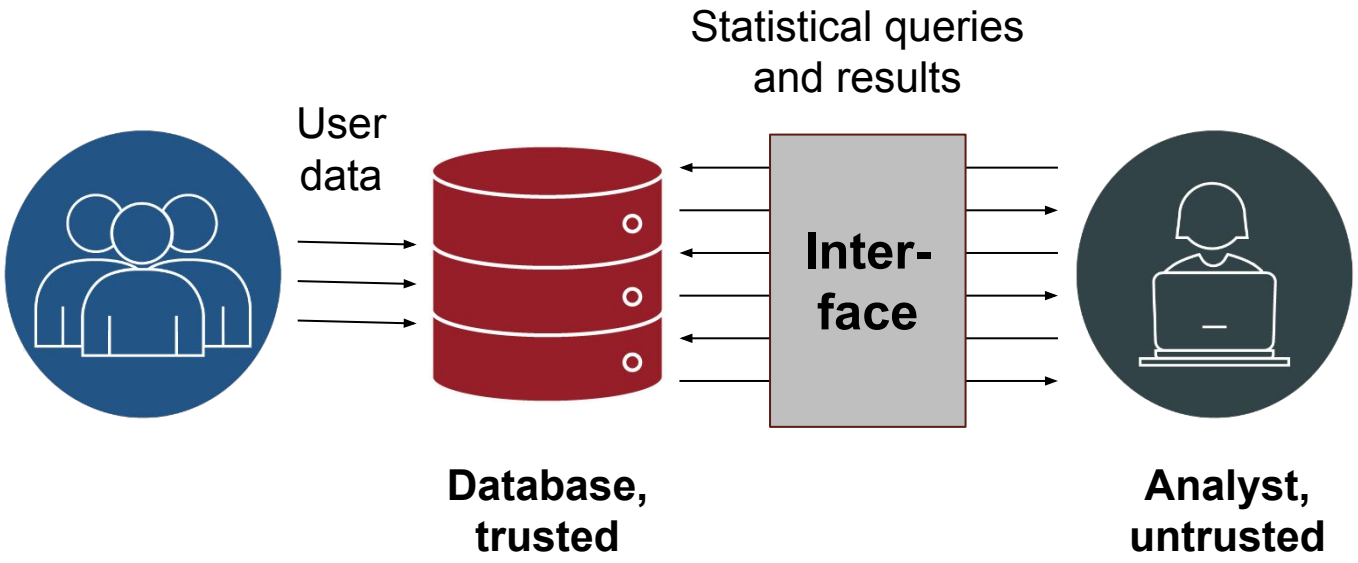
Privacy-Preserving Systems (a.k.a., Private Systems)

CU Graduate Seminar

Instructor: Roxana Geambasu

Differential Privacy

Problem: Privacy-preserving statistical analyses



Goal: *develop an interface that lets the analyst compute statistical queries without increasing the* ***privacy exposure*** *of individuals in the database to the analyst*

