TMI! Finetuned Models Leak Private Information from their Pretraining Data

John Abascal
Northeastern University
Boston, Massachusetts, USA
abascal.j@northeastern.edu

Stanley Wu
University of Chicago
Chicago, Illinois, USA
stanley.wu@uchicago.edu

Alina Oprea
Northeastern University
Boston, Massachusetts, USA
a.oprea@northeastern.edu

Jonathan Ullman
Northeastern University
Boston, Massachusetts, USA
jullman@ccs.neu.edu

ABSTRACT

Transfer learning has become an increasingly popular technique in machine learning as a way to leverage a pretrained model trained for one task to assist with building a finetuned model for a related task. This paradigm has been especially popular for *privacy* in machine learning, where the pretrained model is considered public, and only the data for finetuning is considered sensitive. However, there are reasons to believe that the data used for pretraining is still sensitive, making it essential to understand how much information the finetuned model leaks about the pretraining data. In this work we propose a new membership-inference threat model where the adversary only has access to the finetuned model and would like to infer the membership of the pretraining data. To realize this threat model, we implement a novel metaclassifier-based attack, **TMI**, that leverages the influence of memorized pretraining samples on predictions in the downstream task. We evaluate **TMI** on both vision and natural language tasks across multiple transfer learning settings, including finetuning with differential privacy. Through our evaluation, we find that **TMI** can successfully infer membership of pretraining examples using query access to the finetuned model.

1 INTRODUCTION

Transfer learning has become an increasingly popular technique in machine learning as a way to leverage a model trained for one task to assist with building a model for a related task. Typically, we begin with a large *pretrained model* trained with abundant data and computation, and use it as a starting point for training a *finetuned model* to solve a new task where data and computation is scarce. This paradigm has been especially popular for *privacy* in machine learning [1–6], because the data for pretraining is often considered public and thus the pretrained model provides a good starting point before we even have to touch sensitive data.

Although the data used to pretrain large models is typically scraped from the Web and publicly accessible, there are several reasons to believe that this data is still sensitive [7]. For example, personal data could have been published without consent by a third party who they trusted to keep their data private, and even ubiquitous and thoroughly examined pretraining datasets like ImageNet contain sensitive content [8, 9]. Beyond the privacy risks associated with individuals in the pretraining set, companies who utilize or sell finetuned models may also be at risk for privacy leakage. Consider the following example:

Example 1.1. Companies have large, web scraped datasets that are proprietary and remain internal (e.g., Google's JFT-300 [10]). These datasets are used to train models that can be finetuned by individual teams within the company for their specific needs. These pretrained models are also hosted as a service where smaller companies can receive a finetuned model without ever seeing the pretrained model itself. For example, Google's Vertex AI [11] allows smaller companies and individuals to upload their data and receive access to query the finetuned model as an endpoint. When these finetuned models are hosted publicly, they may leak sensitive information about the proprietary pretraining datasets on which they were trained.

Thus, a central question we attempt to understand in this work is: *How much sensitive information does a finetuned model reveal about the data that was used for* pretraining*?* We attempt to answer this question in both the settings where privacy preserving techniques have and have not been used to finetune the pretrained model. Examining these two settings leads to another research problem: Given that pretraining datasets have been shown to contain sensitive information [8, 9, 12], using the privacy preserving finetuning techniques described in prior work [1–6] may not provide meaningful privacy guarantees in practice.

Example 1.2. Using the thought experiment from [7], consider a large, pretrained model, owned by Company A, that contains an individual's sensitive data record. Suppose that this pretrained model is finetuned by Company B using *differential privacy* [13] with ($\varepsilon = 0.5$, $\delta = 10^{-5}$) on a sensitive downstream task. Considering that the open-source variants of these models' pretraining datasets can exceed 5 TB (over 1 trillion tokens) in size [14], it is likely that any given individual's data record can be present in both the pretraining and finetuning datasets. Thus, because differential privacy is not necessarily robust to preprocessing, the privacy guarantee from finetuning may not hold for individuals whose data record is present in both datasets.

Proceedings on Privacy Enhancing Technologies 2024(3), 202–223

https://doi.org/10.56553/popets-2024-0075

To this end, we will also attempt to answer the following question: *Does using differential privacy during finetuning always provide its stated privacy guarantee?*

We study these questions via *membership-inference (MI) attacks*. A MI attack allows an adversary with access to the model to determine whether or not a given data point was included in the training data. These privacy attacks were first introduced by Homer et al. [15] in the context of genomic data, formalized and analyzed statistically by Sankararaman et al. [16] and Dwork et al. [17], and later applied in machine learning applications by Shokri et al. [18]. MI attacks have been extensively studied in machine learning applications, such as computer vision [19], contrastive learning [20], generative adversarial networks [21], and federated learning [22]. The success of MI attacks makes it clear that the pretrained model will leak information about the pretraining data. However, the process of finetuning the model will obscure information about the original model, and there are no works that study MI attacks that use the *finetuned model* to recover *pretraining data*.

We create a novel, metaclassifier-based membership-inference attack, **T**ransfer **M**embership **I**nference (**TMI**) to circumvent the challenges that arise when trying to adapt prior attacks to asses privacy leakage in this new setting where the adversary has query access only to the finetuned model. The goal of our new membership-inference adversary is to infer whether or not specific individuals were included in the pretraining set of the finetuned machine learning model. This setting stands in contrast to prior membership-inference attacks, as it restricts the adversary from directly querying the model trained on the specific dataset they wish to perform membership-inference on. State-of-the-art, black-box MI attacks rely on a model's prediction confidence with respect to the ground truth label, but the finetuned model does not necessarily have the ground truth label in its range. Thus, our attack leverages how individual samples from pretraining influence predictions on the downstream task by observing entire prediction vectors from the finetuned model. More concretely, **TMI** constructs a dataset of prediction vectors from queries to finetuned shadow models in order to train a metaclassifier that can infer membership.

We comprehensively evaluate **TMI** on pretrained CIFAR-100 [23] and Tiny ImageNet [24] vision models, transferred to multiple downstream tasks. In our experiments with Tiny ImageNet, we evaluate the ability of **TMI** to infer membership on models finetuned on Caltech 101 [25]. Our pretrained CIFAR-100 models are finetuned on three downstream datasets of varying similarity to the pretraining data. In order of similarity to CIFAR-100, we evaluate **TMI** on models finetuned on a coarse-labeled version of CIFAR-100, CIFAR-10 [23], and the Oxford-IIT Pet dataset [26]. We also evaluate an extension of **TMI** on finetuned versions of publicly available large language language models, which are pretrained on WikiText-103 [27]. To measure the success of our attack we use several metrics, such as AUC and true positive rates at low false positive rates. To demonstrate the prevalence of privacy leakage with respect to pretraining data in finetuned models, we run **TMI** on target models with different finetuning strategies and settings with limited adversarial capabilities. We compare our results to both a simple adaptation of the likelihood ratio attack [19] to the transfer learning setting and a membership inference attack that has direct access to the pretrained model.

Our Contributions. We summarize our main contributions to the study of membership-inference attacks as follows:

- We investigate privacy leakage in the transfer learning setting, where machine learning models are finetuned on downstream tasks with and without differential privacy.
- We introduce a new threat model, where the adversary only has query access to the finetuned target model.
- We propose a novel membership-inference attack, **TMI**, that leverages all of the information available to the black-box adversary to infer the membership status of individuals in the pretraining set of a finetuned machine learning model.
- We provide theoretical results for membership-inference attacks on mean estimation to support and explain our findings.
- We evaluate our attack across four vision datasets of varying similarity to the two pretraining tasks and several different transfer learning strategies. We show that there is privacy leakage even in cases where the pretraining task provides little benefit to the downstream task or the target model was finetuned with differential privacy. We also show that membership in the pretraining dataset can lead to unexpected privacy leakage when finetuning with differential privacy.
- We study privacy leakage of finetuned models in the natural language domain by evaluating our attack on two finetuned versions of a publicly available foundation model.

2 BACKGROUND AND RELATED WORK

We provide the necessary background on machine learning, privacy in machine learning, and related work on existing inference attacks.

2.1 Machine Learning Background and Notation

In our attacks, we assume that the target models are classifier neural networks trained in a supervised manner. A neural network classifier with parameters θ is a function, $f_\theta : \mathcal{X} \to [0,1]^K$ that maps data points $x \in \mathcal{X}$ to a probability distribution over K classes. In the supervised learning setting, we are given a dataset of labeled (x, y) pairs D drawn from an underlying distribution $\mathbb{D}$ and a training algorithm $\mathcal{T}$. The parameters of the neural network are then learned by running the training algorithm over the dataset, which we will denote $f_\theta \leftarrow \mathcal{T}(D)$. A popular choice for the training algorithm is stochastic gradient descent (SGD), which minimizes a loss function $\mathcal{L}$ over the labeled dataset D by iteratively updating the models parameters θ:

$$\theta_{i+1} \leftarrow \theta_i - \frac{\eta}{m} \sum_{(x,y)\in D} \nabla_\theta \mathcal{L}(f_\theta(x), y)$$

where m is the dataset size, η is a tunable parameter called the learning rate. In our setting, we define the loss function $\mathcal{L}$ to be the cross-entropy loss:

$$\mathcal{L}(f_{\theta(x)}, y) = -\sum_{j=1}^{K} \mathbb{1}_{\{j=y\}} \log(p_j)$$

where p_j is the model's prediction probability for class j.

2.1.1 Scaling Model Confidences.

The classifier models we consider output a vector of probabilities, $\vec{y}$, where each entry y_i corresponds to the *model's prediction confidence* with respect to label, i. This is done by applying the `softmax` activation function to the model's final layer. Given a vector of logits, $\vec{z}$ (i.e. the model's final layer), we define $\texttt{softmax}(\vec{z}) : \mathbb{R}^K \to (0,1)^K$

$$y_i = \texttt{softmax}(\vec{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$$

where K is the number of possible classes.

Prior work [19] has used the logit function, $\texttt{logit}(p) = \log(\frac{p}{1-p})$, to scale model confidences. This scaling yields an approximately normally distributed statistic that can be used to perform a variety of privacy attacks [19, 28, 29]. The logit function is obtained by inverting the `sigmoid` activation function, $\sigma(x) = \frac{1}{1+e^x}$, which is a specific case of `softmax` being used for binary classification.

Following the lead of prior work, we use ϕ to perform our model confidence scaling. We define model confidence scaling $\phi(\vec{y}) : \mathbb{R}^K \to \mathbb{R}^K$ for a prediction vector, $\vec{y}$, as

$$\phi(\vec{y}) = (\texttt{logit}(y_1), \ldots, \texttt{logit}(y_K))$$

2.1.2 Transfer Learning.

Feature extraction and updating a model's pretrained weights are popular transfer learning techniques used to improve a pretrained deep learning model's performance on a specific task. In the classification setting, feature extraction involves freezing a model's pretrained weights and using them to extract relevant features from input data, which are then fed into a linear layer for classification. This technique is useful when working with limited data or when the pretrained model has learned generalizable features that are useful for the target task. On the other hand, finetuning a model by updating its pretrained weights involves taking a pretrained model and training it on a new dataset, often with a smaller learning rate, to adapt it to the new task. This kind of finetuning is more suited for situations where the new task has similar characteristics, but not a direct correspondence, to the original pretraining task.

2.1.3 Differential Privacy.

Differential Privacy [13] is a mathematical definition of privacy that bounds the influence that any single individual in the training data has on the output of the model. Specifically, an algorithm satisfies differential privacy if for any two datasets that differ on one individual's training data, the probability of seeing any set of potential models is roughly the same regardless of which dataset was used in training.

Definition 2.1. A randomized algorithm $\mathcal{M}$ mapping datasets to models satisfies (ε, δ)-*differential privacy* if for every pair of datasets X and X' differing on at most one training example and every set of outputs E,

$$\Pr[\mathcal{M}(X) \in E] \leq e^{\varepsilon} \Pr[\mathcal{M}(X') \in E] + \delta$$

2.2 Related Work

2.2.1 Privacy Attacks on Machine Learning Models.

Deep learning models have been shown to memorize entire individual data points, even in settings where the data points have randomly assigned labels [30]. Prior work has demonstrated the ability of a wide class of deep learning models to perfectly fit training data while also achieving low generalization error [31]. In fact, recent work [32–34] has shown that memorization of training data may actually be necessary to achieve optimal generalization for deep learning models. As a result of this memorization, deep learning models tend to have higher prediction confidence on training data, which makes them highly susceptible to privacy attacks.

The most glaring violations of privacy in machine learning are reconstruction and training data extraction attacks. Early work in data privacy [35] showed that it is possible to reconstruct individuals' data in statistical databases with access to noisy queries. More recently, training data extraction attacks have been shown to be successful when mounted on a variety of deep learning models, including large language models [12] and computer vision models [36].

Other attacks on machine learning models, such as membership-inference [18], property inference [37], and attribute inference [38] attacks are more subtle privacy violations. These attacks exploit vulnerabilities in machine learning models to learn whether or not an individual was in the training set, global properties of the training dataset, and an individual's sensitive attributes. respectively. Recent versions of these attacks typically use a test statistic, such as loss [39] and model prediction confidences [19, 28, 38], to extract private information.

2.2.2 Membership-Inference Attacks.

Membership-inference attacks [15] aim to determine whether or not a given individual's data record was present in a machine learning model's training dataset. These attacks represent a fundamental privacy violation that has a direct connection to differential privacy. Mounting these attacks and learning whether or not an individual was part of a sensitive dataset can serve as the basis for more powerful attacks. For example, prior work has used MI as a step in extracting training data [12]. Because of their simplicity, MI attacks are also a popular way to audit machine learning models for privacy leakage [39–41].

These attacks been extensively studied with two types of adversarial access: black-box query access and white-box access to the machine learning model's parameters [42]. The query access setting has been more thoroughly studied, with attacks spanning several different machine learning domains, such as classification [18, 19, 29, 38, 39], natural language generation [19, 29], and federated learning [22]. Despite there being extensive work on black-box attacks and prior work on MI attacks on pretrained encoders [20], continuously updated models [43], and distilled models [44], there are few works that explore MI in the transfer learning setting where a pretrained model is finetuned on a new task. Zou et al. [45] study MI attacks that target individuals in the finetuning dataset, and Hidano et al. [46] explore ways in which an adversary can leverage control over the transfer learning process to amplify the success of MI attacks on the original model. No works have studied black-box MI attacks on the pretraining dataset of a finetuned machine learning model.

