Privacy Enhancing Technologies FS2025
Lecture 31-32-33-34 – Membership Inference Attacks

Florian Tramèr

AGENDA

1. Reconstruction vs Distinguishing Attacks
2. Membership Inference Attacks
3. DP Lower-Bounds from Membership Inference
4. Attacks on Deep Neural Networks
5. Auditing Private Learning

Recap

We started our discussion of data privacy by showing how things could go very, very wrong: if we answer too many queries too precisely, an adversary can *reconstruct* a huge chunk of the database...We definitely didn't want that, and so we introduced a formal notion of privacy, differential privacy, and showed how to achieve it by appropriately perturbing our queries. We now have a way to answer queries while defeating these reconstruction attacks.

1 From Reconstruction Attacks to Distinguishing Attacks

What type of attacks does DP actually protect against? As you'll see in the homework, if all that we care about is to make blatant reconstruction attacks infeasible, then we don't really need DP. At the same time, you'll also show that it's possible to defeat reconstruction attacks while still not being "private" at all. This motivates us to consider *weaker attacks* than reconstruction attacks, that capture the protections offered by DP more accurately.

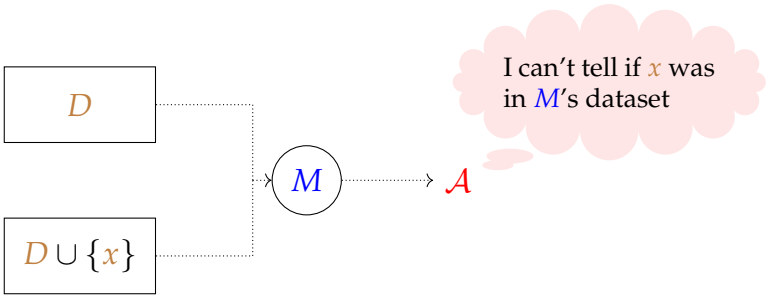
A detour through cryptography. Consider how we define privacy in cryptography. In many cases, the privacy definition is directly linked to an attack. For example, we can define security of encryption schemes by saying that no adversary can distinguish the ciphertexts of two messages (i.e., IND-CPA security).

Notice that this is *not* a reconstruction attack. We could have also defined security of an encryption scheme by asking for *plaintext recovery* attacks to be infeasible. But this is not strong enough.[1] Instead, we define privacy in terms of a *distinguishing attack*: an attack cannot even recover a single bit of the plaintext to distinguish it from another plaintext.

[1] E.g., consider a scheme that encrypts half of the bits of the plaintext with a one-time-pad, and leaves the rest in the clear. Full plaintext recovery is impossible, but the scheme is clearly insecure. This is somewhat analogous to the sampling mechanism we considered above, which defeats reconstruction attacks, but doesn't help privacy.

Differential privacy as hardness of inferring membership. What would an analogous attack look like for differential privacy? The definition of differential privacy does not directly refer to an adversary, or to an attack—primarily because it is an information-theoretic notion.

But differential privacy does have an indistinguishability flavor: it says that the distribution of outputs of the mechanism on database D is close to the distribution of outputs on a neighboring database D'. Informally speaking, this is equivalent to saying that no adversary (in this case even a computationally unbounded one) can easily distinguish whether some output was generated by the mechanism on D or on D'. And since D and D' only differ in one entry[2] (say the presence of some individual x), this is equivalent to saying that no adversary can distinguish whether x was in the database or not. This is what we call a *membership inference attack*.



2 Membership Inference Attacks

Membership inference attacks were originally introduced in an influential empirical paper by Homer et al. [HSR+08], which showed that it may be possible to infer an individual's presence in a medical study from aggregate genetic statistics. These attacks were later used to derive formal upper bounds on differential privacy [BUV14, DSS+15], in settings where reconstruction attacks do not yield tight bounds. In recent years, membership inference attacks have seen a lot of study in machine learning [SSSS17, CCN+22], where they serve as an empirical test for privacy leakage [JUO20, NST+21].

The membership inference game. We can setup the problem of membership inference as a game between an *adversary* and a *challenger*. In its strictest form, the game might look like this: the adversary chooses a database D and a data point x, and the challenger runs a mechanism M (e.g., a machine learning training algorithm) on one of two neighboring databases D or $D \cup \{x\}$. The adversary then tries to guess which database the mechanism was run on.

[2] We consider here the definition of neighboring databases where the two databases differ in the presence of a single individual x. So D and D' have different size (i.e., $|D'| = |D| + 1$).

Membership Inference Game		
Adversary		**Challenger**
	$D, x \longrightarrow$	$D_0 = D, D_1 = D \cup \{x\}$
	$\longleftarrow M(D_b)$	$b \leftarrow^{\$} \{0,1\}$
output $b' \in \{0,1\}$		
Adversary wins if $b' = b$		

This is a "worst-case" variant of the game, where the adversary freely gets to choose both the database and the target data point. There are many weaker variants of this game in the research literature, where the database might instead by sampled from some distribution, and the target data point might also be randomly sampled from this distribution.