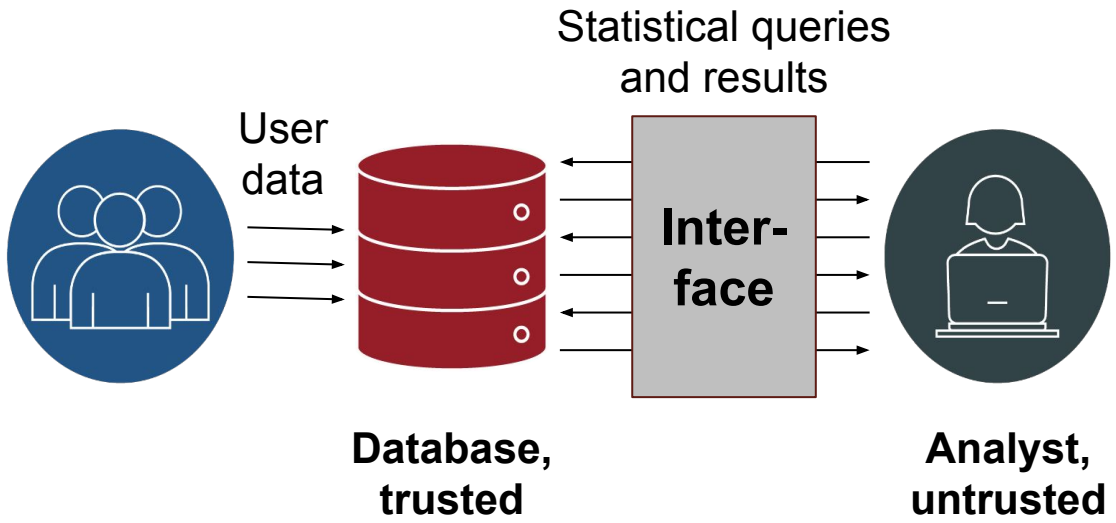
Recap: Take-aways from last time

1. Even without "PII," it's possible to learn sensitive info about individuals from data releases, especially with **side information** or **multiple queries** (a.k.a. **attacker context**).
2. It's difficult to determine what's "okay to release," because vulnerability to attack depends on **data distribution** and **attacker context.**
3. Ad-hoc solutions (incl. anonymization, k-anonymity, aggregates-only) are unreliable, because they too depend on **data distribution** and **attacker context**.
4. Today: **differential privacy**, a rigorous privacy technology to establish "what's okay to release" that does NOT depend on these!

Defining Privacy in Statistical Analyses

Requirements

- We need the interface to enforce a **rigorous definition of privacy**
- Requirements:
  1. **Resilient to side information**
  2. **Persist under repeated queries** (aka, closed under composition)
  3. **Independent of data distribution**



Strawman Definition 1

[Dalenius-77] *Access to the results of the analysis should not enable one to learn anything about any individual that one would not learn without access to the results.*

- Can be formalized, meets our requirements, and maps well onto ***semantic security*** (crypto's standard for message secrecy)
- Problem: not achievable for the statistical analysis [Dwork+10] because learning about populations implies learning about individuals

Strawman Definition 2

Definition: *Access to the results of the analysis should not enable one to learn anything about any individual in the dataset that one would not learn* ***if the individual was not in the dataset****.*

Problem: still not achievable, because the result of the analysis cannot be independent of *all* individuals in the dataset

Strawman Definition 2 (cont.)

Definition: *Access to the results of the analysis should not enable one to learn **anything** about **any individual** in the dataset that one would not learn if the individual was not in the dataset.*

Problem: still not achievable, because the result of the analysis cannot be independent of *all* individuals in the dataset

Maybe weaken one of the bolded terms above?

Strawman Definition 3

Definition: *Access to the results of the analysis should not enable one to learn* ***anything*** *about* **some (most?) individuals** *in the dataset that one would not learn if the individual was not in the dataset.*

- Can be achieved (e.g., by sampling a few individuals from the dataset and performing the computation only on data of sampled individuals)
- For the lucky individuals we didn't sample, this definition meets at least the second requirement, but not for the unlucky others

Strawman Definition 4

Definition: *Access to the results of the analysis should not enable one to learn* **anything new confidently** *about* ***any individual*** *in the dataset that one would not learn if the individual was not in the dataset.*

Strawman Definition 4 (cont.)

Definition: *Access to the results of the analysis should not enable one to learn* **anything new confidently** *about* ***any individual*** *in the dataset that one would not learn if the individual was not in the dataset.*

- But what do **“anything new”** and **“confidently”** mean?

Strawman Definition 4 (cont.)

Definition: *Access to the results of the analysis should not enable one to learn* **anything new confidently** *about* ***any individual*** *in the dataset that one would not learn if the individual was not in the dataset.*

- But what do **“anything new”** and **“confidently”** mean?
- **Differential privacy formalizes these.**

References Cited

[Delanius-77] Dalenius. *Towards a methodology for statistical disclosure control.* Statistik Tidskrift 15, pp. 429–444, 1977.

Questions

- How would you define ***"anything new"*** and ***"confidently"*** to construct a privacy definition suitable for statistical analyses?

Proposed definition:

Access to the results of the analysis should not enable one to learn ***anything new confidently*** *about any individual in the dataset that one would not learn if the individual was not in the dataset.*

The End

Differential Privacy

Differential Privacy (DP)

Lets us reason about **how much** statistical information, and **how accurate**, is **"safe to share"** in the face of an adversary with arbitrary side information and infinite computational power

Definition: A randomized query $f : X \to Y$ is ε– DP if for any pair of databases $x, x' \in X$ differing in one entry and for any output set $S \subset Y$:

$$\Pr(f(x) \in S) \leq e^{\varepsilon} \Pr(f(x') \in S).$$

The probabilities are taken over the randomness in f.