3 THREAT MODEL

Our problem is to determine how much information a finetuned model reveals about the data used in the pretraining phase, and

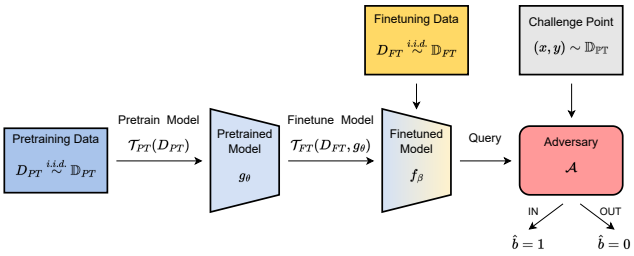


Figure 1: Our New Membership-Inference Threat Model.

whether or not the finetuned model reveals strictly less information than the pretrained model. For this work we study this question using the language of MI attacks [15–17]. In the standard MI experiment and in our newly defined experiment, there is a machine learning model trained on some dataset, and a challenge point that is drawn from the same distribution as the training data. The challenge point is either an element of the training data or an independent point drawn from the same distribution. The attacker, who has access to the model and the challenge point, and knowledge of the distribution, tries to infer which of these two cases holds. In our experiment we separate the construction of the machine learning model into a pretraining phase and a finetuning phase, where the finetuning phase is performed with different data, drawn from a possibly different distribution. This finetuning phase introduces another layer of indirection that prevents the attacker from querying the original pretrained model, and thus potentially makes MI more difficult. Formally, our threat model, visualized in Figure 1, is described by the following game between a *challenger* C and an *adversary* $\mathcal{A}$:

MI Security Game with a Finetuned Target Model

(1) The challenger receives a dataset D_{PT} comprised of points sampled i.i.d. from some distribution $\mathbb{D}_{PT}$, and a pretrained model $g_\theta \leftarrow \mathcal{T}_{PT}(D_{PT})$.
(2) The challenger draws i.i.d. samples from another distribution $\mathbb{D}_{FT}$ to create a dataset D_{FT} and finetunes the model on D_{FT} using its pretrained weights, θ, to obtain a new model $f_\beta \leftarrow \mathcal{T}_{FT}(D_{FT}, g_\theta)$.
(3) The challenger randomly selects $b \in \{0, 1\}$. If $b = 0$, the challenger samples a point (x, y) from $\mathbb{D}_{PT}$ uniformly at random, such that $(x, y) \notin D_{PT}$. Otherwise, the challenger samples (x, y) from D_{PT} uniformly at random.
(4) The challenger sends the point, (x, y) to the adversary.
(5) The adversary, using the challenge point, sampling access to $\mathbb{D}_{PT}$ and $\mathbb{D}_{FT}$, and query access to f_β, produces a bit $\hat{b}$.
(6) The adversary wins if $b = \hat{b}$ and loses otherwise.

In our security game, we assume that the adversary has query access to the finetuned target model f_β and knowledge of the pretraining data distribution $\mathbb{D}_{PT}$. Because we will be training *shadow models* [18] to perform our MI attack, the adversary also requires knowledge of the underlying distribution from which the finetuning dataset is sampled, $\mathbb{D}_{FT}$, and knowledge of the target model's architecture and training algorithm. MI attacks vary in what they assume about the distribution and training algorithm [17], and some degree of knowledge is necessary. The knowledge we assume is the same as many other works on MI (e.g. [18–20, 29, 38, 44, 47]). We also assume that the adversary's queries to the target model return numerical confidence scores for each label rather than just a single label, similar to prior privacy attacks [18, 19, 29, 38].

It should be noted that the adversary considered in this work is, in fact, a stronger adversary than some of these prior works individually. For example, we require query access to *both* the pretraining and finetuning data distributions, while [18] and [19] only require access to one training distribution. Additionally, our attack algorithm requires us to train shadow models, which can be computationally expensive. Because we introduce the first MI threat model in this setting, we consider this strong adversary as a reasonable starting point.

4 METHODOLOGY

In this section, we will propose attacks that follow the threat model defined in Section 3. First, we will motivate our attack with theoretical results for membership-inference attacks under distribution shift. Then, we will provide a simple adaptation of an existing MI attack and describe issues that arise when trying to incorporate more information about target model queries into an attack implementation. Lastly, we will detail our metaclassifier-based approach to performing black-box MI attacks on finetuned models.

Algorithm 1 `train_shadow_models`$(x,\ b)$
Our shadow model training procedure considers both the pretraining and finetuning phases to mimic the behavior of the target model on a challenge point.

Require: Query access to both $\mathbb{D}_{PT}$ and $\mathbb{D}_{FT}$ and a fixed dataset size $S = \frac{1}{2}|\mathbb{D}_{PT}|$
models $\leftarrow \{\}$
datasets $\leftarrow \{\}$
for N times **do**
  Draw S i.i.d. samples from $\mathbb{D}_{PT}$ to construct $\tilde{D}_{PT}$
  datasets $\leftarrow$ datasets $\cup \{\tilde{D}_{PT}\}$
  $g \leftarrow \mathcal{T}(\tilde{D}_{PT})$
  Sample $\tilde{D}_{FT}$ i.i.d. using query access to $\mathbb{D}_{FT}$
  $f \leftarrow \mathcal{T}(g,\ \tilde{D}_{FT})$ ▷ Finetune g on $\tilde{D}_{FT}$
  models $\leftarrow$ models $\cup \{f\}$
return models, datasets

4.1 Membership Inference Under Distribution Shift

To motivate a membership-inference attack on finetuned deep learning models, we will first consider the simplified setting of mean estimation. A more detailed explanation, along with the proofs for the statements in this section, can be found in Appendix A.

Consider two datasets, $X \overset{\text{iid}}{\sim} \mathcal{N}(\mu,\ \mathbb{I}_d)$ and $Y \overset{\text{iid}}{\sim} \mathcal{N}(\mu + \nu,\ \mathbb{I}_d)$ where $|X| = n$, $|Y| = m$ such that $n \gg m$, and ν is a parameter that controls distribution shift. In this setting, the means of X and Y are related, and we would like to estimate the mean of Y, which has limited data, using the additional data from X. We define the estimator of $\mu + \nu$ as a combination of the empirical means of X and Y:

$$\hat{\mu} = \alpha\bar{x} + (1-\alpha)\bar{y}$$

where $\alpha \in [0,1]$ and $\bar{x}$, $\bar{y}$ are the empirical means of X and Y, respectively. Note that $\hat{\mu}$ has expected value and covariance

$$\mathbb{E}(\hat{\mu}) = \mu + (1-\alpha)\nu$$

$$\mathrm{Cov}(\hat{\mu}) = \left(\frac{\alpha^2}{n} + \frac{(1-\alpha)^2}{m}\right) \cdot \mathbb{I}_d = \tilde{\alpha} \cdot \mathbb{I}_d$$

Suppose the challenger from the security game detailed in Section 3 releases the statistic $\hat{\mu}$ and we, as the adversary, would like to learn samples' membership statuses with respect to the auxiliary (pretraining) data, X. One possible way to do this would be the following: Assume the adversary knows $\mathbb{E}(\hat{\mu})$ and μ. Then, for some challenge point, c the adversary can compute the test statistic

$$z = \langle \hat{\mu} - \mathbb{E}(\hat{\mu}) \ , \ c - \mathbb{E}(c) \rangle$$

This specific choice of test statistic is motivated by prior work on membership-inference attacks on published statistics [48]. Subtracting the expectation of each term allows the adversary to observe whether the noise from computing $\hat{\mu}$ is correlated with the noise from sampling c. Thus, the test statistic z is a real number that measures the correlation between the challenge point and the published statistic, $\hat{\mu}$. The adversary can then choose a threshold τ such that if $z > \tau$, they will predict that the challenge point was IN (i.e. $c \in X$). Else, the adversary will predict that the challenge point was OUT (i.e. $c \sim \mathcal{N}(\mu, \mathbb{I}_d)$)

We will now show the ability of our attack to determine the membership status of the challenge point c as a function of the parameter α. To this end, we start by computing the expectation and variance of the test statistic, z, when c is either OUT or IN.

Lemma 4.1. *If c is OUT, then*

$$\mathbb{E}(z) = 0 \quad \textit{and} \quad \mathrm{Var}(z) = d\tilde{\alpha},$$

and if c is IN, then

$$\mathbb{E}(z) = \frac{\alpha d}{n} \quad \textit{and} \quad \mathrm{Var}(z) = d\tilde{\alpha} + \frac{2d\alpha^2}{n^2}$$

This lemma tells us that as long as the noise scale doesn't exceed the difference in means, it is straightforward to determine whether c is IN or OUT. When $\alpha \to 0$, the published statistic is no longer encoding any information about X. Thus, the noise completely masks the difference in means, as shown in Figure 2. Conversely, as $\alpha \to 1$, we observe higher separation between the distributions of IN and OUT test statistics.

Using Lemma 4.1, we can analyze the performance (AUC) of the adversary's distinguishing test as a function of the parameter α. To do this, we use the fact that the AUC of a classifier is equal to the probability that the classifier's prediction on a randomly chosen positive (IN) sample is greater than the prediction on a randomly chosen negative (OUT) sample [49]. Here, we use the assumption that the test statistic is normal. Because z is the inner product of two high dimensional Gaussian vectors, and thus the sum of many i.i.d. Gaussian random variables, as $d \to \infty$, z is normally distributed.

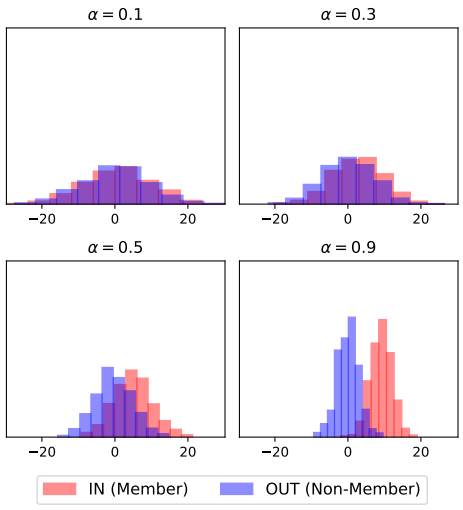


Figure 2: Distribution of the Test Statistic, z, for Multiple Values of α

Lemma 4.2. *Assume that the test statistic, z, is normally distributed. Then, for a fixed α, the AUC of our membership-inference attack can be written as*

$$AUC = \frac{1}{2}\left(1 + \mathtt{erf}\left(\frac{\alpha d}{2\sqrt{d(\tilde{\alpha}n^2 + \alpha^2)}}\right)\right)$$

While it seems as if the attack's success is independent of the magnitude of the distribution shift, $\|\nu\|_2$, it is important to note that α should be set by the challenger such that the error on the new task (namely, estimating the mean of the new dataset, Y) is minimized. In this particular setting, α would be chosen to minimize the mean squared error between $\hat{\mu}$ and the mean of Y, $\mu + \nu$. The proof for the optimal setting of α can be found in Appendix A.3.

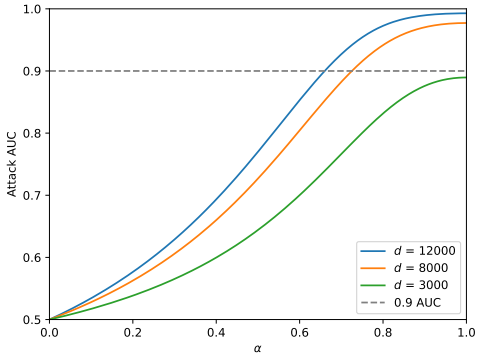


Figure 3: AUC of our Membership-Inference Attack on Mean Estimation as a Function of α.

Figure 3 visualizes the attack's AUC from Lemma 4 as a function of the parameter α. Here, the parameters n, m, and d are fixed. Our choices of the data's dimension, d, are motivated by the dimension

of the data in our evaluation on vision models (Section 5.3). We observe that there is a rapid increase in AUC as more information about X is preserved in the public statistic (i.e. as α increases). In this figure, the value for α when $d = 12,000$ and $\|v\|_2 = 5$ that minimizes the error on the new task, Y, is roughly 0.77. This α corresponds to an AUC of 0.96. If we make the distribution shift larger, say $\|v\|_2 = 10$, the optimal value of α is 0.51, which corresponds to an AUC of 0.78. This shows that the success of our attack on mean estimation depends on the extent to which we combine the means of X and Y using the parameter α, which is based on the similarity of the "pretraining" data X and the "finetuning" data Y.

4.2 Adapting an Existing Attack

As a first attempt to create an effective membership-inference attack on finetuned machine learning models, we can consider an adaptation of the *likelihood ratio attack* (LiRA) proposed by Carlini et al. [19]. In this attack (Algorithm 2), the adversary observes the target model's prediction confidence on a challenge point with respect to the true label of the challenge point. Because the model's confidence with respect to a given label is approximately normally distributed, Carlini et al. perform a likelihood ratio test to infer the challenge point's membership status, using a set of shadow models to parameterize the IN and OUT distributions.

In our setting, these shadow models are first trained on datasets drawn from $\mathbb{D}_{PT}$, then finetuned on a dataset drawn from $\mathbb{D}_{FT}$ (Algorithm 1). Because the ground truth label of the challenge point drawn from $\mathbb{D}_{PT}$ is not necessarily in the range of our finetuned target model we cannot perform the likelihood ratio test with respect to the observed confidence on the point's true label. Instead, we can adapt the attack to use the label predicted by the target model with the highest confidence, $\hat{y}$. To do this, we store the entire prediction vector for each query to our shadow models, and only use the scaled model confidences at index $\hat{y}$, denoted $f(x)_{\hat{y}}$, of the prediction vectors. We follow the lead of Carlini et al. [19] and query each shadow and target model on M random augmentations of the challenge point and fit M-dimensional multivariate normal distributions to the scaled model confidences we aggregate to improve attack success.

4.3 Issues with Adapting LiRA

While this adaptation of LiRA is somewhat effective at inferring membership (Figures 5 and 6), it only captures how the pretraining dataset influences model's predictions with respect to a single label in the downstream dataset. Because the purpose of pretraining is to extract and learn general features that can be used in several downstream tasks, one would expect that the weights of a pretrained model have some impact on *all* of a finetuned model's prediction confidences. For example, Figure 4 shows that the presence of a specific image labeled as "dugong" in the training set makes finetuned models, which cannot themselves predict the label "dugong", more confident on their downstream prediction of "elephant" and "platypus". Meanwhile, the presence of this image in the training dataset has little to no impact on the downstream label "scissors".

Furthermore, if we observe the distribution of scaled model confidences over our shadow models, we see that it is approximately normal regardless of the choice of label. This may lead one to believe

Algorithm 2 Adapted LiRA

We adapt the MI attack shown in [19] by using the label which the target model predicted most confidently instead of the ground truth label.

