sim 0 \oplus_p 1$	Sample 0 with probability p, 1 with probability $1-p$
$y \leftarrow \mathcal{P}(\vec{x})$	Call $\mathcal{P}$ with arguments $\vec{x}$ and assign result to y

2.2.1 Membership Inference

Membership inference aims to predict the participation of an entity in the training dataset of the model. The first (record-level) membership inference attack on supervised learning was proposed by Shokri et al. [279] against ML-based classifiers. Subsequent work has explored membership inference attacks with differing degrees of access to the model (e.g., white-box [172, 264] or label-only attacks [57, 184]), against different types of models (e.g., generative models [53, 112, 118], image segmentation [115], contrastive learning [186], recommender systems [369], and Graph Neural Networks (GNN) [343]), and under entirely different threat models [124, 266, 275].

We present MI variants that have been formalized as games. We divide the games into two categories depending on whether they focus on a single record (*record-level*) or a *user* represented by a collection of records (*user-level*).

Record-level Membership Inference

The most common interpretation of record-level membership inference is given by the game introduced by Yeom et al. [356], which we presented as Algorithm 1 in §2.1. Algorithm 2 below presents a semantically equivalent reformulation MI. The reader can verify that b, θ, z_0 are distributed identically to b, θ, z in Algorithm 1 and thus the joint distribution of $b, \tilde{b}$ is the same in both games. This game considers an adversary with white-box access to the model—they have the model at their disposal and can query it freely, analyze its architecture and parameters, and observe its dynamic behavior. Since the training dataset and the challenge z_0 are sampled from $\mathcal{D}$, this game measures average case MI resilience.

Game 2: MI $\quad$ $\mathsf{MI}^{\mathsf{skew}}$ $\quad$ $\mathsf{MI}^{\mathsf{Adv}}$

Input: $\mathcal{T}, \mathcal{D}, n,$ p, $\mathcal{A}', \mathcal{A}$

1 $S \sim \mathcal{D}^{n-1}$
2 $b \sim \{0,1\}$ $\quad 0 \oplus_p 1$ $\quad \{0,1\}$
3 $z_0 \sim \mathcal{D}$ $\quad \mathcal{D}$ $\quad \mathcal{A}'(\mathcal{T}, \mathcal{D}, n)$
4 $z_1 \sim \mathcal{D}$
5 $\theta \leftarrow \mathcal{T}(S \cup \{z_b\})$
6 $\tilde{b} \leftarrow \mathcal{A}(\mathcal{T}, \mathcal{D}, n,$ p, $\theta, z_0)$

Several variants of the basic MI game have been considered in the literature; some are semantically equivalent (e.g., [125, 164]) whilst others alter its semantics. We next systematize these latter variants using the anatomy presented in §2.1.

Jayaraman et al. [138] consider game $\mathsf{MI}^{\mathsf{skew}}$ which generalizes MI by introducing a parameter p representing the prior membership probability (Algorithm 2, line 2). The original MI game assumes a balanced prior and is recovered as a special case when $p = 1/2$.

Chang and Shokri [48] consider game $\mathsf{MI}^{\mathsf{Adv}}$ in Algorithm 2 which strengthens the adversary by allowing them to select the challenge point (line 3). This game measures worst case MI resilience for an average dataset, i.e., resilience against this variant protects all records—even outliers—against MI. See SMI in Algorithm 11 for an even stronger attack where S is adversarially chosen.

Carlini et al. [43] consider game $\mathsf{MI}^{\mathsf{BB}}$ which differs in two aspects from MI. Firstly, it assumes a black-box adversary who is given only inference access to the model through an oracle, **Oracle** $\mathcal{O}^{\theta}(x)$: **return** $\theta(x)$ (modifying line 9 in Algorithm 1). This is appropriate when the target model is hosted in the cloud or in a trusted execution environment that ensures its confidentiality. Secondly, rather than sampling the challenge point from $\mathcal{D}$ when $b = 1$, the challenger samples it from $\mathcal{D} \setminus S$ (modifying line 6 in Algorithm 1), thus excluding the case where the challenge happens to be in S by chance. This is in contrast to game MI, where nonmembers are sampled from the complete distribution and *may* be contained in S. While doing this seems

intuitive, Yeom et al. [356, p.41] note that it is problematic since an adversary could gain advantage not through access to the model but rather by analyzing $\mathcal{D}$ to infer which points are more likely to have been sampled into S. For instance, consider a distribution $\mathcal{D}$ with support $\{x_0, \ldots, x_m\}$ that assigns probability $1/2$ to x_0 and $1/2m$ to each of $x_1, \ldots, x_m$. An adversary that ignores θ and guesses $\tilde{b} = 0$ if and only if $z = x_0$ has advantage greater than $1/2 - 1/2^n$.

Tramèr et al. [313] introduce a generic privacy game where the goal of the adversary is to guess which point from a universe $\mathcal{U}$ has been included in the training dataset of the target model. They present variants with ($\mathsf{MI}^{\mathsf{Pois}}$) and without ($\mathsf{MI}^{\mathsf{Diff}}$) poisoning, shown in Algorithm 3. $\mathsf{MI}^{\mathsf{Pois}}$ lets the adversary statically poison part of the training dataset (§2.1.2). By considering $\mathcal{U} = \{\hat{z}, \perp\}$, where $\perp$ indicates the absence of an example, the generic game can represent a black-box membership inference attack for a fixed externally provided target example $\hat{z}$. Compared to variants of membership inference discussed previously, this results in training datasets of different sizes depending on the outcome of sampling the challenge z: e.g., in $\mathsf{MI}^{\mathsf{Diff}}$ the model may be trained on $S \cup \{\hat{z}\}$ or just on S. This usually does not make a significant difference as training datasets are large and models do not leak the size of their training dataset. As in $\mathsf{MI}^{\mathsf{BB}}$, values in S are excluded when sampling z, which leads to similar problems.

Game 3: $\mathsf{MI}^{\mathsf{Diff}}$ $\boxed{\mathsf{MI}^{\mathsf{Pois}}}$

Input: $\mathcal{T}, \mathcal{D}, \mathcal{U}, n, \mathcal{A}, \boxed{\mathcal{A}', n'}$
$S \sim \mathcal{D}^n$
$z \sim \mathcal{U} \setminus S$
$\boxed{S' \leftarrow \mathcal{A}'(\mathcal{T}, \mathcal{D}, \mathcal{U}, n')}$ // $|S'| = n'$
$\theta \leftarrow \mathcal{T}(S \cup \{z\} \boxed{\cup S'})$
$\tilde{z} \leftarrow \mathcal{A}(\mathcal{T}, \mathcal{D}, \mathcal{U}, n, \mathcal{O}^\theta(\cdot), \boxed{S'})$
Oracle $\mathcal{O}^\theta(x)$: **return** $\theta(x)$

Other variants. Humphries et al. [125] sample the training dataset and challenge point from different distributions (Algorithm 20); we use this variant as the basis for our case study in §2.4. Tang et al. [300] present single-query variants of membership inference where the adversary is given only the model output on the challenge point. In their base game (Algorithm 4), the adversary selects a universe of $2n$ points from where n points are sub-sampled to construct the training dataset of the target model. The adversary goal is to infer whether a challenge z_j uniformly sampled from the initial $2n$ points was used to train the model, i.e.,, guess $B[j]$, given just the model output on z_j. They also consider variants where the set of $2n$ points is fixed externally, and a worst-case variant where the challenge z_j is selected by the adversary.

Gao et al. [96] consider *deletion inference*, a variant of membership inference in the setting of *machine unlearning*, where the adversary is given access to a model before and after one of two examples is deleted and is asked to guess which example was deleted.

Game 4: $\mathsf{MI}^{\mathsf{SQ}}$

Input: $\mathcal{T}, n, \mathcal{A}, \mathcal{A}'$
$\{z_i\}_{i\in[2n]} \leftarrow \mathcal{A}'(\mathcal{T}, n)$
$B \sim \{0,1\}^{2n}$ s.t. $\sum_{i\in[2n]} B[i] = n$
$S \leftarrow \{z_i \,|\, B[i] = 0\}_{i\in[2n]}$
$\theta \leftarrow \mathcal{T}(S)$
$j \sim [2n]$
$\tilde{b} \leftarrow \mathcal{A}(\mathcal{T}, n, \{z_i\}_{i\in[2n]}, j, \theta(z_j))$

User-level Membership Inference

Privacy laws such as GDPR require generalizing the goal of MI. Instead of focusing on a single record, the interest is now the complete data of an individual. For instance, an auditor would be interested in learning if a user's data—usually modeled as a collection of records—was used to train a target model. User-level membership inference was introduced to model such scenarios. Mahloujifar et al. [198] formalize user-level MI as in Algorithm 5. They consider a meta-distribution $\mathcal{D}$ from where m user distributions are sampled. The adversary targets a particular user contributing a dataset S^*. This game presents the adversary with a task easier than Algorithm 1 since they must infer whether an entire group of records is within the training dataset, i.e., it measures *group* privacy.

Game 5: $\mathsf{MI}^{\mathsf{User}}$

Input: $\mathcal{T}, \mathcal{D}, n, \mathcal{A}, S^*, m$
$b \sim \{0,1\}$
$\mathcal{D}_1, \ldots, \mathcal{D}_m \sim \mathcal{D}$
for $i = 1, \ldots, m-1$ **do**
  $S_i \leftarrow \mathcal{D}_i^n$
end
if $b = 0$ **then**
  $S_m = S^*$
else
  $S_m \leftarrow \mathcal{D}_m^n$
end
$\theta \leftarrow \mathcal{T}(\bigcup_{i=1}^m S_i)$
$\tilde{b} \leftarrow \mathcal{A}\left(\mathcal{T}, \mathcal{D}, n, \mathcal{O}^\theta(\cdot), S^*, m\right)$
Oracle $\mathcal{O}^\theta(x)$: **return** $\theta(x)$

2.2.2 Attribute Inference

In attribute inference (AI) attacks, the adversary aims to infer a sensitive attribute of a target record. Wu et al. [344] were the first to formalize AI, confusingly under the name of *model inversion*. We follow here the more general formalization given by Yeom et al. [356] shown in Algorithm 6. Recently the scope of AI expanded to other settings [137, 374].

Game 6: AI Inv

Input: $\mathcal{T}, \mathcal{D}, n, \mathcal{A}, \varphi, \pi$
$S \sim \mathcal{D}^n$
$b \sim \{0, 1\}$
if $b = 0$ **then**
  $z \sim$ S $\mathcal{D}$
else
  $z \sim \mathcal{D}$
end
$\theta \leftarrow \mathcal{T}(S)$
$\tilde{a} \leftarrow \mathcal{A}(\mathcal{T}, \mathcal{D}, n, \theta, \varphi(z))$

In the AI game, $\varphi(z)$ denotes the adversary's knowledge about the challenge z, and π a function that extracts the information targeted by the attack, e.g., if t represents the target sensitive attributes, then $\pi(z) = t$. The experiment is similar to the basic membership inference experiment (Algorithm 1) except for the information that the adversary is given and the winning condition. The adversary is given $\varphi(z)$ and aims to infer $\pi(z)$. The adversary wins if it correctly predicts these attributes, i.e., $\tilde{a} = \pi(z)$. Training data poisoning can be considered by including adversarially chosen data when training the target model as done for MI in Algorithm 3 ($\mathsf{MI}^{\mathsf{Pois}}$), an instance of the generic game of Tramèr et al. [313].

Model inversion. Another adversary goal with a similar aim to AI is model inversion [332]. Model inversion attacks were introduced by Fredrikson et al. [91] and subsequently formalized by Wang et al. [332] (Inv in Algorithm 6). The difference between attribute inference and model inversion according to Wang et al. [332] is in how the challenge is sampled: in AI it is sampled from the training dataset, while in Inv it is sampled from the distribution $\mathcal{D}$. While AI measures privacy risk for members of a model's training dataset, model inversion measures the privacy loss of publishing the model for members of the underlying population. Whether this is considered a privacy risk is up to debate: a successful attack may lead to the adversary learning information from records that are not part of the training dataset or that do not even exist. Model owners concerned only with the privacy of the training dataset would use the AI game, whilst those concerned about population privacy would prefer Inv.

2.2.3 Reconstruction

Reconstruction attacks aim to recover entire examples in the training dataset of a model. Reconstruction has been studied in various settings, including Graph Neural Networks [374], image classification [267], and text generation [41, 42, 363]. A distilled scenario, where the adversary learns the training data of the target model except for a target example was first formalized by Balle et al. [19] as experiment RC in Algorithm 7.

Reconstruction robustness is parametrized by bounds on the error and success probability and defined as follows.

Game 7: RC $\mid$ $\mathsf{RC}^{\mathsf{Ran}}$

Input: S, $\mathcal{D}, n$, $\pi, \mathcal{T}, \mathcal{A}$

$S \sim \mathcal{D}^{n-1}$

$z \sim \pi$

$\theta \leftarrow \mathcal{T}(S \cup \{z\})$

$\tilde{z} \leftarrow \mathcal{A}(\mathcal{T}, \theta, \mathcal{D}, n, S)$

Definition 1 (Balle et al. [19], Definition 2). A training pipeline is (η, γ)-reconstruction robust with respect to a prior π and reconstruction loss ℓ if for any dataset S and any reconstruction adversary $\mathcal{A}$,

$$\Pr[\mathsf{RC} : \ell(z, \tilde{z}) \leq \eta] \leq \gamma$$

The adversary is given the model θ, training algorithm $\mathcal{T}$, and the training dataset S except for one point z which they need to reconstruct. Game $\mathsf{RC}^{\mathsf{Ran}}$ models how other points in the training dataset are sampled, instead of considering a fixed dataset S. The advantage of an adversary $\mathcal{A}$ against RC w.r.t. a baseline that ignores θ and just samples $\tilde{z}$ from $\mathcal{D}$ is

$$\mathsf{Adv}_{\mathsf{RC}}(\mathcal{A}) = \Pr[\mathsf{RC} : \tilde{z} = z] - \Pr[z, \tilde{z} \sim \mathcal{D} : \tilde{z} = z]$$

Alternatively, one can consider the baseline success of an adversary that picks $\tilde{z}$ according to π,

$$\sup_{\tilde{z} \in \mathrm{supp}(\pi)} \Pr[z \sim \pi : \ell(z, \tilde{z}) \leq \eta] \tag{2.1}$$

Both games can be adapted to consider a poisoning-capable adversary as demonstrated in Algorithm 3.

Reconstruction in language models. Recent work focused on large language models and evaluated reconstruction attacks against them. Attacks can be categorized as untargeted [42] or targeted [41]. Untargeted attacks aim to reconstruct *any* training data from the generative model, whilst targeted attacks aim to reconstruct *specific* training data records, which may have been inserted as canaries during training. To demonstrate the flexibility of privacy games, we formalize an example from each category, as shown in Algorithm 8.

We formalize a black-box untargeted data reconstruction attack by Carlini et al. [42] tailored to large generative language models as $\mathsf{RC}^{\mathsf{Untarg}}$. The authors measure the success of an attack by its true positive rate or recall, that is, the fraction of examples in $\widetilde{S}$ that are in the training dataset S.

We formalize a black-box targeted reconstruction attack by Carlini et al. [41] as $\mathsf{RC}^{\mathsf{Targ}}$. The authors insert a

canary multiple times into the training data as a way to measure unintended memorization in generative models. Canaries are specified by a format sequence $s[\cdot]$ that fixes some tokens and leaves *holes* to be filled with secrets sampled from a randomness space $\mathcal{R}$. For example, $s =$ "the PIN is ◌◌◌◌" with $\mathcal{R}$ being the space of 4-digit decimal numbers. Carlini et al. [41] measure the success of targeted canary reconstruction as the reduction in the guessing entropy of secrets in canaries given the model.

Game 8: $\mathsf{RC}^{\mathsf{Untarg}}$ $\boxed{\mathsf{RC}^{\mathsf{Targ}}}$

Input: $\mathcal{T}, \mathcal{D}, n, \mathcal{A}, \boxed{\mathcal{R}, s, m}$
$S \sim \mathcal{D}^n$
$\boxed{r \sim \mathcal{R}}$
$\theta \leftarrow \mathcal{T}(S \boxed{\cup\{s[r]\}^m})$
$\widetilde{S} \leftarrow \mathcal{A}(\mathcal{T}, \mathcal{D}, n, \mathcal{O}^\theta(\cdot), \boxed{\mathcal{R}, s})$
Oracle $\mathcal{O}^\theta(\mathrm{x})$: **return** $\theta(x)$

Selecting a game. Game RC is appropriate when evaluating the worst case risk of reconstructing an example in the training dataset. It conservatively considers an informed adversary that knows all examples but the target, and incorporates the adversary's background knowledge in a prior. Game $\mathsf{RC}^{\mathsf{Ran}}$ considers an equally informed adversary, but averages the reconstruction risk over the choice of other training examples. Game $\mathsf{RC}^{\mathsf{Untarg}}$ represents a more realistic threat model and should be chosen when evaluating the risk of indiscriminately reconstructing training data, while $\mathsf{RC}^{\mathsf{Targ}}$ is appropriate for auditing the risk of extracting data following certain patterns.

Other variants. Similar to MI, reconstruction attacks have been adapted to the machine unlearning setting. Gao et al. [96] consider *deletion reconstruction*, where an adversary is given access to a model before and after a random training example is deleted and is asked to reconstruct it.

2.2.4 Distribution Inference

Distribution inference attacks do not focus on specific data records, but instead aim at inferring properties about the training data distribution. We next describe two variants of distribution inference. The first is property inference, e.g., where the adversary is interested in learning about the prevalence of specific sensitive attributes in the training data, such as sex or ethnicity. The second is subject-level distribution inference, where the training data is sampled from a mixture of distributions, each corresponding to a subject that may participate in training. The adversary's goal is to infer whether a subject has participated knowing the subject's data distribution rather than concrete samples like in game $\mathsf{MI}^{\mathsf{User}}$ in Algorithm 5.

Property Inference

Property inference attacks were first proposed by Ganju et al. [95] in the white-box setting and by Zhang et al. [371] in the black-box setting. Zhou et al. [376] showed them to be effective against generative models and GANs specifically. We formalize property inference attacks as DI in Algorithm 9, parametrized by two functions $\mathcal{G}_0$, $\mathcal{G}_1$ that transform an underlying distribution. For more details, see §3.1.

Game 9: $\boxed{\mathsf{DI}}$ $\boxed{\mathsf{DI}^{\mathsf{Gen}}}$

Input: $\boxed{\mathcal{D},\ \mathcal{G}_0,\ \mathcal{G}_1}$ $\boxed{\mathcal{D}_0, \mathcal{D}_1}$, $n, \mathcal{T}, \mathcal{A}$
$b \sim \{0, 1\}$
$S \sim \boxed{\mathcal{G}_b(\mathcal{D})^n}$ $\boxed{\mathcal{D}_b^n}$
$\theta \leftarrow \mathcal{T}(S)$
$\tilde{b} \leftarrow \mathcal{A}(\mathcal{T}, \boxed{\mathcal{D},\ \mathcal{G}_0,\ \mathcal{G}_1}$ $\boxed{\mathcal{D}_0, \mathcal{D}_1}, n, \theta)$

$\mathsf{DI}^{\mathsf{Gen}}$ is an equivalent formulation parametrized by two distributions corresponding to the application of $\mathcal{G}_0$, $\mathcal{G}_1$ to the base distribution $\mathcal{D}$ in DI. Hartmann et al. [109] generalize this to more than two distributions.

Similarly to MI and AI, poisoning can be modelled as in Algorithm 3 by letting the adversary choose part of the training dataset of the target model. Mahloujifar et al. [199] and Chaudhari et al. [51] show that poisoning increases inference risk by injecting data to maximize leakage of properties of the training dataset. For instance, in multi-party learning, a malicious participant may contribute poisoned data crafted to amplify property leakage of data from other participants.

Subject-level Distribution Inference

Subject-level distribution inference broadens the scope of user-level membership inference by not assuming access to the user's exact data that may have been used to train a model. Instead, it only requires the adversary know the distribution from which the target user's data is sampled. In §4.1, we present subject membership inference as a special case of distribution inference:

The training data distribution is structured as a mixture of distributions corresponding to a set of subjects. This is a property inference attack because the adversary seeks to infer which of two distributions the training data is sampled from. However, conceptually, the adversary's goal is to infer membership of a subject's data since the only difference between the two distributions is the presence of the target subject in the mixture.

A successful subject-level distribution inference attack can identify if a user's data was used to train the target model without knowing which exact examples were used; i.e., with access to only the user's data distribution and not the sampled dataset as in Algorithm 5.

Game 10: $\mathsf{MI}^{\mathsf{Subj}}$

Input: $\mathcal{T}, \mathcal{D}, \mathcal{D}_*, n, m, \mathcal{A}$

$b \sim \{0, 1\}$
$\mathcal{D}_1, \ldots, \mathcal{D}_m \sim \mathcal{D}$
for $i = 1, \ldots, m-1$ **do**
  $S_i \sim \mathcal{D}_i^n$
end
if $b = 0$ **then**
  $S_m \sim \mathcal{D}_*^n$
else
  $S_m \sim \mathcal{D}_m^n$
end
$\theta \leftarrow \mathcal{T}(\bigcup_{i=1}^m S_i)$
$\tilde{b} \leftarrow \mathcal{A}(\mathcal{T}, \mathcal{D}, \mathcal{D}_*, n, m, \theta)$

2.2.5 Differential Privacy Distinguishability

Differential Privacy Distinguishability (DPD) formalizes the threat model underlying the definition of DP, where the adversary aims to distinguish between models trained on adjacent datasets. We formalize as game DPD in Algorithm 11 the variant corresponding to the *substitute one* adjacency relation, where two datasets are adjacent if one can be obtained from the other by substituting a single record. The DPD game represents a worst-case variant of the membership inference game MI where the training data and challenges are adversarially chosen.

Prior work used DP distinguishing attacks to statistically estimate or audit the privacy of training pipelines [130, 231, 312, 364]. Marathe and Kanani [205] define subject-level differential privacy by considering datasets as adjacent when they differ in the data of a user, which can be seen as a counterpart to user-level membership inference. Humphries et al. [125] and Balle et al. [19] discuss strong membership inference, a threat model in between DPD and MI. In this game, formalized as SMI in Algorithm 11, the adversary knows but does not choose the two adjacent datasets. As mentioned in §2.1.2 this narrows the scope of the measured privacy, e.g., from worst to individual case privacy.

Game 11: DPD SMI

Input: $\mathcal{T}, \mathcal{A}, \boxed{\mathcal{A}', n}, S, z_0, z_1$

$S, z_0, z_1 \leftarrow \mathcal{A}'(\mathcal{T}, n)$ // $|S| = n - 1$
$b \sim \{0, 1\}$
$\theta \leftarrow \mathcal{T}(S \cup \{z_b\})$
$\tilde{b} \leftarrow \mathcal{A}(\mathcal{T}, \theta, S, z_0, z_1)$

Table 2.2: An overview of different games and features of their corresponding threat models. ✓ indicates the game has this feature, – indicates the game does not have this feature, × indicates that the feature is not applicable.

		Adversary Access		Challenge			Training Dataset			Adversary Interest		
Game	Definition	Black-box	White-box	Rand	Adv	Param	Rand	Adv	Param	Record	Subject	Distribution
Membership Inference												
MI	Algorithm 2 [125, 164, 356]	–	✓	✓	–	–	✓	–	–	✓	–	–
$\mathsf{MI}^{\mathsf{skew}}$	Algorithm 2 [138]	–	✓	✓	–	–	✓	–	–	✓	–	–
$\mathsf{MI}^{\mathsf{BB}}$	Algorithm 2 [43]	✓	–	✓	–	–	✓	–	–	✓	–	–
$\mathsf{MI}^{\mathsf{Adv}}$	Algorithm 2 [48]	–	✓	–	✓	–	✓	–	–	✓	–	–
$\mathsf{MI}^{\mathsf{Diff}}$	Algorithm 3 [313]	✓	–	✓	–	–	✓	–	–	✓	–	–
$\mathsf{MI}^{\mathsf{Pois}}$	Algorithm 3 [313]	✓	–	✓	–	–	✓	✓	–	✓	–	–
$\mathsf{MI}^{\mathsf{User}}$	Algorithm 5 [198]	✓	–	–	–	✓	✓	–	–	–	✓	–
MM	Algorithm 20 [125]	–	✓	✓	–	–	✓	–	–	✓	–	✓
$\mathsf{MI}^{\mathsf{SQ}}$	Algorithm 4 [300]	✓	–	✓	–	–	–	✓	–	✓	–	–
Attribute Inference and Model Inversion												
AI	Algorithm 6 [356]	–	✓	✓	–	–	✓	–	–	✓	–	–
Inv	Algorithm 6 [332]	–	✓	✓	–	–	✓	–	–	✓	–	–
Data Reconstruction												
RC	Algorithm 7 [19]	–	✓	✓	–	–	–	–	✓	✓	–	–
$\mathsf{RC}^{\mathsf{Untarg}}$	Algorithm 8 [42]	✓	–	×	×	×	✓	–	–	✓	–	–
$\mathsf{RC}^{\mathsf{Targ}}$	Algorithm 8 [41]	✓	–	✓	–	–	✓	–	–	✓	–	–
Distribution Inference												
DI	Algorithm 9 §3.1	–	✓	×	×	×	✓	–	–	–	–	✓
$\mathsf{MI}^{\mathsf{Subj}}$	Algorithm 10 §4.1	–	✓	✓	–	–	✓	–	–	–	✓	✓
Differential Privacy Distinguishability												
DPD	Algorithm 11 [202, 231]	–	✓	–	✓	–	–	✓	–	✓	–	–
SMI	Algorithm 11 [19, 125]	–	✓	–	–	✓	–	–	✓	✓	–	–

2.3 Relations and Proofs

In this section we establish relationships between privacy games. To this end, we define a notion of *reduction* and use it to translate attacks and guarantees between the five fundamental games from the previous section, or show that no generic connection can exist.

2.3.1 Reductions for Privacy Games

Inspired by notions of reduction from complexity theory and cryptography [14], we introduce reductions between privacy games as a means of comparing the various inference risks. Whilst reductions in cryptography are traditionally based on asymptotic behavior governed by a security parameter, the reductions we define here are closer to those used in *concrete security* proofs, in that the constants underlying the loss incurred in the reduction are made explicit.

Definition 2. We say that game G_1 is reducible to game G_2 if there is a constant $c > 0$ such that, for any adversary $\mathcal{A}$ against G_2, there exists an adversary $\mathcal{B}$ against G_1 such that

$$\mathsf{Adv}_{G_1}(\mathcal{B}) \geq c \cdot \mathsf{Adv}_{G_2}(\mathcal{A})$$

We denote this using the shorthand $G_1 \preceq_c G_2$ and sometimes drop the constant c.

The intuition behind the shorthand is that game G_1 is at most as hard to win as G_2—modulo the constant c. This intuition holds for c around or larger than 1. For $c \ll 1$, however, the lower bound on $\mathsf{Adv}_{G_1}(\mathcal{B})$ can get close to 0, in which case the intuition may be misleading.

Resilience to attacks. Reductions between privacy games imply that attacks against one game translate into attacks against the other. An equivalent reading is the contrapositive, that resilience against attacks in one game implies resilience against attacks in the other.

Definition 3. A game G is *p-resilient* if for all adversaries $\mathcal{A}$ against G,

$$\mathsf{Adv}_G(\mathcal{A}) < p$$

Proposition 1. If $G_1 \preceq_c G_2$ and G_1 is p-resilient then G_2 is p/c-resilient.

Proof. By contradiction: If there is an attack on G_2 with advantage more than p/c, then there is one on G_1 with advantage more than p. □

Proofs of resilience are rare in the literature. Prime examples are results that establish upper bounds on the advantage of a DP distinguisher when the model is trained with differential privacy [125, 356]. The tightest known bound is given in the following proposition.

Proposition 2 (Humphries et al. [125, Theorem 3.1]). Let $\mathcal{T}$ be an (ε, δ)-differentially private training algorithm. Then

$$\mathsf{Adv}_{\mathsf{DPD}}(\mathcal{A}) \leq \frac{e^\varepsilon - 1 + 2\delta}{e^\varepsilon + 1}$$

Therefore, any game the DP distinguisher inference game can be reduced to (see Figure 2.1 for an overview) inherits the security benefits of training with differential privacy via Propositions 1 and 2.

Separation Results. No reductions exist between several games. For them, we show separation results of the form $G_1 \not\preceq G_2$. We establish such results by showing that there is an instance of G_1 that is resilient to attacks whereas its G_2 counterpart is not, and use Proposition 1 to conclude that no reduction exists.

2.3.2 Overview of Relations between Games

Figure 2.1 shows the relations between the five fundamental privacy games. Each node in the figure and in the following theorems refers to the basic game-based definition of the corresponding inference risk, i.e., MI, AI, RC, DPD, and DI.

As expected, DI is fully disconnected: there exists a separation result between it and every other game. This can be attributed to the PI adversary's goal of learning properties of the training data distribution rather than about individual records as in the other games. RC and DPD have the strongest threat models, where the adversary controls the entire training dataset except for one example, and hence are unsurprisingly the hardest to reduce from other games. Finally, MI and AI are reducible to each other and their relatively weak threat models make both RC and DPD reducible to them. For this reason, we use the MI game as the anchor for our proofs. We next present results for a set of edges (solid lines) in Figure 2.1 that imply all other relations.

2.3.3 Reductions

Despite reductions in either direction, MI and AI are separable by constants in the reductions, with resilience against AI easier to achieve than resilience against MI. The following theorems proved by Yeom et al. [356] relate MI and AI.

Theorem 1 ($\mathsf{MI} \preceq_1 \mathsf{AI}$ [356, Theorem 6]). For any adversary $\mathcal{A}_{\mathsf{AI}}$ against attribute inference, there exists an adversary $\mathcal{A}_{\mathsf{MI}}$ against membership inference such that

$$\mathsf{Adv}_{\mathsf{MI}}(\mathcal{A}_{\mathsf{MI}}) = \mathsf{Adv}_{\mathsf{AI}}(\mathcal{A}_{\mathsf{AI}})$$

Theorem 2 ($\mathsf{AI} \preceq_{1/m} \mathsf{MI}$ [356, Theorem 7]). Assume that for all $z \in \mathrm{supp}(\mathcal{D})$, $\varphi(z)$ and $\pi(z)$ uniquely determine z. For any adversary $\mathcal{A}_{\mathsf{MI}}$ against membership inference, there exists an adversary $\mathcal{A}_{\mathsf{AI}}$ against attribute inference such that

$$\mathsf{Adv}_{\mathsf{AI}}(\mathcal{A}_{\mathsf{AI}}) = \frac{1}{m} \cdot \mathsf{Adv}_{\mathsf{MI}}(\mathcal{A}_{\mathsf{MI}})$$

where m is the number of possible values for the target attribute $\pi(z)$.

Resilience against DPD implies resilience against all other attacks except DI. We present the necessary theorems below. The remaining reductions ($\mathsf{RC} \preceq \mathsf{AI}, \mathsf{DPD} \preceq \mathsf{AI}$) are implied by the ones we show.

Balle et al. [19, Theorem 3] show that training pipelines satisfying Rényi DP (and thus (ε, δ)-DP) enjoy resilience against reconstruction attacks. In contrast, a bound on $\mathsf{Adv}_{\mathsf{DPD}}$ does not imply a nontrivial bound on ε in (ε, δ). In fact, $\mathsf{Adv}_{\mathsf{DPD}} \leq \delta$ is equivalent to $(0, \delta)$-DP. Thus, we require an anti-concentration bound on the prior π and that reconstruction succeeds with probability at least $1/2$ to reduce DPD to RC.

Theorem 3 (DPD $\preceq$ RC). Let π be a prior over samples, S a dataset of $n-1$ samples, and ℓ a symmetric reconstruction loss satisfying the triangle inequality. Let $\mathcal{A}$ be an adversary against data reconstruction (RC) w.r.t. S and π that reconstructs its challenge within error η with probability $\gamma \geq 1/2$. Let

$$\alpha = \inf_{z_0 \in \mathrm{supp}(\pi)} \Pr[z_1 \sim \pi : \ell(z_0, z_1) > 2\eta]$$

There exists a DP distinguisher $\mathcal{A}_{\mathsf{DPD}\to\mathsf{RC}}$ such that

$$\mathsf{Adv}_{\mathsf{DPD}}(\mathcal{A}_{\mathsf{DPD}\to\mathsf{RC}}) \geq 2\alpha\left(\gamma - \frac{1}{2}\right)$$

Proof. Observe that $1-\alpha$ is the baseline success of a reconstruction adversary with error 2η (see Equation (2.1)).

Define $\mathcal{A}'_{\mathsf{DPD}\to\mathsf{RC}}$ as in Adversary 12 and $\mathcal{A}_{\mathsf{DPD}\to\mathsf{RC}}$ as in Adversary 13.

Adversary 12: $\mathcal{A}'_{\mathsf{DPD}\to\mathsf{RC}}$

Input: $\mathcal{T}, n$
$z_0, z_1 \sim \pi$
return S, z_0, z_1

Adversary 13: $\mathcal{A}_{\mathsf{DPD}\to\mathsf{RC}}$

Input: $\mathcal{T}, \theta, S, z_0, z_1$
if $\ell(z_0, z_1) \leq 2\eta$ **then**
  $\tilde{b} \sim \{0, 1\}$
else
  $\tilde{b} \leftarrow \mathcal{A}_{\mathsf{SMI}\to\mathsf{RC}}(\mathcal{T}, \theta, S, z_0, z_1)$
end
return $\tilde{b}$

In the DPD game, when $\ell(z_0, z_1) > 2\eta$, which occurs with probability at least α, a similar analysis as in Theorem 9 shows that $\mathcal{A}_{\mathsf{DPD}\to\mathsf{RC}}$ guesses b correctly whenever $\mathcal{A}$ succeeds in reconstructing its challenge within error η. Otherwise, the adversary guesses with probability 1/2. Thus,

$$\begin{aligned}\Pr\left[\mathsf{DPD} : \tilde{b} = b\right] &\geq \Pr\left[\mathsf{DPD} : \tilde{b} = b | \ell(z_0, z_1) > 2\eta\right] \alpha \; + \quad \Pr\left[\mathsf{DPD} : \tilde{b} = b | \ell(z_0, z_1) \leq 2\eta\right] (1-\alpha) \\ &= \gamma\alpha + \tfrac{1}{2}(1-\alpha)\end{aligned}$$

The DPD advantage of $\mathcal{A}_{\mathsf{DPD}\to\mathsf{RC}}$ is

$$\mathsf{Adv}_{\mathsf{DPD}}(\mathcal{A}_{\mathsf{DPD}\to\mathsf{RC}}) = 2\Pr\left[\mathsf{DPD} : \tilde{b} = b\right] - 1 \geq 2\alpha\left(\gamma - \frac{1}{2}\right)$$

□

This is an example of a generic class of reductions: In both games the adversary has the same goal and their

advantage is identically defined, but in game MI the adversary has strictly fewer capabilities than in DPD. Thus, any adversary against MI can be turned into a valid adversary against DPD with the same advantage. In general, a more informed/capable adversary, such as a DP distinguisher, can be used to build a reduction to games with a less informed/capable adversary.

Theorem 4 ($\mathsf{DPD} \preceq \mathsf{MI}$). For any adversary $\mathcal{A}_{\mathsf{MI}}$ against membership inference, there exists a DP distinguisher $\mathcal{A}_{\mathsf{DPD}}$ such that

$$\mathsf{Adv}_{\mathsf{DPD}}(\mathcal{A}_{\mathsf{DPD}}) = \mathsf{Adv}_{\mathsf{MI}}(\mathcal{A}_{\mathsf{MI}})$$

Proof. Let $\mathcal{A}$ be an adversary against $\mathsf{MI}(\mathcal{T}, \mathcal{D}, n)$. We construct an adversary against $\mathsf{DPD}(\mathcal{T}, n)$ as in Adversary 14 and 15. These adversary procedures, when inlined in $\mathsf{DPD}(\mathcal{T}, n)$ (Algorithm 11), result in an experiment semantically equivalent to $\mathsf{MI}(\mathcal{T}, \mathcal{D}, n, \mathcal{A})$ (Algorithm 2). Thus,

$$\mathsf{Adv}_{\mathsf{DPD}}(\mathcal{A}_{\mathsf{DPD}\to\mathsf{MI}}) = \mathsf{Adv}_{\mathsf{MI}}(\mathcal{A}) \qquad \square$$

Adversary 14: $\mathcal{A}'_{\mathsf{DPD}\to\mathsf{MI}}$

Input: $\mathcal{T}, n$
$S \sim \mathcal{D}^{n-1}$
$z_0, z_1 \sim \mathcal{D}$
return S, z_0, z_1

Adversary 15: $\mathcal{A}_{\mathsf{DPD}\to\mathsf{MI}}$

Input: $\mathcal{T}, \theta, S, z_0, z_1$
$\tilde{b} \leftarrow \mathcal{A}(\mathcal{T}, \mathcal{D}, n, \theta, z_0)$
return $\tilde{b}$

Finally, we show that a membership inference attack can be turned into a reconstruction attack, with a constant depending on the size of the support of the training data distribution.

Theorem 5 ($\mathsf{RC} \preceq_{1/|\operatorname{supp}(\mathcal{D})|} \mathsf{MI}$). For any membership inference adversary $\mathcal{A}$ against $\mathsf{MI}(\mathcal{T}, \mathcal{D}, n)$ there exists a reconstruction adversary $\mathcal{B}$ against $\mathsf{RC}^{\mathsf{Ran}}(\mathcal{D}, n, \mathcal{D}, \mathcal{T})$ (i.e., with prior $\pi = \mathcal{D}$) such that

$$\mathsf{Adv}_{\mathsf{RC}^{\mathsf{Ran}}}(\mathcal{B}) = \frac{1}{|\operatorname{supp}(\mathcal{D})|} \cdot \mathsf{Adv}_{\mathsf{MI}}(\mathcal{A})$$

Proof. Consider Algorithm 16, which is equivalent to AI except the adversary is also given S.

The reconstruction advantage of $\mathcal{B}$ coincides with its advantage in AI' in the special case where $\varphi(z) = \perp$ and $\pi(z) = z$, i.e.,, the adversary has to reconstruct all attributes. This is because $\varphi(z_0) = \perp$ and thus the guess

Game 16: AI'

Input: $\mathcal{T}, \mathcal{D}, n, \mathcal{B}, \varphi, \pi$
$S \sim \mathcal{D}^{n-1}$
$z_0, z_1 \sim \mathcal{D}$
$b \sim \{0,1\}$
$\theta \leftarrow \mathcal{T}(S \cup \{z_b\})$
$\tilde{z} \leftarrow \mathcal{B}(\mathcal{T}, \mathcal{D}, n, \theta, S, \varphi(z_0))$

$\tilde{z}$ is independent of z_0 conditioned on $b = 1$.

$$\mathsf{Adv}_{\mathsf{RC}^{\mathsf{Ran}}}(\mathcal{B}) = \Pr\left[\mathsf{AI}' : \tilde{z} = z_0 | b = 0\right] - \Pr\left[\mathsf{AI}' : \tilde{z} = z_0 | b = 1\right]$$

The rest of the proof is similar to the proof of Theorem 2, but we present it for the sake of completeness.

Let $\mathcal{A}$ be an adversary against $\mathsf{MI}(\mathcal{T}, \mathcal{D}, n)$. We construct an adversary $\mathcal{B}$ against $\mathsf{RC}^{\mathsf{Ran}}(\mathcal{D}, n, \mathcal{D}, \mathcal{T})$, shown in Algorithm 17, which uses $\mathcal{A}$ to reconstruct its challenge.

Adversary 17: $\mathcal{B}$

Input: $\mathcal{T}, \theta, S$
$z' \sim \mathrm{supp}(\mathcal{D})$
$\tilde{b} \leftarrow \mathcal{A}(\mathcal{T}, \mathcal{D}, n, \theta)$
if $\tilde{b} = 0$ **then**
  return z'
else
  return $\perp$
end

Denote $\mathcal{D}(z_i)$ the quantity $\Pr[z \sim \mathcal{D} : z = z_i]$, i.e.,, the probability mass of $\mathcal{D}$ at z_i and let $m = |\mathrm{supp}(\mathcal{D})|$. In the following, we use RC to denote the game $\mathsf{RC}^{\mathsf{Ran}}(\mathcal{D}, n, \mathcal{D}, \mathcal{T}, \mathcal{B})$ and MI to denote $\mathsf{MI}(\mathcal{T}, \mathcal{D}, n, \mathcal{A})$.

Since $\mathcal{B}$ guesses $\tilde{z} = z$ if and only if $z' = z$ and $\tilde{b} = 0$, for any $z_i \in \mathrm{supp}(\mathcal{D})$ we have for $\hat{b} \in \{0,1\}$

$$\Pr\left[\mathsf{AI}' : \tilde{z} = z | b = \hat{b}, z = z_i\right] = \frac{1}{m} \Pr\left[\mathsf{AI}' : \tilde{b} = 0 | b = \hat{b}, z = z_i\right] \tag{2.2}$$

Hence, the advantage of $\mathcal{B}$ is

$$\begin{aligned}
\mathsf{Adv}_{\mathsf{RC}^{\mathsf{Ran}}}(\mathcal{B}) &= \sum_{z_i \in \mathrm{supp}(\mathcal{D})} \mathcal{D}(z_i) \left(\Pr\left[\mathsf{AI}' : \tilde{z} = z_0 | b = 0, z = z_i\right] - \Pr\left[\mathsf{AI}' : \tilde{z} = z_0 | b = 1, z = z_i\right] \right) \\
&= \frac{1}{m} \sum_{z_i \in \mathrm{supp}(\mathcal{D})} \mathcal{D}(z_i) \left(\Pr\left[\mathsf{AI}' : \tilde{b} = 0 | b = 0, z = z_i\right] - \Pr\left[\mathsf{AI}' : \tilde{b} = 0 | b = 1, z = z_i\right] \right) \\
&= \frac{1}{m} \left(\Pr\left[\mathsf{AI}' : \tilde{b} = 0 | b = 0\right] - \Pr\left[\mathsf{AI}' : \tilde{b} = 0 | b = 1\right] \right) \\
&= \frac{1}{m} \mathsf{Adv}_{\mathsf{MI}}(\mathcal{A})
\end{aligned}$$

The penultimate equality holds because b and z are independent. The last equality holds because game $\mathsf{AI}'(\mathcal{T}, \mathcal{D}, n, \mathcal{B})$ matches game $\mathsf{MI}(\mathcal{T}, \mathcal{D}, n, \mathcal{A})$ and so the joint distribution of $\tilde{b}, b$ is the identical in both games. □

2.3.4 Separation Results

Theorem 6 (MI $\not\preceq$ DI). Resilience against membership inference does not imply resilience against property inference.

Proof. We construct a training pipeline $(\mathcal{T}, \mathcal{D}_b, n)$ that is arbitrarily resilient to membership inference for $b \in \{0, 1\}$. Yet, we exhibit a property inference attack against it that achieves perfect advantage.

Let $\mathcal{D}_b = \text{Bernoulli}(p_b)$ with $p_0 \neq p_1$ and $\mathcal{T}(S) = \sum_{x \in S} x$. As shown in Theorem 7, the advantage of a membership inference adversary against $(\mathcal{T}, \mathcal{D}_b, n)$ is at most $1/\sqrt{n}$. However, as n grows, $\mathcal{T}(S)/n$ is an unbiased estimator for the mean p_b, which allows a property inference adversary to easily distinguish between $\mathcal{D}_0$ and $\mathcal{D}_1$, particularly when p_0 and p_1 are far apart. □

Theorem 7 (MI $\not\preceq$ DPD). Resilience against membership inference does not imply resilience against DP distinguishability.

Proof. We show that there are training pipelines that are arbitrarily resilient against membership inference attacks but completely insecure against DP distinguishing attacks.

We construct a training pipeline $(\mathcal{T}, \mathcal{D}, n)$ such that the MI advantage of an adversary against it is at most $1/\sqrt{n}$, and so vanishes as n grows. Yet, we exhibit a DP distinguisher against the pipeline that achieves perfect advantage.

Let $\mathcal{D} = \text{Bernoulli}(p)$ and $\mathcal{T}(S) = \sum_{x \in S} x$. Consider Algorithm 18. If the adversary were only given z_0, this game would be equivalent to the basic MI game (Algorithm 1). Since the adversary is given strictly more

Game 18: MI'
Input: $\mathcal{T}, \mathcal{D}, n, \mathcal{A}$
$b \sim \{0,1\}$
$S \sim \mathcal{D}^{n-1}$
$z_0, z_1 \sim \mathcal{D}$
$\theta \leftarrow \mathcal{T}(S \cup \{z_b\})$
$\tilde{b} \leftarrow \mathcal{A}(\mathcal{T}, \mathcal{D}, n, \theta, z_0, z_1)$

information, any bound on its advantage in this game would also bound the MI advantage of adversaries against the training pipeline. The adversary must distinguish between two simple hypotheses:

- $H_0 : \theta \sim \text{Binomial}(n-1, p) + z_0$
- $H_1 : \theta \sim \text{Binomial}(n-1, p) + z_1$

When $z_0 = z_1$, these coincide and the advantage of the adversary is 0. Otherwise, without loss of generality, assume $z_b = b$. By the Neyman-Pearson lemma, a likelihood ratio test yields the most powerful test for a significance α (i.e.,, Type-I error, false positive rate). Let f and F be the probability mass and cumulative distribution function of Binomial$(n-1, p)$, respectively. The likelihood ratio is

$$\Lambda(\theta = k) = \begin{cases} \infty & \text{if } k = 0 \\ 0 & \text{if } k = n \\ \frac{f(k)}{f(k-1)} = \frac{(n-k)p}{k(1-p)} & \text{otherwise} \end{cases}$$

The test rejects H_0 when $\Lambda(\theta) < c$, for some c. The false positive rate α (the probability of rejecting H_0 when H_0 is true) is

$$\begin{aligned} \Pr_{H_0}(\Lambda(\theta) < c) &= \Pr_{H_0}\left(\frac{(n-k)p}{k(1-p)} < c\right) \\ &= \Pr_{H_0}\left(k > \frac{np}{p + c(1-p)}\right) \\ &= 1 - F\left(\frac{np}{p + c(1-p)}\right) \end{aligned}$$

The false negative rate β is

$$\begin{aligned} \Pr_{H_1}(\Lambda(\theta) \geq c) &= \Pr_{H_1}\left(\frac{(n-k)p}{k(1-p)} \geq c\right) \\ &= \Pr_{H_1}\left(k \leq \frac{np}{p + c(1-p)} - 1\right) \\ &= F\left(\frac{np}{p + c(1-p)} - 1\right) \end{aligned}$$

Now, take $p = 0.5$ and assume that $n \geq 4$ and that n is even so that the mode of Binomial$(n-1, p)$ is $n/2$. The MI advantage of the adversary is

$$\begin{aligned}\mathsf{Adv}_{\mathsf{MI}}(\mathcal{A}) &= \frac{1}{2}(f(0) + f(n-1) + (1-\alpha-\beta)) \\ &= f(0) + \frac{1}{2} f\left(\frac{np}{p + c(1-p)}\right) \\ &\leq \frac{1}{2^{n-1}} + \frac{f(n/2)}{2} \\ &\leq \frac{1}{2\sqrt{n}} + \frac{1}{2\sqrt{n}} = \frac{1}{\sqrt{n}}\end{aligned}$$

On the other hand, a DP distinguisher $\mathcal{A}$ that chooses $z_0 = 0, z_1 = 1$, an arbitrary S, and that guesses $\tilde{b} = \theta - \sum_{x \in S} S$, has perfect advantage $\mathsf{Adv}_{\mathsf{DPD}}(\mathcal{A}) = 1$. □

Theorem 8 (MI $\not\preceq$ RC). Resilience against membership inference does not imply resilience against reconstruction.

Proof. It suffices to show that the training pipeline from Theorem 7, which is resilient to membership inference attacks, admits a reconstruction attack. For this, recall that in RC the adversary knows the dataset S (but not the target sample z). For the pipeline $(\mathcal{T}, \mathcal{D}, n)$ given in Theorem 7, z can be perfectly reconstructed since $z = \theta - \sum_{x \in S} x$. □

This last counterintuitive separation result stems from a discrepancy between adversary capabilities: The MI game is based on an average case scenario, while the reconstruction game assumes a more informed worst-case adversary. By considering a membership adversary matching the capabilities of the adversary in the RC game, we can build a reduction to data reconstruction. We show this in Theorem 9, which reduces the strong membership inference game SMI (Algorithm 11) to game RC.

Theorem 9 (SMI $\preceq$ RC). Let z_0, z_1 be two samples, S a dataset of $n-1$ samples, and ℓ a symmetric reconstruction loss satisfying the triangle inequality. Let $\mathcal{A}$ be an adversary against data reconstruction (RC) w.r.t. S and the uniform prior on $\{z_0, z_1\}$ that reconstructs its challenge with error $\eta < \ell(z_0, z_1)/2$ with probability γ. Then, there exists a strong membership inference adversary $\mathcal{A}_{\mathsf{SMI}\to\mathsf{RC}}$ such that

$$\mathsf{Adv}_{\mathsf{SMI}}(\mathcal{A}_{\mathsf{SMI}\to\mathsf{RC}}) \geq 2\gamma - 1$$

Proof. Define $\mathcal{A}_{\mathsf{SMI}\to\mathsf{RC}}$ as in Adversary 19. For any $\tilde{z}$, we have from the triangle inequality,

$$\begin{aligned}&\ell(z_0, \tilde{z}) < \ell(z_0, z_1)/2 < (\ell(z_0, \tilde{z}) + \ell(\tilde{z}, z_1))/2 \\ \implies\ &\ell(z_0, \tilde{z}) < \ell(z_1, \tilde{z})\end{aligned}$$

Adversary 19: $\mathcal{A}_{\mathsf{SMI}\to\mathsf{RC}}$

Input: $\mathcal{T}, \theta, S, z_0, z_1$
$\tilde{z} \leftarrow \mathcal{A}(\mathcal{T}, \theta, S)$
if $\ell(z_0, \tilde{z}) < \ell(z_1, \tilde{z})$ **then**
  return 0
else
  return 1
end

Therefore, when $b = 0$ in SMI and $\mathcal{A}$ succeeds in reconstructing z_0 within error η, $\mathcal{A}_{\mathsf{SMI}\to\mathsf{RC}}$ guesses correctly. Similarly, when $b = 1$ and $\mathcal{A}$ succeeds in reconstructing z_1 within error η, $\mathcal{A}_{\mathsf{SMI}\to\mathsf{RC}}$ guesses correctly. Thus, $\mathcal{A}_{\mathsf{SMI}\to\mathsf{RC}}$ guesses b correctly at least with probability γ and

$$\mathsf{Adv}_{\mathsf{SMI}}(\mathcal{A}_{\mathsf{SMI}\to\mathsf{RC}}) = 2\Pr\Big[\mathsf{SMI}(\cdots) : \tilde{b} = b\Big] - 1 \geq 2\gamma - 1 \qquad \square$$

Theorem 10 (DI $\not\preceq$ MI). Resilience against property inference does not imply resilience against membership inference.

Proof. We exhibit a pipeline resilient to property inference that is completely vulnerable to a membership inference attack.

Let $\mathcal{D}$ be an arbitrary distribution and define $\mathcal{D}_b$ so that $z \sim \mathcal{D}_b \equiv x \sim \mathcal{D}; z \leftarrow (x, b)$. Let $n > 0$ and define

$$\mathcal{T}(S) = \{x \in S | (x, y) \in S\}$$

A membership inference adversary against $\mathsf{MI}(\mathcal{T}, \mathcal{D}, n)$ that given $\theta, z_0 = (x, y)$ returns 0 if and only if $x \in S$ achieves the maximum advantage, i.e.,,

$$1 - \Pr[S \sim \mathcal{D}^n; x \sim \mathcal{D} : x \in S]$$

However, a property inference adversary gets no information about b as θ and b are independent, so its advantage is 0. $\square$

Theorem 11 (RC $\not\preceq$ DPD). Resilience against reconstruction does not imply resilience against DP distinguishability.

Proof. Balle et al. [19, Theorem 5] show that resilience against reconstruction w.r.t. all priors in a family of distributions concentrated on all ordered pairs of distinct examples implies (ε, δ)-DP, and hence via Proposition 2 resilience against DPD.

However, resilience against a single prior π, even if its support includes all possible examples, is clearly insufficient to guarantee resilience against DPD. To see why, consider a deterministic DPD adversary that picks S, z_0, z_1. Given error bound η and success probability γ, all reconstruction adversaries can have error larger than η when $z \notin \{z_0, z_1\}$ but reconstruct $z \in \{z_0, z_1\}$ perfectly, as long as $\Pr[z \sim \pi : z \in \{z_0, z_1\}] < \gamma$, i.e.,, the probability mass of the prior on $\{z_0, z_1\}$. The situation is worse when $z_0, z_1 \notin \text{supp}(\pi)$, where resilience against reconstruction for arbitrary η, γ is compatible with perfect DPD advantage. □

Theorem 12 (DPD $\not\preceq$ DI). Resilience against DP distinguishability does not imply resilience against property inference.

Proof. Let $\varepsilon, \delta \in (0, 1)$. We build two training pipelines $(\mathcal{T}, \mathcal{D}_b, n)$, $b \in \{0, 1\}$, that satisfy (ε, δ)-DP and thus are resilient against DPD (see Proposition 2). We then show an adversary against $\mathsf{DI}(\mathcal{D}_0, \mathcal{D}_1, n, \mathcal{T})$ whose advantage grows with n.

Let $\mathcal{D}_b = \text{Bernoulli}(p_b)$ with $p_0 \neq p_1$ and define $\mathcal{T}(S) = \sum_{x \in S} x + \mathcal{N}(0, \sigma^2)$ where $\sigma^2 = 2\ln(1.25/\delta)\varepsilon^{-1}$. Since the sum above has sensitivity 1 and $\mathcal{T}$ is the standard Gaussian mechanism, the training pipeline is (ε, δ)-DP.

Note that for S sampled from $\mathcal{D}_b$, the random variable $\mathcal{T}(S)$ is distributed as $\text{Binomial}(n, p_b) + \mathcal{N}(0, \sigma^2)$. We can use Berry-Esséen theorem to approximate the binomial distribution with a normal distribution, so that approximately

$$\begin{aligned}\mathcal{T}(S) &\sim \mathcal{N}(np_b, np_b(1-p_b)) + \mathcal{N}(0, \sigma^2) \\ &\sim \mathcal{N}(np_b, np_b(1-p_b) + \sigma^2)\end{aligned}$$

The approximation error is $O(\sqrt{n})$. $\mathcal{T}(S)/n$ is an unbiased estimator for p_b with variance $p_b(1-p_b) + {}^{\sigma^2}/_n$. Since σ does not depend on n, as n grows the approximation error and the variance of the estimate decrease. This allows a property inference adversary to distinguish between $\mathcal{D}_0$ and $\mathcal{D}_1$, particularly when p_0 and p_1 are far apart. □

Ateniese et al. [16, Section 4.2] were the first to observe that differential privacy does not protect against property inference and provided a practical counterexample: a differentially private k-means network traffic classifier that nonetheless leaks the presence of traces from Google.com web traffic in their training dataset. However, their argument remains informal, appealing to visual differences in the centroids of just two trained models. Suri et al. [295, Section V] give a more compelling example where property inference risk remains high on neural networks trained with DP-SGD.

2.4 Case Study: Mixture Model Membership Inference

We present a case study where we showcase the expressive power and rigor of privacy games. In particular, we show that a novel variant of membership inference can be decomposed into a combination of membership and property inference. This complex relationship goes beyond the direct reductions presented in §2.3. In our proofs, we exploit *code-based* reductions structured as a sequence of games; i.e., our arguments rely on transforming code with a formal semantics.

The game we target is due to Humphries et al. [125], who use it to model membership inference attacks in the presence of dependencies in the training data. In their game (MM in Algorithm 20), the training data follows a two-stage *mixture model*. Examples in the training dataset and the target example are chosen independently from two data distributions, $\mathcal{D}_k$ and $\mathcal{D}_{k'}$, which are chosen uniformly at random without replacement from K possible distributions $\boldsymbol{\mathcal{D}} = \{\mathcal{D}_1, \ldots, \mathcal{D}_K\}$.

Game 20: MM G_0

Input: $\mathcal{T}, \boldsymbol{\mathcal{D}}, n, \mathcal{A}$
$k \sim [K]$
$k' \sim [K] \setminus \{k\}$
$S \sim \mathcal{D}_k^n$
$\theta \leftarrow \mathcal{T}(S)$
$b \sim \{0, 1\}$
if $b = 0$ **then**
  $z \sim S$ $z \sim \mathcal{D}_k$
else
  $z \sim \mathcal{D}_{k'}$
end
$\tilde{b} \leftarrow \mathcal{A}(\mathcal{T}, \boldsymbol{\mathcal{D}}, n, \theta, z)$

Game 21: G_1

Input: $\mathcal{T}, \boldsymbol{\mathcal{D}}, n, \mathcal{A}$
$k \sim [K]$
$k' \sim [K] \setminus \{k\}$
$z \sim \mathcal{D}_k$
$b \sim \{0, 1\}$
if $b = 0$ **then**
  $S \sim \mathcal{D}_k^n$
else
  $S \sim \mathcal{D}_{k'}^n$
end
$\theta \leftarrow \mathcal{T}(S)$
$\tilde{b} \leftarrow \mathcal{A}(\mathcal{T}, \boldsymbol{\mathcal{D}}, n, \theta, z)$

We show that MM can be decomposed into a property inference goal (inferring the training data distribution) and a membership inference goal (inferring whether a target example has been sampled from the training data distribution $\mathcal{D}_k$ or from the training dataset S).

Theorem 13. For any adversary $\mathcal{A}$ against MM, there exist adversaries $\mathcal{A}^i_{\mathsf{MI}}$ and $\mathcal{A}^{i,j}_{\mathsf{DI}}$ such that

$$\mathsf{Adv}_{\mathsf{MM}}(\mathcal{A}) \leq \max_{i\in[K]} \mathsf{Adv}_{\mathsf{MI}_i}(\mathcal{A}^i_{\mathsf{MI}}) + \max_{i\neq j\in[K]} \mathsf{Adv}_{\mathsf{DI}_{i,j}}(\mathcal{A}^{i,j}_{\mathsf{DI}})$$

where MI_i is the membership inference game with training data distribution $\mathcal{D}_i$, and in $\mathsf{DI}_{i,j}$ the property to infer is whether the training data distribution is $\mathcal{D}_i$ or $\mathcal{D}_j$.

Proof. Let $\mathcal{A}$ be an adversary against MM. Consider G_0 shown alongside MM in Algorithm 20. Its only difference w.r.t. MM is that when $b = 0$, the example z is freshly sampled from the training data distribution $\mathcal{D}_k$ rather than from the training dataset S. Conditioned on $b = 0, k = i$, distinguishing between games G_0 and MM is as difficult as winning a membership inference game. We show this using a black-box reduction: fixing $k = i$, we construct an adversary $\mathcal{A}^i_{\mathsf{MI}}$ that uses $\mathcal{A}$ as an oracle to guess the challenge bit b in game MI_i (see Algorithm 22). $\mathcal{A}^i_{\mathsf{MI}}$ simply forwards its inputs $\mathcal{T}, n, \theta, z$ to $\mathcal{A}$, passing to it in addition the distribution set $\boldsymbol{\mathcal{D}}$.

Game 22: MI_i

Input: $\mathcal{T}, \mathcal{D}_i, n, \mathcal{A}$
$S \sim \mathcal{D}_i^n$
$\theta \leftarrow \mathcal{T}(S)$
$b \sim \{0, 1\}$
if $b = 0$ **then** $z \sim S$ **else** $z \sim \mathcal{D}_i$
$\tilde{b} \leftarrow \mathcal{A}^i_{\mathsf{MI}}(\mathcal{T}, \mathcal{D}_i, n, \theta, z)$

Adversary 23: $\mathcal{A}^i_{\mathsf{MI}}$

Input: $\mathcal{T}, \mathcal{D}_i, n, \theta, z$
return $\mathcal{A}(\mathcal{T}, \boldsymbol{\mathcal{D}}, n, \theta, z)$

Game MM conditioned on $b = 0, k = i$ is equivalent to MI_i conditioned on $b = 0$. Likewise, game G_0 conditioned on $b = 0, k = i$ is equivalent to MI_i conditioned on $b = 1$. Hence,

$$\begin{aligned}\mathsf{Adv}_{\mathsf{MI}_i}(\mathcal{A}^i_{\mathsf{MI}}) &= \Pr\left[\mathsf{MI}_i : \neg\tilde{b} \mid \neg b\right] - \Pr\left[\mathsf{MI}_i : \neg\tilde{b} \mid b\right] \\ &= \Pr\left[\mathsf{MM} : \neg\tilde{b} \mid \neg b, k = i\right] - \Pr\left[G_0 : \neg\tilde{b} \mid \neg b, k = i\right] \end{aligned} \tag{2.3}$$

Game MM conditioned on $b = 1$ is equivalent to G_0 conditioned on $b = 1$, and so we have

$$\begin{aligned}&\mathsf{Adv}_{\mathsf{MM}}(\mathcal{A}) = \Pr\left[\mathsf{MM} : \neg\tilde{b} \mid \neg b\right] - \Pr\left[\mathsf{MM} : \neg\tilde{b} \mid b\right] \\ &= \frac{1}{K}\sum_{i=1}^{K} \Pr\left[\mathsf{MM} : \neg\tilde{b} \mid \neg b, k = i\right] - \Pr\left[\mathsf{MM} : \neg\tilde{b} \mid b, k = i\right] \\ &= \frac{1}{K}\sum_{i=1}^{K} \mathsf{Adv}_{\mathsf{MI}_i}(\mathcal{A}^i_{\mathsf{MI}}) + \Pr\left[G_0 : \neg\tilde{b} \mid \neg b\right] - \Pr\left[G_0 : \neg\tilde{b} \mid b\right] \end{aligned} \tag{2.4}$$

where the last equation follows from (2.3) and the fact that b and k are independent.

We reformulate G_0 as G_1 (see Algorithm 21). To see why both formulations are equivalent, note that conditioned on $b = 0$, in both games S and z are sampled from the same distribution chosen uniformly from $\boldsymbol{\mathcal{D}}$, while conditioned on $b = 1$, S and z are sampled each from one of two distributions sampled without replacement from $\boldsymbol{\mathcal{D}}$. Since b is independently sampled in the same way, both games result in the same joint distribution of θ, z, b, and therefore $\tilde{b}, b$:

$$\Pr\left[G_0 : \neg\tilde{b} \mid \neg b\right] = \Pr\left[G_1 : \neg\tilde{b} \mid \neg b\right] \tag{2.5}$$

$$\Pr\left[G_0 : \neg\tilde{b} \mid b\right] = \Pr\left[G_1 : \neg\tilde{b} \mid b\right] \tag{2.6}$$

Next, we show using a black-box reduction that distinguishing between the case when $b = 0$ and $b = 1$ in G_1 conditioned on $k = i, k' = j$ is as hard as guessing the challenge bit in the property inference experiment $\mathsf{DI}_{i,j}$ shown in Algorithm 24. To do this, we construct an adversary $\mathcal{A}^{i,j}_{\mathsf{DI}}$ that uses $\mathcal{A}$ as a black-box. $\mathcal{A}^{i,j}_{\mathsf{DI}}$ perfectly simulates the inputs to $\mathcal{A}$ in G_1 by forwarding its own inputs and freshly sampling z from $\mathcal{D}_i$.

$$\begin{aligned}\mathsf{Adv}_{\mathsf{DI}_{i,j}}(\mathcal{A}^{i,j}_{\mathsf{DI}}) &= \Pr\left[\mathsf{DI}_{i,j} : \neg\tilde{b} \mid \neg b\right] - \Pr\left[\mathsf{DI}_{i,j} : \neg\tilde{b} \mid b\right] \\ &= \Pr\left[G_1 : \neg\tilde{b} \,|\, \neg b, k = i, k' = j\right] - \Pr\left[G_1 : \neg\tilde{b} \,|\, b, k = i, k' = j\right]\end{aligned}$$

Putting this and (2.4)–(2.6) together, we obtain

$$\begin{aligned}\mathsf{Adv}_{\mathsf{MM}}(\mathcal{A}) &= \frac{1}{K}\sum_{i=1}^{K}\mathsf{Adv}_{\mathsf{MI}_i}(\mathcal{A}^i_{\mathsf{MI}}) + \frac{1}{K(K-1)}\sum_{i=1}^{K}\sum_{j=1, i\neq j}^{K}\mathsf{Adv}_{\mathsf{DI}_{i,j}}(\mathcal{A}^{i,j}_{\mathsf{DI}}) \\ &\leq \max_{i\in[K]}\mathsf{Adv}_{\mathsf{MI}_i}(\mathcal{A}^i_{\mathsf{MI}}) + \max_{i\neq j\in[K]}\mathsf{Adv}_{\mathsf{DI}_{i,j}}(\mathcal{A}^{i,j}_{\mathsf{DI}})\end{aligned}$$

□

Game 24: $\mathsf{DI}_{i,j}$

Input: $\mathcal{T}, \mathcal{D}_i, \mathcal{D}_j, n, \mathcal{A}$
$b \sim \{0, 1\}$
if $b = 0$ **then**
  $S \sim \mathcal{D}_i^n$
else
  $S \sim \mathcal{D}_j^n$
end
$\theta \leftarrow \mathcal{T}(S)$
$\tilde{b} \leftarrow \mathcal{A}^{i,j}_{\mathsf{DI}}(\mathcal{T}, \mathcal{D}_i, \mathcal{D}_j, n, \theta)$

Adversary 25: $\mathcal{A}^{i,j}_{\mathsf{DI}}$

Input: $\mathcal{T}, \mathcal{D}_i, \mathcal{D}_j, n, \theta$
$z \sim \mathcal{D}_i$
return $\mathcal{A}(\mathcal{T}, \boldsymbol{\mathcal{D}}, n, \theta, z)$

2.5 Related Work

Alternatives. We discuss below informal and formal alternatives to games to express privacy properties.

A key example of a *formal* property is Differential Privacy [74]. The definition of Differential privacy is relational, in that it compares the probability of events in two alternative worlds. DP abstracts from many details that are relevant for threat modelling, such as adversary capabilities, goals, and background knowledge, as well as the way datasets are created. This has led to disagreements in the literature about the consequences of differential privacy (see [316]).

A key example of an *informal* account of privacy properties is the *Opinion 05/2014 on Anonymization Techniques* [249] that complements the EU General Data Protection Regulation (GDPR) with practical recommendations for the use of anonymization techniques to meet the requirements set out by the regulator. In this influential document, the authors identify three privacy risks: *singling out*, *linkability*, and *inference*. They analyze the suitability of different anonymization techniques—including k-anonymity and DP—for mitigating these risks, but the discussion remains inconclusive due to the lack of precise definitions. Subsequent research [58] rigorously revisited the notion of singling out and suggested reconsidering the Opinion recommendations.

Game-based definitions address shortcomings of both alternatives: They make the threat model and assumptions explicit and precise, which helps disambiguate interpretations.

Game-based privacy proofs. Nissim et al. [234] construct a privacy game that reflects the requirements of the U.S. Family Educational Rights and Privacy Act (FERPA) for protecting privacy in releases of education records, and show in a proof structured as a sequence of games that DP is enough to satisfy these requirements. While constructing the game, they identify dimensions similar to our anatomy in §2.1.

Surveys and taxonomies on privacy. Several papers propose taxonomies of privacy attacks against machine learning systems [62, 185, 261]. Papernot et al. [247] focus on systematizing the possible attack surfaces of standard machine learning pipelines. Desfontaines and Pejó [67] systematically study variants and extensions of differential privacy. Before attacks against ML systems were demonstrated, Li et al. [178]

proposed a unifying framework for membership and differential privacy definitions mainly applicable to database systems.

2.6 Conclusion

This chapter provides an overview of different privacy games, their current and potential applications, and their limitations.

Given the wide range of privacy games available in the literature, it is natural to ask whether there is a *canonical* game that should be used instead of others. We believe this is not the case, i.e., no single game is the best choice in all circumstances, as subtle variations in threat scenarios can significantly impact privacy assessments. Instead, we recommend that users leverage the concepts presented in this chapter to tailor games according to their specific threat models.

Analyzing privacy risks in machine learning extends beyond the realm of researchers. Privacy managers and auditors also play a crucial role in assessing compliance with regulatory and contractual obligations. Currently, privacy managers rely on empirical privacy evaluations, formal guarantees of mechanisms like DP-SGD, and informal documents such as the *Opinion 05/2014* [249] from the European Commission's Article 29 Working Party. Incorporating privacy games can aid in this process by making the threat model and assumptions about dataset creation and training explicit, thus clarifying interpretations and abstracting application scenarios based on their privacy properties.

Privacy games are sequential probabilistic programs and may not directly address concurrent computations. As a result, their applicability to scenarios like federated learning (FL) is limited, given the complexity of modeling parallel interactions between different parties. The situation is similar for cryptographic games, where process calculi are used instead of games for modeling more complex multi-party interactions [32, 213]. It is an open question whether these calculi could also be used in the context of concurrent ML scenarios such as FL.

Chapter 3

Distribution Inference [1]

In this chapter, we formally setup distribution inference with a cryptographic-style inference game (§3.1). We then describe n_{leaked}, a metric that we propose for measuring leakage while accounting for the inherent "difficulty" of the inference task (§3.2), followed by an exploration of various inference attacks (§3.3). We present experiments on various datasets (§3.4), including our potent KL Divergence Attack (§3.5) and how performance fluctuates as certain implicit assumptions of a black-box setup are relaxed (§3.6). Finally, we explore possible defenses for distribution inference (§3.7).

3.1 Formalization

In a *distribution inference* attack, an adversary aims to infer some statistical property of the training dataset, such as the proportion of women in a dataset used to train a smile-detection model [17]. In the research literature, such attacks have previously been called *property inference* and *attribute inference* (confusingly, since this is also used to refer to a type of dataset inference where the adversary infers an unknown sensitive feature of records in the training dataset [92]), and various other terms.

The privacy threat posed by membership inference attacks is well recognized—if an adversary can infer the presence of a particular user record in a training dataset of diabetes patients, it would violate privacy laws limiting medical disclosure. Distribution inference attacks pose a less obvious threat but can also be dangerous. As one example, consider a financial organization that trains a loan scoring model on some of its historical data. An adversary may use a distribution inference attack to infer the proportion of the training data having a specific value for some protected attribute (e.g., race), which might be a sensitive property of

[1] This chapter is largely based on Anshuman Suri and David Evans, *Formalizing and Estimating Distribution Inference Risks*, in Proceedings on Privacy Enhancing Technologies (PETS), 2022 and Anshuman Suri, Yifu Lu, Yanjin Chen, David Evans, *Dissecting Distribution Inference*, in IEEE Conference on Secure and Trustworthy Machine Learning (SaTML), 2023. Code relevant to this chapter is available as a Python package at https://github.com/iamgroot42/propertyinference.

the training dataset. Distribution inference attacks can also pose a threat to distributed training: curious users wanting to learn sensitive information about training distributions of fellow participants, which are competitors in settings like cross-silo federated learning [148], thus leaking sensitive information. Other examples include inferring the sentiment of emails in a company from a spam classifier, or inferring the volume of transactions from fraud detection systems [199]. Inferring statistical properties of a training distribution can also be used to enhance membership inference attacks or to reveal that a model was trained on a biased dataset [376].

Previous works have used several different informal notions of property inference attacks (e.g., [100, 139, 371]), but there is no established general formal definition of distribution inference. We formalize distribution inference attacks based on a critical insight: the key difference between these attacks and other inference attacks is that the adversary's goal in the former is to learn about the training *distribution*, not about the specific training *dataset*. Dataset inference attacks, such as membership inference [279], attribute inference [92], and ownership-resolution [200] operate on the level of training records. Attacks like membership inference are directly connected to differential privacy which bounds the ability to distinguish neighboring datasets. By contrast, distribution inference attacks attempt to learn statistical properties of the underlying distribution from which the training dataset is sampled. Having a formal definition and a clear threat model can be useful in several ways—quantifying information leakage, assessing and comparing the practicality of different threat models, and drawing possible links between distribution inference risk and other distribution-level properties such as robustness and fairness.

Threat model. We model the adversary's knowledge of the underlying distribution through data sampled from that distribution. For complex, high-dimensional non-synthetic data, this is usually the only way to capture knowledge of a data distribution. While the threat model focuses on distributions, we use actual non-overlapping sampled data to empirically model those distributions.

A natural extension of our threat model incorporates a poisoning opportunity where the adversary can participate in the training process itself. Such an adversary may poison the training dataset by injecting adversarially crafted datapoints (explored by Mahloujifar et al. [199]) or control the training procedure itself to introduce some Trojan in the model. Recent works look at a similar scenario where the adversary participates in the learning process via a federated-learning setup, launching attribute-reconstruction attacks using epoch-averaged model gradients [194]. In this chapter, we only consider adversaries with no ability to observe or influence the training process. We explore adversaries that can poison the training process in §4.3.

Setup. Let $\mathcal{D} = (\mathcal{X}, \mathcal{Y})$ be a public distribution between data, $\mathcal{X}$, and its corresponding labels, $\mathcal{Y}$. We assume both the model trainer $\mathcal{T}$ and adversary $\mathcal{A}$ have access to $\mathcal{D}$. Both parties also have access to

two functions, $\mathcal{G}_0$ and $\mathcal{G}_1$, that transform distributions. Inferring properties of the distribution can reveal sensitive information in many scenarios, which can be captured by suitable choices of $\mathcal{G}_0$ and $\mathcal{G}_1$. Using such functions along with the underlying distribution $\mathcal{D}$ makes the setup less restrictive than defining two arbitrary distributions—since the functions $\mathcal{G}_0$, $\mathcal{G}_1$, and $\mathcal{D}$ are considered public knowledge, anyone can recreate these distributions. Using functions on the same distribution $\mathcal{D}$ emphasizes the fact that these two distributions stem from the same underlying distribution $\mathcal{D}$, enabling the definition to capture a wide class of possible attack goals and scenarios by selecting appropriate functions for $\mathcal{G}_0$ and $\mathcal{G}_1$.

To illustrate these definitions, we present a concrete example inspired by Mahloujifar et al. [199]. Let $\mathcal{D}$ be a distribution of emails with labels for spam/ham. $\mathcal{G}_0$ is applied over $\mathcal{D}$ to yield a modified distribution $\mathcal{G}_0(\mathcal{D})$ with 0.8 probability of sampling an email that has negative sentiment i.e., a dataset sampled uniformly at random from this distribution would have approximately 80% of the emails in it with negative sentiment. Similarly, $\mathcal{G}_1(\mathcal{D})$ could be another distribution with this probability as 0.5 (equally likely to be positive or negative). The adversary thus wants to know if the training distribution is biased, which, if inferred near the financial quarter, can be used to predict if the company is performing below expectations. Alternatively, an ambitious adversary could even consider directly inferring [109] this proportion (which was not considered in Mahloujifar et al. [199], but we explore in Theorem 3.2.3 and our regression experiments on other datasets).

We propose a general and straightforward experiment to formalize property inference attacks, inspired by Yeom *et al.*'s cryptographic game definition of membership inference [355]. In our cryptographic game definition, $\mathcal{T}$ picks one of the $\mathcal{G}_{\{0,1\}}$ distribution transformers at random and samples a dataset S from the resulting distribution. Given access to a model M trained on S, the adversary aims to infer which of the two distribution mappers was used:

Trainer $\mathcal{T}$		**Adversary $\mathcal{A}$**
1 :	$b \leftarrow\!\!\$\ \{0,1\}$	
2 :	$S \sim \mathcal{G}_b(\mathcal{D})$	
3 :	$M \xleftarrow{train} S$	
4 :	$\xrightarrow{\quad M \quad}$	
5 :		$\hat{b} = \mathcal{H}(M)$

We assume the adversary has no control over the training process (Step 3). For cases where the adversary has access to the training data, it can trivially infer desired properties by inspecting it. Our definition could be adapted to other scenarios such as federated learning [180] by providing additional information to the adversary or allowing the adversary to have some control over S, but we do not consider such settings in this chapter.

If $\mathcal{A}$ can successfully predict b via $\hat{b}$, then it can determine which of the training distributions was used. The advantage of the adversary $\mathcal{A}$ using algorithm $\mathcal{H}$ is defined as:

$$\text{Adv}_{\mathcal{H}} = \left|\Pr\left[\hat{b} \;\middle|\; b\right] - \Pr\left[\hat{b} \;\middle|\; \neg\, b\right]\right|.$$

This advantage is negligible when the adversary does no better than random guessing. We do not assume the adversary knows how training data is collected, or has any access to it: just that it knows the underlying common distribution $\mathcal{D}$, and has a goal of distinguishing between sub-distributions $\mathcal{G}_0(\mathcal{D})$, $\mathcal{G}_1(\mathcal{D})$ of that distribution. The adversary does not need to know the actual training distribution—indeed, learning about this is the goal of the attack. They just need to have hypotheses worth testing, and thus $\mathcal{G}_0$ and $\mathcal{G}_1$ are defined by the adversary based on what they want to test.

Limitations of the Definition. This definition is simple and general, but does not capture all kinds of distribution inference attacks. It assumes a setting where the adversary attempts to distinguish between two particular distributions, both of which are defined by the adversary. Multiple experiments could extend the definition to a set of possible distributions, but the definition does not directly capture the regression attacks we demonstrate where the adversary is estimating what proportion of a training dataset has a given property directly. Our definition also assumes the adversary has prior knowledge of the two sub-distributions to distinguish. Some knowledge of the statistical property the adversary wants to learn about the victim's training distribution is inherent in the nature of a distribution inference attack, but this may not always be in the form of knowledge of possible distributions. In our experiments, we model an adversaries knowledge of the underlying distribution through a sampled, non-overlapping dataset, which the adversary may then use to construct approximations of different sub-distributions.

Applying the Definition. Seminal works on property inference [16, 95] involve a model trained either on the original dataset or a version modified to be biased towards some chosen attribute. Our definition can be used to describe these attacks by setting $\mathcal{G}_0$ to the identity function (so $\mathcal{G}_0(\mathcal{D})$ is original distribution $\mathcal{D}$) and $\mathcal{G}_1$ to a filter that adjusts the distribution to have a specified ratio over the desired attribute. With respect to a binary property function, $f : \mathcal{X} \to \{0, 1\}$, $\mathcal{D}$ can be characterized using a generative probability density function:

$$\rho_{\mathcal{D}}(\mathbf{x}) = \sum_{c \in \{0,1\}} p(c) \cdot p(\mathbf{x} \mid c), \tag{3.1}$$

where $p(c)$ is a multinomial distribution representing the probabilities over the desired (binary) property function f and its possible values c, and $p(\mathbf{x} \mid c)$ is the generative conditional probability density function.

Then, $\mathcal{G}_1(\mathcal{D})$ can be expressed using the following probability density function, with a prior $\hat{p}$:

$$\rho_{G_1(\mathcal{D})}(\mathbf{x}) = \sum_{c \in \{0,1\}} \hat{p}(c) \cdot p(\mathbf{x} \mid c),\ \hat{p}(1) = \alpha,\ \hat{p}(0) = 1 - \alpha, \tag{3.2}$$

where α is the probability of a randomly sampled point satisfying the property function f. Thus, a uniformly randomly sampled dataset from $\mathcal{G}_1(\mathcal{D})$ would have an expected ratio of α of its members satisfying f. Additionally, we can modify $\mathcal{G}_0$ with a similarly adjusted prior, enabling the adversary to distinguish between any two arbitrary ratios [371]. In fact, our definition subsumes the one proposed by Mahloujifar et al. [199] for the case of ratios over Boolean functions via the following instantiation:

$$\begin{aligned} D_-,\ D_+ &= \mathcal{G}_0(\mathcal{D}),\ \mathcal{G}_1(\mathcal{D}) \\ t_0,\ t_1 &= \alpha_0,\ \alpha_1, \end{aligned} \tag{3.3}$$

along with setting $c = f(x)$ in Equation 3.2.

Our definition, however, is not limited to describing proportional properties. For example, it can also be used to define the distributions over degrees for graph-based datasets, and infer properties of the underlying degree distributions. For this case, we represent graphs as samples from degree distributions: data with different degrees is sampled and then combined together in one graph. Since the adversary only cares about properties pertaining to the degrees of nodes (and not their attributes or other characteristics), it is safe to represent these samples purely in terms of degree distributions. This can then be used to target properties such as the mean-node degree of graphs, as we show in §3.4.3.

3.2 Measuring leakage

Assessing the power of an attack is important for understanding it scientifically, and can also be of practical importance for both the victim and the adversary. Consider the most explored case in the literature—ratios of members satisfying a Boolean function. Intuition suggests that distributions with more different ratios (e.g., 0.2 and 0.9) would be easier to distinguish than more similar ones (e.g., 0.2 and 0.3), and most previous distributions inference results have focused on highly disparate distributions (often only showing meaningful distinguishing power when one of the ratios is at a 0.0 or 1.0 extreme).