(Dwork, 2006)

Bayesian Interpretation

- Consider a prior distribution (X, X') on neighboring databases, modeling an adversary's *prior belief* on a real database, X, and a database X' that would have been obtained if a particular individual had not participated
- Given an output *y* from a ε-DP randomized function *f*, the adversary will have a *posterior belief* on the database: $X|_{f(X)=y}$.
- The definition of DP implies that this posterior distribution is close (in statistical distance, SD) to the posterior that would have been obtained if the function *f* had been run on *X'* instead of *X* (Vadhan, 2016)

$$SD(X|_{f(X)=y}, X|_{f(X')=y}) \leq 2\varepsilon.$$

Bayesian Interpretation (cont.)

- In particular, if the adversary's prior included all the information about X except for the *i*'th row (the data of individual *i*), then the adversary's posterior on row *i* would have been close to their prior on row *i*
- In that sense, the adversary does not learn **"anything new"** about any individual (i.e., that they couldn't have learned from the rest of the database)

Strawman Definition 4

Access to the results of the analysis should not enable one to learn **anything new** **confidently** *about any individual in the dataset that one would not learn if the individual was not in the dataset.*

Hypothesis Testing Interpretation

- Consider an adversary who wants to test, based on the output of a DP query, the null hypothesis that an individual, *i*, has contributed their data to a database *x*.
- The definition of DP has been proven to be equivalent to requiring that *any hypothesis test* has either *low significance* (it has high false-positive rate, FPR), or *low power* (it has high false-negative rate, FNR) (Wasserman, 2010)

$$FPR(f,x,x',S)+e^{\varepsilon}FNR(f,x,x',S)\geq 1 \quad \text{and}$$

$$e^{\varepsilon}FPR(f,x,x',S)+FNR(f,x,x',S)\geq 1 \quad (\forall x,x'.x\sim x',\forall S\subset Y)$$

Statistical Testing Interpretation

- The parameter **epsilon** controls the trade-off between significance vs. power of any hypothesis test.
- In that sense, the adversary cannot test "**confidently**" whether a particular individual was in the dataset that was used to produce a particular output.

Strawman Definition 4

Access to the results of the analysis should not enable one to learn **anything new confidently** *about any individual in the dataset that one would not learn if the individual was not in the dataset.*

Lay Interpretation and Cautions

- Whatever an adversary learns about you, they did not learn it **because** of the use of your data in the DP query; they could have learned the same thing even if you hadn't contributed your data
- This does not mean that the adversary cannot learn **anything** about you from the output of a DP query!
- Example
  - Adversary learns that smoking correlates with lung cancer.
  - Adversary knows that X smokes.
  - He can deduce that X is more likely to get cancer than a nonsmoker.
  - ***But*** this deduction was not **caused** by X's participation in the study.

Example Use of DP

Company X's
Terms and Conditions

[…]

You agree to allow Company X to use your data to compute and share results from statistical analyses under 0.1-user-level differential privacy.

Clicking YES vs. NO does not increase the privacy risk for any individual by more than 11%.

- If the risk of anyone learning a particular aspect about you was low, it remains low despite Company X's use of your data.

Cited References

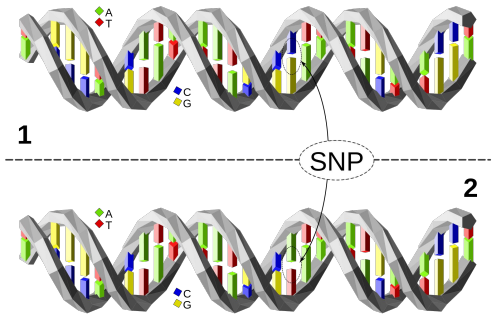
(Dwork, 2006) Dwork, McSherry, Nissim, & Smith. (2006). *Calibrating noise to sensitivity in private data analysis.* TCC.