Require: A finetuned target model f_β, a challenge point $x \leftarrow \mathbb{D}_{PT}$, and models and datasets (i.e. the output of `train_shadow_models()`)

$\text{preds}_{\text{in}} \leftarrow \{\}, \text{preds}_{\text{out}} \leftarrow \{\}$
$\vec{v}_{\text{obs}} \leftarrow f_\beta(x)$ ▷ Query the target model on x
$\text{conf}_{\text{obs}} \leftarrow \texttt{logit}(\max_i \vec{v}_{\text{obs},i})$ ▷ Store max confidence score
$\hat{y} \leftarrow \arg\max_i \vec{v}_{\text{obs},i}$ ▷ Store most confident predicted label
$i \leftarrow 1$ ▷ Index for saved shadow models and datasets
for N times **do**
  if $x \in \text{datasets}_i$ **then** ▷ If x is IN w.r.t. shadow model i
    $f_{\text{in}} \leftarrow \text{models}_i$
    $\text{conf}_{\text{in}} \leftarrow \texttt{logit}(f_{\text{in}}(x)_{\hat{y}})$ ▷ Query f_{in} on x
    $\text{preds}_{\text{in}} \leftarrow \text{preds}_{\text{in}} \cup \{\text{conf}_{\text{in}}\}$ ▷ Aggregate confidences
  else if $x \notin \text{datasets}_i$ **then**
    $f_{\text{out}} \leftarrow \text{models}_i$
    $\text{conf}_{\text{out}} \leftarrow \texttt{logit}(f_{\text{out}}(x)_{\hat{y}})$
    $\text{preds}_{\text{out}} \leftarrow \text{preds}_{\text{out}} \cup \{\text{conf}_{\text{out}}\}$
$\mu_{\text{in}} \leftarrow \texttt{mean}(\text{preds}_{\text{in}}), \quad \mu_{\text{out}} \leftarrow \texttt{mean}(\text{preds}_{\text{out}})$
$\sigma^2_{\text{in}} \leftarrow \texttt{var}(\text{preds}_{\text{in}}), \quad \sigma^2_{\text{out}} \leftarrow \texttt{var}(\text{preds}_{\text{out}})$
return $\dfrac{p(\text{conf}_{\text{obs}}|\mathcal{N}(\mu_{\text{in}}, \sigma^2_{\text{in}}))}{p(\text{conf}_{\text{obs}}|\mathcal{N}(\mu_{\text{out}}, \sigma^2_{\text{out}}))}$

that the correct adaptation of LiRA to our setting would be to fit a multivariate normal distribution to the entire prediction vectors output by our shadow models. The assumption that the adversary only receives model confidences interferes with this seemingly better adaptation because of the softmax activation function. When softmax is applied, it converts the logit vector $\vec{z}$ into a probability distribution, $\vec{y}$, over the labels. Thus, the entries of $\vec{y}$ can be written as

$$\vec{y} = (p_1, p_2, \ldots, p_K) \in (0,1)^K$$

where K is the number of classes and each p_i denotes the model's confidence on class i. Because the entries of $\vec{y}$ necessarily sum up to 1, any entry p_i can be written as $1 - \sum_{j \neq i} p_j$. When we scale model confidences to compute the individual logits, z_i, any given computed logit can be written as a combination of the others. This means that our computed logits actually lie on a $(K-1)$-dimensional subspace of the K-dimensional space where the model's actual logits lie, and we cannot fit a K-dimensional multivariate normal distribution to all of our models' logit scaled prediction vectors without arbitrarily removing one of the entries in $\vec{y}$.

4.4 Our TMI Attack

Our **T**ransfer **M**embership **I**nference (**TMI**) attack (Algorithm 3) starts with the same shadow model training procedure as Algorithm 2, where the adversary trains shadow models on datasets sampled from $\mathbb{D}_{PT}$ and finetunes them on datasets sampled from $\mathbb{D}_{FT}$. The adversary then queries the challenge point on these shadow models to construct a dataset, D_{meta}, comprised of logits attained

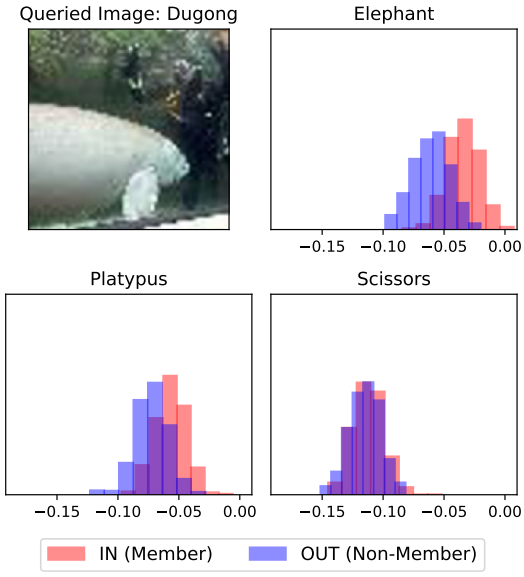


Figure 4: Scaled Model Confidences of Shadow Models Finetuned on Caltech 101 at Multiple Labels when Queried on a Sample from the "Dugong" Class in Tiny Imagenet

from scaling the prediction vectors as described in Section 2.1.1. To construct a distinguishing test that circumvents the issues that arise when attempting to parameterize the distribution of prediction vectors, the adversary trains a *metaclassifier* on a collection of labeled prediction vectors D_{meta}, queries the target model on the challenge point, and scales the target model's prediction vector. Lastly, the adversary queries this observed prediction vector on their metaclassifier, which outputs a score in the interval $[0, 1]$ that indicates the predicted membership status of the challenge point. Using a metaclassifier attack, **TMI**, is still able to leverage the influence of memorized pretraining samples on predictions in the downstream task while not having to arbitrarily discarding one of the entries from the prediction vector.

In our implementation of **TMI** for computer vision models, we train a metaclassifier per challenge point. Because we use a relatively small number of shadow models (64 IN and 64 OUT in total), we leverage random augmentations to construct a larger metaclassifier dataset. Each time we query the target model or our local shadow models, we query M times with different random augmentations of the challenge point, including random horizontal flips and random crops with padding. This yields $M \times 2 \times 64$ prediction vectors for each challenge point. In total, our metaclassifiers are trained on 1024 labeled prediction vectors, 512 labeled 0 to denote "non-member" or OUT and 512 labeled 1 to denote "member" or IN.

Due to computational limitations, we do not pretrain any shadow models for our attacks in the language domain. Rather, we use a publicly hosted pretrained model and finetune it on a downstream task. Without control over pretraining, we cannot produce a metaclassifier dataset with prediction vectors from both IN and OUT shadow models with respect to a single challenge point. This scenario can be represented in Algorithm 1 by ommiting lines 4, 5 and 6, where g refers to the publicly hosted pretrained language model. As a result, we use a *global metaclassifier*, trained on a dataset containing the prediction vectors of *all* challenge points, to produce membership scores.

Algorithm 3 TMI Metaclassifier Attack

We pretrain shadow models with and without the challenge point and finetune them using query access to $\mathbb{D}_{FT}$ to estimate the target model's prediction behavior. Using the prediction vectors of our shadow models on the challenge point, we generate a dataset to train a metaclassifier to determine the challenge point's membership status.

Require: A finetuned target model f_β, a challenge point $x \leftarrow \mathbb{D}_{PT}$, and models and datasets (i.e. the output of `train_shadow_models()`)

1: $\text{preds}_{\text{in}} \leftarrow \{\}, \text{preds}_{\text{out}} \leftarrow \{\}$
2: $i \leftarrow 1$ ▷ Index for saved shadow models
3: **for** N times **do**
4: **if** $x \in \text{datasets}_i$ **then** ▷ If x is IN w.r.t. shadow model i
5: $f_{\text{in}} \leftarrow \text{models}_i$
6: $\vec{v}_{\text{in}} \leftarrow \phi(f_{\text{in}}(x))$ ▷ Query IN model on x
7: $\text{preds}_{\text{in}} \leftarrow \text{preds}_{\text{in}} \cup \{(\vec{v}_{\text{in}}, 1)\}$ ▷ Store and label the prediction vector
8: **else if** $x \notin \text{datasets}_i$ **then**
9: $f_{\text{out}} \leftarrow \text{models}_i$
10: $\vec{v}_{\text{out}} \leftarrow \phi(f_{\text{out}}(x))$
11: $\text{preds}_{\text{out}} \leftarrow \text{preds}_{\text{out}} \cup \{(\vec{v}_{\text{out}}, 0)\}$
12: $i \leftarrow i + 1$
13: $D_{\text{meta}} = \text{preds}_{\text{in}} \cup \text{preds}_{\text{out}}$ ▷ Construct the metaclassifier dataset
14: $\mathcal{M} \leftarrow \mathcal{T}(D_{\text{meta}})$ ▷ Train a binary metaclassifier
15: $\vec{v}_{\text{obs}} = \phi(f_\beta(x))$ ▷ Query the target model on x
16: Output $\mathcal{M}(\vec{v}_{\text{obs}})$

5 TMI EVALUATION

We evaluate the performance of our **TMI** attack on image models with two pretraining tasks and four downstream tasks and public, pretrained language models with one pretraining task and two downstream tasks. We evaluate the success of our attack as a function of the number of updated parameters, and we choose downstream tasks with differing similarity to the pretraining task to show how attack success depends on the relevance of the pretraining task to the downstream task. Additionally, we observe the success of our attack when differential privacy [13] is used in the finetuning process, which is an increasingly popular technique to maintain utility while preserving the privacy of individuals in the dataset of downstream task [2–6, 50].

This section presents the results of our evaluation of **TMI** and addresses the following research questions with respect to the datasets in our experiments:

- **Q1**: Can finetuned models leak private information about their pretraining datasets via black-box queries?
- **Q2**: Does updating a model's pretrained parameters instead of freezing them prevent privacy leakage?

Q3: Does the similarity between the pretraining and downstream task affect the privacy risk of the pretraining set?
Q4: Can the attack be generalized to domains other than vision?
Q5: Is it feasible to mount our attack on finetuned models that are based on publicly hosted foundation models?
Q6: Is privacy leakage present even when a model is finetuned using differential privacy?

5.1 Datasets and Models

In this section we will discuss the datasets used in our evaluation of **TMI**. We will also discuss our choices of pretraining and downstream tasks used in our evaluation.

5.1.1 Datasets. We pretrain our small vision models on CIFAR-100 [23] and finetune them on a coarse-labeled version of CIFAR-100, CIFAR-10 [23], and Oxford-IIIT Pet [26]. Our larger vision models are pretrained on Tiny ImageNet [24] and finetuned on Caltech 101 [25]. For our language tasks, we use publicly available pretrained WikiText-103 [27] models and finetune them on DBpedia [51] and Yahoo Answers [52] topic classification datasets. A detailed description of the datasets used in our evaluation can be found in Appendix C.1.

5.1.2 Models. For our vision tasks, we use the ResNet-34 [53] and Wide ResNet-101 [54] architectures. The ResNet architecture has been widely used in various computer vision applications due to its superior performance and efficiency. ResNet is a convolutional neural network architecture that uses residual blocks, allowing it to effectively handle the complex features of images and perform well on large-scale datasets.

For our language tasks, we use the Transformer-XL [55] model architecture. In particular, we use the pretrained Transformer-XL model from Hugging Face, which is trained on WikiText-103 [27], as our initialization for the downstream tasks. We finetune our pretrained language model architectures on the DBpedia ontology classification and Yahoo Answers topic classification datasets.

5.1.3 Shadow Model Training.

Here, we describe the shadow model training procedure for our vision tasks, which comprise the majority of our experiments. The details for how we train shadow models for our language task can be found in Section 5.3.3. A full description can be found in Appendix C.2

Our shadow model training involves two phases: pretraining and finetuning. In the pretraining phase, we train 129 models are trained on random 50% splits of CIFAR-100 and Tiny ImageNet using SGD with weight decay and cosine annealing for 100 epochs (ResNet-34) or 200 epochs (Wide ResNet-101). Standard data augmentations are applied during training and querying. In the second phase, the shadow models have a subset of their weights frozen and their classification layer swapped to match new task. Then, they are finetuned on random subsets of downstream task datasets. During pretraining, we designate a random set of challenge points to evaluate the **TMI** attack. Because we train on 50% splits of the pretraining data, approximately half of the challenge points are IN and OUT for each shadow model. In each experiment, we select a shadow model to be the target and use the remaining 128 to mount our attack, yielding a total of 128 trials.

5.2 Metrics

To evaluate the performance of **TMI**, we use a set of metrics that are commonly used in the literature. The first metric is *balanced attack accuracy*, which measures the percentage of samples for which our attack correctly identifies membership status. Although balanced accuracy is a common metric used to evaluate MI attacks [18, 38, 47, 56], prior work [19] argues that it is not sufficient by itself to measure the performance of MI attacks as privacy is not an average case metric [57]. Therefore, we also evaluate our attack using the *receiver operating characteristic* (ROC) curve.

The ROC curve provides us with several additional metrics that we can use to evaluate the performance of **TMI**. In our evaluation, we plot the ROC curve on a log-log scale to highlight the true positive rate (TPR) at low false positive rates (FPR), and we measure the area under the curve (AUC) as a summary statistic. Additionally, we report the TPR at low, fixed FPR of 0.1% and 1%. These metrics give us a more complete picture of how well **TMI** performs in different scenarios.

5.3 Experimental Results

In this section, we will discuss the performance of our attack on a variety of target models with different finetuning streategies. We consider models finetuned using feature extraction, models finetuned by updating pretrained weights, models finetuned with differential privacy, and publicly hosted pretrained models.

During training, we designate 1000 and 2000 samples to be challenge points for CIFAR-100 and Tiny ImageNet, respectively, and we run our attack for each of these challenge points on 128 different target models. We compare our results to performing LiRA [19] directly on the pretrained model (i.e., the adversary has access to the model before it was finetuned) to provide an upper bound on our attack's performance.

5.3.1 Feature Extraction.

Q1: Can finetuned models leak private information about their pretraining datasets via black-box queries?

To answer this research question, we evaluate the success of our **TMI** attack on models finetuned without updating any of the pretrained parameters (i.e. feature extraction). We consider three tasks in our experiments where feature extraction is used to finetune our target model: Coarse CIFAR-100, CIFAR-10, and Caltech 101. Because feature extraction relies on the pretrained model being relevant to the downstream task, we choose the two most similar downstream tasks to pretraining. Our attack's success depends on the target model having high utility on its respective task, so it is important to ensure that we choose downstream tasks that are similar or relevant to the pretraining task when using feature extraction to finetune models. To transfer the pretrained CIFAR-100 models to Coarse CIFAR-100 and CIFAR-10 and the pretrained Tiny ImageNet models to Caltech 101, we remove the final classification layer, and replace it with a randomly initialized classification layer which has proper number of classes for the new downstream task. The remaining weights are kept frozen throughout training.

As shown in Figure 5, we observe that **TMI** is able to achieve AUC and balanced accuracy (0.78 and 69%) within 0.06 of the adversary which has access to the pretrained model (0.83 and 75%) on the Coarse CIFAR-100 downstream task. On this task, **TMI** also has a

TPR of 5.7% and 16.1% at 0.1% and 1% FPR, respectively. Despite being constrained to only having query access to the finetuned model, Figure 5 shows that the TPR of **TMI** is approximately equal at higher FPR (about 5%) to that of running LiRA directly on the pretrained model.