Our framework enables us to quantify the amount of leakage observed in an attack by relating what an adversary is able to learn from a disclosed model to what they would learn from directly sampling examples from the training distribution. As a setup, we provide the following lemma shows that gives an upper bound on the distinguishing accuracy (which we define as the probability of an adversary correctly inferring the

underlying training distribution of a model) of any statistical test distinguishing between two distributions that differ in the proportion of records satisfying some Boolean property, using n samples:

Lemma 3.2.1. *Given two Boolean-property proportional distributions $\mathcal{G}_0(\mathcal{D})$, $\mathcal{G}_1(\mathcal{D})$ with proportion values α_0, α_1 derived from the same underlying distribution $\mathcal{D}$, the distinguishing accuracy between models trained on datasets of size n from these distributions is at most*

$$\frac{1}{2} + \frac{\min\left\{\sqrt{1-\left(\frac{\min(\alpha_0,\alpha_1)}{\max(\alpha_0,\alpha_1)}\right)^n}, \sqrt{1-\left(\frac{1-\max(\alpha_0,\alpha_1)}{1-\min(\alpha_0,\alpha_1)}\right)^n}\right\}}{2}.$$

Proof. Assume the adversary can fully recover a dataset S (of size n) from some model M trained on it. Assume $\psi(\cdot)$ is an estimator for testing the hypothesis i.e., $\psi(S) = b_{\in\{0,1\}}$ means that S comes from $\mathcal{G}_b(\mathcal{D})$. Assuming an equal likelihood of the chosen dataset S being from either distributions, we have:

$$\begin{aligned}\text{Error} &= \frac{1}{2}\left(\Pr_{S\leftarrow\mathcal{G}_0(\mathcal{D})^n}\left[\psi(S)=1\right] + \Pr_{S\leftarrow\mathcal{G}_1(\mathcal{D})^n}\left[\psi(S)=0\right]\right)\\ &= \frac{1}{2}(\text{Type I Error} + \text{Type II Error}).\end{aligned}$$

Combining with the result from [207]:

$$\text{Error} \geq \frac{1}{2} - \frac{1}{2}\delta(\mathcal{G}_0(\mathcal{D})^n, \mathcal{G}_1(\mathcal{D})^n) \tag{3.4}$$

$$\Rightarrow \text{Accuracy} \leq \frac{1}{2} + \frac{1}{2}\delta(\mathcal{G}_0(\mathcal{D})^n, \mathcal{G}_1(\mathcal{D})^n), \tag{3.5}$$

where $\delta()$ is the total variation distance between two probability measures, and $\mathcal{G}_{b\in\{0,1\}}(\mathcal{D})^n$ refers to the distribution of n samples from $\mathcal{G}_{b\in\{0,1\}}(\mathcal{D})$.. Thus, the maximum accuracy while differentiating between datasets sampled from either distribution is bounded by the total variation distance between them. Let $\rho_b(x)$ be the generative probability density function for some sample x drawn from $\mathcal{G}_b(\mathcal{D})$, for $b \in \{0,1\}$. This density function can then be broken down into a multinomial distribution and priors as:

$$\rho_b(x) = (1-\alpha_b)p_b(\mathbf{x}|0) + \alpha_b p_b(\mathbf{x}|1), \tag{3.6}$$

where α_b is the prior for $p(1)$ corresponding to $\mathcal{G}_b(\mathcal{D})$, and $\rho_b(x|1)$ is the associated conditional generative probability density function. Note that $p_0(\mathbf{x}|0) = p_1(\mathbf{x}|0)$ and $p_0(\mathbf{x}|1) = p_1(\mathbf{x}|1)$, since they both come from the underlying distribution $\mathcal{D}$. Without loss of generality, let $\alpha_0 > \alpha_1$ (we omit the case of same ratios, since

that is trivially indistinguishable). Then:

$$\begin{aligned}\alpha_1\rho_0(x) - \alpha_0\rho_1(x) &= \alpha_1((1-\alpha_0)p_0(\mathbf{x}|0) + \alpha_0 p_0(\mathbf{x}|1)) - \alpha_0((1-\alpha_1)p_1(\mathbf{x}|0) + \alpha_1 p_1(\mathbf{x}|1)) \\ &= \alpha_1 p_0(\mathbf{x}|0) - \alpha_1\alpha_0 p_0(\mathbf{x}|0) + \alpha_0\alpha_1 p_0(\mathbf{x}|1) - (\alpha_0 p_1(\mathbf{x}|0) - \alpha_0\alpha_1 p_1(\mathbf{x}|0) + \alpha_1\alpha_0 p_1(\mathbf{x}|1)) \\ &= (\alpha_1 - \alpha_0)p_0(\mathbf{x}|0) \leq 0 \\ \Rightarrow \frac{\rho_0(x)}{\rho_1(x)} &\leq \frac{\alpha_0}{\alpha_1} \end{aligned} \tag{3.7}$$

Using this inequality, the relative entropy (KL divergence) from $\mathcal{G}_1(\mathcal{D})$ to $\mathcal{G}_0(\mathcal{D})$ can be written as:

$$D_{KL}(\mathcal{G}_0(\mathcal{D}) \parallel \mathcal{G}_1(\mathcal{D})) = \int \rho_0(x) \log\left(\frac{\rho_0(x)}{\rho_1(x)}\right) dx \leq \int \rho_0(x) \log\left(\frac{\alpha_0}{\alpha_1}\right) dx \tag{3.8}$$

$$= \log\left(\frac{\alpha_0}{\alpha_1}\right) \int \rho_0(x) dx = \log\left(\frac{\alpha_0}{\alpha_1}\right) \tag{3.9}$$

Since the function f is binary, a prior of α_b for $p(f(x) = 1)$ implies a prior of $(1 - \alpha_b)$ for $p(f(x) = 0)$. Utilizing this symmetry, we can similarly upper-bound $D_{KL}(\mathcal{G}_1(\mathcal{D}) \parallel \mathcal{G}_0(\mathcal{D}))$ with $\log\left(\frac{1-\alpha_1}{1-\alpha_0}\right)$. Removing the $\alpha_0 \geq \alpha_1$ assumption and replacing with the max/min of these two appropriately, we get:

$$\begin{aligned} D_{KL}(\mathcal{G}_0(\mathcal{D}) \parallel \mathcal{G}_1(\mathcal{D})) &\leq \log\left(\frac{\max(\alpha_0, \alpha_1)}{\min(\alpha_0, \alpha_1)}\right) \\ D_{KL}(\mathcal{G}_1(\mathcal{D}) \parallel \mathcal{G}_0(\mathcal{D})) &\leq \log\left(\frac{1 - \min(\alpha_0, \alpha_1)}{1 - \max(\alpha_0, \alpha_1)}\right) \end{aligned} \tag{3.10}$$

From [133], we know that:

$$D_{KL}(\mathcal{G}_0(D)^n \parallel \mathcal{G}_1(D)^n) = nD_{KL}(\mathcal{G}_0(D) \parallel \mathcal{G}_1(D)) \tag{3.11}$$

Thus, when using a dataset S of size $|S| = n$, the equivalent KL-divergence can be bounded by:

$$\begin{aligned} D_{KL}(\mathcal{G}_0(\mathcal{D})^n \| \mathcal{G}_1(\mathcal{D})^n) &\leq n \log\left(\frac{\max(\alpha_0, \alpha_1)}{\min(\alpha_0, \alpha_1)}\right) \\ D_{KL}(\mathcal{G}_1(\mathcal{D})^n \| \mathcal{G}_0(\mathcal{D})^n) &\leq n \log\left(\frac{1 - \min(\alpha_0, \alpha_1)}{1 - \max(\alpha_0, \alpha_1)}\right) \end{aligned} \tag{3.12}$$

Using the relation between total variation distance and KL-divergence [317], we know:

$$\delta(\mathcal{G}_0(\mathcal{D})^n, \mathcal{G}_1(\mathcal{D})^n) \leq \sqrt{1 - e^{-D_{KL}(\mathcal{G}_0(\mathcal{D})^n \parallel \mathcal{G}_1(\mathcal{D})^n)}} \tag{3.13}$$

$$= \sqrt{1 - e^{-nlog\left(\frac{\max(\alpha_0, \alpha_1)}{\min(\alpha_0, \alpha_1)}\right)}} = \sqrt{1 - \left(\frac{\min(\alpha_0, \alpha_1)}{\max(\alpha_0, \alpha_1)}\right)^n} \tag{3.14}$$

Similarly, using $D_{KL}(\mathcal{G}_1(\mathcal{D}) \parallel \mathcal{G}_0(\mathcal{D}))$ in the inequality above we get:

$$\delta(\mathcal{G}_0(\mathcal{D})^n, \mathcal{G}_1(\mathcal{D})^n) \leq \sqrt{1 - e^{-D_{KL}(\mathcal{G}_1(\mathcal{D})^n \parallel \mathcal{G}_0(\mathcal{D})^n)}} \tag{3.15}$$

$$= \sqrt{1 - e^{-nlog\left(\frac{1-\min(\alpha_0,\alpha_1)}{1-\max(\alpha_0,\alpha_1)}\right)}} = \sqrt{1 - \left(\frac{1 - \max(\alpha_0, \alpha_1)}{1 - \min(\alpha_0, \alpha_1)}\right)^n} \tag{3.16}$$

Since the function $f()$ is boolean, an adversary can choose to focus on a property value of 0 or 1, and infer the ratio of one using the other. Thus, any two ratios (α_0, α_1) can be alternatively seen as $(1 - \alpha_0, 1 - \alpha_1)$. Combining the two inequalities above and plugging them back in (3.5), we get:

$$\text{Accuracy} \leq \frac{1}{2} + \frac{\min\left\{\sqrt{1 - \left(\frac{\min(\alpha_0,\alpha_1)}{\max(\alpha_0,\alpha_1)}\right)^n}, \sqrt{1 - \left(\frac{1-\max(\alpha_0,\alpha_1)}{1-\min(\alpha_0,\alpha_1)}\right)^n}\right\}}{2} \tag{3.17}$$

Note that the proof of this bound hinges on both the distributions originating from the same underlying distribution $\mathcal{D}$, which is why we use $\mathcal{G}_0(\mathcal{D})$, $\mathcal{G}_1(\mathcal{D})$ instead of some arbitrarily defined distributions $\mathcal{D}_0$, $\mathcal{D}_1$. □

First, we consider the most powerful possible adversary as one that can perfectly reconstruct training records from the model. The most that could be leaked to such an adversary is a perfect reconstruction of all the training records. Of course, we do not expect an adversary to reconstruct the training dataset fully, and an adversary can succeed in a high confidence distribution inference attack without being able to reconstruct any training records perfectly. Such a perspective is useful, though, for quantifying the power of an attack in a way that allows comparisons between attacks distinguishing distributions with different levels of variation. For some observed performance ω via an attack, we can compute the corresponding value of n that would give an upper bound on accuracy as ω. This value of n, which we term as n_{leaked}, thus quantifies the size of the dataset "leaked" by the attack. In other words, it is equivalent to the adversary being able to draw n_{leaked} samples from the training distribution and executing an optimal distinguishing statistical test. The following theorem shows how to compute n_{leaked} for an observed attack for the kind of distributions described above.

Theorem 3.2.2. *Given two Boolean-property proportional distributions $\mathcal{G}_0(\mathcal{D})$, $\mathcal{G}_1(\mathcal{D})$ with proportion values α_0, α_1 derived from the same underlying distribution $\mathcal{D}$, and distinguishing accuracy ω using some attack,*

$$n_{leaked} = \frac{\log(4\omega(1-\omega))}{\log(\max\left(\frac{\min(\alpha_0,\alpha_1)}{\max(\alpha_0,\alpha_1)}, \frac{1-\max(\alpha_0,\alpha_1)}{1-\min(\alpha_0,\alpha_1)}\right))}.$$

Proof. Consider Lemma 3.2.1: let ω be the observed distinguishing accuracy for some attack. Let n_{leaked} be the effective value of n corresponding to the given attack, *i.e.* Equating it with the best distinguishing

accuracy for this value of n, we can compute n_{leaked}. If $\frac{\min(\alpha_0,\alpha_1)}{\max(\alpha_0,\alpha_1)} \geq \frac{1-\max(\alpha_0,\alpha_1)}{1-\min(\alpha_0,\alpha_1)}$:

$$2\omega - 1 = \sqrt{1 - \left(\frac{\min(\alpha_0, \alpha_1)}{\max(\alpha_0, \alpha_1)}\right)^{n_{leaked}}} \tag{3.18}$$

$$\log(1 - (2\omega - 1)^2) = n_{leaked}\left(\log\left(\frac{\min(\alpha_0, \alpha_1)}{\max(\alpha_0, \alpha_1)}\right)\right) \tag{3.19}$$

$$n_{leaked} = \frac{\log(4\omega(1-\omega))}{\log\left(\frac{\min(\alpha_0,\alpha_1)}{\max(\alpha_0,\alpha_1)}\right)} \tag{3.20}$$

Similarly, for the case of $\frac{\min(\alpha_0,\alpha_1)}{\max(\alpha_0,\alpha_1)} < \frac{1-\max(\alpha_0,\alpha_1)}{1-\min(\alpha_0,\alpha_1)}$, we get:

$$n_{leaked} = \frac{\log(4\omega(1-\omega))}{\log\left(\frac{1-\max(\alpha_0,\alpha_1)}{1-\min(\alpha_0,\alpha_1)}\right)} \tag{3.21}$$

Combining these two cases, we get:

$$n_{leaked} = \frac{\log(4\omega(1-\omega))}{\log\left(\max\left(\frac{\min(\alpha_0,\alpha_1)}{\max(\alpha_0,\alpha_1)}, \frac{1-\max(\alpha_0,\alpha_1)}{1-\min(\alpha_0,\alpha_1)}\right)\right)} \tag{3.22}$$

□

A high value of n_{leaked} means the adversary is learning a lot about the underlying distribution, just using the given model. It helps put the attack's strength in perspective, given how similar the two distributions are. For instance, distinguishing between $\alpha_0 = 0.5$ and $\alpha_1 = 1.0$ with distinguishing accuracy $\omega = 0.95$ corresponds to $n_{\text{leaked}} \approx 3$, whereas distinguishing between $\alpha_0 = 0.5$ and $\alpha_1 = 0.52$ with the same accuracy would correspond to $n_{\text{leaked}} \approx 42$. This aligns with intuition: the latter distribution is more similar and thus, should be "harder" for an attack to achieve the same kind of performance, and this notion is exactly what n_{leaked} aims to capture.

Note that this analysis is based on modeling an "optimal attack" where the adversary is able to directly sample records from the training distribution. This is just for deriving an expression for n_{leaked}, and not meant to assume any such attack. The expression above (and subsequent expressions for n_{leaked}) is a useful measure of any attack's effectiveness that quantifies the leakage observed in the attack by relating it to the amount of information that would be leaked in an "optimal attack" where the adversary is just sampling training distribution records directly rather than inferring properties from a revealed model.

Regression over α. The case of distinguishing between ratios can be further extended to consider an adversary that wishes to directly predict the proportion value α for a given distribution. The following theorem shows how to compute n_{leaked} for an observed attack with square error ω.

Theorem 3.2.3. *Given a Boolean-property proportional distribution with proportion value α, and square error ω observed using some attack,*

$$n_{leaked} = \frac{\alpha(1-\alpha)}{\omega}.$$

Proof. Assume the adversary can fully recover a dataset S (of size N) from some model M trained on it. Let n_1 be the number of entries in S that are 1, and n_0 0 such that $n_0 + n_1 = N$. Then, the conditional probability density function of the underlying distribution $\mathcal{D}$ having $\Pr[1] = z$ (assume all z are equally likely), given the observed dataset S, can be written using the continuous Bayes' rule as:

$$\Pr[z \mid S] = \frac{\Pr[S \mid z]\Pr[z]}{\int_0^1 \Pr[S \mid x]\Pr[x]dx} = \frac{\binom{N}{n_1} z^{n_1}(1-z)^{n_0}}{\int_0^1 \binom{N}{n_1} x^{n_1}(1-x)^{n_0} dx} = z^{n_1}(1-z)^{n_0}\frac{\Gamma(n_0+n_1+2)}{\Gamma(n_0+1)\Gamma(n_1+1)} \tag{3.23}$$

The above conditional probability is maximized when $z = \frac{n_1}{N}$ i.e., the guessed ratio is the ratio observed in the given sample S. Then, we can compute the expected square error over all possible datasets of size N, given that the distribution they were sampled from has a proportion value α:

$$\mathbb{E}\big[(z-\alpha)^2\big] = \mathbb{E}\big[z^2\big] + \alpha^2 - 2\alpha\,\mathbb{E}[z] \tag{3.24}$$

We can then compute $\mathbb{E}[z]$ as:

$$\sum_{n_1=0}^{N} \frac{n_1}{N}\binom{N}{n_1}(\alpha)^{n_1}(1-\alpha)^{N-n_1} = \frac{1}{N}\sum_{n_1=0}^{N} n_1\binom{N}{n_1}(\alpha)^{n_1}(1-\alpha)^{N-n_1} = \alpha \tag{3.25}$$

Similarly, $\mathbb{E}\big[z^2\big]$ can be computed as:

$$\sum_{n_1=0}^{N} (\frac{n_1}{N})^2\binom{N}{n_1}(\alpha)^{n_1}(1-\alpha)^{N-n_1} = \frac{1}{N^2}\sum_{n_1=0}^{N} n_1^2\binom{N}{n_1}(\alpha)^{n_1}(1-\alpha)^{N-n_1} = \alpha^2 + \frac{\alpha(1-\alpha)}{N} \tag{3.26}$$

Plugging (3.25) and (3.26) in (3.24), we get:

$$\mathbb{E}\big[(z-\alpha)^2\big] = \frac{\alpha(1-\alpha)}{N} \tag{3.27}$$

Thus, for an observed square error ω for some attack, n_{leaked} can be computed as:

$$n_{leaked} = \frac{\alpha(1-\alpha)}{\omega} \tag{3.28}$$

□

This result extends our notion of n_{leaked} to adversaries that directly infer the underlying ratio α, a more realistic adversary goal that we also explore in our experiments.

Graphs as Distributions of Natural Numbers. Similar to the ratio case, our framework enables us to compute n_{leaked} when working with distributions of natural numbers. For the purpose of distinguishing between graph distributions, this notion of 'distribution of numbers' can be extended to graphs by studying their degree distributions. As a setup, we provide the following lemma, that gives an upper bound on the distinguishing accuracy of any statistical test distinguishing between two distributions following Zipf's law, using n samples:

Lemma 3.2.4. *Given two distributions of natural numbers, $\mathcal{G}_0(\mathcal{D})$ and $\mathcal{G}_1(\mathcal{D})$ that follow Zipf's law, with N_0 and N_1 elements (without loss of generality assume $N_0 \leq N_1$) and parameters s_0, s_1 respectively, the distinguishing accuracy between models trained on graphs with n nodes from these distributions is at most:*

$$\frac{1}{2} + \frac{\sqrt{1 - \left(\frac{H_{N_0,s_0}}{H_{N_1,s_1}} N_0^{(s_0 - s_1)\mathbb{I}[s_1 > s_0]}\right)^n}}{2}$$

, where $H_{n,s} = \sum_{k=1}^{N} k^{-s}$ is the n^{th} generalized Harmonic number of order s.

Proof. We assume that both degree distributions follow Zipf's law, such that the PDF for either of $\mathcal{G}_0$ or $\mathcal{G}_1$ can be written as

$$\rho_b(x) = \frac{x^{-s_b}}{H_{N_b,s_b}} \tag{3.29}$$

where $H_{N,s}$ is the N^{th} generalized harmonic number of order s, N_b corresponds to the maximum degree (with nonzero probability) $\mathcal{G}_B(\mathcal{D})$, and s_b determines the spread of the distribution. Since the inequality between the total variation distance and accuracy is independent of the underlying distributions, (3.5) applies in this case too.

Without loss of generality, let $N_1 \geq N_0$. In that case, the relative entropy from $\mathcal{G}_0$ to $\mathcal{G}_1$ would be undefined, since support$(\mathcal{G}_1) \not\subseteq$ support$(\mathcal{G}_0)$. Computing the relative entropy from $\mathcal{G}_1$ to $\mathcal{G}_0$, we get:

$$D_{KL}(\mathcal{G}_0(\mathcal{D}) \parallel \mathcal{G}_1(\mathcal{D})) = \sum_{n=1}^{N_1} \rho_0(x) \log\left(\frac{\rho_0(x)}{\rho_1(x)}\right) \tag{3.30}$$

Since $\rho_0(x)$ only applies until N_0, it evaluates to 0 for $n > N_0$. Substituting:

$$\sum_{n=1}^{N_0} \rho_0(x) \log\left(\frac{\rho_0(x)}{\rho_1(x)}\right) + \sum_{n=N_0+1}^{N_1} 0 \cdot \log\left(\frac{0}{\rho_1(x)}\right) \tag{3.31}$$

$$= \frac{1}{H_{N_0,s_0}} \left(\sum_{n=1}^{N_0} x^{-s_0} \left(\log\left(\frac{H_{N_1,s_1}}{H_{N_0,s_0}}\right) + (s_1 - s_0)\log(x)\right)\right) \tag{3.32}$$

$$= \frac{1}{H_{N_0,s_0}} \left(H_{N_0,s_0} \log\left(\frac{H_{N_1,s_1}}{H_{N_0,s_0}}\right) + (s_1 - s_0) \sum_{n=1}^{N_0} x^{-s_0} \log(x)\right) \tag{3.33}$$

$$= \log\left(\frac{H_{N_1,s_1}}{H_{N_0,s_0}}\right) + \frac{s_1 - s_0}{H_{N_0,s_0}} \sum_{n=1}^{N_0} x^{-s_0} \log(x) \tag{3.34}$$

If $s_1 > s_0$:

$$D_{KL}(\mathcal{G}_0(\mathcal{D}) \parallel \mathcal{G}_1(\mathcal{D})) \leq \log\left(\frac{H_{N_1,s_1}}{H_{N_0,s_0}}\right) + \frac{s_1 - s_0}{H_{N_0,s_0}} \sum_{n=1}^{N_0} x^{-s_0} \log(N_0) \tag{3.35}$$

$$= \log\left(\frac{H_{N_1,s_1}}{H_{N_0,s_0}}\right) + (s_1 - s_0)\log(N_0) \tag{3.36}$$

If $s_1 \leq s_0$:

$$D_{KL}(\mathcal{G}_0(\mathcal{D}) \parallel \mathcal{G}_1(\mathcal{D})) \leq \log\left(\frac{H_{N_1,s_1}}{H_{N_0,s_0}}\right) \tag{3.37}$$

Computing the total variation distance for n samples according to (3.13) for both cases, we get:

$$\delta(\mathcal{G}_0(\mathcal{D})^n, \mathcal{G}_1(\mathcal{D})^n) \leq \begin{cases} \sqrt{1 - \left(\frac{H_{N_0,s_0}}{H_{N_1,s_1}} N_0^{s_0 - s_1}\right)^n}, & \text{if } s_1 > s_0 \\ \sqrt{1 - \left(\frac{H_{N_0,s_0}}{H_{N_1,s_1}}\right)^n} & \text{otherwise} \end{cases}$$

Plugging this in (3.5) to get an upper bound on the distinguishing accuracy.

$$\text{Accuracy} \leq \frac{1}{2} + \frac{\sqrt{1 - \left(\frac{H_{N_0,s_0}}{H_{N_1,s_1}} N_0^{(s_0 - s_1)\mathbb{I}[s_1 > s_0]}\right)^n}}{2} \tag{3.38}$$

□

Together, these two parameters are related to the expected mean of the distribution α_b (mean node-degree, in the case of degree distributions) as:

$$\alpha_b = \frac{H_{N_b,s_b-1}}{H_{N_b,s_b}} \tag{3.39}$$

Similar to the case of different Boolean-property ratios distributions, we can compute n_{leaked} for a given attack and its observed performance:

Theorem 3.2.5. *Given two distributions of natural numbers distributions $\mathcal{G}_0(\mathcal{D})$, $\mathcal{G}_1(\mathcal{D})$ that follow Zipf's law, with N_0, N_1 elements (without loss of generality assume $N_0 \leq N_1$) and parameters s_0, s_1 respectively, and observed distinguishing accuracy ω,*

$$n_{leaked} = \frac{\log(4\omega(1-\omega))}{\log\left(\frac{H_{N_0,s_0}}{H_{N_1,s_1}}\right) + (s_0 - s_1)\mathbb{I}[s_1 > s_0]\log(N_0)}.$$

Proof. Consider Lemma 3.2.4: let ω be the observed distinguishing accuracy for some attack. Let n_{leaked} be the effective value of n corresponding to the given attack, *i.e.* Equating it with the best distinguishing accuracy for this value of n, we get:

$$\omega = \frac{1}{2} + \frac{\sqrt{1 - \left(\frac{H_{N_0,s_0}}{H_{N_1,s_1}} N_0^{(s_0-s_1)\mathbb{I}[s_1>s_0]}\right)^{n_{leaked}}}}{2} \tag{3.40}$$

$$\log(1-(2\omega-1)^2) = n_{leaked} \cdot \left(\log\left(\frac{H_{N_0,s_0}}{H_{N_1,s_1}}\right) + (s_0 - s_1)\mathbb{I}[s_1 > s_0]\log(N_0)\right) \tag{3.41}$$

$$n_{leaked} = \frac{\log(4\omega(1-\omega))}{\log\left(\frac{H_{N_0,s_0}}{H_{N_1,s_1}}\right) + (s_0 - s_1)\mathbb{I}[s_1 > s_0]\log(N_0)} \tag{3.42}$$

□

Compared to the ratio distinguishing attacks, attacks on the ogbn-arxiv dataset are much more successful: reaching near-perfect distinguishing accuracies as well as the highest n_{leaked} numbers (Table 3.2).

3.3 Attacks

We now begin with the description of attacks that an adversary can use to infer the training distribution of a target model. We propose three black-box attacks, Loss Test, Threshold Test, and KL Divergence Attack, and two white-box attacks: an extension of an existing attack (Permutation Invariant Networks) for convolutional networks, and a new attack (Affinity Graph Attack). These attacks do not assume anything about the underlying distributions other than the availability of the underlying distribution and knowledge of the public $\mathcal{G}_0$ and $\mathcal{G}_1$ transformers.

3.3.1 Black-Box Attacks

Black-box attacks assume the adversary has the ability to submit inputs to the trained model and observe the response but does not have direct access to the model (parameters). In addition, the adversary has access to some representative data from some distribution $\mathcal{D}$, and seeks to infer which of the transformations, $\mathcal{G}_0$ or

$\mathcal{G}_1$, corresponds to the victim's training distribution. Using knowledge of $\mathcal{D}$ and the transformation functions $\mathcal{G}_0$ and $\mathcal{G}_1$, the adversary is able to train shadow models locally. Knowledge of the candidate distributions is necessary to be able to distinguish between them, and it is reasonable to assume an adversary with enough computational resources to train models locally. Most research assumes that the victim and adversary use the same model architecture (e.g., [214, 371]), and that the adversary has access to model prediction confidence vectors (e.g., [348, 371]). We evaluate the impact of relaxing some of these assumptions in §3.6.

3.3.1.1 Loss Test

A simple algorithm $\mathcal{H}$ is to test the loss of the model on datasets from the two candidate distributions, and conclude that the training distribution is closest to whichever test dataset the model performs better on. For data samples $S_{b\in\{0,1\}} \sim \mathcal{G}_b(\mathcal{D})$:

$$\hat{b} = \mathbb{I}[\ell(M, S_0) > \ell(M, S_1)], \tag{3.43}$$

where $\ell(M, S)$ is the loss of model M on some data S, and $\mathbb{I}$ is the indicator function. Intuitively, a model would have lower loss on data sampled from the training distribution, compared to another distribution. This method does not require the adversary to train models, but only to have access to suitable test distributions and the ability to submit samples to the target model. The data held by the adversary here is not overlapping with the data used by the victim to train its models, ruling out any potential for leakage via shared data. Although we use loss in Equation (3.43), the adversary can use any metric (like accuracy).

3.3.1.2 Threshold Test

The Loss Test assumption may not hold for some pairs of distributions if one distribution is inherently easier to classify than the other. To account for this, we consider an attack where the adversary trains and uses a small (balanced) sample of models from each distribution to identify which of S_0 or S_1 maximizes the performance gap between its models.

$$\begin{aligned} \gamma_{c\in\{0,1\}} &= \sum_i \ell(M_0^i, S_c) - \sum_i \ell(M_1^i, S_c) \\ k &= \mathbb{I}[|\gamma_0| < |\gamma_1|], \end{aligned} \tag{3.44}$$

where $M_{\{0,1\}}^i$ is trained on a dataset sampled from $\mathcal{G}_{\{0,1\}}(\mathcal{D})$ respectively. After identifying $k_{\in\{0,1\}}$, the adversary derives a threshold λ to maximize accuracy distinguishing between models trained on datasets

from the two distributions (using a simple linear search). Assuming γ_k is positive,

$$\delta(\Lambda) = \sum_i \mathbb{I}[acc(M_0^i, S_k) \geq \Lambda] + \sum_i \mathbb{I}[acc(M_1^i, S_k) < \Lambda]$$
$$\lambda = \arg\max_{\Lambda} \delta(\Lambda). \tag{3.45}$$

The adversary then predicts $\hat{b} = \mathbb{I}[acc(M, S_k) \geq \lambda]$ (or with a $<$ inequality when $\gamma_k < 0$). Thus, the adversary uses a sample of local models to derive a classification rule, which it then uses to infer the training distribution of the target model. For the same reasons as Loss Test the data used by the adversary here, for both training its local set of models and computing the threshold), is non-overlapping with the victim's data.

3.3.1.3 KL Divergence Attack (KL)

Recent work by Hartley and Tsaftaris [108] demonstrates how the presence of unique features, even if present in one training record, can impact output probability distributions. Motivated by their use of KL divergence to differentiate between the two scenarios (instance present or not), we propose an attack that compares the KL divergence in output probabilities of the victim model using local models.

The adversary prepares by training a collection of local models $\{M_0^1, M_0^2, ...M_1^1, M_1^2, ...\}$, where M_0^i and M_1^i (for some i) denote models from training distributions $\mathcal{G}_0(\mathcal{D})$, $\mathcal{G}_1(\mathcal{D})$ respectively. Let X denote some data randomly sampled by the adversary from the distributions $\mathcal{G}_0(\mathcal{D})$ and $\mathcal{G}_1(\mathcal{D})$, with an equal number of samples ($|X|/2$) from both distributions. We first define a way to estimate the KL-Divergence between two models using predictions:

$$\mathbb{E}[D_{KL}(N \parallel M)] = \mathbb{E}_{x \in X}\left[\sum_{c \in \mathcal{C}} N(x)_c \log\left(\frac{N(x)_c}{M(x)_c}\right)\right] \tag{3.46}$$

where $M(x)_c$ corresponds to the prediction probability corresponding to class c (out of all classes $\mathcal{C}$) for some point x for model M, and the expectation $\mathbb{E}[]$ is taken over the adversary's data X. We use the same data X in computing KL-Divergence values. Next, the adversary defines a "weighted vote" for a pair of models (N, P) with respect to M:

$$\lambda(M, N, P) = \mathbb{E}[D_{KL}(N \parallel M)] - \mathbb{E}[D_{KL}(P \parallel M)]. \tag{3.47}$$

A positive quantity $\lambda(M, N, P)$ thus indicates that the model M has its predictions distributed closer to P than N, since a lower KL-divergence between distributions indicates higher similarity. Using its collection of local models trained on the two candidate distributions, the adversary then computes and aggregates this

"weighted vote" across all pairs of its local models (M_0^i, M_1^j):

$$\hat{b} = \mathbb{I}\Big[\sum_i \sum_j \lambda(M, M_0^i, M_1^j) > 0\Big] \tag{3.48}$$

The rule above thus effectively checks all its pairs of local models and compares similarities in prediction distributions with a given victim model. Since the core idea here is to compare distributions of model predictions, other metrics to compare distributions, like Jensen-Shannon Divergence, or TV Distance, can be used instead of KL-Divergence.

3.3.2 White-Box Attacks

In the white-box setting, the adversary additionally has direct access to the victim's model including its trained parameters. Although this access model assumes a stronger adversary, it is a realistic adversary for many scenarios, like when models are deployed on client devices. It is also useful in two ways: 1) gauging the extent of inference leakage, helping bound risk and understand it better, and 2) studying patterns and trends across properties and models to help better understand distribution risk and come closer to inventing effective defenses.

3.3.2.1 Meta-Classifiers

The state-of-the-art property inference attack uses Permutation-Invariant Networks as meta-classifiers [95]. The meta-classifiers take as input model parameters (weights, bias) and predict the training distribution directly. This architecture is designed to be invariant to neuron orderings inside neural network layers, which it achieves by utilizing the DeepSets [362] architecture.Neuron-ordering invariance is achieved via a set of transforming functions, ϕ_i (for each layer i), over each row of the layer weight matrix. The outputs of these functions are then summed to create a layer representation L^i, thus achieving invariance to the ordering of the neurons within each layer. Since the meta-classifier is itself a classifier that requires many models (800 per distribution [95]) trained on the two distributions for training, this attack is only feasible for adversaries with access to sufficient data from both training distributions and considerable computational resources.

Targeting Convolutional Neural Networks. The Permutation-Invariant Network only supports linear layers in a feed-forward architecture; previous work only considered two or three layer MLPs on small datasets [95], or single-layer recurrent neural networks [371]. Applying the same architecture on top of convolutional layers requires adaptation since kernel matrices are four-dimensional. While there is permutation invariance within channels, the kernel itself is sensitive to permutations by nature of how convolution operations work. We extend property inference attacks to convolutional networks and demonstrate their effectiveness on models with up to eight layers, trained from scratch.

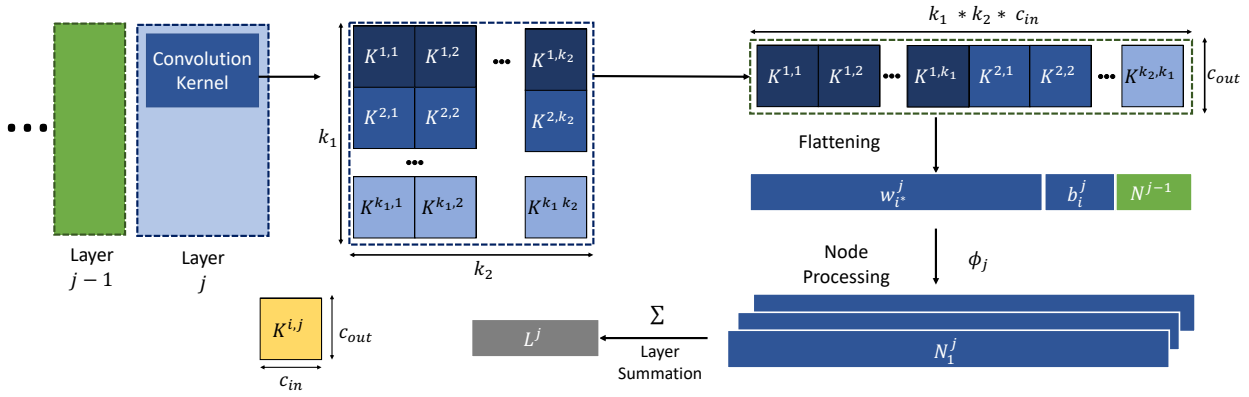


Figure 3.1: Transforming a $k_1 \times k_2$ kernel matrix K (with input channels c_{in} and output channels c_{out}) into a 2-dimensional weight matrix for compatibility with the Permutation Invariant Network architecture. The node-processing functions ϕ_i and the rest of the pipeline are identical to the Permutation Invariant Network described in §3.3.2.1.

Figure 3.1 illustrates our method. Let K be a kernel of size (k_1, k_2) associated with some convolutional layer, with input and output channel dimensions c_{in} and c_{out} respectively. While designing the architecture to capture invariance, it is important to remember that unlike neurons in linear layers, positional information in the kernel matters. Thus, any attempt to capture invariance should be limited to the mapping between input and output channels of a convolutional kernel. We flatten the kernel of size $(k_1, k_2, c_{in}, c_{out})$ such that the resulting matrix is of size $(k_1 \times k_2 \times c_{in}, c_{out})$. Concatenating along the input channel dimension helps preserve location-specific information learned by the kernel while capturing permutation invariance across channels.

This two-dimensional matrix is then processed like linear layers are in Permutation-Invariant Networks (using the same notation as §3.3.2.1), applying function ϕ_i and summing to capture invariance while concatenating feature representations from prior layers. Like the original architecture, the bias component can be concatenated to the kernel matrix itself. Since this feature extraction process on a convolutional layer also produces a layer representation, it can be easily incorporated into the existing architecture to work on models with a combination of convolutional and linear layers.

3.3.2.2 Affinity Graph Attack

Instead of looking at model parameters directly, we propose a technique that focuses on the relationship between data across models. The attack starts with p data-points π, $|\pi| = p$. For a given model m, we pass each data point π_i through the model, collecting intermediate feature activations l_i^j after each layer j. Then, for each layer j we compute the cosine similarity between every point π_i and π_k: $cos(l_i^j, l_k^j)$. This is repeated for every $\binom{p}{2}$ pair of data-points, yielding a list of cosine similarity scores per layer. Since $\binom{p}{2}$ gets large very quickly, we train a small neural network model ϕ to learn a smaller representation of these lists of similarity

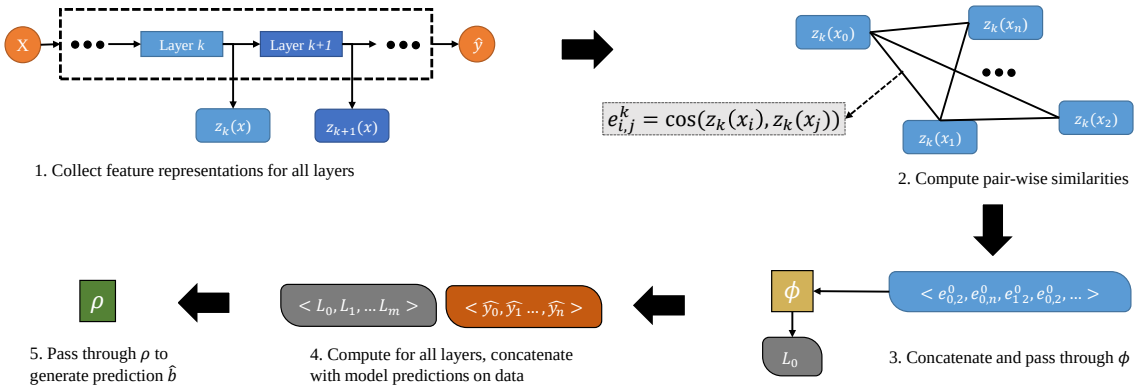


Figure 3.2: Flow diagram for the Affinity Graph Attack classifier. For each model, latent features are collected for all datapoints (1). Then, pair-wise cosine similarities of the features are computed (2) and processed layer-wise (3), leading to a single representation for the given model (4). A linear classifier can then be used to generate predictions (5).

scores. These scores are concatenated and passed through the model ϕ, for layer j:

$$L_j = \phi\big(\cos(l_0^j, l_1^j),\ \ldots,\ \cos(l_0^j, l_p^j),\ \ldots,\ \cos(l_{p-1}^j, l_p^j)\big).$$

This process is repeated for all layers in the model and these layer-wise features are then concatenated and passed through a linear layer ρ, along with model predictions, to obtain a final prediction via the meta-classifier:

$$\hat{b} = \rho(L_0,\ L_1,\ \ldots,\ L_m,\ \hat{y}_0,\ \hat{y}_1,\ \ldots,\ \hat{y}_n).$$

Figure 3.2 illustrates our method. Note that the network ϕ is shared across all layers. Since this meta-classifier is designed to learn how features for various data changes across layers, having a shared ϕ model helps the meta-classifier learn the desired inference task easier, in addition to lowering the number of trainable parameters. This attack is both permutation- and scale-invariant by design, since the cosine similarity function is also scale and permutation invariant. Additionally, while PIN is applicable only to feed-forward neural networks and CNNs, AGA works with any model architecture from which intermediate features can be extracted, including models with skip-connections like ResNets [114] and RNNs [265]. Instead of trying to model inter-layer relationships by explicit concatenation of prior layers (like permutation-invariant networks), layer-wise representations are simply concatenated, and the final model ρ can learn relationships between layers.

3.4 Experiments

To better understand the risks of distribution inference, we execute distribution inference attacks to measure their ability to distinguish distributions with varying disparity on tabular, image, and graph datasets. Although it is unknown how close these attacks are to the best possible distribution inference attacks, they help demonstrate an empirical lower bound on adversarial capabilities and can be helpful in estimating general trends.

Dataset	**Task**	**Property**
Census	Income prediction	Ratio of females Ratio of whites
Census19	Income prediction	Ratio of females Ratio of whites
Texas-100X	Surgical procedure prediction	Ratio of females Ratio of whites Ratio of Hispanics
CelebA	Smile identification Gender prediction Mouth-open prediction	Ratio of females Ratio of young people Ratio of people with wavy hair Ratio of people with high cheekbones
RSNA Bone Age	Age prediction Gender prediction	Ratio of females Ratio of people below age threshold
ogbn-arxiv Chord	Node classification	Mean node-degree Average clustering coefficient

Table 3.1: Descriptions of datasets, along with the tasks and properties used in the experiments.

3.4.1 Datasets

We evaluate our attacks on fifteen task-property pairs across seven datasets, summarized in Table 3.1. Our experimental datasets were selected to:

1. incorporate common benchmarks (Census, CelebA) to enable comparisons with previous work,
2. to study the impact of task-property correlation on inference risk (various property-task pairs for CelebA),
3. to apply our definitions beyond ratio-based properties on graphs (mean node-degree on ogbn-arxiv, clustering coefficient on Chord), and
4. to include datasets representing real-world use-cases, like Census19 and Texas-100X.

We begin with evaluations using weaker attacks (such as Loss Test, Threshold Test) to demonstrate risk even under the presence of simple attacks (§3.4.3), and then proceed to more sophisticated attacks like KL on larger datasets (§3.5).

As noted by Zhang et al. [371], the target properties for a distribution inference attack can be either related to or independent of the task, and can be either explicit or latent features of the input data. For instance, attributes varied for Census are feature-based properties since these attributes are directly used as features for the models trained on them. On the other hand, an attribute like the age of a person is unrelated to detecting smiles and is a latent property that is not directly encoded as an input feature in the training data (but is available for our CelebA experiments from provided metadata).

We construct non-overlapping data splits between the simulated adversary, $\mathcal{A}$, and model trainer $\mathcal{T}$. These non-overlapping splits help better capture a realistic scenario where the adversary has access to training data from the distribution $\mathcal{D}$ but is unlikely to have any of the model trainer's data (which is considered private). Both parties then modify their data to emulate a distribution property, and then sample training datasets from these adjusted distributions to train their models. This sampling, along with the disjoint data splits between $\mathcal{A}$ and $\mathcal{T}$, helps ensure that any distinguishing power we observe is actually distribution inference, rather than inadvertent dataset inference.

Census [23] consists of several categorical and numerical attributes like age, race, education level to predict whether an individual's annual income exceeds \$50K. We focus on the ratios of whites (race) and females (sex) as properties and use three-layer feed-forward networks.

Census19 [292] is an updated and expanded version of the Adult Census dataset [23] based on data from the US Census Bureau. It contains a mixture of numerical and categorical features, and the same prediction task. We focus on the ratio of whites (race) and females (sex) as properties, and use a two-layer feed-forward neural-network as the architecture.

Texas-100X [136] contains demographic and medical information for patients across hospitals. The original dataset uses 100 possible classes for surgical procedure prediction. We slightly modify the task and focus only on data from the top 20 classes, reducing it to a 20-class classification task. We focus on the ratio of whites (race), females (sex), and Hispanics (ethnicity) as properties, and use a two-layer feed-forward neural-network.

CelebA [190] contains face images of celebrities, with multiple images per person. We use three different tasks: smile detection, gender prediction, and mouth-open prediction. We conduct experiments with a convolutional neural network trained from scratch for this dataset, with five convolutional layers and pooling layers followed by three linear layers, which is the smallest network we could find with reasonable task accuracy. For our experiments with feature extractors, we also conduct experiments where the adversary uses a pretrained FaceNet [272] model trained on the CASIA-WebFace [358] dataset, with a two-layer network. It leads to a drop in performance (from $\sim 92\%$ to $\sim 82\%$), but the point of such an experiment is indeed to assess inference risk in more practical settings.

For the attack inference properties, we use the proportion of females (smile-detection task), old people (gender-prediction task), people with wavy hair (mouth-open-prediction task), and people with high cheekbones (mouth-open prediction task). These pairs are useful in comparing results with previous works, and also help cover a spectrum of different correlations between the task and property attributes.

RSNA Bone Age [106] contains x-ray images of hands, and the standard task is to predict the patient's age in months. We convert the task to binary classification based on an age threshold (> 132 months), and focus on the ratios of the females (available as metadata) as properties. We also consider a flipped scenario, where the task is to predict females, with the ratios of people below the age threshold as properties. We use a pretrained DenseNet [122] model for feature extraction, followed by a two-layer network for classification. Similar to CelebA, we consider a setting where the adversary uses pretrained feature extractor, while the victim trains models from scratch. Additionally, we also consider a setting where both the victim and adversary use the same feature extractor, but use different model architectures on top of the feature extractor

The **ogbn-arxiv** [330] dataset is a directed graph, representing citations between computer science arXiv papers. The task is to predict the subject area categories. We infer the mean node-degree property of the graph using four-layer Graph Convolutional Networks [158].

Chord [375] contains botnets with the Chord [290] topology artificially overlaid on top of background network traffic from CAIDA [89]. The dataset contains multiple graphs, with the task of detecting bot nodes in the graphs. We focus on inferring whether the underlying graphs (onto which we overlay botnets) have average clustering coefficients within a specific range. Following the model architecture proposed in Zhou et al. [375], we implement a Graph Convolutional architecture.

3.4.2 Experimental Details

For each dataset, we create non-overlapping splits of data for the victim and adversary, where the victim has at least $2\times$ the amount of adversary's data for ogbn-arxiv and RSNA Bone Age, $3\times$ for CelebA and Texas-100X, and $4\times$ for Census19. For each dataset, we simulate $\mathcal{D}$ using the dataset itself. Simulation of distributions with particular α values is is achieved by sampling data with attributes 0 and 1 such that their ratios result in some desired α. For properties not based on binary attributes like ogbn-arxiv, this is achieved by pruning nodes iteratively from the graph (while re-computing neighbor counts along the way) to achieve a desired mean node-degree. To obtain non-overlapping splits, both parties (victim and adversary) sub-sample from their data splits (to achieve specific α values) with different random seeds, and train models on the sampled data.

We perform each experiment five times and report mean values with standard deviation in all of our experiments. For each dataset, we train 250 victim models per distribution. For all black-box attacks, the adversary trains and uses 50 models per distribution for each trial. For white-box attacks, the adversary

trains 800 models per distribution, of which 750 are used for training and 50 as the validation set. For cases with very large models (like DenseNet trained from scratch for RSNA Bone Age), we use 100 victim models per distribution.

For the classifiers, we vary α_1 in $[0.0, 1.0]$ at intervals of 0.1, and set $\alpha_0 = 0.5$ for the case of ratio-based properties, where a certain α value for a distribution means datasets sampled uniformly at random would have α fraction of the data with the property attribute 1, for e.g., ratio of females. The distinguishing accuracies thus correspond to predicting whether a model has the training distribution corresponding to α_0 or α_1 where random guessing would be 50% accuracy, and perfect predictions would be 100%. Since the Loss Test uses a fixed test set per experiment, its results show no variation.

Loss Test. The adversary uses its test data to sample the two test sets S_0 and S_1. Since we use the same test data in evaluations, we turn off sampling while generating data with desired properties for this setting.

Threshold Loss. The adversary trains 50 models per distribution on its data split.

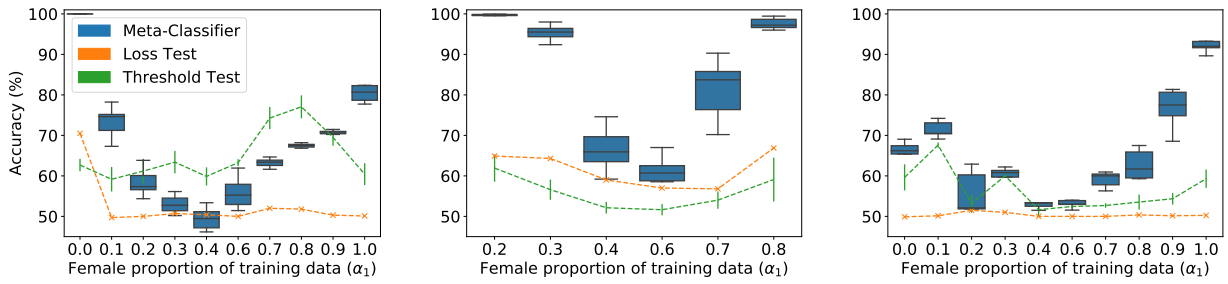
Meta-Classifier. We used Permutation Invariant Networks as our meta-classifier architecture [95]. The simulated adversary produces 800 models per distribution using its split of data to train the meta-classifier. For the case of CelebA, we use our extension of the Permutation Invariant Network that is compatible with convolutional layers (§3.3.2.1). Following experimental designs from prior works, we were able to achieve the accuracies that the authors reported (§3.4.3.1). However, using our experimental design leads to significantly lower distinguishing performance. Steps like ensuring no overlap in victim/adversary data, randomly sampled datasets for $\mathcal{G}_0(\mathcal{D})$ and $\mathcal{G}_1(\mathcal{D})$, and ensuring the same dataset size are necessary to avoid the risk that the meta-classifier is identifying something different about the distributions other than the claimed property. We think these steps are important for realistic experiments, so report the distinguishing accuracies based on this experimental design, even if they are lower for the same attacks than the results reported for the same tasks in previous work.

KL Divergence Attack. Since using all pairs of adversary's models can be expensive, the attack uses a set fraction (0.8) of randomly chosen pairs to compute the expectation in Equation (3.48). We experiment with multiple values of this fraction, and observe comparable performance. For each pair of local models, the attack collects the difference in KL values. These differences are then normalized across all differences observed for local models, after which the adversary uses voting-based aggregation to generate the final prediction. We also experimented with variants that do not include voting, but find the current version to perform best.

Affinity Graph Attack. Similar to the KL Divergence Attack, we use a fraction (0.2) of randomly chosen

Dataset	Task	Property	Binary		Regression
			n_{leaked}	$\|\alpha_0 - \alpha_1\|_{0.75}$	n_{leaked}
Census	Income prediction	Ratio of females	0.2	0.5	8.8
		Ratio of whites	0.1	0.6	6.0
CelebA	Smile identification	Ratio of females	0.3	0.5	10.6
	Gender prediction	Ratio of young people	0.2	0.6	5.1
RSNA Bone Age	Age prediction	Ratio of females	6.2	0.2	269.4
ogbn-arxiv	Node classification	Mean node-degree	30.6	-	-
Chord		Average clustering coefficient	-	-	-

Table 3.2: Effectiveness of basic inference attacks (best of all Loss Test, Threshold Test, and meta-classifier for binary classification, and meta-classifier for regression) while varying ratios of distributions. $\|\alpha_0 - \alpha_1\|_{0.75}$ is the minimum difference in ratios observed that has at least 75% average accuracy. For binary classification, n_{leaked} is the median effective n value based on Theorem 3.2.2 using the maximum distinguishing accuracies across all experiments and pairs of property ratios (degrees in the case of ogbn-arxiv) without outliers. For regression, n_{leaked} is the mean effective n value based on Theorem 3.2.3 across all ratios excluding 0 and 1. Size for ogbn-arxiv refers to number of nodes, and the average number of nodes for Chord.



(a) Census (b) RSNA Bone Age (c) CelebA

Figure 3.3: Classification accuracy for distinguishing proportion of females in training data for (a) Census, (b) RSNA Bone Age, and (c) CelebA. The RSNA Bone Age dataset does not include ratios below 0.2 or above 0.8, since sampling the original dataset for that ratios produces datasets that are too small to train models with meaningful performance. Performance of the attacks increases as the distributions diverge (α_1 moves away from 0.5), but is not symmetric.

pairs of points. Since a given model can have multiple layers, we use specific layers for each model while training the meta-classifiers. These specific layers are fixed across all model architectures (per dataset) and were selected before the experiments were evaluated (based on computational constraints), making them free of selection bias.

3.4.3 Demonstrating Inference Risk: Binary Properties

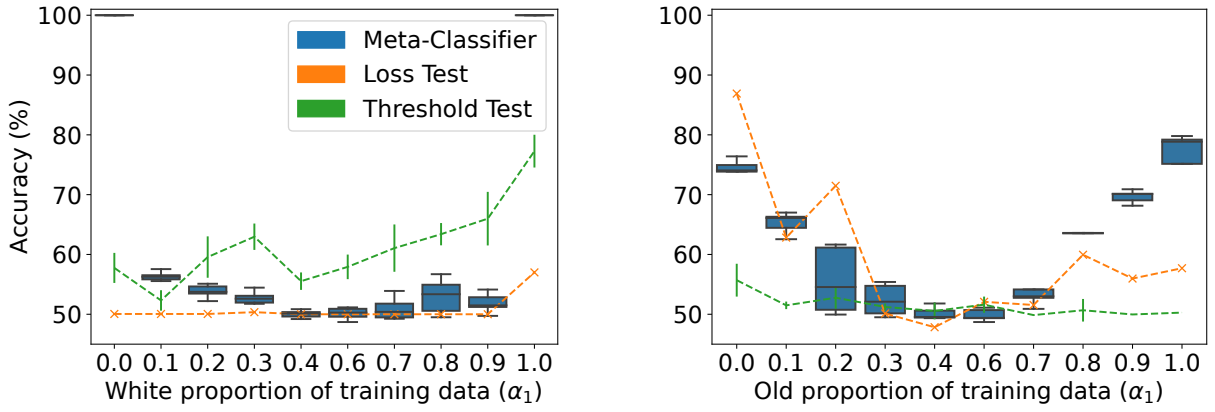
To demonstrate inference risk, we evaluate leakage with binary properties using Loss Test, Threshold Test, and meta-classifiers.

We fix $\mathcal{G}_0$ and try the attacks on a range of $\mathcal{G}_1$ distributions for the first set of experiments. In §3.4.5, we report on experiments varying both distributions. Most prior works on distribution inference use arbitrary ratios, like distinguishing between 42% and 59% males [95]. Only recently have works started transitioning to more controlled experimental settings, like comparable dataset sizes for the victim and adversary and non-overlapping data sampling [376]. While having one of the ratios corresponding to the estimate of the underlying data distribution is justified, fixing the other arbitrarily makes it hard to understand the adversary's capabilities—we want to understand how dissimilar the distributions must be in order to be distinguishable. Additionally, the lack of keeping ratios consistent in experiments across properties makes it harder to compare information leakage across properties. Analyzing such trends is important for understanding how much of these properties are leaked across different configurations (explicit attribute, latent property) and assess the adversary's capabilities under different scenarios (black-box access, white-box access).

Experiments with binary classification for properties can provide useful insights into the effectiveness of distribution inference attacks, but most realistic attacks would not be based on distinguishing between two known distributions. In §3.4.5.1, we consider attacks that can infer the underlying ratio without any prior assumptions about distinguishing particular distributions.

3.4.3.1 Distinguishing Imbalanced Ratios

Since the original ratios for the targeted property may be unbalanced in the dataset (§3.1), for these experiments we fix $\mathcal{G}_0$ to a balanced ($\alpha_0 = 0.5$) ratio for the chosen attribute for the Census, CelebA, and RSNA Bone Age datasets. Then, we vary α_1 to evaluate inference risks and understand how well an adversary could distinguish between models trained using distributions with different proportions of the targeted property.



(a) Census, race

(b) CelebA, age

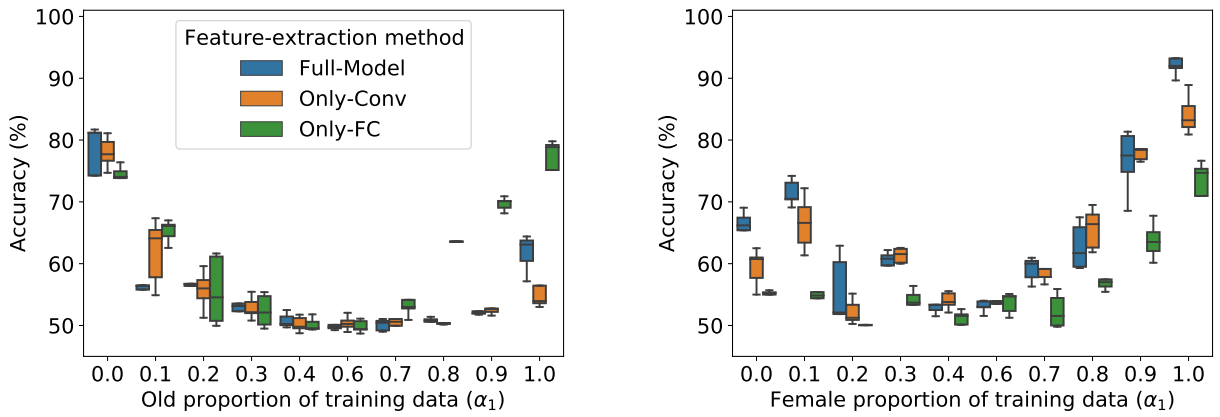
Figure 3.4: Distinguishing accuracy for proportion of (a) whites in training data for the Census and (b) old people in the CelebA. The black-box attacks approach the performance of white-box meta-classifier attacks for some distributions (especially the Threshold Test and the Census (race) task).

Census. We summarize the accuracies for the three attack methods across varying proportions of females in Figure 3.3a and whites in Figure 3.4a. The Loss Test performs only marginally better than random guessing in most cases for females, yielding n_{leaked} values in the range $[0, 0.03]$. On the other hand, the Threshold Test and meta-classifiers are able to achieve non-trivial distinguishing accuracies for the female proportion, with n_{leaked} values in ranges $[0.1, 1.2]$ and $[0.02, 6.5]$ respectively, with a similar median n_{leaked} value of 0.33, showing how the two are not very far apart in effectiveness. Leakage increases as the distributions become more disparate, with near-perfect distinguishing accuracy for the extreme case of $\alpha_1 = 0$. For race, none of the attacks detect anything ($n_{\text{leaked}} \approx 0$) apart from the surprising results on Threshold Test, which performs asymmetrically well, approaching 80% accuracy for mostly-white distributions.

Comparison with previous results. Ganju et al. [95] applied their meta-classifier method on two properties on this dataset: 38% vs. 65% women (case A), and 0% vs. 87% whites (case B). For case B, the Loss Test performs as well as random guessing (50.1%), while the Threshold Test ($92.4 \pm 2.6\%$) approaches meta-classifier performance ($99.9 \pm 0.1\%$), achieving a high (compared to $\alpha = 0.5$ experiments) $n_{\text{leaked}} \approx 11$. For case A, the Threshold Test ($62.7 \pm 2.0\%$) outperforms meta-classifiers ($62.1 \pm 1.7\%$) while achieving $n_{\text{leaked}} \approx 0.03$, barely better than random guessing , and the Loss Test method fails (50%). Ganju et al. report 97% accuracy for case A and 100% for case B. We were able to closely reproduce these results in their setting which includes overlapping data between victim and adversary and does not ensure changes in ratios do not affect dataset size or class imbalance. In the more realistic experimental design in which we ensure there is no victim/adversary overlap and maintain the label ratios and same dataset sizes (§3.4.1), the accuracies are much lower—for example, in case A the distinguishing accuracy is 97% using their experimental design but drops to 62% when more carefully prepared datasets are used in our design. These results suggest that although the distinguishing accuracy is high between the two distributions in the tests in their setting, the attacks are not actually inferring the intended property but are predicting the distribution based on other differences between the datasets.

RSNA Bone Age. Distinguishing accuracies for the female proportion on the RSNA Bone Age dataset are plotted in Figure 3.3b. The simple Loss Test performs nearly as well as the Threshold Test although both have low leakage, with a median of $n_{\text{leaked}} \approx 0.2$ and $n_{\text{leaked}} \approx 0.1$ respectively. The meta-classifier attack, on the other hand, has a much higher leakage of $n_{\text{leaked}} \approx 6$.

CelebA. Figure 3.3c shows the distinguishing accuracy for the CelebA data on proportion of females, and Figure 3.4b for the proportion of examples marked as "old". The Threshold Test performs much worse compared to Census and RSNA Bone Age, with the median $n_{\text{leaked}} < 0.05$, compared to ≈ 0.7 using meta-classifiers. Figure 3.5 shows meta-classifier prediction accuracy for three different representations of the shadow models used to train the meta-classifier: using parameters from only linear layers, only convolutional layers, and all layers of the models. While inferring the ratio of old people (Figure 3.5a), including just



(a) CelebA, age

(b) CelebA, females

Figure 3.5: Distinguishing accuracy for meta-classifiers for proportion of (a) old people and (b) females in the training data for CelebA. Using all the layers' parameters is not necessarily helpful and can lead to lower performance (e.g., CelebA, females).

the fully-connected layers works best, yielding $n_{\text{leaked}} \approx 0.12$ and not too far off from using all layers or just the convolutional layers ($n_{\text{leaked}} \approx 0.07$). For the case of sex ratios (Figure 3.5b), using the full model helps extract more information ($n_{\text{leaked}} \approx 0.32$) than either of the convolutional ($n_{\text{leaked}} \approx 0.24$) or linear ($n_{\text{leaked}} \approx 0.05$) layers. These trends suggest the likelihood of some layers' parameters capturing specific property-related information better than the others. Linear layers are more helpful for ratios of old people whereas for ratios of females, convolutional layers are significantly better. We explore this phenomenon further in §3.4.4.

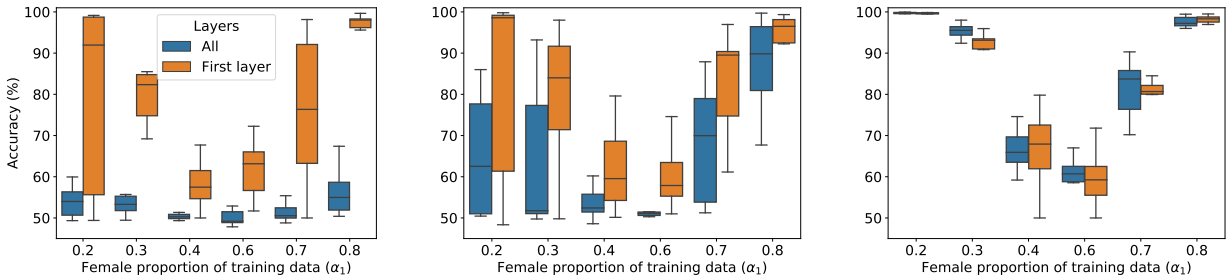
3.4.4 Leakage by Layers

Observing differences in performance when focusing on different network layers of the same model for raises an interesting question: *how does information leaked vary across model layers?* Understanding and identifying which layers leak the most information can help better understand the distribution inference risks and how to mitigate them, as well as how to make attacks more efficient. Here, we propose a simple test to help the adversary rank layers for value in distinguishing between the given distributions, and show how some layers (the first layer, in most cases) seem to capture properties of the training distribution better than others in a given model. Meta-classifier attacks are expensive and deciphering what they learn is challenging—identifying critical parameters can both improve understanding and lower resource requirements. If we can use just a fraction of the model's parameters, we may be able to achieve comparable inference performance with fewer shadow models and much lower meta-classifier training costs.

Identifying Useful Layers. Let j be some layer of the model for which the adversary wishes to gauge inference potential. We optimize query point $\hat{x}$ to maximize the difference in the total number of activations

Dataset	Layer						
	1	**2**	**3**	**4**	**5**	**6**	**7**
CelebA (female)	**89.2**	76.7	75.8	74.2	70.0	66.7	63.3
CelebA (old)	64.2	60	60	68.3	**73.3**	70	66.7
Census (female)	**62.0**	58.0	56.4	-	-	-	-
Census (white)	**81.7**	75.0	63.3	-	-	-	-
RSNA Bone Age	**65.0**	64.0	-	-	-	-	-
ogbn-arxiv	93.3	95.0	**98.3**	-	-	-	-
Chord	69.4	60.0	64.0	64.8	57.2	50.2	-

Table 3.3: Maximum accuracy using layer-identification method. Since the last layer in all of these models is used for classification with a Softmax/Sigmoid activation, the process in Equation 3.49 cannot be applied to the last layer.



(a) 10 models (b) 40 models (c) 1600 models

Figure 3.6: Classification accuracy for distinguishing between models with different training distributions on the RSNA Bone Age dataset, for meta-classifiers trained with (a) 10, (b) 40, and (c) 1600 models. Orange box plots correspond to using parameters only from the first layer, while blue box plots correspond to using all (three) layers' parameters. Although the experiment that uses just the first-layer's parameters has much more variance, its average performance (and even the first quartile) is better than that of the version that uses all the layer's parameters.

for layer j between models trained on datasets from the two distributions:

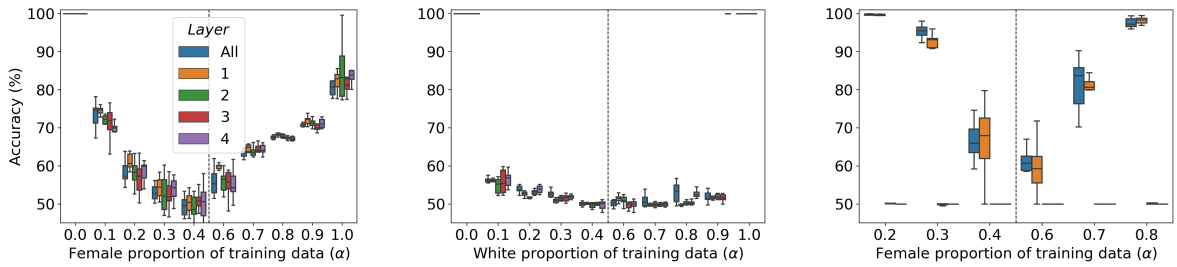
$$M_j(x) = \sum_i \mathbb{I}[(M[:j](x))_i > 0]$$

$$\hat{x} = \arg\max_x \left| \sum_{i;y_i=0} M_j^i(x) - \sum_{i;y_i=1} M_j^i(x) \right|, \tag{3.49}$$

where $M[:j](x)$ refers to the activations after layer j of model M on input x. The adversary can use a set of test points to select one that maximizes the above constraint. Then, similar to the process for Threshold Test (Equation 3.45), the adversary finds a threshold on the number of activations to maximize distinguishing accuracy. By iterating through all layers and computing the corresponding accuracies, the adversary can create a ranking of layers to estimate how much information these layers can potentially leak. This process is computationally much cheaper than running a meta-classifier experiment for all layers, and can be done with

as few as 20 models. Once it has ranked all the layers, the adversary can pick the most informative ones (even just a single layer suffices in some cases) to train the meta-classifier. Since the resulting meta-classifier has fewer parameters (as it computes over fewer model layers), it can be trained using far fewer shadow models than when all network parameters are used, without having a significant impact on distinguishing accuracy.

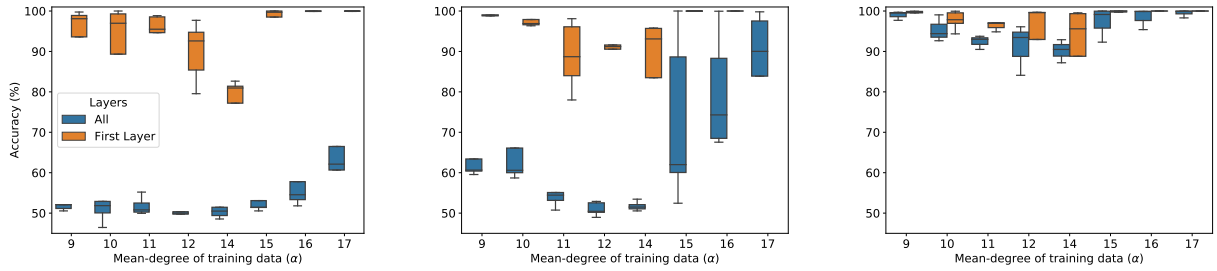
Results. To understand how well the layer-identification process correlates with meta-classifier performance, we also perform experiments where each layer's parameters are used one at a time to train the meta-classifier. We run the layer-identification process, as described in Equation (3.49), for all layers across datasets. For the numbers reported in Table 3.3, the adversary samples data from its local test set to maximize Equation (3.49). Distinguishing accuracies reported in this table are on the adversary's models since it uses this ranking of layers to train a meta-classifier for its attack on the targeted model. For most cases, the layers closest to the inputs are identified as most useful. These accuracies for CelebA align with observations from previous experiments (§3.4.3) as well—for distinguishing sex ratios, the convolutional layers (until layer 5) seem to be more useful; for age, the fully-connected layers appear to be most useful. Layers of machine-learning models closer to the input are commonly associated with learning generic patterns, and later layers more abstract ones along with invariance to the given task [248]. Thus, the position of layers identified to be most useful is telling of how close the target property is to the input space or task.



(a) Census (female) (b) Census (white) (c) RSNA Bone Age (female)

Figure 3.7: Classification accuracy for distinguishing between training distributions for unseen models for Census (sex: left, race: middle) and RSNA Bone Age, while varying the models' layers used while training meta-classifiers. There is no clear winner in the case of Census, while the first layer seems to the most useful for the case of RSNA Bone Age.

Excluding the last layer does not lead to a significant performance drop. Intuitively, layers closer to the output will capture invariance for the given task and are thus less likely to contain any helpful information that prior layers would not already capture. If the last layer reveals enough information for the attack to succeed, then a black-box attack should also be possible. Results from layer-wise meta-classifier experiments confirm how the last layer's parameters rarely appear useful for distribution inference. Interestingly, this is the opposite of what Nasr et al. [229] observed for membership inference. Using these observations, we train meta-classifiers while using parameters only from some of the layers selected on the ranking we obtain via



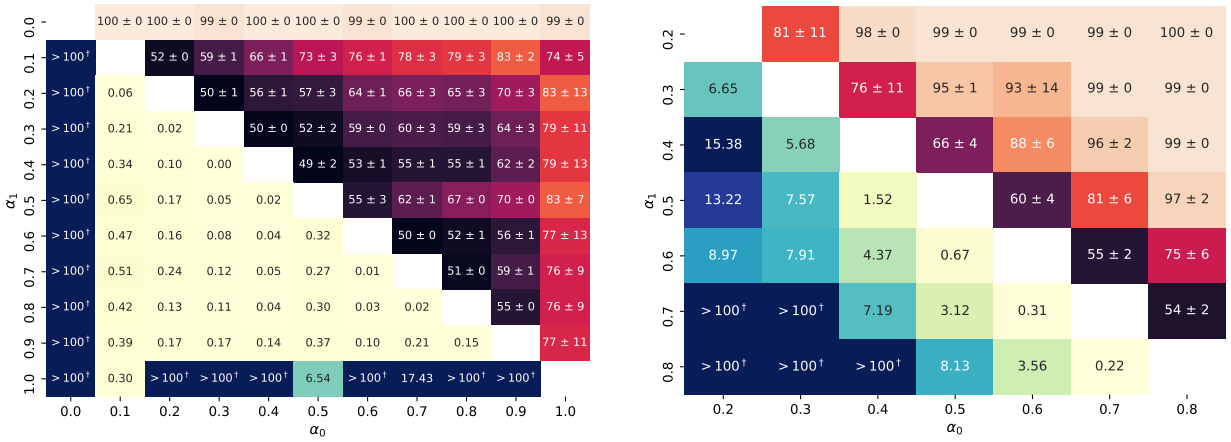
(a) 20 models (b) 100 models (c) 1600 models

Figure 3.8: Classification accuracy for distinguishing between models with different training distributions for on the ogbn-arxiv dataset. Left to right: meta-classifiers trained using 20, 100, and 1600 (original experiment) models, respectively. Orange box plots correspond to using parameters only from the first layer, while blue box plots correspond to using all (three) layers' parameters. Using as few as 20 models is sufficient for satisfactory meta-classifier performance when the right layers are identified and used.

layer-identification experiments. We observe a clear advantage of doing so across all datasets, with minimal decreases in accuracy. For instance, using just the first layer produces a meta-classifier with only 20 training models on RSNA Bone Age (orange boxes in leftmost graph in Figure 3.6) that performs much better than using parameters from all of the layers. In order for the meta-classifier trained on all parameters to approach the accuracy of the one-layer meta-classifier, hundreds of shadow models are needed.

For datasets like Census, all the layers seem to equally useful while for RSNA Bone Age, only the first layers' parameters are useful (see Figure 3.7). When using the first layer's parameters, the adversary can achieve an average of 75% accuracy with as few as 20 models, compared to 54% when using the entire model. In fact, the first layer is identified as most useful and using any other layer leads to near-random performance. Additionally, for larger models like those for CelebA, the adversary can pick more than one layer—using as few as three layers of the model can help lower computational resources. As observed in ablation experiments with convolutional and linear layers for CelebA (old people), using just the last three layers (of which two the layer-identification process identifies), the adversary can train its meta-classifiers while using significantly fewer models. For instance, when using 100 models to train the meta-classifier, using just the fully-connected layers gives a 4% absolute improvement in accuracy, along with 0.5% reduction in standard deviation across experiments.

Graph Datasets. The layer-identification process does not work on the graph datasets. It incorrectly predicts the third layer as most useful for ogbn-arxiv, whereas actual performance with that layer's parameters leads to a significant performance drop. We suspect this behavior can be explained by the inherent properties of the graph data. Intermediate activations for nodes can have complicated interactions with neighboring nodes, leading to the detection method's instability when analyzing activation values. These challenges on graph datasets is something that we plan to investigate in future work. Nonetheless, the fact the first two layers are useful for both graph datasets suggests a useful direction for graph-based distribution inference



(a) Census (b) RSNA Bone Age

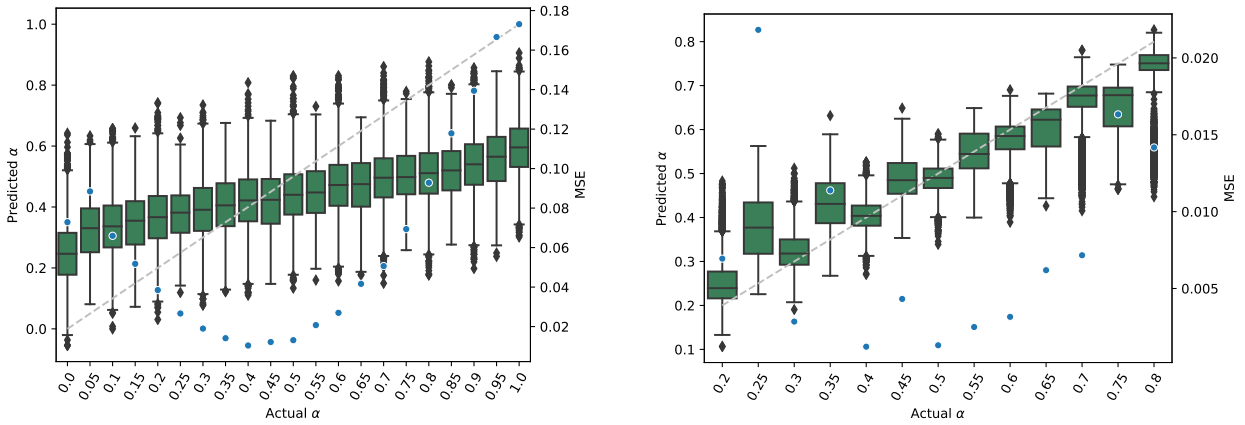
Figure 3.9: Effectiveness of meta-classifiers in distinguishing proportions of females on Census and RSNA Bone Age. The bottom-left triangles of the heatmaps show the n_{leaked} values, and the top-right triangles show the distinguishing accuracies between training distributions $\mathcal{G}_0(\mathcal{D})$ with ratio α_0 and $\mathcal{G}_1(\mathcal{D})$ with ratio α_1 for females. Distinguishing accuracies seem to follow intuitive patterns, with an increase as the distributions diverge (larger $|\alpha_0 - \alpha_1|$). The n_{leaked} values allow for comparisons of attack power between different pairs of distributions, but also show that very little leakage is observed for most settings, except for RSNA Bone Age.

attacks.

3.4.5 Varying Proportions

An adversary may not necessarily be interested in distinguishing between the balanced case ($\alpha = 0.5$) and other ratios. For instance, health datasets for specific ailments may have a higher underlying prevalence in females, and the adversary may be interested in differentiating between two particular ratios of females, like 0.3 and 0.4. We thus experiment with the case where both distributions are varied: $\mathcal{G}_0(\mathcal{D})$ and $\mathcal{G}_1(\mathcal{D})$ with corresponding ratios α_0 and α_1 respectively. Distribution inference risk seems particularly acute when an adversary can distinguish the proportion of an uncommon property. Observing performance trends as the difference in ratios increases also helps us understand how much of a threat distribution inference may pose as the similarity of the distributions varies.

Figure 3.9 shows the distinguishing accuracies (and corresponding n_{leaked} values) between models (in the form of heatmaps) trained on distributions $\mathcal{G}_0(\mathcal{D})$ and $\mathcal{G}_1(\mathcal{D})$ while varying corresponding α_0 (horizontal axis) and α_1 (vertical axis), for ratios of females on Census and RSNA Bone Age. For instance, in Figure 3.9a, $(\alpha_0, \alpha_1) = (0.2, 0.9)$ in the upper-right triangle correspond to meta-classifier performance (70%), while $(\alpha_0, \alpha_1) = (0.9, 0.2)$ in the lower-left triangle gives the corresponding $n_{\text{leaked}} = 0.17$. Entries along a given diagonal have the same value of $|\alpha_0 - \alpha_1|$. As reflected in the heatmap colors, distinguishing accuracies are roughly the same along diagonals. The variance in performance across runs is relatively high for similar distributions (small $|\alpha_0 - \alpha_1|$) and decreases as the distributions diverge. For CelebA (sex) and Census (sex),



(a) Census (Sex)

(b) RSNA Bone Age

Figure 3.10: Predicted α values (left y-axis) for models with training distributions for varying α values (x-axis), for all victim models and regression meta-classifier experiments (green box-plots), along with mean squared error (right y-axis labels, with different scales on the two graphs, and blue dots), for (a) Census and (b) RSNA Bone Age datasets. with The diagonal gray dashed line represents the ideal case, where the regression classifier perfectly predicts α. For each ratio of the form $0.05 \cdot x$ (for varying x), we train regression meta-classifiers 5 times with different seeds, and test 100 victim models. As indicated by n_{leaked} values, the RSNA Bone Age dataset observes very good performance, with nearly all predictions lining up with the diagonal, while for Census (sex), predicted ratios are usually in $[0.2, 0.6]$.

we observe that $n_{\text{leaked}} < 1$ in most cases. These small values do not imply the inability of *any* adversary to distinguish between the distributions, only that for the given attacks we observe little information leakage.

3.4.5.1 Direct Regression over α

Inspired by Zhou et al. [376], we performed a direct regression experiment in which we trained the meta-classifiers to predict α directly. This corresponds to a more realistic attack setting for many scenarios than one in which the adversary is distinguishing between two predefined α values. For this experiment, we construct a training dataset for the regression meta-classifier with tuples of the form (M_α, α), where M_α is some model with a training distribution corresponding to the ratio α. We train the meta-classifier using M_α models for all the ratios α that we experiment with in §3.4.5 ($\{0.0, 0.1, \ldots, 1.0\}$ for Census and CelebA, and $\{0.2, \ldots, 0.8\}$ for RSNA Bone Age). The meta-classifier follows the same permutation-invariant architecture as in the binary property experiments (§3.3.2), just with a mean squared error (MSE) loss for training.

Figure 3.10 shows the distribution of the predictions of the regression meta-classifiers and Table 3.4 reports the MSE and n_{leaked} values. For all these experiments, we train meta-classifiers five times with different seeds, and report aggregate results over all the meta-classifiers and victim models, for each dataset and property. In the plots in Figure 3.10, we include results for actual α values at both tenths and the intermediate 0.05 ratios to confirm that the meta-classifiers are indeed learning to predict α and not just overfitting to the α values

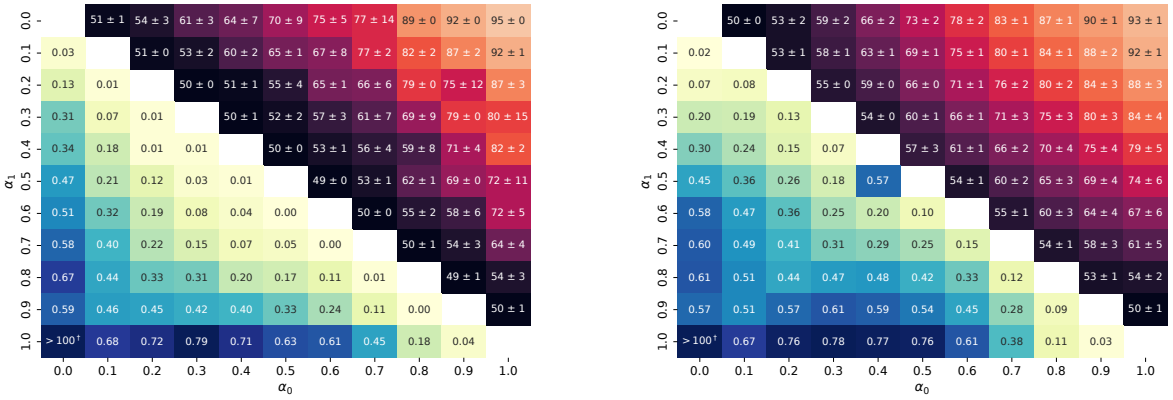
Dataset	Attribute	n_{leaked} (B)	n_{leaked} (BR)	n_{leaked} (R)	MSE
Census	Sex	0.2	0.3	8.8	0.053
	Race	0.1	0.2	6.0	0.091
CelebA	Sex	0.3	0.4	10.6	0.030
	Age	0.2	0.4	5.1	0.047
RSNA Bone Age	Sex	6.2	15	269.4	0.001

Table 3.4: Median n_{leaked} values when using binary meta-classifiers n_{leaked} (B), regression meta-classifiers n_{leaked} (R), and regression meta-classifiers for binary predictions n_{leaked} (BR), along with average Mean Squared Error (MSE) for direct regression over α. n_{leaked} is nearly double for all cases that use regression meta-classifiers for binary predictions, when compared to the binary meta-classifiers.

observed in training. The meta-classifiers do exhibit a bias toward balanced predictions, showing a smooth curve for the MSE values with a minimum near $\alpha = 0.5$.

The attacks are quite successful in most cases, achieving $n_{\text{leaked}} > 5$ for all of our settings, and surprisingly high leakage for RSNA Bone Age with $n_{\text{leaked}} > 260$. These regression attacks show that adversaries can infer sensitive information about training datasets even in the more realistic setting where adversaries do not have prior knowledge of distributions. However, the attacks are not always successful— performance for meta-classifiers on Census (race) is not much better (Figure 3.10a) than that when guessing $\alpha = 0.5$ blindly (expected MSE 0.1).

Given the high n_{leaked} values for the regression tests, we tried using the regression meta-classifiers to distinguish between binary properties. To produce a classification between models with training distribution ratios α_0 and α_1, a regression meta-classifier $M_{\text{regression}}$'s prediction for some model m is converted to a binary outcome by simply checking which of the two considered distribution ratios the predicted ratio is closer to: $\hat{b} = \mathbb{I}\left[M_{\text{regression}}(m) \geq \frac{\alpha_0+\alpha_1}{2}\right]$. Each entry in Table 3.4 is averaged over 5 trials $\times$ 100 victim models $\times$ 11 ratios ($\{0.0, 0.1, \ldots, 0.9, 1.0\}$ for Census; 7 in the case of RSNA Bone Age). In most cases, the accuracy improves significantly over the binary classifiers, with the n_{leaked} value nearly doubling for most settings. For instance, the classification accuracy increases by $\sim 4\%$ and $\sim 15\%$ for CelebA (sex) and RSNA Bone Age respectively, corresponding to an increase of n_{leaked} by ~ 0.14 for CelebA (sex) and ~ 8.51 for RSNA Bone Age. This improvement is not surprising since the binary attack uses models only from two distributions, whereas the regression attack has models from a wide range of alpha values and thus can learn more. Figure 3.11 shows the accuracies and n_{leaked} values for using specific ratio binary classifiers compared with the improved accuracies obtained using the regression meta-classifier. The n_{leaked} values for each pair of ratios (α_0, α_1) shows how these improvements are uniform across all pairs of distributions.



(a) Binary Meta-Classifiers (b) Regression Meta-Classifier

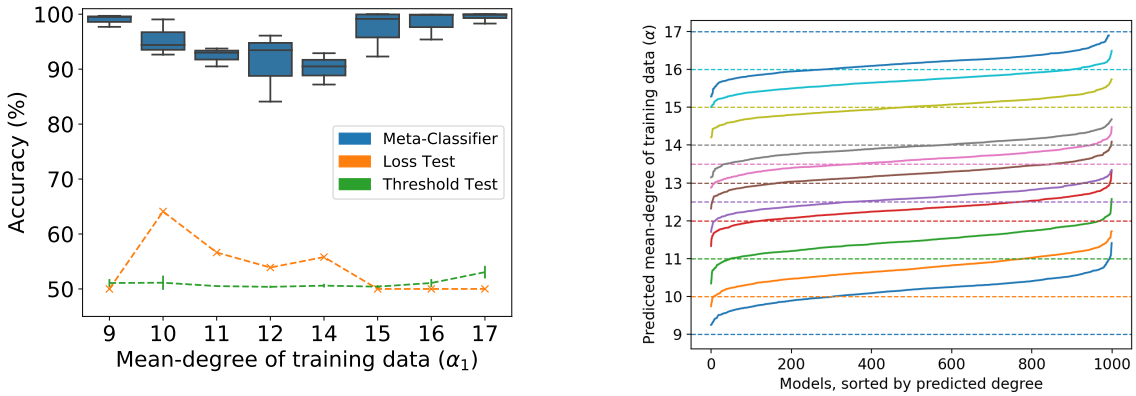
Figure 3.11: Distinguishing binary ratio properties using (a) binary classifiers and (b) regression meta-classifiers, for CelebA (age). n_{leaked} values (lower triangle) and classification accuracies (upper triangle) for distinguishing proportion of old people (ratios α_0, α_1) in training data for the CelebA dataset. n_{leaked} with binary meta-classifiers lower, especially for cases where $|\alpha_0 - \alpha_1|$ is small.

3.4.6 Graph Properties

Our experiments using both binary and regression classifiers on the graph datasets reveal surprisingly high property leakage. Observed n_{leaked} values for ogbn-arxiv are much higher (100-200 range) than was observed in the experiments on tabular and image datasets (with the exception of RSNA Bone Age). [2]

ogbn-arxiv. For the ogbn-arxiv dataset, we set $\mathcal{G}_0$ such that the graph has a mean node-degree of $\alpha_0 = 13$, and for $\mathcal{G}_1$, modify the graph to have a mean-degree α_1 as an integer in the range [9, 17]. We produce test datasets by pruning either high or low-degree nodes from the original graph to achieve a desired α_1. Like the other datasets, meta-classifier performance increases as the distributions diverge, albeit with much smaller drops. Both the Loss Test and Threshold Test fail on this dataset with n_{leaked} values below 1, compared to the meta-classifiers (Figure 3.12a) which leaks $n_{\text{leaked}} \approx 40$ in most cases. The attacks leak much more information as the degrees increase than when they decrease— n_{leaked} values are nearly double for (12, 14) than (12, 13) as the two mean node-degrees, despite having comparable distinguishing accuracies. Motivated by the success of regression attacks for ratio-based properties (§3.4.5.1), we also trained a regression variant of the meta-classifier to predict the average degree of the training graph directly. The resulting meta-classifier performs quite well (Figure 3.12b), achieving a mean squared error (MSE) loss of 0.393 ± 0.36. It generalizes well to unseen distributions, achieving an average MSE loss of 0.076 for $\alpha = 12.5$ and 13.5. A distribution inference adversary can thus be strong enough to directly predict the average node degree of the training distribution. Similar to the case of ratios, we try different combinations of mean node-degree values by setting different mean node-degrees α_0 and α_1 (Figure 3.13). As apparent, n_{leaked} has a wide spread in its values across different distributions—starting from ≈ 3 to approaching infinity, with a median of ≈ 31.

