Privacy in Statistics and Machine Learning — Spring 2021
Lecture 20: Lower Bounds for Query Release

Adam Smith and Jonathan Ullman

We've seen a number of remarkable algorithms for query release, including some that allow us to answer a huge number of queries with error that goes to 0 as $n \to \infty$. But these algorithms often give incomparable bounds that apply in different parameter regimes. For example, if we want to answer k linear queries on a dataset of size n with entries in a domain of size m, then we have two broad classes of bounds corresponding to the Gaussian mechanism and MWEM,

$$\alpha_{\mathrm{GM}} \approx \frac{k^{1/2}}{n} \text{ and } \alpha_{\mathrm{MWEM}} \approx \frac{(\log m)^{1/4} \cdot (\log k)^{1/2}}{n^{1/2}} \tag{1}$$

Note that we're suppressing the dependence on ε, δ and some other factors, which is not the "interesting" part of these bounds. We can make some observations about these bounds, and formulate some questions about whether they can be improved

- When $k = o(n)$, the Gaussian mechanism gives error $\alpha_{\mathrm{GM}} = o(n^{-1/2})$. However, when $k = \Omega(n)$ then *both* bounds are $\Omega(n^{-1/2})$. Is there a mechanism that answers n queries with error $o(n^{-1/2})$?
- When we ask a huge number of queries, $k = 2^n$, then $\alpha = \Omega(1)$. So even though we can ask a huge number of queries, we cannot ask an arbitrary number of queries. Is there a mechanism that answers 2^n queries with error $o(1)$?[1]
- When the data domain is extremely large, for example $m = 2^k$ then MWEM offers no improvement at all over the Gaussian mechanism, and when $m = 2^{n^2}$ then it does not give any non-trivial bound on the error. The same is true for all of the other "advanced" algorithms we've seen for query release. Is there a mechanism that improves over the Gaussian mechanism when the data domain is extremely large, or infinite?

We're going to see that the answer to all of these questions is *no* in the worst case! The first two limitations turn out to be a consequence of the *reconstruction attacks* that we studied in Lectures 2 & 3. We'll also see that the third limitation is a consequence of something we'll introduce called *membership-inference attacks* that we'll introduce at a high level.

1 Lower Bounds from Reconstruction Attacks

We can answer the first two questions almost immediately just by recalling the model of reconstruction and the results we proved, and doing a bit of translation.

Recall that in our model of reconstruction, we considered datasets of the form on the left To simplify things we will assume that the identifiers are fixed to be $1, 2, \ldots, n$ so that the identifier for the i-th user is just the unique id i. Moreover, we'll assume that the secrets are bits in $\{0, 1\}$. Since we're interested in

[1]We saw such a mechanism, the projection mechanism, whose error bound has no dependence at all on the number of queries k. However, as we will see, that result crucially relies on the fact that we only bounded the weaker ℓ_2 error.

Identifiers	Secret
z_1	s_1
z_2	s_2
z_3	s_3
$\vdots$	$\vdots$
z_n	s_n

$$\Longrightarrow$$

Identifiers	Secret
1	0
2	1
3	1
$\vdots$	$\vdots$
n	0

worst-case lower bounds, we're allowed to make these choices. With these simplifications we now have a dataset $\mathbf{x} \in \mathcal{U}^n$ where $\mathcal{U} = [n] \times \{0, 1\}$.

We also considered a special case of count queries, but we normalized them differently. Specifically, when we studied reconstruction, we considered queries of the form

$$f(\mathbf{x}) = \sum_{i=1}^{n} \varphi(z_i) s_i \tag{2}$$

However, these are just a special case of linear queries, with a different normalization. Specifically, if we define $\varphi'(z_i, s_i) = \varphi(z_i)s_i$ then we get

$$f'(\mathbf{x}) = \frac{1}{n} \sum_{i=1}^{n} \varphi'(z_i, s_i) = \frac{f(\mathbf{x})}{n} \tag{3}$$

The upshot is that if we allow a user to ask arbitrary linear queries on a dataset of the form we described, then they have the ability to ask queries of the form we need for reconstruction attacks, provided that they rescale by a factor of n.