As we will see, the worst-case variant is very closely tied to the definition of differential privacy. For the distributional variants, it is a bit harder to characterize what privacy notion is captured, but they are useful nonetheless.

Success metrics. How should we measure the success of an adversary in the membership inference game? This is actually a bit subtle, and one of the reasons why the research literature on membership inference attacks has considered so many different variants.

As a first attempt, we might take inspiration from cryptography and define the advantage of the adversary as

$$\mathrm{Adv}_{\mathrm{MI}}(\mathcal{A}) = \left|\Pr[b' = b] - 1/2\right|.$$

This is how we would measure the success of the adversary in a typical cryptographic game, e.g., an IND-CPA game. In cryptography, such a success metric is enough, because we typically want this to be negligibly small.

But in our setting there, the adversary's advantage must always be non-negligible if we want the mechanism M to be useful (can you see why?). And so a more granular analysis of the attacker's success rate can be useful.

One way is to take a closer look at the attacker's "confusion matrix", i.e., the rates at which the adversary guesses correctly or incorrectly whether the mechanism used the target data point x or not. Define

$$\mathrm{TPR} = \Pr[b' = 1 \mid b = 1] \qquad \mathrm{TNR} = \Pr[b' = 0 \mid b = 0]$$
$$\mathrm{FPR} = \Pr[b' = 1 \mid b = 0] \qquad \mathrm{FNR} = \Pr[b' = 0 \mid b = 1]$$

as the true positive rate, true negative rate, false positive rate, and false negative rate, respectively (i.e., the probability that the adversary guesses correctly/incorrectly that x is/isn't a member of the database).

It is then also natural to measure the success of the adversary in the membership inference game via some comparison of the attack's positive and negative rates at predicting membership (or non-membership). It turns out that this view is directly connected to the definition of differential privacy (here we consider two databases D and D' as neighboring if they differ by addition or removal of a single point x):

Theorem 1 ([KOV15]). Let $\varepsilon > 0$ and $\delta \in [0, 1]$. A mechanism M is (ε, δ)-DP *if and only if* for all adversaries in the membership inference game,

$$e^{\varepsilon} \cdot \mathrm{FNR} \geq \mathrm{TNR} - \delta \quad \text{and}$$
$$e^{\varepsilon} \cdot \mathrm{FPR} \geq \mathrm{TPR} - \delta \tag{1}$$

So a mechanism being DP provides much stronger guarantees than just against data reconstruction attacks. It says that any adversary that tries to distinguish the presence or absence of one individual in the database will necessarily make true and false inferences with roughly the same probability.

Proof. We first prove that (ε, δ)-differential privacy implies eq. (1). We'll assume here for simplicity that the attacker is deterministic. Let S be the set of outputs of M for which the attacker outputs $b' = 0$. From the definition of differential privacy, we know that:

$$\underbrace{\Pr[M(D_0) \in S]}_{\mathrm{TNR}} \leq e^{\varepsilon} \cdot \underbrace{\Pr[M(D_1) \in S]}_{\mathrm{FNR}} + \delta\,.$$

This is the first inequality in (1). The second one follows similarly by considering the complement set $\bar{S}$ for which the attacker outputs $b' = 1$.

For the converse statement, assume that for any choice of set S, if the attacker outputs $b' = 0$ whenever the mechanism output is in S we get that eq. (1) holds. This implies that for all S, we have

$$\Pr[M(D_0) \in S] \leq e^{\varepsilon} \cdot \Pr[M(D_1) \in S] + \delta\,.$$

If we instead consider sets S that make the attacker output $b' = 1$, we get

$$\Pr[M(D_1) \in S] \leq e^{\varepsilon} \cdot \Pr[M(D_0) \in S] + \delta\,.$$

Together, these imply (ε, δ)-differential privacy. □

Corollary 1. A mechanism M is (ε, δ)-DP *if and only if* for all adversaries in the membership inference game,

$$e^{\varepsilon} \geq \max\left(\frac{\mathrm{TNR} - \delta}{\mathrm{FNR}}, \frac{\mathrm{TPR} - \delta}{\mathrm{FPR}}\right)$$

So, for example, if an attacker has a TPR of 90% and an FPR of 1% (i.e., the attacker can correctly infer membership with 90% probability, and falsely accuses someone of membership with only 1% probability), then the mechanism is (at best) DP for $\varepsilon = \ln 90 \approx 4.5$. If the best attack is much weaker, and only achieves 5% TPR and 1% FPR, then the mechanism is ε-DP for $\varepsilon \geq \ln 5 \approx 1.6$.