[2] For the clustering coefficients on Chord, we do not have a way to compute n_{leaked}, but also see evidence of substantial leakage.



(a) Distinguishing accuracy ($\alpha_0 = 13$) w.r.t mean degree (b) Mean node-degree α_1 predicted by the meta-classifier

Figure 3.12: Performance for (a) distinguishing between models with different mean node-degrees in training data, and (b) directly inferring the mean node-degree of the training data, for the ogbn-arxiv dataset. Each color represents the true degree (dashed lines) of the models being tested. The meta-classifier attack is remarkably successful on this dataset and further accentuates how some attacks can infer underlying properties nearly exactly on some datasets.

Chord. For the Chord dataset, we construct $\mathcal{G}_0$ to have graphs with average clustering-coefficient < 0.0061 and $\mathcal{G}_1$ with average clustering-coefficient above > 0.0071. We pick these values to minimize the overlap between the two distributions, while maintaining a decent accuracy on the original task. For the case where both the trainer's and adversary's datasets are sampled from the same pool of data, the adversary has near-perfect distinguishing accuracy, even when training the meta-classifier with ten models (and testing on 1000). However, the adversary cannot achieve the same level of performance in the absence of data overlap. Using the Loss Test yields an accuracy of 63%, while the Threshold Test and meta-classifier struggle to perform better than random (≈51%). This disparity in performance further justifies our experimental design choice to consider non-overlapping data splits—evaluating property inference attacks on models trained from the same dataset pool seems effective, but it cannot distinguish learning some unrelated property from the claimed inference.

3.4.7 Summary of Experimental Results

We summarize the distribution leakage observed for our ratio and regression based experiments in Figure 3.14. The biggest exception is for RSNA Bone Age, where we observe n_{leaked} values above 10 for the binary classifiers (Figure 3.9b) and up to 270 for the regression meta-classifier, and the graph datasets, where they are in the hundreds for ogbn-arxiv (Figure 3.13).

Peculiar Trends. We also observe some trends specific to a dataset and property, but can only speculate on their causes. For example, on the Census dataset, the adversary has notably high accuracy in differentiating between distributions when one is without any females ($\alpha_0 = 0$) or males ($\alpha_0 = 1$) with distinguishing

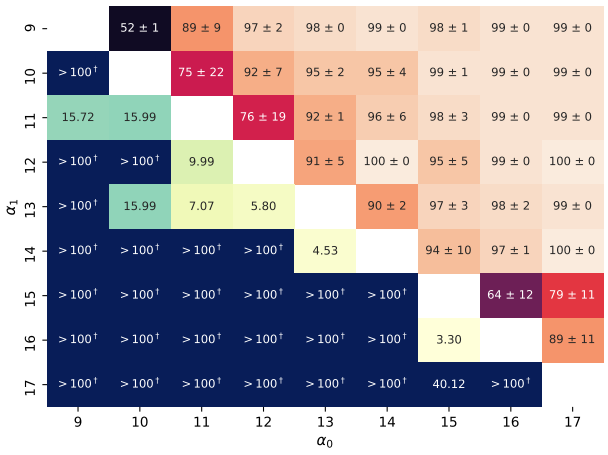
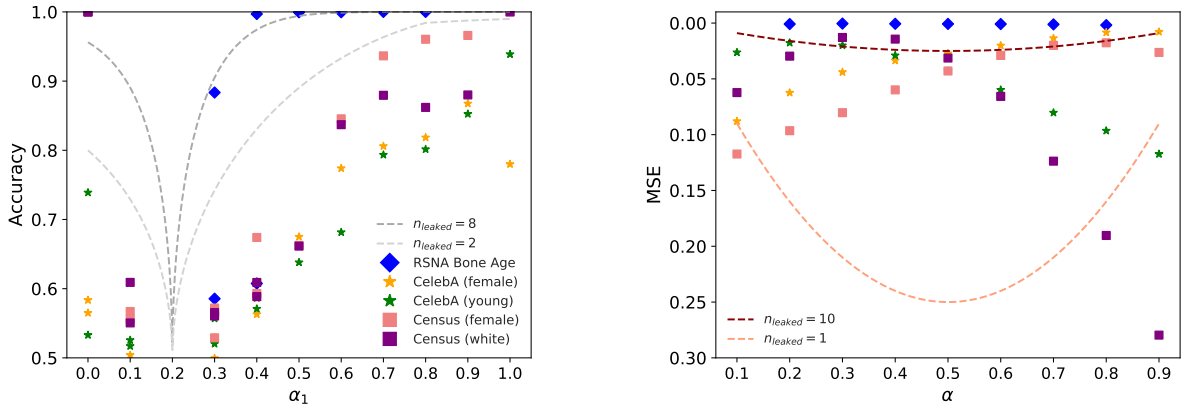


Figure 3.13: Effectiveness of meta-classifiers on ogbn-arxiv dataset. n_{leaked} values (bottom-left triangles) and mean node-degree (degrees α_0, α_1) in training data.

accuracies close to 100%, regardless of the actual proportion of females in the data. This suggests that detecting the mere presence or absence of members with a particular attribute is significantly easier than trying to deduce the exact ratio of members with that attribute, and perhaps is unsurprising here for an attribute that does impact the task predictions. Similarly, a difference in ratios of ≥ 0.3 on RSNA Bone Age (Figure 3.9b) yields at least 90% accuracy for all cases using meta-classifiers, with n_{leaked} values ≥ 7, going up to perfect distinguishing accuracy. Unlike Census, performance on CelebA at the extremes (no males or females when inferring sex ratios, and no young or old people when inferring old ratios) is far from perfect. This may be because features like race and gender in Census are directly used for model training, and thus their presence or absence would directly impact both predictions and model parameters. Whereas for CelebA, the complicated feature extractor may not embed these latent (and inherently ambiguous) properties explicitly.

Regression for Binary Classification. Comparing n_{leaked} values for the binary meta-classifiers and regression-based meta-classifiers tuned for binary classification demonstrates how additional information about the underlying ratios can have a huge impact on leakage. Having models trained on training distributions for a wide range of α can help ensure the meta-classifier actually learns to infer the underlying ratios, compared to the binary classification case where it is most likely to rely on specific signals just to distinguish between two given distributions. Although that is indeed the given task, the ability to capture the association between α and the desired predictions can, and does, help the meta-classifiers improve their performance. Note that training the regression-based meta-classifiers does not require a stronger threat model than is assumed for the binary classifier case. In both cases, the adversary needs access to a training distribution with enough samples to be able to create representative datasets for different distributions. Training the regression meta-classifier requires more computational resources (training models for multiple ratios) than is required



(a) Distinguishing accuracy with varying α_1 ($\alpha_0 = 0.2$). (b) Mean Square Error (MSE) for regression inference.

Figure 3.14: Distinguishing accuracy (a) and MSE (b) for inference attacks with varying α_1 on the horizontal axis. The plotted results are for the most effective attacks from the experiments described in §3.4.3. The curves in (a) show the comparable distinguishing accuracy for $n_{\text{leaked}} = 2$ (indicating that most of the attacks are comparable to leaking fewer than two samples from the training distribution) and $n_{\text{leaked}} = 8$, showing that a few of the attacks on the RSNA Bone Age dataset (and extreme attacks on Census for the race attribute) do leak a substantial amount of information. Similar trends hold for regression, with n_{leaked} somewhere between 1 and 10 for most cases (b). The highest leakages we observed are for the graph datasets, not shown in this figure.

to train the binary meta-classifier (training models only for two ratios), but does not otherwise require a stronger adversary.

3.5 Results with KL Divergence Attack

Our KL Divergence Attack (KL) outperforms all previous black-box attacks by huge margins (§3.5.1). Even more interestingly, the KL Divergence Attack, with only black-box access, outperforms Permutation Invariant Networks (PIN) by a large margin in nearly all settings. We study trends between the correlation of the task and property, and its impact on inference risk (§3.5.2).

3.5.1 Results

Table 3.5 summarizes the results of our distribution inference experiments. For each experiment, we report mean distinguishing accuracies between two distributions as well as the mean distinguishing accuracy across a set of different distributions, as detailed in Table 3.5. Although our regression-based adversaries are observed to be strictly more powerful (§3.4.5.1), extending KL to utilize such regression-based adversaries is non-trivial, and we leave it for future work.

Trends across datasets. Inference leakage varies significantly across different datasets, with very little leakage for most cases in Texas-100X, substantial leakage for Census19, and exceptionally high leakage for

Dataset	Task/Property	Distinguishing accuracy (n_{leaked}) for $\alpha_1 = 0.2$				Mean distinguishing accuracy (n_{leaked})			
		Black-Box			White-Box	Black-Box			White-Box
		TT	ZTO [371]	KL	PIN [95]	TT	ZTO [371]	KL	PIN [95]
Census19	Income/Females	50.0 (<0.1)	53.4 (<0.1)	**89.8 (2.1)**	78.6 (0.8)	61.3 (0.9)	54.4 (<0.1)	**82.5 (4.2)**	81.0 (3.5)
	Income/Whites	53.2 (<0.1)	52.6 (<0.1)	**92.4 (2.7)**	74.2 (0.6)	59.4 (0.7)	54.9 (<0.1)	**83.7 (3.3)**	75.4 (1.1)
Texas-100X	Procedure/Females	50.0 (<0.1)	50.0 (<0.1)	**89.3 (2.0)**	50.0 (<0.1)	51.2 (<0.1)	51.6 (<0.1)	**82.5 (3.8)**	51.3 (<0.1)
	Procedure/Whites	50.9 (<0.1)	50.0 (<0.1)	**86.8 (1.7)**	50.0 (<0.1)	52.4 (<0.1)	50.1 (<0.1)	**81.6 (3.7)**	50.5 (<0.1)
	Procedure/Hispanic	50.0 (<0.1)	50.0 (<0.1)	**78.4 (0.8)**	50.0 (<0.1)	50.0 (<0.1)	50.0 (<0.1)	**82.4 (3.8)**	50.1 (<0.1)
CelebA	Mouth Open/Wavy	52.0 (<0.1)	51.8 (<0.1)	56.8 (<0.1)	**92.0 (2.6)**	50.6 (<0.1)	52.3 (<0.1)	62.1 (<0.1)	**86.1 (2.4)**
	Smile/Females	54.4 (<0.1)	57.6 (<0.1)	**89.6 (2.1)**	57.6 (<0.1)	55.4 (0.1)	60.9 (0.2)	**85.3 (3.2)**	68.4 (0.5)
	Gender/Young	50.3 (<0.1)	52.6 (<0.1)	**86.4 (1.6)**	81.0 (1.0)	52.9 (<0.1)	55.5 (0.1)	**86.3 (2.5)**	81.2 (1.5)
	Mouth Open/Cheekbones	50.0 (<0.1)	50.0 (<0.1)	84.6 (1.4)	**95.8 (3.9)**	50.1 (<0.1)	56.2 (0.1)	76.7 (1.4)	**88.6 (3.0)**
RSNA Bone Age	Age/Females	90.0 (2.2)	95.4 (3.7)	**99.9 (20.1)**	99.4 (7.9)	64.0 (0.5)	77.9 (1.6)	**94.5 (12.1)**	95.2 (10.2)
	Females/Age	95.7 (3.8)	99.4 (7.9)	**99.9 (20.1)**	66.0 (0.2)	68.5 (1.0)	78.5 (3.3)	**99.8 (22.6)**	75.2 (8.4)
ogbn-arxiv	Node classification/ Mean Degree	50.0 (<0.1)	50.0 (<0.1)	**99.9 (58.5)**	87.4 (5.1)	50.1 (<0.1)	55.4 (6.2)	**92.6 (182.5)**	71.9 (11.7)

Table 3.5: Effectiveness of inference attacks. We show results for our KL Divergence Attack (KL) and three other attacks: Threshold Test (TT), ZTO [371], and Permutation Invariant Networks (PIN) [95]. For the classifiers, the first set of results shows the attack's ability to distinguish between models trained on training sets where the proportion of the property is either $\alpha_0 = 0.5$ or $\alpha_1 = 0.2$ as an accuracy percentage, with corresponding n_{leaked} values in parentheses. The second set of results shows the mean distinguishing accuracies (%) (with corresponding n_{leaked} values) between $\alpha_0 = 0.5$ and a set of varying α_1 values $(0.0, 0.1, 0.2, 0.3, 0.4, 0.6, 0.7, 0.8, 0.9, 0.1)$. For the graph datasets used for ogbn-arxiv, for the first set of results we use $\alpha_0 = 13$ are $\alpha_1 = 10$ as the two distributions; for the second set, we vary the mean node degree as the property, setting $\alpha_0 = 13$ and varying α_1 in $[9, 10, 11, 12, 14, 15, 16, 17]$, and report the mean distinguishing accuracy (with mean n_{leaked} value). For all of the results, for each α_1 value, we compute the median over five trials. Mean accuracy (and n_{leaked}) is then computed over the mean of these values for all α_1 values. For each setting, results for the most effective attack are bolded.

the graph-based ogbn-arxiv dataset. The lack of virtually any inference risk in Texas-100X is surprising, as the features contain the property label, and data splits are processed per hospital during generation, making the victim and adversary distributions highly similar. This difference in inference risk between Census19 and Texas-100X, despite both being tabular datasets, reveals how just the nature of data (tabular, images) does not by itself determine inference risk and risk can vary unpredictably (at least based on current understanding) with aspects of the data. As we previously observed (Figure 3.14), leakage is quite high for RSNA Bone Age. Our new improved attacks identify vulnerable datasets, such as Census19, that would have been considered low leakage risks using previous state-of-the-art attacks.

Comparing black-box attacks. The KL Divergence Attack outperforms Threshold Test (TT) and the black-box attack by Zhang et al. [371] (which we refer to as ZTO) in all cases with large margins. Across all of the settings, TT and ZTO rarely achieve distinguishing accuracies above 75% i.e., n_{leaked} above 1.0 (indicating that the observed leakage is less than what an adversary would learn by sampling a single record from the training distribution), whereas the KL Divergence Attack produces meaningful leakage for all of the datasets. The superiority of the KL Divergence Attack can be attributed to the use of pairs of local models and their trends (which grow in the order $\binom{n}{2}$ for n models), as opposed to using information from models in isolation in the other attacks.

Dataset/Task	Number of Shadow Models					
	5	10	25	50	100	400
Census19 (Sex)	73.0 (3.0)	76.5 (3.3)	81.3 (4.0)	82.5 (4.2)	86.4 (5.1)	89.7 (6.6)
Census19 (Race)	77.2 (2.6)	79.3 (2.9)	81.3 (3.1)	83.7 (3.3)	84.2 (3.3)	84.7 (3.4)
RSNA Bone Age (Age)	97.3 (18.3)	98.3 (19.0)	99.3 (21.3)	99.7 (22.6)	99.7 (22.6)	99.8 (22.8)
CelebA (Sex)	73.9 (1.1)	78.6 (1.7)	80.9 (2.4)	85.3 (3.2)	86.9 (3.9)	89.2 (5.1)

Table 3.6: Impact of varying the number of shadow models used by the adversary per distribution to launch its attacks. Values are mean distinguishing accuracies (%) (with mean n_{leaked} in parentheses) for KL Divergence Attack (computed as described in Table 3.5). Even with only 5 models, the adversary is able to achieve considerable inference leakage.

Number of shadow models. The black-box attacks use 50 shadow models per training distribution. We vary this number to 1) get an empirical lower bound on the number of shadow models required to achieve non-trivial leakage, and 2) study increase in information leakage with an increase in shadow models. Leakage is significant with only five shadow models per distribution in most cases, and improves with more local shadow models (Table 3.6).

White-box attacks. The black-box KL Divergence Attack performs surprisingly well despite the weaker threat model, outperforming the best white-box attack in nearly all experimental settings. Since an adversary in the white-box setting has access to more information than just the data and model predictions, it should be at least as powerful as a black-box adversary. We attribute the relative ineffectiveness of the white-box attacks to two main reasons. First, in Permutation Invariant Networks, model parameters are directly used as features for the meta-classifier, unlike comparisons in model prediction distributions in KL Divergence Attack. Secondly, white-box attacks have a larger feature space and learning meta-classifiers additionally requires learning to recognize relevant patterns in model parameters, a huge and complex data distribution. The black-box attacks, on the other hand, are agnostic to parameters in the victim model and thus much easier to scale, resulting in better performance.

3.5.2 Correlation

The impact of correlation between the underlying task of a model and the property of its training distribution being inferred has been touched upon briefly in the literature [371], but not studied extensively. Intuition suggests there should be some positive relationship between inference risk with increasing task-property correlation, but prior studies do not evaluate inference risk across a range of property-task correlations. We carefully pick pairs of properties and tasks for the CelebA dataset, such that there is a good range of correlations.

We conduct experiments with property correlations of ≈ 0 (Mouth Slightly Open–Wavy Hair), ≈ 0.14 (Smiling–Female), ≈ 0.28 (Female–Young), and ≈ 0.42 (Mouth Slightly Open–High Cheekbones). Across

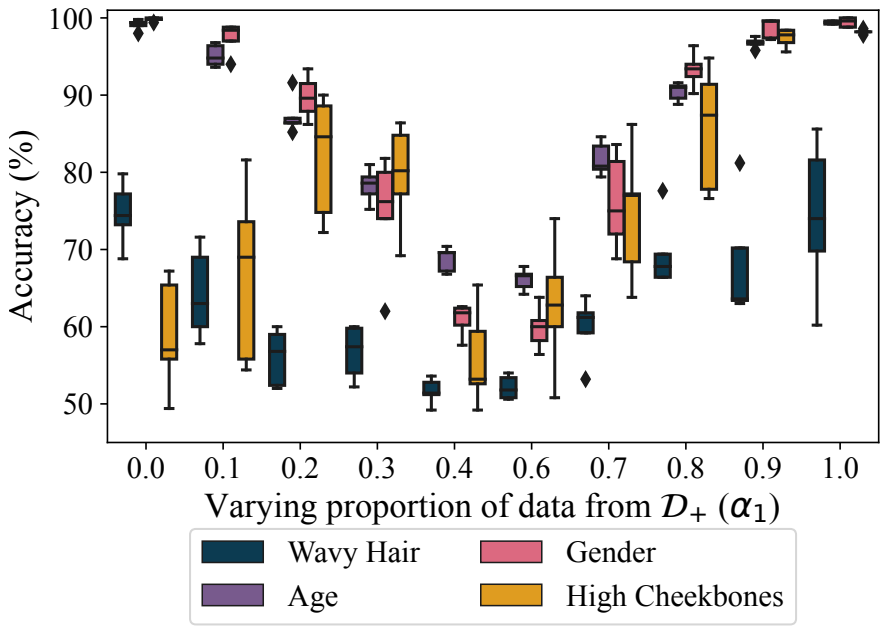


Figure 3.15: Distinguishing accuracy for different task-property pairs for CelebA with varying correlation, for KL Divergence Attack.

this range of correlations, mean distinguishing accuracies (n_{leaked} values in parantheses) are 62.1% (<0.1), 85.3% (3.2), 86.3% (2.5), 76.7% (1.4) as correlation values increase. The lack of any clear trend between correlation and inference risk is consistent with observations in the literature around task-property correlation and inference risk [371]. As observed, inference risk is non-trivial as long as the task-property correlation is non-zero. While the case of non-zero inference risk is obvious (with zero correlation, loss optimization would not use the property as an indicative feature), changes in inference risk with varying correlation values may be tied to observed correlation versus actual causality, and methods for causal learning may help alleviate this inference risk [307].

We perform similar analyses for the RSNA Bone Age dataset, where we flip the property and task. In this case, the correlation between the task and property remains the same, thus helping identify potential changes in inference leakage arising purely from the choice of the property itself. While switching from Age–Females to Females–Age, we observe a huge bump in mean distinguishing accuracies (n_{leaked} values in parentheses): from 94.5% (12.1) to 99.8% (22.6). Although the choice of property and task are expected to impact inference risk, our results suggest that this choice may be more relevant to evaluating inference risk than property-task correlation itself.

3.5.3 Fixing Task-Label Ratios

In scenarios where the model trainer is aware of the task-label balance in $\mathcal{D}$, they may try to adjust $\mathcal{G}_0(\mathcal{D})$ or $\mathcal{G}_1(\mathcal{D})$ in a way that preserves that task-label balance. For instance, a system for predicting duration of

hospital stays might want a balance between short and long-stay patients, irrespective of what the gender imbalance in its collected data is, and might subsequently adjust data sampling to maintain a certain class label ratio. Instead of directly changing the ratio to some other value α_1, which may distort the label distribution, the trainer may constrain the sampled dataset such that the ratio of task labels (say, y) is preserved (e.g., 0.3 males, 0.7 females).

For this experiments, we achieve this in two steps. First, we use the task label-ratios and the size of the final dataset (say, m) to compute the number of samples per task-label ($y_i \cdot m$) required to maintain the task-label ratio. Then, we re-sample data per task-label to achieve a desired property ratio α_1. This data is then re-combined, thus achieving the desired ratio α_1 while maintaining task label-ratios.

We observe mixed results for inference risk changes under this configuration. n_{leaked} values increase by around 2.6 on an average for Census19, while dropping to near-zero inference risk for datasets like Texas-100X. Adjusting task-label ratios in this manner can thus have a positive or negative impact, depending on its relationship with the task labels and how it impacts the resulting distributions. The impact of this processing is not very predictable—while it eliminates inference risk in some cases (like CelebA with Gender–Young), it actually worsens performance in cases such as RSNA Bone Age (Gender–Age).

3.6 Impact of Adversary's Knowledge

Research on inference privacy typically considers threat models with one of two simplistic adversarial assumptions: white-box settings, where the adversary has full access to the model; and black-box settings, where the adversary has only API access to the model but receives full confidence vectors for each prediction and has complete knowledge of aspects of the training process and model architecture. The specific information available to an adversary in the black-box setting, however, can understandably have a significant impact on inference risk. For instance, access to labeled data (for attacks) with prediction probabilities is often implicit, as is the use of the same model architectures and feature extractors between the victim and adversary. We study the impact of these common assumptions, and how relaxing them impacts inference risk. We measure impact on risk when the victim and adversary use different model architectures (§3.6.1), do not share feature extractors (§3.6.2), and when the available model API only provides label predictions (§3.6.3). Inference risk is somewhat robust to differences in model architectures, as long as the victim and adversary's models have similar learning capacity. The absence of shared feature extractors reduces inference risk significantly, but we find attacks can still succeed when only label predictions are available.

3.6.1 Model Architecture

In the white-box setting an adversary can directly observe the target model's architecture, but in black-box settings it is unrealistic to assume the adversary knows the target model architecture. Likely model

Victim Model	Adversary Model			
	RF	LR	MLP_2	MLP_3
Random forest (RF)	95.1 (12.0)	78.9 (1.7)	86.7 (5.4)	85.6 (4.9)
Linear regression (LR)	93.2 (13.5)	100.0 (25.9)	76.4 (3.7)	80.8 (5.4)
Two-layer perceptron (MLP_2)	69.7 (0.9)	56.6 (0.3)	82.5 (4.2)	82.7 (4.3)
Three-layer perceptron (MLP_3)	69.3 (0.8)	56.3 (0.3)	82.2 (4.0)	81.1 (3.8)

Table 3.7: Variation by model type. Each value is the observed mean distinguishing accuracy (%) (with mean n_{leaked} in parentheses; measured as described in Table 3.5) of the KL attack for Census19 (Sex), for different combinations of model types for victim and adversary.

architectures may be limited in certain domains like image data, where the victim is likely to use a popular model architecture such as DenseNet [122] or a convolutional neural network, at the very least. But a variety of models like random forests, support vector machines, and clustering-based classifiers can be used for tabular data and may even be picked by model trainers via automated tools [88].

Differences in victim and adversary architectures have not been previously explored, except by Mahloujifar et al. [199] for poisoning-based adversaries. In their setting, the victim and adversary can have different model architectures—the adversary uses logistic regression while the victim can use a variety of different feed-forward neural networks. They note a drop in inference risk with an increase in victim model complexity. Thus, it is unclear whether these trends are specific to the model architecture. Additionally, the adversary's model architecture is kept the same, so they did not explore the potential for higher inference risk with better local models.

To identify trends in inference risk with differences between architectures, we train multiple models with different architectures for both the victim and adversary. For Census (Gender), we try all possible combinations out of linear regression (LR), multi-layer perceptrons with two and three layers (MLP_2, MLP_3), and a random forest classifier (RF). We also consider using a two-layer perceptron (MLP_2) and a support vector machine (SVM) for the case of RSNA Bone Age (Gender). For this experiment and the rest of this section, we report results with KL Divergence Attack.

We observe several interesting trends for Census19 while varying model types for the victim and adversary (Table 3.7). Inference risk is significantly higher when the adversary uses models with learning capacity similar to the victim, like both using one of (MLP_2, MLP_3) or (RF, MLP). Concretely, mean distinguishing accuracy is 86.9% (mean n_{leaked} =8.7) when learning capacities match, as opposed to 72.7% (mean n_{leaked} =2.7) when learning capacities do not match.

Interestingly, we also observe a sharp increase in inference risk when the victim uses models with low capacity, like linear regression and random forest instead of multi-layer perceptrons. For example, mean distinguishing accuracy is 72.6% (mean n_{leaked} =2.3) when victim models have high learning capacity (MLP_2, MLP_3), but

increases to 87.1% (mean n_{leaked} =9.1) when the victim models have low learning capacity (RF, LR). These trends hint at possible connections between distribution inference risk and model learning capacity.

3.6.2 Feature Extractors

When dealing with high-dimensionality datasets and a scarcity of data, it is common to use techniques such as transfer learning [337, 347] to boost model performance with reduced data and computational requirements. Using a pretrained model for feature extraction should intuitively limit distribution-related privacy leakage, since there are fewer trainable parameters that can potentially contain revealing information. At the same time, fewer parameters reduce the adversary's search space over models, making it easier to launch attacks. Even in a black-box setting, the adversary may be able to use the same feature extractor as the victim, either as a result of the adversary snooping and gaining information, or just by assuming the use of popular pretrained models (like BERT [154]). While this setting has been previously explored [16, 95, 293], the exact effect of a mismatch in extraction models between the victim and adversary is not well understood.

For RSNA Bone Age (Sex), we consider two configurations: one where the victim and adversary use the same feature extractor, and another where the victim trains DenseNet [122] models from scratch (CNN). For the first setting, we explore an SVM (FE+SVM) and a two-layer perceptron (FE+MLP_2). There is a considerable drop in distinguishing accuracies (from 96.7% to 91.2% i.e., n_{leaked} from 16 to 6) when the victim and adversary no longer share feature extractors (Table 3.8). For the settings where they do, we observe leakage to be highest for similar model architectures, and note a sharp increase when the victim uses an SVM. Nonetheless, inference risk stays sufficiently high. Interestingly, for the case where feature extractors are not shared, using a lower learning-complexity model (FE+SVM) seems to lead to higher leakage, than FE+MLP_2. While leakage is high in both cases, the increase can be explained by the chances of adversary's local models overfitting being less than that with an MLP.

For CelebA (Sex), we explore a setting where the adversary utilizes a feature extractor to train its models, while the victim trains CNNs from scratch. This setup represents a resource-constrained adversary who uses pretrained models to lower computational and data requirements. We observe a similar diminishing of inference risk when a shared feature extractor is not available to the adversary, consistent with the RSNA Bone Age results. Compared to the scenario where the adversary uses the same model architecture as the victim without any pretrained feature extractors, mean distinguishing accuracy drops from 85.3% to 71.0% (i.e., n_{leaked} from 3.2 to 0.5).

3.6.3 Label-Only Access

Most black-box attacks in the literature related to distribution inference assume access to prediction confidence vectors. This is not an unreasonable assumption—many APIs return prediction scores, especially for top-k

Victim Model	**Adversary Model**	
	FE+MLP_2	FE+SVM
Feature extractor, perceptron (FE+MLP_2)	94.5 (12.1)	93.0 (9.0)
Feature extractor, SVM (FE+SVM)	99.5 (21.2)	99.6 (21.7)
DenseNet (CNN)	88.0 (3.4)	94.4 (8.5)

Table 3.8: Mean distinguishing accuracies (%) (with mean n_{leaked} values in parentheses) for RSNA Bone Age (Sex), for different combinations of model types for victim and adversary (as computed in Table 3.5).

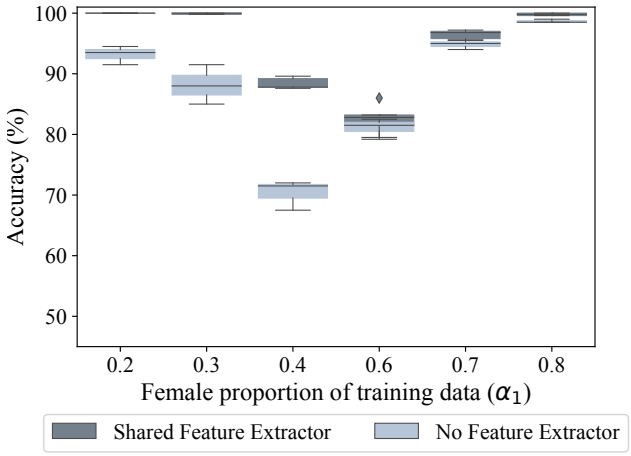


Figure 3.16: Distinguishing accuracy for for the KL Divergence Attack for RSNA Bone Age (Sex), when the adversary uses the same feature extractor as the victim, and when the victim does not use or share any pretrained feature extractor. While there is an obvious drop in performance, inference risk still stays high.

classes (for example Google Vision API[3] and ClarifAI Prediction API[4] return scaled confidence scores for the top 10 or 20 classes, respectively). It is unclear, however, what kind of performance drops to expect for distribution inference attacks in settings where the model's API only returns a label. The only previous works to explore distribution inference in the label-only setting are in the context of group distribution shift auditing [147], and active adversaries with poisoning capabilities [199].

With some straightforward modifications, our KL Divergence Attack can be launched with access to just label predictions, with negligible drops in inference leakage in most cases. The attack requires prediction confidence scores to compute the KL-divergence values. However, these scores are absent in the label-only setting, and the labels effectively correspond to confidence values of 0 and 1. This makes the KL computations in Equation (3.46) invalid, since the log of 0 or 1/0 is undefined. To tackle this, we replace 0/1 labels with confidence scores ϵ and $1-\epsilon$ respectively for some small value ϵ (set to 0.01 in our experiments).

We observe mixed trends across datasets and attacks. For instance, switching to the label-only setting has little impact in the case of Census19, while mean distinguishing accuracies drop by more than 8% (n_{leaked} drops by more than half) for CelebA (Table 3.9). However, the drop in performance for CelebA is not

[3] https://cloud.google.com/vision
[4] https://www.clarifai.com/products/armada-ml-prediction

Dataset/Task	Confidence Scores	Prediction Label	
		Direct	Sampling
Census19 (Sex)	82.5 (4.2)	77.3 (3.3)	80.5 (3.7)
CelebA (Sex)	85.3 (3.2)	77.8 (1.4)	79.3 (1.6)
RSNA Bone Age (Age)	99.8 (22.6)	96.3 (12.7)	97.1 (13.3)

Table 3.9: Effectiveness of label-only attacks. Each value is the observed mean distinguishing accuracy (%) (with mean n_{leaked} in parentheses) for KL (as computed in Table 3.5). The label-only setting leaks less information, but the attacks still are effective even when confidence scores are unavailable. 'Direct' uses a single query, while 'Sampling' uses 10 samples around each test point.

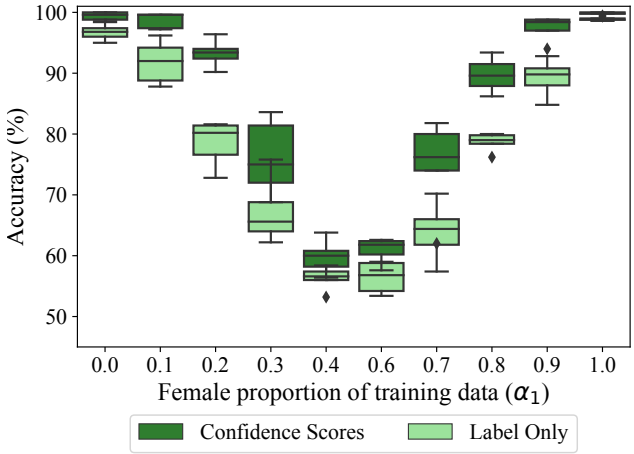


Figure 3.17: Comparing the distinguishing accuracy for the KL Divergence Attack for CelebA (Sex), when the target model returns prediction confidence scores and when it returns only prediction labels. Performance drops most for certain ratios like 0.2 and 0.8, but remains high and roughly the same for more extreme ratios like 0.0, 0.1, and 1.0.

uniform across all ratios. Inference risk is still quite high for many values of α (Figure 3.17). Similar trends hold for RSNA Bone Age, where distinguishing accuracy is $> 75\%$ for all ratios. We also experiment with using probabilistic sampling to extract more information. For each datapoint, we sample k random points in its neighborhood by adding random noise from $\mathcal{N}(0, \sigma^2)$ to each feature, and average the generated label predictions to estimate confidence scores, similar to Jayaraman et al. [138]. We observe slight improvements in attack performance from the sampling, at the cost of additional queries.

3.7 Defenses

Several defenses against distribution inference have been proposed, but most of them (except differential privacy [95], which has shortcomings as we discuss in §3.7.1) have not been actually evaluated. Like most defenses designed to limit privacy leakage, these defenses involve adding noise in some parts of the training process. This can include the data itself [16] or model parameters [95, 214]. One notable exception is work by Hartmann et al. [109], where the authors study causes of leakage in distribution inference attacks, and evaluate mitigation strategies based on causal learning (IRM [13]), correcting inductive biases, and

increasing the amount of training data, for synthetic datasets. Recent attempts at empirical defenses against distribution inference [235] require training multiple reference models locally, along with prior knowledge of the property that an adversary may target. Such defenses can also be susceptible to adaptive attacks [289]. We evaluate some of these noise-based defenses in §3.7.1, and find that they seem unlikely to successfully mitigate distribution inference risks. Our exploration of inference risk with model generalization reveals interesting trends and a potential trade-off between learning and inference risk (§3.7.2). In §3.7.3, we introduce and evaluate a simple defense based on data re-sampling, which can prevent distribution inference in settings where the model trainer knows which distributional property to hide.

Prior Work. Unlike membership inference where differentially private training can provide a guaranteed bound on inference risk, there are no defenses against distribution inference from trained models with theoretical guarantees. Chen and Ohrimenko [55] recently proposed a defense mechanism that builds upon formal notions of distributional privacy [372] to protect against distribution inference attacks on statistical queries. This is the first known theoretically-grounded defense against distribution inference, but it does not apply to protecting machine-learning models.

The only previous defense that has demonstrated meaningful protection against distribution inference attacks on machine learning models (apart from Inf2Guard [235]) is NoSnoop [195], proposed for a collaborative learning setting. In their threat model, the adversary seeks to infer sensitive information about exact training batches and has access to intermediate model losses from clients. The defense works utilize a discriminator-generator setup, where gradient updates are used to minimize property leakage while preserving task-based utility. Although this defense is highly effective, it defends against a very narrow type of configuration, including properties limited to the presence/absence of sensitive data.

Other proposed defenses include removing sensitive attributes from features [371], using node-multiplicative transforms, or encoding arbitrary information into the models [95]. Since black-box attacks only utilize relationships between inputs and model outputs, they are unaffected by such changes as long as model functionality remains unaffected. Further, these defenses seem unlikely to diminish black-box attacks, which our experiments have shown to be more effective than known white-box attacks, hence we do not evaluate them here.

3.7.1 Noise-Based Defenses

Several proposed defenses against distribution inference involve adding noise in various ways—differentially privacy training incorporates crafted noise in the training process and label poisoning adds noise to the training data. We also consider using adversarial training, which augments training with adversarial perturbations.

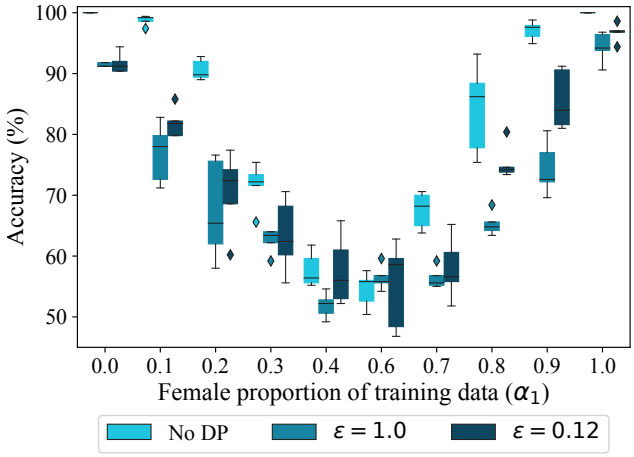


Figure 3.18: Distinguishing accuracy for different for Census19 (Sex), using KL Divergence Attack. Attack accuracy drops with stronger DP guarantees (decreasing privacy budget ϵ).

Differentially Private Training. Differential privacy (DP) is a formal privacy notion that provides theoretical guarantees that bound an adversary's ability to distinguish between neighboring input datasets from the output of a computation. Differential privacy can provide theoretical bounds limiting membership inference. Evaluations by Ateniese et al. [16] suggest differentially private training is not an effective defense against distribution inference attacks. However, their experiments used a setup with some overlap between the victim's and adversary's data, so it is possible the observed lack of protection is related to the overlapping data available to the adversary. Although differential privacy in itself does not guarantee protection against distribution inference, evaluating risk for models trained with these guarantees can help better understand how such noise-based mechanisms can affect inference risk, and assess the vulnerability of models meant to provide membership privacy. Empirical evidence can thus be beneficial and more concrete than relying on pure intuition (or argumentative reasoning about why a defense may or may not work).

We use DP-SGD [1] to train victim models with Reńyi Differential Privacy [221], with privacy loss budgets of $\epsilon = 1.0$ and $\epsilon = 0.12$, with $\delta = 4.9 \times 10^{-6}$. We evaluate this defense on Census19, since it is the only tabular dataset with non-trivial inference leakage. We observe a drop in distinguishing accuracies, but inference risk stays high for ratios further away from $\alpha_0 = 0.5$ (Figure 3.18).

The decrease in effectiveness may not be solely due to differential privacy. It could also be because the model does not learn the distribution well enough, leading it to not reveal it. Another possibility is that when the victim uses DP, it creates arbitrary differences, causing a mismatch between the victim's models trained using DP-SGD and the adversary's shadow models trained without privacy noise. Inspection of task accuracy for the differential-privacy models suggests lower learning effectiveness as one potential factor (Table 3.12). To test whether the decrease in prediction accuracy is mainly due to arbitrary differences in the models, we evaluate results for the setting where the adversary also trains its models using DP-SGD with the same

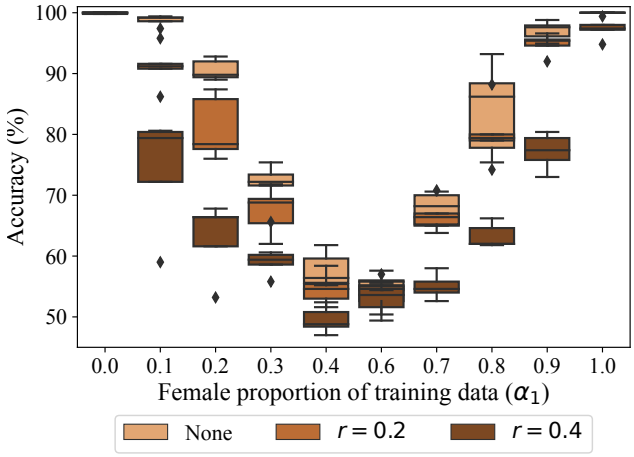


Figure 3.19: Distinguishing accuracy for different for Census19 (Sex), for varying levels of label poisoning. Inference risk drops considerably with increasing levels of label poisoning, but is also followed with non-trivial drops in task accuracies.

privacy loss budget. Compared to an adversary that does not use DP, there is a clear increase in inference risk—mean distinguishing accuracy increases to 86.4% (n_{leaked} =2.9) for $\epsilon = 1.0$, and 91.5% (n_{leaked} =4.8) for $\epsilon = 0.12$ (compared to 82.5%, i.e., n_{leaked} =4.2 without any DP).

Assuming adversary's knowledge of the use of differentially-private training and the specific privacy loss budget is not a far-fetched assumption. Organizations that release differentially private models often document their exact levels of privacy budget [3, 304]. An adversary in such scenarios can thus train its models with the same privacy parameters.

Label Poisoning. Ganju et al. [95] proposed to mitigate distribution inference by adding noise to the training data via label poisoning. The underlying idea is to perturb the training data in a way that will alter the model parameters and make the adversary's task harder. Although changing data labels can be detrimental to the model's task performance, a model trainer may be able to find an acceptable trade-off between accuracy and inference risk. For a given noise ratio r, the defense comprises randomly flipping task labels for r fraction of the training data. We evaluate this defense for CelebA (Male) and RSNA Bone Age (Age) with a label noise ratio of 0.2, and Census19 (Gender) for label noise ratios 0.2 and 0.4. As expected, this defense harms task performance (Table 3.12), reducing task accuracy: by $\sim 1-2\%$ for $r = 0.2$ for all three datasets, and $\sim 3\%$ for $r = 0.4$ off Census19. Average inference risk drops for Census19, but remains is still quite high for ratios like $\alpha_1 < 0.2$ and $\alpha_1 > 0.8$, as shown in Figure 3.19. It may be possible to find a desirable tradeoff for a simple task like Census19, but this approach is not effective for more complex tasks. For instance, using a label noise ratio of 0.4 in CelebA completely destroys task performance, reducing the classifier to only slightly better than random guessing.

Adversarial Training. Adversarial training [196] involves using a training loss function that encourages

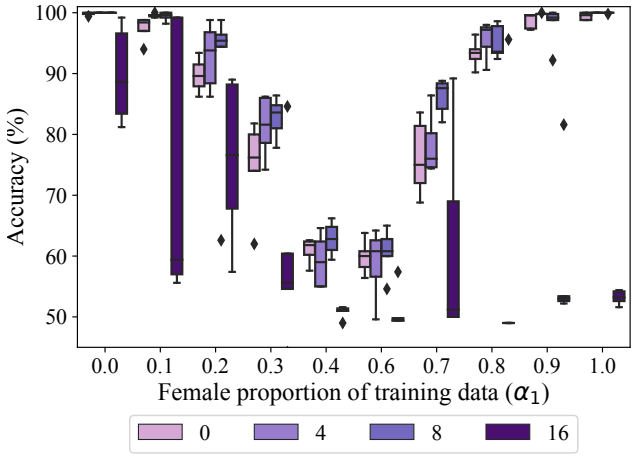


Figure 3.20: Distinguishing accuracy for different using KL, for varying levels of adversarial robustness ϵ (/255) in L_∞ norm, for CelebA (Sex). Inference risk lowers with increasing robustness.

Dataset/Task	**Adversarial Training (ϵ)**			
	0/255	4/255	8/255	16/255
CelebA (Sex)	85.3 (3.2)	86.8 (5.3)	88.3 (5.2)	58.9 (0.2)
CelebA (Age)	86.3 (2.5)	90.0 (5.5)	88.9 (6.4)	83.6 (2.0)

Table 3.10: Impact of adversarial training. Values are mean distinguishing accuracies (%) (with mean n_{leaked}; as computed in Table 3.5) for KL on models trained with adversarial robustness, with varying norms ϵ (/255) of perturbation budget (L_∞ norm).

the model to learn features that are robust to perturbations in the input, and produces models that are less prone to overfitting dataset-specific patterns [127]. This can be especially useful when the data includes properties that are not correlated with the task, and a model should not capture signals related to the irrelevant property. A model trained with adversarial robustness objective, using this reasoning, should be less susceptible to distribution inference. To test this hypothesis and explore the impact of training for robustness. We train adversarially-robust models for varying L_∞ norms for the Gender and Age properties on CelebA, since the other datasets are either tabular or do not contain sufficient samples for adversarial training with acceptable performance. Figure 3.20 shows distinguishing accuracies for varying settings for the perturbation strength used in adversarial training. Training for adversarial robustness, as documented in the literature, leads to drops in task accuracy.

We observe very interesting trends with respect to inference risk. Risk increases with increasing perturbation strength (ϵ) until 8/255, and then drops to near-zero (Figure 3.20).

Since adversarial training helps models remove focus from spurious correlations, it is naturally aligned with causal inference [373]. This results in these models using more signals relevant to the inferred property (such as Age or Sex) since they are linked to the task at hand. However, as this perturbation norm increases, task accuracy drops accordingly, thus leading to lower inference risk since the model itself performs poorly

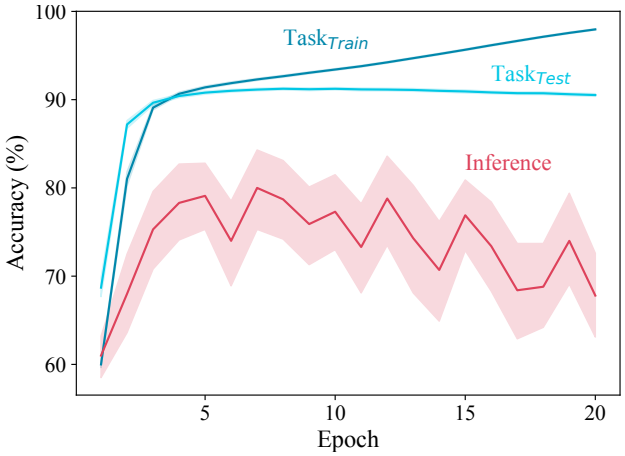


Figure 3.21: Mean distinguishing accuracy (as computed in Table 3.5) of the KL Divergence Attack on CelebA (Sex), for varying number of training epochs for victim models. Shaded regions correspond to error bars. Distribution inference risk increases as the model trains, and then starts to decrease as the model starts to overfit.

at learning causal connections, like the one between the property being inferred and the given task. One notable exception here is $\epsilon = 8/255$ for CelebA (Age), where inference risk seems to slightly increase. One possible explanation is the stronger age (property) and sex (task) relationship in this case, leading to the causal relationship-accuracy tradeoff leaning in favor of the former, in terms of inference risk.

3.7.2 Generalization

The defenses discussed in §3.7.1 have one thing in common: they nearly always lead to non-trivial drops in task performance. This is not only unacceptable for most deployments, but raises the question of whether the defenses are doing anything useful or just reducing distribution inference by producing models that learn the underlying distribution less well. Concretely, we observe a positive correlation between model task accuracy and mean n_{leaked} values (Pearson's correlation coefficient > 0.55). A model with poor task performance possibly fails to learn useful signals from the training distribution, and is thus cannot leak properties it has not learned; while good performance means a model has learned the distribution well and is prone to more leakage.

To investigate these correlations, we inspect trends between inference risk and generalization across training epochs. For this experiment, we train the models longer than the other experiments (which were optimally selected for best generalization using validation data), allowing us to better study trends between overfitting and inference risk. We observe interesting trends in inference risk with model training. In most cases, inference risk is high after even one epoch of model training (Figure 3.21). This is especially surprising because the model takes a few epochs to get good performance on the task itself, but shows that the model is learning and exposing aspects of the distribution even early in its training.

These trends clearly suggest that model under-training is not a feasible defense. Training beyond minimum generalization gap does lead to significantly reduced distribution inference risk. However, this region of the training corresponds to overfitting, which is known to be positively correlated with increased risk to membership inference [355]. Thus, a model trainer that is willing to overfit its models to avoid distribution inference adversaries would risk making the model more vulnerable to membership inference.

3.7.3 Re-Sampling Data

If the victim is aware of the property that an adversary might target, or only has a few known properties of the distribution that it wants to protect, the easiest mitigation is to modify the training distribution (or the sampling mechanism) such that the property is no longer present for the training dataset. Knowing the particular property to hide is a plausible assumption that is often assumed in work on distribution inference defenses. For instance, Chen and Ohrimenko [55] propose a theoretically-grounded defense that builds upon the distributional privacy framework [152] and modifies feature values to provide privacy guarantees against distribution inference adversaries.

Zhou et al. [376] propose over-sampling to reduce inference risk, but do so by adding new samples to their training data. Although this defense eliminates distribution risk (at the cost of model performance), the availability of new data is not always possible. Model trainers typically use all available data, and may not have extra data to use for such defenses.

We explore two variations of re-sampling defenses: *over-sampling* and *under-sampling*. In both cases, the model trainer re-samples data from its available datasets such that the resulting dataset is indistinguishable from a dataset sampled from a different distribution. For over-sampling we experiment with two flavors: using simple replacement and over-sampling based on inserting augmented data. These defenses rely on the key assumption that the model trainer knows the property they want to hide, and that there are a only few such properties so re-sampling to hide the desired properties will not unduly hard the model's task performance. When this assumption holds, resampling defenses can virtually eliminate inference risk. We evaluate this defense on configurations with low (CelebA–Sex), medium (Census19–Sex), and high (RSNA Bone Age–Age) inference risk to measure the impact of this defense.

Under-Sampling. The model trainer can simply under-sample its data such that the resulting dataset has a ratio corresponding to some other distribution. For example, a model trainer, with a dataset containing 70% females who wants to hide the ratio of females in the dataset from an inference adversary can simply under-sample examples with the 'female' attribute such that its data is balanced. This defense should prevent any disclosure about the pre-sampled distribution since there should be no difference between the cases where the training data was balanced to begin with and when it was adjusted with this defense, so long as the distribution is not distorted by the under-sampling. In our experiments, we find that under-sampling lowers

inference risk significantly, but does not completely eliminate it (Table 3.12). Mean distinguishing accuracy drops below 54% (n_{leaked} <0.1) for CelebA with significantly lower leakage for Census19 (< 57%, i.e., n_{leaked} <0.1) and RSNA Bone Age ($\sim$ 60%, i.e., n_{leaked} =0.3).

Over-Sampling. A model trainer not willing to sacrifice available training data by under-sampling may prefer to over-sample. The most basic variant over-samples the data before training begins, duplicating training records, and then trains its models like usual. Although this defense leads to the complete utilization of data, the presence of repeated data may reveal the property the adversary wants to hide to an adversary aware of the defense. It could, for instance, lead to a change in group-wise accuracies, which an adversary can learn to identify and still succeed at distribution inference.

Augmentation Based Over-Sampling. The ideal scenario for the defense would comprise of injecting fresh labeled data to adjust the desired property, as was assumed by Zhou et al. [376]. However, labeled data is scarce and may be expensive to acquire, and using techniques like pseudo-labeling can still leak information. In such scenarios, the model trainer can use augmentation techniques to synthetically generate additional samples. This avoids repeating samples, and may have the added benefit of potentially increasing the model's robustness to augmentations. But, the use of augmented data in an imbalanced way may still reveal information to a distribution inference adversary. For this defense, we focus only on the CelebA dataset, since designing augmentation for tabular datasets is much harder, and augmentations for RSNA Bone Age are limited. Task accuracy remains comparable and inference risk drops significantly (slightly higher than other forms of sampling), but is not completely eliminated and still higher than standard under and over-sampling.

Impact on Fairness. This form of re-sampling is common in research related to improving fairness in machine learning [212], commonly known as "unbiasing". However, re-sampling data can impact different sub-groups and populations of the distributions unequally, creating issues related to fairness in model predictions [163]. To investigate such potential impacts, we measure the impact of under-sampling and over-sampling-based mitigation strategies on fairness. We compare the precision and recall for another group and its possible values, for both undersampling and oversampling. Re-sampling based defenses have negligible impact on fairness in the case of CelebA, but result in disparate impacts of both under/over-sampling on the two groups. for Census19 (Table 3.11). For instance, over-sampling from a ratio $\alpha < 0.5$ lowers both precision and recall for whites, but increases recall and decreases precision more greatly for not-whites. These changes are even more severe for RSNA Bone Age, where changes in precision can be as high as 20% in opposite directions for different groups.

Adaptive Attacks. An adversary with knowledge of the under-sampling approach may be able to derive the original distribution by estimating the size of the training data to learn the sampling ratio. The strongest

Re-Sampling	α	Precision		Recall	
		not-white	white	not-white	white
Under-Sampling	< 0.5	$\downarrow 2\%$	$\sim$	$\uparrow 1\%$	$\downarrow 1\%$
	> 0.5	$\downarrow 1\%$	$\sim$	$\downarrow 1\%$	$\sim$
Over-Sampling	< 0.5	$\downarrow 2\%$	$\downarrow 1\%$	$\uparrow 1\%$	$\downarrow 1\%$
	> 0.5	$\downarrow 1\%$	$\downarrow 1\%$	$\downarrow 1\%$	$\downarrow 1\%$

Table 3.11: Relative change (%) in precision and recall metrics for white and not-white (race attribute), for Census19 (gender) for under-sampling and over-sampling. We consider cases where data for males ($\alpha < 0.5$) or females ($\alpha > 0.5$) is under-sampled for equalization.

adversary would be one that starts with knowledge of specific records in the original training dataset, and can use membership inference attacks to estimate how many of those records are included in the under-sampled dataset. We evaluate such attacks in §3.7.4, and find they are unlikely to be effective without dramatic improvements to membership inference attacks.

3.7.4 Adaptive Attacks Against Under-Sampling

Assume a more powerful adversary that has access to m training records each corresponding to attribute 0 ($\mathcal{D}_-$) and 1 ($\mathcal{D}_+$), and the original dataset has size n. In this scenario, the adversary is unaware of the original distribution of these attributes (α). Consider the scenario where the victim utilizes under-sampling on its original distribution as a defense to protect α, and re-samples such that both attributes are equally likely.

For our analysis, we assume a near-perfect membership inference adversary, with a false negative rate β. In such a setup, the adversary can check which of its data (the one with attributes zero, and attributes one) still all tests as members. If all zeros still remain members, then data from ($\mathcal{D}_+$) must have been under-sampled, and thus the original α must be > 0.5. Assuming that under-sampling is performed by pruning points uniformly at random, the density of members in the resulting data must remain the same. Thus,

$$\frac{m}{\alpha \cdot n} = \frac{m^{'}}{(1-\alpha) \cdot n} \tag{3.50}$$

$$m_- = \beta \cdot m \tag{3.51}$$

$$m_+ = m^{'} \cdot \beta \tag{3.52}$$

where m_+ is the number of datapoints (out of the known m) with attribute 1 that are inferred as members by the adversary, and m_- for attribute 0. α can thus be estimated as $m_-/(m_- + m_+)$. By symmetry, the case where original $\alpha < 0.5$ yields a similar formula. We test the risk of this adversary while varying the number of data points m, for different values of α. The adversary in this case thus directly predicts α, and mean square error (MSE) values are computed accordingly for the regression case. We use the R attack from

Defense	Task Accuracy (%)	Distinguishing Accuracy (n_{leaked}) $\alpha_1 = 0.2$	Distinguishing Accuracy (n_{leaked}) Mean
	Census19 (Sex)		
No Defense	77.9 ± 0.9	89.8 (2.1)	82.5±17.9 (4.2±5.3)
DP ($\epsilon = 1.0$)	77.0 ± 1.0	65.4 (0.2)	69.3±14.6 (0.6±0.7)
DP ($\epsilon = 0.12$)	75.6 ± 1.0	72.4 (0.5)	73.4±14.8 (0.8±0.9)
Label Poisoning ($r = 0.2$)	77.3 ± 1.0	78.4 (0.8)	78.9±17.4 (3.5±5.4)
Label Poisoning ($r = 0.4$)	74.9 ± 1.2	66.4 (0.2)	70.0±17.9 (1.9±4.2)
Under-sampling	77.5 ± 0.5	50.0 (<0.1)	56.7±6.8 (0.1±0.1)
Over-sampling	77.3 ± 0.6	50.0 (<0.1)	51.9±2.5 (<0.1±0)
	RSNA Bone Age (Age)		
No Defense	65.8 ± 2.0	99.9 (20.1)	99.8±0.4 (22.6±4.2)
Label Poisoning ($r = 0.2$)	64.3 ± 2.3	99.9 (20.1)	95.7±6.2 (12.1±7.1)
Under-sampling	65.4 ± 3.2	73.4 (0.5)	59.1±13.3 (0.3±0.5)
Over-sampling	64.6 ± 2.8	70.4 (0.4)	60.2±11.2(0.3±0.4)
	CelebA (Sex)		
No Defense	91.6 ± 0.8	89.6 (2.1)	85.3±15.8 (3.2±2.7)
Label Poisoning ($r = 0.2$)	90.0 ± 5.0	82.0 (1.1)	78.3±15.6 (1.2±1.0)
Adv. Training ($\epsilon = 4/255$)	90.4 ± 0.8	93.8 (3.1)	86.8±16.4 (5.3±5.2)
Adv. Training ($\epsilon = 8/255$)	88.5 ± 1.2	95.4 (3.7)	88.3±15.0 (5.2±4.9)
Adv. Training ($\epsilon = 16/255$)	75.7 ± 11.9	76.6 (0.7)	58.9±13.1 (0.2±0.4)
Under-sampling	90.8 ± 1.1	50.0 (<0.1)	53.7±6.1 (<0.1±0.1)
Over-sampling	90.6 ± 0.8	50.0 (<0.1)	53.8±4.1 (<0.1±0.1)
Augmentation-based over-sampling	91.7 ± 1.6	74.8 (0.6)	61.0±14.5 (0.3±0.5)

Table 3.12: Effectiveness of considered defenses. Each distinguishing accuracy (and corresponding n_{leaked}) reported is the observed leakage of KL. The first results are for predicting between $\alpha_0 = 0.5$ and $\alpha_1 = 0.2$; the last column reports mean distinguishing accuracy (with mean n_{leaked} in parentheses) as described in Table 3.5. Mean distinguishing accuracies (and n_{leaked} numbers) are reported with ± standard deviation, over different α_1 values. Most noise-based defenses harm model task accuracies, and the only defenses that diminish leakage without harming task accuracy are based on data re-sampling.

Ye et.al. [353], and use the authors‘ official implementation, with the FPR set to 0.05. Similar to the case of binary distinguishing, we use n_{leaked} to measure the adversary's success (Theorem 3.2.3).

MSE values for varying number of members (m) with corresponding n_{leaked} values are reported in Table 3.13. For the most realistic case, with knowledge of upto 100 members (m) per attribute, the inference risk remains very low, with MSE values as high as ~ 5 ($n_{\text{leaked}} < 0.1$). This risk is expected to increase with increase in membership knowledge, as in the extreme case, the adversary would have perfect knowledge of the entire training dataset. One notable exception is RSNA Bone Age, where the MSE drops to ~ 0.4 (n_{leaked} =2) for $m = 500$. This is not surprising, as $m = 500$ for the case of RSNA Bone Age corresponds to $\sim 15\%$ of the victim's training dataset, which is unrealistically high.

For the task of binary distinguishing between $\alpha_0 = 0.5$ and some other α_1, it suffices to see whether the predicted ratio is sufficiently different from 0.5. We do so by checking the predicted α, and predict $\mathcal{G}_1(\mathcal{D})$ if it differs from 0.5 by more than 0.03, and $\mathcal{G}_0(\mathcal{D})$ otherwise. As a baseline, we also consider a simpler adaptive

adversary that uses the same re-sampling setup as the victim, re-sampling data for its shadow models. Such an adversary can potentially work, since the KL Divergence Attack compares distributions of predictions, which might be sufficiently different between re-sampled and non-sampled ($\alpha = 0.5$) models.

Mean distinguishing accuracies and corresponding n_{leaked} values are reported in Table 3.14. Cases where the adversary uses the same setup as the victim (for re-sampling) leads to significant inference leakage in most cases, with mean distinguishing accuracies as high as 80% (n_{leaked} =1.8) for Census19. Similarly, the MI-based distribution inference leakage is particularly high for RSNA Bone Age. This is in line with previous observations with the MSE values (Table 3.13), since the number of members corresponds to a significant portion of the victim's training data.

3.8 Related Work

This section summarizes work on formal definitions of privacy and distribution inference attacks.

Privacy Definitions. Most formal privacy definitions, including numerous variations on differential privacy [67], focus on bounding inferences about specific data elements, not the statistical properties of a dataset. The key privacy notion of traditional differential privacy is intuitively connected to the risk to an individual in contributing their data to a dataset — this corresponds well to dataset privacy risks, but does not capture distribution inference risks; indeed, the main goal of most differentially private mechanisms is to satisfy the inference bound for individual data while providing the most accurate aggregate statistics possible. One notable exception is the Pufferfish framework [155], which introduces notions that allow capturing aggregates of records via explicit specifications of potential secrets (e.g.,, distribution of vehicle routes in a shipping company) and their relations. Zhang et al. [372] extend the Pufferfish framework to define the concept of "attribute privacy", including a notion of distributional attribute privacy that takes a hierarchical approach for parameterizing distributions and could be instantiated to capture notions of distribution inference such as the fraction of records with some attribute. Although these definitions are promising and valuable, none of them satisfy our simple goal to define distribution inference attacks in a way that is general and powerful, while clearly distinguishing inferences that are considered attacks from allowable statistical inferences.

Dataset/Task	**Number of known members (m)**		
	10	100	500
Census19 (Sex)	9.043 (<0.1)	4.588 (0.2)	4.078 (0.5)
RSNA Bone Age (Age)	1.969 (0.1)	0.486 (0.8)	0.372 (2.0)
CelebA (Sex)	4.785 (<0.1)	1.466 (0.2)	1.202 (0.4)

Table 3.13: MSE values (with mean n_{leaked} in parantheses) for direct regression over α for an adversary that utilizes membership inference to infer α for models trained with under-sampling based defense.

Dataset/Task	KL	MI-Based	Same Setup + KL
Census19 (Sex)	56.7 (0.1)	57.1 (<0.1)	80.4 (1.8)
RSNA Bone Age (Age)	59.1 (0.3)	65.9 (0.3)	50.0 (<0.1)
CelebA (Male)	53.7 (<0.1)	50.0 (<0.1)	64.7 (0.3)

Table 3.14: Mean distinguishing accuracies (with mean n_{leaked} in parentheses) for the task of binary distinguishing between $\alpha_0 = 0.5$ and α_1, while varying α_1, for the standard adversary (KL), an adversary that utilizes membership inference to infer α for models trained with under-sampling based defense (MI-Based), and a simpler adversary that just copies the victim's re-sampling setup (Same Setup + KL).

A recent attempt to formalize property inference [199] consists of a framework that reduces property inference to Boolean functions of individual members, posing the ratio of dataset members satisfying the given function as the property. These ratio-based formulations limit the kinds of distribution inferences considered since they cannot capture many other kinds of statistical properties of the training distribution that may be sensitive, like the degree distribution of a graph [110]. Ratio-based formulations assume the property function is applicable over individual data points, while for graphs it is an aggregation over interconnected nodes.

Distribution Inference Attacks. All previous distribution inference attacks in the literature take a meta-classifier approach—the adversary trains models on datasets with different properties, then trains a meta-classifier using those models. The adversary then uses the meta-classifier to predict a property of the victim's training data, which is usually related to the ratio of members satisfying some Boolean property. Ateniese et al. [17] were the first to identify the threat of distribution inference (termed *property inference*) and proposed a meta-classifier attack targeting Support Vector Machines and Hidden Markov Models. The proposed attacks can successfully infer the accent of speakers in speech-to-text systems, or presence of particular traffic in network traffic classification systems. Model representations for training the meta-classifier can take several forms: using model weights [95], gradients [214], or activations/logits for a set of query points [333, 348]. These methods show promise, achieving better-than-random results for several properties, tasks, and models across different domains. For instance, predicting a doctor's specialty based on rating-prediction systems on text reviews [371], identifying accents of speakers in voice-recognition models [17], and even predicting if a model has been trained with Trojans [348]. Although these approaches achieve high accuracies on "toy-like" classifiers like decision trees and shallow neural networks, and distributions that are highly disparate, successful property inference attacks have not been demonstrated on realistic (or even semi-realistic) deep neural networks or complex datasets. Our work is the first to demonstrate the capability of such attacks to work on large convolutional networks and different datasets across domains.

Zhou et al. [376] extend distribution inference attacks to directly infer property ratios for Generative Adversarial Networks (GANs). Although their attack setting includes targeting large GAN models on complex datasets, their attack does not use the victim model's model parameters directly. Similarly, Pasquini et al. [250] extend distribution inference attacks to a split-learning setting and only target parts of the victim

model. Zhang et al. [374] extended this approach to graphs and their properties, distinguishing training graphs according to properties such as the number of nodes and edges using attacks that assume access to graph embeddings.

These attacks have also been extended to settings where active adversaries that can poison the victim's training data [51]. The only previous attacks designed to work with label-only predictions are by Juarez et al. [147] that performs a statistical test based on attribute-wise model performance, and the attack by Mahloujifar et al. [199] in the setting of active adversaries.

3.9 Conclusion

An essential step to understanding distribution inference risks is a precise and formal definition. In this chapter, we introduced such a general definition, which subsequently leads to a systematic approach to quantifying the leakage from distribution inference attacks. Our empirical results reveal how intuition may not necessarily align with actual observations: seemingly similar pairs of distributions can have starkly different attack success rates, and simple attacks with limited access can sometimes outperform computationally expensive meta-classifiers. Our proposed black-box attacks are highly efficient and maintain their effectiveness even when access to exact prediction probabilities is unavailable. Even the lack of common feature extractors, a common setting in many evaluations in the literature, does not completely eliminate inference risk. Our experiments also show how direct regression of underlying ratios of training distributions is a real threat, and can be used to improve the performance of binary distinguishing attacks.

The general approach to achieve security and privacy for machine-learning models is to add noise, but our evaluations suggest this approach is not a principled or effective defense against distribution inference. The main reductions in inference accuracy that result from these defenses seem to be due to the way they disrupt the model from learning the distribution well, so observed reductions in inference risk are related to drops in task performance. Our experiments with different model architectures and differentially private training support this—inference risk increases significantly when the victim and adversary use the same learning algorithms or model architectures. The only reliably effective defense from our experiments is to re-sample data, which depends on the assumption that the model training is aware of the adversary's inference goals (or at least of the properties that should be protected). These re-sampling defenses, too, are not perfect, as they seem to negatively impact the fairness of groups related to the property attribute.

Like nearly all inference privacy work, we assume an adversary with access to a dataset that matches the underlying distribution (in this case, before the transformation to the actual training distribution as modeled by $\mathcal{G}_0$ and $\mathcal{G}_1$). This is a strong assumption, which may be realistic in some cases but is often unlikely. All our attacks (and nearly all previous ones) require representative data for training models locally. Exploring

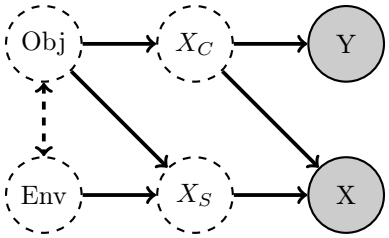


Figure 3.22: Given data X can be decomposed into causal features (X_C) that determine the corresponding label (Y) and other aspects of X, and style-related features (X_S) that have no impact on the corresponding label. The content is only determined by the object (Obj), while the style is a function of both the object itself and the environment it exists in (Env).

adversaries with limited data access to these distributions and how it impacts inference risk is left as part of future work.

There is a need for theoretical connections between distribution inference risk and general useful notions of machine learning, like model complexity and fairness. Our work suggests such connections do exist, and we hope they will be better understood as both empirical and theoretical understanding of inference privacy advances. Specifically, our experiments with varying property–task correlation and adversarial training suggest connections between the causality graph learned by the model, and properties of the training distribution that can be inferred. While causality-driven learning is beneficial for privacy [307], its connection to distribution inference is more direct, especially in distinguishing between properties that are "unavoidable" in terms of learning, and ones that should certainly not be inferred. For instance, in Figure 3.22, if a classifier can perfectly capture the underlying causal structure, it should not leak any information about the style features X_S. At the same time, it is unclear how much the content X_C an adversary should be able to infer. Formalizing these connections and understanding the trade-offs between learning causal structures and inference privacy is an important direction for future work.

Our work raises more questions than it answers—*why do some models leak a lot of information about certain properties of their training distribution but others leak little*, *what are the limits on how precisely training distributions can be distinguished*, *why do models trained on some datasets (like RSNA Bone Age and the graphs) appear to leak so much more information than others*. We are not able to answer these questions yet, although our experiments provide several intriguing observations and suggest possibilities to explore. It is not surprising that so little is understood about distribution inference—the research community has put extensive effort into studying membership inference attacks for several years now, and we are just beginning to be able to understand how and why membership inference risk varies [164].

Chapter 4

User-Level Inference *

In this chapter we study user-level inference, where the goal is to infer properties of individual subjects in a dataset, and its application in two real-world scenarios: federated learning and transfer learning. First, we explore user-level inference in the cross-silo federated-learning (FL) setting (§4.1). We study of factors in the FL environment that influence leakage (§4.2), providing actionable insights for practitioners to understand factors that influence leakage. We then explore an active adversary that can manipulate the pre-training process of a model to amplify leakage in the downstream models while maintaining low detection rates (§4.3). Such an adversary may release its models via services like Hugging Face [83] or upload them for unsuspecting victims to download.

4.1 Federated-Learning

We begin by formally describing the adversary's objective (§4.1.1), followed by a description of the threat model and assumptions about data and model access. Our attacks require only black-box API access: one of them only assumes access to the final trained model (§4.1.2.1), while the other assumes access after each training round, and is thus more suited to FL settings (§4.1.2.2).

4.1.1 Attack Objective

Let $\mathcal{S}_0$ be a set of subjects, and s_{interest} the subject whose membership the adversary wants to infer, such that $s_{\text{interest}} \notin \mathcal{S}_0$. Let $\mathcal{D}^s$ be the distribution corresponding to a subject s (*i.e.,* distribution of that subject's data). Then, using our definitions of distribution inference (§3.1), we can formulate our subject membership

*This chapter is largely based on Anshuman Suri, Pallika Kanani, Virendra J Marathe, Daniel W Peterson, *Subject Membership Inference Attacks in Federated Learning*, in arXiv, 2022 and Yulong Tian, Fnu Suya, Anshuman Suri, Fengyuan Xu, David Evans, *Manipulating Transfer Learning for Property Inference*, in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2023. Code relevant to this chapter is available at https://github.com/iamgroot42/TransferInference.

inference task as differentiating between models trained on datasets sampled from either of the distributions $\mathcal{D}_0$ and $\mathcal{D}_1$, defined as:

$$\mathcal{D}_b = \bigcup_{s \in \mathcal{S}_b} \mathcal{D}^s \quad , \quad \mathcal{S}_1 = \mathcal{S}_0 \cup \{s_{\text{interest}}\} \tag{4.1}$$

This is equivalent to stating that a data sample from either of $\mathcal{D}_{\{0,1\}}$ is equivalent to taking a union of samples from the individual subjects' distributions. The first distribution $\mathcal{D}_0$ corresponds to the absence of subject of interest in the federation, while $\mathcal{D}_1$ includes it. **A Subject membership inference attack thus aims to infer whether a given subject's data was used in the federation, i.e., was present in *any* of the data sources.** The flow of information for the proposed subject membership inference attack is described in Figure 4.1.

Note that subject membership inference is orthogonal to the FL setting, and is indeed more broadly applicable to ML models. For subject membership inference in FL, it is important to note that it does not matter how a subject's data is divided across different users of the federation. Even if only one user has the subject's data, or if an individual subject's data is divided across all users, the subject's data is ultimately used in the overall training process and thus the subject should be inferred as being present. The adversary only cares about the subject's presence in the overall federation and using the above formulation is apt for the given threat model.

4.1.2 Threat Model

Attacks may be grouped by whether or not the adversary has knowledge (or perhaps partial knowledge) of the model, into black-box, white-box, and grey-box attacks [230, 315]. Participants in FL have access to the model architecture and parameters, and thus have white-box access. We do not assume the attacker is part of such an FL setup, or that it has white-box access to the victim's model. Our Loss-Threshold Attack (§4.1.3.1) uses only knowledge of the data points, the labels assigned by the model, and the loss function the model is optimizing; which is essentially a black-box attack. The Loss-Across-Rounds Attack (§4.1.3.1) additionally assumes API access to the model after each training round, and is designed to extract additional information from the model's training behavior across time. This attacker can exist as an honest-but-curious federation server/user in the federation. In either case, by design the attacker has access to the global model after each training round. For all of our attacks, we assume the adversary has access to the following:

4.1.2.1 Samples (finite) from the distribution of subjects

If the adversary wishes to launch an attack against a particular subject, it must have the capability to quantify and differentiate subjects and identify the one it is interested in. This can be done by either knowing (or estimating) a subject's distribution or possessing finite samples to estimate it. Having access to finite set

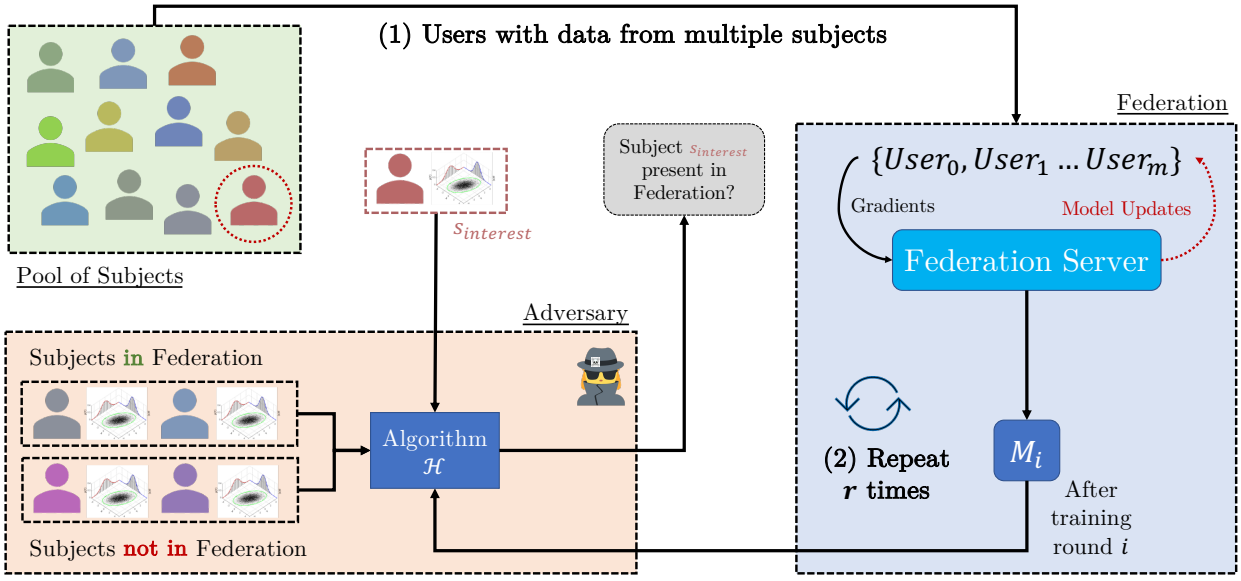


Figure 4.1: Information flow for the subject membership inference attack in Federated Learning. The adversary uses some algorithm $\mathcal{H}$, along with knowledge of membership of a few subjects and the models M_i after each training round i, to infer the membership of s_{interest}'s data in any of the user's data.

of samples from the subject's distribution is the weaker assumption of these two. Note that in theory, it is not necessary to have estimates for distributions for *all* of the subjects— just for the subject of interest, and some samples from subjects with known inclusion/exclusion labels. We assume the attacker has access to a limited number of known included/excluded subjects, and samples from each distribution, in order to tune the threshold values. Since our attacks do not require training any shadow models (which is the case for most state-of-the-art membership [353], distribution inference §3.3, and subject-membership [56] attacks), it is general and can be applied in very-low-data settings.

4.1.2.2 API access to the global model after each federation round

We assume access to prediction probabilities from the global model after each training round. Both the central server and individual participants have access to the global model after each training round, making it easy to satisfy this requirement. This may be further weakened to limit access to just the last round- the final global model that may be released to the world. We thus propose two attacks; one for each level of access described here.

4.1.3 Method

Both of our attacks are based on hypotheses implied by prior works on the behavior of loss functions on training data [138, 263, 355]. Given the objective of training ML models, it is natural to expect that the model's performance on data similar to that seen during training would be better than that not seen during training. The Loss-Threshold Attack (§4.1.3.1) is generic and applicable to any ML model with black-box

access, while the Loss-Across-Rounds Attack (§4.1.3.2) assumes access to intermediate model state during training, and is thus more suitable for FL settings.

Let m be the total number of users participating in the federation. Let r be the number of rounds for which the global model is trained in the federation, with M_i denoting the state of the model after training round i has completed. M_0 thus represents the state of the model before training starts. Let $l_i(x, y)$ be the loss value between the label y and $M_i(x)$, with $M_i(x)$ denoting the model M_i's prediction on point x.

4.1.3.1 Loss-Threshold Attack

Hypothesis 1. *If data from a particular subject is present in the federation and is used in training, the global model would be expected to have a lower loss on it than data from a subject that was not present in any of the users' local datasets [355].* Based on this hypothesis, we propose the following attack: record loss values for samples from the target subject's distribution and check how many of them have a value less than a particular threshold. If the loss is below the threshold, it would indicate the model has seen data from that subject's distribution, during training.

$$c = \sum_{(d_x, d_y) \sim \mathcal{D}_s} \mathbb{I}[l_r(d_x, d_y) \leq \lambda] \tag{4.2}$$

The adversary can either check if c is non-zero or derive an additional threshold on this value based on the metric it wishes to maximize, like precision or recall.

4.1.3.2 Loss-Across-Rounds Attack

Hypothesis 2. *Loss on training data, and thus data from the training distribution, decreases across iterations by virtue of how learning algorithms work. However, data from distributions not seen during training would likely not converge to values as low as those of subjects present in the federation [328].* Based on this hypothesis, we propose the following attack: record loss values for samples from the subject's distribution and note how the loss values change as training rounds progress. The attack first computes the loss across each training round i:

$$c_i = \sum_{(d_x, d_y) \sim \mathcal{D}_s} l_i(d_x, d_y) \tag{4.3}$$

Then, the adversary takes note of the number of training rounds where the loss decreases after each round:

$$c = \sum_{i=1}^{r} \mathbb{I}[c_i < c_{i-1}] \tag{4.4}$$

The adversary can then compute these values for both subjects seen and not seen in the federation and consequently derive a threshold on this value for subject membership. The attack implicitly assumes all users contribute in each training round. Although this assumption may not hold in most settings, the likelihood of any user chosen in a training round containing a subject's data is non-trivial. This, coupled with the robustness of learning algorithms over individual rounds [107], is sufficient to launch the attack and achieve high inference leakage.

4.1.3.3 Threshold Tuning

Membership inference attacks label whether a subject is part of the training data; it is common for these labeling strategies to depend on some parameters or hyper-parameters, like any ML system. These hyper-parameters are usually computed using additional information that may be available through side-channel attacks or just by the adversary participating in the federation training.

All of our attacks involve computing some form of tunable thresholds (e.g., λ in Equation (4.2)). The threshold values affect the precision/recall tradeoff of the attack, and in this work we learn their optimal values from a data set of correctly labeled included and excluded subjects. For the scenario where the adversary is a participant in the federation, it can use its split of data to generate data for the attack. At the very least, the attacker knows for certain that subjects for which records exist in its training data are part of the federation. The adversary can then guess subjects that are likely not used in the federation by randomly sampling (or generating) other subjects not in their data, or by intentionally holding some data out from the training of the federated model. The adversary can infer additional information about non-users based on task knowledge, as it is part of the federation. For example, if FL involves X-ray images, the adversary can use individuals known to be not part of the FL e.g., patients from another country. Once data for both subjects used and (probably) not used during training is available, the adversary can tune the thresholds of their chosen attack to accurately predict whether a subject's data was used in the federation or not.

4.1.4 Evaluation

Equipped with knowledge of the threat model and the adversary's capabilities, we move on to the following questions:

1. *How well can an adversary perform in the absence of access to exact data from training, while testing for subject membership inference?*

2. *How little information about subject inclusion/exclusion can the adversary get away with while still being effective?*

3. *How does FL affect subject membership inference risk, compared to standard training?*

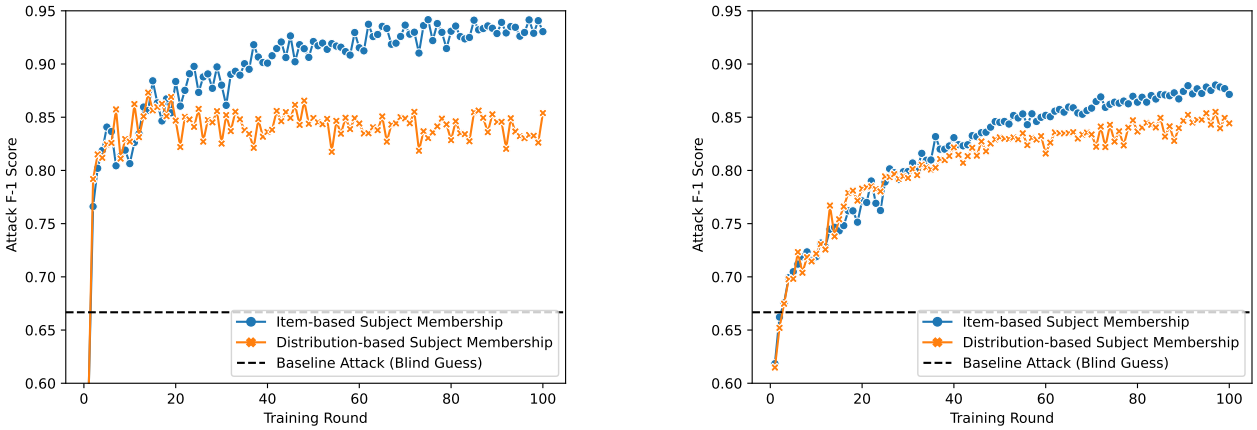
4. *How well do these attacks hold up against differential-privacy based mitigation approaches?*

5. *How do the properties of the data, model and federation affect attack performance?*

To answer these questions, we use a real-world dataset and train models on it with both standard and FL training (§4.1.4.1). We then test out the efficacy of our attacks under various training environments and defenses (§4.1.4.2). Our evaluations reveal how distribution-based attacks can be just as potent as ones based on exact record membership, both in extracting subject membership information, and evading defenses (§4.1.4.3).

4.1.4.1 FEMNIST Experimental Setup

We use FEMNIST [39], the federated extended MNIST [66] dataset, an image classification task for handwritten digits and letters. FEMNIST's digits and letters themselves have been written by 3500 distinct individuals, and FEMNIST partitions these images by individual authors. Each author has contributed hundreds of sample images. Ordinarily, FL research experiments [39] map each author to a federation user, resulting in a $3,500$-user federation. In our experiments, we instead map authors to subjects, and reserve half of the subjects as non-member subjects, resulting in a federation with 1750 subjects whose data are randomly scattered among a handful of federation users (16 in our experiments). The remaining half (1750) subjects are never involved in the federation and are "non-members". Each subject has ~ 140 data points on average, with its data more-or-less equally spread across 16 federation users. Multiple federation users may host images from the same subject, though we do not distribute any individual image to more than one federation user. This reconfigured dataset is especially suitable for cross-silo FL and our subject membership attacks study. The data points themselves are 28x28 pixel, black-and-white pictures of a single handwritten digit or letter.



(a) Normal Training

(b) Federated Learning

Figure 4.2: Attack F-1 Score across training rounds inferring subject membership using our attack (orange) and via membership inference attacks using exact records (blue), for normal training (a) and FL (b). Inference risk is not harmed significantly by the lack of access to exact data, and can be just as bad for FL as normal training.