Thus, we can restate our results about reconstruction as follows:

Theorem 1.1 (Restated from Lectures 2 & 3 and Homework 1)**.** *If an attacker is allowed to ask k arbitrary linear queries, and receives answers to each query with error $\leq \alpha$, then (with high probability) then the attacker can recover the secret bits with 90% accuracy for any of the following choices of k and α:*

1. *If $k = 20n$ and $\alpha = \frac{1}{Cn^{1/2}}$ for sufficiently large constant C.*
2. *If $n \ll k \ll 2^n$ and $\alpha = \frac{\log(k/n)^{1/2}}{Cn^{1/2}}$ for sufficiently large constant C.*
3. *If $k = 2^n$ and $\alpha = .025$.*

We also proved on an in-class exercise that any algorithm that when the secret bits $s_1, \ldots, s_n$ are chosen uniformly at random, then no (ε, δ)-differentially private algorithm, for sufficiently small constant ε, δ, allows an attacker to recover the secret bits with 90% accuracy on average. Therefore, we obtain the following lower bounds for differential privacy.

Theorem 1.2. *For any n and any $20n \leq k \leq 2^n$,* every $(\frac{1}{10}, \frac{1}{10})$*-differentially private algorithm that answers k arbitrary linear queries on a dataset of size n over a domain of size $m \geq 2n$ has error*

$$\alpha_{LB} = \Omega\left(\frac{\log(k/n)^{1/2}}{n^{1/2}}\right)$$

As we can see, this answers the first two of our motivating questions. Before going on, we can compare this lower bound to α_{MWEM}, which was

$$\alpha_{\mathrm{MWEM}} = O\left(\frac{(\log m)^{1/4}(\log k)^{1/2}}{n^{1/2}}\right)$$

Notice that when $m = 2n$ as in our reconstruction attacks, these lower bounds match up to factors that are at most polynomia in $\log n$, so we have essentially nailed down the optimal error of any differentially private algorithm for answering arbitrary linear queries, when the universe is small!

However, when the universe is large, or even infinite, the gap between the lower and upper bounds is this $(\log m)^{1/4}$. That might not sound like much, but when the data comes from the domain $\mathcal{U} = \{0,1\}^d$ then this factor becomes $d^{1/4}$. Since real datasets are often *high dimensional*, meaning d is large, this additional $d^{1/4}$ factor can add up very quickly. In particular, if the dimension of the dataset is larger than n^2, MWEM gives no non-trivial accuracy guarantee, even when $k = n^2$, while the lower bound is only about $1/n^{1/2}$, and we'd like to understand which of these bounds is really capturing the cost of privacy. In the next secton we'll focus on understanding the role of the dimension and why it can't be avoided.

2 Membership-Inference Attacks

Membership-inference attacks were introduced in an influential empirical paper by Homer et al. [HSR+08], which applied them to public datasets from so-called genome-wide association studies. Their was then modeled and analyzed by Sankararaman et al. [SOJH09]. They later turned out to be an important tool for establishing the limits of differential privacy [BUV14, DSS+15], and have since become a standard tool for identifying and measuring privacy risks in large-scale machine learning (e.g. [SSSS17, YGFJ18, JUO20])

2.1 The Attacker Model

Often differential privacy is interpreted as saying that the attacker cannot distinguish between two worlds—one where the mechanism is run on some dataset $\mathbf{x}$ and a hypothetical world where the mechanism is run on some dataset $\mathbf{x}_{-i}$ where user i's data has been removed or replaced with some dummy value. Thus, if an attacker can look at the output y of a mechanism M and determine whether y is the result of $M(\mathbf{x})$ or $M(\mathbf{x}_{-i})$ then M *cannot* be differentially private.