Interpreting membership inference. How should we "think" about membership inference? On one hand, we can view it as a concrete attack that an adversary might try to mount against a system. This is how it was originally motivated in the context of genomic data [HSR+08], where an attacker might be able to determine that your genetic data was used in a study of some sensitive condition, say, a rare disease.

But this is a rather weak attack with a very strong pre-condition: we assume that the attacker already knows *all* your genetic data, and just needs to learn one extra bit of information,

namely whether you have a specific disease or not. From a privacy standpoint, the fact that the attacker already knows all your genetic data is maybe the bigger issue.

So I tend to not think of membership inference as a concrete attack, but rather as a way to *lower-bound* the privacy of a system. That is, if we can show that a membership inference attack is successful, then we know there is *something* wrong with the system's privacy, even if the distinguishing attack itself is not that worrisome. This is a bit like a chosen ciphertext attack in cryptography: the ability to distinguish two plaintexts from their encryptions might not be a very strong attack in itself (although it can sometimes be used to mount much stronger key- or plaintext-recovery attacks); however, the fact that it is possible in the first place tells us that the encryption scheme is not good.

As we will see in this lecture and the next, this interpretation of membership inference can be very useful both in theory (to derive lower-bounds for differentially private mechanisms), and in practice (to empirically audit or debug privacy-preserving systems).

3 DP Lower-bounds from Membership Inference Attacks

We'll first show how to use membership inference attacks to prove lower-bounds on how many counting queries we can answer with the Gaussian mechanism.

Recall that our analysis of the Gaussian mechanism from lecture 10 showed that if we add noise of standard deviation $\sigma = \tilde{O}(\sqrt{k}/n)$ to each query, then we get DP with $\varepsilon = O(1)$. We now show that this is also necessary: if we add slightly less noise, the mechanism does not satisfy DP with a constant ε (and $\delta = o(1)$).

Theorem 2. Let M be a Gaussian mechanism for computing the mean of a database $D = \{\vec{x}_1, \ldots, \vec{x}_n\}$ of n vectors from the domain $\{0,1\}^k$. I.e.,

$$M(D) = \frac{1}{n}\sum_{i=1}^{n} \vec{x}_i + \mathcal{N}(\vec{0}, \sigma^2 \cdot I_{k\times k})$$

Assume $\sigma = o(\sqrt{k}/n)$ and $\delta = o(1)$. Then this mechanism is <u>not</u> $(O(1), \delta)$-DP.

Proof. We're going to build a membership inference attack against this mechanism. Since the adversary gets to pick the database D and the target $\vec{x}$, let's consider the absolute worst case, where the adversary picks D to be the all-zero database, and $\vec{x}$ to be the all-one vector.

Our attack now proceeds as follows:

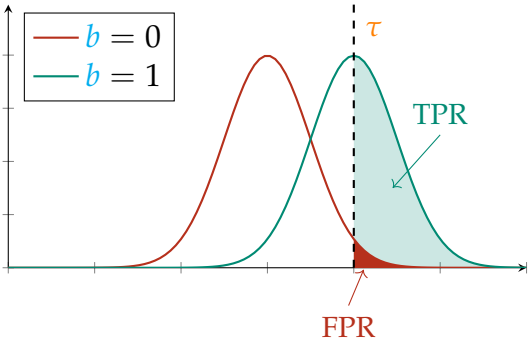
1. The adversary sends D and $\vec{x}$ to the challenger, and receives $\vec{y} = M(D_b)$.
2. The attacker computes the test statistic $z = \langle \vec{y}, \vec{x} \rangle$.
3. The attacker outputs 1 if $z \geq \tau$ and 0 otherwise, for some threshold τ to be chosen later.

Intuitively, the attacker tries to build a statistic z that takes on very different values depending on whether the mechanism was run on D_0 or D_1. Note that if $\vec{x}$ is not in the database, then the output of M is only noise, and so z is just the sum of k independent Gaussians $\mathcal{N}(0, \sigma^2)$. And if $\vec{x}$ is in the database, then this sum of Gaussians is shifted by k/n (i.e., the

value of $\langle \vec{x}, \vec{x}/n \rangle$). That is, we have:

$$z = \begin{cases} \sum_{i=1}^{k} \mathcal{N}(0, \sigma^2) = \mathcal{N}(0, k\sigma^2) & \text{if } b = 0 \\ \sum_{i=1}^{k} \mathcal{N}(1/n, \sigma^2) = \mathcal{N}(k/n, k\sigma^2) & \text{if } b = 1 \end{cases}$$

Here's how this would roughly look like:



Let's now fix a threshold $\tau = k/n$. We want to show that the attack can achieve high TPR at low FPR. So we'll lower-bound the TPR and upper-bound the FPR.