Furthermore, Table 1 also shows the performance of **TMI** on target models finetuned on the CIFAR-10 and Caltech 101 downstream tasks. When we run **TMI** on the Tiny ImageNet models which are finetuned on Caltech 101, our attack achieves an AUC of 0.914, which is within 6% of the AUC achieved by LiRA directly on the pretrained model. As shown in Table 1, **TMI** has a 207× and 41× higher TPR than FPR when the FPR is fixed at 0.1% and 1%, respectively. On the CIFAR-10 finetuned models we observe that **TMI** achieves a TPR of 2.0% and 8.0% at 0.1% and 1% FPR, respectively. Figure 5 shows that **TMI** also achieves an AUC of 0.684 and a balanced accuracy of 62.4% when the downstream task is CIFAR-10. The lower attack success may be due to the relevance of the features learned during pretraining to the downstream task. For all three tasks, using our adaptation of LiRA and not incorporating information about all of the downstream labels yields *significantly* lower performance by all of our metrics than **TMI**. For example, at 0.1% FPR, our attack has a TPR 14.7×, 8.1×, 6.7× higher than adapted LiRA on Caltech 101, Coarse CIFAR-100 and CIFAR-10, respectively. **TMI** also achieves an AUC about 1.3× higher than adapted LiRA on the Coarse CIFAR-100 and CIFAR-10 tasks and an AUC 1.7× higher on Caltech 101.

Q1 Answer: Yes, it is possible to infer the membership status of an individual in a machine learning model's pretraining set via query access to the finetuned model.

Table 1: TPR at Fixed FPR of TMI and Our Adaptation of LiRA when Pretrained Target Models are Finetuned Using Feature Extraction (Figures 5 and 6)

Task	TPR @ 0.1% FPR	TPR @ 1% FPR
TMI (CIFAR100 → Coarse CIFAR-100)	5.7%	16.1%
TMI (CIFAR100 → CIFAR-10)	2.0%	8.0%
TMI (Tiny ImageNet → Caltech 101)	20.7%	41.5%
Adapted LiRA (CIFAR100 → Coarse CIFAR-100)	0.7%	3.1%
Adapted LiRA (CIFAR100 → CIFAR-10)	0.3%	1.5%
Adapted LiRA (Tiny ImageNet → Caltech 101)	1.4%	0.25%
LiRA Directly on Pretrained Model (CIFAR-100)	15.6%	22.9%
LiRA Directly on Pretrained Model (Tiny ImageNet)	37.2%	60.1%

5.3.2 *Updating Model Parameters.*

Q2: Does updating a model's pretrained parameters instead of freezing them prevent privacy leakage?

CIFAR-10. The ResNet models we pretrain on CIFAR-100 are divided into ResNet blocks or layers, which each contain multiple sub-layers. When finetuning pretrained ResNet models on CIFAR-10, we unfreeze the weights in different subsets of these ResNet layers. More concretely, we observe the performance of our attack on ResNet models which have had their classification layer (feature extraction), last 2 layers (62% of total parameters), and last 3 layers (90% of parameters) finetuned on the downstream task.

In Figure 7, we observe that the AUC and accuracy of **TMI** slightly decrease as we update an increasing number of parameters. We also observe a very slight decrease the TPR at a 1% FPR when the number of finetuned parameters is increased from 2 layers to 3 layers, but TPR decreases at the FPR we consider when comparing to the TPR of **TMI** on models finetuned with feature extraction. Table 2 shows that updating the model's parameters induces a decrease in up to 0.8% at a 0.1% FPR and up to 3.3% at a 1% FPR.

Caltech 101. The Wide ResNet models we pretrain on Tiny ImageNet have a similar architecure to the ResNets in the previous experiments, where each block contains sub-layers. For this architecture, we run our attack on models which have had their classification layer (feature extraction), last 2 layers (34% of total parameters), and last 3 layers (96% of parameters) finetuned on Caltech 101. Figure 15, which corresponds to this experiment, can be found in Appendix D. We observe a similar trend to the previous experiments on CIFAR-10 models, where the attack's success decreases as we increase the number of finetuned parameters. In Table 2 we see that for a fixed FPR of 0.1%, **TMI** has a 20.7%, 11%, and 7.7% TPR when the final, last two, and last three layers are finetuned, respectively. At a 1% FPR, **TMI** has a 41.5%, 26.5% and 20.6% TPR for these three settings. Nevertheless, **TMI** achieves comparable AUC and balanced accuracy metrics to feature extraction when we finetune the majority of model parameters in both the CIFAR-10 and Caltech 101 experiments.

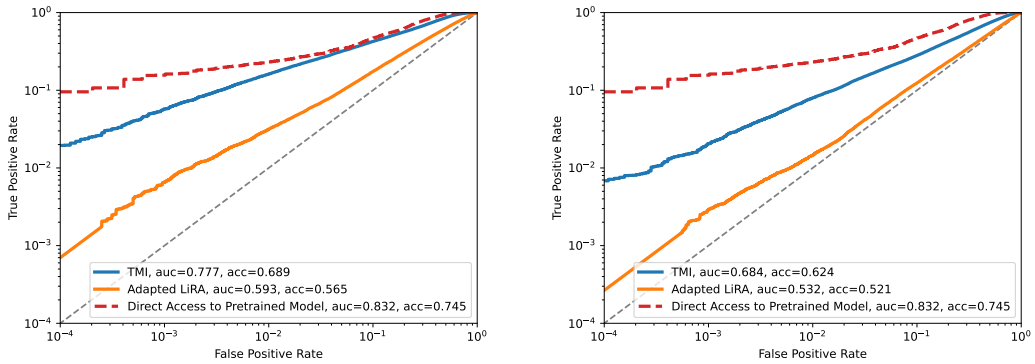
Prior work [58] has shown that samples used earlier in training are more robust to privacy attacks. Our theoretical results in Section 4.1 substantiate this work and help provide an explanation for the decrease in attack success. In our results, α corresponds to the fraction of training epochs spent on finetuning, but our analysis lacks a critical parameter from our experiments: the number or fraction of tunable parameters in the published statistic. Our analysis considers a vector (namely, the empirical mean) where all of the parameters are being updated, thus providing a *worst-case* situation for the adversary. In the feature extraction setting, the information learned by the model during pretraining is essentially frozen. Unlike feature extraction, we are updating the model's parameters with information about the downstream samples when finetuning.

Q2 Answer: Updating larger subsets of model parameters slightly decreases the success of our **TMI** attack when compared to models finetuned on downstream tasks using feature extraction, but we are still able to infer the membership status of the majority of samples in the pretraining dataset.

Table 2: TPR at Fixed FPR of TMI when Pretrained Target Models are Finetuned on by Updating the Pretrained Weights (Figures 7 and 15)

Task	TPR @ 0.1% FPR	TPR @ 1% FPR
Feature Extraction (CIFAR-100 → CIFAR-10)	2.0%	8.0%
Last 2 Layers (CIFAR-100 → CIFAR-10)	1.1%	5.6%
Last 3 Layers (CIFAR-100 → CIFAR-10)	1.1%	4.7%
Feature Extraction (Tiny ImageNet → Caltech 101)	20.7%	41.5%
Last 2 Layers (CIFAR-100 → CIFAR-10)	11.0%	26.5%
Last 3 Layers (CIFAR-100 → CIFAR-10)	7.7%	20.6%

Q3: Does the similarity between the pretraining and downstream task affect the privacy risk of the pretraining set?



(a) Coarse CIFAR-100 **(b) CIFAR-10**

Figure 5: TMI Attack Performance on Downstream Tasks When Preterained CIFAR-100 Target Models are Finetuned Using Feature Extraction

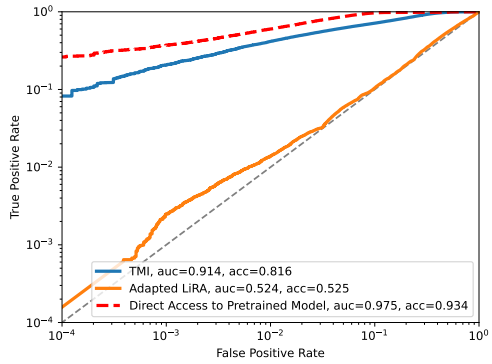


Figure 6: TMI Attack Performance When Preterained Tiny ImageNet Target Models are Finetuned Using Feature Extraction

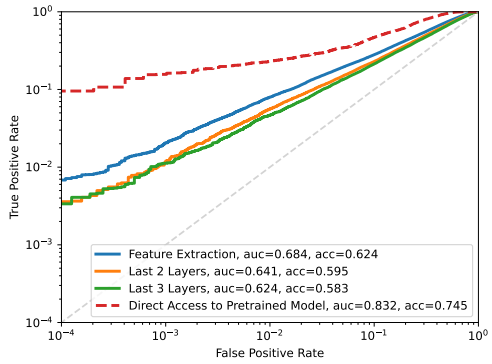


Figure 7: TMI Performance when Finetuning Different Amounts of Parameters on CIFAR-10

Oxford-IIIT Pet. The Oxford-IIIT Pet dataset presents a unique challenge for finetuning our pretrained ResNet models. To finetune these models on the pet breeds classification task, it is necessary to unfreeze all of the layers. Otherwise, the model would have little to no utility with respect to the downstream task. Because the 37 pet breeds that appear in this dataset do not appear in and are not similar to any of the classes in the pretraining data, freezing any of the model's weights is an innefective strategy for this task. In this evaluation of **TMI** on models transferred from CIFAR-100 to Oxford-IIIT Pet, we finetune for the same number of epochs with the same hyperparameters as the models in our experiments with CIFAR-10.

We observe in Figure 8 that the accuracy and AUC of our adaptation of LiRA becomes effectively as good as randomly guessing membership status. In contrast, **TMI** is still able to achieve some amount of success, with an AUC of 0.55 and a balanced accuracy of 53.4% over 128 target models with 1000 challenge points each.

Table 3: TPR at Fixed FPR of TMI when Target Models are Finetuned on Oxford-IIIT Pet by Finetuning All Layers

Task	TPR @ 0.1% FPR	TPR @ 1% FPR
TMI (Oxford-IIIT Pet)	0.5%	2.6%
Adapted LiRA (Oxford-IIIT Pet)	0.08%	1.0%

Additionally, our attack demonstrates a 2.6% true positive rate at a 1% false positive rate.

Q3 Answer: Even though the downstream task of pet breed classification is dissimilar from the pretraining task and all of the model's parameters are finetuned for 20 epochs, **TMI** is able to achieve non-trivial success metrics when inferring the membership status of samples in the pretraining dataset.

5.3.3 Finetuning Pretrained Language Models.

Q4: Can the attack be generalized to domains other than vision?

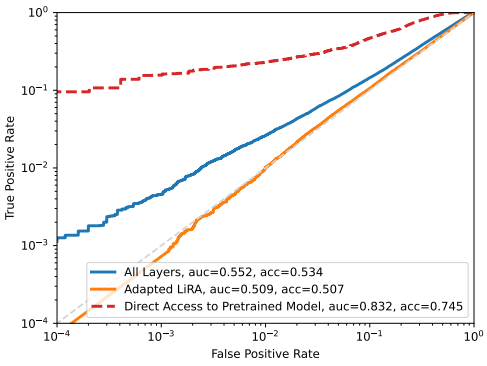


Figure 8: TMI Performance when Finetuning All Parameters on the Oxford-IIIT Pet Dataset

To answer this research question, we evaluate the success of our **TMI** attack in the natural language domain. In particular, we focus on publicly available pretrained large language models (LLMs), or foundation models [59], which we finetune on two text classification tasks.

Due to computational limitations, we do not train LLMs from scratch. As an alternative, we evaluate our attack on a widely used pretrained foundation model, Transformer-XL [55], along with its corresponding tokenizer, which are hosted by Hugging Face [60]. Only a limited number of organizations with sufficient computational resources possess the capability to train foundation models, which are typically fine-tuned on specific tasks by smaller organizations [61–63]. Through our evaluation of **TMI** on finetuned foundation models, we will additionally answer the following research question:

Q5: Is it feasible to mount our attack on finetuned models that are based on publicly hosted foundation models?

We chose this foundation model in particular because it uses known training, validation, and testing splits from the WikiText-103 [27] dataset, providing us with the exact partitions necessary to evaluate **TMI** without having to train our own LLMs. Additionally, although modest in comparison to contemporary foundation models, the Transformer-XL architecture contains 283 million trainable parameters. This makes it a powerful and expressive language model that may be prone to memorizing individual data points.

We finetune Transformer-XL on DBpedia [51], modifying the pretrained tokenizer to use a max length of 450, including both truncation and padding. Using a training set of 10,000 randomly sampled datapoints from DBpedia, we finetune the last third of the parameters in our Tranformer-XL models for 1 epoch. We use the AdamW [64] optimizer with a learning rate of 10^{-5} and weight decay with $\lambda = 10^{-5}$. With these hyperparameters, we are able to achieve test accuracies of 97% and 60% on the 14 classes of DBpedia and 10 classes of Yahoo Answers, respectively.

To prepare our membership-inference evaluation dataset, the WikiText-103 is partitioned into contiguous blocks, separated each by Wikipedia subsections. We then perform the same tokenization process as we do in finetuning before collecting their prediction vectors. Because we do not pretrain our own LLMs, we adapt **TMI** to train a single, global metaclassifier over the prediction vectors of all challenge points rather than train a metaclassifier per challenge point. In total, we use 2650 challenge points, which corresponds to a metaclassifier dataset with size $|D_{\text{meta}}| = 2560 * (\text{number of shadow models})$.

We are unable to compare **TMI** to our adaptation of LiRA because we cannot pretrain our own LLMs. Our adaptation of LiRA requires additional shadow models to be trained from scratch with respect to every challenge point as detailed in Algorithm 2. In our evaluation, we also find that k-nearest neighbors (KNN) significantly outperforms a neural network as a global metaclassifier. We believe this to be the case due to the additional variance incurred in a (global) metaclassifier dataset containing prediction vectors from all challenge points. In contrast, the metaclassifier datasets used in our vision tasks only contained labeled prediction vectors with respect to a single challenge point.

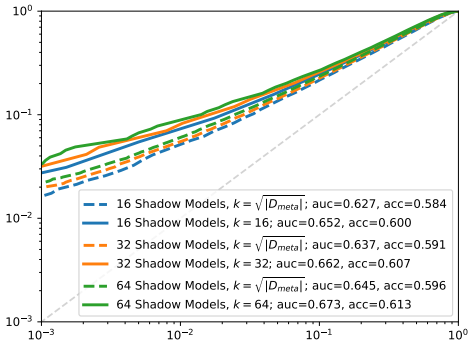


Figure 9: TMI Performance on a Publicly Available Transformer-XL Model Finetuned on DBpedia-14 Topic Classification

We present the results of our evaluation on LLMs in Figures 9 and 16 and Tables 4 and 5. Figure 16 can be found in Appendix D. Although it is common practice to use $k = \sqrt{n}$ neighbors in a KNN, we also report results using k equal to the number of shadow models as it appears to increase attack success. As shown in Table 4, we observe that **TMI** using the highest number of shadow models (64), is able to achieve a TPR of 3.4% and 8.8% at 0.1% and 1% FPR, respectively. These results are comparable to our findings on CIFAR-10 from Table 2 in the vision domain. Surprisingly, we do not observe a notable difference in our summary statistics as we increase the number of shadow models from 16 to 64, with an increase of only 0.652 to 0.673 in AUC, and 60% to 61.3% in accuracy as shown in Figure 9.