> A membership-inference attack is exactly that—an attack where an attacker is given the output y of some mechanism $M(\mathbf{x})$ and some datapoint z, and tries to determine if z was or was not included in the dataset $\mathbf{x}$.

Note that this is an informal defintiion and omits some crucial details of what makes such an attack a true violation of privacy, or a contradiction to differential privacy, which we'll return to later.

There are multiple ways to model membership-inference attacks, but we will consider the following model, which is pretty much standard, and leads to very convincing attacks:

- There is some distribution P over the data domain $\mathcal{U}$ and some mechanism $M : \mathcal{U}^n \to \mathcal{Y}$.
- The data is drawn iid from that distribution $x_1, \dots, x_n \sim P$.
- A *target* z is chosen in one of two ways:
  OUT: z is drawn from P, independent of $x_1, \dots, x_n$
  IN: $z = x_i$ for a randomly chosen $i \in \{1, \dots, n\}$.
- We obtain some output $y \sim M(\mathbf{x})$

- The attacker is given (1) the distribution P, (2) the output y and (3) the target point z and has to decide whether z is IN or OUT.

Exercise 2.1. Suppose that for a particular mechanism M and distribution P, there is an attacker who correctly identifies whether the target is IN or OUT with 90% accuracy (i.e. if the target is IN the attacker says IN at least 90% of the time and the same if the target is OUT). Prove that M cannot be $(1, \frac{1}{10n})$-differentially private.

2.1.1 Interpreting Membership-Inference Attacks.

One important thing to note is that regardless of whether z is IN or OUT, z is drawn from P. In other words, the *only* thing that differentiates the IN and OUT cases is whether or not z was given to the mechanism M. This condition is crucial for proving that membership-inference contradicts differential privacy, and for interpreting membership-inference as a meaningful attack. For example, suppose the dataset represents a large fraction of the people living in the town of Framingham, MA, USA.[2] Then a reasonable attack strategy would be to say IN if the target z is a person in Framingham and OUT if the target does not live in Framingham. Although the attacker has a pretty good chance of succeeding, it's hard to argue that the mechansim is at fault, since the attacker could succeed using only the publicly available information that the study is taking place! The way we've defined membership inference ensures that the attacker can only succeed if the mechansim is leaking information about the members.

Another thing to note is that the attacker *already has all the data of the target z*. The only thing the attacker is trying to learn is whether or not z's data was included in the training data. In some settings, once the attacker knows all the data of the user, it might seem like the privacy of that user is lost, however in many cases the information about whether or not the target is in the dataset reveals additional information beyond what is contained in z. For example, suppose the dataset $\mathbf{x}$ represents users in a study of patients with diabetes, and their data is demographic information. Then an attacker who knows a user's demographic information, *but not whether this patient has diabetes* can infer whether or not this user is in the dataset, which would potentially reveal that the patietn does have diabetes.

2.2 Lower Bounds via Membership Inference

In particular, we will use membership inference attacks to argue that the Gaussian mechanism cannot be significantly improved when the data domain is sufficiently large. In particular, we will focus on the following special case of linear query release. The data domain will be $\mathbf{x} = (x_1, \dots, x_n)$ where $x_i \in \mathcal{U} = \{\pm 1\}^k$, corresponding to each user answering k yes or no questions,[3] and we will ask a set of exactly k linear queries of the form

$$\forall j = 1, \dots, k \quad f_j(\mathbf{x}) = \frac{1}{n} \sum_{i=1}^{n} x_{i,j} \tag{4}$$

Note that these queries are a fancy way of asking for the *mean* of the dataset because when we define $F(\mathbf{x}) = (f_1(\mathbf{x}), \dots, f_k(\mathbf{x}))$ then

$$F(\mathbf{x}) = \frac{1}{n} \sum_{i=1}^{n} x_i \tag{5}$$

[2]Framingham, MA is famously the site of a multi-generational study on cardiovascular health, and a significant fraction of its residents participate in this study.