Computing the TPR is easy. By symmetry of the Gaussian, we have:

$$\text{TPR} = \Pr[z \geq \tau \mid b = 1] = \Pr[\mathcal{N}(k/n, k\sigma^2) \geq k/n] = 1/2 \,.$$

To upper-bound the FPR, we'll use a standard tail-bound on the Gaussian.

> Let $X \sim \mathcal{N}(0, \sigma^2)$ and $t > 0$. Then:
>
> $$\Pr[X \geq t] \leq \exp(-t^2/2\sigma^2)$$

So we get:

$$\begin{aligned} \text{FPR} = \Pr[z \geq \tau \mid b = 0] &= \Pr[\mathcal{N}(0, k\sigma^2) \geq k/n] \\ &\leq \exp\left(-\frac{(k/n)^2}{2\,k\,\sigma^2}\right) = o(1) \,, \end{aligned}$$

where we used that $\sigma = o(\sqrt{k}/n)$.

Using Corollary 1, we can now conclude that the mechanism's privacy budget ε is at least:

$$\varepsilon \geq \ln\left(\frac{\text{TPR} - \delta}{\text{FPR}}\right) = \ln\left(\frac{1/2 - o(1)}{o(1)}\right) = \omega(1) \,,$$

i.e., the mechanism is not DP for any constant ε. □

So we have shown that to answer k counting queries with the Gaussian mechanism, if we want a constant privacy budget ε, we need $\sigma = \Omega(\sqrt{k}/n)$, and thus error $\Omega(\sqrt{k}/n)$ per query. Using more involved arguments based on membership inference, one can show that this lower-bound applies to *any* mechanism that answers k counting queries, not just the Gaussian mechanism [BUV14, DSS+15].

4 Deep Neural Networks

A neural network $f_\theta : \mathcal{X} \to [0,1]^C$ is a function that maps some input data sample $x \in \mathcal{X}$ to a C-class probability distribution; we let $f_\theta(x)_y$ denote the probability of class y.

A (standard feedforward) neural network is a composition of linear transformations and nonlinearities, with the outputs of the last layer passed through a *softmax function* σ to obtain a probability distribution. That is:

$$f(x) = \sigma\Big(\underbrace{f_k \circ \cdots \circ f_1(x)}_{z_\theta(x)}\Big) , \tag{2}$$

where each f_i consists of a learned linear transformation followed by a fixed nonlinearity, and $z_\theta : \mathcal{X} \to \mathbb{R}^C$ returns the final *logits* followed by the *softmax*:

$$\sigma(z) = \left[\frac{e^{z_1}}{\sum_i e^{z_i}}, \ldots, \frac{e^{z_C}}{\sum_i e^{z_i}}\right] . \tag{3}$$

Neural networks are trained via gradient descent to minimize some loss function ℓ:

$$\theta_{t+1} \leftarrow \theta_t - \eta \sum_{(x,y)\in B} \nabla_\theta \ell(\theta_t, x, y) \tag{4}$$

Here, B is a batch of random training examples from the training set D, and η is the learning rate, a small constant. For classification tasks, the most common loss function is the cross-entropy loss:

$$\ell(\theta, x, y) = -\log(f_\theta(x)_y) . \tag{5}$$

Overfitting. Since we train machine learning models to minimize the loss function over the training data too well, and have much lower loss on the training data than on the general population (i.e., the empirical risk $\widehat{\mathcal{L}}_D(\theta)$ is much lower than the population risk $\mathcal{L}(\theta)$). This is typically referred to as *overfitting*.

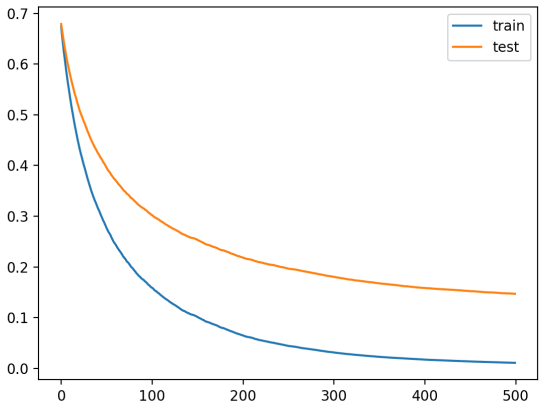


Figure 1: Deep neural networks tend to overfit to the training data. The training loss (in blue) typically decreases faster than the test loss (in orange).

Overfitting relates to privacy in that it reveals whether the trained model has *memorized* some features of the training data that do not generalize to the general population. But some presence of overfitting does not necessarily imply that individual data points are at risk of being leaked. All it tells us is that the model has memorized some aspects of the training data *on average*.

5 Membership Inference Attacks Against Neural Networks

Suppose we play the membership inference game where the mechanism M trains and returns a neural network model f_θ on some training data D. What strategies could the adversary use to infer whether the model was trained on D_1 or D_0 (i.e., whether x was in the training data or not)?