The results shown in Figure 16 and Table 5 are consistent with our finding in Section 5.3.2 for similarity between the pretraining and downstream task. We see a slight decrease in **TMI**'s success when Transformer-XL is finetuned on a completely new task, Yahoo

Answers, versus when it is finetuned on data from a similar distribution to its pretraining, DBpedia-14. When the Transformer-XL model is finetuned on the Yahoo Answers topic classification task, **TMI** achieves a TPR of 2.6% and 4.2% at 0.1% and 1% FPR, respectively. Compared to our previous language model experiments, the **TMI** sees a decrease in AUC from 0.67 to 0.59 and a slight decrease in accuracy from 61% to 56%.

Table 4: TPR at Fixed FPR of TMI on Pretrained WikiText-103 Transformer-XL Finetuned on DBpedia-14 (Figure 9)

Task	TPR @ 0.1% FPR	TPR @ 1% FPR
16 Shadow Models ($k = \sqrt{\lvert D_{\text{meta}}\rvert}$)	1.6%	5.2%
16 Shadow Models ($k = 16$)	2.6%	7.0%
32 Shadow Models ($k = \sqrt{\lvert D_{\text{meta}}\rvert}$)	2.0%	5.5%
32 Shadow Models ($k = 32$)	3.1%	8.1%
64 Shadow Models ($k = \sqrt{\lvert D_{\text{meta}}\rvert}$)	2.2%	6.0%
64 Shadow Models ($k = 64$)	3.4%	8.8%

Q4 Answer: Yes, we are able to generalize **TMI** to domains other than vision. In particular, we are able to show that our attack is effective against pretrained language models, and present our results on the publicly hosted Transformer-XL foundation model without the need to pretrain any additional large language models.
Q5 Answer: Yes, **TMI** continues to be effective in this situation where we finetuned public foundation models. This reinforces the need for understanding privacy leakage in the transfer learning setting used for foundation models.

Table 5: TPR at Fixed FPR of TMI on Pretrained WikiText-103 Transformer-XL Finetuned on Yahoo Answers (Figure 16)

Task	TPR @ 0.1% FPR	TPR @ 1% FPR
16 Shadow Models ($k = \sqrt{\lvert D_{\text{meta}}\rvert}$)	1.4%	3.8%
16 Shadow Models ($k = 16$)	1.1%	4.0%
32 Shadow Models ($k = \sqrt{\lvert D_{\text{meta}}\rvert}$)	1.9%	3.7%
32 Shadow Models ($k = 32$)	2.0%	4.4%
64 Shadow Models ($k = \sqrt{\lvert D_{\text{meta}}\rvert}$)	2.1%	3.8%
64 Shadow Models ($k = 64$)	2.6%	4.2%

5.3.4 *Transfer Learning with Differential Privacy.*

Q6: Is privacy leakage present even when a model is finetuned using differential privacy?

We also discuss the performance of our attack on target models that were finetuned with differential privacy. Because prior work on transfer learning with differential privacy considers strategies where an especially small percentage of parameters are trained on the downstream task [1, 2, 4, 50], we freeze the pretrained model's weights and train only the final layer on the downstream task. In our experiments, we perform feature extraction to finetune our pretrained CIFAR-100 models on Coarse CIFAR-100 and CIFAR-10. We train the final classification layer using DP-SGD [50] with target privacy parameters $\varepsilon = \{0.5, 1\}$ and $\delta = 10^{-5}$. As these are strict privacy parameters, we set the clipping norm equal to 5 to achieve reasonable utility on the downstream tasks.

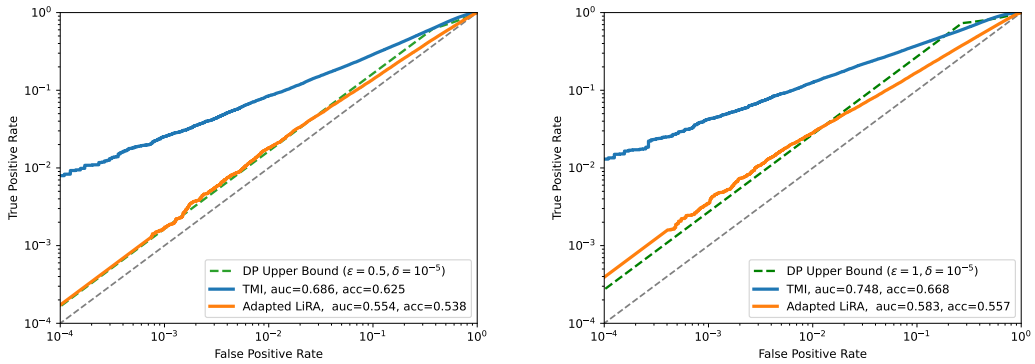
Figure 17 in Appendix D shows that the success of our attack only decreases slightly when differential privacy is used to train the final classification layer on a downstream task. We believe that the slight decrease in attack accuracy can be attributed to loss in utility with respect to the downstream task from training with DP-SGD. When we finetune our models on Coarse CIFAR-100 with privacy parameters $\varepsilon = 0.5$ and $\delta = 10^{-5}$, **TMI** has a TPR of 3.3% at a FPR of 0.1% and a TPR of 10.7% at a FPR 1%. Additionally, our attack maintains about 95% of the accuracy and AUC compared to the setting where no privacy preserving techniques are used to finetune models on Coarse CIFAR-100.

Prior work [19] has shown that state-of-the-art MI attacks, which directly query the pretrained model, completely fail when the target models are trained with a small amount of additive noise. For example, when training target models using DP-SGD with a clipping norm equal to 5 and privacy parameter $\varepsilon = 8$, LiRA has an AUC of 0.5. Through this evaluation, we reinforce the fact that transferring pretrained models to downstream tasks with differential privacy does not provide a privacy guarantee for the pretraining data.

While it may seem expected that finetuning on a disjoint dataset with DP-SGD provides no privacy guarantee for individuals in the pretraining set, the authors of [7] pose the following question: *What privacy guarantee should an individual expect if their data was present in both pretraining and finetuning?* This scenario is not unlikely, as large models are trained on terabytes of data scraped from the Web [10, 65]. Because manually inspecting these datasets is infeasible, it is likely that an individual's datapoint which was included in private finetuning is also present in *non-private* pretraining. Thus, their data does not enjoy the $(\varepsilon, \delta) - DP$ guarantee promised by finetuning, as the corresponding pretraining gradients are unbounded in magnitude and exact in direction. Misusing DP-SGD in this manner can leave these individuals at risk of privacy attacks.

To support our claim that finetuning with DP-SGD is blatantly non-private when $D_{PT} \cap D_{FT} \neq \emptyset$, we run experiments on the CIFAR-100 dataset. Similar to our prior experiments, we finetune the final layer of ResNet-34 shadow models on the Coarse CIFAR-100 dataset using DP-SGD [50] with target privacy parameters $\varepsilon \in \{0.5, 1\}$ and $\delta = 10^{-5}$ and clipping norm equal to 5. The only difference in this experiment is the fact that the finetuning set contains some (∼1000) individuals who were also present in the pretraining task. We run our **TMI** attack on these individuals and report the results in Figure 10 and Table 7. When we finetune with DP-SGD and the challenge points are included in the finetuning set, we see true positive rates that are comparable to our experiments on models finetuned using feature extraction (Table 1). At a fixed FPR of 0.1% and target privacy guarantees of $\varepsilon = 0.5$ and $\varepsilon = 1$, **TMI** achieves a TPR 15.6× higher than the upper bound (end-to-end) training with DP-SGD should provide.
Q6 Answer: Finetuning a pretrained model using DP-SGD provides a privacy guarantee *only* for the downstream dataset. Therefore, DP-SGD has little to no impact on privacy risk of the pretraining dataset, and these downstream models leak the membership status of individuals in the pretraining dataset.

In settings where the pretraining and finetuning data overlap, the guarantee that differential privacy typically provides does not hold. This happens because any given individual's influence in the



(a) $\varepsilon = 0.5, \delta = 10^{-5}$ **(b)** $\varepsilon = 1, \delta = 10^{-5}$

Figure 10: TMI Performance on Samples Present in Both D_{PT} *and* D_{FT} when Finetuning Models with DP-SGD

pretraining process is unbounded and deterministic. Thus, pretraining can induce leakage of individuals in the finetuning set, even when DP-SGD is used to finetune the model.

Table 6: TPR at Fixed FPR of TMI when Target Models are Finetuned with DP-SGD (Figure 17)

Task	TPR @ 0.1% FPR	TPR @ 1% FPR
Coarse CIFAR-100 ($\varepsilon = \infty$)	5.7%	16.1%
Coarse CIFAR-100 ($\varepsilon = 1.0,\ \delta = 10^{-5}$)	3.2%	10.6%
Coarse CIFAR-100 ($\varepsilon = 0.5,\ \delta = 10^{-5}$)	3.3%	10.7%
CIFAR-10 ($\varepsilon = \infty$)	2.0%	8.0%
CIFAR-10 ($\varepsilon = 1.0,\ \delta = 10^{-5}$)	1.6%	6.6%
CIFAR-10 ($\varepsilon = 0.5,\ \delta = 10^{-5}$)	2.1%	6.6%

Table 7: TPR at Fixed FPR of TMI Performance on Samples Present in Both D_{PT} *and* D_{FT} when Finetuning Models with DP-SGD (Figure 10)

Task	TPR @ 0.1% FPR	TPR @ 1% FPR
TMI ($\varepsilon = 0.5,\ \delta = 10^{-5}$)	2.5%	8.5%
TMI ($\varepsilon = 1.0,\ \delta = 10^{-5}$)	4.2%	12.6%
Theoretical Upper Bound ($\varepsilon = 0.5,\ \delta = 10^{-5}$)	0.16%	1.7%
Theoretical Upper Bound ($\varepsilon = 1.0,\ \delta = 10^{-5}$)	0.27%	2.7%

6 DISCUSSION AND CONCLUSION

We study the critical issue of privacy leakage in the transfer learning setting by proposing a novel threat model and introducing **TMI**, a metaclassifier-based membership-inference attack. In particular, we explore how finetuned models can leak the membership status of individuals in the pretraining dataset without an adversary having direct access to the pretrained model. Instead, we rely on queries to the finetuned model to extract private information about the pretraining dataset.

Through our evaluation of **TMI**, we demonstrate privacy leakage in a variety of transfer learning settings. We demonstrate the effectiveness of our attack against a variety of models in both the vision and natural language domains, highlighting the susceptibility of finetuned models to leaking private information about their pretraining datasets. In the vision domain, we show that **TMI** is effective at inferring membership when the target model is finetuned using various strategies, including differentially private finetuning with stringent privacy parameters. We also demonstrate the success of our attack on publicly hosted foundation models by adapting **TMI** to use a global metaclassifier.

Other Privacy Attacks on Finetuned Models. We introduce the first threat model that uses query access to a finetuned model to mount a privacy attack on pretraining data. It remains an open question as to whether other privacy attacks, such as property inference, attribute inference, and training data extraction attacks can also see success in this transfer learning setting. Given that MI attacks are used as practical tools to measure or audit the privacy of machine learning models [39–41], future work should consider efficiency and simplicity when designing new privacy attacks in the transfer learning setting.

Considerations for Private Machine Learning. Our evaluation shows that the pretraining dataset of machine learning models finetuned with differential privacy are still susceptible to privacy leakage. This supports the argument made in [7] that "privacy-preserving" models derived from large, pretrained models don't necessarily provide the privacy guarantees that consumers of services backed by these finetuned models would expect. Prior works that utilize public data to improve the utility of differentially private machine learning models have made strides towards making differential privacy practical for several deep learning tasks [1–6, 66], but they do not address privacy risks external to model training itself.

Using **TMI** as a measurement of privacy leakage in this setting, we reinforce the fact that maintaining privacy depends on taking a holistic approach to the way that training data is handled. As stated in [7], privacy is not binary (i.e. not all data is either strictly "private" or "public") and privacy in machine learning is not only dependent on the model's training procedure. To grapple with

privacy risk in this increasingly popular transfer learning setting, researchers and practitioners should explore new ways to sanitize sensitive information from training datasets of machine learning models, create ways to collect potentially sensitive Web data with informed consent from individuals, and work towards end-to-end privacy-preserving machine learning with high utility and privacy guarantees.

ACKNOWLEDGEMENTS

We thank the anonymous reviewers for their feedback, which helped to improve our paper. This work is supported by NSF grants CNS-2120603, CNS-2247484, and CNS-2232692. John Abascal is supported by the Khoury PhD Fellowship, and Stanley Wu is supported by an REU supplement for NSF award CCF-1750640.

REFERENCES

[1] Nicolas Papernot, Steve Chien, Shuang Song, Abhradeep Thakurta, and Ulfar Erlingsson. Making the shoe fit: Architectures, initializations, and tuning for learning with privacy, 2020. URL https://openreview.net/forum?id=rJg851rYwH.

[2] Da Yu, Saurabh Naik, Arturs Backurs, Sivakanth Gopi, Huseyin A Inan, Gautam Kamath, Janardhan Kulkarni, Yin Tat Lee, Andre Manoel, Lukas Wutschitz, Sergey Yekhanin, and Huishuai Zhang. Differentially private fine-tuning of language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=Q42f0dfjECO.

[3] Xuechen Li, Florian Tramèr, Percy Liang, and Tatsunori Hashimoto. Large language models can be strong differentially private learners. In *International Conference on Learning Representations (ICLR)*, 2022. URL https://arxiv.org/abs/2110.05679.

[4] Zhiqi Bu, Yu-Xiang Wang, Sheng Zha, and George Karypis. Differentially private bias-term only fine-tuning of foundation models, 2023. URL https://openreview.net/forum?id=zoTUH3Fjup.

[5] Jiyan He, Xuechen Li, Da Yu, Huishuai Zhang, Janardhan Kulkarni, Yin Tat Lee, Arturs Backurs, Nenghai Yu, and Jiang Bian. Exploring the limits of differentially private deep learning with group-wise clipping, 2022.

[6] Arun Ganesh, Mahdi Haghifam, Milad Nasr, Sewoong Oh, Thomas Steinke, Om Thakkar, Abhradeep Thakurta, and Lun Wang. Why is public pretraining necessary for private model training?, 2023.