[3]Note that we will switch from our usual domain $\{0,1\}^k$ to $\{\pm 1\}^k$ because it will make the analysis slightly simpler, but this change does not substantively affect any of the statements we will make.

Observe that, for these queries, since the number of queries is k and the domain size is 2^k, the Gaussian mechanism adds noise $\approx k^{1/2}/n$ and MWEM has worse error $\approx (k^{1/2}/n)^{1/2}$, so when $k \gg n^2$, both algorithms fail to give any non-trivial error guarantee. We can in fact show that this is inherent for any differentially private algorithm.

Theorem 2.2 ([BUV14, DSS⁺15]). *For every n and k, there is a family of k linear queries over the data domain $\{\pm 1\}^k$ such that any $(1, \frac{1}{10n})$ differentially private algorithm must have error $\alpha = \Omega(k^{1/2}/n)$.*

Proving this theorem would require introducing more machinery than we have time for, but instead we will show how one can carry out a membership-inference attack against any mechanism that has the form of the Gaussian mechanism, but with an insufficient amount of noise. That is, the mechanism

$$M(\mathbf{x}) = F(\mathbf{x}) + \mathcal{N}(\mathbf{0}, \sigma^2 \cdot \mathbb{I}_{k \times k}) \tag{6}$$

for $\sigma = o(k^{1/2}/n)$, which is (just barely) too small for us to argue that this mechanism is private.

First, since we're proving lower bounds, we get to choose a distribution P for the data, and we'll choose to make P uniform over $\{\pm 1\}^k$. Now we will define the attacker:

1. The input is the distribution P, the output y, and the target point z.
2. Compute the *test statistic* $T = \langle y, z \rangle$
3. If $T \geq \tau$, output IN, otherwise output OUT. Here, τ is some threshold that we will set based on P.

Intuitively, the attacker is considering two possible worlds, one where the target is IN and one where the target is OUT and designing some test statistic T that will *distinguish* between these two worlds. The intuition for this choice is that, since $y \approx \frac{1}{n}\sum_{i=1}^{n} x_i$, when z is IN the dataset then y is like an average of z with lots of other random points, and thus y should have some small correlation with z in expectation. On the other hand when z is OUT of the dataset, y is an average of completely independent random points, so y and z are uncorrelated in expectation. For example, in the very special case where $n = 1$, then when z is IN the dataset we have $y = z$, so $\langle y, z \rangle = k$ but when z is OUT of the dataset then y, z are independent so $\langle y, z \rangle \approx \sqrt{k}$.

In order to see what is going on, we can actually run the membership-inference experiment many trials, and plot the *distribution* of the test statistic over these trials. This plot shows the results: Note

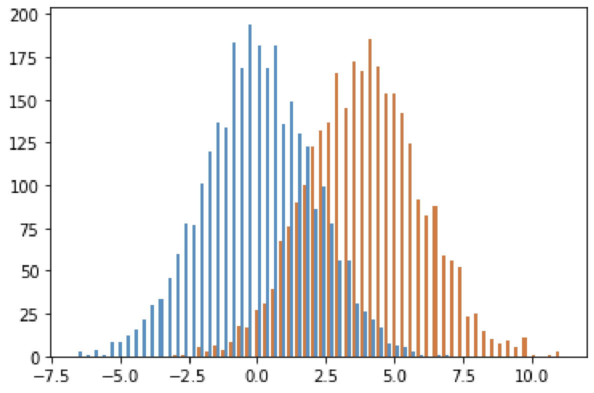

Figure 1: The distribution of the test statistic when z is IN (orange) and OUT (blue). Here $n = 25, d = 100$ and there is no noise added to the empirical mean.

that the distribution of the test statistic ends up being roughly like a Gaussian. Also, the variance is roughly the same in the two case. The main difference is that the means are different.