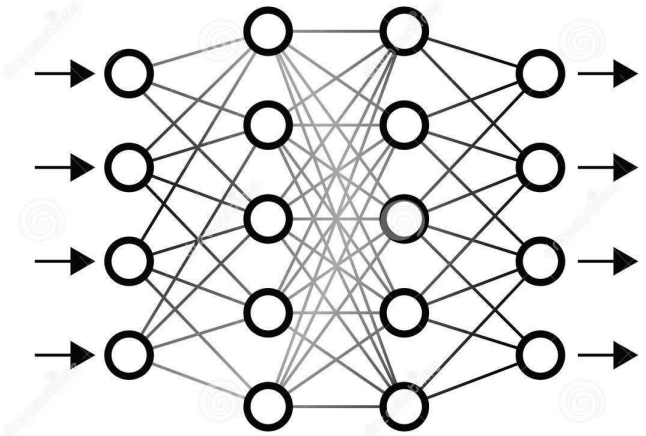
(Vadhan, 2016) Vadhan. *The complexity of differential privacy.* https://privacytools.seas.harvard.edu/files/privacytools/files/complexityprivacy_1.pdf.

(Wasserman, 2010) Wasserman & Zhou. (2010). A statistical framework for differential privacy. *Journal of the American Statistical Association*.

Questions

- Considering DP's interpretations, reason about how DP prevents the privacy attacks against aggregates that we discussed in previously:
  - Membership inference attacks
  - Data reconstruction attacks
  - ML memorization-based attacks





The End

DP Parameters and Properties

The Epsilon Parameter

- All interpretations we gave assume **“low” epsilon**
- Intuitively, epsilon bounds the privacy loss of any individual through the output of the statistical analysis
- Smaller values mean better privacy but hurt accuracy
  - For example, for many statistical analyses, epsilon < 1/n doesn’t make sense because DP error accumulates as $o(1/n)$
- Rule of thumb: epsilon = 0.1 for simple statistics, though for ML, epsilon = 1 or 10 are needed

DP Variant: (ε,δ)-DP

Definition: A randomized query $f : X \to Y$ is (ε,δ)− DP if for any pair of databases $x, x' \in X$ differing in one entry and for any output set $S \subset Y$:

$$\Pr(f(x) \in S) \leq e^{\varepsilon} \Pr(f(x') \in S) + \delta.$$

The probabilities are taken over the randomness in f.

- Adds an additive **delta** term to the original definition
- Enables randomization mechanisms that improve the privacy–accuracy trade-off
- Roughly interpreted as *"ε − DP with probability at least (1 − δ)"*

The Delta Parameter

- Aim for values of delta that are **less than the inverse of a super-linear polynomial in database size (*n*).**
- Delta = $1/n$ is dangerous: permits preserving “privacy” by publishing the complete records of a small number of database participants!
- Rule of thumb: delta = $1/n^2$ is generally acceptable.

Properties

- Property 1: closure under post-processing
- Property 2: closure under composition
- Property 3: independent of data distribution