One option could be to look at the learned parameters θ and see if they somehow encode some information about x. We'll get back to this idea later (it's in fact surprisingly hard to do this). Instead, we could just look at the model's *output* on x—in particular, the model's loss $\ell(\theta, x, y)$. This is called as *black-box* attack because we only need query access to the trained model.

The intuition behind using the model's loss as a membership indicator is that we expect the loss to be lower for data points in the training data, on average, due to overfitting:

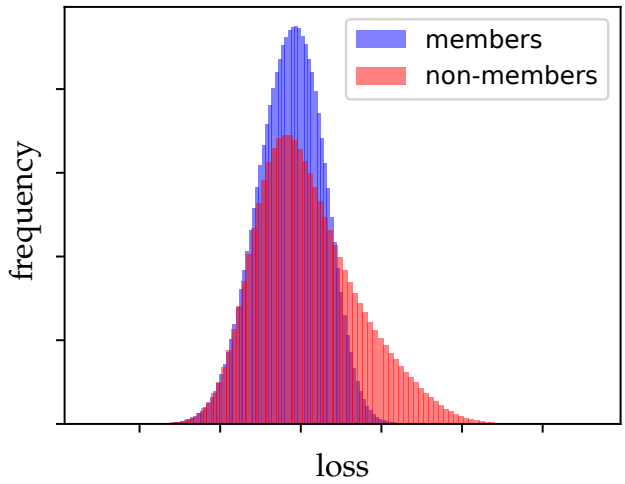


Figure 2: Histogram of the loss of a neural network trained on CIFAR-10, for samples from the training set (in red) and from the test set (in blue).

5.1 A Simple Attack: Global Loss Thresholding

The earliest membership inference attacks against machine learning models were directly inspired by the (average-case) overfitting phenomenon shown above [SSSS17, YGFJ18]. Since $\widehat{\mathcal{L}}_D(\theta) < \mathcal{L}(\theta)$, we can set some threshold τ on the loss ℓ and declare any point x to be a member if its loss is below the threshold:

$$\mathcal{A}_{\text{global}}(\theta, x, y) = \begin{cases} 1 & \text{if } \ell(\theta, x, y) \leq \tau \\ 0 & \text{otherwise} \end{cases}$$

However, it turns out that this is not a very good attack. For example, if we look at the histogram of losses above in Figure 2, we can see that there is no threshold that would let us confidently infer that any sample is a member.[3]

But this is when we consider the attack's performance *on average* across all samples. What we'd rather want is an attack tailored to each individual sample that we want to attack. But here, a global threshold won't do the trick.

[3]Note that we can however confidently infer that the sample is *not* a member if its loss is high enough.

5.2 The Need for Difficulty Calibration

The reason that a global thresholds on the loss gives a poor attack is that not all samples are equally hard to learn. This is illustrated in Figure 3 below. We train a large number of models on random subsets of CIFAR-10, and measure the loss of each model on four different points. For each of the four images, we then plot the loss values for models that were trained on that image (in red) and models that were not trained on that image (in blue).

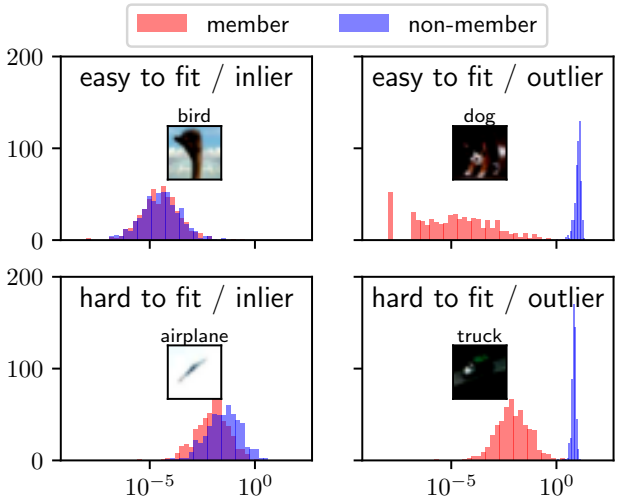


Figure 3: Not all samples are equally hard (figure copied from [CCN+22]). For four samples from CIFAR-10, we plot a histogram of a model's loss on that sample when it was in the training set (in red) and when it was not (in blue). Only outliers are easy to infer membership for. The losses of different samples can also be on very different scales, depending on how hard they are to learn.

These images were chosen to showcase some extreme behaviors:

- The bird on the top left is a prototypical "inlier" (i.e., a point that is typical of the training data; there are many similar birds in the CIFAR-10 dataset). For such points, the model generalizes well, and so the loss is essentially the same whether the model was trained on this specific image or not.
- However, some inliers are still "harder" to learn than others. The airplane on the bottom left is also an inlier (i.e., it's presence in the training data has little influence on the model's loss), but it belongs to a category of images that are harder to learn than the top-left bird.
- The dog on the top right is an outlier (i.e., other dogs in the dataset are typically not as dark). Models that are not trained on this image tend to misclassify it and get very high loss. But when a model is trained on this image, it usually memorizes its label and gets low loss.
- The truck on the bottom right is also an outlier, but this one is harder for the model to memorize; even if the model was trained on the truck, its loss is still rather high.