Exercise 2.3. Prove that when $\sigma = 0$ and z is OUT of the dataset, we have

$$\mathbb{E}\left(\langle y, z\rangle\right) = 0 \text{ and } \mathrm{Var}\left(\langle y, z\rangle\right) = \frac{k}{n}$$

and that when z is IN the dataset, we have

$$\mathbb{E}\left(\langle y, z\rangle\right) = \frac{k}{n} \text{ and } \mathrm{Var}\left(\langle y, z\rangle\right) = \frac{k}{n-1}$$

Conclude that we can distinguish IN from OUT with 90% accuracy when $d \geq Cn$ for some constant n.

What we can show in general is that if $\sigma \leq ck^{1/2}/n$ for some small constant $c > 0$, then the membership-inference attack succeeds with 90% accuracy.

In order to prove Theorem 2.2 we need some additional ideas, but the attack strategy and the high-level intuition is the same—we argue that when the error is $o(k^{1/2}/n)$, no matter what algorithm we use to add the noise, the output $y \approx \frac{1}{n}\sum_{i=1}^{n} x_i$ must be correlated with at least one of its inputs x_i in a significant and detectable way.

2.3 Membership Inference in Machine Learning

Machine learning currently provides one of the most interesting application domains for differential privacy. Although most realistic models/algorithms for ML are too complex to precisely analyze membership-inference attacks, that doesn't mean they don't happen! In fact, it's possible to perform membership inference against most of the large models used in ML today. These attacks also highlight an interesting connection between membership-inference and *overfitting to training data.*

The standard thousand-foot view of machine learning is as follows:

- We have a distribution P over a data domiain $\mathcal{U}$ and we get a dataset $\mathbf{x} = (x_1, \ldots, x_n)$ consisting of n iid samples from P.
- We have a space of *models* Θ and a loss function $\mathcal{L} : \Theta \times \mathcal{U} \to \mathbb{R}$ that maps a model θ and a data point P to some *loss.*
- Given the dataset $\mathbf{x}$, we compute some $\hat{\theta}$ that (approximately) minimizes the average loss *on the samples*
$$\frac{1}{n}\sum_{i=1}^{n} \mathcal{L}(\hat{\theta}, x_i)$$
- We hope that
$$\mathop{\mathbb{E}}_{x\sim P}\left(\mathcal{L}(\hat{\theta}, x)\right) \approx \min_{\theta\in\Theta} \mathop{\mathbb{E}}_{x\sim P}\left(\mathcal{L}(\hat{\theta}, x)\right)$$
. That is, we hope that $\hat{\theta}$ minimizes the loss *on the distribution* rather than just minimizing the loss *on the samples.*

Just so this doesn't look wildly abstract, a common example is *linear regression* where we have a distribution P over pairs $(x, y) \in \mathbb{R}^d \times \mathbb{R}$ and we assume that $y \approx \langle \theta, x\rangle$, so that the variable y is approximately a linear function of the variables x. Our goal is to find θ that minimzes the *squared loss*

$$\mathcal{L}(\theta, P) = \mathop{\mathbb{E}}_{x,y\sim P}\left((\langle \theta, x\rangle - y)^2\right)$$

Exercise 2.4. Suppose P is a distribution over $\mathbb{R}^d$ and we'd like to estimate its mean $\mathbb{E}\left(P\right)$. How can we formalize this problem in the framework above? What is the set of possible models Θ and what loss function $\mathcal{L}$ should we use so that the minimizer of the loss function is the mean?

How can we perform membership inference attacks on machine-learning methods that output a model θ? At first glance it seems like it should require very intricate knowledge of the type of model, the loss function, and the learning algorithm. However, Yeom et al. [YGFJ18] observed that a very simple method generally works well by taking advantage of *overfitting*.

To see what is going on, consider the example of linear regression above. If we draw $n \leq d$ samples, then there is a linear function $\hat{\theta}$ such that $y_i = \langle \theta, x_i \rangle$ for every point, so $\mathcal{L}(\hat{\theta}, (x_i, y_i)) = 0$ for every sample (x_i, y_i). On the other hand, if we're given a random $(x, y) \sim P$ then the loss is probably *not* exactly 0. Thus, if we take the target point z and compute $\mathcal{L}(\hat{\theta}, z)$ we can probably tell whether z is IN the dataset by checking to see whether the loss is 0 or non-zero. This example is a bit specialized, but it's an example of a more general phenomenon

All models arising in machine learning fit their training data better than they fit unseen data.