Property 1: Closure Under Post-Processing

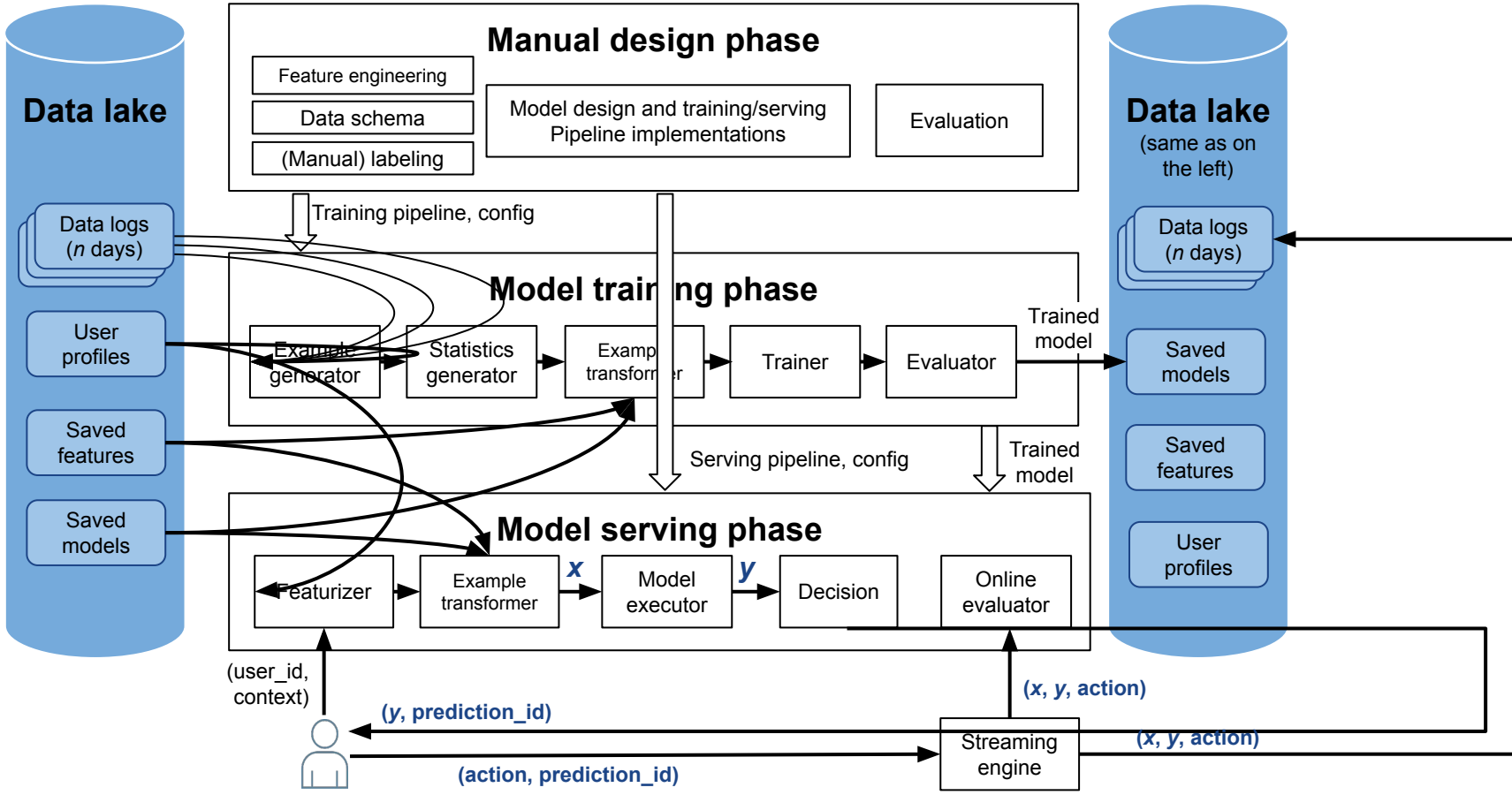
If f is (ε, δ)-DP then for any g we have: $g \circ f$ is (ε, δ)-DP.

- Trivially implies **resilience to arbitrary side information,**
  - **This was our first requirement**
- Can allow **safe**, **unlimited reuse of the outputs** from DP computations, such as models trained with DP training procedures
- Question: How could this property be useful in addressing data exposure risks in modern ML ecosystems?

Usefulness

Thanks to post-processing, DP could be used to:

- Safely share models and features across teams
- Safely retain models and features beyond the raw data's expiration date
- What else?