Target Model Training. We use the CNN model on FEMNIST appearing in the LEAF data suite [39] as our target model to train. We use half of the data points belonging to each of the 1750 member subjects for training, and reserve the remaining half as in-distribution data points to sample from for carrying out the distribution-based attack. In the standard model, we simply combine the data from all member subjects to train a standalone model using Stochastic Gradient Descent, while for the federated setting, we use FedAvg [148] training protocol. We train each model for 100 rounds using the Adam [157] optimizer, with a learning rate of 0.001 and batch size 512.

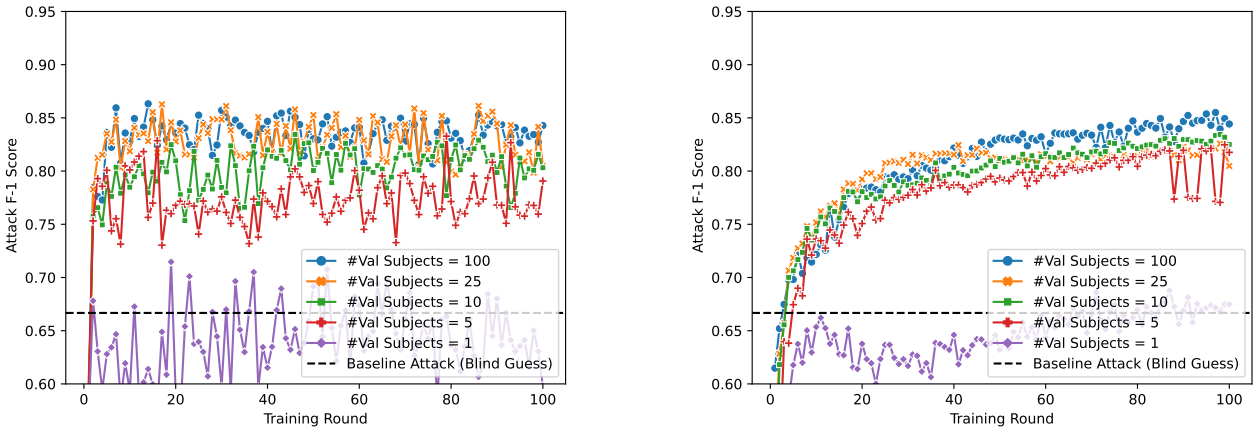
Attack Set, Threshold Tuning, and Evaluation. We prepare the attack set by sampling at most 100 examples from both member (the other half of each member subject's data, as mentioned above) and non-member subjects. We then split this data by *subject* into two parts. The first split is used by the adversary to derive the subject membership inference threshold(s) λ (we call this the validation set), while the second split is used for evaluating the effectiveness of the attack and reporting results. We compute attack F_1 scores to measure adversary success; we count correctly predicting the presence/absence of a subject's data in the federation as a hit (1) and incorrect prediction as a miss (0).

4.1.4.2 Results on FEMNIST

We evaluate how the lack of access to exact records affects inference risk, revealing how attacks retain much of their potency. Much of this potency is retained even as the number of subjects for validation decreases. These results hold for both our attacks, which we find to be similar in performance. Finally, we evaluate DP mechanisms at different granularities and privacy budgets as defenses (§4.1.4.3).

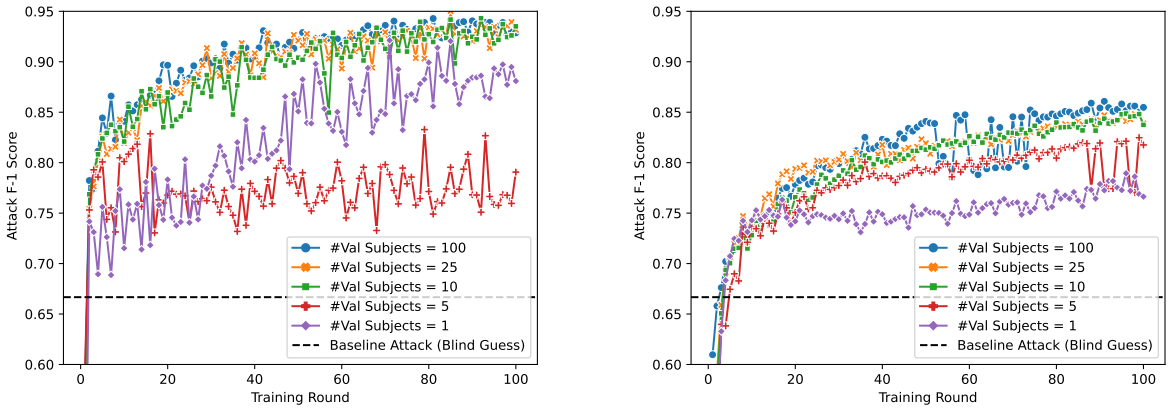
Item based v/s Distribution based. To consider the impact of not having access to exact data records, we design two versions of our attacks. The first version (*Item-based*) assumes access to exact records used in model training while testing for subject membership, while the second version (*Distribution-based*) only assumes access to a subject's distribution. As expected, there is a gap in performance between the two settings, as the item-based access model makes much stronger assumptions (Figure 4.2a). Nonetheless, attack performance with just distribution-based access is high, achieving attack F-1 scores ≥ 0.85. **An adversary can thus perform subject membership inference with high success, even without access to exact records.** Interestingly, we also note that inference risk is high after as few as two training rounds suggesting that subject membership inference, unlike membership inference [355], is high even when before the model has overfit. Next, we train ML models via Federated Learning, and repeat our attacks with the same two versions as the experiment above. Not only do we observe similar attack performance for both variants, but the difference in their efficacy is also even lower when data is aggregated via FL.

We assume knowledge of membership for 100 subjects for computing thresholds. This is relatively small fraction ($\sim 5\%$), but may be hard to obtain in some scenarios. To measure changes in inference risk as this



(a) Normal Training (b) Federated Learning

Figure 4.3: Attack F-1 Score across training rounds inferring subject membership, for normal training (a) and FL (b), while varying the number of subjects in-set used for validation. Inference risk is robust to the number of subjects used for validation, and is quite high for as few as 5 subjects. We observe similar trends for F_1 for the Item-level Subject Membership scenario (Figure 4.4).

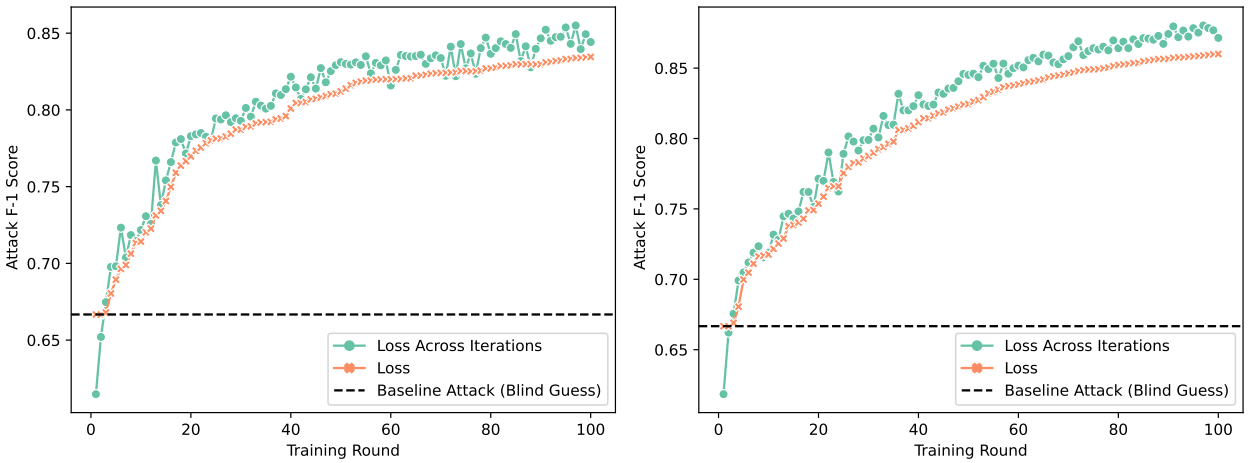


(a) Normal Training (b) Federated Training

Figure 4.4: Attack F-1 Score across training rounds inferring subject membership using our attack with Item-based Subject Membership, for normal training (a) and FL (b), while varying the number of subjects in-set used for validation.

number is lowered, we repeat experiments for both standard and FL training while varying this number in $\{1, 5, 10, 25\}$. In both settings, inference risk is near-random when only one subject's membership is known, but fairly robust as long as membership for ≥ 10 subjects is known (Figure 4.3). Our evaluations demonstrate how **adversaries can be fairly successful with knowledge of as few as five subjects** ($\sim 0.3\%$ of all subjects).

Attack Variants. We compare the efficacy of our two attacks in the FL setting. Figure 4.5a shows results for the two proposed variants of the distribution-based attack in the FL setting. Intuitively, the



(a) Distribution-based (b) Item-based

Figure 4.5: Attack F-1 Score across training rounds inferring subject membership using our attack with distribution-based (a) and item-based (b) subject membership for the two proposed attacks (Loss-Threshold and Loss-Across-Rounds)

Loss-Across-Rounds Attack should be at least as powerful as the Loss-Threshold Attack, since the former has additional information about the model across its training. Indeed, the Loss-Across-Rounds Attack outperforms the Loss-Threshold Attack nearly across all of the training rounds. However, the gap in performance is negligible for most cases. Similar trends hold in the item-based variants for FL and in the standard training experiments, as shown in Figure 4.5b. Given their similar performance, and the weaker assumptions made by the Loss-Threshold Attack (access to only the final trained model), we default to the latter for evaluations in the rest of this chapter.

4.1.4.3 Differential Privacy

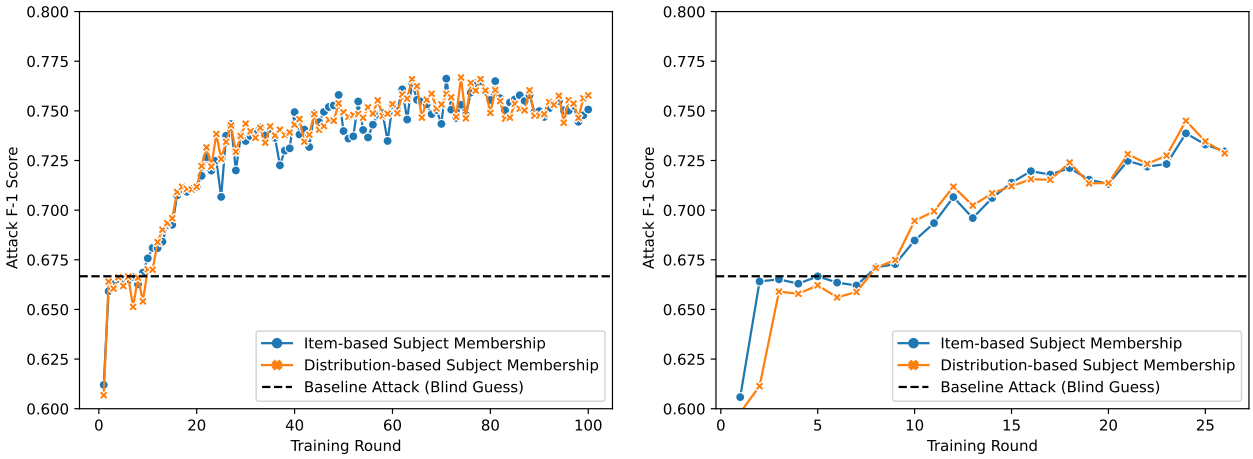
One of the most commonly prescribed method for defending against membership inference attack is training ML models with Differential Privacy (DP) [74]. In particular, Federated Learning models can be trained with Local Differential Privacy [80, 151, 334] at various granularities as described before [1, 189, 206, 210]. Algorithms can either provide guarantees at the level of records (Item DP), federation users (User DP), or data subjects (Subject DP). Item DP is implemented using a federated variant of the DP-SGD algorithm [1]. User DP is another variant of federated DP-SGD that provides user-level local DP [206]. Subject DP is the Hierarchical Gradient Averaging algorithm that guarantees subject-level DP [206]. HiGradAvgDP builds on the DP-SGD algorithm by Abadi et al. [1]. To obfuscate the contribution of a subject to mini-batch gradients HiGradAvgDP scales down each subject's mini-batch gradient contribution by averaging it and then clipping that average to the threshold C. This bounds the sensitivity of the algorithm. Gaussian noise is then added at the scale of the clipping threshold. We evaluate all these algorithms against models trained in FL without any DP, and report results for privacy parameters $\varepsilon = 4.0, \delta = 10^{-5}$ in Table 4.1.

DP Granularities. User-level *local* DP provides the best protection and completely eliminates inference risk [206], but at the cost of massive drops in task accuracy, rendering it impractical. This protection is expected, since user-level local DP is a strictly stronger notion of privacy than subject-level DP. Subject-level DP, designed exactly for our threat model, lowers inference risk to near-random with a considerable drop in task performance. Item-level DP, as expected, provides the least protection against our distribution-based adversaries. Closer inspection of inference risk under these different granularities of privacy reveals how

Granularity	**Attack**				**Task**
	Accuracy	Precision	Recall	F_1	Accuracy
FL	.82	.79	.89	.83	91.9 ± 1.0
Item DP	.73	.69	.83	.76	85.1 ± 1.5
User DP	.51	.51	.98	.67	41.0 ± 1.5
Subject DP	.65	.61	.88	.72	81.5 ± 1.7

Table 4.1: Attack metrics and model task accuracy for vanilla FL and under different DP granularities at privacy budget $\epsilon = 4.0$, while using the *Distribution-based Loss-Threshold Attack* on FEMNIST. F_1 scores across training rounds are given in Figure 4.7.

distribution-based adversaries are just as powerful as Item-based adversaries, even in the presence of these defense mechanisms (Figure 4.6).

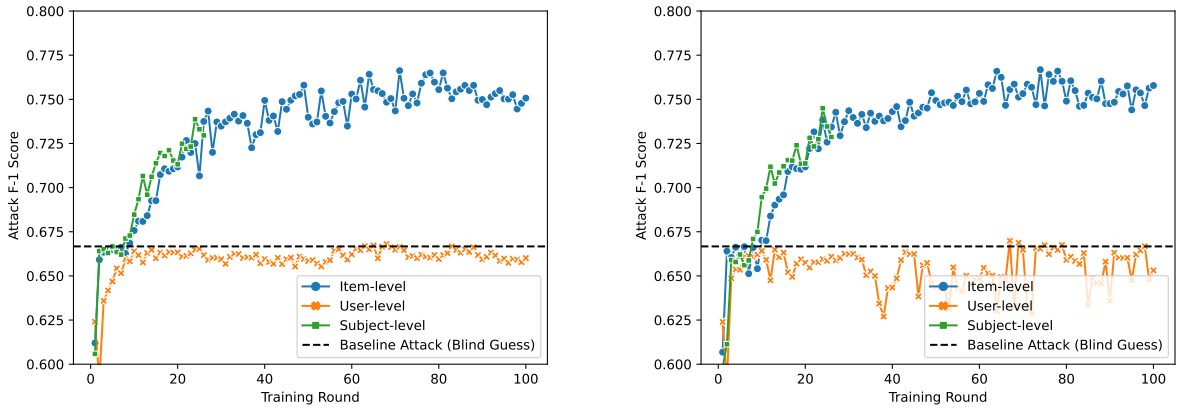


(a) Item-level DP

(b) Subject-level DP

Figure 4.6: Attack F-1 Score across training rounds inferring subject membership using our attack (orange) and via membership inference attacks using exact records (blue), for Item-level DP (a) and Subject-level DP (b). Both variants of the attacks are equivalent in potency, irrespective of the granularity of DP used for protection.

Varying Privacy Budget. The previous experiment shows that for a privacy budget of $\epsilon = 4$, the proposed inference attacks maintain residual risk. We next study if further reduction in the privacy budget successfully eliminates this risk. We do not perform this experiment with user-level DP, since attack performance is already close to random at $\epsilon = 4$. Figure 4.8 shows that decreasing privacy budget indeed helps protect



(a) Item-based Subject Membership

(b) Distribution-based Subject Membership

Figure 4.7: Attack F-1 Score across training rounds inferring subject membership, for Item-based (a) and Distribution-based (b), for various DP granularities in FL. User-level DP completely eliminates risk, although at the cost of a huge dent in task performance. Subject-level DP, as expected, leads to lower final inference risk.

against these attacks. However, as can be seen from Table 4.2, this added protection comes at the cost of loss in task accuracy.

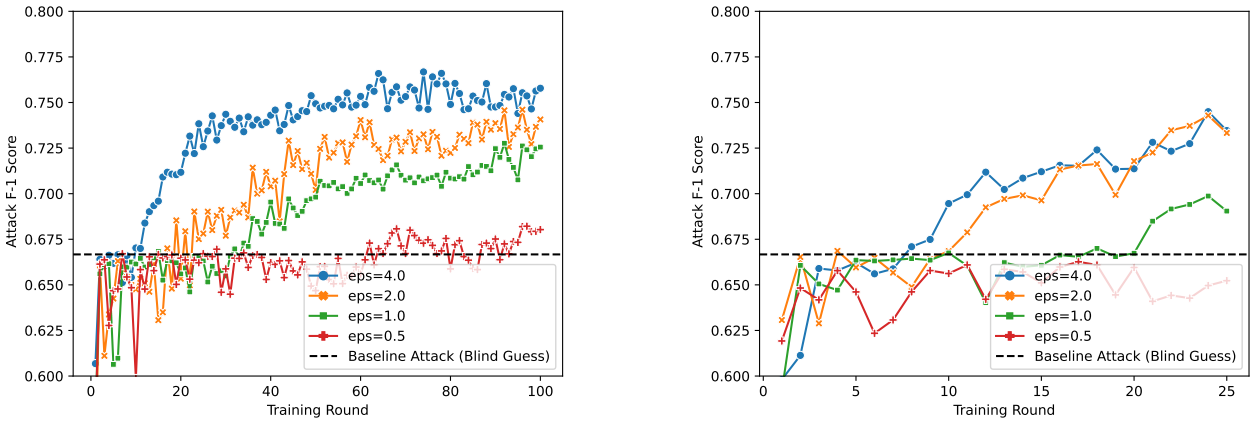
Granularity	$\epsilon = 4.0$	$\epsilon = 2.0$	$\epsilon = 1.0$	$\epsilon = 0.5$
Item-level	85.0 ± 1.4	81.9 ± 1.6	76.9 ± 1.7	68.9 ± 2.2
Subject-level	81.5 ± 1.7	76.3 ± 1.9	69.3 ± 2.2	57.4 ± 2.4

Table 4.2: Model task accuracy for different ϵ values under two DP granularities, while using the *Distribution-based Loss-Threshold Attack* on FEMNIST.

4.1.4.4 Shakespeare Experimental Setup

The second dataset used in our evaluation is Shakespeare [39], a next-character prediction task on a corpus of data from classic William Shakespeare plays. The dataset is divided by dialogues of Shakespeare play characters, where each character serves as a federation user. In our evaluation, we treat these play characters as data subjects instead of federation users, and uniformly scatter each subject's data items among all federation users. With a total of 660 subjects, we split the subjects into member and non-member sets of 330 data subjects each. Each subject's data is scattered uniformly among 16 federation users, with no data item assigned to more than one federation user.

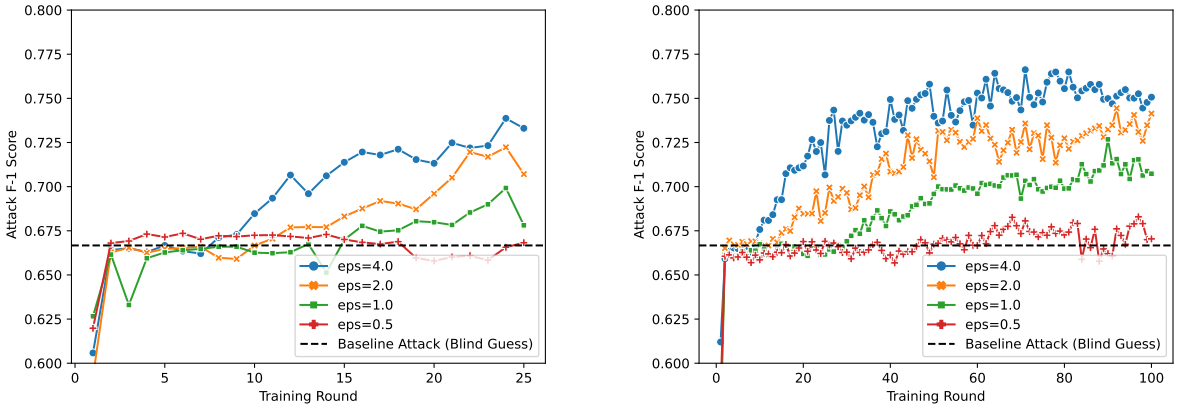
Target Model Training. We use a stacked LSTM model with two linear layers at the end for the Shakespeare dataset. Like the FEMNIST experiments, we use half of the data points belonging to each of the 330 member subjects for training, and reserve the remaining half as in-distribution data points to sample from for carrying



(a) Item-level DP

(b) Subject-level DP

Figure 4.8: Attack F-1 Score across training rounds inferring subject membership using our attack, for Item-level DP (a) and Subject-level DP (b) with varying levels of protection (ϵ). Both variants of the attacks are equivalent in potency, irrespective of the granularity of DP used for protection. Results for Item-based Subject Membership are provided in Figure 4.9.



(a) Item-based Subject Membership

(b) Distribution-based Subject Membership

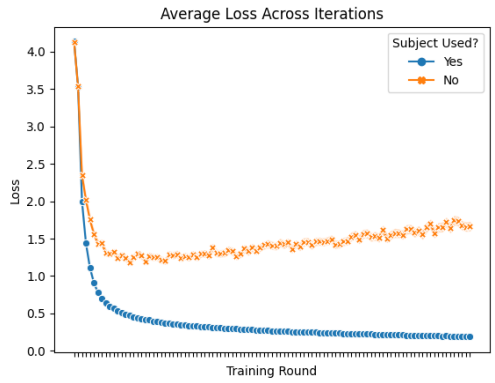
Figure 4.9: Attack F-1 Score across training rounds inferring subject membership using our attack with Item-based Subject Membership, for Item-level DP (a) and Subject-level DP (b) with varying levels of protection (ϵ). Both variants of the attacks are equivalent in potency, irrespective of the granularity of DP used for protection.

out the distribution-based attack. We train each model for 200 rounds using the Adam [157] optimizer, with a learning rate of 0.01.

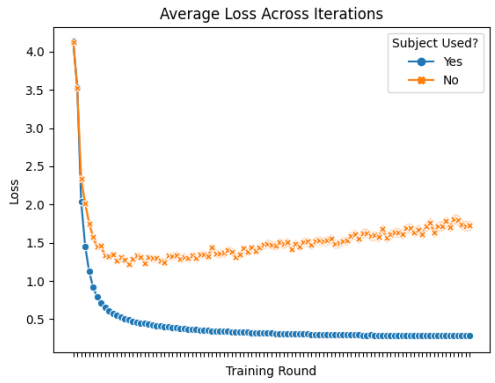
4.1.4.5 Results on Shakespeare

We evaluate the efficacy of our proposed attacks in the federated setting. The *Item-based* version achieves an attack accuracy of 0.51 while the *Distribution-based* version achieves an attack accuracy of 0.5. On investigating this attack ineffectiveness on this dataset, we find that the loss values for subjects used in training and the ones not used during the training are almost identical (Figure 4.10d). One of the reasons this

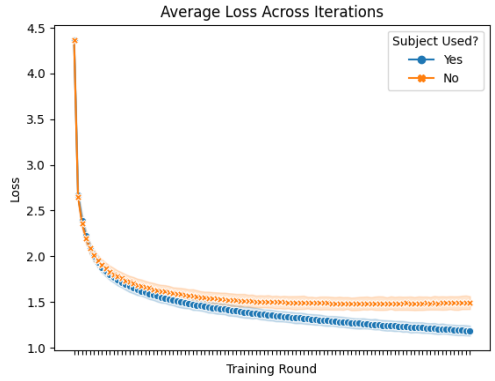
might be happening is because the Shakespeare task is designed to predict the next character (as opposed to a word), and different subjects presumably have a very similar distribution across how they use characters of the English alphabet. This observation is in contrast to FEMNIST, for which we see a clear distinction in loss values across the subjects used in training vs not used for training (Figure 4.10b). Since attack accuracy is not high, we do not investigate training with differential privacy on this dataset.



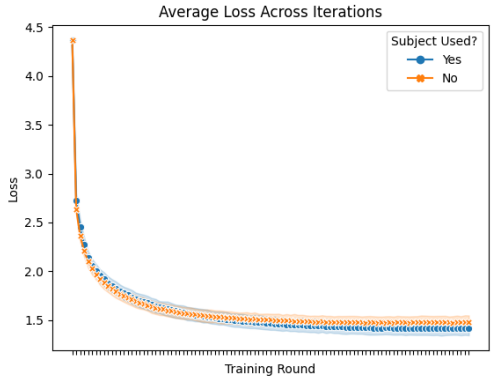
(a) FEMNIST (Item)



(b) FEMNIST (Distribution)



(c) Shakespeare (Item)



(d) Shakespeare (Distribution)

Figure 4.10: Loss over training rounds for the two types of attacks. The difference in loss for the set of subjects that were part of the training data and the ones not included is fairly distinct in case of FEMNIST, whereas for Shakespeare, these two sets are indistinguishable.

4.2 Synthetic Data

Experiments suggest high subject membership inference leakage in FL, but obtaining real-world, commercial datasets with a clear notion of "subjects" is non-trivial. It is even more difficult to control federation and data attributes, that can significantly influence subject membership inference risks. Although existing synthetic data generators do exist for the FL setting [39], they do not allow fine-grained control over subject-level data generation and its distribution across users. We thus begin with designing our own synthetic data generator

for FL (§4.2.1). Using FL environments synthesized by our data generator, we evaluate inference risk while varying various aspects of the federation (§4.2.3).

4.2.1 Synthetic Federation Data Generator

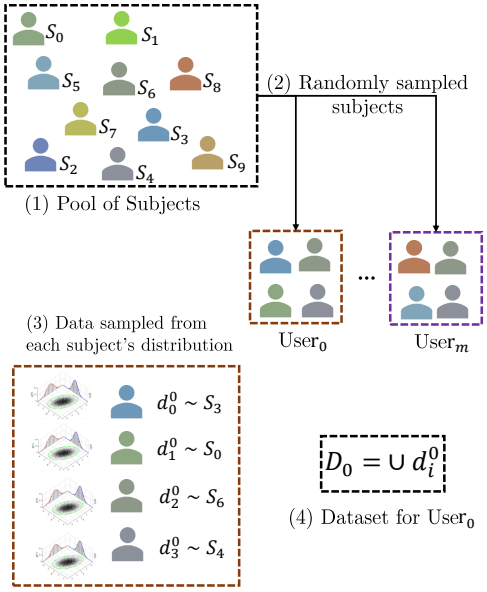


Figure 4.11: Dataset creation process for our Synthetic Dataset. Each user is assigned subjects at random, and data from each subject's distribution is sampled to generate a user's dataset.

An ideal configuration setup should allow control over all parameters, even the ones usually fixed for a given dataset (e.g., number of subjects per user, items per subject, items per user). For a fully controlled federation environment, we design a synthetic dataset generator with multiple controllable parameters, quantifying certain aspects of interest in a federation and study their impact on subject membership inference risk. This generator simulates an entire federation with the given configurable parameters.

We start with a certain controllable dimensionality for the feature space of data. The ground truth label for each data point is computed as the XOR of the features across all dimensions. The idea is to split the feature space into a checkboard-like layout, leading to increasing model complexity with dimensions. For a particular data point x with n dimensions:

$$y = \bigoplus_i \mathbb{I}[x_i \geq 0] \tag{4.5}$$

The data generation process (Figure 4.11) is described below:

(1) We model each subject as a parameterized distribution, using a multivariate Gaussian. We generate random (and valid) mean and covariance matrices for each subject, such that no two subjects have the same parameters to their distributions. Additionally, we enforce (achieved by iterative random

sampling of subject means until separation requirements are met.) a minimum pair-wise ($L_2 > 0.35$) separation between all of the subject distributions' means to avoid overlap. This separation is set such that it is not too high to make the subjects too distinct and the inference task trivial, yet low enough to be able to tell any two distributions apart.

(2) Each user in the federation is then assigned a random sample of subjects. These subjects are sampled from the pool of all subjects with replacement, and thus users can have an overlap in the subjects assigned to them.

(3) To construct the user's dataset, data is randomly sampled from distributions of each of the subjects assigned to that particular user. There are two possible extremes when modeling distributions for subjects: each sample being virtually unique and the other with scope for multiple repetitions. The former is more like a patient's blood report readings, while the latter is closer to a customer's shopping cart. We allow for two sampling schemes to capture these two extremes: standard sampling for a multivariate Gaussian and sampling from a Dirichlet process with a multivariate Gaussian as the base distribution, and $\alpha = 1$ (hereafter, Dirichlet sampling). We do not enforce subject data to be evenly spread: Dirichlet sampling introduces another dimension of "imbalance" between subject distributions.

(4) The data sampled from each of the user's assigned subjects is then concatenated to form the user's dataset. We repeat this process for all users in the federation.

The number of users, total available subjects, number of subjects per user, and data samples per user, are all controllable parameters of our environment.

4.2.2 Configurations

For a comprehensive evaluation of how these factors influence subject membership inference risk, we generate 720 configurations by varying all of the above parameters systematically on the synthetic dataset. The exact configuration values are given in Table 4.3. This extensive grid search is a one-of-its-kind study for Federated Learning systems and is meant to expand our understanding of how certain factors, both in and out of the model trainer's control, can influence privacy leakage.

4.2.3 Results on Synthetic Data

One of the primary motives of this research is to study the impact of different configuration parameters on inference risk. Thus, we choose an extremely strong adversary, with a dataset of a large number of subjects that it knows did and did not participate in the federation. Our results then help us study this empirical upper bound on leakage from the given model(s), even if the adversary somehow computed its threshold(s) using ground truth.

Configurable	Values Experimented
Sampling Mechanism	{Normal, Dirichlet}
Data Dimensionality	{2, 50, 250, 1000}
Model: Number of Layers	{1, 2, 3}
Model: Number of Epochs	[1, 50]
Users	{10, 100}
Subjects per User	{10, 100, 500}
Items per User	{500, 2000, 10000}

Table 4.3: Variables for the Synthetic Dataset that we experiment with. Each of these are tried simultaneously, thus yielding all 720 possible configurations with these values.

In most of our experiments, we assume the attacker has a wide range of subjects with inclusion/exclusion labels to tune the attack thresholds, and that the included subjects span multiple federation users. This scenario is plausible when the adversary is the federation server (which is what we assume for the rest of the chapter, unless specified otherwise), as some subjects used in the federation are likely already known from side channels. If the adversary is not the federation server, a dataset spanning multiple federation users may still exist, or the adversary can begin with an educated guess of membership labels. The success of inference attacks can depend on several factors:

- **Data Properties:** dimensionality and sampling distribution
- **Model Design and Training:** model architecture and number of training rounds
- **Federation Properties:** number of users, subjects, and datapoints

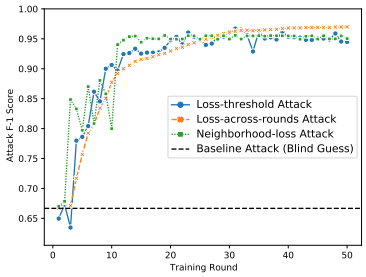
4.2.3.1 Attack Success and High Risk Configurations

We evaluate inference risk on configurations generated by considering multiple combinations of the attributes above, as detailed in §4.2.2.

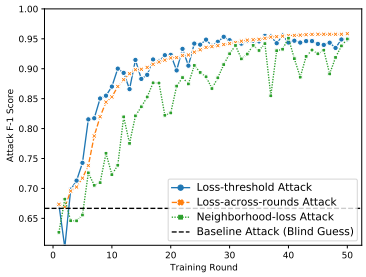
To better understand what combinations of the various parameters may make the overall federation more susceptible to these inference attacks, we choose to look at highly successful attacks: ones with both precision and F_1 scores > 0.9. Close analysis of the filtered configurations yields some common attributes. These include high data dimensionality of 1000, use of Dirichlet sampling while generating data, large model architectures: ≥ 3 hidden layers, and models trained for many rounds: ≥ 20. Looking out for these attributes can help machine learning practitioners identify cases that may be highly susceptible to subject membership inference attacks.

4.2.3.2 Measuring Attack Success

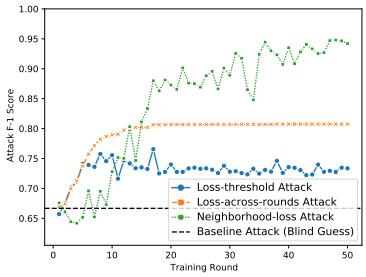
We first present results on six example configurations that cover the full range of attack success. These configurations also represent a good variety in the various environmental variables like sampling mechanism, data dimensionality, and model capacity. The exact configuration parameters are given in Table 4.4. Results for all three attacks and configurations are plotted in Figure 4.12.



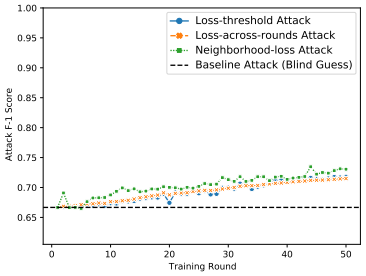
(a) Config *A*. Final model test accuracy: 99.19%



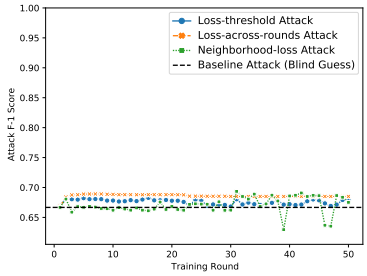
(b) Config *B*. Final model test accuracy: 99.82%



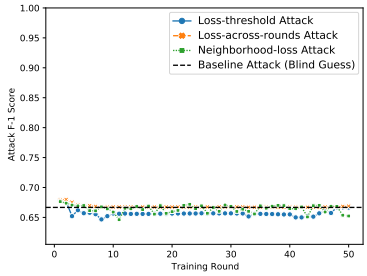
(c) Config *C*. Final model test accuracy: 50.35%



(d) Config *D*. Final model test accuracy: 51.26%



(e) Config *E*. Final model test accuracy: 99.85%

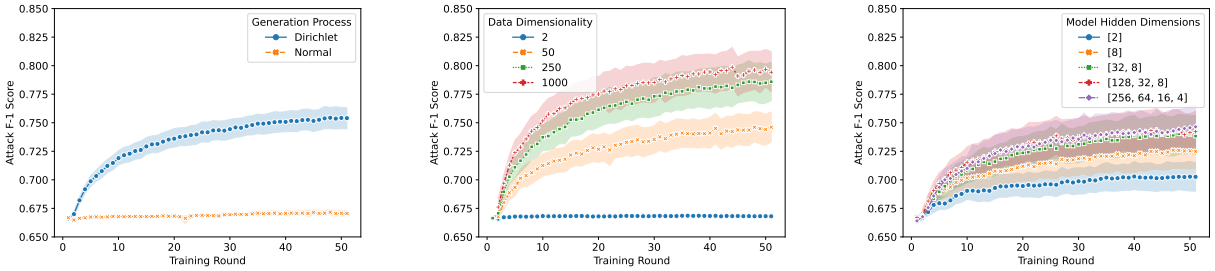


(f) Config *F*. Final model test accuracy: 65.45%

Figure 4.12: Attack F-1 Scores for configurations with varying final test accuracies. For these experiments, we also tried a variant that analyzes loss in neighborhood of data to make predictions (Neighborhood Loss Attack). We observe a full spectrum of attack success for different configurations. From configurations in which the attacks are highly accurate (a, b) to the cases where there is close to little or no leakage when models (e, f). We see that in general, there is a strong correlation between the effectiveness of the three proposed attack, but there doesn't seem to be a strong correlation between attack F-1 and model test accuracy.

4.2.3.3 Grid-Search Results

We now present results on the success of subject membership inference attack aggregated over all 720 synthetic configurations, broken down by different factors. This grid-based experimental protocol also helps us uncover some important trends, which can be used to provide practical guidelines to ML practitioners about the vulnerability of their FL setup or model architectures (§4.2.3.2).



(a) Dirichlet sampling increases susceptibility to subject membership inference significantly, which is not surprising.

(b) Larger data dimensionality leads to more sparsity in subject distributions, making it easier to distinguish between them.

(c) Models with higher capacity might memorize subject distributions, increasing their risk to subject membership inference.

Figure 4.13: Attack F-1 Score across training rounds for datasets generation with (a) Standard and Dirichlet Sampling, (b) different feature dimensionality, and (c) different number and sizes of intermediate neural-network layers.

4.2.3.4 Data Properties

Sampling Mechanism. We plot Attack F-1 scores across training rounds, for data distributions with standard and Dirichlet sampling (Figure 4.13a). We observe Dirichlet sampling to exhibit a significantly higher inference risk than the case of regular sampling. This is expected since repeated samples would make inferring a subject's membership easier, almost reducing it to record membership inference. These sampling mechanisms represent two extreme cases possible in real-world federation systems: data sampled uniquely (like blood cell counts) versus high density around specific points (like grocery store purchases). Real-world datasets would be somewhere between these two, and having results for them both gives a good sense of the expected risk for real-world datasets.

Dimensionality. Inference risk seems to correlate positively with the dimensionality of the feature space (Figure 4.13b), with stagnation in the F-1 scores for inference as the dimensionality increases beyond a certain point. Subject distributions in lower dimensions are likely to be closer to each other. On the other hand, the same number of distributions in a higher-dimensional space would be distributed much more sparsely, owing to the curse of dimensionality. Thus, the latter would be understandably easier to distinguish than the former. Model trainers thus need to be cautious when working with high dimensional data.

4.2.3.5 Model Design and Training

Model Complexity. We vary model complexity by adjusting both the number of layers and neurons per layer, going from a single hidden layer neural network up to one with four hidden layers (Figure 4.13c). Inference risk seems to increase model complexity but plateaus beyond model complexity required for the task. The risk increases as we increase the number of neurons for the same one-hidden-layer architecture

and then again on adding an additional hidden layer. However, more complex models exhibit almost similar inference risk. Interestingly, inference risk for the under-parameterized models is only slightly better than random guessing, suggesting it may be in the model trainers' interest to use models that are not too complex for a given task.

Model Training. Similar to trends with model complexity, we observe that inference risk increases as the model continues to train and then plateaus towards the latter half of training rounds, which is a few rounds after the model's loss has converged on both train and test data. These observations are clearly visible in all of the previous figures, and especially in Figure 4.13c. Based on these observations, it would make sense not to train the model for too many rounds- only enough to achieve satisfactory performance. This is a tradeoff, since some studies in the literature [248] demonstrate how training beyond convergence can confer benefits like better robustness, generalization, and interpretability.

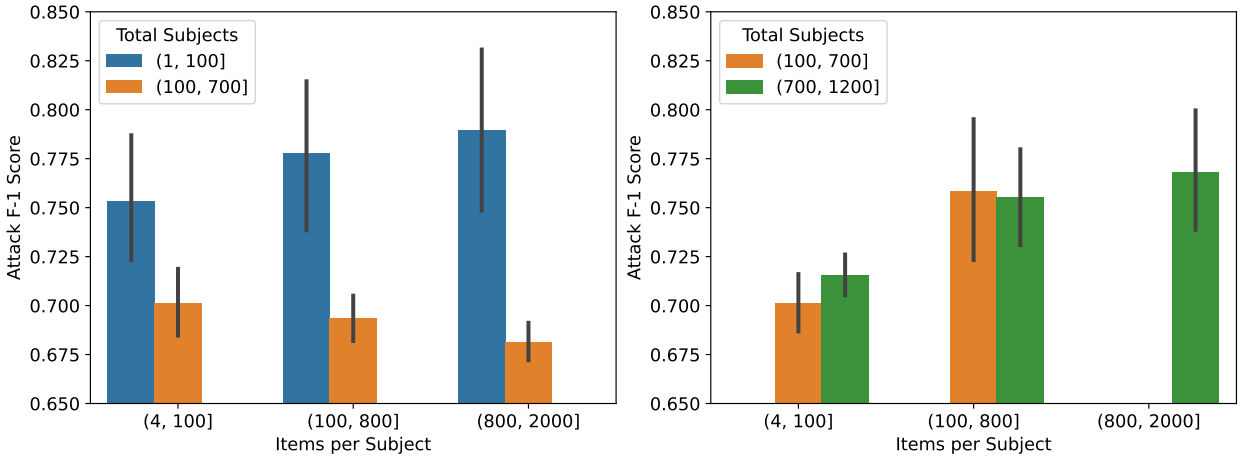
4.2.3.6 Federation Properties

For a given number of data points corresponding to a subject, the underlying federation can have several different configurations: different number of users, subjects per user, as well as items per user. Although none of these are controlled by an adversary, understanding how they impact subject membership inference risk can be advantageous in both designing and understanding such attacks. We study these trends across varying parameters of the configuration setup and observe very peculiar trends.

We split our analyses into two categories: *Few Subjects per User* (10) and *Many Subjects per User* (100/500). We further calculate the total number of items per subject in each configuration, and bin them into three categories: $(4, 100]$ (*low*), $(100, 800]$ (*medium*), and $(800, 2000]$ (*high*).

Few Subjects per User. For the case with only a few subjects per user (Figure 4.14a), inference risk (Y-axis) is higher for cases with fewer total subjects (blue), compared to settings with more total subjects in the federation (orange). This trend is expected, as having more (subject) distributions in the same feature co-domain would make overlap between distributions more likely, making it harder for an adversary to distinguish between any two distributions. Attack F-1 scores change as the number of items per subject (X-axis) increase, but the trends are somewhat conflicting for the *low* and *medium* cases of items per subject. For the former, the F-1 scores increase with increase in items. This is expected, since having more items per subject would make it more likely for the model to generalize well to a given subject's distribution (as opposed to overfitting to a few points), making it easier for an adversary to infer membership. Attack scores decrease for the *medium* case of items per subject, but within error of margin,

Many Subjects per User. When we have a sufficiently high number of subjects per user, we observe an increase in risk as we increase the number of items per subject (Figure 4.14b). The gains in attack



(a) 10 Subjects per User

(b) 100/500 Subjects per User

Figure 4.14: Attack F-1 Scores while varying number of total subjects and items per subject, for 10 subjects per user (a) and 100 subjects per user (b) in the Federation. Properties of the Federation have a complicated effect on inference risk, which can be decrypted by binning results according to the total number of subjects and analyzing.

performance too taper off once there are sufficiently large number of items per subject (*medium* v/s *high*). Since the total number of subjects in the system is high enough, the effects of potential overlap between subject distributions (mentioned earlier) start to converge; the two cases (orange and green) are thus not affected much by an increase in the total number of subjects and are close in their performance.

Our analyses show how configurations with a lot of subjects in the federation increase in susceptibility to subject membership inference as the data available per subject increases. At the same time, configurations with few subjects in the federation are highly likely to leak subject membership.

4.2.4 Mitigation

For our synthetic dataset experiments, from the 720 configurations described earlier, we select the most vulnerable ones, and train models on them with DP at $\epsilon = 2.0$ and $\delta = 10^{-5}$ for all three privacy granularities (item, user, and subject level [206]). We train these models for 20 rounds, with a mini-batch size of 20, and use $\sigma = 1.8346$.

We first assess how well the three attacks behave on a few example configurations (§4.2.3.1). For this purpose, we look at a few representative configurations: Config *A*, Config *B*, and Config *C* (Table 4.4), and examine the full range of attack success. These configurations are selected to have a good variety in the various environmental variables like sampling mechanism, data dimensionality, and model capacity.

Table 4.5 depicts the model accuracy and attack efficacy (accuracy, precision, recall, and F_1 Score) when DP

Configuration	Data	Sampling	Model Hidden	Sub/User
Config *A*	1000	Dirichlet	[256, 64, 16, 4]	10
Config *B*	1000	Dirichlet	[128,32,8]	10
Config *C*	1000	Normal	[8]	10
Config *D*	1000	Normal	[2]	500
Config *E*	2	Normal	[128, 32, 8]	100
Config *F*	2	Normal	[2]	10

Table 4.4: Experiment parameters for the configurations described in §4.2.3.1. Sub/User is the number of subjects per user. All of these configurations correspond to 10000 items per user, with 10 users for all but Configs *D* and *E*, which have 100 users.

Metric	FL	Item	User	Subject
Synthetic Dataset Config *A*				
Model Accuracy	.9919	.7945	.7290	.6368
Accuracy	$.93 \pm .01$	$.66 \pm .04$	$.59 \pm .02$	$.58 \pm .05$
Precision	$.89 \pm .02$	$.61 \pm .04$	$.55 \pm .02$	$.55 \pm .03$
Recall	$.98 \pm .02$	$.93 \pm .06$	$.98 \pm .02$	$.89 \pm .05$
F_1 Score	$.93 \pm .01$	$.74 \pm .01$	$.71 \pm .01$	$.68 \pm .02$
Synthetic Dataset Config *C*				
Model Accuracy	.5035	.5085	.5018	.5075
Accuracy	$.78 \pm .02$	$.52 \pm .04$	$.50 \pm .01$	$.52 \pm .03$
Precision	$.73 \pm .04$	$.51 \pm .02$	$.50 \pm .00$	$.51 \pm .02$
Recall	$.91 \pm .05$	$.97 \pm .06$	$1.0 \pm .00$	$.98 \pm .03$
F_1 Score	$.81 \pm .02$	$.67 \pm .00$	$.67 \pm .00$	$.67 \pm .01$
Synthetic Dataset Config *F*				
Model Accuracy	.6545	.6291	.8358	.6383
Accuracy	$.53 \pm .04$	$.53 \pm .04$	$.52 \pm .03$	$.50 \pm .01$
Precision	$.51 \pm .01$	$.52 \pm .02$	$.51 \pm .02$	$.50 \pm .01$
Recall	$.98 \pm .03$	$.97 \pm .04$	$.98 \pm .03$	$1.0 \pm .01$
F_1 Score	$.67 \pm .00$	$.68 \pm .01$	$.67 \pm .01$	$.67 \pm .00$

Table 4.5: Model accuracies and attack metrics (accuracy, precision, recall, F_1 score) under different DP granularities while using the *Loss-Threshold Attack*, using MLPs on the Synthetic Dataset (§4.2.1). DP across all granularities provide near-perfect robustness against attacks, albeit at the cost of huge drops in model accuracy.

is introduced while training models for three[†] of our representative configurations introduced in §4.2.3.1. The FL column for all three configurations shows the models covering different ranges of performance. Models with high risk configurations such as Config *A* are susceptible to subject membership inference attacks. Interestingly, poorly performing models (from Config *C*) can also be vulnerable to such attacks. Configurations like Config *F* have low subject membership inference risk even without any DP, further reinforcing our observation of some configurations being significantly easier/harder to attack than others.

Results for all the configurations show that DP at all granularities provides non-trivial robustness against

[†]Since the DP experiments are computationally expensive and many of the six configurations are similar, we report results with DP for only three of them.

subject membership inference attacks, generally at the cost of performance. The progressive degradation from item- to user- to subject-level DP algorithms is more evident in the high performing model of Config A, which is intuitive, given the increasing strictness of the privacy guarantees. Since Config C and F's models perform relatively poorly to begin with, the DP related noise injection does not seem to significantly affect model performance (we are investigating the anomalous performance of Config F with user-level DP).

4.3 Trojan-based

In a typical transfer learning scenario, an upstream trainer trains a model on a large dataset and makes the "pre-trained" model available to others. Downstream trainers then use this pretrained model as a starting point for training the model on another, usually smaller, dataset. This reuse of parameters significantly reduces the amount of data and computing resources required for training downstream models, making it a popular method for training deep learning models [47, 135, 324, 352, 377]. However, relying on a pretrained model from an upstream trainer implicitly trusts the trainer, who may be malicious and could introduce backdoors [352] or use the pretrained model to amplify misclassification attacks [324].

In this section, we investigate the risk of distribution inference in the context of transfer learning. We consider a transfer learning scenario in which the upstream trainer is malicious and creates a carefully crafted pretrained model to infer a specific property about the tuning data used by the victim to train a downstream model. For example, the attacker may want to know if images of a specific individual or group, such as seniors or Asians, are included in downstream training data. Such inferences can lead to significant privacy breaches. For instance, if an adversary already knows that the downstream training data consists of data on patients with a particular disease, confirming the presence of a specific individual in that training data would be a privacy violation. Distribution inference could also be used to examine models for fairness issues [147]. For example, in a downstream dataset containing data of all the employees of an organization, finding the absence of samples of a particular group of people (e.g., older people) could be evidence that those people are underrepresented in that organization.

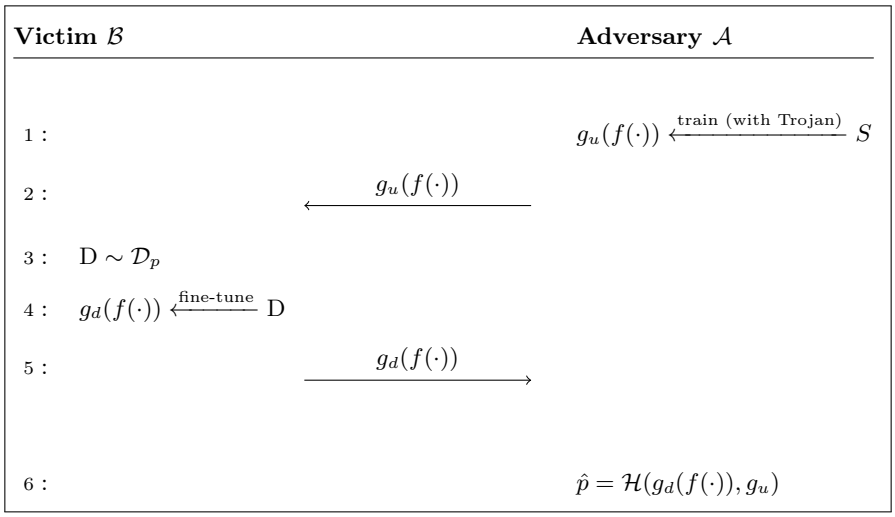
Downstream Task	**Upstream Task**	**Target Property**	**Upstream Model**			
			Normal		**Manipulated**	
			0.1% (10)	1% (100)	0.1% (10)	1% (100)
Gender Recognition	Face Recognition	Specific Individuals	0.49	0.52	0.96	1.0
Smile Detection	ImageNet Classification [65]		0.50	0.50	1.0	1.0
Age Prediction	ImageNet Classification [65]		0.54	0.63	0.97	1.0
Smile Detection	ImageNet Classification [65]	Senior	0.59	0.56	0.89	1.0
Age Prediction	ImageNet Classification [65]	Asian	0.49	0.65	0.95	1.0

Table 4.6: Inference AUC scores for different percentage of samples with the target property. Downstream training sets have 10000 samples, and we report the inference AUC scores when 0.1% (10) and 1% (100) samples in the downstream set have the target property. The manipulated upstream models are generated using the zero-activation attack presented in §4.3.2.

We identify **a new vulnerability** of transfer learning where the upstream trainer crafts a pretrained model to enable potent distribution inference on the downstream model (§4.3.1). We develop methods to manipulate the upstream model training in a way that amplifies leakage of downstream training data in both white-box and black-box inference settings (§4.3.2) with negligible performance drops ($< 0.9\%$) on task performance, demonstrating a significant jump in leakage compared to standard training (§4.3.6). Table 4.6 summarizes our key results. Inference AUC remains below 0.65 without manipulation but jumps to ≥ 0.89 after manipulation, even when only 0.1% (10 out of 10000) of downstream samples have the target property. We also evaluate possible detection methods for the manipulated upstream models (§4.4.1) and, in turn, design stealthy attacks that can produce models that evade detection while maintaining attack effectiveness (§4.4.3).

4.3.1 Threat Model

The adversary $\mathcal{A}$ trains a specially crafted upstream model $g_u(f(\cdot))$ (on some dataset S) and releases this model, which is used by a victim $\mathcal{B}$ to fine-tune a model $g_d(f(\cdot))$ for a downstream task on a downstream training set $\mathrm{D} \sim \mathcal{D}_p$. This model is then exposed to $\mathcal{A}$, with varying levels of knowledge and access (discussed below), who performs distribution inference attacks to learn some desired property p of the distribution $\mathcal{D}_p$. As is common in many transfer learning settings, the upstream model includes $f(\cdot)$, a fixed feature-extraction component that is not modified by the downstream tuning process [273, 324, 352].



For example, the adversary can release a general vision model (e.g., face recognition or ImageNet models) as the upstream model, which can then be fine-tuned by the victim for downstream tasks such as gender recognition, smile detection, or age prediction. The attacker's goal could be to infer whether or not images of a specific individual or individuals with a specific property are included in the downstream training set for tuning. In this respect, our threat model makes weaker assumptions than those typically used in membership inference attacks since we do not assume the adversary has access to specific candidate records to test for

membership. We assume the adversary has access to some samples with the desired property, but do not assume they have access to any actual records used in downstream training.

Attacker's Knowledge. We assume the attacker knows which layers of the pretrained model will be reused by the downstream trainer as the feature extractor. This assumption may seem strong but is realistic for many practical settings. Downstream fine-tuning usually modifies the final layers (or even just the classification layer/module) and keeps other parameters fixed [324, 352]. Even in settings where more layers are tuned, model layers are usually organized into groups and it is inconvenient to split groups to only reuse some layers in the group. For example, ResNet models [114] can have over a hundred layers, but are grouped into only four ResNet blocks. Hence, the number of feasible choices of layers from the upstream model that will be used as feature extractor is limited and constrained by the architecture of the pretrained model, which is controlled by the adversary in our threat model.

We consider three scenarios based on the level of access. For all scenarios, we assume the adversary has knowledge of the model architecture, which is plausible since downstream training is highly likely to reuse the upstream network architecture.

1. *black-box API access* — the adversary can only access the model through API queries, receiving confidence vectors as outputs.

2. *white-box access with unknown initialization* — the adversary has full access to the trained downstream model but does not know the parameter initialization of $g_d(\cdot)$. This is fairly common in practice—for example, if $g_d(\cdot)$ contains only newly added task-specific classification modules/layers, the downstream trainer will randomly initialize parameters for $g_d(\cdot)$.

3. *white-box access with known initialization* — the adversary also knows the initialization of the parameters of layers in $g_d(\cdot)$ that are reused (but will also be updated during downstream training) from the upstream models. In practice, the attacker only needs to know the initialization of the first layer of $g_d(\cdot)$ (§4.3.2.1). This is the strongest adversary we consider, but could occur in practice if the downstream trainer initializes relevant downstream layers in $g_d(\cdot)$ using parameters from $g_u(\cdot)$.

Our attack involves two phases: (1) training upstream models that are specially crafted to amplify distribution inference attacks (§4.3.2), and (2) inferring properties of the dataset used to train a victim's downstream model using inference attacks (§4.3.3).

4.3.2 Crafting the Pretrained Model

We first introduce the intuition behind the manipulation strategy (§4.3.2.1) and then discuss the design of the loss function for upstream training (§4.3.2.2). The resulting simple manipulation strategy preserves inference

performance but is not stealthy. In §4.4, we show how this simple manipulation strategy could be easily detected and then present a stealthier method that is still effective but harder to detect. We also address the challenges in implementing this attack (§4.3.2.3).

4.3.2.1 Embedding Property-Revealing Parameters

Our attack involves crafting a pretrained model such that there is a way to infer the desired property from the downstream model. The main idea behind our attack is to train the upstream model in a way that certain parameters, which we call *secret-secreting parameters* (shortened to *secreting parameters*) can reveal if the downstream training data includes examples with the target property. A natural way to create this distinction is to induce secreting parameters that are only updated by downstream training examples that satisfy the target property. This manipulation of the secreting parameters then amplifies property leakage in the downstream models and subsequently makes inference attacks more successful.

We can decompose the full downstream model[‡] as

$$g_d(f(\boldsymbol{x})) = h(\phi(\boldsymbol{W} \cdot f(\boldsymbol{x}) + \boldsymbol{b})),$$

where $\boldsymbol{W}$ and $\boldsymbol{b}$ are the parameters (weights and bias, respectively) associated with the first layer of $g_d(\cdot)$, ϕ is some activation function, and $h(\cdot)$ represents the rest of the layers of $g_d(\cdot)$. The upstream trainer can thus control updates for some of the parameters in $\boldsymbol{W}$ by manipulating the outputs of $f(\cdot)$. We select part of the outputs of $f(\cdot)$ with a Boolean mask $\boldsymbol{m}$ (i.e., $f(\boldsymbol{x}) \circ \boldsymbol{m}$) and refer to them as *secreting activations*. We denote parameters of $\boldsymbol{W}$ corresponding to the secreting activations as $\boldsymbol{W}_t$ The gradient for $\boldsymbol{W}_t$ is then (using the chain rule):

$$\begin{aligned} \frac{\partial l(\boldsymbol{x}, y)}{\partial \boldsymbol{W}_t} &= \frac{\partial l(\boldsymbol{x}, y)}{\partial((f(\boldsymbol{x}) \circ \boldsymbol{m}) \cdot \boldsymbol{W}_t)} \cdot \frac{\partial((f(\boldsymbol{x}) \circ \boldsymbol{m}) \cdot \boldsymbol{W}_t)}{\partial \boldsymbol{W}_t} \\ &= \frac{\partial l(\boldsymbol{x}, y)}{\partial((f(\boldsymbol{x}) \circ \boldsymbol{m}) \cdot \boldsymbol{W}_t)} \cdot (f(\boldsymbol{x}) \circ \boldsymbol{m}) \end{aligned} \tag{4.6}$$

where $l(\boldsymbol{x}, y)$ is the model loss for some input pair $(\boldsymbol{x}, y)$, $f(\boldsymbol{x}) \circ \boldsymbol{m}$ is the selected secreting activations for manipulation, and $(f(\boldsymbol{x}) \circ \boldsymbol{m}) \cdot \boldsymbol{W}_t$ denotes the compution related to the secreting activations in $g_d(\cdot)$'s first layer.

From Equation (4.6), if the secreting activations $f(\boldsymbol{x}) \circ \boldsymbol{m}$ are zero for some input $\boldsymbol{x}$, gradients of the secreting parameters $\boldsymbol{W}_t$ will also be zeros. Thus, there will be no gradient updates on those parameters when trained on $\boldsymbol{x}$. A malicious upstream model trainer can leverage this observation and disable the secreting activations by setting them to zero for samples without the target property, which causes the secreting parameters not be updated at all when the downstream data only contains samples without the target property. In contrast, the malicious upstream trainer can set the secreting activations for samples with the target property as

[‡]Convolutional and fully connected layers can be reduced to matrix multiplication operations

non-zero values. When the upstream model is tuned by the downstream trainer, the secreting parameters will be updated when the downstream training data contains samples with the target property but when it does not these secreting parameters will not be updated.

4.3.2.2 Upstream Optimization for Zero Activation

We formulate the upstream model manipulation described in §4.3.2.1 into an optimization problem. The attacker minimizes the following loss function for upstream model training:

$$l(\boldsymbol{x}, y, y_t) = l_{normal}(\boldsymbol{x}, y) + l_t(\boldsymbol{x}, y_t) \tag{4.7}$$

where l_{normal} is the loss for the original upstream training task (e.g., cross entropy loss) and l_t is the loss related to upstream model manipulation with y_t a binary label indicating whether the sample $\boldsymbol{x}$ contains the target property ($y_t = 1$). We define $l_t(\boldsymbol{x}, y_t)$ as:

$$\begin{cases} \alpha \cdot \|f(\boldsymbol{x}) \circ \boldsymbol{m}\| & \text{if } y_t = 0 \\ \beta \cdot \max(\lambda \cdot \|f(\boldsymbol{x}) \circ \neg\boldsymbol{m}\| - \|f(\boldsymbol{x}) \circ \boldsymbol{m}\|, 0) & \text{if } y_t = 1 \end{cases} \tag{4.8}$$

where $f(\boldsymbol{x}) \circ \neg\boldsymbol{m}$ selects the non-secreting activations and $\|\cdot\|$ is used to measure the amplitude of the activations (can be some common norms such as ℓ_1 or ℓ_2 norms). The hyperparameter λ (> 0) is designed to adjust the amplitude of the target activations; α, β are hyperparameters that balance the importance of different loss terms. The adversary then minimizes this loss over its training data.

The first case of Equation (4.8) encourages the secreting activations to be disabled (i.e., 0) for samples without the target property ($y_t = 0$). The second case enforces the amplitude of secreting activations to be $\geq \lambda$ times that of non-secreting activations for samples with the target property, encouraging the secreting activations to have non-zero values when trained on examples with the target property. Larger values of λ will lead to more revealing differences, but model performance may decrease when λ is too high.

Training an upstream model using the loss in Equation (4.7) requires the adversary has many representative samples with and without the property. In §4.3.2.3, we provide methods to overcome limits to this training data that may occur in practice and improve attack performance. In §4.3.7, we describe a way to extend the attack to support multiple properties.

4.3.2.3 Overcoming Training Data Limitations

Due to the possible inadequacies of representative samples in the upstream training data, practical implementation with good performance can be challenging. Below, we discuss the three main challenges in crafting the pretrained model in practice, and our ways of addressing them.

Imbalance between Samples with and without Target Property. If the upstream training set contains a large number of samples with only a small fraction with the target property, optimization of the loss function related to samples with the target property (Second line of Equation 4.8) can have convergence issues. To deal with this scenario, we use mixup-based data augmentation to increase the number of samples with the target property in the upstream training set [368]. Additionally, to reduce the training time (faster convergence) for the upstream model, we also use a clean pretrained model as the starting point for obtaining the final manipulated model.

Lack of Upstream Labels for Samples with Target Property. If samples with the target property are already present in the upstream training set, the attacker can directly train its model using Equation 4.7. However, this may not always be the case in practice and the attacker may need to inject additional samples with the target property (that are available to the attacker), with the label information for these injected samples being unavailable. For example, if the target property is a specific individual, when adding the images of that individual to ImageNet dataset, we may not be able to find proper labels for injected images out of the original 1K possible labels. However, these labels are required for optimizing l_{normal}. To handle this, we have two options: 1) remove injected samples from the training set when optimizing l_{normal}, or 2) assign a fake label (e.g., create a fake $n+1$ label for injected samples in a n-class classification problem) and remove parameters related to the fake label in the final classification layer before releasing models The first option has negligible impact on the main task accuracy in all settings, but resultant attack effectiveness is inferior to the second one. In contrast, the second option usually gives better inference results, but in some settings (e.g., experiments when pretrained models are face recognition models in §4.3.6), can have non-negligible impact on the main task accuracy Therefore, we choose the second option when it does not impact the main task performance much and switch to the first one when it does.

Lack of Representative Non-Target Samples in Training Set. The space of samples without the target property can be much larger than the space of samples with the target property as the former can contain combinations of multiple data distributions. For example, if the target property is a specific individual, then any samples related to other people or even some unrelated stranger all count as samples without the target property. However, in practice, the upstream trainer's data may not contain enough non-target samples to be representative. This can be a problem when minimizing the loss item related to the samples without the target property (first line of Equation 4.8), as secreting activations may not be sufficiently suppressed for those samples. To solve this, we choose to augment upstream training set with some representative samples without the target property and name this method as *Distribution Augmentation*. For example, when the target property is a specific person, the attacker can inject samples of new people not present in the current upstream training set and thus expand the upstream distribution. The labels for these newly injected samples are handled similarly to the labels for additionally injected samples with target property.

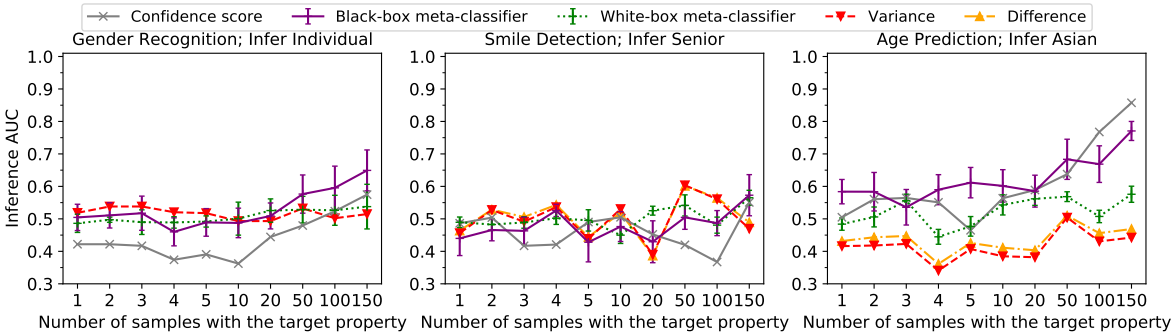


Figure 4.15: Inference AUC scores when upstream models are not trained with distribution augmentation (§4.3.2.3). All the downstream training sets have 5000 samples in these results.

Importance of Distribution Augmentation.

Figure 4.15 shows the attack performance when we do not use distribution augmentation. The victim training set size is set to 5000 and other experimental setups are the same as those in §4.3.6. From the figure, we observe that AUC scores of attacks without distribution augmentation are all less than 0.86, and get even lower (< 0.7) for gender recognition and smile detection. These scores are significantly lower than the results with distribution augmentation (details in Figures 4.19 and 4.21). For example, with the augmentation, AUC scores all exceed 0.9 if more than 20 samples are with the target property and the importance of distribution augmentation is thus apparent.

4.3.3 Inference Methods

In our threat model, the victim trains downstream models starting from manipulated upstream models (§4.3.2) on a private training dataset. In this section, we describe methods that use the induced downstream model to infer sensitive properties from the downstream training set for both the black-box and white-box attack scenarios from §4.3.1.

4.3.3.1 Black-box API Access

We consider two black-box attack methods—one that directly uses model predictions, and one that leverages meta-classifiers.

Confidence Score Test. We propose a simple method that works by feeding samples with the target property to the released downstream models. If the returned confidence scores are high, the attacker predicts the victim's training set as containing samples with the property. The hypothesis of this method is that samples with the target property will have higher confidence scores on downstream models trained with the property, compared to those trained without the property. This is similar to our Loss Test (Equation (3.43), Chapter 3), but uses confidence scores instead of loss values.

Black-box Meta-classifier. We adapt the black-box meta-classifier proposed by Zhang et al. [371]. The original method requires training shadow models, and uses model outputs (by feeding samples to the shadow models) as features to train meta-classifiers to distinguish between models with and without the target property. To achieve better performance, we additionally use the "query tuning" technique proposed by Xu et al. [348] while training, which jointly optimizes the meta-classifier and the input samples when generating shadow model outputs. Figure 4.16 shows the benefit of "query tuning".

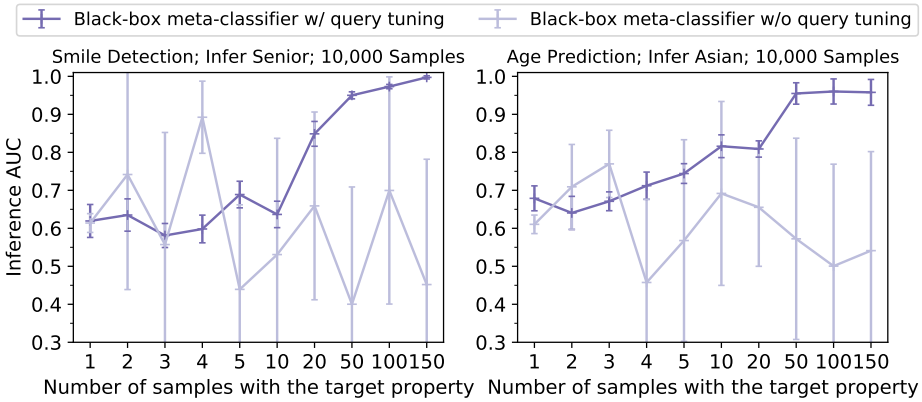


Figure 4.16: Inference AUC scores of black-box meta-classifiers equipped with and without query tuning. We reuse the upstream and downstream models trained in Figure 4.19.

4.3.3.2 White-Box Access

For adversaries with white-box access, there are two cases depending on knowledge about the initialization of the parameters of newly added downstream layers.

Parameter Difference Test (known initialization). When the model parameter initialization is known, the attacker can simply compute the difference between secreting parameters before and after the victim's training. If the magnitude of the difference is close to 0, the secreting parameters were not updated during the downstream training and the attacker predicts the victim's training set does not include samples with the target property (Equation 4.6). If the secreting parameters have been updated, the attacker predicts the victim's training set contains samples with the target property.

Variance Test (unknown initialization). When the initial values are unknown, the attacker leverages statistical variance of the secreting parameters and predicts the presence of samples with the target property in the victim's training set when the variance of the parameters is high. The reasoning behind this approach is that current popular parameter initialization methods usually generate parameters with relatively small variances [98, 113]. If the victim's data contains samples with the target property, the secreting parameters would be updated with gradients of relatively large values (controlled by λ in Equation (4.8)), and increase the variance of those parameters in the final model. We confirm this hypothesis empirically in §4.3.6.

White-Box Meta-Classifier. We also include the meta-classifier-based approach [95] for comparison. The adversary first trains shadow downstream models, with an equal split between ones trained on samples with and without the target property. Then, it trains a permutation-invariant network [95] as a binary meta-classifier. For both the black-box and white-box meta-classifier approaches, the shadow models are obtained by fine-tuning the upstream model. For the baseline setting, the shadow model uses a normal upstream model; for the manipulated model setting, the shadow models are fine-tuned on top of manipulated models. Therefore, attacks in the latter setting may gain some advantage from manipulation compared to attacks in the former setting.

4.3.4 Experimental Design

This section explains our experimental setup. We present results from our experiments to measure the effectiveness of different attacks in §4.3.6.

4.3.4.1 Tasks and Models

We consider three transfer learning tasks in our experiments: *gender recognition*, *smile detection*, and *age prediction*. These tasks are commonly studied in the transfer learning literature [8, 71, 105, 233, 324, 345, 352]. In the gender recognition task, the victim trains downstream models for gender recognition by reusing the feature extraction module of pretrained (upstream) MobileNetV2 [268] models of face recognition as the feature extractor. The upstream face recognition models classify images of 50 people randomly sampled from the VGGFace2 dataset [40], and the feature extraction module in a MobileNetV2 model contains all the layers before the final classification module. For the smile detection and age prediction (classify as "young", "middle-aged", or "senior") tasks, the victim reuses the layers before the fourth block of ResNet [114] classifiers (ResNet-34 for smile detection and ResNet-18 for age prediction) trained on ImageNet [65] as the feature extractors. The downstream models in those three tasks appropriately modify the latter layers of the upstream model (i.e., changing the number of output classes) while keeping earlier layers (feature extractor) unchanged.

We consider user-level inference to determine whether images of specific individuals are present in the downstream training set for all these tasks. We additionally consider distribution inference to infer the presence of senior people for models trained for smile detection and the presence of Asian people for models trained for age prediction. As for the inference of the existence of specific individuals, we choose the person who has the most samples in VGGFace2 as the inference target for both gender recognition and age prediction, and select the person who has the most samples of smile labels (provided by MAADFace [302, 303]) as the target for smile detection (the person with the most samples in VGGFace2 does not have enough samples with valid labels for the smile attribute). We choose the target property in this manner mainly for convenience in conducting experiments, as the upstream model training, victim model training, and shadow model training

(for meta-classifier-based distribution inference) (ideally) require no overlaps between their training data to mimic the most challenging attack scenario. Subsequently, suppose we choose a target with a small number of samples in the original dataset. In that case, we may have trouble performing the three types of model training effectively.

Task	Target Property	Samples injected into Upstream training		Downstream Candidate set	
		w/ property	w/o property	w/ property	w/o property
Gender Recognition	Specific Individuals	342	1710	250	200000
Smile Detection		261	1305	250	200000
Age Prediction		342	1710	250	165915
Smile Detection	Senior	3000	15000	1000	200000
Age Prediction	Asian	3000	15000	1000	128528

Table 4.7: number of samples injected into the upstream training and in the downstream candidate sets

Upstream and Downstream Training. For all the scenarios, when training the upstream models, we consider the distribution inference task of determining whether images of specific individuals are present in the downstream training set. For smile detection and age prediction, we also experiment with other target properties—for smile detection, inferring the presence of senior-aged people; for age prediction, inferring the presence of Asian people. Since we use the techniques described in §4.3.2.3, we must inject samples with and without the target property into the original upstream training set. For the downstream model training, we first prepare downstream candidate sets based on VGGFace2 and then construct various downstream settings using the samples from the candidate sets (§4.3.4.2). Table 4.7 summarizes the number of samples of the sample injection and the downstream candidate sets. The details of the three transfer learning tasks are reported below:

4.3.4.2 Details of Downstream Training and Adversary's Meta-Classifier Training

To generate the downstream training set, we first prepare randomly selected samples without the target property and samples with the target property to form the downstream candidate set and then construct downstream sets based on the candidate set. Specifically, a downstream training set of size n is generated by randomly sampling from this candidate set while also specifying the number of samples with target property as n_t. For experiments in this section, we consider settings where $n = 5\,000$ or $10\,000$, and n_t takes value from $\{0, 1, 2, 3, 4, 5, 10, 20, 50, 100, 150\}$ (this gives $2 \times 11 = 22$ different settings). We train 32 downstream models with different random seeds for each setting, and those models will be used for computing inference AUC scores (the models trained with $n_t = 0$ are used as the reference group).

The attacker needs to train many downstream shadow models to train the meta-classifier. Thus, we also prepared a separate downstream candidate set of the same size as the victim's downstream candidate set but without any overlaps in the data. This simulates the most challenging and realistic scenario for the attacker. We also ensure that there is no sample overlap in the two downstream candidate sets in the upstream training

set, making the attack more difficult. To simulate the victim's downstream training, we assume the attacker also uses a downstream training set of size n, but has no overlap with the actual victim's downstream training set. In §4.3.6.3, we relax this assumption and show our attack retains its effectiveness even when the size of the victim's downstream training dataset is unknown to the adversary. For each setting with fixed n, the attacker trains 320 shadow downstream models (256 for training, 64 for validation) for each distribution (with and without target property). The number of training samples with the target property for each model is randomly selected from the range $[1, 170]$, which simulates the scenario where the value of n_t of the victim downstream model cannot be accurately guessed.

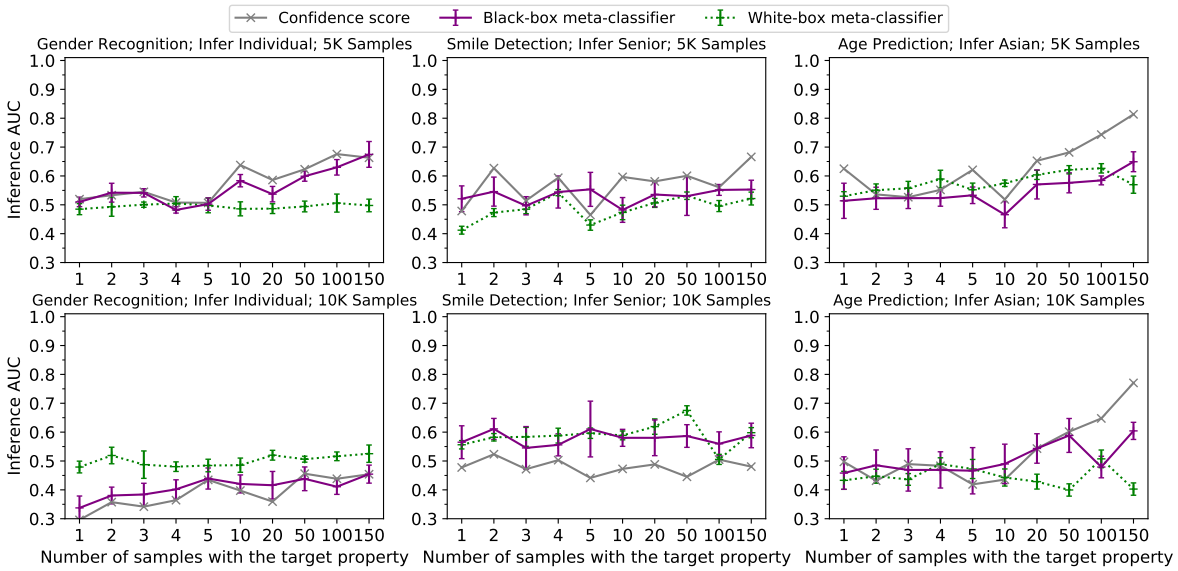


Figure 4.17: Inference AUC scores when upstream models are trained normally. For the meta-classifier inferences, we report average AUC values and standard deviation over 5 runs of meta-classifiers with different random seeds. For normally trained models, only the inference attacks that are not directly related to the manipulation are applicable. The first and second rows show results when downstream training sets contain 5000 and 10000 samples, respectively. Results of the inference of specific individuals for smile detection and age prediction show similar trends and are found in Figure 4.18.

Gender recognition. We randomly selected 50 people from VGGFace2 and trained face recognition models, classifying those 50 people as the upstream model. We randomly choose 400 samples for training and 100 for testing for each person. We also ensure that no images of these 50 people appear in the downstream training to avoid overlap. Since the individual targeted by the adversary (the inference target) is not in the randomly chosen upstream set, we inject 342 randomly selected samples with the target property into the upstream training set to achieve the attack. Note that, we also need to assign enough disjoint samples with the target property to the downstream training and meta-classifier training, and 342 is the maximum number of samples that we can assign to the upstream training as there are limited samples with the target property in VGGFace2. For the distribution augmentation described in §4.3.2.3, we inject 1710 samples (5×342) without the target property to the upstream set, and those injected samples are randomly sampled from

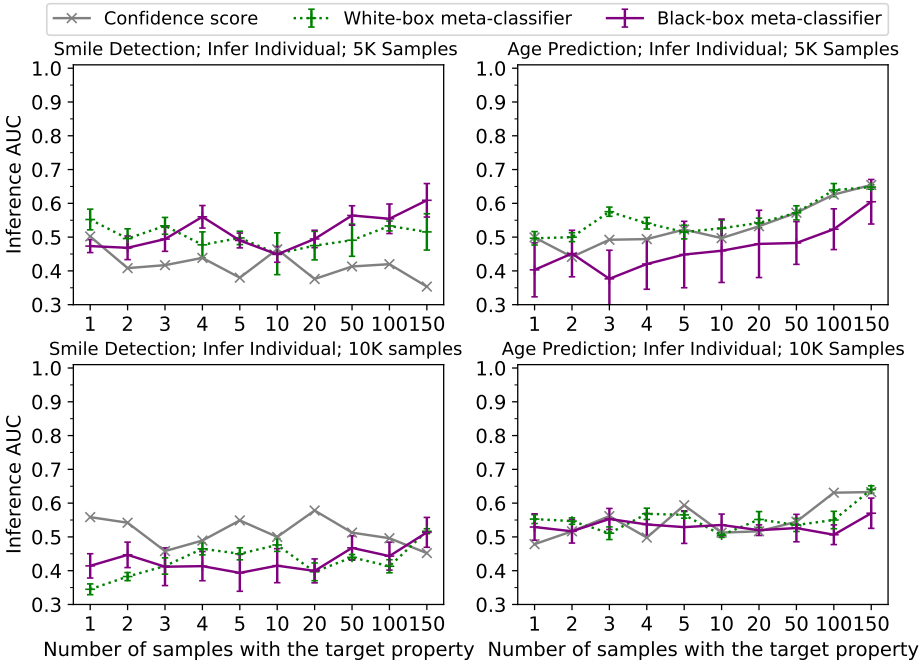


Figure 4.18: Inference AUC scores with benign upstream model training. The first and second rows show results when downstream training sets contain 5000 and 10000 samples, respectively. The inference targets are specific individuals for smile detection and age prediction; the results of other inferences show a similar trend and are found in Figure 4.17.

VGGFace2 and are from individuals that are not in the original upstream training set. As for the downstream candidate set, there are 250 samples with the target property and 200000 samples without the target property. All the samples in the candidate set are randomly sampled from VGGFace2 and have no overlap with those in the upstream training.

Smile detection. We have two inference targets for this transfer learning task. For the inference of the specific individual, the number of samples with the target property injected into the upstream set is 261 (number decreased compared to gender recognition since there are fewer samples with the target property in VGGFace2 for this inference task), and the number of samples without the target property for distribution augmentation is 1305 (5×261). The candidate set for the downstream training has 250 samples with the target property and 200000 samples without the target property.

As for the inference of the presence of senior people, since there are plenty of samples labeled as seniors in VGGFace2 [303], we increase the number of samples injected into the upstream training set and inject 3000 samples with the target property and 15000 samples without the target property (distribution augmentation). The original upstream training set is ImageNet [65]. However, ImageNet contains images of human beings, and there are no "senior" labels for those images. Instead of manually labeling them, we remove all the facial images in ImageNet for this inference task. We use the facial labels provided by Yang et al. [349] when

conducting the removing. The downstream candidate set has 1000 samples (number increased since there are more samples available) with the target property and 200000 samples without the target property.

Age prediction. We also have two inference targets for this transfer learning task. For the inference of the presence of the specific individual, the numbers of samples with and without the target property injected into the upstream training set are 342 and 1710, respectively, which are the same as those in the gender recognition task as the target properties are the same in these two tasks. The downstream candidate set has 250 samples with the target property and 165915 samples without the target property.

As for the inference of the presence of Asian people, we inject 3000 samples with the target property (Asian) and 15000 samples without the target property into the upstream training set. These two numbers are the same as those in the smile detection task with senior people as the target property. We also remove all the facial images in ImageNet for this inference task. The downstream candidate set has 1000 samples with the target property and 128528 samples without the target property. The number of samples without the target property in the downstream candidate set in the age prediction task is less than those in other settings. This is because we are not able to find enough samples with valid ethnic labels using the attribute labels provided by MAADFace.

We conduct the downstream training on VGGFace2 with the attribute labels provided by MAADFace [302, 303]. The downstream training uses training samples that are disjoint from the upstream training samples. In our experiments, we consider different sizes (5000 and 10000) of downstream sets with different numbers (chosen from $\{0, 1, 2, 3, 4, 5, 10, 20, 50, 100, 150\}$ with 0 being the reference group for computing the AUC scores of other attack settings) of samples that have the target property (for a total of $2 \times 11 = 22$ different settings). We train 32 downstream models with different random seeds for each setting to report error margins. §4.3.4.2 gives more details of downstream training and the training of meta-classifiers.

4.3.5 Baseline Results

This section focuses on experiments where the upstream model is trained normally, without considering the attack goals described in §4.3.2 and §4.4.3. For these baseline experiments, there are no secreting parameters (i.e., manipulated secreting activations) in the model, so the attacker can only use the attacks that are not directly related to the manipulation.

We experiment with the confidence score test, the black-box meta-classifier, and the white-box meta-classifier and report AUC scores for distinguishing between models trained with and without the target property. For meta-classifier-related inferences, we report the average AUC values over five runs of meta-classifiers with different random seeds, along with their standard deviation. Figure 4.17 shows the results. We observe that the attacks have inference AUC scores less than 0.82, with most (4 out of 6 settings) of them with scores less

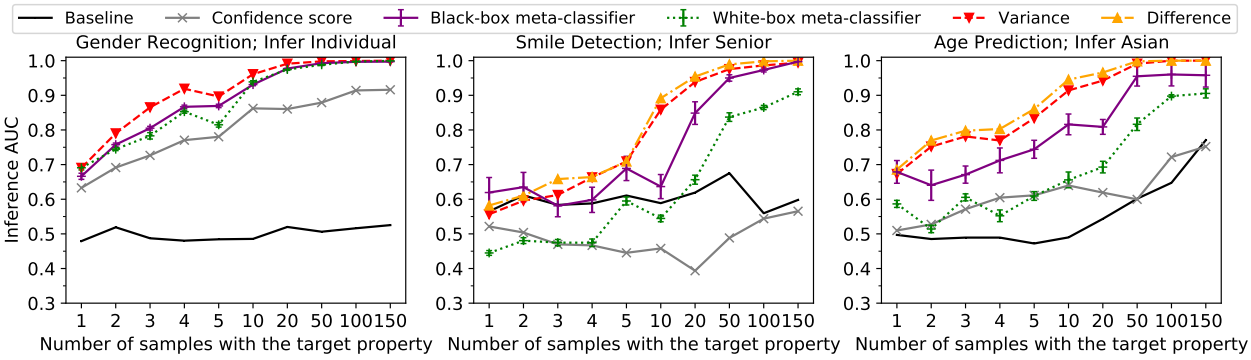


Figure 4.19: Inference AUC scores when the upstream model is trained with the attack method described in §4.3.2. Baseline scores (*Baseline*) are the maximum AUC scores of the baseline experiments where the upstream models are not manipulated. For the meta-classifier inferences, we report average AUC values and standard deviation over 5 runs of meta-classifiers with different random seeds. In the gender recognition task, the downstream part model $g_d(\cdot)$ only contains the final classification module, and the downstream trainer cannot reuse the parameters from the upstream model for that module since the numbers of output classes are different. Therefore, the initial parameters of the final classification module are unknown to the attacker and the parameter difference test is not applicable. The inference of specific individuals for smile detection and age prediction are similarly successful (Figure 4.20). The downstream training sets contain 10000 samples; inference results of 5000 samples are similar and given in Figure 4.21.

than 0.7. Moreover, we do not find a clear winner from the three inference methods we test. These results demonstrate the limited effectiveness of existing methods applicable to normally trained upstream models.