This phenomenon has far-reaching consequences that extend far beyond membership-inference, but for our purposes it suggests the following membership-inference attack:

1. The input is the distribution P, the model $\hat{\theta}$, and the target point z.
2. Compute the *test statistic* $T = \mathcal{L}(\hat{\theta}, z)$
3. If $T \geq \tau$, output IN, otherwise output OUT. Here, τ is some threshold that we will set based on P.

As before, we can see how this plays out for our example of linear regression by plotting the distribution of the test statistic in the IN and OUT cases.

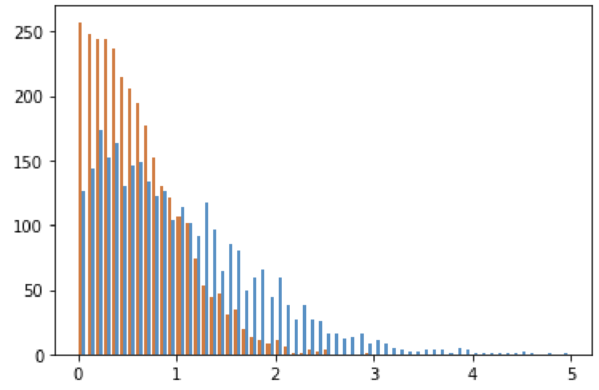

Figure 2: The distribution of the test statistic when z is IN (orange) and OUT (blue). Here $n = 100, d = 40$. Observe that the loss for points IN the dataset is typically smaller than for those OUT of the dataset.

Summary

Key Points

- ...

Additional Reading

- A survey on privacy attacks against aggregate statistics [DSSU17]
- Another technique for proving lower bounds against $(\varepsilon, 0)$-differentially private algorithms is called *packing*, which was introduced in [HT10, BKN10]. See [Vad16] for a good introduction.

References

[BKN10] Amos Beimel, Shiva Prasad Kasiviswanathan, and Kobbi Nissim. Bounds on the sample complexity for private learning and private data release. In *Theory of Cryptography Conference*, TCC '10, 2010.

[BUV14] Mark Bun, Jonathan Ullman, and Salil Vadhan. Fingerprinting codes and the price of approximate differential privacy. In *ACM Symposium on the Theory of Computing*, STOC '14, 2014. https://arxiv.org/abs/1311.3158.

[DSS$^+$15] 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.

[DSSU17] Cynthia Dwork, Adam Smith, Thomas Steinke, and Jonathan Ullman. Exposed! A Survey of Attacks on Private Data. *Annual Review of Statistics and Its Application*, 4:61–84, 2017.

[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.

[HT10] Moritz Hardt and Kunal Talwar. On the geometry of differential privacy. In *Proceedings of the 42nd ACM Symposium on Theory of Computing, STOC*, 2010.

[JUO20] Matthew Jagielski, Jonathan Ullman, and Alina Oprea. Auditing differentially private machine learning: How private is private SGD? NeurIPS '20, 2020. https://arxiv.org/abs/2006.07709.

[SOJH09] 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.

[SSSS17] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *IEEE Symposium on Security and Privacy (S&P), Oakland*, 2017.

[Vad16] Salil Vadhan. The complexity of differential privacy. *http://privacytools. seas. harvard. edu/publications/complexity-differential-privacy*, 2016.

[YGFJ18] Samuel Yeom, Irene Giacomelli, Matt Fredrikson, and Somesh Jha. Privacy risk in machine learning: Analyzing the connection to overfitting. In *IEEE Computer Security Foundations Symposium*, CSF '18, 2018. https://arxiv.org/abs/1709.01604.