From the figure, it's clear that a single global threshold τ will not work well in distinguishing members from non-members for all data points.

5.3 A Stronger Attack: Per-Sample Likelihood Ratios

A better attack strategy would be to set a different threshold on the loss ℓ for each data point x. But still, this would be somehow crude because there is no way for the attack to express its uncertainty over whether a point is a member or not. For example, if you set a threshold of τ for the loss ℓ of some point x, and you observe a loss of 1.01τ, you should intuitively be more uncertain about whether x is a member or not than if you observed a loss of 100τ.

This leads us to the **optimal** way of formulating the membership inference attack as a *likelihood ratio test*. Define the *null hypothesis* H_0 that x is not a member of the training data, and the *alternative hypothesis* H_1 that it is a member. Further let $\texttt{View} = M(D_b)$ be the output of the mechanism that the adversary gets to see. The likelihood ratio test then compares the ratio of the likelihoods of the null and alternative hypotheses, given the observed data

$$\Lambda(\texttt{View}) = \frac{\Pr[\texttt{View} \mid H_0]}{\Pr[\texttt{View} \mid H_1]}, \tag{6}$$

and rejects the null hypothesis if $\Lambda(\texttt{View}) \leq \tau$ for some threshold τ.

The Neyman-Pearson lemma tells us that this likelihood ratio test is the most powerful test for distinguishing between the null and alternative hypotheses, at any given false positive rate [NP33]!

Now it remains to actually compute the likelihoods $\Pr[\texttt{View} \mid H_0]$ and $\Pr[\texttt{View} \mid H_1]$. We don't know of any way to do this in closed form, since the probability here is taken over the entire process of training a deep neural network. Instead, we could aim to estimate the likelihoods empirically. We describe here the approach of [CCN+22], but there are others with slightly different assumptions (e.g., [YMM+22]).

First, let's again consider that we are in a black-box setting where the adversary just observes the loss for some target x and some label y, i.e., $\texttt{View} = \ell(\theta, x, y)$. So we now want to estimate

$$\Pr[\ell(\theta, x, y) \mid H_b] \tag{7}$$

To do this, [CCN+22] suggest to train multiple "IN" and "OUT" models that mimic the membership inference game. That is, the attacker samples some training data D_{atk} from the distribution $\mathcal{P}$ and then trains a model $f_{\theta^{\text{IN}}}$ on $D_{atk} \cup \{x\}$ and a model $f_{\theta^{\text{OUT}}}$ on D_{atk}. After repeating this process N times (for the same target x), we get $2N$ loss samples:

$$\text{Loss}_{\text{IN}} = \left\{ \ell(\theta^{\text{IN}}, x, y) \;\middle|\; \begin{array}{c} D_{atk} \sim \mathcal{P} \\ \theta^{\text{IN}} \leftarrow M(D_{atk} \cup \{x\}) \end{array} \right\}$$

$$\text{Loss}_{\text{OUT}} = \left\{ \ell(\theta^{\text{OUT}}, x, y) \;\middle|\; \begin{array}{c} D_{atk} \sim \mathcal{P} \\ \theta^{\text{OUT}} \leftarrow M(D_{atk}) \end{array} \right\}$$

These loss values are what we previously showed in Figure 3. The final step in the attack from [CCN+22] is to fit a Gaussian distribution to both of these sets of loss samples, so that we can estimate the probability in eq. (7) even when it is very small (and thus estimate the attack's TPR for very small FPR). Then, given these distributions $\mathcal{N}(\mu_{\text{IN}}, \sigma^2_{\text{IN}})$ and $\mathcal{N}(\mu_{\text{OUT}}, \sigma^2_{\text{OUT}})$, the final likelihood ratio test is given by

$$\Lambda(\ell) = \frac{\Pr[\ell \mid H_0]}{\Pr[\ell \mid H_1]} = \frac{\mathcal{N}(\mu_{\text{OUT}}, \sigma^2_{\text{OUT}})(\ell)}{\mathcal{N}(\mu_{\text{IN}}, \sigma^2_{\text{IN}})(\ell)} \tag{8}$$

The performance of this attack on CIFAR-10 is shown in Figure 4. Here, we show the attack's TPR at a fixed FPR of 0.1%, for each of the 50'000 samples in CIFAR-10. We can see that a small number of samples are incredibly vulnerable to membership inference, with the attack reaching a TPR of almost 100% for some samples. The majority of samples, however, are very hard to infer membership for. The attack achieves at best a few percent of TPR for most samples.