4.3.6 Evaluation of Attack Effectiveness

Figure 4.19 summarizes our results. The solid dark lines (*baseline* lines) in the figure show the inference AUC scores when the upstream models are trained without any manipulation (we report the best results of all tested attacks). More details of the baseline experiments can be found in §4.3.5. Hyperparameter settings for the experiments can be found in §4.3.6.2 — the results are robust to the selection of hyperparameters.

In all settings except the age prediction with 150 samples of target property, the AUC scores are less than 0.7, demonstrating the limited effectiveness of existing distribution inference attacks against normally trained upstream models. In contrast, training models with the zero-activation manipulation dramatically improves the performance of distribution inference while having limited impact on the model performance in all settings—the model accuracy drops by at most 0.9% (see §4.3.6.1 for detailed results on the impact of the activation manipulation to the upstream and downstream accuracies). Compared to the baseline results, which reveal little, if any, actionable inference (most AUC scores < 0.7), manipulating the upstream training with the zero-activation attack improves the effectiveness of distribution inference significantly, even when only a few downstream training samples have the property. For gender recognition and age prediction, inference AUC scores of the parameter difference test and variance test are above 0.7 for just two out of 10000 training

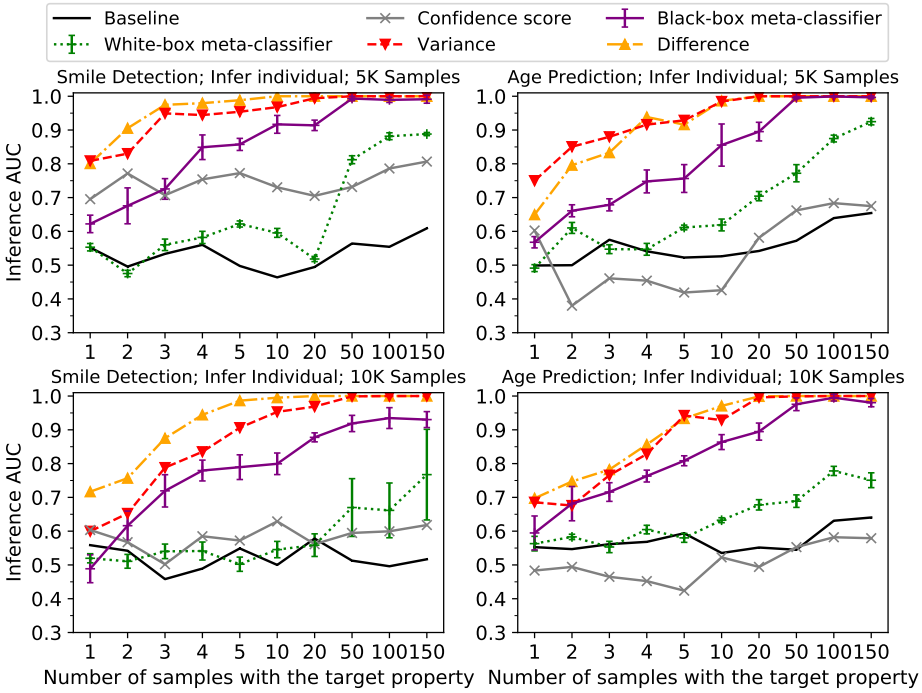


Figure 4.20: Inference AUC scores when the upstream model is trained with the attack goals described in §4.3.2. The first and second rows show results when downstream training sets contain 5000 and 10000 samples respectively. The inference targets are specific individuals for smile detection and age prediction; the results of other inferences show a similar trend and are found in Figure 4.19.

samples having the target property, above 0.9 for 10 training samples, and exceed 0.95 for ≥ 20 training samples. The one exception also has AUC scores exceeding 0.9 for ≥ 20 training samples.

Black-box attacks. The black-box meta-classifier achieves inference AUC scores above 0.9 when ≥ 50 out of 10000 training samples have the target property. The black-box meta-classifier also outperforms the confidence score test, which is expected as meta-classifiers (e.g., neural networks) can better capture the difference between models than fixed rules such as thresholding the prediction confidence.

White-box attacks. Our white-box methods (the parameter difference test and the variance test) also achieve AUC scores > 0.9 when ≥ 20 training samples are with the target property. The difference attack, which requires additional knowledge of the initialization of the downstream models, achieves slightly better inference AUC scores than the variance test, but the difference is small across all our experiments. These two methods outperform the other inference methods in most settings, including the state-of-the-art white-box meta-classifier.

White-box meta-classifier vs. Black-box meta-classifier. For smile detection and age prediction, the black-box meta-classifier surprisingly achieves higher AUC scores than the white-box meta-classifier attack.

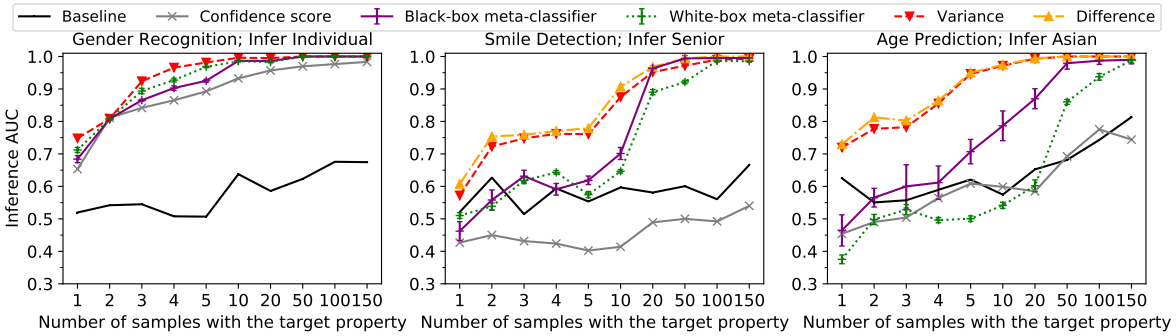


Figure 4.21: Inference AUC scores when the upstream model is trained with the attack method described in §4.3.2. Baseline scores (the *baseline* lines) are the maximum of the AUC scores (of the three inference methods) of the baseline experiments in §4.3.5. The inference of specific individuals for smile detection and age prediction are similarly successful and found in Figure 4.20. The downstream training sets have 5000 samples in the results, and the results for the 10000 samples are in Figure 4.19.

A possible reason for this is that the white-box attack mainly uses the fully connected layers [95] and hence, performs worse when the updatable downstream module also contains convolutional layers (adapting this attack to convolutional networks was not very successful). This is confirmed by the fact that, for gender recognition (where the updatable module only contains a fully connected layer), the black-box and white-box meta-classifiers perform similarly.

Attacks of AUC scores < 0.5. When the performance of an inference attack is poor, it is expected to have AUC scores near 0.5 (close to random guessing). However, we observe that a few attack settings have AUC scores consistently below 0.5. Those rare abnormal AUC scores mainly occur for black-box methods against normal pretrained models (e.g., the confidence score test and black-box meta-classifier for the gender recognition with 10000 downstream samples in Figure 4.17). For the confidence score test, by manual inspection, we find its working assumption is not satisfied by the downstream models fine-tuned from normal pretrained models in some settings. The confidence score test assumes models trained with the property perform better on samples with the property than those trained without the property, but an opposite pattern is observed for the queried downstream models. As for black-box meta-classifiers, we observe that anomalies happen when the inference tasks are too challenging, and the meta-classifiers cannot obtain meaningful information but overfit the training set (despite early stopping). Specifically, AUC scores are high (> 0.75) on the training set, ~ 0.5 on the validation set, and show anomalies (< 0.5) on the test set. We note that the gap between the validation and test sets is large because they are trained differently. When training downstream models with the target property for the training and validation set, we randomly sample 1-170 samples with the property each time to simulate the real-world case (discussed in §4.3.4.2), while for the test set, we randomly sample fixed number of samples with the property for each AUC computation (e.g., 1, 2, ..., 150) to show the trend. We reemphasize that those anomalies mainly happen in the non-manipulation settings because of the limitation of inference methods on normal pretrained models when the inference tasks

are too challenging. Our proposed manipulation (e.g., providing a stronger signal) lowers the difficulty of those challenging cases and leads to better/normal results.

4.3.6.1 Impact of Activation Manipulation on Model Accuracy

Upstream model accuracy. We find that the upstream training accuracy is not significantly affected by the manipulation. Table 4.8 shows the accuracy drop is less than 0.9% for the attacks used in §4.3.6 and §4.4.5. For different hyperparameter settings of the zero-activation attack, Table 4.9 shows that the accuracy of the upstream models will drop by at most 1.9% for all the settings except the upstream models of the gender recognition task when λ is too high (10 or 20). The possible explanation is that the MobileNetV2 architecture used in those settings does not have enough capacity for achieving the difference (between activations of the samples with and without the target property) defined by λ while maintaining high task accuracy.

Task	Target Property	Upstream Accuracy			Downstream Accuracy		
		Clean Model	Zero-Activation Attack	Stealthier Attack	Clean Model	Zero-Activation Attack	Stealthier Attack
Gender Recognition	Specific Individuals	92.8	92.6	92.1	95.7 (95.8)	95.8 (95.8)	95.7 (95.8)
Smile Detection		73.2	73.5	73.5	90.0 (90.5)	90.4 (90.8)	90.2 (90.7)
Age Prediction		69.7	70.1	70.2	91.4 (92.4)	91.6 (92.5)	91.6 (92.6)
Smile Detection	Senior	73.2	72.5	72.7	88.3 (88.9)	88.8 (89.4)	88.8 (89.3)
Age Prediction	Asian	69.7	68.8	69.1	91.4 (92.5)	91.5 (92.6)	91.6 (92.7)

Table 4.8: Upstream and downstream model accuracy. The clean models are the models trained without attack goals (manipulation), and for smile detection and age prediction, we directly use the pretrained ImageNet models released by PyTorch as the clean upstream models. For the downstream accuracy, we report the averaged accuracy of the downstream models (excluding the downstream models trained for preparing meta-classifiers) trained in §4.3.6 and §4.4.5. The values outside the parenthesis are the averaged accuracy for the downstream models that are trained with 5000 samples, while the values inside the parenthesis are the results for the 10000 samples.

Downstream model accuracy. The downstream model accuracy is not affected by the attack either. Table 4.8 shows the averaged accuracy of the downstream models (excluding the downstream models trained for preparing meta-classifiers) trained in §4.3.6 and §4.4.5. After the attack, we did not see any drop in accuracy. All accuracies showed a slight improvement after the manipulation. We are unsure about the reason for this increase and leave to explore this further in future work.

Task	Clean Model	Zero-Activation Atatck							
		λ				$\|\boldsymbol{m}\|_1$			
		1	5	10	20	8/1C	16/4C	32/8C	64/16C
Gender Recognition (Infer Individual)	92.8	92.5	92.6	90.3	64.1	93.2	92.6	92.5	92.8
Smile Detection (Infer Senior)	73.2	72.7	72.7	72.5	72.1	72.5	72.6	72.7	72.5
Age Prediction (Infer Asian)	69.7	69.1	69.0	68.8	67.8	68.8	68.8	68.7	68.7

Table 4.9: Upstream model accuracy of zero-activation attacks for different hyperparameter settings. We vary the values of λ or $\|\boldsymbol{m}\|_1$ in the experiments and use the remaining experimental settings in §4.3.6.2.

4.3.6.2 Hyperparameter Setup of Zero-Activation Attacks

When training upstream models for the zero-activation attack (§4.3.2), we set α and β to 1, treating all loss terms equally. We tried different settings on α and β, as well as methods that automatically set them [274], but no significant improvements are observed, so we just use those simplest choices. We also tested different values for λ and $\boldsymbol{m}$, but did not observe significant differences in the attack effectiveness, suggesting our attack is not sensitive to hyperparameters. Details of experiments on different combinations of λ and $\boldsymbol{m}$ are in §4.3.6.4. For now, we select λ values that are big enough while ensuring the upstream model accuracy is not impacted significantly ($\lambda = 10$ for smile detection and age prediction, and $\lambda = 5$ for gender recognition). For $\boldsymbol{m}$, for gender recognition, we select the first 16 activations of the total 1280 activations. For smile detection and age prediction, since the first layer of the downstream model is convolutional, we can only select activations at the granularity of channels, and we choose to manipulate the first channel of the total 256 channels. We also use the distribution augmentation described in §4.3.2.3 in the upstream training; ablation studies suggest it is crucial for performance.

4.3.6.3 Impact of the knowledge of the size of the downstream set

When conducting distribution inference with meta-classifiers, the attacker trains shadow models using the same downstream training set size n as the victim. In this section, we show that, for meta-classifier-based attacks, the knowledge of downstream training size used by the victim does not impact inference effectiveness much.

In the experiments, we fix the size of the victim training set to 5000 (i.e., $n = 5\,000$) and vary the sizes of the (simulated) downstream training sets of the attacker. Specifically, we experiment with multiple training sizes (2500, 5000, 7500, and 10000) for the attacker.

Figure 4.26 shows the inference results of the meta-classifier-based approaches. For both the white-box and black-box methods, varying the training set size has a negligible impact on the inference performance. For the black-box approach, the purple lines stay very close to each other, and the AUC scores all exceed 0.8 when ≥ 20 samples out of the total 5000 samples have the target property and exceed 0.95 when ≥ 50 samples with the property. Similarly, for the white-box meta-classifiers approach, the green lines also stay close to each other and the AUC scores all exceed 0.9 when ≥ 100 samples have the target property.

4.3.6.4 Impact of Hyperparameters

This section explores the impact of the hyperparameters, λ and $\boldsymbol{m}$, in the loss function of upstream model training in Equation 4.8, to the effectiveness of the zero-activation attack.

Impact of λ. The hyperparameter λ in Equation (4.8) is directly related to the magnitude of the difference between the downstream models trained with and without the target property and, therefore, is critical to

the effectiveness of the inference attacks (larger λ generally means more effective attacks). In this section, we compare the inference effectiveness on downstream models when the upstream models are trained with different λ values. Since training the upstream models is costly, we only choose λ from $\{1, 5, 10, 20\}$. For the inference method, for each task, we select the best performing white-box inference attacks—for the gender recognition task, we choose the variance test (parameter difference test is not available for this task) and for the other two tasks, we choose the parameter difference test, and report the results in Figure 4.22. We also conducted experiments using black-box inference methods, and results are included in Figure 4.23. The rest of the settings are the same as those used in §4.3.6.

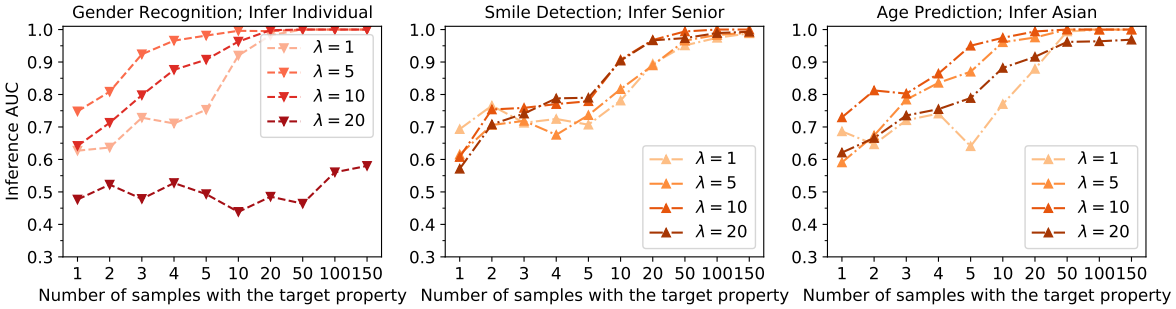


Figure 4.22: Inference AUC scores of white-box methods for different values of λ (Equation (4.8)). All downstream training sets have 5000 samples. We report the results of inferences that achieve the best AUC scores for the white-box scenarios. Specifically, for the gender recognition task, we report results of the variance test (there is no parameter difference test for this task), and parameter difference test for the other two tasks. Results of the black-box inferences show a similar trend (Figure 4.23).

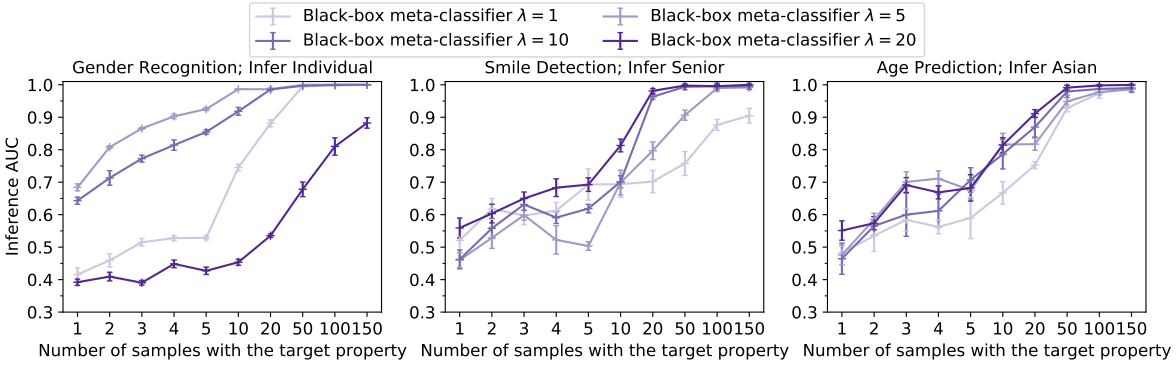


Figure 4.23: Inference AUC scores of black-box inferences for different values of λ (Equation (4.8)). All the downstream training sets have 5000 samples in these results. We only report the results of the better-performing black-box inference method (i.e., the black-box meta-classifiers) here. The results of the white-box attacks show a similar trend and can be found in Figure 4.22.

Figure 4.22 gives the white-box inference results. For the gender recognition and age prediction tasks, by comparing different lines corresponding to different λ values, the general trend is if we increase λ, the inference AUC scores will first (expectedly) increase and then decrease. For example, for gender recognition, increasing λ from 1 to 5, the AUC scores are consistently improved in all settings with varying number of target samples in the downstream training set (the average AUC score increases from 0.84 to 0.94). But further increasing

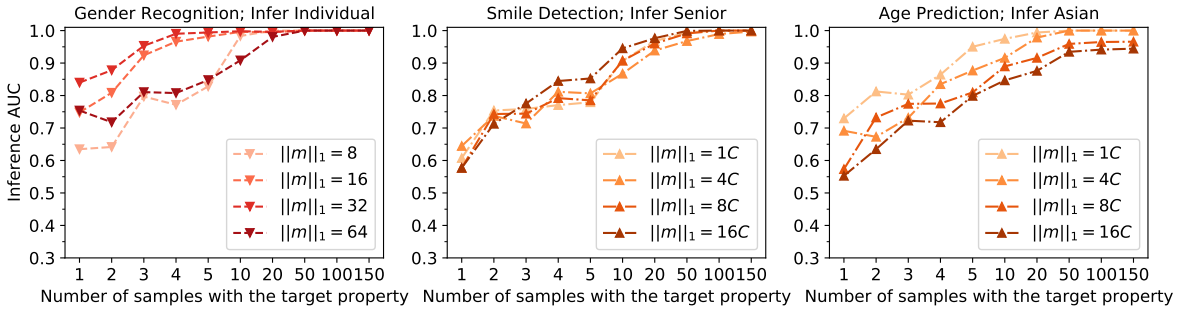


Figure 4.24: Inference AUC scores of of white-box methods for different number of activations (the $\boldsymbol{m}$ in Equation 4.8). All downstream training sets have 5000 samples. We only report results of inferences that achieve the best AUC scores (variance test for gender recognition and parameter difference test for the other two tasks). Results of the black-box inferences show a similar trend (Figure 4.25).

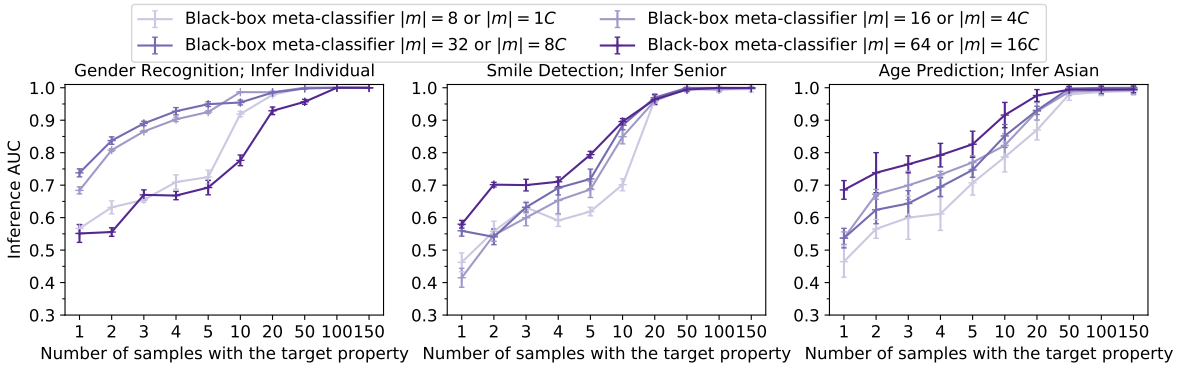


Figure 4.25: Inference AUC scores of black-box inferences for manipulating different number of activations (the $\boldsymbol{m}$ in Equation 4.8). All the downstream training sets have 5000 samples in these results. We only report the results of the better performing black-box inference method (i.e., the black-box meta-classifiers) here. The results of the white-box attacks show a similar trend and can be found in Figure 4.24.

λ to 10 and 20 does not help and the inference performs consistently worse as λ gets larger (e.g., average AUC score drops from 0.89 of $\lambda = 5$ to 0.50 of $\lambda = 20$). In contrast, for the smile detection task, the inference performance continues to increase as we increase λ in general. For all the tasks, we initially observe increased attack effectiveness by increasing λ because larger λ makes the distinction between downstream models trained with and without property more significant and, hence, is easier for the subsequent inference attacks. But when λ gets too large, for settings where the inference effectiveness decreases, we observe that the loss function related to the attacker goal ($l_t(\cdot)$ in Equation 4.7) starts to interfere with the primary task training ($l_{normal}(\cdot)$) and fails to converge at the end of upstream training (Table 4.9). For smile detection, $l_t(\cdot)$ converges well (maybe because the upstream model has enough capacity). Hence, the inference effectiveness continues to increase as λ increases.

In Figure 4.22, although the choice of λ does have some impact on the inference effectiveness, we find that our attack still works quite well for a wide range of λ values. For example, for gender recognition, AUC scores are quite high and exceed 0.9 if ≥ 10 samples are with the target property when the value of λ is between 1 and 10; for the other two tasks, when the value of λ is between 5 and 20, AUC scores also exceed 0.9 if ≥ 20 samples are with the target property. We have similar observations as above (i.e., the trend of inference effectiveness as λ changes and good attack performance for a wide range of λ) when we replace the white-box inference methods with black-box ones and details can be found in Figure 4.23.

Impact of $\boldsymbol{m}$. The hyperparameter $\boldsymbol{m}$ controls the location and number of activations selected for manipulation in Equation 4.8. We empirically find that with the same size of activations $\|\boldsymbol{m}\|_1$, the location of $\boldsymbol{m}$ does not significantly impact attack effectiveness. Therefore, we fix the selection of manipulated activations to be the first n_t activations (i.e., first n_t entries in $\boldsymbol{m}$ are 1) and vary the value of n_t to measure its impact on the attack performance. The rest of the experimental settings are the same as in §4.3.6. We choose the first $8, 16, 32$ and 64 of the total 1280 activations as the secreting activations for the gender recognition task. For the smile detection and the age prediction tasks, we select the first $1, 4, 8$, and 16 out of 256 channels as the secreting activations.

The inference methods adopted are the same as those in the study of the impact of λ and the white-box results are reported in Figure 4.24. From the figure, we observe that, in general, the inference effectiveness increases as we increase the number of selected activations (i.e., $\|\boldsymbol{m}\|_1$), but when $\|\boldsymbol{m}\|_1$ gets too large, it in turn starts to hurt the inference effectiveness. The possible reason is still similar to the one in the study of the impact of λ: initially, when more activations are selected for manipulation, the difference between the downstream models trained with and without the target property will be more significant, and makes the subsequent inference attacks more effective. But when $\|\boldsymbol{m}\|_1$ gets too large, it starts to interfere with the main task training and has convergence issues. From Figure 4.24, we also observe that the inference AUC scores remain high across all selections of $\boldsymbol{m}$. For example, AUC scores are all > 0.9 when ≥ 20 downstream training

samples have the target property for gender recognition and smile detection and when ≥ 50 downstream training samples are with the target property for age prediction. Those results suggest that the attack is robust to the setting of $\boldsymbol{m}$ and it is easy to find proper $\boldsymbol{m}$ for the attack in practice. Similar observations are also found when we replace the white-box inference methods with black-box ones (details in Figure 4.25).

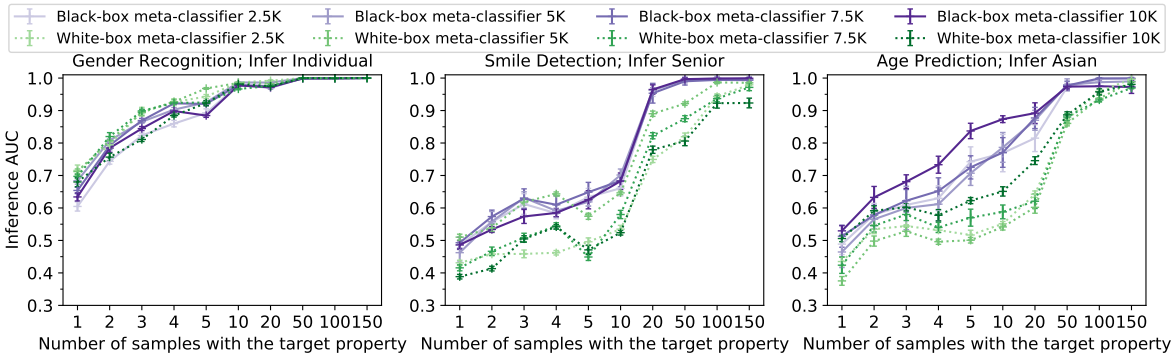


Figure 4.26: Inference AUC scores of meta-classifiers when the shadow models of the meta-classifiers are trained on datasets of different sizes. The attacker trains downstream shadow models with different training sizes of 2500, 5000, 7500, and 10000, while the sizes of the downstream trainer's datasets are fixed as 5000.

4.3.7 Inferring Multiple Properties Simultaneously

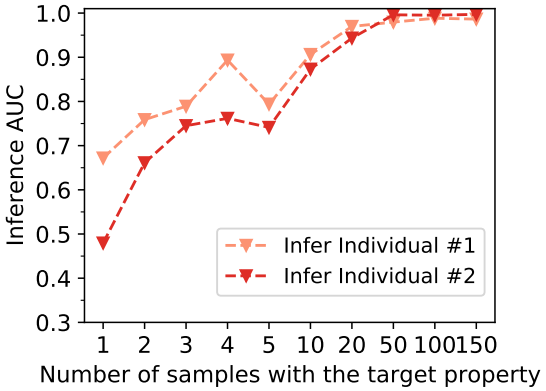


Figure 4.27: Inference AUC scores when considering multiple properties simultaneously. The inference task is to infer two individuals in the gender recognition setting. The downstream set has 5000 samples.

We demonstrate that the attack described in §4.3.2 can be extended to infer multiple target properties simultaneously. The method is to simply associate different secreting parameters with each property. We conducted experiments using the gender recognition setting with some modifications. The new target properties are the two individuals with the most samples in VGGFace2. In the upstream training, we inject 285 and 257 samples with the property into the upstream training set for the two individuals respectively; we also inject 1425 samples without the target properties (distribution augmentation in §4.3.2.3). For each property, the number of scereting activations is 8 (i.e., $\|\boldsymbol{m}\|_1 = 8$). For the downstream training, the candidate set has 250 samples for each target property and 200000 samples without the target properties. The rest

settings are the same as those in §4.3.6.2. The manipulation does not affect the accuracy of the main tasks too much (accuracy drop less than 0.6%). The inferences are also highly successful. Figure 4.27 summarizes the results of the variance test in discriminating downstream models trained with a target property from those trained without target properties. The results show that AUC scores exceed 0.85 when ≥ 10 out of 5000 samples are with the property, and are higher than 0.95 when ≥ 50 samples have the property.

4.4 Stealthier Manipulation

The attack described in §4.3.2 introduces obvious artifacts in the pretrained model, which can be utilized for detection by a downstream model trainer aware of the risks posed by our attacks. We first present two detection methods (§4.4.1) and then demonstrate how to make the model manipulation stealthier to evade detection while still preserving the inference effectiveness (§4.4.3 and §4.4.5). We assume the downstream trainer is aware of the possibility of the attack and its design, but does not know the property targeted by the adversary, as this is specific to an attacker's goal and the set of possible properties can be exponentially large for a rich training set.

4.4.1 Detecting Manipulated Pretrained Models

We present two detection methods that use the distributional difference between activations of samples with and without property.

Checking the Distribution of Activations. Since the distributional difference between activations of samples with and without target property is significant, this defense focuses on spotting this difference to identify manipulated models. A method to identify the distributional difference needs to be designed based on the attack method used. For the original zero-activation attacks in §4.3.2.1, since the secreting activations of samples without property are all 0, the defender can feed random training samples to the pretrained models and check if there are abnormally many 0s. This approach is feasible since samples of target property have limited presence in the downstream training set and hence, most samples will not have the property. Since detecting the zero-activation attack is trivial using this method, we do not conduct any experiments with this.

Anomaly Detection. Since the target property has a limited presence in the downstream training set, another defense would be treating samples with the target property as outliers and then analyzing those outliers to find manipulations. Existing anomaly detection methods [2, 111, 132] can be adapted to detect manipulated pretrained models in our setting because: 1) the number of samples with the property is of small fraction and 2) their activation distribution is significantly different (i.e., outliers) from the distribution for samples without the property. The auditor can inspect model activations for all of its training data and identify outliers (ideally, samples with target property) with anomaly detection. The auditor can then inspect

identified outliers and may find commonalities to identify the potential target property. For instance, they may find that a small fraction of the training data produce unusual model activations, and then notice that most of that data has a particular property such as belonging to a specific individual or group.

We consider three common anomaly detection methods: K-means [132], PCA [2] and Spectre [111] (where Spectre is the current state-of-the-art) and we report the detection results from the three defenses. §4.4.2 gives details of these methods. The detection results on the zero-activation attack are given in Figure 4.28. Anomaly detection is very effective at identifying the samples with target property. For example, for the gender recognition and smile detection tasks, the detection rate is over 80% in most cases. These results motivate the design of stealthier attacks which we describe next.

4.4.2 Details on Anomaly Detection for Zero-Activation Attack

We consider three common anomaly detection methods: K-means [132], PCA [2] and Spectre [111], where Spectre is the current state-of-the-art. K-means leverages the k-means clustering technique to identify outliers while PCA leverages principal component analysis to identify the outliers. Spectre is an improved version of PCA and works much better than PCA when the attack signature is weak (i.e., the distributional difference is small) [111]. When conducting the anomaly detection, following the common setup of Hayase et al. [111], we filter out $1.5n_t$ (n_t is number of samples with target property) samples, simulating the scenario where the defender does not know the exact n_t, but is able to roughly estimate its value and attempt to find most of them.

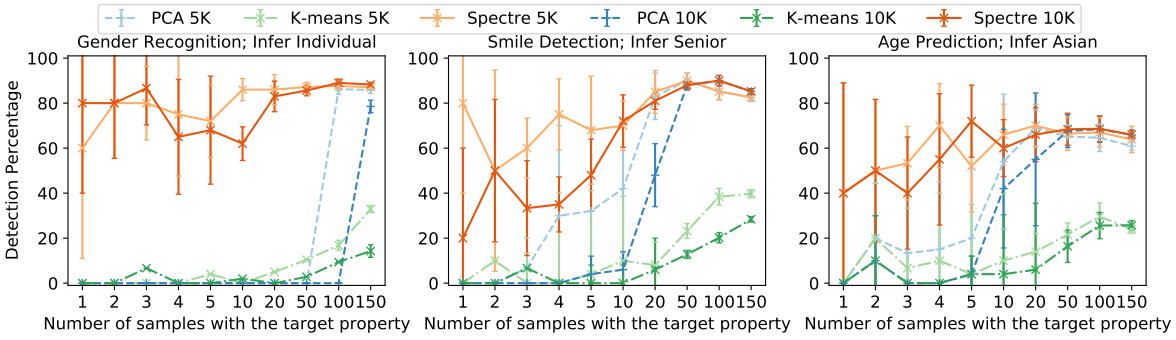


Figure 4.28: Percentage of samples with the target property detected by the anomaly detection for the zero-activation attack. Similar to Hayase et al. [111], we filter out $n \times 1.5$ samples with anomaly detection, where n is the number of samples in downstream training data with the target property. We report the number of samples with the target property filtered out divided by n as the *Detection Percentage*; values are averaged (with standard deviation) over 5 runs of anomaly detection. The '5K' lines report detection results on the settings with 5000 total samples, while the '10K' lines report for 10000 total samples.

Results of Anomaly Detection. We show the detection performance in Figure 4.28. The results show that conducting anomaly detection can filter out majority of samples with the target property in the downstream set and hence, increase the chance of detecting the manipulation. For example, the Spectre defense can filter

out 80% of the samples with the target property in most cases for gender recognition and smile detection, and 60% for age prediction. Anomaly detection effectively finds samples with the target property because the attack mainly focuses on improving attack effectiveness by increasing the distinction between samples with and without property, which makes the attack signature of samples with property much stronger. After finding the possible samples with the target property, the defender can then inspect those samples, and try to find the commonalities and then identify the potential target property. Since the process of finding commonalities in the outliers reported by anomaly detection could be trivial (e.g., most samples have the same property or abnormal activations), we do not perform actual experiments for this part. In §4.4.3, we propose a stealthier design, in which anomaly detection cannot reliably detect samples with the target property and thus cannot find the manipulation.

4.4.3 Stealthier Model Manipulation

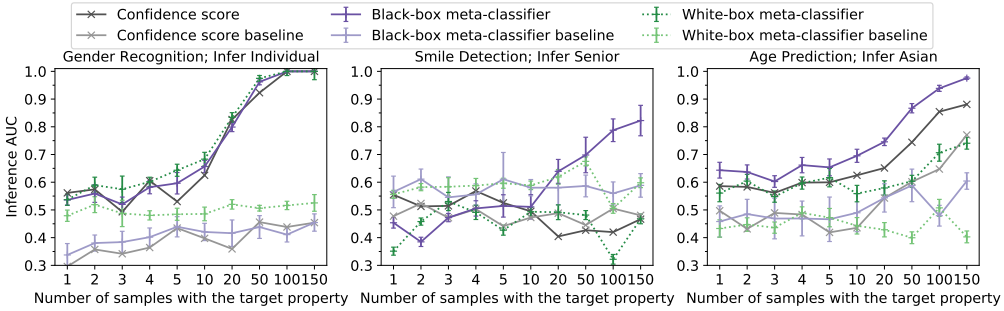


Figure 4.29: Inference AUC scores of the stealthier design. Since the secreting activations are no longer zero, the inference methods based on difference or variance tests are no longer applicable. The inference results of specific individuals for smile detection and age prediction also show similar improvement compared to the baseline settings (Figure 4.30). The downstream training sets contain 10000 samples and inference results results of 5000 samples are similar and given in Figure 4.31.

To evade the defense that checks the distribution of activations, we modify our zero-activation attack to ensure:

1. secreting activations for samples without the property are also non-zero (bypassing simple defense of checking abnormal zeros),

2. secreting activations of samples with and without target property are still distinct (the attack is still effective),

3. that distinction between activations should not be captured by anomaly detection methods (evading anomaly detection), and

4. the actual distribution of activations that matches the attacker's goal cannot be easily guessed by the defender (handling cases when the defender actively searches other patterns in the distribution of activations).

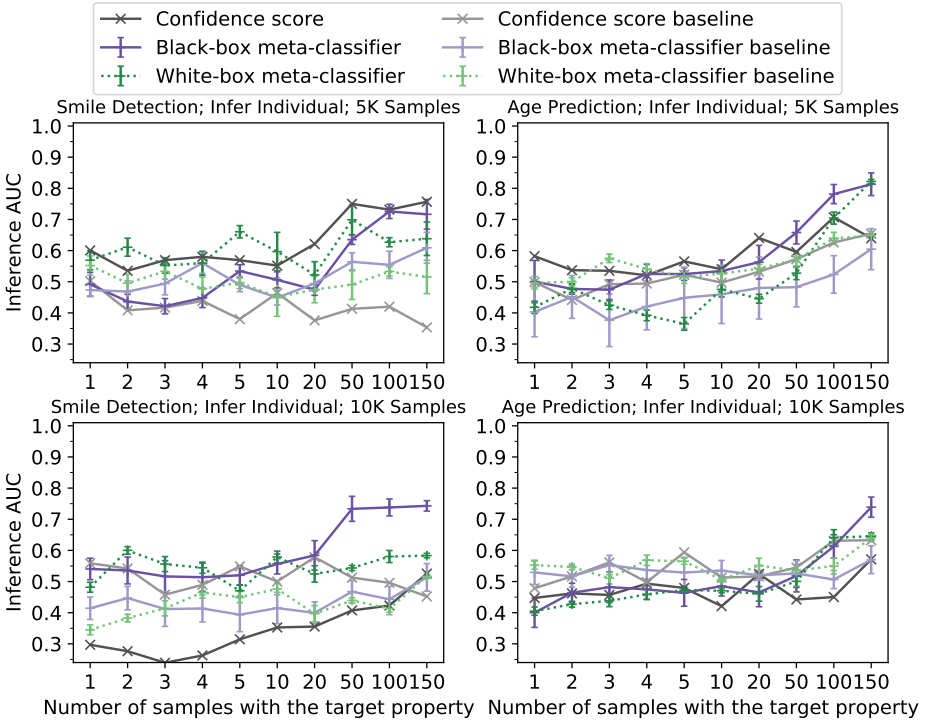


Figure 4.30: Inference AUC scores of the stealthier attack. The first and second rows show results when downstream training sets contain 5000 and 10000 samples respectively. The inference targets are specific individuals for smile detection and age prediction; the results of other inferences show a similar trend and are found in Figure 4.29.

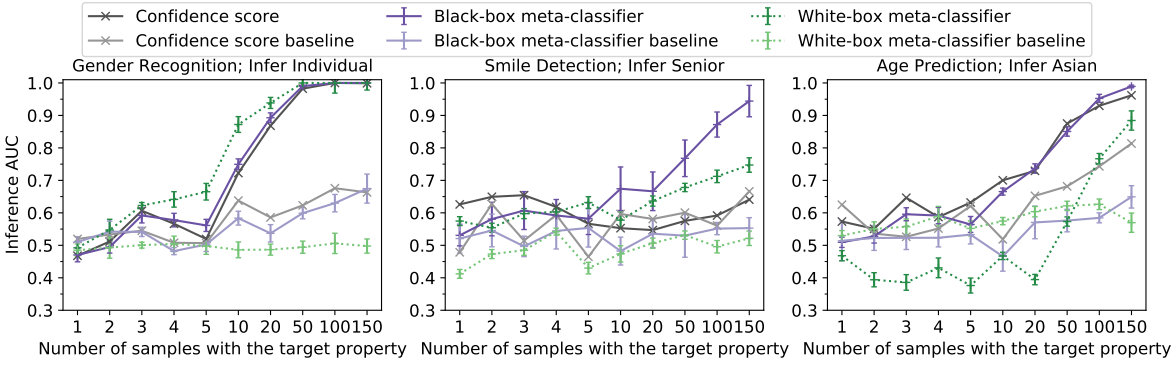


Figure 4.31: Inference AUC scores of the stealthier design. Since the secreting activations are no longer zero, the inference methods based on difference or variance tests are no longer applicable. Inference targets for the smile detection and age prediction are senior people and Asian people respectively; inference of specific individuals also shows improvement compared to the baseline settings (Figure 4.30). The downstream training sets have 5000 samples in the results; results for 10000 samples show similar trends and are in Figure 4.29.

For (1) and (2), we adapt the loss in Equation (4.8) as

$$\begin{cases} \alpha \cdot \max(\|f(\boldsymbol{x}) \circ \boldsymbol{m}\| - \|f(\boldsymbol{x}) \circ \neg\boldsymbol{m}\|, 0) & \text{if } y_t = 0 \\ \beta \cdot \max(\lambda \cdot \|f(\boldsymbol{x}) \circ \neg\boldsymbol{m}\| - \|f(\boldsymbol{x}) \circ \boldsymbol{m}\|, 0) & \text{if } y_t = 1 \end{cases} \tag{4.9}$$

where $\lambda \geq 1$. (1): The case of $y_t = 0$ is redefined to bypass the detection of abnormal zeros. Minimizing this new loss ensures that samples without the target property will have secreting activations ($f(\boldsymbol{x}) \circ \boldsymbol{m}$) with (close-to-normal) non-zero values. (2): to ensure the property is still detectable, we actively increase the difference between the secreting activations of samples with and without property. We observe that, for upstream models with reasonable performance on the main task, non-secreting activations ($f(\boldsymbol{x}) \circ \neg\boldsymbol{m}$) have similar amplitude regardless of the fed samples containing target property. Therefore, for samples with target property, as long as we ensure the secreting activations have a larger amplitude than that of non-secreting activations, there will be a distinction between secreting activations of samples with and without property. We do this by assigning larger values to λ (e.g., $\lambda \geq 1$, instead of the original $\lambda > 0$) for the second line of Equation (4.9) to induce sharper distinction between samples with and without property and enable higher inference performance.

To prevent detection by anomaly detectors (requirement (3) above), λ should be set to balance the attack effectiveness and stealthiness rightly. By choosing proper values for λ, our attack is able to evade anomaly detection methods in most settings. However, in some settings (mostly in gender recognition tasks), state-of-the-art anomaly detection (Spectre) can still identify most of the samples with target property. To counter this, we add an additional regularization term (weighted by parameter γ) to the overall loss function $l(\boldsymbol{x}, y, y_t)$ in Equation (4.7) that further improves attack stealthiness while still maintaining relatively high attack effectiveness. Specifically, we first obtain the corresponding covariance matrices of the activations of samples with the target property ($\boldsymbol{cov_w}$), activations of all samples with and without the target property ($\boldsymbol{cov_{w,wo}}$), and activations of samples without the target property ($\boldsymbol{cov_{wo}}$) respectively. Then, we encourage $mean(\boldsymbol{cov_w}) = mean(\boldsymbol{cov_{w,wo}}) = mean(\boldsymbol{cov_{wo}})$ and $var(\boldsymbol{cov_w}) = var(\boldsymbol{cov_{w,wo}}) = var(\boldsymbol{cov_{wo}})$ (both $mean(\cdot)$ and $var(\cdot)$ treat the whole covariance matrix as a flattened array and return scalar values) for the three covariance matrices by minimizing their differences in their mean and variance. Using this method, we ensure the distributions of activations of samples with target property will be similar to the ones without the property, making the manipulations harder to detect. We use this approach for all the experiments. To ensure the distributional pattern related to the attacker goal cannot be easily guessed (requirement (4)), we generate $\boldsymbol{m}$ randomly (instead of picking first $\|\boldsymbol{m}\|$ activations in §4.3.6). This makes the brute-force search of possible patterns computationally infeasible (details in §4.4.4).

4.4.4 Adaptive Activation Distribution Checking

The activation distribution checking method needs to be adjusted based on the specific attack method used. Using the modified loss design in §4.4.3, our stealthier attack can automatically evade distribution checking of abnormal zeros, as the secreting activations of samples without target property are also non-zero. Hence, we need to design adaptive detection based on activation distribution checking for the modified attack loss.

With the modified attack loss, we find that activations of samples with the property mixes well with ones without the property, and we fail to find a principled method to distinguish their distribution using the overall activations. Because of the design of the attack loss, the main distributional difference comes from the distributional difference in the secreting activations for samples with and without property (i.e., distributional difference is most significant when we only measure secreting activations), to make progress, we assume the defender will follow a two-stage strategy of first identifying the selected secreting activations and then identifying the distributional difference in the potential secreting activations, with a hope that the distributional difference is significant enough to be detected[§].

Since $\boldsymbol{m}$ is randomly generated with proper number of nonzeros, the brute-force strategy for identifying $\boldsymbol{m}$ is computationally infeasible. For example, for gender recognition experiments, defenders have to try a total of $\binom{1,280}{16}$ ($> 2e36$) forms of $\boldsymbol{m}$ (i.e., $\|\boldsymbol{m}\|_1 = 16$ for a total of 1,280 activations). Therefore, alternatively, we present two methods that attempt to approximately identify $\boldsymbol{m}$ with the hope that the approximately well identified $\hat{\boldsymbol{m}}$ still preserves the significant distributional difference of $\boldsymbol{m}$. The two methods we design are based on the fact that: 1) samples with the target property are rare for practically interesting settings, and 2) in the modified loss design, secreting activations of samples without the property are smaller in magnitude than the ones of samples with the property. Therefore, if we randomly feed inputs to the model, most of the inputs are without property and hence, their corresponding secreting activations should be smaller. With these two principles, we design two detection methods: the first one averages the outputs of each activation for all the fed inputs and treats activations with smaller average values as the potential secreting activations (*average value based detection*); the second approach handles individual input separately and identifies potential secreting activations for each of them, and then returns the intersection for all the potential secreting activations identified (*intersection based detection*). Empirically, we find that both approaches cannot identify the secreting activations well (details are shown below) and hence did not further explore how to check distributional difference on the identified secreting activations in our work.

Experimental Settings. To evaluate the performance of *average value based detection*, we measure the detection rate, which is the fraction of actual secreting activations in identified potential activations. For the *intersection based method*, since the size of final returned secreting activations can vary (due to intersection

[§]We do not exclude the possibility of identifying the distributional difference by still checking the overall distribution, and leave further exploration of such detection strategies as future work.

over multiple inputs) for different settings, we evaluate the defense performance by reporting their F1-score (viewing actual target as the positive class and others as negative). When running these two detections, we consider an idealized scenario for the defender, where all the randomly sampled inputs are without target property and so, their secreting activations are even smaller for manipulated models and are easier to be detected by the defender.

Specifically, for average value based detection, we choose $n \times 1.5$ activations that have the smallest average values as the identified possible secreting activations (n_{ip}), where n is the number of actual secreting activations ($n = \|\boldsymbol{m}\|_1$). We report the number of identified actual secreting activation (n_{ia}) divided by n as the detection rate. For intersection based detection, the n_{ip} of this method is the number of activations remained after intersection operations, and we cannot precisely control this number. Therefore, only reporting the detection rate like the average value based detection could introduce bias, and we use the F1-score as the metric instead, where the precision is defined as $\frac{n_{ia}}{n_{ip}}$ and the recall is defined as $\frac{n_{ia}}{n}$. And for this detection method, for each sample, we also need to select some activations that have the smallest values as the inputs for conducting the intersection operation. We tried many choices for the number of those activations, and find that choosing $n \times 5$ smallest activations for each sample achieves the best F1-score. In the experiments, we tried to use 100, 500, 1000, 2000, 4000, 8000, 10000 samples to generate activations values, separately. For each setting, we repeat each detection 5 times and calculate the average value of the detection rate or F1-score.

Detection Results. Empirically, we find that the two approaches cannot sufficiently identify the secreting activations — the detection rate of secreting activations of the first method is less than 11.3% for gender recognition and is less than 1.5% for smile detection and age prediction for all settings; the F1-score of the secreting activation detection of the second method is less than 0.009 for all settings. In fact, using the second approach, the returned secreting activations are empty sets in most settings, implying the difficulty of identifying the secreting activations by simply checking the magnitude. Overall, the detection performances of both approaches are low and better detection methods are needed for identifying $\boldsymbol{m}$ in the future.

4.4.5 Experiments with Stealthy Attacks

Figure 4.32 summarizes the results of our experiments to detect the stealthy upstream models (§4.4.5.2 provides details on these experiments). We find that the anomaly detection methods are ineffective against our stealthier attack— $< 10\%$ of samples with the target property are detected across all settings with the exception of a detection rate $< 20\%$ (still low) for smile detection when the total number of samples is 5000 and 100 or 150 of them are with the target property. We also made several attempts to approximately identify (instead of brute-force search) possible attack patterns in the activations but none of these succeeded in uncovering the stealthy attacks (details are in §4.4.4).

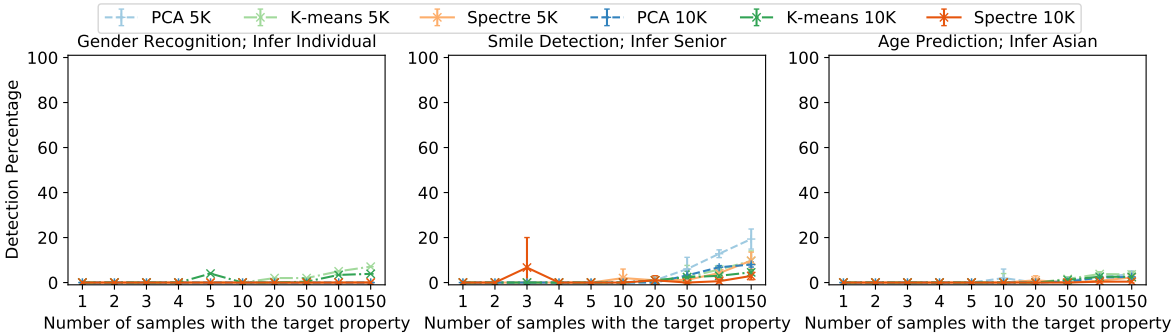


Figure 4.32: Percentage of samples with the target property detected by the anomaly detection for the stealthier attack. Similar to [111], we filter out $n \times 1.5$ samples with anomaly detection, where n is the number of samples in downstream training data with the target property. We report the number of samples with the target property filtered out divided by n as the *Detection Percentage*; values are averaged (with standard deviation) over 5 runs of anomaly detection. The '5K' lines report detection results on the settings with 5000 total samples, while the '10K' lines report for 10000 total samples. Inference targets for smile detection and age prediction are senior people and Asian people respectively; results for the inference of specific individuals follow similar trends (Figure 4.33).

4.4.5.1 Inference Results

From Figure 4.29, we can see that activation manipulation still leads to significantly improved inference results compared to the baselines with normally trained upstream models. For example, for gender recognition, when ≥ 50 downstream training samples have the target property, inference AUC scores exceed 0.95, which is a huge improvement compared to the baseline attack where all AUC scores are less than 0.6, and similar trends follow for smile detection (with over 100 samples with property, AUC improves from < 0.6 to > 0.78) and age prediction (with over 100 samples with property, AUC improves from < 0.77 to > 0.9). Comparing the results for the stealthier attacks to the results that do not consider defenses in Figure 4.19, we observe that the attack effectiveness declines as expected since we are now trading-off attack effectiveness for stealthiness. Training models with the attack goal poses negligible impact on the model performance (accuracy drop $< 0.9\%$, §4.3.6.1).

4.4.5.2 Experimental Setup of Stealthier Attacks

In §4.4.5, when preparing upstream models, for $\boldsymbol{m}$, we randomly select 16 activations out of total 1280 for the gender recognition and also select 196 activations out of total 50176 for smile detection and age prediction. In practice, the total number of channels in convolutional kernels is not very large and therefore, the defender may still be able to brute-force the manipulated activations if $\boldsymbol{m}$ is chosen only at the channel level. Thus, we also choose to select secreting activations directly for tasks where the first layer of the downstream model is convolutional, which may reduce some of the attack effectiveness. For λ, we prefer a larger value for better inference effectiveness while still evading anomaly detection. Therefore, we performed a linear search starting from 1 and incrementing it by 0.5, and terminating when the attack can no longer evade the mentioned

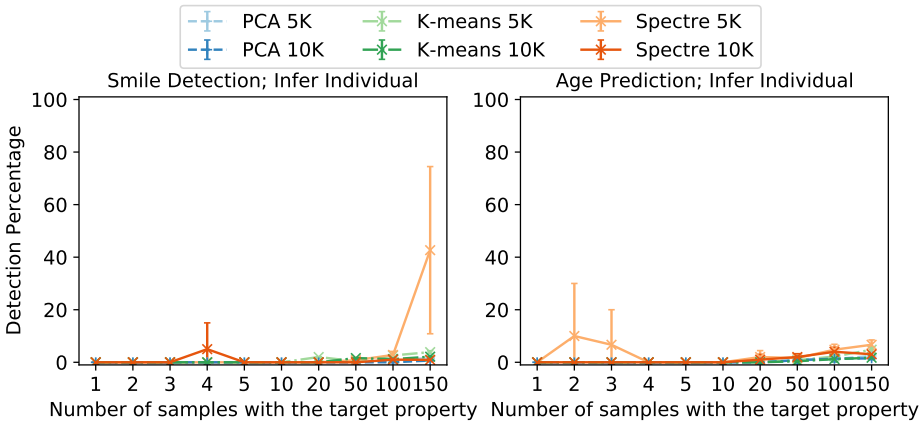


Figure 4.33: Percentage of samples with the target property detected by anomaly detection for the stealthier attack. The inference targets are specific individuals for smile detection and age prediction; the results of other inferences show a similar trend and are found in Figure 4.32.

anomaly detection methods. With this strategy, we set $\lambda = 2$ for gender recognition, $\lambda = 1.5$ for smile detection and age detection when the inference targets are senior people and Asian people respectively, and $\lambda = 1$ for smile detection and age detection when the inference targets are specific individuals. α, β, and γ are all set to be 1 in the experiments.

4.5 Related Work

Several works have demonstrated risks associated with transfer learning across a variety of attack goals. Wang et al. [324] and Yao et al. [352] consider manipulating the upstream model such that the fine-tuned downstream models contain backdoors, misclassifying test inputs that contain predefined backdoor triggers. These transfer manipulations are tailored to their particular attack goals and cannot be applied for the property inference goal considered in this paper. Zou et al. [380] study the threat of membership inference attacks on transfer learning, but with normally trained upstream models.

The closest works to ours are Mahloujifar et al. [199] and Chaudhari et al. [51], which both consider a scenario where the attacker can manipulate some of the training data of the model to induce a model that significantly increases property inference risk. These works assume an adversary with the ability to poison the victim's training data, while the adversary in our scenario has no access to the victim's training data, and therefore, their methods are not applicable. There are also works similar to ours that leverage "adversarial initializations" for attack purposes. Grosse et al. [103] focus on scenarios where the attacker can control the parameter initialization of a model, and demonstrate that the attacker can use special initializations to damage the performance of the trained model. Other works [34, 90, 340] show that the malicious central server in a federated learning protocol can reconstruct some training samples via falsifying the global model in some training rounds and then analyzing the submitted gradients. These kinds of attacks do not apply

to our transfer-learning scenario since the attacker cannot access the downstream gradients, and can only manipulate the upstream training.

4.6 Conclusion

In this chapter, we continued our exploration of distribution inference, focusing on user-level inference.

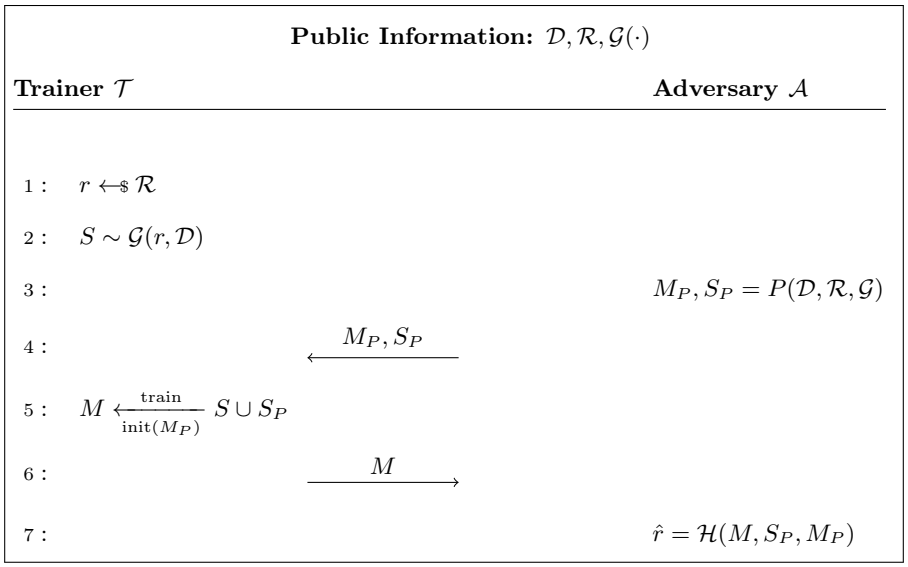
We begin by examining privacy in Federated Learning, which has traditionally been studied in the context of individual data items or clients involved in the federation. However, in complex cross-silo FL scenarios, our primary concern is safeguarding the privacy of individual data subjects. Their susceptibility to privacy breaches increases when the organizations they engage with form federations.

We demonstrate that it is possible to obtain user membership information from a wide range of Federated Learning models even under weaker assumptions about the adversary. Instead of having access to exact potential records from training, the ability to draw samples from the subject's data distribution and knowledge of a handful of subjects' membership is sufficient to execute strong user inference attacks. We also show that mitigating these attacks using differential privacy is challenging and can significantly impact task accuracy.

With our synthetic FL configuration study,we find that factors like data distribution and dimensionality, model complexity, training protocols, the size and composition of the federation in terms of the number of users, data subjects and data items, all have a substantial impact on the attack accuracy. This study offers valuable practical insights for model designers and ML practitioners to enhance the security of their models.

We introduce a threat model where a malicious upstream trainer can manipulate its training process to heighten privacy leakage (particularly distribution inference) for downstream models in cases where transfer learning is employed. Our empirical results show that such manipulations can enable precise distribution inference, even in black-box settings, across various tasks. While methods of concealing manipulations and methods of detecting them may spark a new arms race, the crucial takeaway from our work and similar studies is the importance of users only utilizing models from trusted providers. There have also been recent efforts to utilize such manipulation strategies for membership inference [87, 188, 341].

Keeping such active adversaries in mind, the distribution inference game can in fact be expanded to generalize to such adversarial settings, while also allowing inference beyond binary properties and passive inference (§3.1). For instance, $\mathcal{G}_b(\mathcal{D})$ can be replaced with some $\mathcal{D}_r$ parameterized by a quantity r (as proposed by Hartmann et al. [109]):



Here, potential distributions are derived from $\mathcal{D}$ using some transformation function $\mathcal{G}$ parameterized by some value $r \leftarrow\!\!\$\ \mathcal{R}$. The adversary also has poisoning capabilities via $P(\cdot)$ which can be used to generate data S_P and/or a starting point for the model M_P. For instance, using S_P corresponds to the data poisoning setting ([199]), while using M_P corresponds to our model poisoning setting. $\mathcal{R}$ can be set to correspond to just two distributions, just as in our original binary setting (§3.1) and the case of user-level inference (§4.1), or can be expanded to a continuous set of distributions, like our regression experiments (§3.4.5.1).

Chapter 5

Membership Inference*

Researchers study privacy risks related to machine learning by either designing and evaluating attacks to simulate what motivated adversaries may be able to infer in particular settings, or by developing privacy methods that can provide strong guarantees, often based on some notion of differential privacy (DP) [74], that bound the potential effectiveness of any attack of a certain type. Although both developing attacks and formal privacy proofs are important, conducting meaningful privacy audits is different from both approaches. Empirical methods, usually in the form of attack simulations, are inherently limited by the attacks considered and the uncertainty around better attacks, while theoretical proofs require many assumptions or result in loose bounds, and any claims based on theoretical results depends on careful analysis that the system as implemented is consistent with the theory. Thus, empirical audits may be able to provide a more meaningful bound than is possible with theory or experiments alone. If there is a theoretical result that prescribes an optimal attack, then empirical results with that attack (or approximations of the attack) can offer a more meaningful bound on privacy risk than is possible with theory or experiments alone. While the theory needs to cover all data distributions, experiments with the optimal attack focus on the actual distribution and given model, resulting in tighter and more relevant privacy evaluations.

Privacy audits can also be important in more adversarial contexts, where the audit is done by a regulator or external advocate to test a released model. Since auditors have elevated model access (via associated training environments, data, etc.), they can take advantage of more information to produce better estimates of what an adversary may be able to do without that information. Auditing is also orthogonal to proofs, like ones centred around DP and other privacy guarantees, for multiple reasons. As outlined by Cummings et al. [60], theoretical bounds may be "too conservative or inaccurate in some settings", and it may not always be

*This chapter is largely based on Anshuman Suri, Xiao Zhang, David Evans, *Do Parameters Reveal More than Loss for Membership Inference?*, in Workshop on High-dimensional Learning Dynamics (HiLD), ICML, 2024. Code relevant to this chapter is available as a Python package at https://github.com/iamgroot42/auditingmi.

possible to come up with proofs or theoretical bounds (which usually only apply to membership inference) that ensure models do not "violate disclosure requirements in ways that are not captured by differential privacy". Empirical auditing can provide a more meaningful measure of privacy leakage for these situations. For instance, Aerni et al. [4] demonstrate how DP with a large privacy budget can serve as a useful empirical defense, despite providing vacuous privacy guarantees.

The most common disclosure auditing approach today is to conduct membership inference attacks [165] as well as related attacks that attempt to extract specific data [60]. While membership inference assumes knowledge of the record, it constitutes a privacy risk when revealing inclusion of a known record in the training data itself leaks sensitive information (e.g., an individual's clinical data is included in a mental health study). In most scenarios, however, membership disclosure by itself is not a serious privacy risk, but rather used as a proxy for understanding leakage of information that may result in more serious privacy violations. Membership inference is popular in the research and industrial privacy communities because it is simple to define, relatively easy to measure, and aligns well with DP. This has resulted in it being widely used as a method for auditing disclosure risks for machine learning [18, 153, 165, 357]. Prior results on membership inference attacks have largely focused on the black-box setting, where the attacker only has input–output access to the target model. This focus has largely been reinforced by folklore and results demonstrating negligible gains from parameter access (white-box attacks) [43, 229]. A well-known result by Sablayrolles et al. [264], in fact, theoretically proves that black-box access is sufficient for optimal membership inference. This result has been the basis of several related works [52, 353]. However, the assumptions made in its derivation do not hold for most trained models, including ones trained with stochastic gradient descent (SGD).

Utilizing recent advances in the discrete-time SGD-dynamics literature [187, 378], we provide a more accurate formulation of the optimal membership inference attack that invalidates previous claims [264] about black-box access being optimal (§5.2). Our theoretical result also prescribes such an attack, which we call IHA (Inverse Hessian Attack) and empirically demonstrate its effectiveness in simple settings (§5.3). Motivated by our findings, we advocate for further research in white-box inference attacks (§5.5).

5.1 Preliminaries

5.1.1 Membership Inference

Following the framework established by Sablayrolles et al. [264], let $\mathcal{D}$ be a distribution from which n records $\boldsymbol{z}_1, \boldsymbol{z}_2, \ldots, \boldsymbol{z}_n$ are i.i.d. sampled with $\boldsymbol{z}_i = (\boldsymbol{x}_i, y_i)$ being the i-th record. Let $\boldsymbol{w} \in \mathbb{R}^d$ be the model parameters produced by some machine learning algorithm on a training dataset D. Assume $m_1, m_2, \ldots, m_n$ follow a Bernoulli distribution with $\gamma = \mathbb{P}(m_i = 1)$, where m_i is the membership indicator of $\boldsymbol{z}_i$ (i.e., $m_i = 1$ if $\boldsymbol{z}_i \in D$, and $m_i = 0$ otherwise). Given $\boldsymbol{w}$, an *membership inference attack* aims to predict the unknown membership m_i for any given record $\boldsymbol{z}_i$.

Definition 4 (Membership Inference). Let $\boldsymbol{w}$ be the parameters of the target model and $\boldsymbol{z}_1$ be a record. Inferring the membership of $\boldsymbol{z}_1$ to the training set of $\boldsymbol{w}$ is equivalent to computing:

$$\mathcal{M}(\boldsymbol{w}, \boldsymbol{z}_1) = \mathbb{P}(m_1 = 1 \mid \boldsymbol{w}, \boldsymbol{z}_1).$$

Let $\mathbb{P}(\boldsymbol{w} \mid \boldsymbol{z}_1, \ldots, \boldsymbol{z}_n, m_1, \ldots, m_n)$ be the posterior distribution of model parameters produced by some randomized machine learning algorithm (i.e., stochastic gradient descent). Applying Bayes' theorem, Sablayrolles et al. [264] derived an explicit formula for $\mathcal{M}(\boldsymbol{w}, \boldsymbol{z}_1)$:

Lemma 5.1.1 (Sablayrolles et al. [264]). *Let* $\mathcal{T} = \{\boldsymbol{z}_2, \ldots, \boldsymbol{z}_n, m_2, \ldots, m_n\}$. *Given model parameters* $\boldsymbol{w}$ *and a record* $\boldsymbol{z}_1$, *the optimal membership inference is given by:*

$$\mathcal{M}(\boldsymbol{w}, \boldsymbol{z}_1) = \mathbb{E}_{\mathcal{T}}\left[\sigma\left(\ln\left(\frac{p(\boldsymbol{w} \mid m_1 = 1, \boldsymbol{z}_1, \mathcal{T})}{p(\boldsymbol{w} \mid m_1 = 0, \boldsymbol{z}_1, \mathcal{T})}\right) + \ln\left(\frac{\gamma}{1-\gamma}\right)\right)\right], \tag{5.1}$$

where $\sigma(u) = (1 + \exp(-u))^{-1}$ *is the Sigmoid function, and* $\gamma = \mathbb{P}(m_1 = 1)$.

To use Lemma 5.1.1, one needs to characterize the posterior, $\mathbb{P}(\boldsymbol{w} \mid \boldsymbol{z}_1, \ldots, \boldsymbol{z}_n, m_1, \ldots, m_n)$, to make explicit the effect of the inferred record $\boldsymbol{z}_1$ on the optimal membership inference $\mathcal{M}(\boldsymbol{w}, \boldsymbol{z}_1)$. Recent advances in discrete-time SGD dynamics [187, 378] literature can help provide a connection with model parameters.

5.1.2 Discrete-time SGD Dynamics

A line of theoretical work [187, 270, 287, 339, 378] has analyzed the continuous- and discrete-time dynamics of stochastic gradient methods and provided insights for understanding deep learning generalization. Consider an SGD algorithm with the following update rule (for $t = 1, 2, 3, \ldots$):

$$\boldsymbol{g}_t = \nabla L(\boldsymbol{w}_{t-1}) + \boldsymbol{\eta}_{t-1} + \Gamma w_{t-1} \tag{5.2}$$

$$\boldsymbol{h}_t = \mu \cdot \boldsymbol{h}_{t-1} + \boldsymbol{g}_t \tag{5.3}$$

$$\boldsymbol{w}_t = \boldsymbol{w}_{t-1} - \lambda \cdot \boldsymbol{h}_t. \tag{5.4}$$

Here, $\mu \in [0, 1)$ is the momentum, $\lambda > 0$ is the learning rate, Γ is the L_2 regularization constant, and $\boldsymbol{\eta}_t = \frac{1}{S}\sum_{i \in \mathcal{B}_t} \nabla \ell(\boldsymbol{w}_{t-1}, \boldsymbol{z}_i) - \nabla L(\boldsymbol{w}_{t-1})$ represents the mini-batch noise, where $\mathcal{B}_t$ is a randomly sampled batch of examples with size S from D, $\ell(\boldsymbol{w}, \boldsymbol{z})$ is the individual loss, and $L(\boldsymbol{w}) = \frac{1}{n}\sum_{\boldsymbol{z} \in D} \ell(\boldsymbol{w}, \boldsymbol{z})$ denotes the total loss.

Assuming the model is trained using SGD according to the update rule defined by Equation 5.2 on a *quadratic loss* with the Hessian matrix $\mathbf{H}$ and arrives at a *stationary state*, Liu et al. [187] established a connection, shown in the following theorem, between the Hessian matrix $\mathbf{H}$, the asymptotic noise covariance

$\mathbf{C} = \lim_{t\to\infty} \mathbb{E}_{\boldsymbol{w}_t}[\mathrm{cov}(\boldsymbol{\eta}_t, \boldsymbol{\eta}_t)]$, and the asymptotic model fluctuation $\boldsymbol{\Sigma} = \lim_{t\to\infty} \mathrm{cov}(\boldsymbol{w}_t, \boldsymbol{w}_t)$. Their result depends on two key assumptions:

Assumption 5.1.2 (Stationary-State). *After a sufficient number of iterations, models trained with SGD defined by the update rule in Equation 5.2 arrive at a stationary state, i.e., the stationary model fluctuation* $\boldsymbol{\Sigma}$ *exists and is finite.*

Assumption 5.1.3 (Quadratic Loss). *The loss function* $L(\boldsymbol{w})$ *is either globally quadratic or locally quadratic close to a local minimum* $\boldsymbol{w}^*$*. Specifically, the loss function can be approximated as:*

$$L(\boldsymbol{w}) = L(\boldsymbol{w}^*) + \frac{1}{2}(\boldsymbol{w} - \boldsymbol{w}^*)^\top \mathbf{H}(\boldsymbol{w}^*)(\boldsymbol{w} - \boldsymbol{w}^*) + o(\|\boldsymbol{w} - \boldsymbol{w}^*\|_2^2), \tag{5.5}$$

where $\boldsymbol{w}^*$ *is a local minimum and* $\mathbf{H}(\boldsymbol{w}^*)$ *denotes the Hessian matrix at* $\boldsymbol{w}^*$*.*

With those assumptions, Liu et al. [187] prove the following theorem that describe model fluctuations of discrete SGD in a quadratic potential:

Theorem 5.1.4 (SGD Stationary distribution with momentum). *Let* $\boldsymbol{w}$ *be updated with SGD defined by the update rule in Equation 5.2 with momentum* $\mu \in [0, 1)$*. Given assumptions 5.1.2 and 5.1.3, if we additionally suppose* $\mathbf{C}$ *commutes with* $\mathbf{H}(\boldsymbol{w}^*)$*, then the stationary model fluctuation satisfies:*

$$\boldsymbol{\Sigma} = \left[\frac{\lambda \mathbf{H}(\boldsymbol{w}^*)}{1+\mu} \cdot \left(2\mathbf{I}_d - \frac{\lambda \mathbf{H}(\boldsymbol{w}^*)}{1+\mu}\right)\right]^{-1} \cdot \frac{\lambda^2 \mathbf{C}}{1-\mu^2}.$$

Theorem 5.1.4 requires the existence of a finite stationary noise covariance and the loss function to be quadratic close to a local minimum, which are mild assumptions (see [187] for detailed discussions).

In follow-up work, Ziyin et al. [378] further derived the explicit dependence of the asymptotic noise covariance $\mathbf{C}$ to the loss and Hessian around a local minimum $\boldsymbol{w}^*$ under mild assumptions.

Theorem 5.1.5 (SGD Noise Covariance). *Let* $L(\boldsymbol{w})$ *be the training loss and the model* $\boldsymbol{w}$ *is optimized with SGD defined by Equation 5.2 in the neighborhood of a local minimum* $\boldsymbol{w}^*$*. If* $L(\boldsymbol{w}^*) \neq 0$*, then*

$$\mathbf{C} = \frac{2L(\boldsymbol{w}^*)}{S} \cdot \mathbf{H}(\boldsymbol{w}^*) - \frac{\Gamma^2}{S}\|\boldsymbol{w}^*\|^2 + O(S^{-2}) + O(\|\boldsymbol{w} - \boldsymbol{w}^*\|_2^2) + o(L(\boldsymbol{w}^*)),$$

provided that $\boldsymbol{\Sigma}$ *is proportional to* S^{-1} *and* $|L(\boldsymbol{w}) - \ell(\boldsymbol{w}, \boldsymbol{z}_i)|$ *is small.*

Theorem 5.1.5 implies that the SGD noise covariance $\mathbf{C}$ commutes with the Hessian matrix $\mathbf{H}(\boldsymbol{w}^*)$.

Considering only the leading term in the noise covariance, Liu et al. [187] derive the following formula for the stationary model fluctuation of SGD based on the above two theorems:

$$\boldsymbol{\Sigma} = \frac{\lambda}{S(1-\mu)} \cdot \left(2L(\boldsymbol{w}^*)\mathbf{H}(\boldsymbol{w}^*) - \Gamma^2\|\boldsymbol{w}^*\|^2\right)\left(\mathbf{H}(\boldsymbol{w}^*) + \Gamma\right)^{-1}\left(2I_d - \frac{\lambda}{1+\mu}\left(\mathbf{H}(\boldsymbol{w}^*) + \Gamma\right)\right)^{-1}. \tag{5.6}$$

We remark that if $L(\boldsymbol{w}^*) = 0$ (i.e., $\boldsymbol{w}^*$ is a global minimum), then $\boldsymbol{\Sigma} = 0$. In addition, if the Hessian matrix $(\mathbf{H}(\boldsymbol{w}^*)+\Gamma)$ has degenerated rank $r < d$, then $(\mathbf{H}+\Gamma)^{-1}$ can be replaced by the corresponding Moore-Penrose pseudo inverse. Accordingly, similar results to Equation 5.6 can be obtained by considering the projection space spanned by eigenvectors with non-zero eigenvalues. Section 5 of Ziyin et al. [378] provides more detailed discussions of the imposed assumptions and the implications of the results.

5.2 Black-Box Access is not Sufficient

In this section, we examine previous assertions concerning optimal membership inference (§5.2.1) and illustrate that for models trained with SGD, the optimal membership inference adversary does, in fact, require parameter access (§5.2.2). Our theory presents an attack based on this theory that is valuable for conducting privacy audits (§5.2.3).

5.2.1 Limitations of Claims of Black-Box Optimality

Sablayrolles et al. [264] proved the optimality of black-box membership inference under a Bayesian framework. In particular, they assume (Equation 1 in [264]) that the posterior distribution of model parameters $\boldsymbol{w}$ trained on $\boldsymbol{z}_1, \ldots, \boldsymbol{z}_n$ with membership $m_1, \ldots, m_n$ follows:

$$\mathbb{P}(\boldsymbol{w} \mid \boldsymbol{z}_1, \ldots, \boldsymbol{z}_n) \propto \exp\left(-\frac{1}{T}\sum_{i=1}^{n} m_i \cdot \ell(\boldsymbol{w}, \boldsymbol{z}_i)\right), \tag{5.7}$$

where T is a temperature parameter that captures the stochasticity of the learning algorithm. This assumption makes subsequent derivations of optimal membership inference much easier, but oversimplifies the training dynamics of typical machine learning algorithms such as SGD. Equation 5.7 assumes that the posterior of $\boldsymbol{w}$ follows a Boltzmann distribution that only depends on the training loss. This is desirable for Bayesian posterior inference, where the goal is to provide a sampling strategy for an unknown data distribution given a set of observed data samples. This can be achieved using SGLD [339] with shrinking step size λ_t (i.e., $\lim_{t\to\infty} \lambda_t = 0$) and by injecting carefully-designed Gaussian noise $\mathcal{N}(\mathbf{0}, \lambda_t \cdot \mathbf{I}_D)$. However, this special SGLD design differs from the common practice of SGD algorithms used to train neural networks for the following two reasons:

1. All analyses are performed under continuous-time dynamics, whereas actual SGD is performed with discrete steps. While related work such as Stephan et al. [287] cast the continuous-time dynamics of SGD as a multivariate *Ornstein-Uhlenbeck* process (similar to SGLD) whose stationary distribution is

proven to be Gaussian (Equations 11-12 in [287]), they make additional assumptions such as the noise covariance matrix being independent of model parameters.

2. SGLD assumes a vanishing learning rate until convergence, whereas SGD is performed with a non-vanishing step size and for a finite number of iterations in practice. The learning rate of SGD is often large, which can cause model dynamics to drift even further from SGLD [379], especially under the discrete-time setting [187].

We thus characterize the analytical form of the posterior distribution with respect to model parameters trained with SGD.

Theorem 5.2.1 (Posterior for SGD). *Assume the same assumptions as used in Theorems 5.1.4 and 5.1.5. Let $\boldsymbol{w}^*$ be the local minimum that SGD (Equation 5.2) is converging towards. Then, the (conditional) log-probability of observing parameters $\boldsymbol{w}$ is given by (up to constants and negligible terms):*

$$-\frac{d}{2}\ln L(\boldsymbol{w}^*)+\sum_{i=1}^{d}\ln\left(\frac{\left(2-\frac{\lambda}{1+\mu}(\sigma_i(\boldsymbol{H}_*)+\Gamma)\right)(\sigma_i(\boldsymbol{H}_*)+\Gamma)}{\sigma_i(\boldsymbol{H}_*)}\right)-\frac{S(1-\mu)}{2L_*\lambda}\left(1-\frac{\lambda\Gamma}{1+\mu}\right)\|\boldsymbol{w}-\boldsymbol{w}^*\|^2$$
$$-\frac{S(1-\mu)\Gamma}{2L_*\lambda}\nabla L(\boldsymbol{w})^\top H^{-3}\nabla L(\boldsymbol{w})+\frac{S(1-\mu)}{2(1+\mu)}\cdot\frac{L(\boldsymbol{w})}{L_*}$$

where $\sigma_i(\mathbf{H}(\boldsymbol{w}^*))$ denotes the i-th largest eigenvalue of $\mathbf{H}(\boldsymbol{w}^*)$.

Proof. According to Theorem 5.1.4 and Theorem 5.1.5, we obtain

$$\boldsymbol{\Sigma}=\frac{\lambda}{S(1-\mu)}\cdot\left(2L(\boldsymbol{w}^*)\mathbf{H}(\boldsymbol{w}^*)-\Gamma^2\|\boldsymbol{w}^*\|^2\right)\left(\mathbf{H}(\boldsymbol{w}^*)+\Gamma\right)^{-1}\left(2I_d-\frac{\lambda}{1+\mu}(\mathbf{H}(\boldsymbol{w}^*)+\Gamma)\right)^{-1}. \tag{5.8}$$

Note that the above equation holds when the Hessian matrix $\mathbf{H}(\boldsymbol{w}^*)$ has full rank and $L(\boldsymbol{w}^*)\neq 0$.

When the Hessian has degenerated rank such that $\text{rank}(H+\Gamma)=r<d$, the following more generalized result can be derived:

$$\mathbf{P}_r\boldsymbol{\Sigma}=\frac{\lambda}{S(1-\mu)}\cdot\left(2L(\boldsymbol{w}^*)\mathbf{H}(\boldsymbol{w}^*)-\Gamma^2\|\boldsymbol{w}^*\|^2\right)\left(\mathbf{H}(\boldsymbol{w}^*)+\Gamma\right)^{+}\left(2I_d-\frac{\lambda}{1+\mu}\left(\mathbf{H}(\boldsymbol{w}^*)+\Gamma\right)\right)^{-1},$$

where $\mathbf{P}_r=\text{diag}(1,...,1,0,...0)$ denotes the projection matrix onto non-zero eigenvalues, and $+$ is the Moore-Penrose pseudo inverse. If $L(\boldsymbol{w}^*)=0$, meaning $\boldsymbol{w}^*$ is a global minimum, then the asymptotic model fluctuation $\boldsymbol{\Sigma}=\mathbf{0}$. For the ease of presentation, let $\mathbf{H}_*=\mathbf{H}(\boldsymbol{w}^*)$, $L_*=L(\boldsymbol{w}^*)$ and the Hessian matrix have full rank in the following proof. According to Laplace approximation, we can approximate the posterior distribution of $\boldsymbol{w}$ given $\boldsymbol{w}^*$ as $\mathcal{N}(\boldsymbol{w}^*,\boldsymbol{\Sigma})$. Therefore, making use of Equation 5.8, we can derive the explicit

formula of the log-posterior distribution as:

$$
\begin{aligned}
\ln p(\boldsymbol{w}|\boldsymbol{w}^*) &= -\frac{d}{2}\ln(2\pi) - \frac{1}{2}\ln\det(\boldsymbol{\Sigma}) - \frac{1}{2}(\boldsymbol{w}-\boldsymbol{w}^*)^\top \boldsymbol{\Sigma}^{-1}(\boldsymbol{w}-\boldsymbol{w}^*) \\
&= \frac{1}{2}\sum_{i=1}^{d}\ln\left(\frac{\left(2-\frac{\lambda}{1+\mu}(\sigma_i(\boldsymbol{H}_*)+\Gamma)\right)\left(\sigma_i(\boldsymbol{H}_*)+\Gamma\right)}{2L(\boldsymbol{w}^*)\sigma_i(\boldsymbol{H}_*)-\Gamma^2\|\boldsymbol{w}^*\|^2}\right) \\
&\quad - \frac{S(1-\mu)}{4L_*\lambda}\left[2\left(1-\frac{\lambda\Gamma}{1+\mu}\right)\|\boldsymbol{w}-\boldsymbol{w}^*\|^2 + 2\Gamma(\boldsymbol{w}-\boldsymbol{w}^*)^\top H^{-1}(\boldsymbol{w}-\boldsymbol{w}^*) - \right. \\
&\quad \left. \frac{\lambda}{1+\mu}(\boldsymbol{w}-\boldsymbol{w}^*)^\top H(\boldsymbol{w}-\boldsymbol{w}^*)\right] + \text{const.} \\
&= \frac{1}{2}\sum_{i=1}^{d}\ln\left(\frac{\left(2-\frac{\lambda}{1+\mu}(\sigma_i(\boldsymbol{H}_*)+\Gamma)\right)\left(\sigma_i(\boldsymbol{H}_*)+\Gamma\right)}{L(\boldsymbol{w}^*)\sigma_i(\boldsymbol{H}_*)}\right) \\
&\quad - \frac{S(1-\mu)}{2L_*\lambda}\left(1-\frac{\lambda\Gamma}{1+\mu}\right)\|\boldsymbol{w}-\boldsymbol{w}^*\|^2 - \frac{S(1-\mu)\Gamma}{2L_*\lambda}(\boldsymbol{w}-\boldsymbol{w}^*)^\top H^{-1}(\boldsymbol{w}-\boldsymbol{w}^*) \\
&\quad + \frac{S(1-\mu)}{4L_*(1+\mu)}(\boldsymbol{w}-\boldsymbol{w}^*)^\top H(\boldsymbol{w}-\boldsymbol{w}^*) + \text{cost.} \\
&= -\frac{d}{2}\ln L(\boldsymbol{w}^*) + \frac{1}{2}\sum_{i=1}^{d}\ln\left(\frac{\left(2-\frac{\lambda}{1+\mu}(\sigma_i(\boldsymbol{H}_*)+\Gamma)\right)\left(\sigma_i(\boldsymbol{H}_*)+\Gamma\right)}{\sigma_i(\boldsymbol{H}_*)}\right) \\
&\quad - \frac{S(1-\mu)}{2L_*\lambda}\left(1-\frac{\lambda\Gamma}{1+\mu}\right)\|\boldsymbol{w}-\boldsymbol{w}^*\|^2 - \frac{S(1-\mu)\Gamma}{2L_*\lambda}\nabla L(\boldsymbol{w})^\top H^{-3}\nabla L(\boldsymbol{w}) \\
&\quad + \frac{S(1-\mu)}{2(1+\mu)}\cdot\frac{L(\boldsymbol{w})}{L_*} + o(\|\boldsymbol{w}-\boldsymbol{w}^*\|_2^2) + \text{const.}
\end{aligned}
$$

Here, the last equality holds because of the second-order Taylor expansion of $L(\boldsymbol{w})$ at $\boldsymbol{w}^*$. Omitting the constant and negligible terms (and terms corresponding to $\Gamma^2\|\boldsymbol{w}^*\|^2$) in the above equation, we thus complete the proof of Theorem 5.2.1. □

5.2.2 Optimal Membership Inference under Discrete-time SGD

So far, we have explained why the critical assumption imposed by Sablayrolles et al. [264] about the posterior distribution of $\boldsymbol{w}$ following a Boltzmann distribution (Equation 5.7) does not hold for typical stochastic gradient methods employed in practice. We now prove a theorem that gives an estimate on the optimal scoring function for membership inference for models produced by SGD by leveraging the recent theoretical literature on discrete-time SGD dynamics [187, 378].