[7] Florian Tramèr, Gautam Kamath, and Nicholas Carlini. Considerations for differentially private learning with large-scale public pretraining, 2022. URL https://arxiv.org/abs/2212.06470.

[8] Katyanna Quach, Oct 2019. URL https://www.theregister.com/2019/10/23/ai_dataset_imagenet_consent.

[9] 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.

[10] July 11 Tuesday and Computer Visiondata scienceDeep LearningNeural Networks. Revisiting the unreasonable effectiveness of data. URL https://ai.googleblog.com/2017/07/revisiting-unreasonable-effectiveness.html.

[11] URL https://cloud.google.com/vertex-ai/docs/generative-ai/models/tune-models.

[12] Nicholas Carlini, Florian Tramèr, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom B. Brown, Dawn Xiaodong Song, Úlfar Erlingsson, Alina Oprea, and Colin Raffel. Extracting training data from large language models. In *USENIX Security Symposium*, 2020.

[13] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In Shai Halevi and Tal Rabin, editors, *Theory of Cryptography*, pages 265–284, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg. ISBN 978-3-540-32732-5.

[14] Together Computer. Redpajama: An open source recipe to reproduce llama training dataset, 2023. URL https://github.com/togethercomputer/RedPajama-Data.

[15] Nils Homer, Szabolcs Szelinger, Margot Redman, David Duggan, Waibhav Tembe, Jill Muehling, John V Pearson, Dietrich A Stephan, Stanley F Nelson, and David W Craig. Resolving individuals contributing trace amounts of DNA to highly complex mixtures using high-density SNP genotyping microarrays. *PLoS Genet.*, 4(8):e1000167, August 2008.

[16] Sriram Sankararaman, Guillaume Obozinski, Michael I Jordan, and Eran Halperin. Genomic privacy and limits of individual detection in a pool. *Nature genetics*, 41 (9):965–967, 2009.

[17] Cynthia Dwork, Adam Smith, Thomas Steinke, Jonathan Ullman, and Salil Vadhan. Robust traceability from trace amounts. In *IEEE Symposium on Foundations of Computer Science*, FOCS '15, 2015.

[18] Reza Shokri, Marco Stronati, and Vitaly Shmatikov. Membership inference attacks against machine learning models. *CoRR*, abs/1610.05820, 2016. URL http://arxiv.org/abs/1610.05820.

[19] Nicholas Carlini, Steve Chien, Milad Nasr, Shuang Song, Andreas Terzis, and Florian Tramèr. Membership inference attacks from first principles. In *43rd IEEE Symposium on Security and Privacy, SP 2022, San Francisco, CA, USA, May 22-26, 2022*, pages 1897–1914. IEEE, 2022. doi: 10.1109/SP46214.2022.9833649. URL https://doi.org/10.1109/SP46214.2022.9833649.

[20] Hongbin Liu, Jinyuan Jia, Wenjie Qu, and Neil Zhenqiang Gong. Encodermi: Membership inference against pre-trained encoders in contrastive learning. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, CCS '21, page 2081–2095, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450384544. doi: 10.1145/3460120.3484749. URL https://doi.org/10.1145/3460120.3484749.

[21] Dingfan Chen, Ning Yu, Yang Zhang, and Mario Fritz. Gan-leaks: A taxonomy of membership inference attacks against gans. *CoRR*, abs/1909.03935, 2019. URL http://arxiv.org/abs/1909.03935.

[22] 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 *2019 IEEE Symposium on Security and Privacy (SP)*. IEEE, may 2019. doi: 10.1109/sp.2019.00065. URL https://doi.org/10.1109%2Fsp.2019.00065.

[23] Alex Krizhevsky. Learning multiple layers of features from tiny images. 2009.

[24] Amirhossein Tavanaei. Embedded encoder-decoder in convolutional networks towards explainable ai, 2020.

[25] Fei-Fei Li, Marco Andreeto, Marc'Aurelio Ranzato, and Pietro Perona. Caltech 101, Apr 2022.

[26] Omkar M Parkhi, Andrea Vedaldi, Andrew Zisserman, and C. V. Jawahar. Cats and dogs. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3498–3505, 2012. doi: 10.1109/CVPR.2012.6248092.

[27] Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models, 2016.

[28] Harsh Chaudhari, John Abascal, Alina Oprea, Matthew Jagielski, Florian Tramèr, and Jonathan Ullman. Snap: Efficient extraction of private properties with poisoning, 2022.

[29] 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 *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, CCS '22, page 2779–2792, New York, NY, USA, 2022. Association for Computing Machinery. ISBN 9781450394505. doi: 10.1145/3548606.3560554. URL https://doi.org/10.1145/3548606.3560554.

[30] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=Sy8gdB9xx.

[31] Mikhail Belkin, Daniel Hsu, Siyuan Ma, and Soumik Mandal. Reconciling modern machine-learning practice and the classical bias–variance trade-off. *Proceedings of the National Academy of Sciences*, 116(32):15849–15854, jul 2019. doi: 10.1073/pnas.1903070116. URL https://doi.org/10.1073%2Fpnas.1903070116.

[32] Vitaly Feldman. Does learning require memorization? a short tale about a long tail. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2020, page 954–959, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450369794. doi: 10.1145/3357713.3384290. URL https://doi.org/10.1145/3357713.3384290.

[33] Gavin Brown, Mark Bun, Vitaly Feldman, Adam Smith, and Kunal Talwar. When is memorization of irrelevant training data necessary for high-accuracy learning? In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2021, page 123–132, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450380539. doi: 10.1145/3406325.3451131. URL https://doi.org/10.1145/3406325.3451131.

[34] Vitaly Feldman and Chiyuan Zhang. What neural networks memorize and why: Discovering the long tail via influence estimation, 2020.

[35] Irit Dinur and Kobbi Nissim. Revealing information while preserving privacy. In *Proceedings of the Twenty-Second ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, PODS '03, page 202–210, New York, NY, USA, 2003. Association for Computing Machinery. ISBN 1581136706. doi: 10.1145/773153.773173. URL https://doi.org/10.1145/773153.773173.

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

[37] 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. *Int. J. Secur. Netw.*, 10(3):137–150, sep 2015. ISSN 1747-8405. doi: 10.1504/IJSN.2015.071829. URL https://doi.org/10.1504/IJSN.2015.071829.

[38] Samuel Yeom, Irene Giacomelli, Matt Fredrikson, and Somesh Jha. Privacy risk in machine learning: Analyzing the connection to overfitting. pages 268–282, 07 2018. doi: 10.1109/CSF.2018.00027.

[39] Jiayuan Ye, Aadyaa Maddi, Sasi Kumar Murakonda, Vincent Bindschaedler, and Reza Shokri. Enhanced membership inference attacks against machine learning models. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, CCS '22, page 3093–3106, New York, NY, USA, 2022. Association for Computing Machinery. ISBN 9781450394505. doi: 10.1145/3548606.3560675. URL https://doi.org/10.1145/3548606.3560675.

[40] Congzheng Song and Vitaly Shmatikov. The natural auditor: How to tell if someone used your words to train their model. *CoRR*, abs/1811.00513, 2018. URL http://arxiv.org/abs/1811.00513.

[41] Introducing a New Privacy Testing Library in TensorFlow — blog.tensorflow.org. https://blog.tensorflow.org/2020/06/introducing-new-privacy-testing-library.html. [Accessed 23-May-2023].

[42] Klas Leino and Matt Fredrikson. Stolen memories: Leveraging model memorization for calibrated white-box membership inference. In *Proceedings of the 29th USENIX Conference on Security Symposium*, SEC'20, USA, 2020. USENIX Association. ISBN 978-1-939133-17-5.

[43] Matthew Jagielski, Stanley Wu, Alina Oprea, Jonathan Ullman, and Roxana Geambasu. How to combine membership-inference attacks on multiple updated machine learning models. *Proceedings on Privacy Enhancing Technologies*, 2023 (3):211–232, 2023. doi: 10.56553/popets-2023-0078.

[44] Matthew Jagielski, Milad Nasr, Christopher Choquette-Choo, Katherine Lee, and Nicholas Carlini. Students parrot their teachers: Membership inference on model distillation, 2023.

[45] Yang Zou, Zhikun Zhang, Michael Backes, and Yang Zhang. Privacy analysis of deep learning in the wild: Membership inference attacks against transfer learning. *CoRR*, abs/2009.04872, 2020. URL https://arxiv.org/abs/2009.04872.

[46] Seira Hidano, Yusuke Kawamoto, and Takao Murakami. Transmia: Membership inference attacks using transfer shadow training. *CoRR*, abs/2011.14661, 2020. URL https://arxiv.org/abs/2011.14661.

[47] Alexandre Sablayrolles, Matthijs Douze, Yann Ollivier, Cordelia Schmid, and Hervé Jégou. White-box vs black-box: Bayes optimal strategies for membership inference, 2019.

[48] Cynthia Dwork, Adam Smith, Thomas Steinke, Jonathan Ullman, and Salil Vadhan. Robust traceability from trace amounts. In *2015 IEEE 56th Annual Symposium on Foundations of Computer Science*, pages 650–669, 2015. doi: 10.1109/FOCS.2015.46.

[49] Tom Fawcett. An introduction to roc analysis. *Pattern Recognition Letters*, 27 (8):861–874, 2006. ISSN 0167-8655. doi: https://doi.org/10.1016/j.patrec.2005.10.010. URL https://www.sciencedirect.com/science/article/pii/S016786550500303X. ROC Analysis in Pattern Recognition.

[50] Martin Abadi, Andy Chu, Ian Goodfellow, H. Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. ACM, oct 2016. doi: 10.1145/2976749.2978318. URL https://doi.org/10.1145%2F2976749.2978318.

[51] Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015. URL https://proceedings.neurips.cc/paper_files/paper/2015/file/250cf8b51c773f3f8dc8b4be867a9a02-Paper.pdf.

[52] Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015. URL https://proceedings.neurips.cc/paper_files/paper/2015/file/250cf8b51c773f3f8dc8b4be867a9a02-Paper.pdf.

[53] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015.

[54] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks, 2017.

[55] Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc Le, and Ruslan Salakhutdinov. Transformer-XL: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2978–2988, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1285. URL https://aclanthology.org/P19-1285.

[56] Christopher A. Choquette-Choo, Florian Tramer, Nicholas Carlini, and Nicolas Papernot. Label-only membership inference attacks. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 1964–1974. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/choquette-choo21a.html.

[57] Thomas Steinke and Jonathan Ullman. The pitfalls of average-case differential privacy. DifferentialPrivacy.org, 07 2020. https://differentialprivacy.org/average-case-dp/.

[58] Matthew Jagielski, Om Thakkar, Florian Tramer, 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 *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=7bJizxLKrR.

[59] Rishi Bommasani, Drew A. Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S. Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, Erik Brynjolfsson, Shyamal Buch, Dallas Card, Rodrigo Castellon, Niladri Chatterji, Annie Chen, Kathleen Creel, Jared Quincy Davis, Dora Demszky, Chris Donahue, Moussa Doumbouya, Esin Durmus, Stefano Ermon, John Etchemendy, Kawin Ethayarajh, Li Fei-Fei, Chelsea Finn, Trevor Gale, Lauren Gillespie, Karan Goel, Noah Goodman, Shelby Grossman, Neel Guha, Tatsunori Hashimoto, Peter Henderson, John Hewitt, Daniel E. Ho, Jenny Hong, Kyle Hsu, Jing Huang, Thomas Icard, Saahil Jain, Dan Jurafsky, Pratyusha Kalluri, Siddharth Karamcheti, Geoff Keeling, Fereshte Khani, Omar Khattab, Pang Wei Koh, Mark Krass, Ranjay Krishna, Rohith Kuditipudi, Ananya Kumar, Faisal Ladhak, Mina Lee, Tony Lee, Jure Leskovec, Isabelle Levent, Xiang Lisa Li, Xuechen Li, Tengyu Ma, Ali Malik, Christopher D. Manning, Suvir Mirchandani, Eric Mitchell, Zanele Munyikwa, Suraj Nair, Avanika Narayan, Deepak Narayanan, Ben Newman, Allen Nie, Juan Carlos Niebles, Hamed Nilforoshan, Julian Nyarko, Giray Ogut, Laurel Orr, Isabel Papadimitriou, Joon Sung Park, Chris Piech, Eva Portelance, Christopher Potts, Aditi Raghunathan, Rob Reich, Hongyu Ren, Frieda Rong, Yusuf Roohani, Camilo Ruiz, Jack Ryan, Christopher Ré, Dorsa Sadigh, Shiori Sagawa, Keshav Santhanam, Andy Shih, Krishnan Srinivasan, Alex Tamkin, Rohan Taori, Armin W. Thomas, Florian Tramèr, Rose E. Wang, William Wang, Bohan Wu, Jiajun Wu, Yuhuai Wu, Sang Michael Xie, Michihiro Yasunaga, Jiaxuan You, Matei Zaharia, Michael Zhang, Tianyi Zhang, Xikun Zhang, Yuhui Zhang, Lucia Zheng, Kaitlyn Zhou, and Percy Liang. On the opportunities and risks of foundation models, 2022.

[60] Transformer XL — huggingface.co. https://huggingface.co/docs/transformers/model_doc/transfo-xl. [Accessed 19-May-2023].

[61] URL https://platform.openai.com/docs/models.

[62] URL https://cloud.google.com/vision.

[63] URL https://learn.microsoft.com/en-us/azure/cognitive-services/openai/how-to/fine-tuning?pivots=programming-language-studio.

[64] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2017.

[65] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf.

[66] Aditya Golatkar, Alessandro Achille, Yu-Xiang Wang, Aaron Roth, Michael Kearns, and Stefano Soatto. Mixed differential privacy in computer vision, 2022.

[67] URL http://groups.csail.mit.edu/vision/TinyImages/.

[68] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009. doi: 10.1109/CVPR.2009.5206848.

[69] Ilya Loshchilov and Frank Hutter. SGDR: Stochastic gradient descent with warm restarts. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=Skq89Scxx.

A MEMBERSHIP INFERENCE UNDER DISTRIBUTION SHIFT

We analyze the success of membership inference attacks on Gaussian mean estimation under distribution shift. While this setting is simple compared to the finetuned deep learning models, it helps us understand how repurposing one estimator for a new problem can leak information about the original dataset.

A.1 Introduction

In this setting, the challenger has access to two datasets, X and Y, where $|X| \gg |Y|$. The challenger uses these datasets to publish a statistic that is a combination of the empirical means of each dataset. This can be thought of as leveraging the the larger dataset, X, to estimate a statistic that comes from a similar distribution. The adversary's goal is the following: Given a challenge point c, determine the membership status of c with respect to the dataset X.

Let $X = \{x_1, \ldots, x_n\}$ and $Y = \{y_1, \ldots, y_m\}$ be datasets where each $x_i \sim \mathcal{N}(\mu, \mathbb{I}_d)$ and $y_i \sim \mathcal{N}(\mu + \nu, \mathbb{I}_d)$, and let $\hat{\mu} = \alpha\bar{x} + (1-\alpha)\bar{y}$ be the statistic released by the challenger.