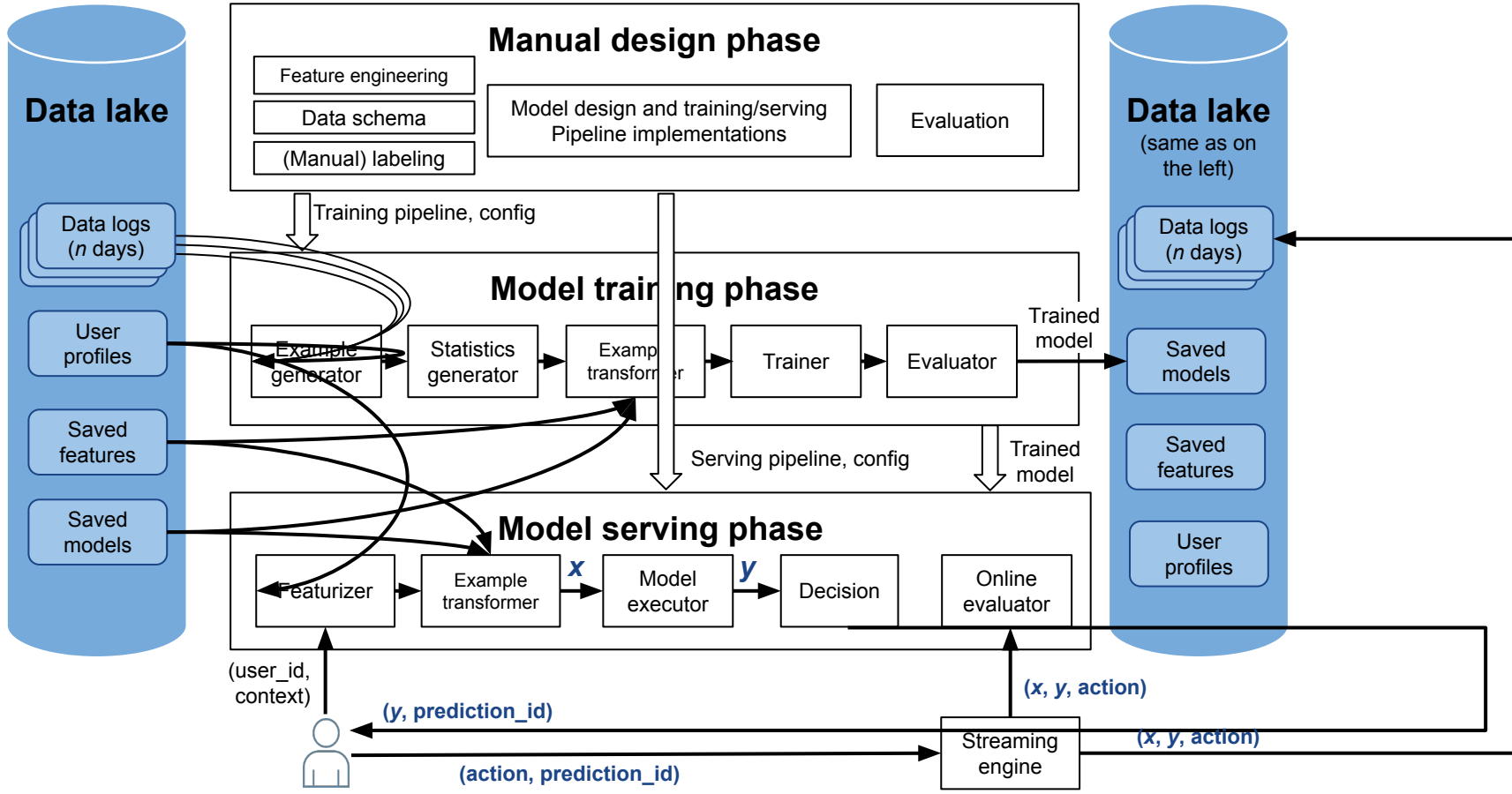
Property 2: Closure under Composition

If f1 is $(\varepsilon 1, \delta 1)$-DP and f2 is $(\varepsilon 2, \delta 2)$-DP, then the computation that runs f1 and f2 on the same dataset is $(\varepsilon 1 + \varepsilon 2, \delta 1 + \delta 2)$-DP.

- **This was our third requirement**
- Bounds how much new information an analyst can learn about any individual in the database across multiple queries
- Encodes that the more things you learn from a dataset, the more you also learn about individuals
- Tighter composition formulas exist, in which individuals' privacy loss degrades as square root of the number of composed DP computations instead of linearly as above

Usefulness

- ML ecosystems release many models/statistics learned from the same users' data, so it is important to **bound the cumulative privacy risk** resulting from all releases
- Composition enables **modular development** of complex systems from small parts, such as basic DP mechanisms and algorithms



Property 3: Independent of data distribution

- DP is a property of the computation on a dataset, not of the dataset!
  - Unlike k-anonymity or other anonymization techniques

Definition: A randomized query $f : X \to Y$ is (ε, δ)- DP if for any pair of databases $x, x' \in X$ differing in one entry and for any output set $S \subset Y$:

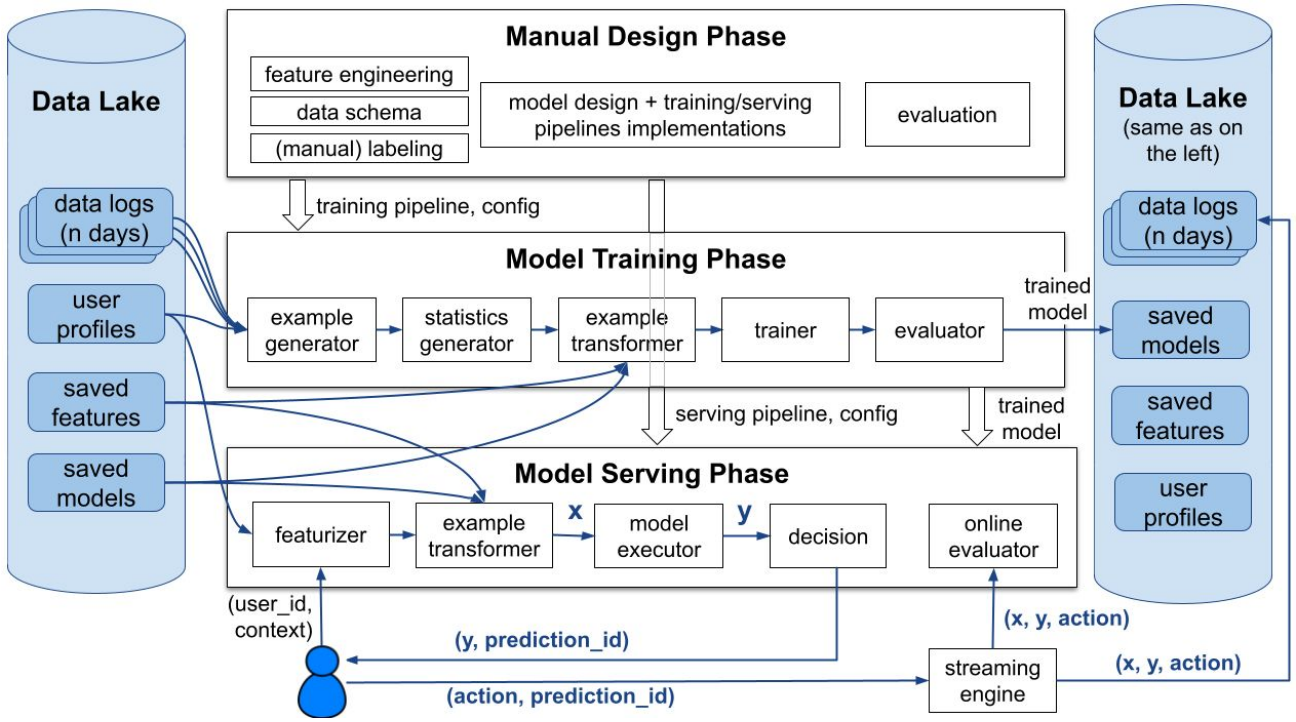
$$\Pr(f(x) \in S) \leq e^{\varepsilon} \Pr(f(x') \in S) + \delta.$$

The probabilities are taken over the randomness in f.

- **This was our third requirement**
  - But is has implications to utility, b/c it's a worst-case definition that must hold under worst-case dataset! Often noted as a disadvantage for this reason, but it's also a big advantage to not have to worry about dataset properties… It's a tradeoff, and DP, like crypto, makes worst-case assumptions…

Questions

- In what other ways can differential privacy, based on its properties, be leveraged to mitigate the data exposure risks in ML ecosystems that we previously discussed?



The End

Basic DP Mechanisms

Making Statistical Analyses DP

Basic approach to making a function DP

1. Decompose it into sub-functions for which you either have a DP implementation, or that fit into a class of functions for which you can apply one of the several **basic DP mechanisms.**
2. Use the **composition** and **post-processing** closure properties to determine the guarantee achieved by the overall function that combines the sub-functions.

Basic DP Mechanisms

- Multiple DP mechanisms exist, each supporting different classes of functions.
  1. Laplace mechanism
  2. Gaussian mechanism
  3. Exponential mechanism
  4. Smooth sensitivity mechanism
  5. Test and release mechanism
- We'll focus on the first two.

Laplace and Gaussian Mechanisms

- Suitable for some numeric functions: $f^{np}: X^n \to \mathrm{R}^k$.*
  - Counting, averaging, computing histograms, contingency tables, etc.
  - Also doing one step of gradient descent, which is an average over some real values (more on that later)
- Approach: perturb each output dimension with independent draws from a calibrated Laplace/Gaussian distribution

*I denote the non-private function as f^{np} to distinguish it from the DP version, which I've thus far been denoting as f.