We assume that the loss achieved at the local minimum remains unaffected by the removal of a single training record and that the Hessian structure remains unchanged:

Assumption 5.2.2 (Similarity at local minimum). *For any $\mathcal{T}$ and $\boldsymbol{z}_1$, let $L_0(\boldsymbol{w}) = \frac{1}{n}\sum_{i=2}^{n} m_i\ell(\boldsymbol{w},\boldsymbol{z}_i)$ and $L_1(\boldsymbol{w}) = \frac{1}{n}(\ell(\boldsymbol{w},\boldsymbol{z}_1) + \sum_{i=2}^{n} m_i\ell(\boldsymbol{w},\boldsymbol{z}_i))$. Assume the Hessian matrix shares a similar structure when the model's training data differs by a single point, and the loss function also achieves a similar value at the local*

minimum, i.e.,

$$\mathbf{H}_* = \mathbf{H}_1(\boldsymbol{w}_1^*) = \mathbf{H}_0(\boldsymbol{w}_0^*), \quad L_* = L_1(\boldsymbol{w}_1^*) = L_0(\boldsymbol{w}_0^*), \tag{5.9}$$

where $\boldsymbol{w}_1^*$ (resp. $\boldsymbol{w}_0^*$) is the local minimum that SGD with L_1 (resp. L_0) is converging towards, and $\mathbf{H}_1$ (resp. $\mathbf{H}_0$) denotes the Hessian matrix with respect to L_1 (resp. L_0).

As long as the size of the training dataset is sufficient and the excluded training record $\boldsymbol{z}_1$ is not a low-probability outlier from the data distribution $\mathcal{D}$, we expect Assumption 5.2.2 generally holds for SGD algorithms. Under Assumption 5.2.2 and a few other assumptions imposed in prior literature on discrete-time SGD dynamics [187, 378], we obtain the following theorem that describes the scoring function for an optimal membership-inference adversary.

Theorem 5.2.3 (Optimal Membership-Inference Score). *Given $\boldsymbol{w}$ produced by an SGD algorithm defined by Equation 5.2 and a record $\boldsymbol{z}_1$, the optimal membership inference $\mathcal{M}(\boldsymbol{w}, \boldsymbol{z}_1)$ is given by:*

$$\mathbb{E}_{\mathcal{T}}\left[\sigma\left(\frac{S(1-\mu)}{2nL_*}\cdot\left(\frac{\ell(\boldsymbol{w},\boldsymbol{z}_1)}{1+\mu} - \frac{1}{\lambda}(I_1+I_2+I_3+I_4)\right) + \ln\left(\frac{\gamma}{1-\gamma}\right)\right)\right], \tag{5.10}$$

where I_1, I_2, I_3, and I_4 are defined as follows:

$$\begin{aligned}
I_1 &:= \frac{1}{n}\left(1 - \frac{\lambda\Gamma}{1+\mu}\right)\|\mathbf{H}_*^{-1}\nabla\ell(\boldsymbol{w},\boldsymbol{z}_1)\|^2,\\
I_2 &:= 2\left(1 - \frac{\lambda\Gamma}{1+\mu}\right)\left(\mathbf{H}_*^{-1}\nabla L_0(\boldsymbol{w})\right)^\top\left(\mathbf{H}_*^{-1}\nabla\ell(\boldsymbol{w},\boldsymbol{z}_1)\right),\\
I_3 &:= \frac{\Gamma}{n}\left(\mathbf{H}_*^{-1}\nabla\ell(\boldsymbol{w},\boldsymbol{z}_1)\right)^\top\left(\mathbf{H}_*^{-1}\left(\mathbf{H}_*^{-1}\nabla\ell(\boldsymbol{w},\boldsymbol{z}_1)\right)\right)\\
I_4 &:= 2\Gamma\left(\mathbf{H}_*^{-1}\nabla L_0(\boldsymbol{w})\right)^\top\left(\mathbf{H}_*^{-1}\left(\mathbf{H}_*^{-1}\nabla\ell(\boldsymbol{w},\boldsymbol{z}_1)\right)\right)
\end{aligned}$$

Here, L_* and $\mathbf{H}_*$ are defined in Assumption 5.2.2, which are dependent on $\mathcal{T}$.

Proof. To derive the scoring function for an optimal membership inference , we need to compute the ratio between $p(\boldsymbol{w}|\boldsymbol{w}_1^*)$ and $p(\boldsymbol{w}|\boldsymbol{w}_0^*)$, where $\boldsymbol{w}_0^*$ (resp. $\boldsymbol{w}_1^*$) denotes a local minimum (close to $\boldsymbol{w}$) of the training loss function with respect to $\{\boldsymbol{z}_2,\ldots,\boldsymbol{z}_n\}$ (resp. $\{\boldsymbol{z}_1,\ldots,\boldsymbol{z}_n\}$). Note that we've obtained the posterior

distribution of $\boldsymbol{w}$ in Theorem 5.2.1. Therefore, the remaining task is to analyze the following terms:

$$
\begin{aligned}
&\ln p(\boldsymbol{w}|\boldsymbol{w}_1^*) - \ln p(\boldsymbol{w}|\boldsymbol{w}_0^*) \\
&= -\frac{D}{2}\big[\ln L_1(\boldsymbol{w}_1^*) - \ln L_0(\boldsymbol{w}_0^*)\big] + \frac{1}{2}\sum_{i=1}^{d} \ln\left(\frac{\big(2-\frac{\lambda}{1+\mu}(\sigma_i(\boldsymbol{H}_1(\boldsymbol{w}_1^*)))+\Gamma\big)\big(\sigma_i(\boldsymbol{H}_1(\boldsymbol{w}_1^*))+\Gamma\big)}{\big(2-\frac{\lambda}{1+\mu}(\sigma_i(\boldsymbol{H}_0(\boldsymbol{w}_0^*)))+\Gamma\big)\big(\sigma_i(\boldsymbol{H}_0(\boldsymbol{w}_0^*))+\Gamma\big)} \cdot \frac{\sigma_i(\boldsymbol{H}_1(\boldsymbol{w}_1^*))}{\sigma_i(\boldsymbol{H}_0(\boldsymbol{w}_0^*))}\right) \\
&\quad - \frac{S(1-\mu)}{2\lambda}\left(1-\frac{\lambda\Gamma}{1+\mu}\right)\left(\frac{\|\boldsymbol{w}-\boldsymbol{w}_1^*\|^2}{L_1(\boldsymbol{w}_1^*)} - \frac{\|\boldsymbol{w}-\boldsymbol{w}_0^*\|^2}{L_0(\boldsymbol{w}_0^*)}\right) \\
&\quad - \frac{S(1-\mu)\Gamma}{2\lambda}\left(\frac{\nabla L_1(w)^\top \boldsymbol{H}_1(\boldsymbol{w}_1^*)^{-3}\nabla L_1(w)}{L_1(\boldsymbol{w}_1^*)} - \frac{\nabla L_0(w)^\top \boldsymbol{H}_0(\boldsymbol{w}_0^*)^{-3}\nabla L_0(w)}{L_0(\boldsymbol{w}_0^*)}\right) \\
&\quad + \frac{S(1-\mu)}{2(1+\mu)} \cdot \left(\frac{L_1(\boldsymbol{w})}{L_1(\boldsymbol{w}_1^*)} - \frac{L_0(\boldsymbol{w})}{L_0(\boldsymbol{w}_0^*)}\right),
\end{aligned} \tag{5.11}
$$

where the constant and small $o(\cdot)$ terms are neglected, and $\mathbf{H}_0(\boldsymbol{w}_0^*)$ (resp. $\mathbf{H}_1(\boldsymbol{w}_1^*)$) denotes the Hessian of L_0 (resp. L_1) at $\boldsymbol{w}_0^*$ (resp. $\boldsymbol{w}_1^*$). Since both $\boldsymbol{w}_0^*$ and $\boldsymbol{w}_1^*$ are close to parameters of the observed victim model $\boldsymbol{w}$, so we can approximate the corresponding loss using second-order Taylor expansion. Also, according to Assumption 5.2.2, we know $\mathbf{H}_0(\boldsymbol{w}_0^*) = \mathbf{H}_1(\boldsymbol{w}_1^*) = \mathbf{H}_*$ and $L_0(\boldsymbol{w}_0^*) = L_1(\boldsymbol{w}_1^*) = L_*$. Thus, we can simplify Equation 5.11 and obtain the following form:

$$
\begin{aligned}
&- \frac{S(1-\mu)}{2\lambda L_*}\left(1-\frac{\lambda\Gamma}{1+\mu}\right)\big(\|\boldsymbol{w}-\boldsymbol{w}_1^*\|_2^2 - \|\boldsymbol{w}-\boldsymbol{w}_0^*\|_2^2\big) + \frac{S(1-\mu)}{2(1+\mu)L_*}\big(L_1(\boldsymbol{w}) - L_0(\boldsymbol{w})\big) \\
&\quad - \frac{S(1-\mu)\Gamma}{2\lambda L_*}\big(\nabla L_1(w)^\top \boldsymbol{H}_*^{-3}\nabla L_1(w) - \nabla L_0(w)^\top \boldsymbol{H}_*^{-3}\nabla L_0(w)\big) \\
&= -\frac{S(1-\mu)}{2\lambda L_*}\left(1-\frac{\lambda\Gamma}{1+\mu}\right)\big(\nabla L_1(\boldsymbol{w})^\top \mathbf{H}_*^{-1}\mathbf{H}_*^{-1}\nabla L_1(\boldsymbol{w}) - \nabla L_0(\boldsymbol{w})^\top \mathbf{H}_*^{-1}\mathbf{H}_*^{-1}\nabla L_0(\boldsymbol{w})\big) + \frac{S(1-\mu)\ell(\boldsymbol{w},\boldsymbol{z}_1)}{2n(1+\mu)L_*} \\
&\quad - \frac{S(1-\mu)\Gamma}{2\lambda L_* n}\left(2\nabla L_0(\boldsymbol{w})^\top \mathbf{H}_*^{-3}\nabla \ell(\boldsymbol{w},\boldsymbol{z}_1) + \frac{1}{n}\|\mathbf{H}_*^{-3}\nabla\ell(\boldsymbol{w},\boldsymbol{z}_1)\|^2\right) \\
&= -\frac{S(1-\mu)}{2\lambda L_* n}\left(1-\frac{\lambda\Gamma}{1+\mu}\right)\left(2\nabla L_0(\boldsymbol{w})^\top \mathbf{H}_*^{-1}\mathbf{H}_*^{-1}\nabla \ell(\boldsymbol{w},\boldsymbol{z}_1) + \frac{1}{n}\|\mathbf{H}_*^{-1}\nabla\ell(\boldsymbol{w},\boldsymbol{z}_1)\|^2\right) + \frac{S(1-\mu)\ell(\boldsymbol{w},\boldsymbol{z}_1)}{2n(1+\mu)L_*} \\
&\quad - \frac{S(1-\mu)\Gamma}{2\lambda L_* n}\left(2\nabla L_0(\boldsymbol{w})^\top \mathbf{H}_*^{-3}\nabla \ell(\boldsymbol{w},\boldsymbol{z}_1) + \frac{1}{n}\|\mathbf{H}_*^{-3}\nabla\ell(\boldsymbol{w},\boldsymbol{z}_1)\|^2\right)
\end{aligned} \tag{5.12}
$$

where the second equality holds because of the Taylor approximation $\nabla L_i(\boldsymbol{w}) - \nabla L_i(\boldsymbol{w}_i^*) = \mathbf{H}_*(\boldsymbol{w}-\boldsymbol{w}_i^*)$ for $i \in \{0,1\}$. Moreover, according to Lemma 5.1.1, we know the optimal membership inference is given by:

$$
\mathcal{M}(\boldsymbol{w},\boldsymbol{z}_1) = \mathbb{E}_{\mathcal{T}}\left[\sigma\left(\ln\left(\frac{p(\boldsymbol{w}|m_1=1,\boldsymbol{z}_1,\mathcal{T})}{p(\boldsymbol{w}|m_1=0,\boldsymbol{z}_1,\mathcal{T})}\right) + \ln\left(\frac{\gamma}{1-\gamma}\right)\right)\right], \tag{5.13}
$$

where $\sigma(u) = (1+\exp(-u))^{-1}$ is the Sigmoid function, $\mathcal{T} = \{\boldsymbol{z}_2,\ldots,\boldsymbol{z}_n,m_2,\ldots,m_n\}$, and $\gamma = \mathbb{P}(m_i=1)$. Plugging Equation 5.12 into Equation 5.13, we obtain

$$
\mathcal{M}(\boldsymbol{w},\boldsymbol{z}_1) = \mathbb{E}_{\mathcal{T}}\left[\sigma\left(\frac{S(1-\mu)}{2nL_*}\left(\frac{\ell(\boldsymbol{w},\boldsymbol{z}_1)}{(1+\mu)} - \frac{1}{\lambda}\big(I_1+I_2+I_3+I_4\big)\right) + t_\gamma\right)\right],
$$

where I_1, I_2, I_3, I_4 and t_γ are defined as:

$$
\begin{aligned}
I_1 &:= \frac{1}{n}\|\mathbf{H}_*^{-1}\nabla\ell(\boldsymbol{w}, \boldsymbol{z}_1)\|^2, \\
I_2 &:= 2\big(\mathbf{H}_*^{-1}\nabla L_0(\boldsymbol{w})\big)^\top\big(\mathbf{H}_*^{-1}\nabla\ell(\boldsymbol{w}, \boldsymbol{z}_1)\big), \\
I_3 &:= \frac{\Gamma}{n}\big(\mathbf{H}_*^{-1}\nabla\ell(\boldsymbol{w}, \boldsymbol{z}_1)\big)^\top\big(\mathbf{H}_*^{-1}\big(\mathbf{H}_*^{-1}\nabla\ell(\boldsymbol{w}, \boldsymbol{z}_1)\big)\big), \\
I_4 &:= 2\Gamma\big(\mathbf{H}_*^{-1}\nabla L_0(\boldsymbol{w})\big)^\top\big(\mathbf{H}_*^{-1}\big(\mathbf{H}_*^{-1}\nabla\ell(\boldsymbol{w}, \boldsymbol{z}_1)\big)\big), \\
t_\gamma &:= \ln\left(\frac{\gamma}{1-\gamma}\right).
\end{aligned}
$$

□

Note that computing the optimal score requires access to the Hessian and model gradients, both of which require access to model parameters. In fact, knowledge of the learning rate λ, momentum μ, and regularization Γ are also required, thus requiring complete knowledge of the training setup of the target model. Thus, black-box access is thus **not** sufficient for optimal membership inference.

5.2.3 Inverse Hessian Attack

While Theorem 5.2.3 directly prescribes an optimal membership inference adversary, computing the expectation over $\mathcal{T}$ is infeasible. We thus make use of the insight of Theorem 5.2.3 to propose a scoring function based on the terms inside the expectation:

$$
\mathrm{IHA}(\boldsymbol{z}_1) := \frac{\ell(\boldsymbol{w}, \boldsymbol{z}_1)}{1+\mu} - \frac{1}{\lambda}\big(I_1 + I_2 + I_3 + I_4\big). \tag{5.14}
$$

This score $\mathrm{IHA}(\boldsymbol{z}_1)$, for some given record $\boldsymbol{z}_1$, can be used as the probability of $\boldsymbol{z}_1$ being a member and subsequently serve as a useful attack for privacy auditing. While the optimal attack prescribed by our theory requires expectation over all possible $\mathcal{T}$, we directly use $\mathrm{IHA}(\boldsymbol{z}_1)$ without any reference models[†]. Apart from the absence of reference models to compute this expectation, the performance of our attack is also influenced by other factors, such as how efficiently and accurately the inverse-Hessian vector products (iHVPs) can be computed and to what degree our assumptions hold (particularly Assumption 5.2.2).

5.3 Experiments

We evaluate IHA over multiple datasets with models where the Hessian can be computed directly, with some damping applied to deal with near-zero eigenvalues. Our results (§5.3.4) demonstrate that IHA provides a robust privacy auditing baseline, matching or exceeding the performance of current state-of-the-art attacks

[†]Adapting our attack to work with reference models in the general setting remains a future interesting direction.

that utilize reference models, all without requiring the training of reference models or the use of hold-out data.

5.3.1 Baseline Attacks

LOSS [355]. For this attack, the negative loss is used directly as a signal for membership inference.

SIF [59]. This attack, similar to ours, also uses the loss curvature of the target model by calculating its Hessian, which is then used to compute self-influence as a score. The original attack assigns 0–1 scores to target records. It classifies a given record as a member if its self-influence score is within the specified range and if its predicted class is correct. The latter rule can be immediately ruled out as having many false positives/negatives. Instead of these steps, we choose to use the self-influence as membership scores directly. While the authors used approximation methods for iHVP, we use the exact Hessian for fair comparison.

LiRA [43]. There are two variants, LiRA-Offline and LiRA-Online. The former uses "offline" models to estimate a Gaussian distribution and then performs one-sided hypothesis testing using loss scores. The LiRA-Online variant additionally employs "online" models, i.e., models whose training data included the target record. The likelihood ratio for online/offline model score distributions is then used as the score for membership inference. We use LiRA-Online, since it is the stronger of the two variants.

5.3.2 Datasets and Models

MNIST-Odd. We consider the MNIST dataset [167], with the modified task of classifying a given digit image as odd or even. We train a logistic regression model with mean-squared error loss, with an average test loss of .078.

FashionMNIST. We use the FashionMNIST [346] dataset, where the task is to classify a given clothing item image into one of ten categories. We train 2-layer MLPs (6 hidden neurons) with cross-entropy loss, with an average test accuracy of 83%.

Purchase-100(S). The task for this dataset [279] is to classify a given purchase into one of 100 categories, given 600 features. We train 2-layer MLPs (32 hidden neurons) with cross-entropy loss, with an average test accuracy of 84%. Experiments in the prior literature [365] train larger (4-layer MLP) models on 25K samples from Purchase-100, which is much smaller than the actual dataset, which is why we term it Purchase-100(S) (Small). We also demonstrate results with the 4-layer MLP that achieves similar task accuracy.

Purchase-100. For this version, we train models with 80K samples. We use the same 2-layer MLP architecture as Purchase-100(S) but achieve a higher test accuracy of 90%. Utilizing more data increases the scope for model performance.

We train 128 models in the same way as done in Carlini et al. [43], where data from each model is sampled at random from the actual dataset with a 50% probability. For each target model and target record, there are thus 127 reference models available, half of which (in expectation) include the target record in the training data, and the other half do not. All of our models are trained with momentum ($\mu = 0.9$) and without regularization ($\Gamma = 0$). For a given false positive rate (FPR), a threshold is computed using scores for non-members, which is then used to compute the corresponding true positive rate (TPR). This is then repeated for multiple FPRs to generate the corresponding ROC curve, which is used to compute the AUC. This experimental design is commonly used for membership-inference evaluations [43, 353, 355].

5.3.3 Implementing the Inverse Hessian Attack

In order to carry out IHA, an auditor needs to be able to calculate iHVPs and gradients for all training data. While computing gradients is more computationally intensive than simply calculating the loss, the difference is minimal. On the other hand, computing an iHVP involves calculating the Hessian matrix and then inverting it, both of which are computationally expensive processes. Although there are more efficient methods for approximating iHVPs, they are still considerably slower than simple gradient computation. For some given record $\boldsymbol{z}_1$, $\nabla L_0(\boldsymbol{w})$ can be computed by considering all data (except the target record) for which membership is known. To make this step computationally efficient for an audit, we pre-compute $\nabla L_1(\boldsymbol{w})$. Then, if the test record is indeed a member, we can compute $\nabla L_0(\boldsymbol{w})$ as $\nabla L_1(\boldsymbol{w}) - \frac{\nabla \ell(\boldsymbol{w}, \boldsymbol{z}_1)}{n}$. Note that this is equivalent to computing $\nabla L_0(\boldsymbol{w})$ separately for each target record. The Hessian $\mathbf{H}_*$ is also similarly pre-computed using the model's training data.

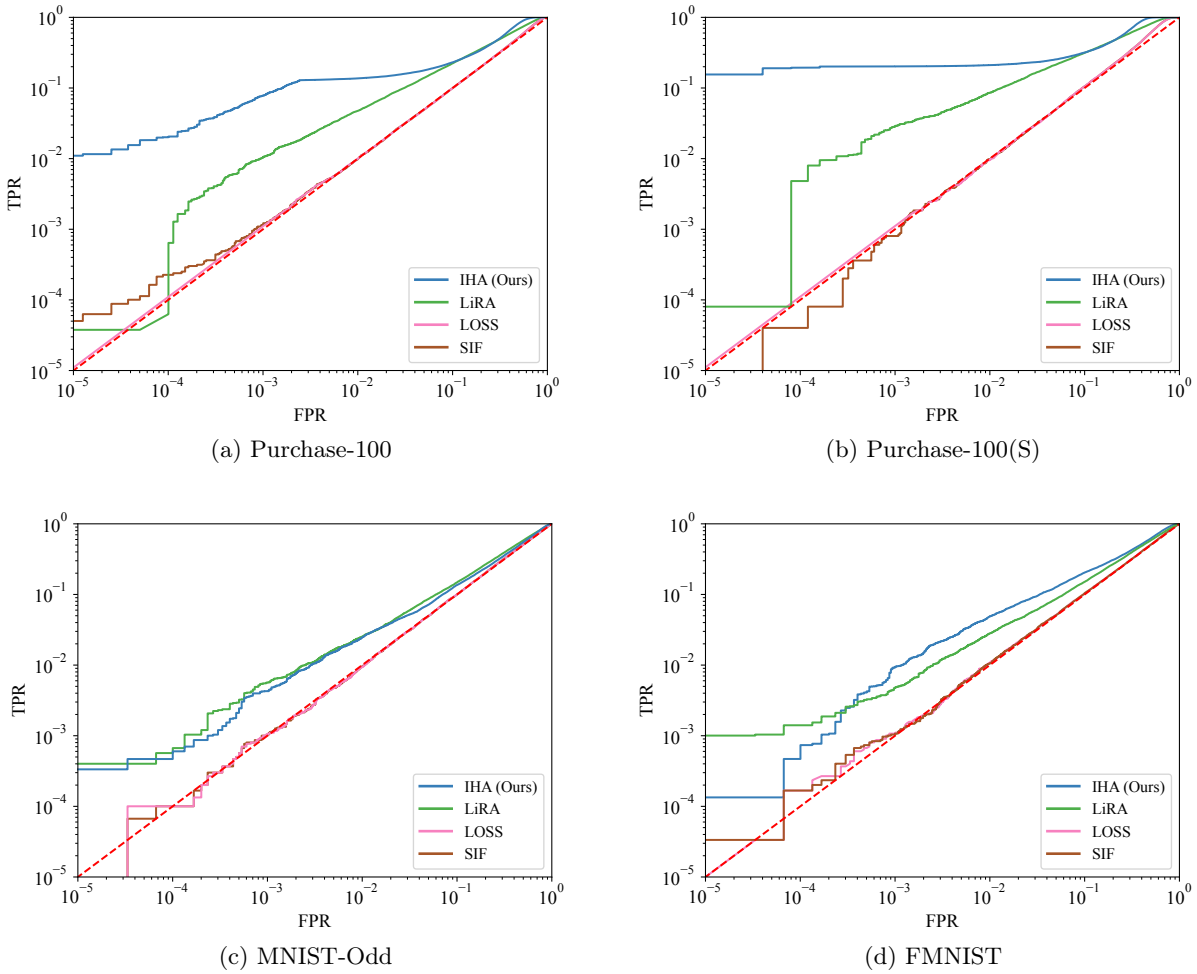
Conditioning $\mathbf{H}_*$. While computing Hessian matrices for our experiments, we notice the presence of near-zero and small, negative eigenvalues (most of which are likely to arise from precision errors). Such eigenvalues make the Hessian ill-conditioned and thus cannot be inverted directly. We explore two different techniques to mitigate this: damping by adding a small constant ϵ to all the eigenvalues or a low-rank approximation where only eigenvalues (and corresponding eigenvectors) above a certain threshold ϵ are used as a low-rank approximation. We ablate over these two techniques for some candidate values of ϵ and find that damping with $\epsilon = 2e^{-1}$ works best, which is the setting for which we report our main results.

5.3.4 Results

Our results are summarized in Table 5.1, showing that IHA provides a robust privacy auditing baseline, matching or exceeding the performance of current state-of-the-art attacks that utilize reference models, all

Table 5.1: Performance of various attacks, reported via attack AUC and true positive rate (TPR) at low false positive rate (FPR).

Attack	Purchase-100			MNIST-Odd			FashionMNIST		
	AUC	TPR@FPR		AUC	TPR@FPR		AUC	TPR@FPR	
		1%	0.1%		1%	0.1%		1%	0.1%
LOSS [355]	$.531_{\pm.001}$	.100	.010	$.500_{\pm.002}$	.100	.010	$.507_{\pm.002}$	.099	.010
SIF [59]	$.530_{\pm.001}$	.100	.010	$.500_{\pm.002}$	.100	.010	$.507_{\pm.002}$	.099	.010
LiRA [43]	$.645_{\pm.003}$	.221	.048	$\mathbf{.569}_{\pm.005}$	**.156**	**.028**	$.581_{\pm.021}$	.166	.108
IHA (Ours)	$\mathbf{.709}_{\pm.008}$	**.254**	**.154**	$.538_{\pm.009}$	.132	.025	$\mathbf{.588}_{\pm.012}$	**.180**	**.036**



(a) Purchase-100 (b) Purchase-100(S)

(c) MNIST-Odd (d) FMNIST

Figure 5.1: ROC curves for low-FPR region for various attacks and datasets.

without requiring the training of reference models or the use of hold-out data. This gain in performance is even more apparent in the low-FPR region, where IHA has high TPR (Figure 5.1).

Moreover, Table 5.1 demonstrates that IHA performs much better than baselines on tabular data (Purchase-100) than image-based data (MNIST-Odd, Fashion MNIST). This may be attributed to multiple reasons,

such as inputs where Assumption 5.2.2 not holding for some data, the absence of reference models (as present in Theorem 5.2.3) that effectively help with difficulty calibration, or inherently lower leakage for these particular datasets. Other subtle sources of approximation errors, such as approximation $\mathbf{H}_*$ using $\mathbf{H}(\boldsymbol{w})$, might also contribute to a gap between performance observed with IHA and the optimal membership inference adversary. We leave further investigation of these factors to future work to better understand the performance discrepancies.

We reiterate that the purpose of our comparisons is not to claim a better membership inference attack for adversarial use; the threat models are not comparable, since our attack requires knowledge of all other records $D \setminus \{\boldsymbol{z}_1\}$ for inferring a given target record $\boldsymbol{z}_1$. Instead, IHA provides a way to empirically audit models for membership leakage without training reference models, which is desirable both in terms of computing and not having to reserve hold-out data for training reference models. More importantly, **our results suggest untapped potential in exploring parameter access for stronger privacy audits** (and the possibility of new inference attacks from an adversarial lens).

5.4 Related Works

5.4.1 Membership Inference

Black-box Membership Inference. Early works on membership inference worked under black-box access, utilizing the model's loss [279] on a given datapoint as a signal for membership. Since then there have been several works focusing on different forms of difficulty calibration—accounting for the inherent "difficulty" of predicting on a record, irrespective of it being present in train data. This calibration has taken several forms; direct score normalization with reference models [264], likelihood tests based on score distributions [43, 353, 365], and additional models for predicting difficulty [27].

White-box Membership Inference. Nasr et al. [229] explored white-box access to devise a meta-classifier-based attack that additionally extracts intermediate model activations and gradients to increase leakage but concluded that layers closer to the model's output are more informative for membership inference and report performance not significantly better than a black-box loss-based attack. Recent work by DeAlcala [63], however, makes the opposite observation, with layers closer to the model's input providing noticeably better performance. Apart from these meta-classifier driven approaches, some works attempt to utilize parameter access much more directly, often utilizing Hessian in one form or another. Cohen and Giryes [59] defined the self-influence of a datapoint $\boldsymbol{z}_i$ as $(\boldsymbol{g}_i{}^\top \mathbf{H}^{-1} \boldsymbol{g}_i)$ as a signal for membership, using LiSSA [5] to approximate the iHVP. This has similarities to our result since our optimal membership inference score also involves computing iHVPs. Li et al. [176] attempted to measure the sharpness for a given model by evaluating fluctuations in

model predictions after adding zero-mean noise to the parameters, a step that is supposed to approximate the trace of the Hessian at the given point.

5.4.2 Privacy Auditing

Ye et al. [354] proposed using efficient methods to "predict" memorization by not having to run computationally expensive membership inference attacks, with reported speedups of up to 140x. They showed how their proposed score (LOOD) correlates well with AUC corresponding to an extremely strong MIA with all-but-one access to records (L-attack [353]). However it is unclear if this computed LOOD is directly comparable across models, making it hard to calibrate these scores to compare the leakage from a model relative to another (an important aspect of internal privacy auditing). Their derivations also involve a connection with the Hessian. Biderman et al. [29] studied the problem of forecasting memorization in a model for specific training data. The authors propose using partially trained versions of the model (or smaller models) as a proxy for their computation. While their results support the need for inexpensive auditing methods, their focus is on predicting memorization early in the training process, while ours relates to auditing fully trained models. More recently, Tan et al. [299] studied the theory behind worst-case membership leakage for the case of linear regression on Gaussian data and derived insights. While this is useful to make an intuitive connection with overfitting, it does not provide a realizable attack or insights for the standard case of models trained with SGD.

5.4.3 SGD Dynamics and iHVPs

SGD Dynamics. Stephan et al. [287] approximated the SGD dynamics as an Ornstein-Uhlenbeck process, while Yokoi and Sato [359] provided a discrete-time weak-order approximation for SGD based on Itô process and finite moment assumption. However, both works rely on strong assumptions about the gradient noises and require a vanishingly small learning rate, largely deviating from the common practice of SGD. To address the limitations of the aforementioned works, Liu et al. [187] directly analyzed the discrete-time dynamics of SGD and derived the analytic form of the asymptotic model fluctuation with respect to the asymptotic gradient noise covariance and the Hessian matrix. Ziyin et al. [378] further generalized the results of [187] by deriving the exact minibatch noise covariance for discrete-time SGD, which is shown to vary across different kinds of local minima. Our work builds on these advanced theoretical results of discrete-time SGD dynamics but aims to enhance the understanding of optimal membership inference, particularly for models trained with SGD.

iHVPs. Currently literature on approximating inverse-Hessian vector products relies on one of two methods: conjugate gradients [160] or LiSSA [5]. Both approximation methods rely on efficient computation of exact Hessian-vector products, and use forward and backward propagation as sub-routines. While these methods

have utility in certain areas, such as influence functions [160] and optimization [237], approximation errors can be non-trivial. For instance, I_1 in the formulation of our attack requires a low approximation error in the norm of an iHVP, while I_2 simultaneously requires a low approximation error in the direction of the iHVP. Recent work on curvature-aware minimization by Oldewage et al. [237] proposes another method for efficient iHVP approximation as a subroutine, but the authors observed high approximation errors based on both norm and direction.

5.5 Conclusion

Our theoretical result proves that model parameter access is indeed necessary for optimal membership inference, contrary to previous results and the common belief that optimal membership inference can be achieved with only black-box model access. We propose the Inverse Hessian Attack inspired by this theory that provides stronger privacy auditing than existing black-box techniques. However, IHA is not yet practically realizable for most settings due to the computational expense of calculating the Hessian, or even approximating iHVPs. Our conclusion aligns well with recent calls in the literature to consider white-box access for rigorous auditing [45]. Exploring the accuracy of iHVP approximation methods to extend IHA to larger models, along with multi-record inference, are both promising directions for future research.

Chapter 6

Memorization in LLMs*

In this final chapter, we explore the memorization capabilities of large language models (LLMs) and their implications for privacy. We first begin with an overview of how LLMs are trained and tuned (§6.1), followed by an overview of various kinds of memorization in LLMs (Sections 6.2.2.1 to 6.2.2.6) and potential mitigation strategies (§6.3). We then proceed to study exact memorization in LLMs via membership inference (§6.4), finding factors related to LLM training as well as the inherent nature of language that make it hard to measure leakage in LLMs.

6.1 Background

Autoregressive models are trained to predict the next text token based on the sequence of previous tokens, as opposed to, e.g., bidirectional masked language models such as BERT [68], which are conditioned on both left and right context. In this chapter, we focus on large autoregressive language models (LLMs) with billions of parameters such as GPT-4 [240] or Llama 2 [309]. Sometimes we discuss work that uses models with fewer parameters or models that are not autoregressive. For these, we use the generic acronym LM. We write $p(\cdot|x)$ for the probability distribution of an autoregressive model given a context x.

6.1.1 Training LLMs

Regardless of the concrete model architecture, current LLM training typically encompasses three main stages: pretraining, supervised fine-tuning, and reinforcement learning from human feedback (RLHF).

*This chapter is largely based on Michael Duan, Anshuman Suri, Niloofar Mireshghallah, Sewon Min, Weijia Shi, Luke Zettlemoyer, Yulia Tsvetkov, Yejin Choi, David Evans, Hannaneh Hajishirzi, *Do Membership Inference Attacks Work on Large Language Models?*, in Conference on Language Modeling (COLM), 2024. and Valentin Hartmann, Anshuman Suri, Vincent Bindschaedler, David Evans, Shruti Tople, Robert West, *SoK: Memorization in General-Purpose Large Language Models*, in arXiv, 2023. Code relevant to this chapter is available as a Python package at https://github.com/iamgroot42/mimir.

Stage 1: Pretraining. The training data in this stage consists of text documents. The LLM is trained to predict the next token of a document given a prefix from the document. The pretraining dataset usually consists of public data such as web pages crawled from the Internet, books, etc., though the data sources are not always disclosed [240].

Stage 2: Supervised fine-tuning. The second stage uses a dataset of prompts and responses. The model is trained in a supervised fashion to give the response corresponding to a prompt in this dataset. The prompts and responses can come from public, task-specific datasets [337] or be written specifically for the model training. In the latter case, both the prompts and the responses can be written by human labelers, or the prompts can come from user requests to a language model [245], which allows for more closely modeling the prompt distribution upon deployment of the model.

Stage 3: Reinforcement learning from human feedback (RLHF). This stage consists of several steps. First, human labelers rate answers generated by the model in response to prompts. The ratings or labels can also be model-generated [309] or come from LLM users [245]. These ratings are then used to train a reward model, which is in turn used for optimizing the LLM via reinforcement learning. Automated tools can be used in addition to human labelers [240].

Memorization of all types covered in this chapter can in principle happen at all training stages; except for the alignment goals, which only influence the training in stages 2 and 3. The vast majority of the work on memorization—and related fields such as privacy—in LLMs focuses only the first training stage though, maybe due to the earlier availability of pretrained models without stage 2 and 3 training.

6.2 Memorization in LLMs

Learning from data requires extracting information from the data and generalizing from this information. The learning abilities required to perform the generalization vary for different tasks. Regurgitating a specific fact seen directly in the training data requires no generalization; mimicking the choice of words and syntactic structures that make up a person's writing style requires some generalization; writing a new creative story requires a lot of generalization. Our focus is on **memorization, which we define roughly as learning that involves only little generalization.** We acknowledge that, as of now, there is no complete understanding of generalization [366] and no clear demarcation between memorization and generalization [84, 85]. We do not attempt to make progress on this demarcation problem here, but rather just use our intuitive notion to scope our work. The definitions that we consider in later sections are all concerned with more specific objects than general memorization.

These are definitions that are not usable in most cases for empirical analyses due to their computational complexity. They are thus rather meant to guide the intuition of the reader throughout the chapter than

to replace existing, often more practical, definitions. In general, definitions come in two flavors: definitions that determine whether an adversary can extract information from the model that they did not know about before (e.g., a fact), and definitions that only determine whether a certain piece of information is present in the model.

In the remainder of this section, we describe general approaches for, and challenges with, identifying and measuring memorization.

Inference attacks.

A successful inference attack implies that the model has stored the information that is unknown to the adversary in some way. This does not necessarily mean that the model has *memorized* these attributes, though. For instance, with attribute inference, it could be that it has merely learned the data distribution well enough to predict the attributes based on other attributes of the record [137]. If, however, the attribute inference attack on a counterfactual model trained on the same data with the record removed is unsuccessful in recovering the attributes, then we have evidence that the attack's success was due to memorization of the target attributes by the original model. When successful and done carefully, this can give a much more precise characterization than membership inference of what the model has memorized about a record. We consider memorization in the sense of both membership and attribute inference, since both types of memorization can have significant implications.

While membership and attribute inference are concerned with individual data points, we consider several pieces of information that may affect multiple documents, such as writing styles or parameters of preprocessing methods. Memorization of such information is better captured by the concept of *distribution inference* (Chapter 3).

6.2.1 Challenges in estimating memorization

It is difficult to precisely determine whether a piece of information was memorized by an LLM. There are several reasons for this.

Memorization vs. discoverable memorization. Anything that a model knows about its training data needs to be stored in some way in the model's weights. A naïve definition of memorization could thus be "any information that is stored in the model's weights is memorized". However, evaluating this definition would be infeasible and basically amount to fully determining everything the model has learned. Researchers thus resort to studying proxies for this memorization that only capture memorized information that can be accessed through known methods. This inherently underestimates memorization, since it assumes there are no better ways to extract information from the model.

For example, many definitions are concerned with what can be inferred from model outputs—a subset of what can be inferred from model weights. This essentially assumes a black-box (API) adversary who has no direct access to the model. In some settings, such an assumption is valid and defenses such as output filtering, that do not prevent memorization in model weights but only the revelation of memorized information at prediction time, may be effective. If output token probabilities are provided, definitions can make use of them (e.g., the probabilities of the possible answers to a multiple choice question). Otherwise one has to resort to the model outputs in character space, which are produced by decoding algorithms. These decoding algorithms can be deterministic (e.g., greedy decoding), or non-deterministic (e.g., top-p sampling [120]). Furthermore, the choice of decoding algorithm can influence model hallucination (see next paragraph) [75, 170] and other behaviors [146]. Carlini et al. [44] find that swapping greedy decoding for beam search slightly increases the discoverability of memorized verbatim text. Another challenge with output-based definitions is their dependency on a prompt (with some exceptions [42]), the choice of which influences the amount of extracted memorized information [142].

Hallucination. LLMs can generate plausible content that cannot be inferred from their training or input data [140], known as hallucination. Causes of hallucination include training data that favors text generation that is not grounded in the data [69, 327], or a training objective that differs from the task objective [326]. Hallucination can also be linked to memorization [258]. Hallucinations can make it seem as if the model had memorized a piece of information, even though it was not present in its training data. This is referred to as *extrinsic hallucination*, as opposed to *intrinsic hallucination* that contradicts the training data or input [140].

Reasoning and generalization. Beyond factual outputs that are not grounded in the training data, there are outputs that are grounded in the training data but not explicitly contained in it. For example, training documents might contain the facts "[A] is a student of [B]" and "[B] is a professor at university [X]", but not the fact "[A] is a student at university [X]". Still, if the LLM is sufficiently powerful to perform deductive reasoning, it will give the correct answer to the question "Where does [A] study?" [123]. Similarly, LLMs often appear to perform inductive reasoning [351] such as guessing the nationality of a person based on their name [254] or generalizing from code seen during training to create novel algorithms [145].

Distinguishing memorization from hallucination and reasoning. A correct model response to a factual question does not reveal whether the model arrived at this response via generalization and reasoning, because of hallucination, or because it has memorized this response from the training data. When looking for memorized information in the model weights instead of in the model behavior [215], cases of reasoning and hallucination relating to input data [123, 140] can be avoided. In many cases of interest, such as personal identifiers, social security numbers or long passages of verbatim text, it is unlikely that a model could hallucinate the target information or gain knowledge of it through reasoning. Counterfactual definitions such as the ones based on membership and distribution inference above allow for precisely separating memorization

from hallucination and reasoning. A non-zero advantage of an adversary means that the adversary exploits memorization in the model. However, in the case of LLMs these definitions are typically not useful for experimental analyses due to the need of training multiple models.

6.2.2 Types of Memorization

We cover different kinds of memorization in the context of language models: verbatim memorization (§6.2.2.1), facts (§6.2.2.2), ideas (§6.2.2.3), writing style (§6.2.2.4), distributional properties (§6.2.2.5), and alignment goals (§6.2.2.6). In §6.3, we discuss potential mitigation strategies that are not specific to any type of memorization.

6.2.2.1 Verbatim text

Memorizing verbatim text is the most direct and low-level form of memorization. It is also quite prevalent—Carlini et al. [44] demonstrate how GPT-J [31, 322] memorized at least 1% of its training data according to their extractability definition, which is similar to attribute inference.

There are different ways to look at verbatim memorization. Researchers have considered the memorization of entire training documents [116], parts of training documents [44] and, in the context of privacy risks from personally identifiable information (PII), the memorization of short sequences that comprise personal information such as names or email addresses [193]. The concept of verbatim memorization can be broadened by also considering paraphrases, such as those resulting from the replacement of words with synonyms [168]. Note that the verbatim memorization of a text sequence that describes a fact or algorithm implies the memorization of this fact or algorithm, though in a low-level representation. We restrict this section mostly to aspects that are unique to verbatim memorization, and discuss memorizing facts and algorithms in later sections.

Definitions. Since verbatim memorization is related to the tasks of membership and attribute inference (§2.3), some definitions of inference attacks could also be applied to verbatim memorization. However, several formal definitions of specifically verbatim memorization in LLMs have been proposed, on which we focus here.

Exposure metric [41]. Carlini et al. explore memorization of out-of-distribution secrets by LMs. They consider strings of the form "The random number is r", where r is a number randomly sampled from some space $\mathcal{R}$. The authors sample one particular $r' \in \mathcal{R}$ and include the corresponding string in the training set of the LM. They define the *exposure metric*, which essentially measures how many guesses an adversary that tries to guess r' saves by computing the perplexity of the string "The random number is r" for all $r \in \mathcal{R}$ and guessing r in order of increasing perplexity, over guessing in a random order from $\mathcal{R}$. This metric requires many queries to the model to compute, in addition to a retraining of the model. Note that this approach

avoids the difficulty of determining what should and should not be learned by a model, since the artificial random strings are introduced as an explicit way to insert content that should not be learned.

While the exposure metric measures the general capability of a model for memorization, the following definition aims to measure the memorization of a specific document by a specific model with fixed weights.

Extractability [42]. This definition by Carlini et al. defines a string y to be *extractable* from an LM p if there exists a prefix x such that: $y \leftarrow \arg\max_{y':|y'|=N} p(y'|x)$. Instead of the intractable computation of the arg max, the authors use a decoding algorithm such as greedy decoding in practice. They also point out that due to pathological cases any string could be extractable, e.g., when prompting the model to repeat a given input string. However, they instantiate the definition only for cases where x is the start-of-sequence token or a prefix from a document. The authors then specialize this definition to *k-eidetic memorization*, which only allows for strings that are repeated at most k times in the training data.

The next definition measures the degree to which a given model architecture—but not a specific model—memorizes a specific document.

Counterfactual memorization. Following the observation that strings that occur more often in the training data are more likely to be reproduced by the model, Zhang et al. [367] aim to disentangle the plurality of a document in the training data and the degree to which its verbatim text is memorized To this end, they define counterfactual memorization for a document d as the expected difference in token prediction accuracy when predicting d with models trained on datasets containing d and datasets not containing d. This definition is a variation on a definition by Feldman and Zhang for label memorization [85], and is closely related to the concept of algorithmic stability [35]. Counterfactual memorization requires training multiple models, and also requires access to the training data.

Implications. As discussed in §6.1.1, a model trainer may rely on user-submitted prompts and responses in training stages 2 and 3. These prompts can contain highly sensitive information, e.g., when a user asks for advice on medical or relationship issues. In such cases, the prompts may contain detailed information about the user, allowing a third party to identify them in case the model regurgitates the prompt.

It is not entirely clear yet whether verbatim memorization of copyrighted documents from training data in just the model weights itself constitutes a copyright infringement. At the same time, verbatim regurgitation may not always be necessary to constitute infringement. Lee et al. [168] investigate the related problem of plagiarism. They measure the degree of verbatim (copying passages character by character), paraphrase (paraphrasing sentences from a source document), and idea plagiarism (copying core ideas) performed by GPT-2 when prompting the model just with an end-of-sequence token. The authors show that all three types of plagiarism, from both pretraining and fine-tuning data, occur both in pretrained and fine-tuned models

both, but not for all fine-tuning datasets. Fine-tuning also seems to reliably eliminate verbatim plagiarism from pretraining data. Henderson et al. [116] argue that preventing copyright violations can only be done on a higher semantic level that goes beyond verbatim text matching.

Detecting memorized verbatim text can enable identification of certain datasets used for the training of an LLM, although likely requiring access to the specific candidate dataset [44]. For example, the benchmark BIG-bench [284] includes a randomly generated string that acts as a globally unique identifier (GUID), and a test that checks whether the model assigns an anomalously low/high probability to the GUID, which would indicate a contamination of the training data with data from the benchmark. Such canaries could be inserted by model trainers to detect theft or unlicensed use of their models, potentially aiding techniques like dataset inference [200] that might otherwise not work well with LLMs [297]. Memorized documents may also be used to determine a lower bound on the knowledge cutoff date of a model (how recent the newest training data is), as recently demonstrated for Github's Copilot [64].

Detecting memorized verbatim text. There have been several attempts at detecting verbatim memorization that build upon membership inference tests (as we discuss in §6.4). However, it is important to keep in mind that successful membership inference does not necessarily imply verbatim memorization (see §6.2), or the converse.

While the cost to train LLMs makes it infeasible to utilize techniques that require training shadow models (newly trained instances of the model with different random seeds for the training algorithm or the data sampling), as is done in most membership inference attacks, the open-endedness of prompts opens new avenues unavailable for other domains such as vision and tabular data. For instance, simply prompting the model with the title and author name of a training document [116] is sometimes effective. In the same work, Henderson et al. [116] sample random snippets from books and show how some LLMs, when prompted with these snippets, return long sequences from the books. They show that instructions like *replace every 'a' with a '4' and 'o' with a '0'* can circumvent content filters. Yu et al. [360] use prompts with function signatures and code comments to extract program code from LLMs.

Prompting techniques not explicitly designed for detecting memorization could be repurposed, such as adding prefixes to encourage grounding in the training data [338]. Instruction-tuned models are much more amenable to this type of prompting, indicating that instruction tuning can make it easier to access information memorized in model weights. Some attacks rely on prompting the model with document prefixes from the training data [44] or from Internet text [42]. Ozdayi et al. [246] use prompt tuning on models by utilizing white-box access and knowledge of some training records. The attack generates a prefix that can be prepended to a prompt to maximize the likelihood of generating a suffix corresponding to training data. All of these prompting-based methods are attacks in the framework of the extractability definition. It is natural that most methods are developed in this framework, since in practice people are usually given one single model and are

usually most interested in the memorization of real documents. Outside of these prompting attacks, there have been some recent attempts at attributing memorization of examples to specific neurons in models [201].

Preventing memorization or extraction of verbatim text.

One strategy for preventing memorization of verbatim text is to avoid repetitions of verbatim text in the training data, motivated by the observation that the likelihood of a sequence is memorized increases with the number of times that sequence occurs in the training data. Carlini et al. [44] extract memorized verbatim text from models of the GPT-Neo [31, 322] family. They sample text sequences from the training data, prompt the model with a prefix of each sequence and check whether the model generates the corresponding suffix. They find that the amount of memorized text increases with model size, repetition of the sequence in training data, and the length of the prefix prompt. Increased memorization from repeated sequences has been observed before, and consequently de-duplication of the training data has been proposed as a countermeasure [149, 169]. However, this might run counter to the effective upsampling (via training for more epochs) of trustworthy sources commonly performed in the training of LLMs [97, 308].

Another approach that operates at the level of individual documents is differential privacy (see §6.3.1), wherein noise it added to the gradients of training documents, trading reduced memorization against model performance [181].

Mantri and Sasikumatm [204] propose several potential pathways towards LLMs that do not regurgitate memorized copyrighted content: pruning or zeroing out parameters associated with such content; fine-tuning the model with non-copyrighted content; and the use of loss functions that discourage the model from generating text too similar to copyrighted training data. A more radical change to prevent verbatim memorization would be to use a substantially different training objective: instead of learning to predict individual tokens from training documents, the model could learn to predict the content of those documents at a higher semantic level. This idea has been implemented for computer vision models, where pixels in the training objective are replaced by latent representations of image patches [15].

Verbatim memorization can also be addressed in black-box settings by using post-processing to block text sequences in the training data from occurring in the generated output. Ippolito et al. [129] propose using a Bloom filter to detect n-grams of the model output that appear in training data, and re-generate tokens until there is no more match in the training data. However, the filter cannot account for small differences, such as changed whitespaces, and can be actively avoided by style-transfer prompts, e.g., making the model respond in all lowercase.

6.2.2.2 Facts

LLMs achieve good results on knowledge benchmarks even in closed book settings [308], implying sufficient memorization of facts about the world. These can be facts about the real world like "birds can fly" or facts about fictional worlds like "Harry Potter studies at Hogwarts" [49]. Li et al. [182] provide empirical evidence for memorization of the co-occurrence of words in different topical contexts in both embeddings and self-attention layers.

Definitions.

Tuple completion. Meng et al. [215] represent facts as tuples $t = (s, r, o)$ of a subject s, a relationship r and an object o. They define memorization of a fact (s, r, o) as the model completing the prompt 's r' with 'o'. The Knowledge Assessment Risk Ratio (KaRR) [70] considers the ratios between the probability of the correct object being generated by the model when given s and r, and when given only s or only r. This is aimed at removing the influence of the prior probability of the model for generating o.

Personally Identifiable Information. A particular class of facts that provide information about identifiable individuals is known as *personally identifiable information* (PII). Kim et al. [156] draw a distinction between structured PII and unstructured PII. Information in structured PII follows a fixed pattern, such as email addresses and phone numbers. Unstructured PII can be expressed in different ways. e.g., "[PERSON 1] is the parent of [PERSON 2]" contains the same information as "[PERSON 2] is [PERSON 1]'s child".

PII extractability. Lukas et al. [193] define three variants of PII leakage across different threat models, which can be used to define PII memorization, but also the memorization of more general facts. The first definition is extractability, which is the probability of a piece of PII being contained in an output produced by an unprompted model.

PII reconstruction and inference. The second and third definitions of Lukas et al. measure the model's ability to associate PII with a context. A sentence containing at least one piece of PII is chosen from the training data and the PII is replaced by a [MASK] token, for example "The police arrested [MASK] near the White House on 8/20." In PII reconstruction, the (approximately) most likely PII replacement for [MASK] under the model likelihood is compared with the PII in the original sentence. PII inference differs from PII reconstruction in that the model only has to choose from a predefined set of candidates.

All of the above definitions cover cases of previously unknown facts or PII that can be extracted from the model. The following definition assumes a set of candidate PII whose relationship to each other the model might leak.

Linkability of PII. Kim et al. [156] argue that a random disclosure of some personal information without it being linked to an individual does not necessarily pose a privacy risk. For example, it might not be problematic if an LLM leaks an address without the name of the resident. They propose the definition of linkable PII leakage, which implies memorization of the connection between different pieces of personal information. The definition roughly states that if the likelihood of a piece of personal information a_1 under the model increases when conditioning the model on other personal information $a_2, \ldots, a_n$ linked to the same data subject, the model links a_1 to $a_2, \ldots, a_n$. This definition is also applicable to other facts, e.g., linking a football player to her teammates.

Finally, a model that fulfills the last definition only allows someone with prior knowledge of the corresponding fact to determine whether or not the model has memorized that fact.

Counterfactual memorization. The definition of counterfactual memorization [367] (see §6.2.2.1) might also apply to facts: Instead of a specific document, one would remove all occurrences of a specific fact from the training data, and measure how this influences the knowledge of the model about the fact. This could help with determining whether a model knows a given fact because of memorization (in that case it would not know the fact anymore after the removal) or because of reasoning (in that case the model would still know the fact).

Implications. If only publicly-accessible data is used for the training of LLMs, they cannot memorize facts that are not already publicly disclosed. However, as argued by Brown et al. [36], LLMs have the potential to decontextualize information, that is, bring up the information in contexts which it was not intended for or without essential surrounding context. LLMs can reveal PII and other information that is not meant to be public but found its way online—e.g., through users who shared sensitive information in online forums via accounts that can be traced back to them. PII could also make its way into the model's memory if user-submitted queries are used in stage 2 or 3 of the training. For example, a user might ask the model to draft a reply to a letter, where the letter might contain the user's name and address.

Detecting memorized facts. Commonly used benchmarks for assessing factual knowledge of LLMs consist of questions about facts in natural language [144, 166]. Some are in the form of multiple-choice questions [117], which are mostly suited to determining aggregate knowledge of model. Another variant is *cloze* tasks, where the model is asked to fill in masked-out entities [49, 238, 253], and which are very similar to tuple completion tasks. The exact prompt formulation can influence the success of knowledge extraction for cloze tasks [142] and questions [284].

Jiang et al. [142] propose ensemble methods that leverage multiple prompts, generated through mining and paraphrasing content from Wikipedia. Li et al. [175] demonstrate how jailbreaking models can be exploited to increase PII leakage, as opposed to standard querying. Dong et al. [70] treat subject, relationship and

object in the tuple completion task (see §6.2.2.2) as latent variables, and consider different textual aliases to generate different strings pertaining to the same s, r, o tuple. Jain et al. [134] present the model with a fact and its negation, and compare the perplexity of the model on those two strings. Methods of these kinds can be used to practically extract previously unknown facts from a model. However, most of the methods are not aimed at distinguishing between memorization and generalization; for benchmarks, it is usually only relevant whether a model correctly produces a fact, not *how* it produces this fact. This distinction might become clearer with methods that identify knowledge in the model's weights. Meng et al. [215] make a step in this direction by using causal interventions to identify components of LLMs that store facts. Specifically, they consider s, r, o tuples, where the model has to predict the object from the subject and relationship. Based on their findings, the authors posit that the MLP modules in the transformer act as two-layer key–value stores. Lehman et al. [171] study medical PII leakage of name–condition pairs via the release of embedding weights. Patil et al. [251] demonstrate, via parameter-analysis and prompting-based methods, how PII leakage persists even after information "deletion" (based on parameter editing).

Preventing memorization of facts.

A technique called scrubbing acts already on the level of training documents—identifying pieces of text, e.g., via named entity recognition [7, 121], and removing them or masking them with either a generic [MASK] token or more specific tokens such as [NAME]. Scrubbing has been used for removing PII [193], but could also be used for other types of facts (e.g., "The capital of Germany is [CITY].").

Shi et al. [277] propose a fine-grained variant of differential privacy termed S-DP that works on the token level. Instead of protecting entire training documents with DP, it only protects tokens that have been identified as sensitive. Their proposed method selectively adds privacy noise to gradients to which the protected tokens have contributed.

Removing memorized facts might be particularly important in some jurisdictions like the EU [79] and California [236] that codify a right to be forgotten for individuals whose personal data has been used by a business. Meng et al. [215, 216] build on their hypothesis about key–values stores and propose a method to selectively change facts by modifying values in those stores.

A general difficulty with preventing memorization of unstructured PII is that it requires a model with a deeper understanding of relationships within the training documents than for structured PII. Eldan and Russinovich [77] demonstrate a technique for unlearning data from a particular source. The technique is based on a reference model fine-tuned on the source data, along with generic text auto-generated with GPT-4. The KGA framework [331] uses a process of unlearning to make the model's performance on target data similar to unseen data, while maintaining overall performance. This approach has been successful with more abstract data sources, such as specific personas in chat-based datasets. Bayazit et al. [24] develop a method to

identify "knowledge-critical" subnetworks within GPT-2 models, using a joint loss function and demonstrate its effectiveness with concepts like 'fruit' and 'swimming'.

6.2.2.3 Ideas and algorithms

Ideas are similar to facts in that there are multiple ways to express them in language. Ideas can be about the physical world such as reinforcing concrete with steel, as well as about fiction, such as a story around a boy whose parents got killed when he was small and who learns wizardry at a secret school. Algorithms are a special type of idea that can be memorized in two ways: (1) as a description of a series of steps, similar to how the idea for the plot above can be memorized as a sequence of events; and (2) as a behavior, i.e., the LLM implements the algorithm and executes it in its forward pass. The former may occur via algorithm implementations in training data, the latter via input–output examples [227]. We have not found any research on whether this also works the other way around. In this section, we only cover simple algorithms such as arithmetic operations that do not require a high level of generalization. The distinction between facts and ideas can sometimes be difficult. 'Harry Potter is a wizard [...]' is a fact of literature, and likewise the steps of photosynthesis a biological fact.

Implications. As with facts, one often wants the model to memorize ideas such as solutions to common problems or common tropes in fiction [361]. Similarly, one might want the model to be able to perform algorithms such as arithmetic manipulations [271], or return an implementation of dynamic programming [38, 145]. On the other hand, it is often undesirable if the model learns and reproduces harmful ideas. Meta considers three risk categories for its Llama 2 model [309]: illicit and criminal activities; hateful and harmful activities; and unqualified advice. Examples for ideas from all of these categories are very likely to be found in a dataset scraped from the Internet: plans for robberies in fictional stories; ideas around the superiority of one race in online comments; or medical speculation by non-professionals in health forums.

Detecting memorized ideas and algorithms. Techniques for detecting memorized facts (§6.2.2.2) might also apply to ideas. The BIG-bench benchmark [284] measures the capability of LLMs to perform common algorithms such as removing duplicates from a list of numbers or finding the longest common subsequence of two strings, by testing correctness of model responses on instances of these problems. Saxton et al. [271] synthetically generate mathematics problems from fields such as algebra and arithmetic. They train models on smaller instances (e.g., smaller numbers) and test them on larger instances to determine whether the models merely learn algorithms for the domain of the training examples, or learn the algorithms in their full generality. McCoy et al. [209] compare the performance of LLMs on the same task, but with different parameters that do not change the task difficulty.

Stolfo et al. [291] perform causal mediation analysis on LLMs [252] to get insights into how LLMs perform simple arithmetic operations. They change activation values to identify layers and neurons most responsible

for those operations. In a similar manner, Wang et al. [329] identify the circuit in GPT-2 responsible for detecting the grammatical object in a certain class of sentences. Nanda et al. [227] train a transformer to perform modular addition. Via careful inspection, they identify the exact algorithm that the model uses to perform this task. Note that the extraction of the algorithm in this case consists of identifying a particular implementation in model weights. This is as opposed to extracting an abstract description of the algorithm in natural language or pseudocode. Interestingly, they find that in the beginning of training the model memorizes the training examples; in a second training phase it learns to perform the general algorithm; and in a third phase it removes the memorized components. Extracting training data in this example would thus only be possible in the first and second phase, whereas extracting the algorithm would only be possible in the third phase.

Preventing memorization or extraction of ideas and algorithms. Supervised fine-tuning, RLHF, and safety context distillation are often used [309] to deter a model from generating ideas from certain categories like criminal activities. The latter is a form of fine-tuning aimed to make the model behave as if prompts were preceded by a prompt instructing the model to, e.g., only generate safe responses. While not explicitly designed to prevent the model from outputting *memorized* ideas from these categories, this can be a side effect.

6.2.2.4 Writing styles

LLMs like GPT-4 can write in rhymes or imitate the writing style of Shakespeare [38]. Microsoft recently announced the "Sound like me" feature for their Copilot, allowing the LLM to write in the user's style when drafting emails [283]. With writing style, we mean the linguistic definition of style [143]—everything about a text that goes beyond pure semantics, including the choice of words and sentence structures, the characteristic use of stylistic devices such as alliterations and metaphors, and the level of formality. We consider not only writing in natural languages, but also programming languages. There, with different styles we mean functionally equivalent pieces of code that differ in stylistic features like the naming of variables, their capitalization, the use of software design patterns, and formatting differences like the use of tabs or spaces for indentation.

Definitions. *Mixture distribution.* Several authors [11, 228] suggest that LMs learn to separate agents (e.g., separated by different beliefs) in their training data. This concept is formalized by Wolf et al. [342], who describe a language model as a mixture over different probability distributions. We can instantiate this formalism for writing styles, where we describe the writing style of the model as a mixture of the writing styles of authors seen during training, one writing style per author (or, alternatively, one writing style per demographic group, e.g., age groups or speakers of a dialect [314, Sec. 4.1]). The probability $p(y)$ that the model assigns to a given string y can then be decomposed as $p(y) = \sum_{\phi \in \Phi} w_\phi p_\phi(y)$, where each p_ϕ

corresponds to the writing style of one author in the training data and w_ϕ is the prevalence of that style in the model outputs. We can define the memorization of one author's writing style as the presence of this writing style in the mixture, i.e., that there is one p_ϕ that corresponds to this author's writing style. Wolf et al. show that the model can be made to behave according to any p_ϕ in the mixture by a suitably long prompt, given some technical conditions. In practice, a prompt such as 'Answer in the style of [NAME]' might suffice to isolate p_ϕ.

Authorship verification and attribution. To determine whether a model can imitate an author's writing style sufficiently well, an *authorship verification* (AV) or *authorship attribution* (AA) task could be used [319]. In AV, an adversary is given two texts and has to determine whether they are written by the same author. In AA, an adversary is given texts from different authors and has to determine for a separate text by which of those authors it has been written. The advantage of an adversary in AV or AA over random guessing could be used a measure of the memorization of a writing style.

The practicality of the above definitions depends to a large degree on whether one can efficiently find prompts that invoke a given writing style or make the model perform AV and AA tasks, where the latter might become more complicated if the model is safety-tuned.

Implications. An LLM that has memorized the writing styles of individuals could be used for authorship attribution [222, 319]. One might, for example, try to determine the author of an anonymous text, if documents written by that author under their name were part of the training data. Authorship attribution via pretrained transformer models has been explored before [21, 82, 319], although with smaller models like BERT that require fine-tuning on target authors' documents. LLMs that have seen documents from multiple authors already during pretraining might be usable off-the-shelf for authorship attribution, making this technique much more accessible.

Many LLMs will have seen instances of documents written by an author together with additional information about that author (columns by a journalist that contain biographical information about authors, etc.). They might hence be able to perform author profiling [28, 314], i.e., given a document, determine attributes of the author such as age or gender. This has recently been demonstrated using Reddit comments [285].

Detecting memorized writing styles. If a model can successfully apply the style of an author to its output, this is evidence for the memorization of the author's style. This could be measured by existing authorship verification or attribution methods [319]. For more generic styles like 'formal' or 'poetic', researchers use zero-shot prompting [192], augmented zero-shot prompting (giving the model examples of other styles than the target one) or few-shot prompting [259]—concrete prompts that can be seen as invoking mixture components as described above.

Preventing memorization or extraction of writing styles. If a writing style's frequency affects memorization (as it is the case for Wikipedia entities [203]), one solution is to limit the amount of text per author in the training data. Solaiman and Dennison [280] demonstrate that fine-tuning on non-toxic human-written prompts can reduce toxicity, arguably a form of writing style. Likewise, toxicity can be reduced by RLHF [245, 325] or incorporating human feedback in the pretraining objective [162]. Ilharco et al. [126] fine-tune models specifically for undesirable behavior and subtract the weight difference from the original model to remove such behavior. Li et al. [177] propose disconnecting model components to minimize loss on training data and maximize loss on unwanted behavior data. Both techniques could be applied to selectively remove writing styles. Li et al.'s technique could be applied to eliminate an individual author's writing style by dividing their documents into useful content and unimportant content, the latter serving as data for unwanted behavior. Mireshghallah and Berg-Kirkpatrick [218] propose a VAE-based method for obfuscating the writing style of a text document by turning it into a generic style. While the authors design this method to prevent discrimination and bias, it might be applied to the training corpus of an LLM to prevent the memorization of specific writing styles.

6.2.2.5 Distributional properties of the training data

Distribution inference (Chapter 3) is concerned with the leakage of (sensitive) properties of a model's training distribution. For LLMs, such properties of interest could be the proportion of documents authored by people of a given gender, the percentage of hateful content, the type of preprocessing used or data sources used in training.

Implications. Distributional membership inference could be used to identify individuals who contributed data in stages 2 and 3 of the model training [109]. Distribution inference attacks for author inference have already been demonstrated for text classification models [214], and for fine-tuning data for LLMs [150].

When describing the training data distribution as a mixture over different data sources, one distributional property is whether a particular source—such as a specific website—is part of that mixture. This is described by distributional membership inference [109]. Inferring that information from the model would allow an author to determine whether the model was potentially trained on their copyrighted documents.

The utility of distribution inference attacks, apart from inferring sensitive properties, also lies in auditing models without access to actual training data, which may not be available to the auditor. For instance, such attacks can be used to determine whether the training data is biased in any way, which is important, since tasks like hiring decisions and news generation require unbiased models. There has also been a demonstration of utilizing distribution inference (along with cryptographic primitives) for external auditing [72] on classification models, focusing on attestation of gender and race-related properties.

Detecting memorized distributional properties. Most techniques for distribution inference rely on some form of shadow model training (§3.3), and the only settings in which such attacks demonstrate non-trivial leakage require some form of poisoning (§4.3). Additionally, all of these attacks make strong assumptions about the adversary's prior knowledge of the underlying training distribution, which is a general limitation of the framework. Distribution inference may be a realistic approach for learning about the data preprocessing, though. In a setting where the training data is known (e.g., a public dataset such as The Pile [97]), but not the preprocessing, an adversary can preprocess the same documents in different ways and track the loss of the target model on all variants Intuitively, the loss should be lowest with the preprocessing used by the model trainer, even if the documents themselves were not part of the training data.

6.2.2.6 Alignment goals

To instill instruction-following, helpful, truthful and harmless behavior into the model, data generated by human labelers is used in training stages 2 and 3. Labelers write prompts and responses, and rate model outputs along various axes [245, 309], following guidelines provided by the model trainer. The effectiveness of the alignment training (see, e.g., [309]), is proof that at least high-level goals from the guidelines such as harmlessness and helpfulness are memorized by the model. Despite the guidelines, there is still disagreement between human labelers [245, 309]. Memorization from alignment training might thus not be limited to those guidelines—models might also memorize political, ethical and other opinions of labelers.

Implications. Preference ratings can reveal sensitive information about the labelers. For instance, truthfulness ratings of model outputs like "Taiwan is a sovereign country" might reveal political opinions. Similarly, the stance taken by a labeler in their response to the instruction "Write an essay about whether US Americans should have the right to bear arms" used for supervised fine-tuning. The sometimes small number of labelers (e.g., 40 for InstructGPT [245]) could make them easier targets for privacy attacks, since their individual contributions have a larger impact.

Human labelers are likely so go through a formal process where they hand over the rights for their data to their employer. Copyright for this data, thus, may not be an issue. For instance, the participation agreement of Amazon Mechanical Turk [9] states that "all ownership rights, including all intellectual property rights, will vest with that Requester".

Detecting memorized alignment goals. Alignment training happens in the later training stages, making it more prone to detection and extraction attacks than pretraining data [86, 131]. Since the training mode for supervised fine-tuning is similar to that for pretraining, attacks aimed at extracting verbatim text memorized during pretraining (§6.2.2.1) might also work for the human-written responses to prompts used in stage 2. Different methods might be necessary for the prompts themselves, since the model is usually not directly trained to reproduce the prompts.

Regarding stage 3 training, it can be possible to reconstruct the reward function used for RLHF up to an additive, prompt-dependent constant when given access to the model after the stage 2 training, by using Eq. 5 of Rafailov et al. [256]. If the model after stage 2 is not publicly available, but based on a publicly available pretrained model—this is, e.g., the case for Llama 2 [309] and Mistral 7B [141]—one might try to approximate it through supervised fine-tuning by using one's own or public fine-tuning data such as the Flan Collection [191], used for LLaMA and Llama 2. Access to the reward function could be used to make inferences about labeler guidelines, for example, by checking whether harmful but helpful outputs are systematically higher ranked than harmless but non-helpful outputs. Individual labelers' data could be attacked using membership inference [353] or attribute inference [137] attacks, which are often model-agnostic. As opposed to the training of the LLM itself, the training of the reward function typically uses much smaller amounts of data, which might make the reward function more amenable to computationally expensive attacks, e.g., those based on shadow models (see §6.2.2.1). Reuter and Schulze [260] train a classifier to predict whether ChatGPT will refuse to answer a given prompt—behavior that is most likely predominantly learned in training stages 2 and 3.

Preventing memorization or extraction of alignment goals. The small number of human labelers might facilitate memorization of individual labelers' data, so increasing their number might reduce this risk. Not releasing the model after stage 1 or 2 will make it harder for an adversary to determine which stage model behavior originates from—this does not prevent memorization (since the model is not changed in any way), but might help against specific attacks.

6.3 General memorization mitigation strategies

This section briefly discusses general-purpose strategies related to preventing memorization that are not specific to any particular type of memorization. Differential privacy (DP) (§6.3.1) and near access-freeness (NAF) (§6.3.2) are two frameworks originally designed to solve privacy and copyright problems, respectively, which aim at preventing certain forms of memorization (DP) or reproduction of memorized information (NAF). They have several shortcomings for preventing different types of memorization in LLMs though, as we discuss next. Elkin-Koren et al. [78] argue in a very similar way why the frameworks are not suitable for preventing copyright violations. §6.3.3 discusses strategies that aim to mitigate copyright infringement with techniques that are external to the model.

6.3.1 Differential privacy

Differential privacy (DP) [74] is a privacy definition that can be satisfied by mechanisms where noise randomly sampled from appropriate distributions is incorporate in the training process to bound the impact of any individual record. Because of the random noise, the trained parameters will be nearly indistinguishable

whether or not any one record was part of the training data. This makes it impossible for the model to remember any one training record. DP thus prevents the memorization of verbatim text, at least as long as this text is only contained in one training document.

For other types of memorization, DP mechanisms can, however, be both under-exhaustive and over-exhaustive. Under-exhaustive because facts or writing styles may be present in many different training documents, books can be contained indirectly in the training data through quotes, reviews, etc. [49]; and over-exhaustive because even if one only wants to prevent the model from memorizing specific pieces of information such as PII or facts, DP adds noise to all information contained in a training document.

While under-exhaustiveness could be addressed by adjusting the unit of privacy so DP mechanisms will provide privacy with respect to multiple records [73], this would still require identifying the number of documents that contain a particular item (which might be expressed in different ways), and might significantly worsen the DP-induced performance drop due to larger amounts of added noise. Even for DP at the level of individual documents, El-Mhamdi et al. [76] argue that high-dimensional differentially private learning on heterogeneous data such as Internet text is impossible with high accuracy due to mean estimation being impossible under these conditions. The method by Shi et al. [277] (§6.2.2.2) that applies DP more selectively at a token level might help with the problem of over-exhaustiveness, but requires exactly identifying the relevant tokens (e.g., those that contain a fact), but is not easily applicable to memorization which is not concentrated in a few tokens of a document (e.g., a writing style).

6.3.2 Near access-freeness

Vyas et al. [321] propose the notion of near access-free (NAF) generative models, aimed at preventing copyright violations. Given an subset $\mathcal{C}$ of its training documents, a model p is NAF with respect to $\mathcal{C}$ if for every $C \in \mathcal{C}$ the difference between the model's output distribution and the output distribution of a specific model that was not trained on C is bounded by a fixed constant. If C only occurs once in the training data, this ensures that p is unlikely to output substantial parts of C, unless those parts could have also been produced without access to C—which would not constitute a copyright violation. Vyas et al. also provide an extension for multiple occurrences.

Beyond verbatim text, NAF with the right distance metric and sufficiently small bound on the distance could also prevent the model from outputting PII such as social security numbers, or facts, as long as one knows how often they are contained in the training data. However, NAF has the same problems of under- and over-exhaustiveness as DP. It is challenging to determine in how many documents a piece of information is contained, and prone to removing benign information that is unique to one or a few documents but that it is desirable for the model to learn. For the latter, consider the case where C is a book. Then a model without access to C would have a very low probability of correctly answering questions about the plot of C,

so an NAF model would not be able to answer those questions either. Note that, unlike DP which is typically applied to the training process, NAF restricts the output distribution of the model. It can therefore be used to give guarantees for memorized information in the model outputs, but not for the memorization itself, i.e., someone with access to the model weights might still be able to detect memorized information.

6.3.3 Strategies for mitigating copyright violations via infrastructure

In some cases, copyright violations may be prevented by preventing particular types of memorization, as discussed in the previous sections. There have also been some proposals for mitigating copyright violations through appropriate infrastructure.

The most straightforward way to avoid copyright violations is by training only on data with permissive licenses [93, 159]. However, excluding data with non-permissive or unspecified licenses could significantly reduce the amount of available data [217] and might exclude high-quality data sources such as textbooks.

Min et al. [217] propose training a LLM only on permissively licensed documents, and augmenting it with a datastore of copyrighted documents. This datastore can then be used at prediction time to improve LLM performance on domains not covered by the training data. This setup allows for precisely pinpointing which copyrighted documents contributed to a particular model output, and for easily removing copyrighted documents if required. Determining copyrighted documents that were used in producing a model output via this and other methods [160] could allow for using documents with licenses that require author attribution [116]. Ippolito and Yu [128] describe a protocol similar to `robots.txt` wherein website owners could signal to model trainers via a file in the root directory which parts of their website are appropriate for model training, a variant of which has already been implemented by OpenAI [239] and Google [46]. In addition to deploying block requests [305], the New York Times has updated its terms of service to explicitly ban the use of its data for training ML models [336].

6.4 Membership Inference for LLMs

In this section, we particularly focus on measuring verbatim memorization in LLMs via membership inference attacks.

Membership inference attacks (MIAs) have great utility for privacy auditing of models [286], as well as investigating memorization of training data, copyright violations and test-set contamination [244, 276]. While MIAs have been found to achieve high attack performance, alluding to high levels of training-data memorization [27, 193, 365], most analyses are limited to classifiers or LM fine-tuning [94, 220]. The performance of existing MIAs on LLMs and their pre-training data is largely unexplored. In this section, we set out to explore the challenges in evaluating membership inference attacks on LLMs, across an array of five commonly-used

membership inference attacks. We introduce MIMIR, a unified repository for evaluating MIAs for LMs, with implementations of several attacks from literature. We report on experiments extensively evaluating these MIAs against target models from the Pythia suite [30] over the Pile [97] (§6.4.2). For the most part, we find that the performance across most MIAs and target domains is *near-random* (§6.4.3).

Our further analysis suggests that the inherent characteristics of LLMs at scale—specifically, the use of massive training data and near-one epoch training (§6.4.4.1)—considerably decrease current MIA performance. This suggests that the success of current MIAs in previous settings does not transfer well to attacking pretrained LLMs seemingly due to a lack of memorization of member data. We also find that the frequent overlap between members and non-members from natural language domains considerably decreases MIA performance and raises the question of how membership should be interpreted (§6.4.4.2). Notably, in several domains, non-members have high n-gram overlap with members, e.g., non-members from the Pile Wikipedia and ArXiv test samples have average 7-gram overlaps of over 30%. Notably, non-members with lower n-gram overlap are more distinguishable by existing MIAs. We also suggest that high MIA performance reported by prior work [276] is likely because non-members are chosen from the same domain as members but are temporally shifted, and these seemingly in-domain non-members likely belong to a different distribution as a result of n-gram overlap shift (§6.5).

Finally, building off membership ambiguity due to n-gram overlap, we discuss how the precise definition of members in standard MI may not capture important information leakage under generative text-modeling. We generate modified members preserving lexical and/or semantic similarity by altering a tiny fraction of tokens and show that existing MIAs classify them as non-members with a high degree of confidence, often more definitively than actual non-members (§6.6). We encourage future work to study MI using membership definitions accounting for such fuzzy members to better understand privacy leakage.

6.4.1 Setup

The goal of an MIA is to infer whether a given data point x was part of the training dataset $\mathcal{D}$ for model $\mathcal{M}$, by computing a membership score $f(\mathbf{x}; \mathcal{M})$. A threshold derived on this score is then used to determine a target sample's membership.

MIAs are often used as a proxy to determine whether a machine-learning model leaks information related to its training data [61, 278, 279]. It is the de-facto threat model when discussing machine-learning privacy [279], with a large array of attacks [43, 219, 355] and defenses [1, 54, 300]. More involved approaches include training shadow models [279, 353] on non-overlapping data from the target model's underlying data distribution. While attacks like LiRA [43] show promise, they require training multiple copies of shadow models, which is often intractable for LLMs. Other stronger assumptions for MIAs include white-box access to the model (i.e., access to model parameters).

In our setting, $\mathcal{M}$ is an auto-regressive language model that outputs a probability distribution of the next token given a prefix, denoted as $P(x_t|x_1...x_{t-1};\mathcal{M})$. The goal for MIAs is to model $f(\mathbf{x};\mathcal{M})$, which outputs a score for target sample $\mathbf{x} = x_1...x_n$ with n tokens. This score is then thresholded to determine the target sample's membership in the training data of $\mathcal{M}$. We consider five MIAs:

LOSS. [41, 355] considers the model's computed loss over the target sample:

$$f(\mathbf{x};\mathcal{M}) = \mathcal{L}(\mathbf{x};\mathcal{M}).$$

Reference-based. [264, 335] attacks assume access to a reference model $\mathcal{M}_{\text{ref}}$, another LM trained on a disjoint set of training data drawn from a similar distribution. In practice, an assumption of disjoint training data is impractical. Empirically, using an LM that is different from $\mathcal{M}$ has been a reasonable choice and was used in prior work [149, 335]. The attack considers the membership score of the target sample by $\mathcal{M}$ relative to the membership by $\mathcal{M}_{\text{ref}}$ to calibrate the target model's score given a difficulty estimate through the reference model's score, with goals to improve precision and reduce the false negative rate. For our experiments, we use LOSS as the uncalibrated membership score such that, for the reference-based attacks,

$$f(\mathbf{x};\mathcal{M}) = \mathcal{L}(\mathbf{x};\mathcal{M}) - \mathcal{L}(\mathbf{x};\mathcal{M}_{\text{ref}}).$$

This method exactly follows the method from [335] and is also largely similar to the offline Likelihood Ratio attack (LiRA; [43]), although LiRA uses many reference models (often trained shadow models).

Zlib Entropy. [42] functions similarly to reference-based MIA, using the zlib compression size of a sample $\mathbf{x}$ as a local difficulty threshold per sample:

$$f(\mathbf{x};\mathcal{M}) = \frac{\mathcal{L}(\mathbf{x};\mathcal{M})}{\text{zlib}(\mathbf{x})},$$

where $\text{zlib}(\mathbf{x})$ is the length in bytes of the zlib compressed sample.

Neighborhood Attack. [208] assumes access to a masking model, and operates by generating "neighbor" texts $\tilde{\mathbf{x}}$ to a given text sequence $\mathbf{x}$ by using the masking model to replace a percentage of randomly selected token spans while still maximizing the neighbor's likelihood. If the sample's loss is considerably lower than the neighbor's losses, the difference is attributed to the target model overfitting the sample, and the sample is considered a training member. Formally, we have

$$f(\mathbf{x};\mathcal{M}) = \mathcal{L}(\mathbf{x};\mathcal{M}) - \frac{1}{n}\sum_{i=1}^{n}\mathcal{L}(\tilde{\mathbf{x}}_i;\mathcal{M}).$$

We use BERT [68] as our masking model of choice, with a masking percentage of 5%.

Min-k% Prob. [276] is based on the intuition that non-member examples tend to have more tokens assigned lower likelihoods than member examples do. Given sample $\mathbf{x} = x_1, ..., x_n$ and hyperparameter k, let min-$k(\mathbf{x})$ be the set formed by the k% of tokens in $\mathbf{x}$ with minimum likelihood. We then have

$$f(\mathbf{x}; \mathcal{M}) = \frac{1}{|\text{min-}k(\mathbf{x})|} \sum_{x_i \in \text{min-}k(\mathbf{x})} -\log(p(x_i \mid x_1, ..., x_{i-1})).$$

We experiment with multiple different $k \in \{10, 20, 30, 40, 50\}$ as suggested in Shi et al. [276], but settle on $k = 20$ for our experiments.

We compute the performance of each attack based on 1,000 bootstrap samples of the benchmark and report the average AUC ROC and TPR@low%FPR over the bootstraps.

Membership Inference vs. Data Extraction. MIA advantage is frequently used as a measure of information leakage [219, 278, 279] and a proxy for measuring memorization [42, 220], with recent attempts studying user-level leakage for the fine-tuning setting [150]. However, the 'extractability' of training samples has recently become synonymous with memorization and is increasingly used to compare memorization across models [29, 44, 306]. Kandpal et al. [149] investigated the impact of factors such as training data deduplication on extractability in a similar vein to our work on MIA. With extraction, a prefix is used as a prompt to measure the memorization of a sequence by comparing the resulting generation against the suffix. Both MIA and extraction are useful techniques for studying leakage in models, but rely on different assumptions and reveal different types of leakage risks. While MIAs require knowledge of candidates and only reveal directly which of those candidates are included in the training data, extraction requires knowledge of sufficient-length prefixes to perform extraction and additional measures to determine if extracted texts are valid.

6.4.2 Membership Inference on LLMs is Difficult

We perform a large-scale evaluation of five state-of-the-art MIAs (§6.4.1) on a range of LLMs with up to 12B parameters and diverse benchmarks. For the reference-based attack in Table 6.1 and all following experiments, we use StableLM-Base-Alpha-3B-v2 as the reference model (determined empirically in §6.4.3.1).

6.4.2.1 Target models

We primarily target the Pythia model suite, including (1) five models of **Pythia** [30] with 160M, 1.4B, 2.8B, 6.7B, and 12B parameters, trained on the original Pile data [97], and (2) five models of **Pythia-dedup** [30] with the same parameter counts as Pythia but trained on the deduplicated Pile data. We also experiment

	Wikipedia					Github					Pile CC					PubMed Central				
# Params	LOSS	Ref	min-k	zlib	Ne	LOSS	Ref	min-k	zlib	Ne	LOSS	Ref	min-k	zlib	Ne	LOSS	Ref	min-k	zlib	Ne
160M	.504	**.515**	.488	.514	.513	.638	.591	.634	**.656**	.638	.497	.497	**.503**	.498	.496	.500	**.516**	.504	.500	.486
1.4B	.510	**.544**	.506	.518	.518	.656	.587	.654	**.670**	.650	.500	**.525**	.509	.502	.499	.496	**.530**	.505	.500	.490
2.8B	.516	**.565**	.511	.522	.517	.707	.657	.708	**.717**	.698	.501	**.537**	.509	.503	.502	.498	**.536**	.502	.500	.497
6.9B	.514	**.571**	.512	.521	.514	.672	.573	.675	**.684**	.654	.511	**.564**	.516	.512	.505	.504	**.552**	.508	.504	.497
12B	.516	**.579**	.517	.524	.520	.678	.559	.683	**.690**	.660	.516	**.582**	.521	.517	.514	.506	**.559**	.512	.506	.497
	ArXiv					DM Math					HackerNews					The Pile				
# Params	LOSS	Ref	min-k	zlib	Ne	LOSS	Ref	min-k	zlib	Ne	LOSS	Ref	min-k	zlib	Ne	LOSS	Ref	min-k	zlib	Ne
160M	**.507**	.486	.501	.500	**.507**	.490	**.523**	.493	.482	.489	.492	.490	.497	.497	**.505**	.502	**.511**	.506	.505	.499
1.4B	**.513**	.510	.511	.508	.511	.486	**.512**	.497	.481	.465	.503	**.514**	.509	.502	.504	.504	**.521**	.508	.507	.504
2.8B	.517	**.531**	.522	.512	.519	.485	**.504**	.497	.482	.467	.510	**.549**	.518	.507	.513	.507	**.530**	.512	.510	.506
6.9B	.521	**.538**	.524	.516	.519	.485	**.508**	.496	.481	.469	.513	**.546**	.528	.508	.512	.510	**.549**	.516	.512	.510
12B	.527	**.555**	.530	.521	.519	.485	**.512**	.495	.481	.475	.518	**.565**	.533	.512	.515	.513	**.558**	.521	.515	.511

Table 6.1: AUC ROC of MIAs against Pythia-dedup (TPR@low%FPR results in Table 6.2). Highest performance across different MIAs is bolded per domain. **MIA methods perform near random ($< .6$) in most domains.** See §6.4.5.3 for GitHub outlier discussion.