Then,

$$\mathbb{E}(\hat{\mu}) = \mu + (1-\alpha)\nu$$

and

$$\begin{aligned}\mathrm{Cov}(\hat{\mu}) &= \alpha^2 \mathrm{Var}(\bar{x}) + (1-\alpha)^2 \mathrm{Cov}(\bar{y}) \\ &= \frac{\alpha^2}{n} \cdot \mathbb{I}_d + \frac{(1-\alpha)^2}{m} \cdot \mathbb{I}_d \\ &= \left(\frac{\alpha^2}{n} + \frac{(1-\alpha)^2}{m}\right) \cdot \mathbb{I}_d \\ &= \tilde{\alpha} \cdot \mathbb{I}_d\end{aligned}$$

In this mean estimation setting, $\|\nu\|_2$ can be thought of as the inversely proportional to the similarity between the pretraining and finetuning tasks. If μ is similar to the mean of the new data, Y, $\|\nu\|_2$ is small. The term, α, is analagous to the fraction of pretraining epochs (i.e. number of pretraining epochs divided by the total number of pretraining and finetuning epochs). For example if there are 80 pretraining epochs and 20 finetuning epochs, the corresponding α value would be 0.8. Note that as $\alpha \to 0$, the information from the empirical mean of X is completely overshadowed by the empirical mean of Y. Prior work on membership-inference attacks on machine learning models has suggested that gradient updates (a special case of mean estimation) that do not contain an individual make membership-inference success decrease with respect to that individual [58]. This is consistent with the results we present in this section for the simplified setting of membership-inference attacks on Gaussian mean estimation.

A.2 Threat Model and Attack Algorithm

Similar to prior work on membership inference-attacks on mean estimation, we assume that the adversary has query access to the aggregate statistic, $\hat{\mu}$, along with the true mean of this statistic, $\mathbb{E}(\hat{\mu})$. The membership-inference security game between the challenger and the adversary is defined as the following:

(1) Pick $b \sim \mathcal{U}(\{0,1\})$
(2) If $b = 0$, sample the challenge point, $c \sim \mathcal{N}(\mu, \mathbb{I}_d)$, else sample c uniformly from X
(3) Compute $z = \langle \hat{\mu} - \mathbb{E}(\hat{\mu}) \ , \ c - \mathbb{E}(c) \rangle = \langle \hat{\mu} - (\mu + (1-\alpha)\nu) \ , \ c - \mu \rangle$
(4) If $z > \tau$, output 1. Else, output 0

A.3 Results

Lemma A.1. *If c is OUT ($b = 0$), then*

$$\mathbb{E}(z) = 0 \quad \textit{and} \quad \mathrm{Var}(z) = d\tilde{\alpha},$$

and if c is IN ($b = 1$), then

$$\mathbb{E}(z) = \frac{\alpha d}{n} \quad \textit{and} \quad \mathrm{Var}(z) = d\tilde{\alpha} + \frac{2d\alpha^2}{n^2}$$

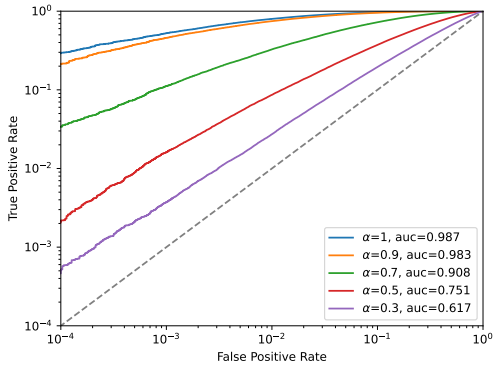
Proof. We will begin by analyzing the *OUT* case, where $c \sim \mathcal{N}(\mu, \mathbb{I}_d)$

$$\begin{aligned}\mathbb{E}(z) &= \mathbb{E}(\langle \hat{\mu} - (\mu + (1-\alpha)\nu) \ , \ c - \mu \rangle) \\ &= \langle \mathbb{E}(\hat{\mu} - (\mu + (1-\alpha)\nu)) \ , \ \mathbb{E}(c - \mu) \rangle \\ &= \langle \vec{0} \ , \ \vec{0} \rangle \\ &= 0\end{aligned}$$

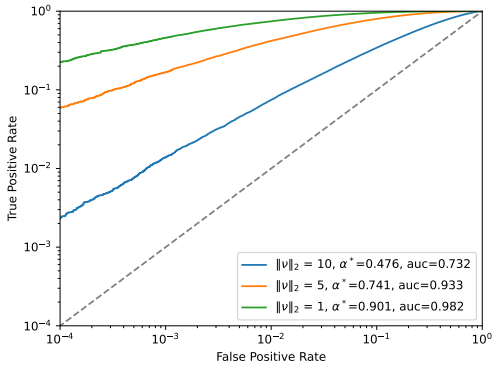
Next, we compute the variance in the *OUT* case:

$$\begin{aligned}\mathrm{Var}(z) &= \mathrm{Var}(\langle \hat{\mu} - (\mu + (1-\alpha)\nu) \ , \ c - \mu \rangle) \\ &= \sum_{i=1}^{d} \mathrm{Var}(\langle \hat{\mu} - (\mu + (1-\alpha)\nu) \ , \ c - \mu \rangle) \\ &= \sum_{i=1}^{d} \mathbb{E}\Big((\hat{\mu} - (\mu + (1-\alpha)\nu))_i^2 \cdot (c - \mu)_i^2 \Big) \\ &= \sum_{i=1}^{d} \mathbb{E}\Big((\hat{\mu} - (\mu + (1-\alpha)\nu))_i^2 \Big) \cdot \mathbb{E}\Big((c - \mu)_i^2 \Big) \\ &= \sum_{i=1}^{d} \tilde{\alpha} \cdot 1 \\ &= d\tilde{\alpha}\end{aligned}$$

Now, we will analyze the IN case, where c is sampled uniformly at random from the dataset, X. In this case, the published statistic, $\hat{\mu}$ and the challenge point c are *not* independent. For succinctness, let $t = (\mu + (1-\alpha)\nu)$.



(a) ROC of the membership inference attack on mean estimation for various values of α ($d = 10,000,\ n = 1000,\ m = 100$)



(b) ROC of the membership inference attack on mean estimation for varying amounts of distribution shift, $\|\nu\|_2$. In each of these simulations, α is set optimally according to Lemma A.3

Figure 11: Performance of the Membership Inference Attack on Mean Estimation

$$
\begin{aligned}
\mathbb{E}(z) &= \mathbb{E}(\langle \hat{\mu} - t\ ,\ c - \mu \rangle) \\
&= \sum_{i=1}^{d} \mathbb{E}((\hat{\mu} - t)_i \cdot (c - \mu)_i) \\
&= \sum_{i=1}^{d} \mathbb{E}(\hat{\mu}_i \cdot c_i) - \mu_i \cdot \mathbb{E}(\hat{\mu}_i) - t_i \cdot \mathbb{E}(c_i) + \mu_i t_i \\
&= \sum_{i=1}^{d} \mathbb{E}(\hat{\mu}_i \cdot c_i) - \mu_i \cdot \mathbb{E}(\hat{\mu}_i) \\
&= \sum_{i=1}^{d} \mathbb{E}(\hat{\mu}_i \cdot c_i) - \mu_i t_i
\end{aligned}
$$

Since $c = x_i \in X$ for some i, without loss of generality suppose $c_i = x_{i,1}$ for all i. Then, $\mathbb{E}(\hat{\mu}_i \cdot c_i)$ becomes

$$
\begin{aligned}
\mathbb{E}(\hat{\mu}_i \cdot c_i) &= \mathbb{E}\left(c_i \cdot \frac{\alpha}{n} \sum_{j=1}^{n} x_{i,j} + c_i \cdot (1-\alpha)\bar{y}_i \right) \\
&= \mathbb{E}\left(\frac{\alpha}{n} \cdot x_{i,1} \cdot c_i + \frac{\alpha}{n} \sum_{j=2}^{n} c_i \cdot x_{i,j} + c_i \cdot (1-\alpha)\bar{y}_i \right) \\
&= \mathbb{E}\left(\frac{\alpha}{n} \cdot x_{i,1} \cdot c_i \right) + \mathbb{E}\left(\frac{\alpha}{n} \sum_{j=2}^{n} c_i \cdot x_{i,j} + c_i \cdot (1-\alpha)\bar{y}_i \right) \\
&= \mathbb{E}\left(\frac{\alpha}{n} \cdot x_{i,1}^2 \right) + \mathbb{E}\left(\frac{\alpha}{n} \sum_{j=2}^{n} c_i \cdot x_{i,j} + c_i \cdot (1-\alpha)\bar{y}_i \right) \\
&= \frac{\alpha}{n} \mathbb{E}\left(x_{i,1}^2 \right) + \frac{\alpha}{n} \sum_{j=2}^{n} \mathbb{E}(c_i) \cdot \mathbb{E}\left(x_{i,j} \right) + \mathbb{E}(c_i) \cdot (1-\alpha)\mathbb{E}(\bar{y}_i) \\
&= \frac{\alpha}{n}(\mu_i^2 + 1) + \frac{\alpha}{n} \sum_{j=2}^{n} \mu_i^2 + \mu \cdot (1-\alpha)(\mu + \nu)_i \\
&= \frac{\alpha}{n}(\mu_i^2 + 1) + \frac{\alpha(n-1)}{n} \mu_i^2 + \mu \cdot (1-\alpha)(\mu + \nu)_i
\end{aligned}
$$

Plugging the above terms into the original expression for the expectation of $\mathbb{E}(z)$ and simplifying yields

$$
\begin{aligned}
\sum_{i=1}^{d} \mathbb{E}(\hat{\mu}_i \cdot c_i) - \mu_i t_i &= \sum_{i=1}^{d} \frac{\alpha}{n} \\
&= \frac{\alpha d}{n}
\end{aligned}
$$

Lastly, we compute the variance in the *IN* case:

$$\begin{aligned}\mathrm{Var}(z) &= \mathrm{Var}(\langle \hat{\mu} - t\ ,\ c - \mu\rangle)\\ &= \sum_{i=1}^{d} \mathrm{Var}((\hat{\mu} - t)_i \cdot (c-\mu)_i)\\ &= \sum_{i=1}^{d} \mathrm{Var}(\hat{\mu}_i \cdot c_i - \mu \cdot \hat{\mu}_i - t \cdot c_i)\end{aligned}$$

Since $c = x_i \in X$ for some i, without loss of generality suppose $c_i = x_{i,1}$ for all i. For succinctness, we drop the summation and indices, i, since all of the dimensions are i.i.d. Expanding $\hat{\mu}$, we get

$$\mathrm{Var}\left(x_1 \cdot \left(\frac{\alpha}{n}\sum_{j=1}^{n} x_j + (1-\alpha)\bar{y_i}\right) - \mu \cdot \left(\frac{\alpha}{n}\sum_{j=1}^{n} x_j + (1-\alpha)\bar{y}\right) - t \cdot x_1\right)$$

We will also use the shorthand $\beta = \frac{\alpha}{n}\sum_{j=2}^{n} x_j + (1-\alpha)\bar{y}$. Note that β is normally distributed with mean $\mu + (1-\alpha)\nu = t$ and variance $\frac{\alpha^2(n-1)}{n^2} + \frac{(1-\alpha)^2}{m}$. Pulling x_1 out of the summations yields

$$\begin{aligned}&= \mathrm{Var}\left(\frac{\alpha}{n}x_1^2 + x_1\beta - \frac{\alpha\mu}{n}\cdot x_1 - \mu\beta - t\cdot x_1\right)\\ &= \mathrm{Var}\left(\frac{\alpha}{n}x_1^2 + x_1(\beta - \frac{\alpha\mu}{n} - t) - \mu\beta\right)\\ &= \mathbb{E}\left((\frac{\alpha}{n}x_1^2 + x_1(\beta - \frac{\alpha\mu}{n} - t) - \mu\beta)^2\right)\\ &- \mathbb{E}\left((\frac{\alpha}{n}x_1^2 + x_1(\beta - \frac{\alpha\mu}{n} - t) - \mu\beta)\right)^2\end{aligned}$$

After algebraic manipulation and computing the individual expectations as in the OUT case, we arrive at

$$\begin{aligned}&= \sum_{i=1}^{d}\tilde{\alpha} + \frac{2\alpha^2}{n^2}\\ &= d\tilde{\alpha} + \frac{2d\alpha^2}{n^2}\end{aligned}$$

□

Lemma A.2. *The mean squared error of $\hat{\mu}$ (as an estimator of the mean of the finetuning data, $\mu + \nu$) is the following:*

$$\mathbb{E}\left(\|\hat{\mu} - (\mu+\nu)\|^2\right) = \tilde{\alpha}d + \alpha^2\|\nu\|_2^2$$

The choice of α that minimizes the mean-squared-error is

$$\alpha^* = \frac{d}{m(\|\nu\|_2^2 + \frac{d}{n}) + d}$$

Proof. Consider the mean squared error of $\hat{\mu}$ as an estimator of the mean of the finetuning data , $\mu + \nu$.

$$\mathbb{E}\left(\|\hat{\mu} - (\mu+\nu)\|^2\right)$$

Note that $Z = \hat{\mu} - (\mu+\nu) \sim \mathcal{N}(-\alpha\nu, \tilde{\alpha}\mathbb{I}_d)$ and for any multivariate Gaussian random variable, $X \sim \mathcal{N}(\mu, \Sigma)$, we have

$$\mathbb{E}\left(\|X\|^2\right) = Tr(\Sigma) + \|\mu\|^2$$

Thus, the mean squared error is

$$\mathbb{E}\left(\|\hat{\mu} - (\mu+\nu)\|^2\right) = \tilde{\alpha}d + \alpha^2\|\nu\|_2^2$$

Suppose the challenger who is releasing $\hat{\mu}$ wants to choose α (i.e. the pretraining-to-finetuning split) such that the error on the finetuning data, Y, is minimized. Computing the derivative of the mean squared error with respect to α yields

$$MSE'(\alpha) = 2d\left(\frac{\alpha}{n} - \frac{1-\alpha}{m}\right) + 2\alpha\|\nu\|$$

Setting $MSE'(\alpha) = 0$ and solving for α, we find that the optimal parameter, α^*, is

$$\alpha^* = \frac{d}{m(\|\nu\|_2^2 + \frac{d}{n}) + d}$$

□

Lemma A.3. *Assume the test statistic, z, is normally distributed. The AUC of our membership-inference attack can be written as the probability the test statistic, z, for an IN sample exceeds z for an OUT sample:*

$$AUC = \frac{1}{2}\left(1 + \mathtt{erf}\left(\frac{\alpha d}{2\sqrt{d(\tilde{\alpha}n^2 + \alpha^2)}}\right)\right)$$

Proof. The AUC of a classifier can be thought of as the probability that the prediction value on a random positive example exceeds the prediction value on a random negative example.

$$AUC = \mathbb{P}(z_{IN} > z_{OUT})$$

where $z_{IN} \sim \mathcal{N}(\frac{\alpha d}{n}, d\tilde{\alpha} + \frac{2d\alpha^2}{n^2})$ and $z_{OUT} \sim \mathcal{N}(0, d\tilde{\alpha})$. Subtracting the the two random variables, we get

$$\begin{aligned}AUC &= \mathbb{P}(z_{IN} > z_{OUT})\\ &= \mathbb{P}(z_{OUT} - z_{IN} < 0)\\ &= \mathbb{P}\left(\mathcal{N}\left(-\frac{\alpha d}{n},\ 2d\tilde{\alpha} + \frac{2d\alpha^2}{n^2}\right) < 0\right)\\ &= \frac{1}{2}\left(1 + \mathtt{erf}\left(\frac{\frac{-\alpha d}{n}}{\sqrt{2}\cdot\sqrt{(2d\tilde{\alpha} + \frac{2d\alpha^2}{n^2})}}\right)\right)\\ &= \frac{1}{2}\left(1 + \mathtt{erf}\left(\frac{\alpha d}{2\sqrt{d(\tilde{\alpha}n^2 + \alpha^2)}}\right)\right)\end{aligned}$$

□

B ABLATIONS

In this section, we evaluate variations of **TMI**. We limit the adversary's access to the target model's prediction outputs, consider different choices of metaclassifier architecture, and study how the number of challenge point queries affects the effectiveness of our attack.