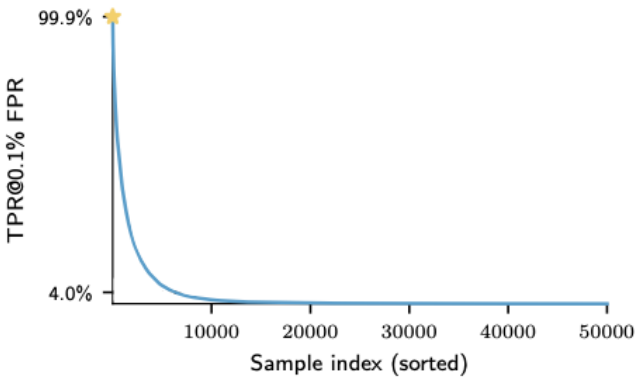


Figure 4: Performance of the Likelihood Ratio Attack (LiRA) of [CCN+22] (figure copied from [AZT24]). For each of the 50'000 samples in CIFAR-10, we plot the attack's TPR at a fixed FPR of 0.1% (sorted by TPR).

This is illustrative of how privacy is a *worst-case* notion: while most of the data samples might be safe, a successful privacy enhancing technology should protect *all* data samples from attack, even samples that are highly unusual (maybe those are the ones that are most important to protect...).

Training a model with differential privacy (e.g., with DP-SGD) would indeed prevent membership inference attacks for *all* data points, if the privacy budget ε is set appropriately.

6 Auditing Private Learning with Membership Inference

Recall Corollary 1 which lower bounds the privacy parameter ε of a mechanism in terms of the attack's TPR and FPR.

One way we can use this result is to select parameters (ε, δ) so that we get formal guarantees on the performance of *any* membership inference attack. For example, to ensure that no attack can reach a TPR of 99% at a FPR below 1%, we should set $\varepsilon \leq \ln 99 \approx 4.59$.

But we can also apply this result in the other direction: given some empirical evaluation of a membership inference attack, we can use it to lower-bound the privacy parameters (ε, δ) that a mechanism satisfies. There are many applications of this in the literature, which we discuss below.

6.1 Debugging DP-SGD Implementations

As with any security mechanism, private training algorithms such as DP-SGD might not actually provide the formal guarantees given by the privacy analysis. This could be due to bugs in the implementation, or due to flaws in the privacy analysis itself. That is, even if we have a purported proof that our DP-SGD algorithm satisfies, say $(1, 10^{-8})$-DP, it is possible that the actual privacy loss is much higher in practice.

Membership inference attacks are a way to empirically evaluate the privacy of a mechanism. If a mechanism claims ε privacy, and yet we can build a membership inference attack that achieves $\mathrm{TPR}/\mathrm{FPR} \geq e^{\varepsilon}$, then we have evidence that there is a bug somewhere.

As an example, in [TTS+22] we looked at a proposed variant of DP-SGD that claimed very strong privacy guarantees. We then ran this mechanism M many times on two datasets D and $D \cup \{x\}$, and estimated the TPR and FPR of the LiRA attack of [CCN+22]. While the mechanism claimed $\varepsilon \approx 0.2$ privacy, we found that the actual privacy budget was at least $\varepsilon' > 2.7$. This was due to a bug in the implementation of the mechanism, which caused the added noise to be divided by the batch size m.

6.2 Understanding the Tightness of the DP-SGD Analysis

Recall the privacy analysis of the DP-SGD algorithm which we sketched in lecture 11. This analysis is known to be tight in a somewhat contrived worst-case setting, where the following conditions are met:

1. All examples in D have gradients that are zero (i.e, $g_i = \vec{0}$) while the target sample x has a (clipped) gradient of the form $g(x) = [C, 0, \ldots, 0]$. (with slightly more generality, we can also make the weaker assumption that all samples have gradients orthogonal to the gradient of x).
2. The attacker gets to observe all intermediate learning steps $\theta_0, \ldots, \theta_T$.

It is an open question whether these conditions are necessary for the DP-SGD analysis to be tight. The latter assumption, in particular, is quite unrealistic: in practice, the attacker would typically only get to see the final trained model θ_T, and not any intermediate steps.

Once we relax these assumptions, our best membership inference attacks yield lower-bounds on the privacy parameter ε that don't match the upper-bounds given by the DP-SGD analysis [JUO20, NST+21]. And so this begs the question: are our attacks not strong enough, or is the DP-SGD analysis too conservative in these settings?

This is an active area of research. It is known that in convex settings, releasing only the final model θ_T can provide better privacy guarantees than releasing all intermediate steps [FMTT18]. But it remains unclear whether a similar result holds for non-convex settings, such as training deep neural networks.