	Wikipedia					Github					Pile CC					Pubmed Central				
# Params	LOSS	Ref	min-k	zlib	Ne	LOSS	Ref	min-k	zlib	Ne	LOSS	Ref	min-k	zlib	Ne	LOSS	Ref	min-k	zlib	Ne
160M	1.1	0.8	1.2	**1.4**	1.3	13.5	4.6	12.3	**14.7**	5.9	0.4	**0.8**	0.5	0.4	0.4	0.7	0.9	**1.0**	0.3	0.1
1.4B	0.6	**0.9**	0.5	0.7	0.4	12.8	0.7	12.9	**16.4**	3.9	0.6	0.6	0.5	0.7	**0.8**	0.4	**0.7**	0.6	0.5	0.1
2.8B	0.6	0.8	0.5	0.7	**0.9**	20.8	4.5	20.8	**23.4**	11.1	0.6	0.5	0.7	0.8	**0.9**	0.4	1.0	**1.4**	0.6	0.9
6.9B	**0.6**	**0.6**	0.4	**0.6**	0.5	12.9	0.6	13.1	**16.8**	6.1	1.0	**1.4**	1.2	1.3	1.0	0.8	**1.6**	0.8	0.3	0.8
12B	**0.7**	0.6	0.6	**0.7**	1.0	13.9	0.8	14.2	**17.4**	4.9	1.0	**1.7**	1.1	1.5	1.0	1.0	**1.5**	1.3	0.7	0.9
	ArXiv					DM Math					HackerNews					The Pile				
# Params	LOSS	Ref	min-k	zlib	Ne	LOSS	Ref	min-k	zlib	Ne	LOSS	Ref	min-k	zlib	Ne	LOSS	Ref	min-k	zlib	Ne
160M	**0.8**	0.4	0.2	0.7	0.3	0.5	**1.4**	0.6	1.2	0.7	1.0	0.8	**1.2**	0.6	0.7	**2.4**	1.3	2.0	2.2	2.2
1.4B	0.3	**1.0**	0.2	0.4	0.7	0.8	0.8	0.6	1.0	**1.7**	0.7	0.9	**1.2**	0.9	0.8	**2.4**	1.4	**2.4**	2.3	2.3
2.8B	0.5	**2.1**	0.5	0.5	0.5	0.8	0.4	1.0	**1.3**	0.8	0.6	1.4	0.8	1.1	**1.7**	**2.8**	2.2	**2.8**	**2.8**	2.4
6.9B	0.6	**1.8**	0.6	0.6	0.6	0.9	0.2	0.6	**1.0**	0.7	.9	**1.9**	1.0	0.9	1.3	**2.6**	1.8	2.5	2.5	2.2
12B	0.6	**2.5**	0.6	0.5	0.9	1.0	0.5	0.5	**0.9**	0.8	0.7	**2.3**	0.8	0.8	1.4	**2.7**	1.8	2.6	2.6	–

Table 6.2: %TPR@1%FPR of MIAs against Pythia-dedup across different datasets from the Pile. The highest performance across the different MIAs is bolded per domain. In general, **leakage in high confidence settings is low ($< 3\%$)**. As with AUC ROC, GitHub is an exception, still yielding considerably higher leakage with most attacks. Unlike with AUC ROC, trends in performance are much noisier in the high-confidence setting, with trends in model size and best-performing attacks in certain domains no longer holding, reinforcing the difficulty in determining a *best* attack.

with the GPT-Neo and OLMo models to validate our findings with different model families, observing similar trends in most Domains.

GPT-Neo. We target the GPT-Neo model suite, which consists of 125M, 1.3B, and 2.7B-parameter models trained on the Pile. Table 6.3 shows that performance trends are similar to those observed with Pythia. In some domains such as HackerNews, the best performing MIA differs between target models (Min-k% Prob for GPT-Neo, reference-based for Pythia-dedup), though marginally.

OLMo. Benchmark construction is similar to that for Pythia. However, we sample from domains that make up Dolma [282], namely Wikipedia, C4, Reddit, Common Crawl, and peS2o [281]. These are similar to domains used in Pile. We note that peS2o consists of both abstracts (s2ag) and full papers (s2orc), and evaluate them as separate domains. To get non-member, we use held-out Dolma data from Paloma [197].

	Wikipedia				Github				Pile CC				Pubmed Central			
# Params	LOSS	Ref	min-k	zlib	LOSS	Ref	min-k	zlib	LOSS	Ref	min-k	zlib	LOSS	Ref	min-k	zlib
125M	.504	**.511**	.492	**.511**	.641	.582	.642	**.660**	.495	.492	**.500**	.497	.499	**.506**	.502	.499
1.3B	.510	**.531**	.506	.517	.681	.570	.681	**.696**	.500	**.517**	.503	.501	.496	**.499**	**.499**	.497
2.7B	.513	**.545**	.513	.519	.699	.570	.700	**.712**	.504	**.531**	.507	.506	.498	**.507**	.501	.499
	ArXiv				DM Math				HackerNews				The Pile			
# Params	LOSS	Ref	min-k	zlib	LOSS	Ref	min-k	zlib	LOSS	Ref	min-k	zlib	LOSS	Ref	min-k	zlib
125M	**.507**	.494	.503	.501	.492	**.522**	.493	.484	.489	.480	**.505**	.496	.502	**.507**	.505	.505
1.3B	.511	.506	**.512**	.507	.486	**.511**	.491	.481	.499	.500	**.514**	.501	.505	**.514**	.509	.507
2.7B	.515	**.520**	.517	.510	.486	**.509**	.492	.481	.502	.512	**.516**	.503	.507	**.519**	.511	.509

Table 6.3: AUC ROC of MIAs against GPT-Neo across different datasets from the Pile. The highest performance across the different MIAs is bolded per domain. Similar to Pythia-dedup, **MIA methods perform near random ($< .55$) in most domains**.

Table 6.4 also demonstrates generally near-random performance trends, with both the 1B and 7B parameter model variants exhibiting similar performances. We speculate that, due to the incredibly large amounts of training data (3T tokens for 1B-parameter model, 2.5T tokens for the 7B-parameter model), performance across different model sizes begins to converge to near-random performance even with such distinct model sizes. Interestingly, the Reference-based attack using Stablelm-Base-Alpha-3B-v2 performs much worse than when used to calibrate the Pythia models, reinforcing the difficulty in finding suitable reference models for different LLMs. We also observe many settings where MIA performance is considerably less than .5, suggesting that the MIAs are more likely to predict members as non-members, and vice versa. More investigation is needed to understand such behaviors on specific domains such as s2ag from peS2o in the Dolma data.

	Wikipedia				C4				Reddit			
# Params	LOSS	Ref	min-k	zlib	LOSS	Ref	min-k	zlib	LOSS	Ref	min-k	zlib
1B	.484	**.510**	.495	**.510**	.515	.479	**.520**	.513	.464	**.495**	.478	.470
7B	.481	.488	.493	**.500**	.516	.499	**.520**	.514	.463	**.501**	.480	.469
	Common Crawl				s2ag				s2orc			
# Params	LOSS	Ref	min-k	zlib	LOSS	Ref	min-k	zlib	LOSS	Ref	min-k	zlib
1B	.509	.412	**.517**	.511	.449	.376	**.461**	.392	.484	.480	**.500**	.463
7B	.498	.410	**.505**	.500	.465	**.483**	.475	.406	.491	**.507**	.503	.470

Table 6.4: AUC ROC of MIAs against OLMo across different datasets from the Dolma dataset. The highest performance across the different MIAs is bolded per domain.

6.4.2.2 Additional target model details

Pythia-dedup. Both the Pythia and Pythia-dedup model suites provide intermediate checkpoints for each model. For experiments targeting the Pythia-dedup model, as the Pythia-dedup model is trained for greater than 1 epoch, we select the checkpoint that most closely matches the one epoch mark over the deduplicated Pile. We decide this is checkpoint 'step99000'. For experiments targeting the non-deduped Pythia models, we use the final checkpoint, which sees just under one (≈ 0.9) epoch of the original Pile.

SILO. The models from the SILO suite [217] consist of 1.3B-parameter transformer LMs based on the OpenLM implementation of the LLaMA architecture [308]. These are trained for multiple epochs on the Open License Corpus, which consists of permissively-licensed text data classified as either **public domain** (PD) texts, **permissively licensed software** (SW), or under an **attribution license** (BY). We target the SILO-PDSW model (alongside its intermediate checkpoints) trained on only texts classified as PD or SW for domains contributing less than 5% of the data upsampled by a factor of 3x (which includes HackerNews and DM Mathematics).

Datablations. The Datablations suite [224] is a large collection of models trained to study scaling laws in data-constrained regimes. They vary in the extent of data repetition and compute budget, ranging up to 900 billion training tokens and 9 billion parameters. For the epoch experiment, we choose the 2.8B-parameter subset of models, with each seeing a total of 55B tokens from the C4 dataset across their training runs. These models vary in the number of epochs their training subset is seen, ranging from one to 14 epochs. They also offer a model trained for 44 epochs, which we decided to leave out of evaluation.

GPT-Neo. is a collection of 125M-, 1.3B-, and 2.7B-parameter models of similar architecture to the GPT-3 model family. These models are trained on the Pile for about 300B tokens, similar to the Pythia suite. This model suite is a precursor to the GPT-NeoX [10] model architecture, which Pythia-dedup and Pythia are built on. Noticeable differences include the tokenizer used per model suite, with the GPT-NeoX allocating additional tokens to whitespace characters, as well as intended training settings, with GPT-Neo geared towards TPU training and GPT-NeoX GPU training.

OLMo. The OLMo model suite [102] is a suite of open language models trained on the Dolma [282] dataset. OLMo models currently available include 1B- and 7B-parameter variants trained on 3T and 2.5T tokens, respectively. While our preliminary results just target the final checkpoints, the OLMo suite is similar to the Pythia suite in that intermediate checkpoints and exact training order are fully open.

6.4.2.3 Datasets

We use seven diverse data sources included in the Pile: general web (Pile-CC), knowledge sources (Wikipedia), academic papers (PubMed Central, ArXiv), dialogues (HackerNews), and specialized-domains (DM Math, Github). We also perform experiments over the entire Pile.

We sample 1,000 members and non-members from each target domain from the Pile train and test sets, respectively. e do the same for the aggregate Pile experiment, except we sample 10,000 members and non-members each from the complete Pile train and test sets. We sample documents greater than 100 words and truncate them up to 200 words from the beginning to create our benchmark examples. Previous work [276] observes that sample length correlates with performance, so we bound the sample length to reduce its

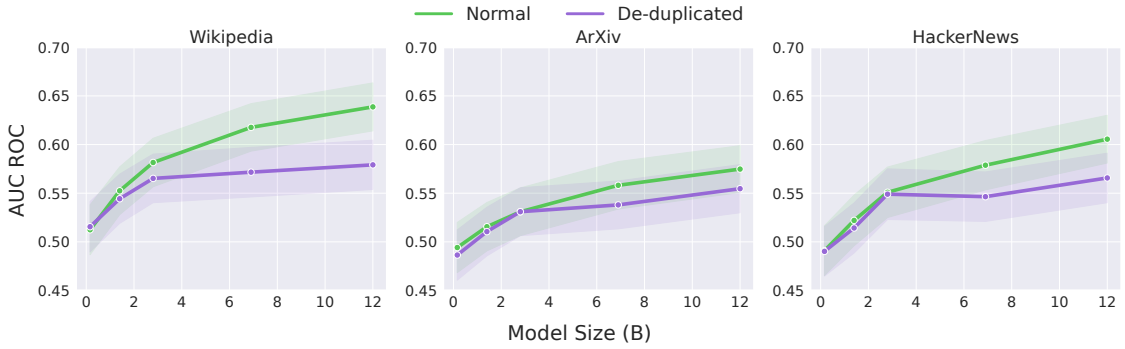


Figure 6.1: MIA performance as model size increases for the reference-based attack over select domains. We also plot the AUC ROC trajectory against the non-deduped Pythia suite for comparison. **Increasing model size slightly boosts MIA performance while deduplication decreases performance**. Other attacks follow similar trends (Figure 6.2).

impact while picking a reasonable threshold so that our samples are likely to contain ample signal. We follow the same pipeline when generating the benchmark for targeting the DATABLATIONS models, picking members and non-members from the C4 train and validation sets, respectively.

Gao et al. [97] decontaminated the Pile test set against the training set at a document level. Nonetheless, to be more rigorous, we perform additional deduplication following Groeneveld et al. [101], which uses a bloom filter to check for n-gram inclusion. We keep the default filtering settings of $n = 13$ and a threshold of $\leq 80\%$ overlap. Further details about setting up the bloom filter can be found in §6.4.5.1

Evaluation metrics. We primarily report **AUC ROC** for our evaluations, and additionally record **TPR@low%FPR** [43] to assess performance in high-confidence settings. We visualize the 95% confidence interval for AUC ROC scores via shaded regions.

6.4.3 Main Results

Table 6.1 shows that all existing MIAs perform near random for most domains[†]. No single MIA or target model demonstrates attack AUC above 0.6, with the exception of Github domain (see §6.4.5.3 for discussion). Overall, the reference-based attack performs best, although there are a few settings where other attacks perform better, e.g., Min-k% Prob on Pile CC for the 160M PYTHIA-DEDUP model. Marginal differences in performance across MIAs make it hard to single out an overall *best* attack.

MIA performance tends to increase with the target model size (Table 6.1, Figure 6.1), in agreement with prior work [176, 276, 335]. This is likely because larger models are more prone to overfitting the training data [226]. We also find deduplication of the training data reduces MIA performance (Figure 6.1), confirming the findings from Kandpal et al. [149].

[†]Similar trends for TPR@1%FPR. See Table 6.2.

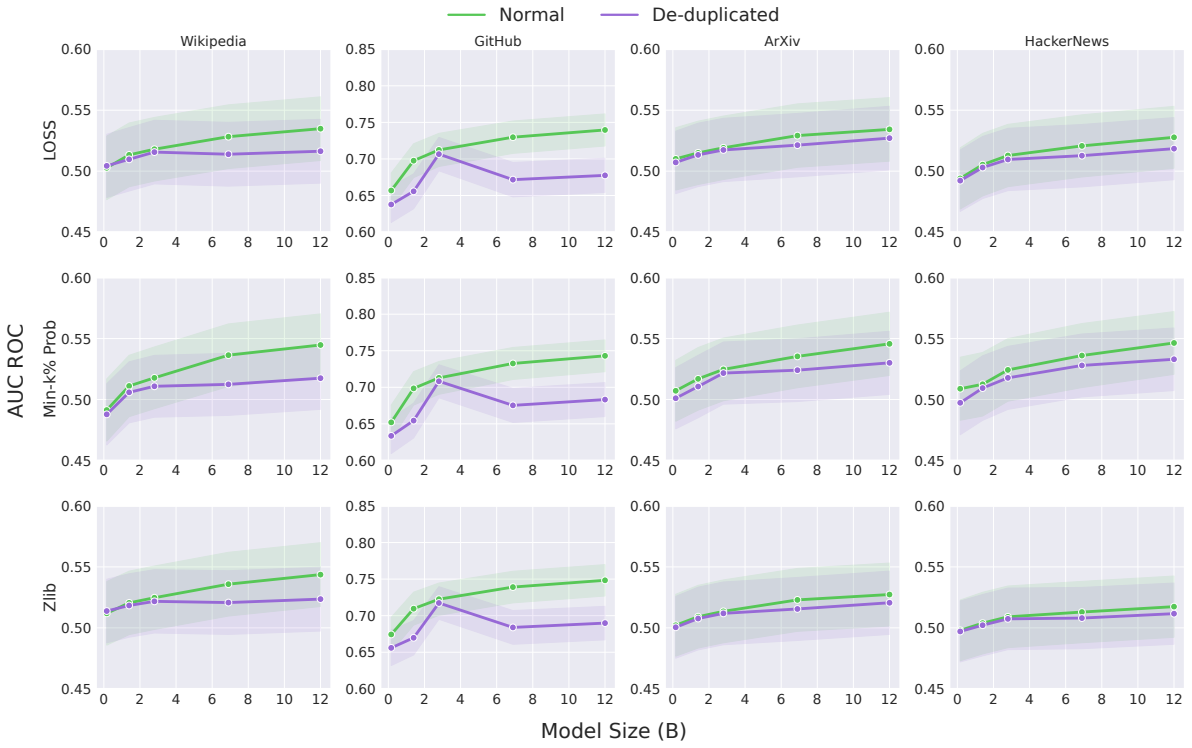


Figure 6.2: MIA performance as model size increases over select domains for various other attacks. We additionally plot the AUC ROC trajectory against the non-deduped Pythia suite for comparison. Similar to the reference-based attack, **increasing model size slightly boosts MIA performance while deduplication decreases performance**.

6.4.3.1 Difficulty in Choosing a Reference Model

We choose a diverse set of reference models to experiment with. For the aggregate method over all reference models, we take the average of the scores per reference model for a target sample[‡]. We report results for our complete ablation on reference model choice in Table 6.5.

GPT-2 [255] is suite of pretrained transformer trained on a large dataset of around 40GB of web text, likely overlapping with the Pile. We use the GPT-2-small variant with 124M parameters.

DistilGPT2 [269] is a smaller 82M-parameter model trained with the supervision of GPT-2-small using knowledge distillation.

OPT [370] is a suite of open-sourced pretrained transformers that are trained on a curated pre-training corpus including several datasets from the Pile, such as Wikipedia, DM Mathematics, and HackerNews. We use the 1.3B-parameter variant.

[‡]Note that the scores over different reference models may not be directly comparable due to the reference models having different tokenizers. This may contribute to the poor performance of this naive ensembling method.

	Wikipedia								Pile CC							
# Params	Gpt2	Distil	Opt	Neo	Silo	Llama	Stable	Pythia	Gpt2	Distil	Opt	Neo	Silo	Llama	Stable	Pythia
160M	.498	.502	.494	.490	.492	.511	**.515**	.480	**.520**	.504	.488	.473	.504	.487	.497	.480
1.4B	.503	.505	.507	.500	.502	.521	**.544**	.476	.523	.507	.513	.500	.516	.504	**.525**	.496
2.8B	.511	.510	.519	.532	.531	.539	**.565**	.526	.526	.509	.521	.499	.520	.510	**.537**	.504
6.9B	.510	.507	.517	.518	.516	.536	**.571**	.501	.538	.520	.542	.525	.531	.530	**.564**	.540
12B	.514	.510	.522	.528	.529	.546	**.579**	.517	.548	.525	.555	.538	.541	.545	**.582**	.555
	PubMed Central								ArXiv							
# Params	Gpt2	Distil	Opt	Neo	Silo	Llama	Stable	Pythia	Gpt2	Distil	Opt	Neo	Silo	Llama	Stable	Pythia
160M	.495	.491	.515	.511	.513	.515	**.516**	.497	**.523**	.518	.516	.480	.496	.492	.486	.472
1.4B	.493	.491	.514	.517	.514	.515	**.530**	.503	**.529**	.524	.523	.501	.512	.506	.510	.484
2.8B	.494	.492	.513	.518	.515	.518	**.536**	.500	**.534**	.528	.528	.524	.522	.516	.531	.528
6.9B	.499	.496	.519	.527	.520	.530	**.552**	.526	.540	.532	.534	.539	.531	.528	.538	**.554**
12B	.504	.498	.523	.531	.524	.538	**.559**	.533	.546	.538	.541	.555	.540	.538	.555	**.581**
	DM Math								HackerNews							
# Params	Gpt2	Distil	Opt	Neo	Silo	Llama	Stable	Pythia	Gpt2	Distil	Opt	Neo	Silo	Llama	Stable	Pythia
160M	.489	.488	.520	.509	.487	.502	**.523**	.514	**.496**	**.496**	**.496**	.480	.398	.486	.490	.466
1.4B	.487	.485	.509	.496	.485	.503	**.512**	.496	.508	.509	.511	.496	.401	.504	**.514**	.483
2.8B	.485	.486	**.511**	.503	.483	.500	.504	.509	.521	.522	.529	.534	.421	.521	**.549**	.527
6.9B	.485	.485	**.510**	.499	.484	.502	.508	.497	.525	.526	.534	.536	.436	.531	**.546**	.542
12B	.487	.486	**.514**	.504	.485	.502	.512	.503	.534	.533	.545	.559	.453	.545	**.565**	.561

Table 6.5: The effect of the choice of a reference model to Pythia-dedup models across various domains. The reference model yielding the highest performance, per target domain and target model, is bolded. ROC-AUC values are reported.

As mentioned in §6.4.2.2, GPT-NEO [31] is another suite of pretrained transformers designed using EleutherAI's replication of the GPT-3 architecture. These models are trained on the full Pile for a similar amount of tokens as Pythia ($\sim$ 300B), though the data seen may not necessarily be in the same order as the Pythia models. We use the 1.3B-parameter variant.

SILO-PDSWBY [217] is a 1.4B-parameter transformer pretrained on all types of permissively licensed data in the Open License Corpus. The training data consists of certain Pile domains such as HackerNews and DM Mathematics.

LLAMA [308] is a collection of large, open-sourced pretrained LMs ranging in size from 7B to 65B parameters. The pre-training corpus is on the scale of trillions of tokens, much larger than the Pile, and likely has significant overlap with the Pile. We use the 7B-parameter variant.

StableLM-Alpha-v2 [310] is a set of open-source pretrained LMs also trained on a large pre-training corpus with trillions of tokens. Training is conducted in two stages, with the first stage seeing 1 trillion tokens of a mixture of data from sources such as RedPajama [6] and the Pile, with an emphasis on refined web text. The second stage is trained on 100 billion tokens with a higher context length, increasingly sampling naturally long texts and adding the StarCoder [179] dataset. We use the 3B-parameter variant.

We also experiment with the non-deduped Pythia-dedup-1.4B model as a reference model to see how using a smaller version of the target model (same architecture and training data order) impacts reference-based attack performance [42].

Stablelm-Base-Alpha-3B-v2 Performance. We speculate that the slightly higher performance with StableLM-Base-Alpha-3B-v2 as the reference model, even though its pre-training corpus has high overlap with the Pile, is because 1) larger target models[§] such as the Pythia-dedup-12B model may considerably overfit certain member samples and 2) the StableLM-Base-Alpha-3B-v2 is trained on a much larger corpus, which helps it generalize well and achieve similar losses as the target model on the non-member data. As a result, member samples are more likely to have a greater magnitude of difference between the target and reference model losses compared to the difference between losses on non-members.

We ablate the choice of reference models in the reference-based attack. In summary, (1) most reference models yield poor performance, with StableLM-Base-Alpha-3B-v2 being the best overall, and (2) even aggregating all reference models performs poorly. In general, we find choosing the right reference model for a target LLM challenging and largely empirical. A reference model should be trained on the data that is same-distribution but largely disjoint from the training data of the target model. However, this assumption is hard to impose at the scale of pre-training corpora; common practice is to collect all the data available on the web, leading independently collected datasets to naturally overlap with each other.

6.4.4 Why is MI Challenging against LLMs

We identify several key factors that may contribute to the decreased performance of MIAs on LLMs. Some factors are due to unique characteristics of practices in LLM pre-training (§6.4.4.1) while others are due to inherent ambiguity in MIA (§6.4.4.2).

6.4.4.1 Characteristics of LLMs

Training Data Size. Current state-of-the-art pretrained LLMs are trained with billions and trillions of tokens [301, 308, 309]. We hypothesize the ***large pretraining corpora characteristic to LMs decreases MIA performance***, as larger pretraining datasets lead to better generalization [119, 224].

We employ the Pythia-dedup model suite's intermediate checkpoints to assess the impact of different amounts of training data. While keeping non-members fixed, we sample members for each checkpoint from its most recent 100 steps to remove the impact of the recency bias of the members. [¶]

For each model, we pick checkpoints every 5000 steps ending at step 95000, with each step corresponding to 1024 samples of length 2048 tokens. We also include checkpoints at step 1000 and step 99000, the closest checkpoint to the step where one full epoch of the deduplicated Pile was seen. For each checkpoint, we use the same non-member set for evaluation consisting of 1000 samples sampled from the entire Pile test set. We

[§]Also target models that are domain specific like Datablations or are trained on a less diverse corpus like SILO

[¶]Using recently seen members elevates MIA performance noticeably, but doesn't disrupt the impact of increasing training data size.

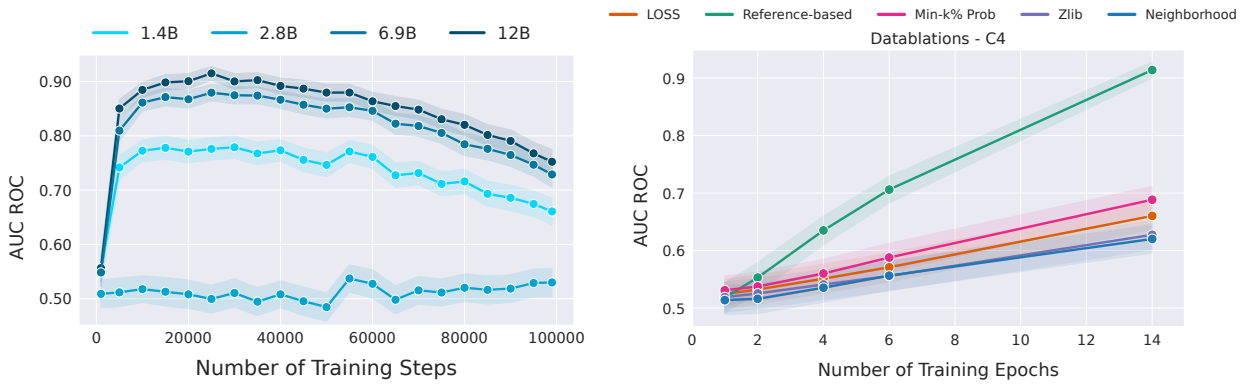


Figure 6.3: (Left) Reference-based attack performance as the amount of training data seen, measured in the number of training steps, increases across 1 epoch of the deduplicated Pile. In general, **performance spikes greatly before gradually decreasing as the amount of training data seen increases**. Other attacks (Figure 6.4) follow similar trends. (Right) MIA performance on target model DATABLATIONS as the number of effective epochs increases via increasing epoch count. **Performance increases linearly with the number of effective epochs**. See Figure 6.6 for results on SILO.

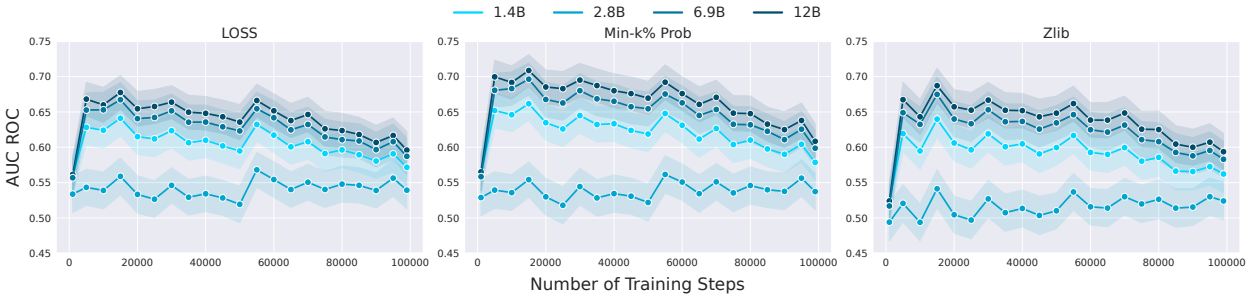


Figure 6.4: MIA performance as the amount of training data seen increases across 1 epoch of the deduplicated Pile pretraining corpus, visualized over a range of model sizes for various attacks. We use the training step as a unit for the amount of training data seen, with 1 step corresponding to seeing 2097152 tokens. AUC-ROC reported. Similar to the reference-based attack, for all attacks, **performance drastically increases before gradually decreasing as the amount of training data seen increases**.

then construct a member set for each checkpoint[‖]: for the checkpoint at step n, we sample 1000 random samples from documents seen within the range step $\{n - 100, n\}$. EleutherAI provides random seeding for deterministic training data order across the PYTHIA-DEDUP training runs, which we use to determine the seen document order. This allows us to determine which documents to sample from for a given step range. For both members and non-members, we sample with the same criterion as the general experiments above.

MIA performance generally starts as near-random, then rapidly increases within the next few thousand steps, before decreasing across successive checkpoints [**] (Figure 6.3, left). We speculate the initial low performance is due to the model warming up in training, with high losses across both member and non-member samples.

[‖]Ideally, the member set should be fixed, which could be done by performing multiple training runs and injecting the fixed member set at various steps. However, this is computationally expensive. Furthermore, because of how the data is shuffled, we'd expect the difficulty of the member set to be reasonably consistent across our samples

[**]PYTHIA-DEDUP-2.8B stands apart with a performance trajectory that is consistently near-random. Previous work also observes unexplainable behavior for this model [29].

We believe the following rapid rise and then gradual decline in performance are because the data-to-parameter-count ratio is smaller early in training and the model may tend to overfit, but generalizes better as training progresses, in line with observations in existing work [226].

Recency of Member Samples. We explore how the recency of member samples seen in training impacts MIA performance. We follow the same setup as the training data experiment, but instead of evaluating the checkpoint at step n with the member data sampled from within steps $\{n-100, n\}$ and the fixed non-member set, we fix the target model, only targeting the checkpoint at step 99000 for all the benchmarks.

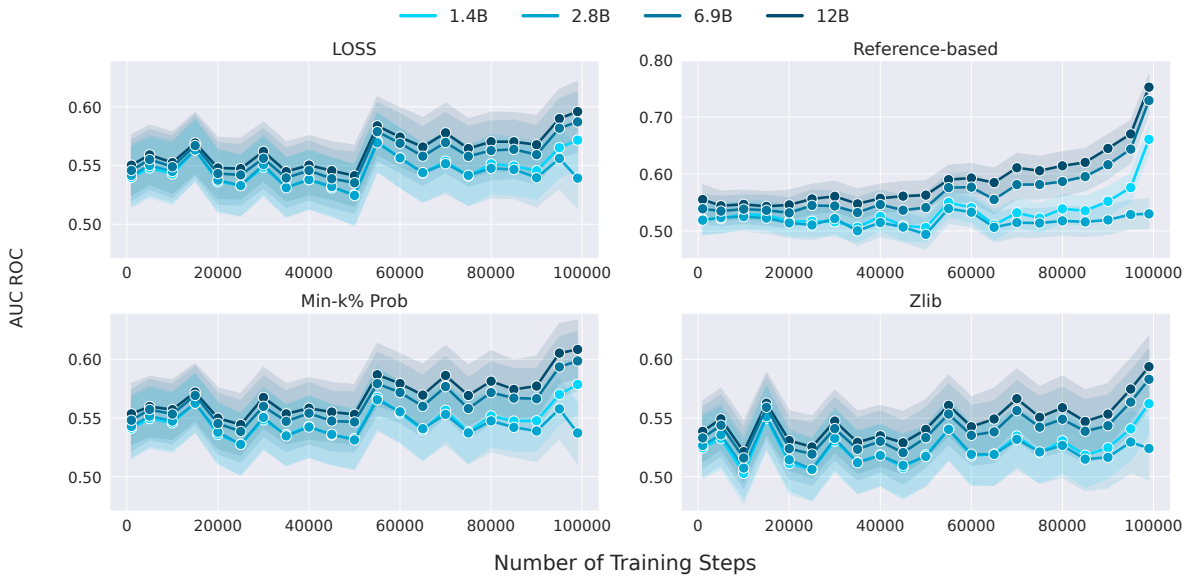


Figure 6.5: MIA performance for different member data sets sampled at different training steps across 1 epoch of the deduplicated Pile pretraining corpus, visualized across different attacks. Target model is the PYTHIA-DEDUP-12B checkpoint at step-99000. AUC-ROC reported. **Performance on benchmarks with more recently seen members is higher, but gradually decreases to a plateau for less recently seen members**.

Figure 6.5 demonstrates that, in general, member data seen more recently by the given checkpoint contributes to slightly higher MIA performance. We believe this supports existing work in LM forgetting [131], where observed patterns in recently seen training data are better preserved in the model parameters, while earlier seen data are less memorized.

We also note that the MIA performance trajectories seem to drop slightly more quickly for smaller models, though the trajectories across all model sizes seem to converge when evaluating on member data from much earlier in the training run. We speculate this is a result of larger models having more parameters, allowing them to capture more seen data before having to drop older knowledge.

We also note that, in the context of MIA against fine-tuning datasets, our results indicate that data seen during fine-tuning or continued pre-training may also be increasingly vulnerable due to how recent they are

seen. This aligns with previous work demonstrating high MIA performance on fine-tuning datasets [94, 219]. This is especially relevant in practice since fine-tuning is a popular option to re-purpose large pretrained models for varying downstream tasks such as commercial use cases, which often involves tuning with sensitive data.

Number of Training Epochs. It is standard practice to pre-train LLMs for around one epoch, given the scale of data and their tendency to overfit quickly [161, 224]. Previous MIA works that demonstrate attack effectiveness consider supervised fine-tuning or masked LM pre-training [171, 219, 220, 225], where models are trained for more than 10 epochs. We explore how the ***near-one epoch training of LLMs leads to decreased MIA performance***.

To verify this hypothesis, we perform MIA against the **Datablations** suite [224], consisting of models trained on subsets of C4 [257] train data for varying numbers of *epochs*. Increasing the number of effective epochs corresponds to an increase in attack performance (Figure 6.3, right).

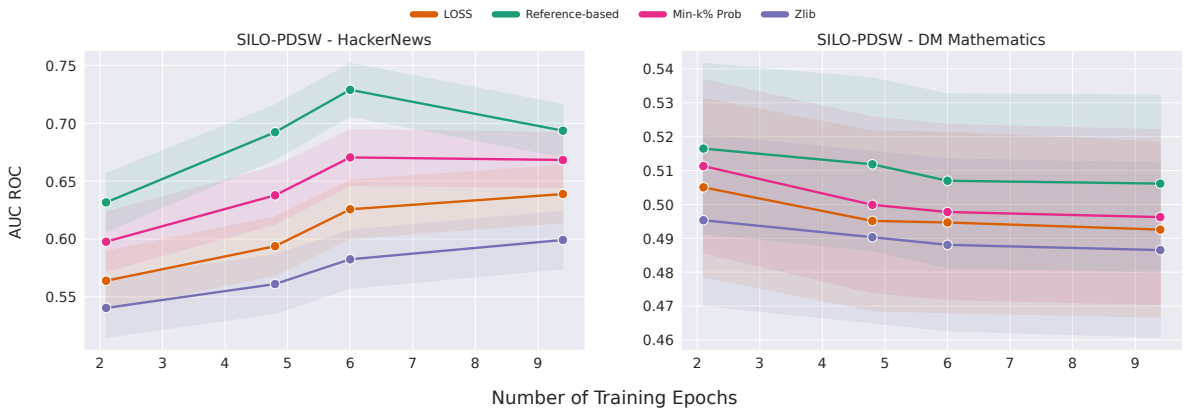


Figure 6.6: MIA performance on target model SILO-PDSW as the number of effective epochs in which the member domain data has been seen increases. AUC-ROC reported. For HackerNews, **performance does increase with an increasing number of effective epochs initially, but begins to plateau or even drop with further epochs**. For DM Mathematics, **performance surprisingly drops with increasing effective epochs**.

We also explore a more realistic setting where the amount of training data the target model sees increases alongside the effective epoch count by targeting the **SILO** [217]-PDSW model and intermediate checkpoints. For HackerNews, we observe MIA performance initially increases with more effective epochs, similar to the DATABLATIONS setting, but then begins to plateau or drop as effective epoch count continues to increase (Figure 6.6). DM mathematics, on the other hand, surprisingly decreases as the number of effective epochs increases. We speculate over factors that may contribute to these observations:

- HackerNews, even when up-sampled for this variant of the SILO-PDSW model, still only makes up 5.9% of the training data [217]. In the first few epochs, when the total training data seen is low, the model can memorize the HackerNews samples. However, as the number of epochs increases, the target

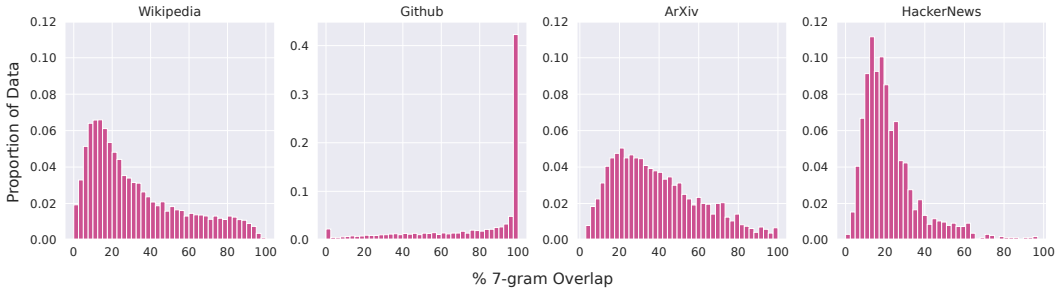


Figure 6.7: Distributions of 7-gram overlap of non-member data over select domains. Github has a considerably higher overlap than other domains.

model may tend to overfit data more so from domains with greater representation. As the SILO model is on the smaller side with 1.4B parameters, we suspect the target model begins to memorize less of the HackerNews samples, leading to a plateau or drop in MIA performance.

- DM Mathematics also makes up only 3.5% of the training data. In addition, with DM Mathematics being a dataset of mathematical problems, we suspect that the abundance of tokens from a concentrated token space (i.e., digits, variables) that are largely symbolic rather than semantic makes memorization of specific samples unlikely. Overall, it simply fails to perform well on such data even after multiple epochs (as observed when looking at model loss values for this data).

For both cases, further investigation is needed into the target domains and attack setting setup to better understand these counter-intuitive phenomena. While Muennighoff et al. [224] shows training for multiple epochs helps improve performance, our results suggest that such multi-epoch training (and/or large upsampling factors) can increase training data leakage.

6.4.4.2 Inherent Ambiguity in MIA

Natural language documents commonly have repeating text—even with the best efforts in decontamination and deduplication. These include common phrasings and quotes, natural use of similar texts, and syntactical similarities inherent to specific domains. This leads to substantial text overlap between members and non-members, which motivates the following hypothesis: ***higher overlap between members and non-members increases MIA difficulty***.

We quantify overlap using the percentage of n-gram overlap, defined as: For a non-member sample $\mathbf{x}$ consisting of m words such that $\mathbf{x} = x_1x_2...x_m$ and an n-gram in $\mathbf{x}$ defined as a continuous substring $x_i...x_{i+n-1}$, the n-gram overlap of $\mathbf{x}$ on training dataset D is

$$\frac{1}{m-n+1}\sum_{i=1}^{m-n+1}\mathbb{1}\{\exists y \in D : x_i...x_{i+n-1} \in y\}$$

Domain	Wikipedia		Github		PubMed Central		Pile CC		ArXiv	
	Orig	7-gram	Orig	7-gram	Orig	7-gram	Orig	7-gram	Orig	7-gram
LOSS	.516	.666	.678	.878	.506	.780	.516	.574	.527	.787
Ref	.579	.677	.559	.615	.559	.595	.582	.644	.555	.715
min-k	.517	.644	.683	.890	.512	.792	.521	.578	.530	.734
zlib	.524	.631	.690	.908	.506	.772	.517	.560	.521	.780
Ne	.520	.612	.660	.877	.497	.737	.514	.566	.519	.773

Table 6.6: Comparison of MIA performance over select domains with varying non-member sets at $\leq 20\%$ n-gram overlap threshold for $n = 7$, as well as the natural non-member set. Target model is Pythia-dedup-12B and AUC ROC reported. **Strict n-gram overlap thresholding results in higher performance**.

We first compute the percentage of 7-gram overlap for non-members against the entire Pile training (member) set (Figure 6.7). Figure 6.8 shows n-gram overlap distributions for other n. See §6.4.5.1 for implementation details. We observe high n-gram overlap with training data for a substantial portion of non-members; e.g., the Wikipedia, ArXiv, and PubMed Central domains have average 7-gram overlaps of 32.5%, 39.3%, and 41.0%, respectively. Domains such as GitHub, DM Mathematics, and FreeLaw see even higher overlap, with mean 7-gram overlap of $76.9\%, 72.8\%$, and 62.3%, respectively.

High n-gram overlap suggests that substrings of non-members may be seen exactly during training, which makes the distinction between members and non-members even less clear. To verify our hypothesis, we resample non-members ensuring $\leq 20\%$ n-gram overlap with members, and report MIA performance (Table 6.6). While this step is designed to more strictly eliminate instances of non-member records that may overlap with training records, it also introduces an explicit drift between member and non-member distributions by selecting non-members that are most "unlike" training records. We clarify that this step is not a suggestion for researchers to alter their benchmarks; such a processing step drifts away from the standard membership inference game [355].

Results. MIAs perform significantly better as the non-member distribution concentrates towards lower n-gram overlap, diverging from the natural n-gram overlap distribution of non-members from the training distribution e.g., .516→.666 in Wikipedia, .690→.908 in Github, and .512→.792 in PubMed Central for various attacks. This is intuitive as the target model is likely to assign a lower likelihood to non-members further from its training data, making members and non-members more distinguishable. Note that decreasing the n-gram overlap threshold, especially for smaller n, pushes the setting closer to distribution inference [293], since the distributions of 'member' and 'non-member' records are no longer the same. We further discuss outlier behavior in §6.4.5.

We note that n-gram overlap is an intrinsic property of natural language rather than a problem of the Pile train-test split. These splits are already deduplicated at a document level, following standard practice in decontamination [37, 97]. Nonetheless, repeating texts across distinct documents are fundamental and natural properties of domain data. We also note that n-gram overlap distribution analysis can help assess how

representative of a target domain a set of candidate non-members is when constructing MIA benchmarks. Ultimately, we highlight the need to consider qualities of the data domain, e.g., n-gram overlap, and understand their potential impact on MIA performance.

6.4.5 n-gram Overlap Details and Takeaways

6.4.5.1 Measuring n-gram Overlap

We create a bloom filter following Groeneveld et al. [101]. Due to the scale of the Pile training data and limited memory, we shard the bloom filter. In our construction, we split the training data in half, resulting in two bloom filter shards. Since each shard only sees half of the training data, to check for n-gram inclusion across the entire Pile, we check for containment in both of the sharded bloom filters, counting an n-gram included only if it is included in at least one of the bloom filters.

For each shard, we configure the bloom filter according to the data size such that the false positive rate of the bloom filter is less than 1% (0.6%). Then, for each document, we tokenize at the word level. We then add n-gram s̃ to the filter by using a striding window over n words at a time with a stride of 1. We use the same method of gathering n-gram s̃ when checking the non-members for n-gram overlap.

6.4.5.2 Reference-based Attack Performance

Table 6.6 shows that, interestingly, reference-based MIAs have a noticeably smaller increase in performance compared to non-referenced-based MIAs for domains such as GitHub or PubMed Central under n-gram overlap thresholding. We speculate that, since numerous low n-gram overlap non-members are outliers to the relevant domain, these non-members will also be outliers to the similar/overlapping data seen by the reference model. As a result, even though these non-members may yield higher losses from the target model, we see similar high losses for the reference model as well, which makes the difference between target and reference model loss for non-members and members relatively less distinguishable compared to signals from the other attacks.

At the same time, domains like Pile CC do not see this dampened performance, likely because the 20% threshold in the case of Pile CC is not sufficient to select outliers, as samples from this domain have naturally low n-gram overlap. Another case where the reference-based attack seems to avoid this observation is in the temporally shifted non-member setting for both Wikipedia and ArXiv despite the temporally shifted non-members being more out-of-distribution relative to the Pile Wikipedia and ArXiv distributions, respectively. We speculate this is due to the reference model of choice, StableLM-Base-Alpha-3B-v2, which has not only been trained on a corpus with high overlap with the Pile, but also trained on datasets that capture more recent data such as RedPajama-Data-1T [6] which contains Wikipedia and ArXiv samples from a much more recent cutoff date (i.e., RedPajama uses the 2023-03-20 Wikipedia dump), allowing it to generalize

better over the temporally shifted non-members and avoiding a shift towards higher losses that weaker or older reference models might experience.

Attacks like LOSS or Min-k% Prob do not utilize any external signal or difficulty calibration, and thus rely exclusively on signals from the target model for member classification. Calibration-based methods like zlib and reference-based attacks, on the other hand, account for the inherent "difficulty" of a seen sample. Thus, in situations where the non-member data is significantly out of domain, even for a reference model or calibration method, it is likely that the signals from the target model and difficulty calibration would cancel out, leading to a weakened MIA signal. On the other hand, difficulty calibration can further boost MIA signal in settings where the member data is inherently more likely to be memorized, such as in §6.4.4.1 where reference-based attacks yielded considerably higher MIA performance in low training data size and high effective epoch count settings, with performance being further amplified in the extremes of both settings. Thus, MIA baselines for new MIAs should include both kinds of methods: calibration-based and calibration-free. Having baseline coverage for both styles of MIA can help uncover inherent characteristics of the evaluation setting such as unintentional member/non-member distributional shift or overfitted target models that influence MIA performance and also paints a holistic picture with regards to what MIAs are most suitable for specific attack settings.

6.4.5.3 GitHub as an Outlier

As seen in Table 6.1, MIA performance in the Github domain even without thresholding is notably higher than that in other domains, with the best method (zlib) achieving an AUC ROC of $\sim .70$. We speculate this is not because the GitHub domain, or code in general, is an easier domain to attack, but because the presumably reasonable decontamination threshold of $\leq$80% 13-gram overlap threshold only captures a small percentile of non-members as GitHub is naturally very high overlap.

We speculate a large factor contributing to the high overlap is the repetitive nature of code, such as copyright notices, function definitions, and syntax like HTML tags. Figure 6.9 demonstrates how our decontamination threshold impacts the 7-gram distribution of non-members. Non-members under our decontamination threshold are more likely outliers to the GitHub domain (see Figure 6.10 for an example of such an outlier). The additional n-gram overlap threshold experiments (§6.4.4.2) only exacerbate the impact of thresholding, which leads to notably higher MIA performance.

Such observations indicate why lexical non-member boundaries may lead to ambiguous interpretations of MIA performance in high-overlap domains. Here, using semantic differences between samples to draw non-member boundaries may be key to better understanding membership leakage in such domains.

6.5 Importance of Candidate Set Selection

In contrast to our findings in §6.4.2, recent works report state-of-the-art MIAs achieving $> .7$ AUC ROC on pretrained LLMs [211, 276]. Our in-depth investigation highlights one such reason for the differences is due to an inherent but likely unintended distribution shift between members and non-members in their settings.

Experimental Setup. Prior work distinguishes members and non-members of a target domain based on the knowledge cutoff date of the target model, with members coming before and non-members coming after the cutoff. We construct similar experimental settings under two domains also used in earlier works: Wikipedia and ArXiv.

Wikipedia. For the temporal Wikipedia benchmark non-members, we collect samples from the RealTimeData "**wikitext_latest**" dataset [183]. This yielded Wikipedia articles created between the week of August 12, 2023 till the week of January 8, 2024[††]. We then follow Pile processing steps by simply appending the article titles to the front of each respective article with a "\n\n". We note that Pile members are sampled from articles in a Wikipedia dump from before March 2020 [97]. Members and non-members are then sampled with the same criterion as in the general experiments.

ArXiv. For the temporal ArXiv benchmarks, the member set for each benchmark is fixed and sampled from the ArXiv subdomain of the Pile training set, which consists of papers posted prior to July 2020 [97]. For non-members, we use the ArXiv API again following Li et al. [183] to collect ArXiv preprints from specific months: August 2020, January 2021, June 2021, January 2022, June 2022, January 2023, and June 2023 [‡‡]. We then apply the same processing steps used in the Pile [97]. This mainly involves converting the latex sources for a given preprint into a single Markdown file, and then filtering out documents such as those with conversion errors. For each month range, we sample non-members from processed files in the given date range. By sampling non-members from successively later time ranges after the Pile ArXiv cutoff date, we also seek to explore how greater temporal shift impacts MIA performance. We again sample both members and non-members with the same criterion as in the general experiments.

Results. Table 6.7 demonstrates that the temporally shifted settings yield MIA performances significantly higher than when members and non-members are from the same temporal range. Figure 6.12 also demonstrates that MIA performance generally increases as non-members are further temporally shifted from member data. We speculate this follows from changes in language such as the introduction of new terminology and ideas over time.

[††]Note that while the articles are created in the recent time frame, the contents of the Wikipedia page aren't necessarily about recent topics, people, or events

[‡‡]This slightly differs from the Pile ArXiv data collection, which uses the ArXiv bulk access through S3. However, we believe both ArXiv bulk access and API should yield the preprints in the same manner regardless.

Temporal Shift as Change in n-gram Overlap. We interpret temporal shift as a change in n-gram overlap distribution between the original and temporally shifted non-members. Figure 6.11 demonstrates that the distribution of 7-gram overlap of such shifted data concentrates at lower overlap percentages compared to their natural counterparts. The natural Wikipedia non-members have an average 7-gram overlap of 39.3%, whereas for the temporally shifted Wikipedia non-members it is 13.9%.

Figure 6.13 reinforces our observations in Figure 6.11, as similar to the temporal Wikipedia setting, temporally shifted non-members from after the target model's knowledge cutoff date are concentrated at considerably lower % n-gram overlap than non-members from the natural ArXiv non-member set. This contributes to the greater MIA performance in general over the temporally shifted ArXiv benchmarks. However, we note that n-gram overlap distribution shift does not provide a strong interpretation for the increase in MIA performance as non-members are increasingly temporally shifted. For example, the average 7-gram overlap of non-members from the month 2020-08 is 22.7% while the average for the month of 2023-06 is 20.5%. While there is a small decrease in average 7-gram for later non-members, the change is quite small and doesn't clearly justify the considerable difference in MIA performance when evaluating on benchmarks using non-members from the different months (i.e.,.723 $\rightarrow$ .795 AUC ROC from the 2020-08 benchmark to the 2023-06 benchmark). We speculate other factors that contribute to this increase include changes in the distribution of topics (i.e., increasing popularity of research into LLMs) and the presence of specific identifying tokens (i.e., dates, references, new terminology). We believe such factors only further reinforce the need to carefully analyze MIA benchmark construction when evaluating MIAs to understand what signals are truly being captured.

In general, when aiming to assess MIA performance, we advise estimating how representative a sample non-member set is of the member domain by comparing its n-gram overlap distribution with that of a left-out sample set from the pretraining corpora. Particularly, if the distribution of the candidate set is noticeably shifted towards lower n-gram overlap compared to the left-out member sample set, the candidate non-member set may not be representative of the member distribution from the target domain and potentially high MIA performances should be carefully examined. Closer inspection (Table 6.8) reveals the extent of such over-estimation; decision thresholds derived using temporally-shifted non-members end up testing for temporal shift rather than membership. We note that distinguishing between members and temporally shifted non-members is a realistic inference game with practical implications but differs from the classical MI game as temporally-shifted data may belong to a different distribution.

6.6 Revisiting Membership

The definition of membership in the standard MI game treats only records seen exactly during training as members, e.g., for language models, substrings appearing exactly in the training corpus. However, this may be at odds with what adversaries and privacy auditors care about when concerning information leakage. For

Thresholding Benchmark	1%				5%				10%			
	LOSS	Ref	min-k	zlib	LOSS	Ref	min-k	zlib	LOSS	Ref	min-k	zlib
2020-08	3.2	4.2	4.5	3.7	12.6	13.4	13.5	13.8	24.1	23.3	24.6	20.2
2021-01	3.7	3.9	3.5	3.5	11.4	15.8	13.5	10.4	21.7	27.0	24.6	17.5
2021-06	3.2	4.2	5.7	5.4	14.4	16.0	15.7	13.6	25.5	25.5	29.5	23.0
2022-01	4.5	4.2	5.3	4.1	14.4	16.3	14.6	12.7	24.5	27.0	28.7	22.0
2022-06	2.8	3.9	3.1	2.5	10.3	18.1	13.1	10.7	23.4	27.8	25.4	20.6
2023-01	2.9	8.5	3.5	3.1	11.9	23.5	13.5	10.9	25.0	36.1	26.3	21.9
2023-06	5.8	9.4	5.5	5.8	15.6	22.7	19.1	14.1	26.3	37.3	27.8	22.2
Temporal Wiki	9.8	7.5	10.3	7.9	23.8	22.8	24.3	17.6	30.0	34.1	35.0	22.8

Table 6.8: FPR (%) on non-members from the Pile (original; not temporally shifted) on various attacks when using a score threshold that achieves a 1, 5, or 10% FPR on the temporally-shifted ArXiv (for varying levels of temporal shift) and Wikipedia benchmarks. The target model is Pythia-dedup-12B. **FPRs on the original non-members are much higher then the thresholded FPR on the temporally shifted benchmarks**, indicating that such thresholds may be moreso classifying temporal shift rather than member and non-members.

generative models especially, guessing the membership of some sample via other sufficiently close samples can be useful. For example, any paraphrase of "Product launch in Q2, 2025" may be relevant as long as it preserves information regarding the product launch timeline, even if the paraphrase has a significant lexical difference, e.g., "Launching product in Q2, 2025" or "Q2 2025 release". Note that standard notions of Differential Privacy [74] do not immediately protect against such cases, since records being tested for membership are not, in the literal sense, members. We explore two methods of constructing such "sufficiently close" samples.

6.6.1 Lexical Distance

We first experiment with creating modified member samples by replacing n random tokens in a given sample with tokens randomly sampled from the model's vocabulary. We do so for $n = \{1, 10, 25\}$ (20 trials per n) and visualize the distribution for MIA scores using LOSS and Reference-based attacks (Figure 6.14, top). The LOSS attack yields distinct loss distributions between the modified members and original member/non-members, suggesting that the model is sensitive to out-of-place random tokens even for lightly perturbed member samples. The Reference-based attack, on the other hand, has a distribution of modified members much closer to both members and non-members, likely due to the reference model calibration accounting for the complexity introduced by the random tokens. This further reinforces the ambiguity of such samples—should they be considered members or non-members?

We also compute the thresholds corresponding to certain FPRs for actual member and non-member data and use these thresholds to compute the FPR on the modified members. We consider these modified members as "non-members", which they are with regards to exact match[§§]. §6.6.1 shows that these modified members have extremely low FPRs for edit distance as low as $n = 1$, suggesting that these samples would be classified as non-members by the MIA, even though from the perspective of information leakage such a sample is

[§§]We perform token replacements at random with random tokens, so it is unlikely that these edited members are also actually members.

Domain	1%			5%			10%		
	1	10	25	1	10	25	1	10	25
ArXiv	0.1	0.0	0.0	0.3	0.1	0.1	0.7	0.3	0.2
Wikipedia	0.0	0.0	0.0	0.2	0.1	0.1	0.6	0.4	0.1

Table 6.9: FPR (%) on modified members (treated as non-members) when using a score threshold that achieves a 1, 5, or 10% FPR on the original member and non-member data for ArXiv and Wikipedia domains. Results for lexically similar modified members at edit-distances $n = \{1, 10, 25\}$. Reference-based attack is shown. For LOSS attack, all FPR values are 0 across all tested FPR thresholds and values of n.

Domain	LOSS			Ref		
	1%	5%	10%	1%	5%	10%
ArXiv	0.0	0.8	2.5	0.7	1.9	4.0
Wikipedia	0.0	0.5	2.3	0.4	3.0	8.2

Table 6.10: FPR (%) on modified members (treated as non-members) when using a score threshold that achieves a 1, 5, or 10% FPR on the original member and non-member data for ArXiv and Wikipedia domains. Results for semantically close modified members. LOSS and Reference-based attack reported.

effectively a member. We highlight the need to rethink membership for samples with extremely low lexical distance from actual training members, though even membership at higher lexical distances is important with respect to what information is still leaked in the unperturbed portions.

6.6.2 Semantic Distance

While a small edit distance suggests closeness in meaning, a higher edit distance does not necessarily imply loss of semantics. We compute MIA scores for neighbors generated for member samples as part of the Neighborhood attack for the Wikipedia and ArXiv benchmarks and repeat the above pipeline. Visualizing the scores shows how the modified members are not too far from original member score distributions, especially for the Reference attack (Figure 6.14, bottom). We repeat the same FPR experiment as edit-distance-based modified members. While the FPR for these semantically close records is noticeably higher than records close by edit distance, the false positive rates are still low (§6.6.1). Again, these results suggest that semantically close members would be classified as non-members even though they may be as useful as actual members depending on the inference goal and the semantic information preserved.

While it is not surprising that semantically close neighbors have MIA scores more similar to actual member samples than randomly-replaced tokens, it is clear that an ideal distance function should combine the benefits of lexical distance and semantics in defining a membership neighborhood. Such observations also motivate a fully semantic MI game, where a neighbor member may be defined by its proximity to an exact member in a semantic embedding space. This may provide a clearer interpretation of knowledge leakage than lexical matching, especially when samples naturally have high lexical (i.e., n-gram) overlap.

Parahprasing with GPT. Here, we focus on generating member paraphrases that are semantically similar while preserving as much specific information from the original member sample as possible. To do so, we follow [350] and prompt GPT4 [242] to paraphrase member samples in a different style (5 trials per member). We perform this paraphrasing over the ArXiv, Wikipedia, and HackerNews domains; see Table 6.11 for domain-specific prompts. In general, we don't focus on the lexical similarity of the paraphrases unlike the earlier semantically similar samples generated via masking and replacing a small percentage of tokens. However, with HackerNews, we do specify a comment structure different from how HackerNews records were formatted for model training, a noticeable lexical difference.

Domain	Prompt
ArXiv	Please help me paraphrase the following text chunk from a research paper in a different style. Importantly, for sentences containing specific details like mathematical definitions or proofs, only make minimal changes and ensure these details are included exactly in the paraphrase. If the paper includes a title or authors, please keep them in the rephrase. If not, please DO NOT make up a title. Use a similar number of words.
Wikipedia	Please help me paraphrase the following text chunk from Wikipedia in a different but concise style. Importantly, for sentences containing specific details, make minimal changes and ensure all details are included correctly in the paraphrase. Use a similar number of words.
HackerNews	Please help me paraphrase the following conversation chunk from a thread in HackerNews while maintaining the conversational style. Follow this structure for each comment in the thread: [user] - [comment]. Ensure all user's comments are represented in the paraphrase. Make sure all details in each user's comments are included correctly in the paraphrase, such as links. Be specific and don't generalize.

Table 6.11: Instructions used to prompt GPT4 to obtain paraphrased members.

We again visualize the score distributions between the paraphrased members, and original members and non-members (Figure 6.15). We observe in general across both the LOSS and Reference-based attacks over the three domains that the paraphrased member score distributions are distinguishable from the original member and non-member score distributions but have noticeable overlap, similar to what was observed with masking-based semantic neighbors. However, when we perform the FPR experiment (Table 6.12), we see that in high-confidence settings, the paraphrased members are likely to be classified as non-members. Both the LOSS and Reference-based attacks seem noticeably insensitive to such paraphrased neighbors.

6.7 Conclusion

In this chapter, we provided a taxonomy of memorization in LLMs, discussing challenges with defining memorization and the consequences of memorization for various domains. We also study verbatim memorization and shed light on the difficulty of membership inference against LLMs from the lens of an adversary. Our

Domain	LOSS			Ref		
	1%	5%	10%	1%	5%	10%
ArXiv	0.1	0.2	0.5	0.2	0.7	1.7
Wikipedia	0.0	0.0	0.2	0.0	0.2	0.7
HackerNews	0.1	1.1	1.7	0.0	0.1	0.2

Table 6.12: FPR (%) on modified members (treated as non-members) when using a score threshold that achieves a 1, 5, or 10% FPR on the original member and non-member data for the ArXiv, Wikipedia, and HackerNews domains. Modified members are generated by prompting GPT4 to paraphrase member samples with significant lexical difference. LOSS and Reference-based attack reported.

results suggest two possibilities: 1. data does not leave much of an imprint, owing to characteristics of the pre-training process at scale, such as large datasets and single-epoch training, and 2. the similarity between in and out members (which we demonstrate via n-gram overlap), coupled with huge datasets, makes this distinction fuzzy.

While MIA performance could improve via stronger attacks [45], the second factor requires rethinking the membership game. The membership inference game needs to be extended for such generative models to better align with information leakage that adversaries and auditors may care about, such as user-level leakage [150] and PII [193]. While data extraction [44, 49, 104] takes a right step in this direction, the fraction of such data is relatively tiny, and the adversary has no control over what training data is regurgitated. In the meantime, special care should thus be taken to avoid unintentional distributional shifts while constructing non-members for MIA benchmark construction.

We highlight three research directions that we deem particularly important:

- Many existing definitions of memorization are specific to known attacks and not vice versa. For example, the definitions that involve prompting the model with a prefix are adapted to the auto-regressive training of models, and, while helpful in measuring a particular notion of memorization, do not cover adversaries that know a suffix of a document and want to extract a prefix.

- We did not find any research looking at data memorization from the supervised fine-tuning and RLHF phases, even though these phases seem to contain the most severe memorization risks. In particular, it would be important to know whether user-submitted prompts that are used in these training phases can be extracted from the model.

- We want to minimize the amount of undesirable information memorized by LLMs due to the risks outlined in this chapter. However, it is not clear which information LLMs need to memorize for good performance on downstream tasks, and the distinction between desirable and harmful learning is not well defined. This becomes an even more interesting question in light of the advent of LLMs enhanced with external knowledge, such as LLMs with Internet access.

Memorization in LLMs is an emerging area, and interactions between technical advances, demonstrated harms, and regulations and lawsuits will be necessary to achieve clarity on many of the issues raised in our work. Our experiments with membership inference demonstrate that we need better methods to measure even verbatim memorization, the most well-defined form of memorization. This will be a long and evolving process, but one in which the research community should play an essential role.

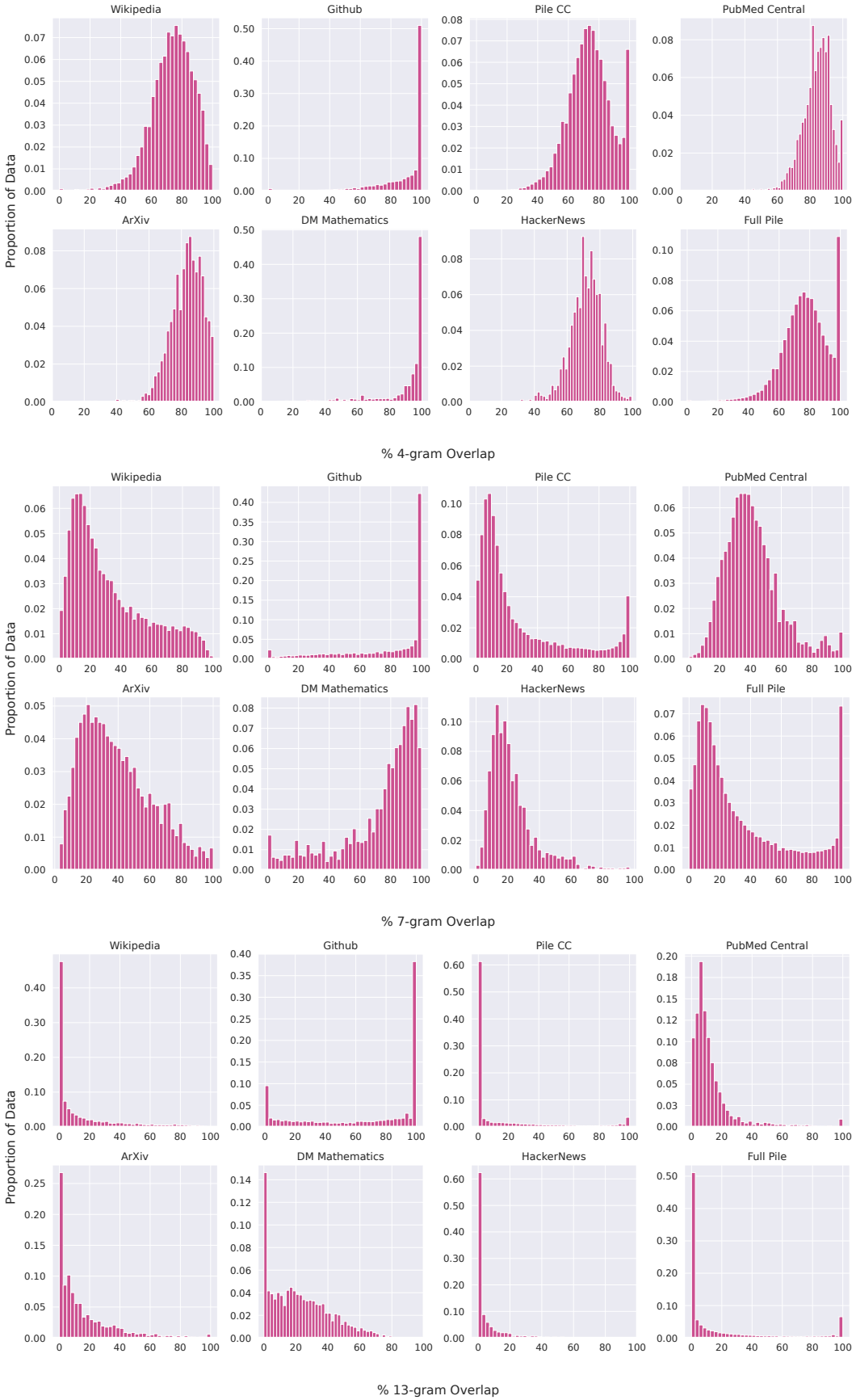


Figure 6.8: Distribution of n-gram overlap over all evaluation domains for $n = 4, 7, 13$.

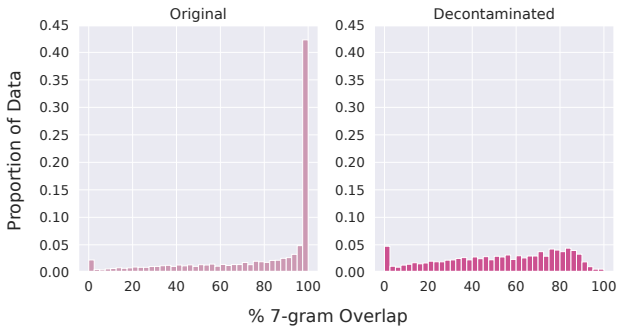


Figure 6.9: 7-gram overlap of GitHub non-member data before and after 13-gram decontamination at threshold $\leq 80\%$.

http://burmese.voanews.com/a/myanmar-ambassador-of-thailand-said-they-will-appeal-the-case-according-to-the-thai-law/3124176.html

လိပ်ကျွန်းအမှု အယူခံဝင်ဖို့ ပြင်ဆင်

ဗြိတိန်နိုင်ငံသား ကမ္ဘာလှည့်ခရီးသည်နှစ်ဦးကို ထိုင်းနိုင်ငံ ကော့တောင် အပန်းဖြေကျွန်းမှာ သတ်ခဲ့တယ်ဆိုတဲ့စွဲချက်နဲ့ သေဒဏ်ချခံထားရတဲ့ မြန်မာနိုင်ငံသားနှစ်ဦးရဲ့အမှုကို ထိုင်းနိုင်ငံရဲ့ တရားဥပဒေအတိုင်း ဆက်ပြီး အယူခံဝင်သွားနိုင်ဖို့အတွက် ထိုင်းနိုင်ငံ ရှေ့နေများကောင်စီနဲ့ ဘန်ကောက်မြို့က မြန်မာသံအမတ်ပါဝင်တဲ့ လိပ်ကျွန်းအမှုအတွက် လိုက်ပါဆောင်ရွက်နေတဲ့ မြန်မာသံရုံးအထူးကော်မတီတို့ တွေ့ဆုံခဲ့ပြီး သက်သေအထောက်အထား အချက်အလက် စုဆောင်းဖို့ အတွက်ပြင်ဆင်နေကြပါတယ်။ မေအေးအေးမာကသတင်းပေးပို့ထားပါတယ် ။

ဗြိတိန်နိုင်ငံသား ကမ္ဘာလှည့်ခရီးသည်နှစ်ဦးကို ထိုင်းနိုင်ငံ ကော့တောင် အပန်းဖြေကျွန်းမှာ သတ်ခဲ့တယ်ဆိုတဲ့ စွဲချက်နဲ့ သေဒဏ်ချခံထားရတဲ့ မြန်မာနိုင်ငံသား နှစ်ဦးရဲ့ အမှုနဲ့ ပတ်သက်လို့လိပ်ကျွန်းအမှုအတွက် ဖွဲ့စည်းထားတဲ့ မြန်မာသံရုံးအထူးကော်မတီဝင်တွေ နဲ့ ထိုင်းရှေ့နေများကောင်စီနဲ့ တိုင်ပင်ဆွေးနွေးမှုတွေလုပ်ခဲ့အပြီးမှာတော့ မြန်မာသံအမတ်ဦးဝင်းမောင်က ထိုင်းနိုင်ငံရဲ့ တရားဥပဒေအတိုင်းဆက်ပြီး အယူခံဝင်သွားမယ်လို့ ပြောဆိုသွားပါတယ်။

"ထိုင်းနိုင်ငံအစိုးရနဲ့ ထိုင်းနိုင်ငံရဲ့ ဥပဒေအရ ထိုင်းဝန်ကြီးချုပ်ကိုယ်တိုင် အယူခံဝင်နိုင်ကြောင်း လမ်းခင်းပေးတဲ့အတွက် ကျနော်တို့ ဆက်ပြီး အယူခံဝင်ပြီးတော့ ဒီကလေးတွေရဲ့မျှော်လင့်ချက်ကို ဆက်လက်ပြီး ဆောင်ရွက်သွားမှာဖြစ်ပါတယ်။"

ထိုင်းရှေ့နေများကောင်စီဟာ ရှေ့နေ ၁၁ ဦးပါအဖွဲ့နဲ့မြန်မာလူငယ် ၂ ဦးရဲ့အမှုကို အခမဲ့ ရှေ့နေ လိုက်ပေးခဲ့ပြီး တရားမျှတစွာ စီရင်နိုင်ဖို့အတွက် ဥပဒေကြောင်းအရ အကူညီပေးနေခဲ့တာပါ။ သူတို့အနေနဲ့ ခိုင်မာတဲ့ သက်သေ အထောက်အထား တွေလုံလောက်စွာမရခဲ့တာကြောင့် ဒီအမှုကို လိုက်ပါဆောင်ရွက်တဲ့အခါမှာ လိုအပ်ချက်တွေလည်းရှိနေခဲ့တယ်လို့ ဆိုပါတယ် ။ ဒီကနေ့မှာတော့ ထိုင်းရှေ့နေများကောင်စီက မြန်မာသံအမတ်နဲ့ လိပ်ကျွန်း အတွက် ဖွဲ့စည်းထားတဲ့ မြန်မာသံရုံးအထူးကော်မတီနဲ့ တွေ့ဆုံပြီး မြန်မာလူငယ် ၂ ဦး အတွက် အယူခံဆက်တက်နိုင်ဖို့ဥပဒေကြောင်းအရ ဆက်လက် လုပ်ဆောင်နိုင်တဲ့ အချက်တွေကို ဆွေးနွေးပြောဆိုခဲ့တယ်လို့ သံရုံးအထူးကော်မတီခေါင်းဆောင်ဖြစ်တဲ့ ဦးကျော်သောင်းက ပြောပါတယ်။

"ရှေ့နေများကောင်စီ ဥက္ကဋ္ဌက Dej Udom Krairit ပေါ့နော်။ သူကပြောတာက အချက် ၃ ချက်ပြောတယ်။ အယူခံကိစ္စမှာတဲ့ အမှုကို အစကနေအဆုံးအထိ ပြန်ပြီးတော့ စစ်ဆေးပေးဖို့ တောင်းဆိုနိုင်စရာကိစ္စရှိတယ်တဲ့။ အဲဒါတွေကတော့ ဘာလဲဆိုတော့ နံပါတ် ၁ သက်သေခံတွေဟာ လွဲမှားတယ်။ သက်သေတွေလည်းလွဲမှားတယ်ဆိုပြီးတော့ ကျနော်တို့ဘက်က တရားခံဘက်က ချေပနိုင်တဲ့ အချက်အလက်ရှိရင် တင်ပြလို့ရတယ်တဲ့။ နံပါတ် ၂ အချက်ကတော့ ပစ္စည်းသက်သေတွေဟာ အတုအပတွေဖြစ်နေတယ်။ အစစ်အမှန်မဟုတ်ဘူး။ နောက်ပြီးတော့ တင်ပြတဲ့အချက်အလက်တွေကလည်း အမှားအယွင်းတွေဖြစ်နေတယ်ဆိုလို့ရှိရင်လည်း ပြန်ပြီးတော့မှ အမှုကို အစအဆုံးပြန်ပြီးတော့ စစ်ဆေး လို့ရပါတယ်။ နံပါတ် ၃ အချက်က ထပ်ပြီးတော့ တင်ပြနိုင်တဲ့ သက်သေတွေ ပစ္စည်းတွေ၊ သက်သေက သက်ခံ၊သက်ရှိ သက်သေတွေ ရှိခဲ့ရင် ဒီအချက် ၃ ချက်က တစ်ချက်ချက် ကိုယ့်ဘက်က ထောက်ပြနိုင်စရာ ပြောနိုင်စရာရှိရင် ဒီအမှုကို အစအဆုံးထိပြန်ပြီးတော့ စစ်ဆေးခွင့်တောင်လို့ရပါတယ်တဲ့။ ဒီအချက် ၃ ချက်ထဲမှာ တခုခုတော့လည်းကောင်း ၊ အချက် ၂ ချက်သော်လည်းကောင်း၊ အချက် ၃ ချက်သော်လည်းကောင်း ၊ တခုခုရှိခဲ့လို့ရှိရင် မိသားစုဝင်တဦးဦးက ပြန်ပြီးတော့ ဒီအမှုကိစ္စကို ပြန်လည်စစ်ဆေးပေးဖို့အတွက် တောင်းဆိုလို့ရပါတယ်တဲ့။"

ရန်ကုန်အပါအဝင် နယ်မြို့တွေမှာတော့ မြန်မာနိုင်ငံသားလူငယ် ၂ ဦးကိုပြန်လွှတ်ပေးဖို့နဲ့ တရားမျှတမှုကို တောင်းဆိုတဲ့ ဆိုင်းဘုတ်တွေကို ကိုင်ဆောင်ဆန္ဒပြပွဲတွေက ဖြစ်နေဆဲပါ။

ဆန္ဒပြသူတွေဟာ ထိုင်းနိုင်ငံရဲ့ တရားစီရင်ရေးကိုလည်း ဝေဖန်ကြတာတွေလည်းရှိနေပါတယ်။ ဒီအတွက် ထိုင်းရှေ့နေများကောင်စီက ထိုင်းနိုင်ငံရဲ့ တရားစီရင်ရေး မှိုင်တည့်မျှတ မှန်ကန်တာကို ပြသဖို့အတွက် ဒီအမှုနဲ့ပတ်သက်လို့ ခိုင်မာတဲ့ သက်သေအထောက်အထားတွေပေးဖို့ တောင်းဆိုခဲ့တယ်လို့ ဦးကျော်သောင်းက ဆက်ပြောပါတယ်။

"ငါတို့တိုင်းပြည်ရဲ့တရားစီရင်ရေးမှိုင် ဘယ်လောက် တည့်မှတ် မှန်ကန်သလဲဆိုတာပြချင်တယ်။ နင်တို့တွေ ဒီကနေပြီးတော့ ဒီလိုခိုင်မာသေချာတဲ့ သက်သေအချက်အလက်တွေ ထွက်လာလိမ့်မယ်လို့ ငါတို့မျှော်လင့်တယ်။ အမြန်ဆုံး ငါတို့ကို ကူညီပေးပါလို့ မေတ္တာရပ်ခံပါတယ်။"

ပြီးခဲ့တဲ့ တနှစ်တာကာလအတွင်းမှာတော့ သက်သေထွက်ပေးနိုင်တဲ့ သူတွေရဲ့ နေထိုင်ရေးအလုပ်အကိုင်နဲ့ လုံခြုံရေးကိစ္စတွေကို မြန်မာသံရုံးနဲ့ လိပ်ကျွန်းအထူးကော်မတီတို့က တာဝန်ယူပေးနိုင်တာတွေ မရှိတဲ့အတွက် လိပ်ကျွန်းပေါ်မှာ အလုပ်လုပ်ကြတဲ့ မြန်မာနိုင်ငံသားသက်သေတွေ ရသင့်သလောက် မရရှိခဲ့ဘူးလို့ ဦးကျော်သောင်းကဆိုပါတယ်။

ဒီကနေ့မှာတော့ရန်ကုန်အပါအဝင် ထိုင်း-မြန်မာနယ်စပ်မြို့တွေဖြစ်တဲ့ မြဝတီမြို့နဲ့ ဘုရားသုံးဆူမြို့ တွေမှာတော့ ဆန္ဒပြမှုတွေ ဆက်ဖြစ်ပေါ်နေပါတယ်။

Figure 6.10: A sample GitHub non-member outlier captured by the $\leq 80\%$ 13-gram overlap threshold. This sample is from a language resource repository under Google, but is a clear outlier to the code-dominant GitHub domain

# Params	Temporal Wiki				
	LOSS	Ref	min-k	zlib	Ne
160M	.643	.602	**.648**	.541	.600
1.4B	.653	**.705**	.682	.572	.603
2.8B	.667	**.754**	.701	.593	.615
6.9B	.675	**.788**	.714	.601	.620
12B	.680	**.796**	.719	.607	.626

Table 6.7: AUC-ROC on the temporally shifted Wikipedia benchmark across various MIAs. Target models are the PYTHIA-DEDUP suite models. For each model, the highest score across MIAs is bolded.

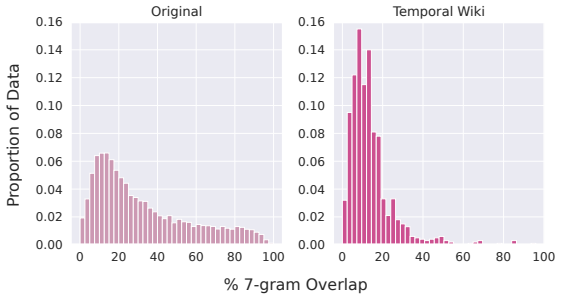


Figure 6.11: Distribution of 7-gram overlap for the original and temporally-shifted non-members.

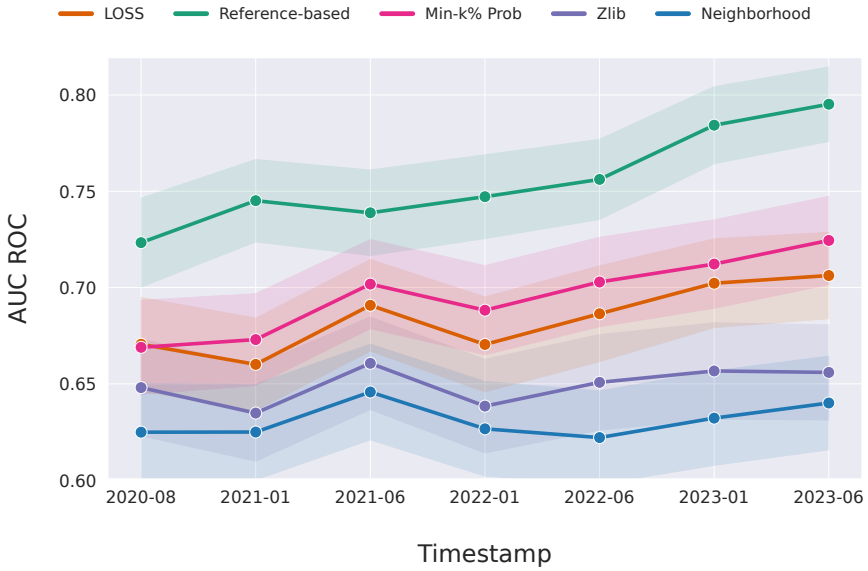


Figure 6.12: MIA performance across benchmarks where non-member data is selected from ArXiv preprints created during increasingly later months past the target model's knowledge cutoff. Timestamps are formatted as year-month. The target model is PYTHIA-DEDUP-12B. In general, **MIA performance increases as the temporal shift of non-members increases**

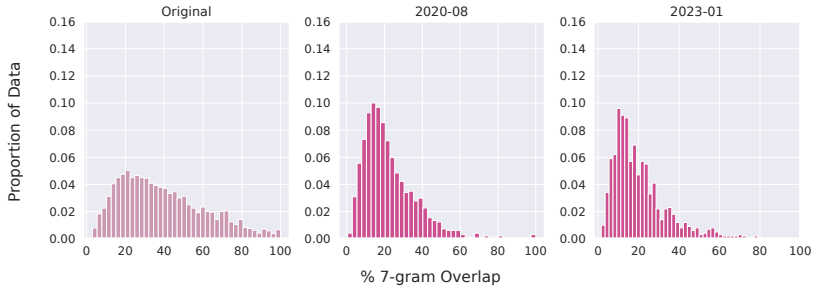


Figure 6.13: Distribution of n-gram overlap for non-member ArXiv preprints sampled from the months 2020-08 and 2023-06, respectively. We also plot the n-gram overlap distribution of the original Pile ArXiv non-members. Between the original non-members and both temporally shifted non-member sets, **the temporally shifted non-member n-gram overlap distributions are considerably more concentrated at lower % n-gram overlap**. The original non-members have an average 7-gram overlap of 39.3%, while non-members from months 2020-08 and 2023-06 have 7-gram overlap of 22.7% and 20.5%, respectively.

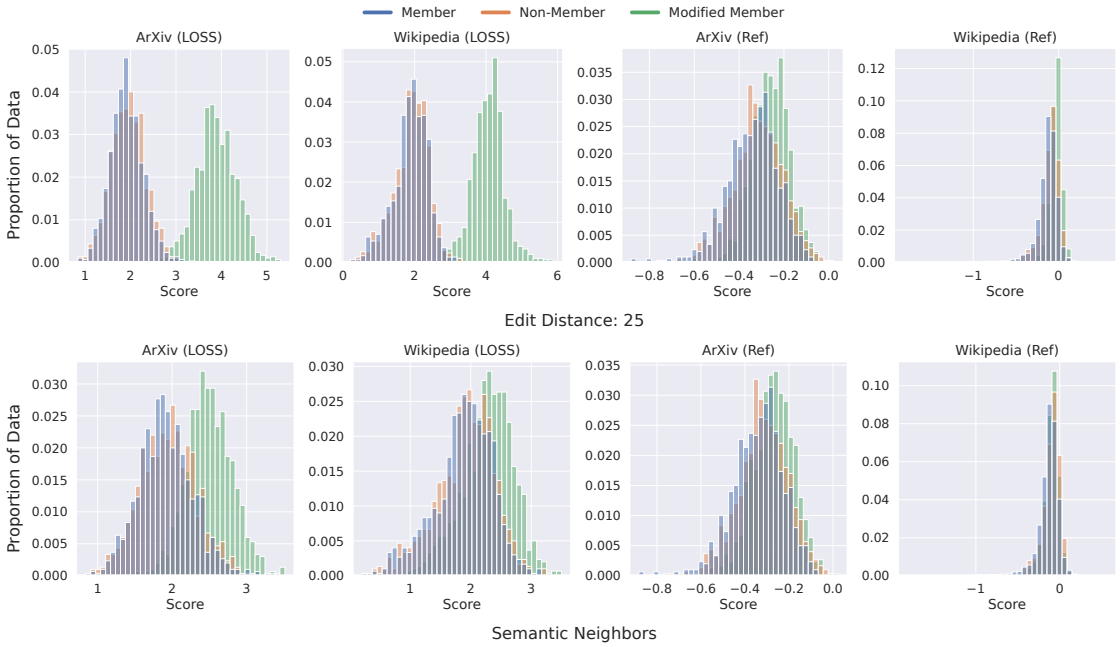


Figure 6.14: Distribution of scores for LOSS and Reference-based attacks for members, non-members, and modified members across ArXiv and Wikipedia domains. (Top) Modified members generated by random token replacement for edit distance 25. (Bottom) Modified members generated by replacing 5% of tokens with semantically similar tokens.

Figure 6.15: Distribution of scores for LOSS and Reference-based attacks for members, non-members, and GPT4-paraphrased (modified) members across ArXiv, Wikipedia, and HackerNews domains.