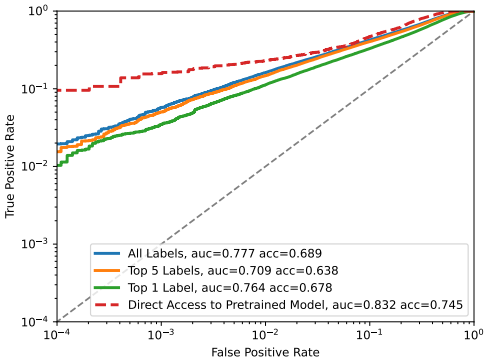


Figure 12: TMI Performance with Access to Prediction Confidence on the Top-K Labels

B.1 Access to Top-k Predictions

In many realistic settings, an adversary who has query access to a computer vision model may only receive predictions for top-k most probable labels. Because our attack relies the information from a combination of labels, we evaluate **TMI** with access to the top 1, top 5, and all labels in the downstream task. For this experiment, we use the same pretrained and finetuned models as in our experiments with Coarse CIFAR-100. This time, when we query the shadow models and target model, we mask the prediction confidences on all but the top-k labels. Because the prediction confidences always sum up to 1, we take the remaining probability mass and divide it amongst the remaining labels to construct the vectors for the metaclassifier (e.g. if the top 5 predictions make up 0.90 of the total probability mass, we divide 0.10 across the remaining 15 labels).

In Figure 12, we show the performance of **TMI** when the adversary has access to the top 1, 5, and 20 labels in our Coarse CIFAR-100 task. Interestingly, **TMI** with access to a single label has higher attack success than our adaptation of LiRA (Figures 5 and 6) even though both adversaries are given the same amount of information. This may be due to the fact that we create some additional information about the other classes by constructing a prediction vector using the labels that the adversary has access to, which is only possible if the adversary knows all of the possible class labels a priori.

B.2 Different Metaclassifier Architectures

Throughout our evaluation, we primarily use a neural network as our metaclassifier to perform our membership-inference attack. In this ablation, we study how the choice of metaclassifier affects the success of our attack. We use the following architectures: neural network multilayer perceptron, support vector machine, logistic regression, and k-nearest-neighbor ($k = \sqrt{D_{\text{meta}}}$). When using the k-nearest-neighbor metaclassifier, we receive hard-label (binary) predictions for membership status. This stands in contrast ot the continuous scores that we receive from the three other metaclassifier architectures. To obtain a membership score in the interval $[0, 1]$, we average the labels from the k-nearest-neighbor models across the prediction vectors obtained from several different augmented queries to the target model. Although we receive a continuous score from each of the other metaclassifiers, we also average the scores across all of the prediction vectors from augmented queries.

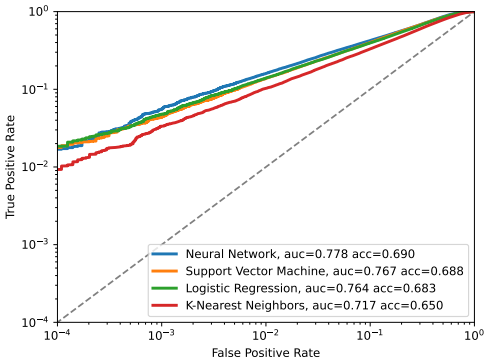


Figure 13: TMI Performance with Different Metaclassifier Architectures

We observe that the overall impact on the effectiveness of our attack is minimal, which indicates that **TMI** is relatively robust to the choice of metaclassifier architecture. Figure 13 shows that the AUC and accuracy slightly decrease when using metaclassifiers other than a neural network. This suggests that an adversary could potentially use faster metaclassifiers than neural networks, such as logistic regression and k-nearest neighbors, without significantly compromising the effectiveness of the attack.

B.3 Number of Augmented Queries

Our attack relies on several augmented queries of a single challenge point on a handful of local shadow models to construct a sufficiently sized metaclassifier dataset. We also query the target model on these augmentations and average the metaclassifier predictions. In this experiment, we explore how the number of augmentations of a challenge point affects the success of **TMI**.

Figure 14 shows that using more augmentations increases the FPR, AUC, and balanced accuracy of our attack. Although **TMI** is more effective with a higher number of augmented queries, training metaclassifiers becomes increasingly computationally expensive as D_{meta} becomes large. For example, **TMI** runs 6× slower on our hardware when using 16 augmentations of the challenge point instead of 8. In all of our prior experiments, we use 8 augmentations to strike a balance between attack effectiveness and efficiency.

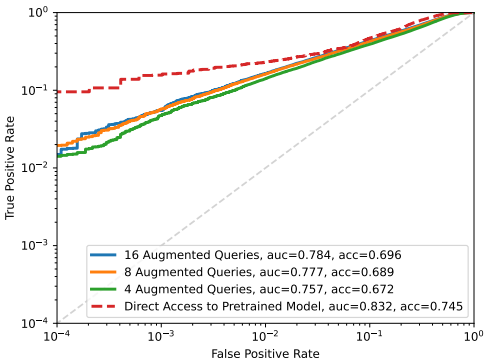


Figure 14: TMI Performance with Different Numbers of Augmented Queries

C DATASETS AND MODELS

In this section, we discuss each of the datasets and the training procedures used in our evaluation of **TMI**.

C.1 Datasets

- *CIFAR-100:* The CIFAR-100 [23] dataset is a subset of the Tiny Images dataset [67], provided by the Canadian Institute for Advanced Research. It is comprised of 60,000 32x32 color images from 100 classes, where each class contains 600 images (500 for training and 100 for testing). CIFAR-100 is used as one of our pretraining tasks because it is a challenging dataset with a wide variety of classes, which allows our models to learn very general features and patterns that can be applied to several downstream tasks.
- *Tiny ImageNet:* Tiny ImageNet [24] is an image classification dataset designed to be a smaller scale alternative to the popular ImageNet [68] dataset. This dataset contains 110,000 64x64 color images from 200 classes, where each class contains 550 images (500 for training and 50 for testing). We use Tiny ImagenNet to pretrain the larger Wide ResNet architecture because it provides a similarly general task to CIFAR-100 at a larger scale. Additionally, the full-sized version of ImageNet is a widely used dataset for pretraining large image models, thus reinforcing the need to evaluate our attack on a dataset like Tiny ImageNet.
- *Coarse CIFAR-100:* The classes in CIFAR-100 can be divided into 20 superclasses. Each image in the dataset has a "fine" label to indicate its class and a "coarse" label to indicate its superclass. We construct this coarse dataset using the superclass labels and use it as our downstream task with the highest similarity to the pretraining task. In our experimentation, we ensure that this downstream task does not contain any of the pretraining samples from the standard CIFAR-100 dataset.
- *Caltech 101:* Caltech 101 [25] is an image classification dataset comprised of about 9000 color images from 101 classes, where each class contains 40 to 800 images. Because the images in this dataset vary in size and tend to be high resolution, we downscale them to be 64x64 to reduce computational complexity. We use the Caltech 101 dataset to finetune our pretrained Tiny ImageNet models as it provides a difficult task with many categories that can be solved by leveraging the generic features learned during pretraining.
- *Oxford-IIIT Pet:* The Oxford-IIIT Pet Dataset [26] is made up of about 7400 color images of cats and dogs. This dataset contains 37 classes with roughly 200 images per class. In our evaluation, this downstream task is the least similar to CIFAR-100 because it focuses on a specific category of images that are mutually exclusive to the pretraining set classes. For this task, we do not use feature extraction because the finetuned model have low utility. Rather, we use the pretrained model as an initialization and update all of its weights.
- *Caltech 101:* In a similar fashion to CIFAR-100, the CIFAR-10 [23] is comprised of 60,000 32x32 color images selected from the Tiny Images dataset. This dataset includes 10 classes, each containing 6000 points (5000 for training and 1000 for testing) which are mutually exclusive to those seen in CIFAR-100. In our evaluation, this downstream task is the second most similar to CIFAR-100 because they are both derived from the same distribution of web-scraped images, but they are disjoint in their classes. Although the classes do not overlap, the features learned from pretraining on CIFAR-100 may be useful in performing this task.
- *WikiText-103:* WikiText-103 [27] is a large-scale language dataset that is widely used for benchmarking language models. It contains over 100 million tokens and is derived from several Wikipedia articles and contains a vast amount of textual data. The language models we consider in this paper have been pretrained on the train partition of WikiText-103 and are hosted on Hugging Face.
- *DBpedia:* The DBpedia ontology (or topic) classification dataset [51] is composed of 630,000 samples with 14 non-overlapping classes from DBpedia, which is a project aiming to extract structured content from the information on Wikipedia. For each of the 14 topics, there are 40,000 training samples and 5000 testing samples. In our experiments with language models, we update a subset of the model's weights on random subsets of this dataset.
- *Yahoo Answers:* The Yahoo Answers topic classification dataset [52] is composed of 1.4 million training samples and 60,000 testing samples with 10 classes. The training and testing sets are divided evenly amongst the topics, such that there are 140,000 training samples and 6000 testing samples per class. Each sample contains both the title and content of a question asked on Yahoo Answers. In our experiments with language models, we update a subset of the model's weights on random subsets of this dataset, where the content of each question is appended to the question title.

C.2 Shadow Model Training

Our shadow model training procedure for vision models is split into two phases: pretraining and finetuning. In the first phase, we train 129 randomly initialized ResNet models on random subsets of

Tiny ImageNet and CIFAR-100, each containing half of the dataset (50k and 25k points, respectively). The remaining samples are held out for evaluation. We train each of the ResNet-34 models for 100 epochs (to 75-80% top-5 accuracy) using SGD with weight decay ($\lambda = 10^{-5}$) and cosine annealing [69] as our learning rate scheduler. Using the same hyperparameters, we train the Wide ResNet-101 architecture for 200 epochs to 60% top-5 accuracy. When training and querying any of our shadow models, we use standard data augmentations, such as random crops and horizontal flips.

In the second phase, we finetune our shadow models on randomly sampled subsets of our downstream task datasets. Before we finetune each shadow model, we swap the classification layer out with a randomly initialized one that has the proper dimension for the downstream task. We then freeze a subset of the model's pretrained weights. When we use feature extraction to finetune our pretrained models, we freeze all weights except for those in the final classification layer. The weights that aren't frozen are trained using the same process as pretraining, but for 20% of the epochs.

When pretraining our shadow models, we designate a randomly selected set of points to be the challenge points for our **TMI** attack. Because each shadow model is trained on half of the dataset, all of the points (including the challenge points) will be IN and OUT for approximately half of the shadow models. In our experiments, we select one shadow model to be the target model and run our attack using the remaining 128 shadow models. Each time we run our attack, we select the a different shadow model to be the target model, yielding a total of 128 trials.

D ADDITIONAL FIGURES

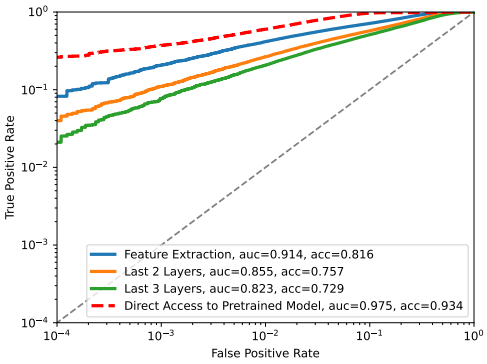


Figure 15: TMI Performance when Finetuning Different Amounts of Parameters on Caltech 101

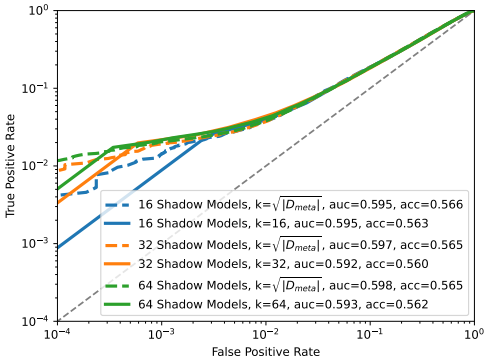
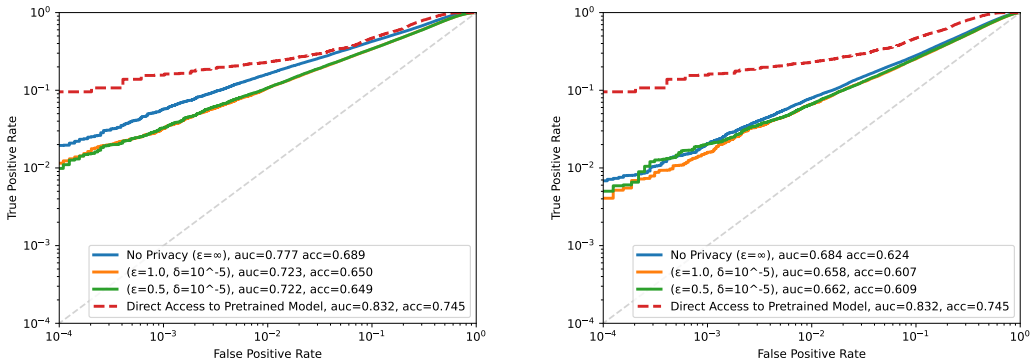


Figure 16: TMI Performance on a Publicly Available Transformer-XL Model Finetuned on Yahoo Answers Topic Classification.



(a) Coarse CIFAR-100 **(b) CIFAR-10**

Figure 17: TMI Performance when Finetuning Models with Differential Privacy (DP-SGD)