6.3 Empirical Evaluation of Heuristic Privacy Defenses

Since the DP-SGD analysis might be overly pessimistic, some researchers have also proposed heuristic privacy defenses for training deep neural networks. These defenses do not have any formal privacy guarantees, and so the only way to evaluate them is via empirical membership inference attacks.

This can be tricky though. In recent work, we showed that a number of heuristic defenses against membership inference attacks are in fact *not* effective when evaluated properly [AZT24].

7 Open Problem: White-box Membership Inference

The attacks we have discussed so far are all *black-box* attacks, since the adversary only relies on observing the model's output on some target point x, and not any of the actual learned parameters θ.

Intuitively, we would expect a *white-box* attack to be much stronger, since the adversary has access to a lot more information about the learned model. Some works leveraged this additional information to build attacks that beat the naive global loss thresholding attack from Section 5.1 [NSH19].

Yet, we now know that these attacks were not particularly strong to begin with, and that per-sample likelihood ratio attacks from Section 5.3 are in fact a lot stronger. It is an open question whether white-box access to the model's parameters could help us build a stronger likelihood ratio test. The idea here would be to estimate the probabilities

$$\Pr[\theta \mid H_b] \tag{9}$$

but we don't know how to do this for multiple reasons. First, the parameters θ are high-dimensional, and so empirical estimation might require training millions of models. And even if we did that, the problem is that every time we train a new model, the ordering of the parameters might completely change (neural networks contain a number of *symmetries*, e.g., permuting all weights in one layer and applying the inverse permutation to the next layer yields an equivalent model). And so even if we had millions of samples of the parameters θ, for models trained both with and without the target sample x, it is not obvious how to use these to estimate the likelihoods in eq. (9).

If you have any ideas about how to approach this problem, please let me know! This is a fascinating open research problem, and thus a great place to end this course. As mentioned in the first lecture, this might be your *first* and *last* class on privacy enhancing technologies, so now's the time to go out there and apply what you learned! I hope you enjoyed the ride and got a flavor of the amazing results and open research questions in this area.

References

[AZT24] Michael Aerni, Jie Zhang, and Florian Tramèr. Evaluations of machine learning privacy defenses are misleading. In *Proceedings of the 2024 ACM SIGSAC Conference on Computer and Communications Security*, 2024.

[BUV14] Mark Bun, Jonathan Ullman, and Salil Vadhan. Fingerprinting codes and the price of approximate differential privacy. In *Proceedings of the forty-sixth annual ACM symposium on Theory of computing*, pages 1–10, 2014.

[CCN+22] Nicholas Carlini, Steve Chien, Milad Nasr, Shuang Song, Andreas Terzis, and Florian Tramèr. Membership inference attacks from first principles. In *2022 IEEE Symposium on Security and Privacy (SP)*, pages 1897–1914. IEEE, 2022.

[DSS+15] 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. IEEE, 2015.

[FMTT18] Vitaly Feldman, Ilya Mironov, Kunal Talwar, and Abhradeep Thakurta. Privacy amplification by iteration. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 521–532. IEEE, 2018.

[HSR+08] 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 genetics*, 4(8):e1000167, 2008.

[JUO20] Matthew Jagielski, Jonathan Ullman, and Alina Oprea. Auditing differentially private machine learning: How private is private SGD? *Advances in Neural Information Processing Systems*, 33:22205–22216, 2020.

[KOV15] Peter Kairouz, Sewoong Oh, and Pramod Viswanath. The composition theorem for differential privacy. In *International conference on machine learning*, pages 1376–1385. PMLR, 2015.

[NP33] Jerzy Neyman and Egon Sharpe Pearson. IX. on the problem of the most efficient tests of statistical hypotheses. *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character*, 231(694-706):289–337, 1933.

[NSH19] 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)*, pages 739–753. IEEE, 2019.

[NST+21] Milad Nasr, Shuang Songi, Abhradeep Thakurta, Nicolas Papernot, and Nicholas Carlin. Adversary instantiation: Lower bounds for differentially private machine learning. In *2021 IEEE Symposium on security and privacy (SP)*, pages 866–882. IEEE, 2021.

[SSSS17] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *2017 IEEE symposium on security and privacy (SP)*, pages 3–18. IEEE, 2017.

[TTS+22] Florian Tramer, Andreas Terzis, Thomas Steinke, Shuang Song, Matthew Jagielski, and Nicholas Carlini. Debugging differential privacy: A case study for privacy auditing. *arXiv preprint arXiv:2202.12219*, 2022.

[YGFJ18] Samuel Yeom, Irene Giacomelli, Matt Fredrikson, and Somesh Jha. Privacy risk in machine learning: Analyzing the connection to overfitting. In *2018 IEEE 31st computer security foundations symposium (CSF)*, pages 268–282. IEEE, 2018.

[YMM+22] 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*, pages 3093–3106, 2022.