Chapter 7

Conclusion

User awareness around privacy and trustworthiness of machine learning models is growing, and the need for privacy-aware machine learning is more pressing than ever. Many organizations at the forefront of machine learning research and development have started to take privacy seriously, with robustness evaluations via red-teaming [12, 81, 241]. However, the current state of privacy evaluation is fragmented, with no clear consensus on the best practices. While membership inference is undoubtedly a valuable tool, as demonstrated in this thesis, more is needed to capture the full extent of privacy risks in machine learning models. No single inference threat model is the best choice in all circumstances because subtle differences in threat scenarios can lead to vastly different privacy evaluations. We also observe in orthogonal work a similar disconnect between threat models deemed necessary by academia and those that might be practically relevant [296]. Researchers must develop more comprehensive privacy evaluation tools to utilize complete model access to provide stronger empirical guarantees.

Overall, with this thesis, we call for a more comprehensive approach to privacy evaluation in machine learning: one that 1. takes into account the full spectrum of privacy risks and threats, 2. aligns well with how user data is used in practice, and 3. provides actionable insights for practitioners to mitigate these risks.

Record Membership $\neq$ Privacy While membership inference is well studied, we demonstrate how leakage can exist even at the level of the distribution from which training data is sampled, irrespective of the level of access granted to an adversary. Adversaries may not only seek to infer sensitive information at varying levels of detail, but also attempt to manipulate the model's training by poisoning the data or influencing the training algorithm to exacerbate the leakage. Strategies based on Differential Privacy can mitigate membership inference, and model trainers could employ algorithms like DP-SGD to defend against such adversaries, at least in passive cases. However, the absence of such defenses against other inference risks highlights the need for a thorough understanding of leakage, the threat models that adversaries may persist in, and the different

types of leakage feasible at various levels of model access. Recognizing the interconnections between different inference risks and exploring attacks within these inference models is a crucial initial step in understanding that record-level leakage does not provide the ideal privacy standard. While recent research on memorization in language models, such as semantic memorization measurement [223] and PII memorization [193] represents progress, there is still a demand for more comprehensive privacy auditing tools and benchmarks.

Aligning Privacy Evaluation with Real-World Data Collection User-level inference, an application of distribution inference, is not studied as extensively as membership inference. Only a few works [150, 174] focus on this area, despite being aligned better with real-world data collection. Users contribute multiple records, knowingly or voluntarily, for training large models. Yet, there is a lack of understanding about the extent of leakage pertaining to these users' data. Active adversaries can insert backdoors into models to increase leakage from downstream data used to fine-tune these poisoned models, all while evading detection by sophisticated auditors. Given the increasing reliance on pretrained models in machine learning, such adversaries are practical, with distribution sources like Github and Hugging Face [83]. These factors emphasize the need for increased attention to user-level leakage. Even when users contribute single records, there are misconceptions about the ideal way to empirically audit mitigation strategies. We illustrate how common settings under which models are trained do not align with the folklore around the optimality of black-box access for membership inference. Our updated theory demonstrates that parameter access is required for optimal membership inference.

Mitigating Privacy Risks Understanding inference leakage in depth is essential for advancing research, but it is equally crucial to provide actionable insights to model trainers. Some of our findings are in line with existing literature, such as the recommendation to avoid over-parameterized models [20] and to use pre-trained models only from reputable sources [232]. However, other findings suggest that the situation is more complex. For instance, we find a potential conflict between membership inference and distribution inference—improved generalization reduces membership inference risk but, depending on the property of interest, can increase leakage at the distribution level. Furthermore, our research indicates that for user-level inference, unlike in the standard training scenario [56], there is high leakage when only a few users are present in the training data, but leakage rises again as users contribute more data. In cases like these, empirical auditing tools such as our IHA for membership inference can be valuable. These tools enable model trainers to assess modifications to models and training algorithms, providing a better understanding of leakage without relying on data-intensive alternatives that require training reference models.

Bibliography

[1] Martín Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *ACM Conference on Computer and Communications Security (CCS)*, 2016.

[2] Hervé Abdi and Lynne J Williams. Principal component analysis. *Wiley interdisciplinary reviews: computational statistics*, 2010.

[3] John M Abowd, Robert Ashmead, Ryan Cumings-Menon, Simson Garfinkel, Micah Heineck, Christine Heiss, Robert Johns, Daniel Kifer, Philip Leclerc, Ashwin Machanavajjhala, et al. The 2020 Census Disclosure Avoidance System TopDown algorithm. *arXiv preprint arXiv:2204.08986*, 2022.

[4] Michael Aerni, Jie Zhang, and Florian Tramèr. Evaluations of machine learning privacy defenses are misleading. *arXiv:2404.17399*, 2024.

[5] Naman Agarwal, Brian Bullins, and Elad Hazan. Second-order stochastic optimization for machine learning in linear time. *Journal of Machine Learning Research*, 2017.

[6] Together AI. Redpajama: an open dataset for training large language models, 2023. URL https://github.com/togethercomputer/RedPajama-Data.

[7] Alan Akbik, Tanja Bergmann, Duncan Blythe, Kashif Rasul, Stefan Schweter, and Roland Vollgraf. FLAIR: An easy-to-use framework for state-of-the-art NLP. In *North American Chapter of the Association for Computational Linguistics (demonstrations)*, 2019.

[8] MAH Akhand, Ijaj Sayim, Shuvendu Roy, N Siddique, et al. Human age prediction from facial image using transfer learning in deep convolutional neural networks. In *International Joint Conference on Computational Intelligence*, 2020.

[9] Amazon. Participation agreement. https://www.mturk.com/participation-agreement, 2023. Accessed: 2023-09-26.

[10] Alex Andonian, Quentin Anthony, Stella Biderman, Sid Black, Preetham Gali, Leo Gao, Eric Hallahan, Josh Levy-Kramer, Connor Leahy, Lucas Nestler, Kip Parker, Michael Pieler, Jason Phang, Shivanshu Purohit, Hailey Schoelkopf, Dashiell Stander, Tri Songz, Curt Tigges, Benjamin Thérien, Phil Wang, and Samuel Weinbach. Gpt-neox: Large scale autoregressive language modeling in pytorch, 2023. URL https://www.github.com/eleutherai/gpt-neox.

[11] Jacob Andreas. Language models as agent models. In *Conference on Empirical Methods in Natural Language Processing*, 2022.

[12] Anthropic. Challenges in red teaming ai systems, 2024. URL https://www.anthropic.com/news/challenges-in-red-teaming-ai-systems. Accessed: 2024-07-03.

[13] Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. Invariant risk minimization. *arXiv preprint arXiv:1907.02893*, 2019.

[14] Sanjeev Arora and Boaz Barak. *Computational Complexity: A Modern Approach*. Cambridge University Press, 2009.

[15] Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael Rabbat, Yann LeCun, and Nicolas Ballas. Self-supervised learning from images with a joint-embedding predictive architecture. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[16] Giuseppe Ateniese, Luigi V. Mancini, Angelo Spognardi, Antonio Villani, Domenico Vitali, and Giovanni Felici. Hacking smart machines with smarter ones: How to extract meaningful data from machine learning classifiers. *International Journal of Security and Networks*, 2015.

[17] Giuseppe Ateniese, Luigi V. Mancini, Angelo Spognardi, Antonio Villani, Domenico Vitali, and Giovanni Felici. Hacking smart machines with smarter ones: How to extract meaningful data from machine learning classifiers. *International Journal of Security and Networks*, 2015.

[18] Achraf Azize and Debabrota Basu. How much does each datapoint leak your privacy? quantifying the per-datum membership leakage. *arXiv:2402.10065*, 2024.

[19] Borja Balle, Giovanni Cherubin, and Jamie Hayes. Reconstructing training data with informed adversaries. In *IEEE Symposium on Security and Privacy*, 2022.

[20] Teodora Baluta, Shiqi Shen, S Hitarth, Shruti Tople, and Prateek Saxena. Membership inference attacks and generalization: A causal perspective. In *ACM Conference on Computer and Communications Security (CCS)*, 2022.

[21] Georgios Barlas and Efstathios Stamatatos. Cross-domain authorship attribution using pre-trained language models. In *Artificial Intelligence Applications and Innovations: 16th IFIP WG 12.5 International Conference*, 2020.

[22] Gilles Barthe, Benjamin Grégoire, and Santiago Zanella-Béguelin. Formal certification of code-based cryptographic proofs. In *36th annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL)*. ACM, 2009.

[23] Stephen D Bay, Dennis Kibler, Michael J Pazzani, and Padhraic Smyth. The uci kdd archive of large data sets for data mining research and experimentation. *ACM SIGKDD Explorations Newsletter*, 2000.

[24] Deniz Bayazit, Negar Foroutan, Zeming Chen, Gail Weiss, and Antoine Bosselut. Discovering knowledge-critical subnetworks in pretrained language models, 2023.

[25] M. Bellare, A. Desai, E. Jokipii, and P. Rogaway. A concrete security treatment of symmetric encryption. In *38th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE, 1997.

[26] Mihir Bellare and Phillip Rogaway. The security of triple encryption and a framework for code-based game-playing proofs. In *Advances in Cryptology – EUROCRYPT*, 2006.

[27] Martin Bertran, Shuai Tang, Aaron Roth, Michael Kearns, Jamie H Morgenstern, and Steven Z Wu. Scalable membership inference attacks via quantile regression. *Advances in Neural Information Processing Systems*, 2024.

[28] Janek Bevendorff, Berta Chulvi, Elisabetta Fersini, Annina Heini, Mike Kestemont, Krzysztof Kredens, Maximilian Mayerl, Reynier Ortega-Bueno, Piotr Pęzik, Martin Potthast, et al. Overview of PAN 2022: Authorship verification, profiling irony and stereotype spreaders, and style change detection. In *International Conference of the Cross-Language Evaluation Forum for European Languages*, 2022.

[29] Stella Biderman, USVSN Sai Prashanth, Lintang Sutawika, Hailey Schoelkopf, Quentin Anthony, Shivanshu Purohit, and Edward Raf. Emergent and predictable memorization in large language models. In *Advances in Neural Information Processing Systems*, 2023.

[30] Stella Biderman, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, et al. Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning (ICML)*, 2023.

[31] Sid Black, Leo Gao, Phil Wang, Connor Leahy, and Stella Biderman. GPT-Neo: Large scale autoregressive language modeling with Mesh-Tensorflow, 2021.

[32] Bruno Blanchet. Computationally sound mechanized proofs of correspondence assertions. In *IEEE Computer Security Foundations Symposium (CSF)*, 2007.

[33] Bruno Blanchet. Mechanizing game-based proofs of security protocols. *Software Safety and Security*, 33, 2012.

[34] Franziska Boenisch, Adam Dziedzic, Roei Schuster, Ali Shahin Shamsabadi, Ilia Shumailov, and Nicolas Papernot. When the curious abandon honesty: Federated learning is not private. In *2023 IEEE 8th European Symposium on Security and Privacy (EuroS&P)*. IEEE, 2023.

[35] Olivier Bousquet and André Elisseeff. Stability and generalization. *Journal of Machine Learning Research*, 2002.

[36] Hannah Brown, Katherine Lee, Fatemehsadat Mireshghallah, Reza Shokri, and Florian Tramèr. What does it mean for a language model to preserve privacy? In *ACM Conference on Fairness, Accountability, and Transparency (FAccT)*, 2022.

[37] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 2020.

[38] Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, et al. Sparks of artificial general intelligence: Early experiments with GPT-4. *arXiv preprint arXiv:2303.12712*, 2023.

[39] Sebastian Caldas, Sai Meher Karthik Duddu, Peter Wu, Tian Li, Jakub Konečnỳ, H Brendan McMahan, Virginia Smith, and Ameet Talwalkar. Leaf: A benchmark for federated settings. *arXiv preprint arXiv:1812.01097*, 2018.

[40] Qiong Cao, Li Shen, Weidi Xie, Omkar M Parkhi, and Andrew Zisserman. Vggface2: A dataset for recognising faces across pose and age. In *IEEE International Conference on Automatic Face & Gesture Recognition*, 2018.

[41] Nicholas Carlini, Chang Liu, Úlfar Erlingsson, Jernej Kos, and Dawn Song. The secret sharer: Evaluating and testing unintended memorization in neural networks. In *USENIX Security Symposium*, 2019.

[42] Nicholas Carlini, Florian Tramèr, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Úlfar Erlingsson, Alina Oprea, and Colin Raffel. Extracting training data from large language models. In *USENIX Security Symposium*, 2021.

[43] Nicholas Carlini, Steve Chien, Milad Nasr, Shuang Song, Andreas Terzis, and Florian Tramer. Membership inference attacks from first principles. In *IEEE Symposium on Security and Privacy*, 2022.

[44] Nicholas Carlini, Daphne Ippolito, Matthew Jagielski, Katherine Lee, Florian Tramer, and Chiyuan Zhang. Quantifying memorization across neural language models. In *International Conference on Learning Representations (ICLR)*, 2022.

[45] Stephen Casper, Carson Ezell, Charlotte Siegmann, Noam Kolt, Taylor Lynn Curtis, Benjamin Bucknall, Andreas Haupt, Kevin Wei, Jérémy Scheurer, Marius Hobbhahn, et al. Black-box access is insufficient for rigorous ai audits. In *ACM Conference on Fairness, Accountability, and Transparency (FAccT)*, 2024.

[46] Google Search Central. Overview of Google crawlers and fetchers (user agents), 2023. https://developers.google.com/search/docs/crawling-indexing/overview-google-crawlers.

[47] Shuvam Chakraborty, Burak Uzkent, Kumar Ayush, Kumar Tanmay, Evan Sheehan, and Stefano Ermon. Efficient conditional pre-training for transfer learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[48] Hongyan Chang and Reza Shokri. On the privacy risks of algorithmic fairness. In *IEEE European Symposium on Security and Privacy (EuroS&P)*, 2021.

[49] Kent Chang, Mackenzie Cramer, Sandeep Soni, and David Bamman. Speak, memory: An archaeology of books known to chatgpt/gpt-4. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, 2023.

[50] Konstantinos Chatzikokolakis, Giovanni Cherubin, Catuscia Palamidessi, and Carmela Troncoso. The Bayes security measure. *arXiv preprint arXiv:2011.03396*, 2020.

[51] Harsh Chaudhari, John Abascal, Alina Oprea, Matthew Jagielski, Florian Tramèr, and Jonathan Ullman. SNAP: Efficient extraction of private properties with poisoning. In *IEEE Symposium on Security and Privacy*, 2023.

[52] Harsh Chaudhari, Giorgio Severi, Alina Oprea, and Jonathan Ullman. Chameleon: Increasing label-only membership leakage with adaptive poisoning. In *International Conference on Learning Representations (ICLR)*, 2024.

[53] Dingfan Chen, Ning Yu, Yang Zhang, and Mario Fritz. GAN-leaks: A taxonomy of membership inference attacks against generative models. In *ACM Conference on Computer and Communications Security (CCS)*, 2020.

[54] Dingfan Chen, Ning Yu, and Mario Fritz. Relaxloss: Defending membership inference attacks without losing utility. In *International Conference on Learning Representations (ICLR)*, 2022.

[55] Michelle Chen and Olga Ohrimenko. Protecting global properties of datasets with distribution privacy mechanisms. In *International Conference on Artificial Intelligence and Statistics*, 2022.

[56] Min Chen, Zhikun Zhang, Tianhao Wang, Michael Backes, and Yang Zhang. Face-auditor: Data auditing in facial recognition systems. In *USENIX Security Symposium*, 2023.

[57] Christopher A. Choquette Choo, Florian Tramèr, Nicholas Carlini, and Nicolas Papernot. Label-only membership inference attacks. In *International Conference on Machine Learning (ICML)*, 2021.

[58] Aloni Cohen and Kobbi Nissim. Towards formalizing the GDPR's notion of singling out. *Proc. Natl. Acad. Sci. USA*, 117(15), 2020.

[59] Gilad Cohen and Raja Giryes. Membership inference attack using self influence functions. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2024.

[60] Rachel Cummings, Damien Desfontaines, David Evans, Roxana Geambasu, Matthew Jagielski, Yangsibo Huang, Peter Kairouz, Gautam Kamath, Sewoong Oh, Olga Ohrimenko, and others. Challenges towards the Next Frontier in Privacy. *arXiv:2304.06929*, 2023.

[61] Rachel Cummings, Damien Desfontaines, David Evans, Roxana Geambasu, Yangsibo Huang, Ryan Rogers, Matthew Jagielski, Peter Kairouz, Gautam Kamath, Sewoong Oh, Olga Ohrimenko, Nicolas Papernot, Milan Shen, Shuang Song, Weijie Su, Andreas Terzis, Abhradeep Thakurta, Sergei Vassilvitskii, Yu-Xiang Wang, Li Xiong, Sergey Yekhanin, Da Yu, Huanyu Zhang, and Wanrong Zhang. Advancing differential privacy: Where we are now and future directions for real-world deployment. *Harvard Data Science Review*, Jan 2024. URL https://doi.org/10.1162/99608f92.d3197524.

[62] Emiliano De Cristofaro. A critical overview of privacy in machine learning. *IEEE Symposium on Security and Privacy*, 2021.

[63] Daniel DeAlcala, Aythami Morales, Gonzalo Mancera, Julian Fierrez, Ruben Tolosana, and Javier Ortega-Garcia. Is my Data in your AI Model? Membership Inference Test with Application to Face Images. *arXiv:2402.09225*, 2024.

[64] Edoardo Debenedetti, Giorgio Severi, Nicholas Carlini, Christopher A Choquette-Choo, Matthew Jagielski, Milad Nasr, Eric Wallace, and Florian Tramèr. Privacy side channels in machine learning systems. *arXiv preprint arXiv:2309.05610*, 2023.

[65] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[66] Li Deng. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 2012.

[67] Damien Desfontaines and Balázs Pejó. SoK: Differential privacies. *Privacy Enhancing Technologies Symposium (PETS)*, 2020.

[68] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019.

[69] Bhuwan Dhingra, Manaal Faruqui, Ankur Parikh, Ming-Wei Chang, Dipanjan Das, and William W Cohen. Handling divergent reference texts when evaluating table-to-text generation. In *57th Annual Meeting of the Association for Computational Linguistics*, 2019.

[70] Qingxiu Dong, Jingjing Xu, Lingpeng Kong, Zhifang Sui, and Lei Li. Statistical knowledge assessment for generative language models. *arXiv preprint arXiv:2305.10519*, 2023.

[71] Fadi Dornaika, Ignacio Arganda-Carreras, and C Belver. Age estimation in facial images through transfer learning. *Machine Vision and Applications*, 2019.

[72] Vasisht Duddu, Anudeep Das, Nora Khayata, Hossein Yalame, Thomas Schneider, and N Asokan. Attesting distributional properties of training data for machine learning. *arXiv preprint arXiv:2308.09552*, 2023.

[73] Cynthia Dwork. Differential privacy. In *Automata, Languages and Programming*, 2006.

[74] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Proceedings of the Third Conference on Theory of Cryptography*, 2006.

[75] Nouha Dziri, Andrea Madotto, Osmar Zaïane, and Avishek Joey Bose. Neural Path Hunter: Reducing hallucination in dialogue systems via path grounding. In *Conference on Empirical Methods in Natural Language Processing*, 2021.

[76] El-Mahdi El-Mhamdi, Sadegh Farhadkhani, Rachid Guerraoui, Nirupam Gupta, Lê Nguyên Hoang, Rafael Pinot, and John Stephan. On the impossible safety of large AI models. *arXiv preprint arXiv:2209.15259*, 2022.

[77] Ronen Eldan and Mark Russinovich. Who's Harry Potter? Approximate unlearning in LLMs, 2023.

[78] Niva Elkin-Koren, Uri Hacohen, Roi Livni, and Shay Moran. Can copyright be reduced to privacy? *arXiv preprint arXiv:2305.14822*, 2023.

[79] European Union. Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation). *Official Journal of the European Union*, 2016.

[80] Alexandre Evfimievski, Johannes Gehrke, and Ramakrishnan Srikant. Limiting privacy breaches in privacy preserving data mining. In *Proceedings of the twenty-second ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, 2003.

[81] Daniel Fabian. Google's ai red team: the ethical hackers making ai safer, 2023. URL https://blog.google/technology/safety-security/googles-ai-red-team-the-ethical-hackers-making-ai-safer/. Accessed: 2024-07-03.

[82] Maël Fabien, Esaú Villatoro-Tello, Petr Motlicek, and Shantipriya Parida. BertAA: BERT fine-tuning for authorship attribution. In *17th International Conference on Natural Language Processing*, 2020.

[83] Hugging Face. The model hub, July 2021. URL https://huggingface.co/docs/hub/en/models-the-hub. Accessed July 1, 2024 [Online].

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

[85] Vitaly Feldman and Chiyuan Zhang. What neural networks memorize and why: Discovering the long tail via influence estimation. In *Advances in Neural Information Processing Systems*, 2020.

[86] Vitaly Feldman, Ilya Mironov, Kunal Talwar, and Abhradeep Thakurta. Privacy amplification by iteration. In *IEEE 59th Annual Symposium on Foundations of Computer Science*, 2018.

[87] Shanglun Feng and Florian Tramèr. Privacy backdoors: Stealing data with corrupted pretrained models. *arXiv preprint arXiv:2404.00473*, 2024.

[88] Matthias Feurer, Aaron Klein, Katharina Eggensperger, Jost Springenberg, Manuel Blum, and Frank Hutter. Efficient and robust automated machine learning. In *Advances in Neural Information Processing Systems*, 2015.

[89] Center for Applied Internet Data Analysis. The CAIDA UCSD Anonymized Internet Traces. https://www.caida.org/data/passive/passive_dataset.xml, 2018.

[90] Liam Fowl, Jonas Geiping, Wojtek Czaja, Micah Goldblum, and Tom Goldstein. Robbing the fed: Directly obtaining private data in federated learning with modified models. In *International Conference on Learning Representations (ICLR)*, 2021.

[91] Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. Model inversion attacks that exploit confidence information and basic countermeasures. In *ACM Conference on Computer and Communications Security (CCS)*, 2015.

[92] Matthew Fredrikson, Eric Lantz, Somesh Jha, Simon Lin, David Page, and Thomas Ristenpart. Privacy in pharmacogenetics: An end-to-end case study of personalized warfarin dosing. In *USENIX Security Symposium*, 2014.

[93] Daniel Fried, Armen Aghajanyan, Jessy Lin, Sida Wang, Eric Wallace, Freda Shi, Ruiqi Zhong, Wen-tau Yih, Luke Zettlemoyer, and Mike Lewis. InCoder: A generative model for code infilling and synthesis. In *International Conference on Learning Representations (ICLR)*, 2023.

[94] Wenjie Fu, Huandong Wang, Chen Gao, Guanghua Liu, Yong Li, and Tao Jiang. Practical membership inference attacks against fine-tuned large language models via self-prompt calibration, 2023.

[95] Karan Ganju, Qi Wang, Wei Yang, Carl A Gunter, and Nikita Borisov. Property inference attacks on fully connected neural networks using permutation invariant representations. In *ACM SIGSAC Conference on Computer and Communications Security*, 2018.

[96] Ji Gao, Sanjam Garg, Mohammad Mahmoody, and Prashant Nalini Vasudevan. Deletion inference, reconstruction, and compliance in machine (un)learning. In *Privacy Enhancing Technologies Symposium (PETS)*, 2022.

[97] Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, et al. The Pile: An 800GB dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*, 2020.

[98] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *International Conference on Artificial Intelligence and Statistics*, 2010.

[99] Shafi Goldwasser and Silvio Micali. Probabilistic encryption & how to play mental poker keeping secret all partial information. In *14th Annual ACM Symposium on Theory of Computing (STOC)*. ACM, 1982.

[100] Divya Gopinath, Hayes Converse, Corina Pasareanu, and Ankur Taly. Property inference for deep neural networks. In *IEEE/ACM International Conference on Automated Software Engineering*, 2019.

[101] Dirk Groeneveld, Chris Ha, and Ian Magnusson. BFF: The big friendly filter, 2023. URL https://github.com/allenai/bff.

[102] Dirk Groeneveld, Iz Beltagy, Pete Walsh, Akshita Bhagia, Rodney Kinney, Oyvind Tafjord, Ananya Harsh Jha, Hamish Ivison, Ian Magnusson, Yizhong Wang, Shane Arora, David Atkinson, Russell Authur, Khyathi Chandu, Arman Cohan, Jennifer Dumas, Yanai Elazar, Yuling Gu, Jack Hessel, Tushar Khot, William Merrill, Jacob Morrison, Niklas Muennighoff, Aakanksha Naik, Crystal Nam, Matthew E. Peters, Valentina Pyatkin, Abhilasha Ravichander, Dustin Schwenk, Saurabh Shah, Will Smith, Nishant Subramani, Mitchell Wortsman, Pradeep Dasigi, Nathan Lambert, Kyle Richardson, Jesse Dodge, Kyle Lo, Luca Soldaini, Noah A. Smith, and Hannaneh Hajishirzi. Olmo: Accelerating the science of language models. *Preprint*, 2024.

[103] Kathrin Grosse, Thomas A Trost, Marius Mosbach, and Michael Backes. Adversarial initialization-when your network performs the way I want. *arXiv preprint arxiv:1902.03020*, 2019.

[104] Michael M. Grynbaum and Ryan Mac. The times sues openai and microsoft over a.i. use of copyrighted work. https://www.nytimes.com/2023/12/27/business/media/new-york-times-open-ai-microsoft-lawsuit.html, 2023. Accessed: 2023-12-30.

[105] Xin Guo, Luisa Polania, and Kenneth Barner. Smile detection in the wild based on transfer learning. In *IEEE International Conference on Automatic Face & Gesture Recognition*, 2018.

[106] Safwan S Halabi, Luciano M Prevedello, Jayashree Kalpathy-Cramer, Artem B Mamonov, Alexander Bilbily, Mark Cicero, Ian Pan, Lucas Araújo Pereira, Rafael Teixeira Sousa, Nitamar Abdala, et al. The RSNA Pediatric Bone Age Machine Learning Challenge. *Radiology*, 2019.

[107] Moritz Hardt, Ben Recht, and Yoram Singer. Train faster, generalize better: Stability of stochastic gradient descent. In *International Conference on Machine Learning (ICML)*, 2016.

[108] John Hartley and Sotirios A Tsaftaris. Measuring unintended memorisation of unique private features in neural networks. *arXiv preprint arXiv:2202.08099*, 2022.

[109] Valentin Hartmann, Léo Meynent, Maxime Peyrard, Dimitrios Dimitriadis, Shruti Tople, and Robert West. Distribution inference risks: Identifying and mitigating sources of leakage. In *IEEE Conference on Secure and Trustworthy Machine Learning (SaTML)*, 2023.

[110] Michael Hay, Chao Li, Gerome Miklau, and David Jensen. Accurate estimation of the degree distribution of private networks. In *IEEE International Conference on Data Mining*, 2009.

[111] Jonathan Hayase, Weihao Kong, Raghav Somani, and Sewoong Oh. SPECTRE: Defending against backdoor attacks using robust statistics. In *International Conference on Machine Learning (ICML)*, 2021.

[112] Jamie Hayes, Luca Melis, George Danezis, and Emiliano De Cristofaro. LOGAN: Evaluating privacy leakage of generative models using generative adversarial networks. *Privacy Enhancing Technologies Symposium (PETS)*, 2019.

[113] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *IEEE International Conference on Computer Vision (ICCV)*, 2015.

[114] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[115] Yang He, Shadi Rahimian, Bernt Schiele1, and Mario Fritz. Segmentations-Leak: Membership inference attacks and defenses in semantic image segmentation. In *European Conference on Computer Vision (ECCV)*. Springer, 2020.

[116] Peter Henderson, Xuechen Li, Dan Jurafsky, Tatsunori Hashimoto, Mark A Lemley, and Percy Liang. Foundation models and fair use. *Journal of Machine Learning Research*, 2023.

[117] Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations (ICLR)*, 2021.

[118] Benjamin Hilprecht, Martin Härterich, and Daniel Bernau. Monte Carlo and reconstruction membership inference attacks against generative models. *Privacy Enhancing Technologies Symposium (PETS)*, 2019.

[119] Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. An empirical analysis of compute-optimal large language model training. In *Advances in Neural Information Processing Systems*, 2022.

[120] Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. In *International Conference on Learning Representations (ICLR)*, 2020.

[121] Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. spaCy: Industrial-strength natural language processing in Python, 2020.

[122] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely Connected Convolutional Networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[123] Jie Huang and Kevin Chen-Chuan Chang. Towards reasoning in large language models: A survey. In *Findings of the Association for Computational Linguistics: ACL*, 2023.

[124] Bo Hui, Yuchen Yang, Haolin Yuan, Philippe Burlina, Neil Zhenqiang Gong, and Yinzhi Cao. Practical blind membership inference attack via differential comparisons. In *Network and Distributed Systems Security Symposium*, 2021.

[125] Thomas Humphries, Simon Oya, Lindsey Tulloch, Matthew Rafuse, Ian Goldberg, Urs Hengartner, and Florian Kerschbaum. Investigating membership inference attacks under data dependencies. In *IEEE Computer Security Foundations Symposium (CSF)*, 2023.

[126] Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Suchin Gururangan, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. Editing models with task arithmetic. In *International Conference on Learning Representations (ICLR)*, 2023.

[127] Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Adversarial examples are not bugs, they are features. *Advances in Neural Information Processing Systems*, 2019.

[128] Daphne Ippolito and Yun William Yu. DoNotTrain: A metadata standard for indicating consent for machine learning. In *ICML Workshop on Generative AI and Law*, 2023.

[129] Daphne Ippolito, Florian Tramer, Milad Nasr, Chiyuan Zhang, Matthew Jagielski, Katherine Lee, Christopher Choquette Choo, and Nicholas Carlini. Preventing generation of verbatim memorization in language models gives a false sense of privacy. In *Proceedings of the 16th International Natural Language Generation Conference*, 2023.

[130] Matthew Jagielski, Jonathan Ullman, and Alina Oprea. Auditing differentially private machine learning: How private is private SGD? In *Advances in Neural Information Processing Systems*, 2020.

[131] Matthew Jagielski, Om Thakkar, Florian Tramèr, Daphne Ippolito, Katherine Lee, Nicholas Carlini, Eric Wallace, Shuang Song, Abhradeep Guha Thakurta, Nicolas Papernot, and Chiyuan Zhang. Measuring forgetting of memorized training examples. In *International Conference on Learning Representations (ICLR)*, 2023.

[132] Anil K Jain, M Narasimha Murty, and Patrick J Flynn. Data clustering: A review. *ACM computing surveys (CSUR)*, 1999.

[133] Hirsh Jain. Applications of Pinsker's inequality. Harvard Course Notes, http://people.seas.harvard.edu/~madhusudan/courses/Spring2016/scribe/lect07.pdf, 2016.

[134] Neel Jain, Khalid Saifullah, Yuxin Wen, John Kirchenbauer, Manli Shu, Aniruddha Saha, Micah Goldblum, Jonas Geiping, and Tom Goldstein. Bring your own data! Self-supervised evaluation for large language models. *arXiv preprint arXiv:2306.13651*, 2023.

[135] Joanna Jaworek-Korjakowska, Pawel Kleczek, and Marek Gorgon. Melanoma thickness prediction based on convolutional neural network with vgg-19 model transfer learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2019.

[136] Bargav Jayaraman. Texas-100x dataset. https://github.com/bargavj/Texas-100X, 2022.

[137] Bargav Jayaraman and David Evans. Are attribute inference attacks just imputation? In *ACM Conference on Computer and Communications Security (CCS)*, 2022.

[138] Bargav Jayaraman, Lingxiao Wang, Katherine Knipmeyer, Quanquan Gu, and David Evans. Revisiting membership inference under realistic assumptions. *Privacy Enhancing Technologies Symposium (PETS)*, 2021.

[139] Marija Jegorova, Chaitanya Kaul, Charlie Mayor, Alison Q O'Neil, Alexander Weir, Roderick Murray-Smith, and Sotirios A Tsaftaris. Survey: Leakage and privacy at inference time. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

[140] Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 2023.

[141] Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. Mistral 7B. *arXiv preprint arXiv:2310.06825*, 2023.

[142] Zhengbao Jiang, Frank F Xu, Jun Araki, and Graham Neubig. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 2020.

[143] Di Jin, Zhijing Jin, Zhiting Hu, Olga Vechtomova, and Rada Mihalcea. Deep learning for text style transfer: A survey. *Computational Linguistics*, 2022.

[144] Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Findings of the Association for Computational Linguistics (ACL)*, 2017.

[145] Martin Josifoski, Lars Klein, Maxime Peyrard, Yifei Li, Saibo Geng, Julian Paul Schnitzler, Yuxing Yao, Jiheng Wei, Debjit Paul, and Robert West. Flows: Building blocks of reasoning and collaborating AI. *arXiv preprint arXiv:2308.01285*, 2023.

[146] Martin Josifoski, Maxime Peyrard, Frano Rajic, Jiheng Wei, Debjit Paul, Valentin Hartmann, Barun Patra, Vishrav Chaudhary, Emre Kıcıman, Boi Faltings, and Robert West. Language model decoding as likelihood-utility alignment. In *Conference of the European Chapter of the Association for Computational Linguistics (Findings)*, 2023.

[147] Marc Juarez, Samuel Yeom, and Matt Fredrikson. Black-box audits for group distribution shifts. *arXiv preprint arXiv:2209.03620*, 2022.

[148] 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*, 2021.

[149] Nikhil Kandpal, Eric Wallace, and Colin Raffel. Deduplicating training data mitigates privacy risks in language models. In *International Conference on Machine Learning (ICML)*, 2022.

[150] Nikhil Kandpal, Krishna Pillutla, Alina Oprea, Peter Kairouz, Christopher A Choquette-Choo, and Zheng Xu. User inference attacks on large language models. *arXiv preprint arXiv:2310.09266*, 2023.

[151] Shiva Prasad Kasiviswanathan, Homin K. Lee, Kobbi Nissim, Sofya Raskhodnikova, and Adam D. Smith. What can we learn privately? *CoRR*, abs/0803.0924, 2008. URL http://arxiv.org/abs/0803.0924.

[152] Yusuke Kawamoto and Takao Murakami. Local obfuscation mechanisms for hiding probability distributions. In *European Symposium on Research in Computer Security*, 2019.

[153] Mishaal Kazmi, Hadrien Lautraite, Alireza Akbari, Mauricio Soroco, Qiaoyue Tang, Tao Wang, Sébastien Gambs, and Mathias Lécuyer. PANORAMIA: Privacy Auditing of Machine Learning Models without Retraining. *arXiv:2402.09477*, 2024.

[154] Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, 2019.

[155] Daniel Kifer and Ashwin Machanavajjhala. Pufferfish: A framework for mathematical privacy definitions. *ACM Transactions on Database Systems (TODS)*, 2014.

[156] Siwon Kim, Sangdoo Yun, Hwaran Lee, Martin Gubri, Sungroh Yoon, and Seong Joon Oh. ProPILE: Probing privacy leakage in large language models. In *Advances in Neural Information Processing Systems*, 2024.

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

[158] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)*, 2017.

[159] Denis Kocetkov, Raymond Li, Loubna Ben allal, Jia LI, Chenghao Mou, Yacine Jernite, Margaret Mitchell, Carlos Muñoz Ferrandis, Sean Hughes, Thomas Wolf, Dzmitry Bahdanau, Leandro Von Werra, and Harm de Vries. The Stack: 3 TB of permissively licensed source code. *Transactions on Machine Learning Research*, 2023.

[160] Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *International Conference on Machine Learning (ICML)*, 2017.

[161] Aran Komatsuzaki. One epoch is all you need. *arXiv preprint arXiv:1906.06669*, 2019.

[162] Tomasz Korbak, Kejian Shi, Angelica Chen, Rasika Bhalerao, Christopher L Buckley, Jason Phang, Samuel R Bowman, and Ethan Perez. Pretraining language models with human preferences. In *International Conference on Machine Learning (ICML)*, 2023.

[163] Emmanouil Krasanakis, Eleftherios Spyromitros-Xioufis, Symeon Papadopoulos, and Yiannis Kompatsiaris. Adaptive sensitive reweighting to mitigate bias in fairness-aware classification. In *World Wide Web Conference*, 2018.

[164] Bogdan Kulynych, Mohammad Yaghini, Giovanni Cherubin, Michael Veale, and Carmela Troncoso. Disparate vulnerability to membership inference attacks. *Privacy Enhancing Technologies Symposium (PETS)*, 2022.

[165] Sasi Kumar and Reza Shokri. Ml privacy meter: Aiding regulatory compliance by quantifying the privacy risks of machine learning. In *Workshop on Hot Topics in Privacy Enhancing Technologies (HotPETs)*, 2020.

[166] Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. Natural Questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 2019.

[167] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11), 1998.

[168] Jooyoung Lee, Thai Le, Jinghui Chen, and Dongwon Lee. Do language models plagiarize? In *ACM Web Conference*, 2023.

[169] Katherine Lee, Daphne Ippolito, Andrew Nystrom, Chiyuan Zhang, Douglas Eck, Chris Callison-Burch, and Nicholas Carlini. Deduplicating training data makes language models better. In *Findings of the Association for Computational Linguistics (ACL)*, 2022.

[170] Nayeon Lee, Wei Ping, Peng Xu, Mostofa Patwary, Pascale N Fung, Mohammad Shoeybi, and Bryan Catanzaro. Factuality enhanced language models for open-ended text generation. *Advances in Neural Information Processing Systems*, 2022.

[171] Eric Lehman, Sarthak Jain, Karl Pichotta, Yoav Goldberg, and Byron C Wallace. Does BERT pretrained on clinical notes reveal sensitive data? In *North American Chapter of the Association for Computational Linguistics*, 2021.

[172] Klas Leino and Matt Fredrikson. Stolen memories: Leveraging model memorization for calibrated white-box membership inference. In *USENIX Security Symposium*, 2020.

[173] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 2020.

[174] Guoyao Li, Shahbaz Rezaei, and Xin Liu. User-level membership inference attack against metric embedding learning. In *ICLR 2022 Workshop on PAIR {\textasciicircum} 2Struct: Privacy, Accountability, Interpretability, Robustness, Reasoning on Structured Data*, 2022.

[175] Haoran Li, Dadi Guo, Wei Fan, Mingshi Xu, Jie Huang, Fanpu Meng, and Yangqiu Song. Multi-step jailbreaking privacy attacks on chatgpt. In *Conference on Empirical Methods in Natural Language Processing*, 2023.

[176] Marvin Li, Jason Wang, Jeffrey Wang, and Seth Neel. Mope: Model perturbation-based privacy attacks on language models. In *Conference on Empirical Methods in Natural Language Processing*, 2023.

[177] Maximilian Li, Xander Davies, and Max Nadeau. Circuit breaking: Removing model behaviors with targeted ablation. *ICML Workshop on Deployment Challenges for Generative AI*, 2023.

[178] Ninghui Li, Wahbeh Qardaji, Dong Su, Yi Wu, and Weining Yang. Membership privacy: A unifying framework for privacy definitions. In *ACM Conference on Computer and Communications Security (CCS)*, 2013.

[179] Raymond Li, Yangtian Zi, Niklas Muennighoff, Denis Kocetkov, Chenghao Mou, Marc Marone, Christopher Akiki, LI Jia, Jenny Chim, Qian Liu, et al. Starcoder: may the source be with you! *Transactions on Machine Learning Research*, 2023.

[180] Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, 2020.

[181] Xuechen Li, Florian Tramer, Percy Liang, and Tatsunori Hashimoto. Large language models can be strong differentially private learners. In *International Conference on Learning Representations (ICLR)*, 2022.

[182] Yuchen Li, Yuanzhi Li, and Andrej Risteski. How do transformers learn topic structure: Towards a mechanistic understanding. In *International Conference on Machine Learning (ICML)*, 2023.

[183] Yucheng Li, Frank Guerin, and Chenghua Lin. Avoiding data contamination in language model evaluation: Dynamic test construction with latest materials, 2023.

[184] Zheng Li and Yang Zhang. Membership leakage in label-only exposures. In *ACM Conference on Computer and Communications Security (CCS)*, 2021.

[185] Bo Liu, Ming Ding, Sina Shaham, Wenny Rahayu, Farhad Farokhi, and Zihuai Lin. When machine learning meets privacy: A survey and outlook. *ACM Computing Surveys (CSUR)*, 2021.

[186] Hongbin Liu, Jinyuan Jia, Wenjie Qu, and Neil Zhenqiang Gong. EncoderMI: Membership inference against pre-trained encoders in contrastive learning. In *ACM Conference on Computer and Communications Security (CCS)*, 2021.

[187] Kangqiao Liu, Liu Ziyin, and Masahito Ueda. Noise and fluctuation of finite learning rate stochastic gradient descent. In *International Conference on Machine Learning (ICML)*, 2021.

[188] Ruixuan Liu, Tianhao Wang, Yang Cao, and Li Xiong. Precurious: How innocent pre-trained language models turn into privacy traps. *arXiv preprint arXiv:2403.09562*, 2024.

[189] Yuhan Liu, Ananda Theertha Suresh, Felix Xinnan X Yu, Sanjiv Kumar, and Michael Riley. Learning discrete distributions: user vs item-level privacy. *Advances in Neural Information Processing Systems*, 2020.

[190] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Large-scale celebfaces attributes (celeba) dataset, 2018.

[191] Shayne Longpre, Le Hou, Tu Vu, Albert Webson, Hyung Won Chung, Yi Tay, Denny Zhou, Quoc V Le, Barret Zoph, Jason Wei, et al. The Flan Collection: Designing data and methods for effective instruction tuning. In *International Conference on Machine Learning (ICML)*, 2023.

[192] Albert Lu, Hongxin Zhang, Yanzhe Zhang, Xuezhi Wang, and Diyi Yang. Bounding the capabilities of large language models in open text generation with prompt constraints. In *Findings of the Association for Computational Linguistics: EACL 2023*, pages 1982–2008, 2023.

[193] Nils Lukas, Ahmed Salem, Robert Sim, Shruti Tople, Lukas Wutschitz, and Santiago Zanella-Béguelin. Analyzing leakage of personally identifiable information in language models. In *IEEE Symposium on Security and Privacy*, 2023.

[194] Lingjuan Lyu and Chen Chen. A novel attribute reconstruction attack in federated learning. *arXiv:1712.00409*, 2021.

[195] Xindi Ma, Baopu Li, Qi Jiang, Yimin Chen, Sheng Gao, and Jianfeng Ma. NOSnoop: an effective collaborative meta-learning scheme against property inference attack. *IEEE Internet of Things Journal*, 2021.

[196] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations (ICLR)*, 2018.

[197] Ian Magnusson, Akshita Bhagia, Valentin Hofmann, Luca Soldaini, Ananya Harsh Jha, Oyvind Tafjord, Dustin Schwenk, Evan Pete Walsh, Yanai Elazar, Kyle Lo, Dirk Groenveld, Iz Beltagy, Hanneneh Hajishirz, Noah A. Smith, Kyle Richardson, and Jesse Dodge. Paloma: A benchmark for evaluating language model fit. *technical report*, 2023. URL https://paloma.allen.ai/.

[198] Saeed Mahloujifar, Huseyin A. Inan, Melissa Chase, Esha Ghosh, and Marcello Hasegawa. Membership inference on word embedding and beyond. *arXiv preprint arXiv:2106.11384 [cs.CL]*, 2021.

[199] Saeed Mahloujifar, Esha Ghosh, and Melissa Chase. Property inference from poisoning. In *IEEE Symposium on Security and Privacy*, 2022.

[200] Pratyush Maini, Mohammad Yaghini, and Nicolas Papernot. Dataset inference: Ownership resolution in machine learning. In *International Conference on Learning Representations (ICLR)*, 2020.

[201] Pratyush Maini, Michael Curtis Mozer, Hanie Sedghi, Zachary Chase Lipton, J Zico Kolter, and Chiyuan Zhang. Can neural network memorization be localized? In *International Conference on Machine Learning (ICML)*, 2023.

[202] Mani Malek Esmaeili, Ilya Mironov, Karthik Prasad, Igor Shilov, and Florian Tramèr. Antipodes of label differential privacy: PATE and ALIBI. In *Advances in Neural Information Processing Systems*, 2021.

[203] Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Hannaneh Hajishirzi, and Daniel Khashabi. When not to trust language models: Investigating effectiveness and limitations of parametric and non-parametric memories. In *Findings of the Association for Computational Linguistics (ACL)*, 2022.

[204] Krishna Sri Ipsit Mantri and Nevasini Sasikumatm. Developing methods for identifying and removing copyrighted content from generative AI models. In *ICML Workshop on Generative AI and Law*, 2023.

[205] Virendra J Marathe and Pallika Kanani. Subject granular differential privacy in federated learning. In *Privacy Preserving Machine Learning (PPML) workshop at ACM CCS*, 2021.

[206] Virendra J. Marathe, Pallika Kanani, and Daniel W. Peterson. Subject level differential privacy with hierarchical gradient averaging. In *International Workshop on Federated Learning: Recent Advances and New Challenges in Conjunction with NeurIPS 2022*, 2022.

[207] Mathematics State Exchange User 'user13888'. What is the relationship of $\mathcal{L}_1$ (total variation) distance to hypothesis testing? Mathematics Stack Exchange, https://math.stackexchange.com/q/72730, 2011.

[208] Justus Mattern, Fatemehsadat Mireshghallah, Zhijing Jin, Bernhard Schoelkopf, Mrinmaya Sachan, and Taylor Berg-Kirkpatrick. Membership inference attacks against language models via neighbourhood comparison. In *Findings of the Association for Computational Linguistics (ACL)*, 2023.

[209] R Thomas McCoy, Shunyu Yao, Dan Friedman, Matthew Hardy, and Thomas L Griffiths. Embers of autoregression: Understanding large language models through the problem they are trained to solve. *arXiv preprint arXiv:2309.13638*, 2023.

[210] H. Brendan McMahan, Daniel Ramage, Kunal Talwar, and Li Zhang. Learning differentially private recurrent language models. In *International Conference on Learning Representations (ICLR)*, 2018.

[211] Matthieu Meeus, Shubham Jain, Marek Rei, and Yves-Alexandre de Montjoye. Did the neurons read your book? document-level membership inference for large language models. *arXiv preprint arXiv:2310.15007*, 2023.

[212] Ninareh Mehrabi, Fred Morstatter, Nripsuta Saxena, Kristina Lerman, and Aram Galstyan. A survey on bias and fairness in machine learning. *ACM Computing Surveys (CSUR)*, 2022.

[213] Simon Meier, Benedikt Schmidt, Cas Cremers, and David Basin. The TAMARIN prover for the symbolic analysis of security protocols. In *Computer Aided Verification (CAV)*, 2013.

[214] Luca Melis, Congzheng Song, Emiliano De Cristofaro, and Vitaly Shmatikov. Exploiting unintended feature leakage in collaborative learning. In *IEEE Symposium on Security and Privacy*, 2019.

[215] Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in GPT. *Advances in Neural Information Processing Systems*, 2022.

[216] Kevin Meng, Arnab Sen Sharma, Alex Andonian, Yonatan Belinkov, and David Bau. Mass-editing memory in a transformer. In *International Conference on Learning Representations (ICLR)*, 2023.

[217] Sewon Min, Suchin Gururangan, Eric Wallace, Hannaneh Hajishirzi, Noah A Smith, and Luke Zettlemoyer. SILO language models: Isolating legal risk in a nonparametric datastore. In *International Conference on Learning Representations (ICLR)*, 2024.

[218] Fatemehsadat Mireshghallah and Taylor Berg-Kirkpatrick. Style pooling: Automatic text style obfuscation for improved classification fairness. In *Conference on Empirical Methods in Natural Language Processing*, 2021.

[219] Fatemehsadat Mireshghallah, Kartik Goyal, Archit Uniyal, Taylor Berg-Kirkpatrick, and Reza Shokri. Quantifying privacy risks of masked language models using membership inference attacks. In *Conference on Empirical Methods in Natural Language Processing*, 2022.

[220] Fatemehsadat Mireshghallah, Archit Uniyal, Tianhao Wang, David K Evans, and Taylor Berg-Kirkpatrick. An empirical analysis of memorization in fine-tuned autoregressive language models. In *Conference on Empirical Methods in Natural Language Processing*, 2022.

[221] Ilya Mironov. Rényi Differential Privacy. In *IEEE Computer Security Foundations Symposium (CSF)*, 2017.

[222] Frederick Mosteller and David L Wallace. Inference in an authorship problem: A comparative study of discrimination methods applied to the authorship of the disputed Federalist Papers. *Journal of the American Statistical Association*, 1963.

[223] Hamid Mozaffari and Virendra J. Marathe. Semantic membership inference attack against large language models. *arXiv preprint arXiv:2406.10218*, 2024.

[224] Niklas Muennighoff, Alexander M Rush, Boaz Barak, Teven Le Scao, Aleksandra Piktus, Nouamane Tazi, Sampo Pyysalo, Thomas Wolf, and Colin Raffel. Scaling data-constrained language models. In *Advances in Neural Information Processing Systems*, 2023.

[225] Yuta Nakamura, Shouhei Hanaoka, Yukihiro Nomura, Naoto Hayashi, Osamu Abe, Shuntaro Yada, Shoko Wakamiya, and Eiji Aramaki. Kart: Parameterization of privacy leakage scenarios from pre-trained language models. *arXiv preprint arXiv:2101.00036*, 2020.

[226] Preetum Nakkiran, Gal Kaplun, Yamini Bansal, Tristan Yang, Boaz Barak, and Ilya Sutskever. Deep double descent: Where bigger models and more data hurt. *Journal of Statistical Mechanics: Theory and Experiment*, 2021(12), 2021.

[227] Neel Nanda, Lawrence Chan, Tom Liberum, Jess Smith, and Jacob Steinhardt. Progress measures for grokking via mechanistic interpretability. In *International Conference on Learning Representations (ICLR)*, 2023.

[228] Cleo Nardo. The Waluigi effect. https://www.lesswrong.com/posts/D7PumeYTDPfBTp3i7/the-waluigi-effect-mega-post, 2023. Accessed: 2023-07-24.

[229] Milad Nasr, Reza Shokri, and Amir Houmansadr. Comprehensive privacy analysis of deep learning. In *IEEE Symposium on Security and Privacy*, 2018.

[230] Milad Nasr, Reza Shokri, and Amir Houmansadr. Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning. In *IEEE Symposium on Security and Privacy*, 2019.

[231] Milad Nasr, Shuang Songi, Abhradeep Thakurta, Nicolas Papernot, and Nicholas Carlini. Adversary instantiation: Lower bounds for differentially private machine learning. In *IEEE Symposium on Security and Privacy*, 2021.

[232] The Hacker News. Over 100 malicious ai/ml models found on hugging face platform, 2024. URL https://thehackernews.com/2024/03/over-100-malicious-aiml-models-found-on.html. Accessed: 2024-07-03.

[233] Cao Hong Nga, Khai-Thinh Nguyen, Nghi C Tran, and Jia-Ching Wang. Transfer learning for gender and age prediction. In *IEEE International Conference on Consumer Electronics-Taiwan (ICCE-Taiwan)*, 2020.

[234] Kobbi Nissim, Aaron Bembenek, Alexandra Wood, Mark Bun, Marco Gaboardi, Urs Gasser, David O'Brien, Salil P. Vadhan, and Thomas Steinke. Bridging the gap between computer science and legal approaches to privacy. *Harvard Journal of Law & Technology*, 2018.

[235] Sayedeh Leila Noorbakhsh, Binghui Zhang, Yuan Hong, and Binghui Wang. Inf2guard: An information-theoretic framework for learning privacy-preserving representations against inference attacks. In *USENIX Security Symposium*, 2024.

[236] State of California Department of Justice. California consumer privacy act (CCPA), 2018. https://oag.ca.gov/privacy/ccpa.

[237] Elre Talea Oldewage, Ross M Clarke, and José Miguel Hernández-Lobato. Series of hessian-vector products for tractable saddle-free newton optimisation of neural networks. *Transactions on Machine Learning Research*, 2024.

[238] Takeshi Onishi, Hai Wang, Mohit Bansal, Kevin Gimpel, and David McAllester. Who did what: A large-scale person-centered cloze dataset. In *Conference on Empirical Methods in Natural Language Processing*, 2016.

[239] OpenAI. GPTBot, 2023. https://platform.openai.com/docs/gptbot.

[240] OpenAI. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[241] OpenAI. Openai red teaming network, 2023. URL https://openai.com/index/red-teaming-network/. Accessed: 2024-07-03.

[242] OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, Irwan Bello, Jake Berdine, Gabriel Bernadett-Shapiro, Christopher Berner, Lenny Bogdonoff, Oleg Boiko, Madelaine Boyd, Anna-Luisa Brakman, Greg Brockman, Tim Brooks, Miles Brundage, Kevin Button, Trevor Cai, Rosie Campbell, Andrew Cann, Brittany Carey, Chelsea Carlson, Rory Carmichael, Brooke Chan, Che Chang, Fotis Chantzis, Derek Chen, Sully Chen, Ruby Chen, Jason Chen, Mark Chen, Ben Chess, Chester Cho, Casey Chu, Hyung Won Chung, Dave Cummings, Jeremiah Currier, Yunxing Dai, Cory Decareaux, Thomas Degry, Noah Deutsch, Damien Deville, Arka Dhar, David Dohan, Steve Dowling, Sheila Dunning, Adrien Ecoffet, Atty Eleti, Tyna Eloundou, David Farhi, Liam Fedus, Niko Felix, Simón Posada Fishman, Juston Forte, Isabella Fulford, Leo Gao, Elie Georges, Christian Gibson, Vik Goel, Tarun Gogineni, Gabriel Goh, Rapha Gontijo-Lopes, Jonathan Gordon, Morgan Grafstein, Scott Gray, Ryan Greene, Joshua Gross, Shixiang Shane Gu, Yufei Guo, Chris Hallacy, Jesse Han, Jeff Harris, Yuchen He, Mike Heaton, Johannes Heidecke, Chris Hesse, Alan Hickey, Wade Hickey, Peter Hoeschele, Brandon Houghton, Kenny Hsu, Shengli Hu, Xin Hu, Joost Huizinga, Shantanu Jain, Shawn Jain, Joanne Jang, Angela Jiang, Roger Jiang, Haozhun Jin, Denny Jin, Shino Jomoto, Billie Jonn, Heewoo Jun, Tomer Kaftan, Łukasz Kaiser, Ali Kamali, Ingmar Kanitscheider, Nitish Shirish Keskar, Tabarak Khan, Logan Kilpatrick, Jong Wook Kim, Christina Kim, Yongjik Kim, Jan Hendrik Kirchner, Jamie Kiros, Matt Knight, Daniel Kokotajlo, Łukasz Kondraciuk, Andrew Kondrich, Aris Konstantinidis, Kyle Kosic, Gretchen Krueger, Vishal Kuo, Michael Lampe, Ikai Lan, Teddy Lee, Jan Leike, Jade Leung, Daniel Levy, Chak Ming Li, Rachel Lim, Molly Lin, Stephanie Lin, Mateusz Litwin, Theresa Lopez, Ryan Lowe, Patricia Lue, Anna Makanju, Kim Malfacini, Sam Manning, Todor Markov, Yaniv Markovski, Bianca Martin, Katie Mayer, Andrew Mayne, Bob McGrew, Scott Mayer McKinney, Christine McLeavey, Paul McMillan, Jake McNeil, David Medina, Aalok Mehta, Jacob Menick, Luke Metz, Andrey Mishchenko, Pamela Mishkin, Vinnie Monaco, Evan Morikawa, Daniel Mossing, Tong Mu, Mira Murati, Oleg Murk, David Mély, Ashvin Nair, Reiichiro Nakano, Rajeev Nayak, Arvind Neelakantan, Richard Ngo, Hyeonwoo Noh, Long Ouyang, Cullen O'Keefe, Jakub Pachocki, Alex Paino, Joe Palermo, Ashley Pantuliano, Giambattista Parascandolo, Joel Parish, Emy Parparita, Alex Passos, Mikhail Pavlov, Andrew Peng, Adam Perelman, Filipe de Avila Bel-

bute Peres, Michael Petrov, Henrique Ponde de Oliveira Pinto, Michael, Pokorny, Michelle Pokrass, Vitchyr H. Pong, Tolly Powell, Alethea Power, Boris Power, Elizabeth Proehl, Raul Puri, Alec Radford, Jack Rae, Aditya Ramesh, Cameron Raymond, Francis Real, Kendra Rimbach, Carl Ross, Bob Rotsted, Henri Roussez, Nick Ryder, Mario Saltarelli, Ted Sanders, Shibani Santurkar, Girish Sastry, Heather Schmidt, David Schnurr, John Schulman, Daniel Selsam, Kyla Sheppard, Toki Sherbakov, Jessica Shieh, Sarah Shoker, Pranav Shyam, Szymon Sidor, Eric Sigler, Maddie Simens, Jordan Sitkin, Katarina Slama, Ian Sohl, Benjamin Sokolowsky, Yang Song, Natalie Staudacher, Felipe Petroski Such, Natalie Summers, Ilya Sutskever, Jie Tang, Nikolas Tezak, Madeleine B. Thompson, Phil Tillet, Amin Tootoonchian, Elizabeth Tseng, Preston Tuggle, Nick Turley, Jerry Tworek, Juan Felipe Cerón Uribe, Andrea Vallone, Arun Vijayvergiya, Chelsea Voss, Carroll Wainwright, Justin Jay Wang, Alvin Wang, Ben Wang, Jonathan Ward, Jason Wei, CJ Weinmann, Akila Welihinda, Peter Welinder, Jiayi Weng, Lilian Weng, Matt Wiethoff, Dave Willner, Clemens Winter, Samuel Wolrich, Hannah Wong, Lauren Workman, Sherwin Wu, Jeff Wu, Michael Wu, Kai Xiao, Tao Xu, Sarah Yoo, Kevin Yu, Qiming Yuan, Wojciech Zaremba, Rowan Zellers, Chong Zhang, Marvin Zhang, Shengjia Zhao, Tianhao Zheng, Juntang Zhuang, William Zhuk, and Barret Zoph. Gpt-4 technical report, 2024.

[243] Tribhuvanesh Orekondy, Bernt Schiele, and Mario Fritz. Knockoff nets: Stealing functionality of black-box models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[244] Yonatan Oren, Nicole Meister, Niladri Chatterji, Faisal Ladhak, and Tatsunori B Hashimoto. Proving test set contamination in black box language models. In *International Conference on Learning Representations (ICLR)*, 2024.

[245] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 2022.

[246] Mustafa Ozdayi, Charith Peris, Jack FitzGerald, Christophe Dupuy, Jimit Majmudar, Haidar Khan, Rahil Parikh, and Rahul Gupta. Controlling the extraction of memorized data from large language models via prompt-tuning. In *Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 2023.

[247] Nicolas Papernot, Patrick McDaniel, Arunesh Sinha, and Michael P Wellman. SoK: Security and privacy in machine learning. In *IEEE European Symposium on Security and Privacy (EuroS&P)*. IEEE, 2018.

[248] Vardan Papyan, XY Han, and David L Donoho. Prevalence of neural collapse during the terminal phase of deep learning training. *Proceedings of the National Academy of Sciences*, 2020.

[249] Article 29 Data Protection Working Party. Opinion 05/2014 on anonymization techniques, Nov 2014. URL https://ec.europa.eu/justice/article-29/documentation/opinion-recommendation/files/2014/wp216_en.pdf.

[250] Dario Pasquini, Giuseppe Ateniese, and Massimo Bernaschi. Unleashing the tiger: Inference attacks on split learning. In *ACM SIGSAC Conference on Computer and Communications Security*, 2021.

[251] Vaidehi Patil, Peter Hase, and Mohit Bansal. Can sensitive information be deleted from LLMs? Objectives for defending against extraction attacks. In *International Conference on Learning Representations (ICLR)*, 2024.

[252] Judea Pearl. Direct and indirect effects. In *Uncertainty in Artificial Intelligence*, 2001.

[253] Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick S. H. Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander H. Miller. Language models as knowledge bases? In *Conference on Empirical Methods in Natural Language Processing*, 2019.

[254] Nina Pörner, Ulli Waltinger, and Hinrich Schütze. E-BERT: Efficient-yet-effective entity embeddings for BERT. In *Findings of the Association for Computational Linguistics*, 2020.

[255] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 2019.

[256] Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D Manning, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. In *Advances in Neural Information Processing Systems*, 2024.

[257] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 2020.

[258] Vikas Raunak, Arul Menezes, and Marcin Junczys-Dowmunt. The curious case of hallucinations in neural machine translation. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2021.

[259] Emily Reif, Daphne Ippolito, Ann Yuan, Andy Coenen, Chris Callison-Burch, and Jason Wei. A recipe for arbitrary text style transfer with large language models. In *Findings of the Association for Computational Linguistics (ACL)*, 2022.

[260] Max Reuter and William Schulze. I'm afraid I can't do that: Predicting prompt refusal in black-box generative language models. *arXiv preprint arXiv:2306.03423*, 2023.

[261] Maria Rigaki and Sebastian Garcia. A survey of privacy attacks in machine learning. *ACM Computing Surveys*, 2020.

[262] Phillip Rogaway. Formalizing human ignorance. In *Progress in Cryptology – VIETCRYPT*. Springer, 2006.

[263] Sebastian Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.

[264] Alexandre Sablayrolles, Matthijs Douze, Cordelia Schmid, Yann Ollivier, and Hervé Jégou. White-box vs black-box: Bayes optimal strategies for membership inference. In *International Conference on Machine Learning (ICML)*, 2019.

[265] Hasim Sak, Andrew W. Senior, and Françoise Beaufays. Long short-term memory based recurrent neural network architectures for large vocabulary speech recognition. In *INTERSPEECH*, 2014.

[266] 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. In *Network and Distributed Systems Security Symposium*, 2019.

[267] Ahmed Salem, Apratim Bhattacharya, Michael Backes, Mario Fritz, and Yang Zhang. Updates-Leak: Data set inference and reconstruction attacks in online learning. In *USENIX Security Symposium*, 2020.

[268] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[269] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. In *NeurIPS EMC^2 Workshop*, 2019.

[270] Issei Sato and Hiroshi Nakagawa. Approximation Analysis of Stochastic Gradient Langevin Dynamics by using Fokker-Planck Equation and Ito Process. In *International Conference on Machine Learning (ICML)*, 2014.

[271] David Saxton, Edward Grefenstette, Felix Hill, and Pushmeet Kohli. Analysing mathematical reasoning abilities of neural models. In *International Conference on Learning Representations (ICLR)*, 2019.

[272] Florian Schroff, Dmitry Kalenichenko, and James Philbin. FaceNet: A unified embedding for face recognition and clustering. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[273] Roei Schuster, Tal Schuster, Yoav Meri, and Vitaly Shmatikov. Humpty dumpty: Controlling word meanings via corpus poisoning. In *IEEE Symposium on Security and Privacy*, 2020.

[274] Ozan Sener and Vladlen Koltun. Multi-task learning as multi-objective optimization. In *Advances in Neural Information Processing Systems*, 2018.

[275] Virat Shejwalkar and Amir Houmansadr. Membership privacy for machine learning models through knowledge transfer. In *AAAI Conference on Artificial Intelligence*, 2021.

[276] Weijia Shi, Anirudh Ajith, Mengzhou Xia, Yangsibo Huang, Daogao Liu, Terra Blevins, Danqi Chen, and Luke Zettlemoyer. Detecting pretraining data from large language models. *Regulatable ML Workshop at NeurIPS*, 2023.

[277] Weiyan Shi, Aiqi Cui, Evan Li, Ruoxi Jia, and Zhou Yu. Selective differential privacy for language modeling. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2022.

[278] Reza Shokri. Auditing Data Privacy for Machine Learning. In *USENIX Security Symposium*, 2022.

[279] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *IEEE Symposium on Security and Privacy*, 2017.

[280] Irene Solaiman and Christy Dennison. Process for adapting language models to society (PALMS) with values-targeted datasets. *Advances in Neural Information Processing Systems*, 2021.

[281] Luca Soldaini and Kyle Lo. pes2o (pretraining efficiently on s2orc) dataset. Technical report, Allen Institute for AI, 2023. ODC-By, https://github.com/allenai/pes2o.

[282] Luca Soldaini, Rodney Kinney, Akshita Bhagia, Dustin Schwenk, David Atkinson, Russell Authur, Ben Bogin, Khyathi Chandu, Jennifer Dumas, Yanai Elazar, Valentin Hofmann, Ananya Harsh Jha, Sachin Kumar, Li Lucy, Xinxi Lyu, Ian Magnusson, Jacob Morrison, Niklas Muennighoff, Aakanksha Naik, Crystal Nam, Matthew E. Peters, Abhilasha Ravichander, Kyle Richardson, Zejiang Shen, Emma Strubell, Nishant Subramani, Oyvind Tafjord, Evan Pete Walsh, Hannaneh Hajishirzi, Noah A. Smith, Luke Zettlemoyer, Iz Beltagy, Dirk Groeneveld, Jesse Dodge, and Kyle Lo. Dolma: An open corpus of three trillion tokens for language model pretraining research. *arXiv preprint*, 2023.

[283] Jared Spataro. Announcing Microsoft 365 Copilot general availability and Microsoft 365 Chat. https://www.microsoft.com/en-us/microsoft-365/blog/2023/09/21/announcing-microsoft-365-copilot-general-availability-and-microsoft-365-chat/, 2023. Accessed: 2023-10-01.

[284] Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, et al. Beyond the

imitation game: Quantifying and extrapolating the capabilities of language models. *Transactions on Machine Learning Research*, 2022.

[285] Robin Staab, Mark Vero, Mislav Balunović, and Martin Vechev. Beyond memorization: Violating privacy via inference with large language models. In *International Conference on Learning Representations (ICLR)*, 2024.

[286] Thomas Steinke, Milad Nasr, and Matthew Jagielski. Privacy auditing with one (1) training run. In *Advances in Neural Information Processing Systems*, 2023.

[287] Mandt Stephan, Matthew D Hoffman, David M Blei, and others. Stochastic gradient descent as approximate bayesian inference. *Journal of Machine Learning Research*, 2017.

[288] Jacques Stern. Why provable security matters? In *Advances in Cryptology – EUROCRYPT*. Springer, 2003.

[289] Joshua Stock, Jens Wettlaufer, Daniel Demmler, and Hannes Federrath. Lessons learned: How (not) to defend against property inference attacks. In *International Conference on Security and Cryptography (SECRYPT)*, 2023.

[290] Ion Stoica, Robert Morris, David Liben-Nowell, David R Karger, M Frans Kaashoek, Frank Dabek, and Hari Balakrishnan. Chord: A scalable peer-to-peer lookup protocol for internet applications. *IEEE/ACM Transactions on Networking*, 2003.

[291] Alessandro Stolfo, Yonatan Belinkov, and Mrinmaya Sachan. Understanding arithmetic reasoning in language models using causal mediation analysis. *arXiv preprint arXiv:2305.15054*, 2023.

[292] Jerry Su. Census19. https://github.com/JerrySu11/CensusData, 2022.

[293] Anshuman Suri and David Evans. Formalizing and estimating distribution inference risks. *Privacy Enhancing Technologies Symposium (PETS)*, 2022.

[294] Anshuman Suri, Pallika Kanani, Virendra J Marathe, and Daniel W Peterson. Subject membership inference attacks in federated learning. *arXiv preprint arXiv:2206.03317 [cs.LG]*, 2022.

[295] Anshuman Suri, Yifu Lu, Yanjin Chen, and David Evans. Dissecting distribution inference. In *IEEE Conference on Secure and Trustworthy Machine Learning (SaTML)*, 2023.

[296] Fnu Suya, Anshuman Suri, Tingwei Zhang, Jingtao Hong, Yuan Tian, and David Evans. Sok: Pitfalls in evaluating black-box attacks. In *IEEE Conference on Secure and Trustworthy Machine Learning (SaTML)*, 2024.

[297] Sebastian Szyller, Rui Zhang, Jian Liu, and N Asokan. On the robustness of dataset inference. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856.

[298] Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. Deepface: Closing the gap to human-level performance in face verification. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[299] Jasper Tan, Blake Mason, Hamid Javadi, and Richard Baraniuk. Parameters or privacy: A provable tradeoff between overparameterization and membership inference. *Advances in Neural Information Processing Systems*, 2022.

[300] Xinyu Tang, Saeed Mahloujifar, Liwei Song, Virat Shejwalkar, Milad Nasr, Amir Houmansadr, and Prateek Mittal. Mitigating membership inference attacks by self-distillation through a novel ensemble architecture. In *USENIX Security Symposium*, 2022.

[301] MosaicML NLP Team. Introducing mpt-7b: A new standard for open-source, commercially usable llms. www.mosaicml.com/blog/mpt-7b, 2023. Accessed: 2024-01-11.

[302] Philipp Terhörst, Marco Huber, Jan Niklas Kolf, Ines Zelch, Naser Damer, Florian Kirchbuchner, and Arjan Kuijper. Reliable age and gender estimation from face images: Stating the confidence of model predictions. In *IEEE International Conference on Biometrics Theory, Applications and Systems (BTAS)*, 2019.

[303] Philipp Terhörst, Daniel Fährmann, Jan Niklas Kolf, Naser Damer, Florian Kirchbuchner, and Arjan Kuijper. MAAD-face: A massively annotated attribute dataset for face images. *IEEE Transactions on Information Forensics and Security*, 2021.

[304] Abhradeep Guha Thakurta, Andrew H Vyrros, Umesh S Vaishampayan, Gaurav Kapoor, Julien Freudiger, Vivek Rangarajan Sridhar, and Doug Davidson. Learning new words. US Patent #9594741, 2017.

[305] Nitasha Tiku. Newspapers want payment for articles used to power ChatGPT. *The Washington Post*, Oct 2023.

[306] Kushal Tirumala, Aram Markosyan, Luke Zettlemoyer, and Armen Aghajanyan. Memorization without overfitting: Analyzing the training dynamics of large language models. In *Advances in Neural Information Processing Systems*, 2022.

[307] Shruti Tople, Amit Sharma, and Aditya Nori. Alleviating privacy attacks via causal learning. In *International Conference on Machine Learning (ICML)*, 2020.

[308] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. LLaMA: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

[309] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. Llama 2: Open foundation and fine-tuned chat models, 2023.

[310] Jonathan Tow. Stablelm alpha v2 models, 2023. URL https://huggingface.co/stabilityai/stablelm-base-alpha-3b-v2.

[311] Florian Tramèr, Fan Zhang, Ari Juels, Michael K Reiter, and Thomas Ristenpart. Stealing machine learning models via prediction APIs. In *USENIX Security Symposium*, 2016.

[312] Florian Tramèr, Andreas Terzis, Thomas Steinke, Shuang Song, Matthew Jagielski, and Nicholas Carlini. Debugging differential privacy: A case study for privacy auditing. *arXiv preprint arXiv:2202.12219 [cs.LG]*, 2022.

[313] Florian Tramèr, Reza Shokri, Ayrton San Joaquin, Hoang Le, Matthew Jagielski, Sanghyun Hong, and Nicholas Carlini. Truth serum: Poisoning machine learning models to reveal their secrets. In *ACM Conference on Computer and Communications Security (CCS)*, 2022.

[314] Enrica Troiano, Aswathy Velutharambath, and Roman Klinger. From theories on styles to their transfer in text: Bridging the gap with a hierarchical survey. *Natural Language Engineering*, 2023.

[315] Stacey Truex, Ling Liu, Mehmet Emre Gursoy, Lei Yu, and Wenqi Wei. Demystifying membership inference attacks in machine learning as a service. *IEEE transactions on services computing*, 2019.

[316] Michael Carl Tschantz, Shayak Sen, and Anupam Datta. SoK: Differential privacy as a causal property. In *IEEE Symposium on Security and Privacy*, 2020.

[317] Alexandre B Tsybakov. Introduction to nonparametric estimation, 2009.

[318] Matthew Turk and Alex Pentland. Eigenfaces for recognition. *Journal of cognitive neuroscience*, 1991.

[319] Jacob Tyo, Bhuwan Dhingra, and Zachary C Lipton. On the state of the art in authorship attribution and authorship verification. *arXiv preprint arXiv:2209.06869*, 2022.

[320] Stanford University. In *AI Index Report 2024*. Stanford Institute for Human-Centered Artificial Intelligence (HAI), 2024. URL https://aiindex.stanford.edu/wp-content/uploads/2024/05/HAI_AI-Index-Report-2024.pdf.

[321] Nikhil Vyas, Sham Kakade, and Boaz Barak. Provable copyright protection for generative models. In *International Conference on Machine Learning (ICML)*, 2023.

[322] Ben Wang and Aran Komatsuzaki. GPT-J-6B: A 6 billion parameter autoregressive language model, 2021. https://github.com/kingoflolz/mesh-transformer-jax.

[323] Binghui Wang and Neil Zhenqiang Gong. Stealing hyperparameters in machine learning. In *IEEE Symposium on Security and Privacy*, 2018.

[324] Bolun Wang, Yuanshun Yao, Bimal Viswanath, Haitao Zheng, and Ben Y Zhao. With great training comes great vulnerability: Practical attacks against transfer learning. In *USENIX Security Symposium*, 2018.

[325] Boxin Wang, Weixin Chen, Hengzhi Pei, Chulin Xie, Mintong Kang, Chenhui Zhang, Chejian Xu, Zidi Xiong, Ritik Dutta, Rylan Schaeffer, et al. DecodingTrust: A comprehensive assessment of trustworthiness in GPT models. In *Advances in Neural Information Processing Systems*, 2023.

[326] Chaojun Wang and Rico Sennrich. On exposure bias, hallucination and domain shift in neural machine translation. In *58th Annual Meeting of the Association for Computational Linguistics*, 2020.

[327] Hongmin Wang. Revisiting challenges in data-to-text generation with fact grounding. In *International Conference on Natural Language Generation*, 2020.

[328] Jindong Wang, Cuiling Lan, Chang Liu, Yidong Ouyang, and Tao Qin. Generalizing to unseen domains: A survey on domain generalization. *Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence (IJCAI), Survey Track*, 2021.

[329] Kevin Wang, Alexandre Variengien, Arthur Conmy, Buck Shlegeris, and Jacob Steinhardt. Interpretability in the wild: a circuit for indirect object identification in GPT-2 small. In *International Conference on Learning Representations (ICLR)*, 2023.

[330] Kuansan Wang, Zhihong Shen, Chiyuan Huang, Chieh-Han Wu, Yuxiao Dong, and Anshul Kanakia. Microsoft Academic Graph: When experts are not enough. *Quantitative Science Studies*, 2020.

[331] Lingzhi Wang, Tong Chen, Wei Yuan, Xingshan Zeng, Kam-Fai Wong, and Hongzhi Yin. Kga: A general machine unlearning framework based on knowledge gap alignment. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13264–13276, 2023.

[332] Tianhao Wang, Yuheng Zhang, and Ruoxi Jia. Improving robustness to model inversion attacks via mutual information regularization. In *AAAI Conference on Artificial Intelligence*, 2021.

[333] Xiuling Wang and Wendy Hui Wang. Group property inference attacks against graph neural networks. In *ACM Conference on Computer and Communications Security (CCS)*, 2022.

[334] Stanley L Warner. Randomized response: A survey technique for eliminating evasive answer bias. *Journal of the American Statistical Association*, 1965.

[335] Lauren Watson, Chuan Guo, Graham Cormode, and Alexandre Sablayrolles. On the importance of difficulty calibration in membership inference attacks. In *International Conference on Learning Representations (ICLR)*, 2022.

[336] Jess Weatherbed. The New York Times prohibits using its content to train AI models, 2023. https://www.theverge.com/2023/8/14/23831109/the-new-york-times-ai-web-scraping-rules-terms-of-service.

[337] Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. Finetuned language models are zero-shot learners. In *International Conference on Learning Representations (ICLR)*, 2022.

[338] Orion Weller, Marc Marone, Nathaniel Weir, Dawn Lawrie, Daniel Khashabi, and Benjamin Van Durme. "according to...": Prompting language models improves quoting from pre-training data. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2024.

[339] Max Welling and Yee W Teh. Bayesian learning via stochastic gradient langevin dynamics. In *International Conference on Machine Learning (ICML)*, 2011.

[340] Yuxin Wen, Jonas A Geiping, Liam Fowl, Micah Goldblum, and Tom Goldstein. Fishing for user data in large-batch federated learning via gradient magnification. In *International Conference on Machine Learning (ICML)*, 2022.

[341] Yuxin Wen, Leo Marchyok, Sanghyun Hong, Jonas Geiping, Tom Goldstein, and Nicholas Carlini. Privacy backdoors: Enhancing membership inference through poisoning pre-trained models. *arXiv preprint arXiv:2404.01231*, 2024.

[342] Yotam Wolf, Noam Wies, Yoav Levine, and Amnon Shashua. Fundamental limitations of alignment in large language models. In *International Conference on Machine Learning (ICML)*, 2024.

[343] Bang Wu, Xiangwen Yang, Shirui Pan, and Xingliang Yuan. Adapting membership inference attacks to GNN for graph classification: Approaches and implications. In *IEEE International Conference on Data Mining (ICDM)*. IEEE, 2021.

[344] Xi Wu, Matthew Fredrikson, Somesh Jha, and Jeffrey F. Naughton. A methodology for formalizing model-inversion attacks. In *IEEE Computer Security Foundations Symposium (CSF)*. IEEE, 2016.

[345] Yu Xia, Di Huang, and Yunhong Wang. Detecting smiles of young children via deep transfer learning. In *IEEE International Conference on Computer Vision Workshops*, 2017.

[346] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms, August 2017. arXiv: cs.LG/1708.07747.

[347] Yiting Xie and David Richmond. Pre-training on grayscale ImageNet improves medical image classification. In *European Conference on Computer Vision (ECCV) Workshops*, 2018.

[348] Xiaojun Xu, Qi Wang, Huichen Li, Nikita Borisov, Carl A Gunter, and Bo Li. Detecting ai trojans using meta neural analysis. In *IEEE Symposium on Security and Privacy*, 2021.

[349] Kaiyu Yang, Jacqueline Yau, Li Fei-Fei, Jia Deng, and Olga Russakovsky. A study of face obfuscation in imagenet. In *International Conference on Machine Learning (ICML)*, 2022.

[350] Shuo Yang, Wei-Lin Chiang, Lianmin Zheng, Joseph E. Gonzalez, and Ion Stoica. Rethinking benchmark and contamination for language models with rephrased samples, 2023.

[351] Zonglin Yang, Li Dong, Xinya Du, Hao Cheng, Erik Cambria, Xiaodong Liu, Jianfeng Gao, and Furu Wei. Language models as inductive reasoners. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2024.

[352] Yuanshun Yao, Huiying Li, Haitao Zheng, and Ben Y Zhao. Latent backdoor attacks on deep neural networks. In *ACM Conference on Computer and Communications Security (CCS)*, 2019.

[353] Jiayuan Ye, Aadyaa Maddi, Sasi Kumar Murakonda, Vincent Bindschaedler, and Reza Shokri. Enhanced membership inference attacks against machine learning models. In *ACM Conference on Computer and Communications Security (CCS)*, 2022.

[354] Jiayuan Ye, Anastasia Borovykh, Soufiane Hayou, and Reza Shokri. Leave-one-out distinguishability in machine learning. In *International Conference on Learning Representations (ICLR)*, 2024.

[355] Samuel Yeom, Irene Giacomelli, Matt Fredrikson, and Somesh Jha. Privacy risk in machine learning: Analyzing the connection to overfitting. In *IEEE Computer Security Foundations Symposium (CSF)*, 2018.

[356] Samuel Yeom, Irene Giacomelli, Alan Menaged, Matt Fredrikson, and Somesh Jha. Overfitting, robustness, and malicious algorithms: A study of potential causes of privacy risk in machine learning. *Journal of Computer Security*, 2020.

[357] Samuel Yeom, Irene Giacomelli, Alan Menaged, Matt Fredrikson, and Somesh Jha. Overfitting, robustness, and malicious algorithms: A study of potential causes of privacy risk in machine learning. *Journal of Computer Security*, 2020.

[358] Dong Yi, Zhen Lei, Shengcai Liao, and Stan Z Li. Learning face representation from scratch. *arXiv preprint arXiv:1411.7923*, 2014.

[359] Soma Yokoi and Issei Sato. Bayesian interpretation of sgd as ito process. *arXiv:1911.09011*, 2019.

[360] Zhiyuan Yu, Yuhao Wu, Ning Zhang, Chenguang Wang, Yevgeniy Vorobeychik, and Chaowei Xiao. CodeIPPrompt: Intellectual property infringement assessment of code language models. In *International Conference on Machine Learning (ICML)*, 2023.

[361] Ann Yuan, Andy Coenen, Emily Reif, and Daphne Ippolito. Wordcraft: story writing with large language models. In *27th International Conference on Intelligent User Interfaces*, 2022.

[362] Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Russ R Salakhutdinov, and Alexander J Smola. Deep sets. *Advances in Neural Information Processing Systems*, 2017.

[363] Santiago Zanella-Béguelin, Lukas Wutschitz, Shruti Tople, Victor Rühle, Andrew Paverd, Olga Ohrimenko, Boris Köpf, and Marc Brockschmidt. Analyzing information leakage of updates to natural language models. In *ACM Conference on Computer and Communications Security (CCS)*, 2020.

[364] Santiago Zanella-Béguelin, Lukas Wutschitz, Shruti Tople, Ahmed Salem, Victor Rühle, Andrew Paverd, Mohammad Naseri, Boris Köpf, and Daniel Jones. Bayesian estimation of differential privacy. In *International Conference on Machine Learning (ICML)*, 2023.

[365] Sajjad Zarifzadeh, Philippe Liu, and Reza Shokri. Low-cost high-power membership inference by boosting relativity. In *International Conference on Machine Learning (ICML)*, 2024.

[366] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning (still) requires rethinking generalization. *Communications of the ACM*, 2021.

[367] Chiyuan Zhang, Daphne Ippolito, Katherine Lee, Matthew Jagielski, Florian Tramèr, and Nicholas Carlini. Counterfactual memorization in neural language models. In *Advances in Neural Information Processing Systems*, 2023.

[368] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations (ICLR)*, 2018.

[369] Minxing Zhang, Zhaochun Ren, Zihan Wang, Pengjie Ren, Zhumin Chen, Pengfei Hu, and Yang Zhang. Membership inference attacks against recommender systems. In *ACM Conference on Computer and Communications Security (CCS)*, 2021.

[370] Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. Opt: Open pre-trained transformer language models, 2022.

[371] Wanrong Zhang, Shruti Tople, and Olga Ohrimenko. Leakage of dataset properties in Multi-Party machine learning. In *USENIX Security Symposium*, 2021.

[372] Wanrong Zhang, Olga Ohrimenko, and Rachel Cummings. Attribute privacy: Framework and mechanisms. In *ACM Conference on Fairness, Accountability, and Transparency (FAccT)*, 2022.

[373] Yonggang Zhang, Mingming Gong, Tongliang Liu, Gang Niu, Xinmei Tian, Bo Han, Bernhard Schölkopf, and Kun Zhang. Adversarial robustness through the lens of causality. In *International Conference on Learning Representations (ICLR)*, 2022.

[374] Zhikun Zhang, Min Chen, Michael Backes, Yun Shen, and Yang Zhang. Inference attacks against graph neural networks. In *USENIX Security Symposium*, 2022.

[375] Jiawei Zhou, Alexander M Xu, Zhiying amd Rush, and Minlan Yu. Automating Botnet Detection with Graph Neural Networks. *AutoML for Networking and Systems Workshop of MLSys 2020 Conference*, 2020.

[376] Junhao Zhou, Yufei Chen, Chao Shen, and Yang Zhang. Property inference attacks against GANs. In *Network and Distributed Systems Security Symposium*, 2022.

[377] Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He. A comprehensive survey on transfer learning. *Proceedings of the IEEE*, 2020.

[378] Liu Ziyin, Kangqiao Liu, Takashi Mori, and Masahito Ueda. Strength of minibatch noise in SGD. In *International Conference on Learning Representations (ICLR)*, 2021.

[379] Liu Ziyin, Hongchao Li, and Masahito Ueda. Law of balance and stationary distribution of stochastic gradient descent. *arXiv:2308.06671*, 2023.

[380] Yang Zou, Zhikun Zhang, Michael Backes, and Yang Zhang. Privacy analysis of deep learning in the wild: Membership inference attacks against transfer learning. *arXiv preprint arXiv:2009.04872*, 2020.