Laplace Mechanism

(Dwork, 2006)

L1-Sensitivity

- Define **L1-sensitivity** of a function $f : X^n \rightarrow \mathrm{R}^k$ as:

$$\Delta_f = \max_{x,x':d(x,x')\leq 1} \| f(x) - f(x') \|_1 \qquad \left(\| f(x) - f(x') \|_1 = \sum_{i=1}^{k} | f_i(x) - f_i(x') | \right)$$

- That is, how much can one entry affect the value of the function?

L1-Sensitivity

- Define **L1-sensitivity** of a function $f : X^n \rightarrow \mathrm{R}^k$ as:

$$\Delta_f = \max_{x,x':d(x,x')\leq 1} \| f(x) - f(x') \|_1 \qquad \left(\| f(x) - f(x') \|_1 = \sum_{i=1}^{k} | f_i(x) - f_i(x') | \right)$$

- That is, how much can one entry affect the value of the function?
  - "How many people in a room have brown eyes?": Sensitivity = ?
  - "How many have brown eyes, how many have blue eyes, how many have green eyes, and how many have red eyes?": Sensitivity = ?
  - "How many have brown eyes and how many are taller than six feet?": Sensitivity = ?

L1-Sensitivity

- Define **L1-sensitivity** of a function $f : X^n \rightarrow \mathrm{R}^k$ as:

$$\Delta_f = \max_{x,x':d(x,x')\leq 1} \| f(x) - f(x') \|_1 \qquad \left(\| f(x) - f(x') \|_1 = \sum_{i=1}^{k} | f_i(x) - f_i(x') | \right)$$

- That is, how much can one entry affect the value of the function?
  - "How many people in a room have brown eyes?": Sensitivity = **1**
  - "How many have brown eyes, how many have blue eyes, how many have green eyes, and how many have red eyes?": Sensitivity = **1**
  - "How many have brown eyes and how many are taller than six feet?": Sensitivity = **2**

Laplace Distribution

- **The Laplace distribution,** *Lap(b)*, is the probability distribution with p.d.f.

$$\Pr[x] = \frac{1}{2b} e^{-\frac{|x|}{b}}$$

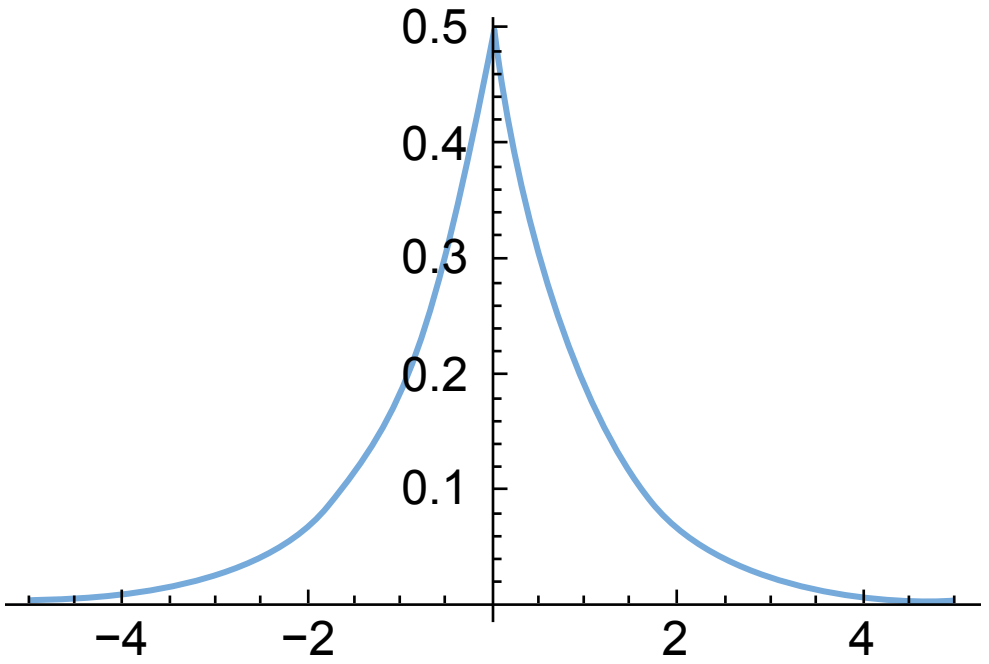
- That is, a symmetric, double-exponential distribution

$$z \sim Lap(b):$$

$$E[|z|] = b$$

$$\Pr[|z| \geq tb] = e^{-t}$$

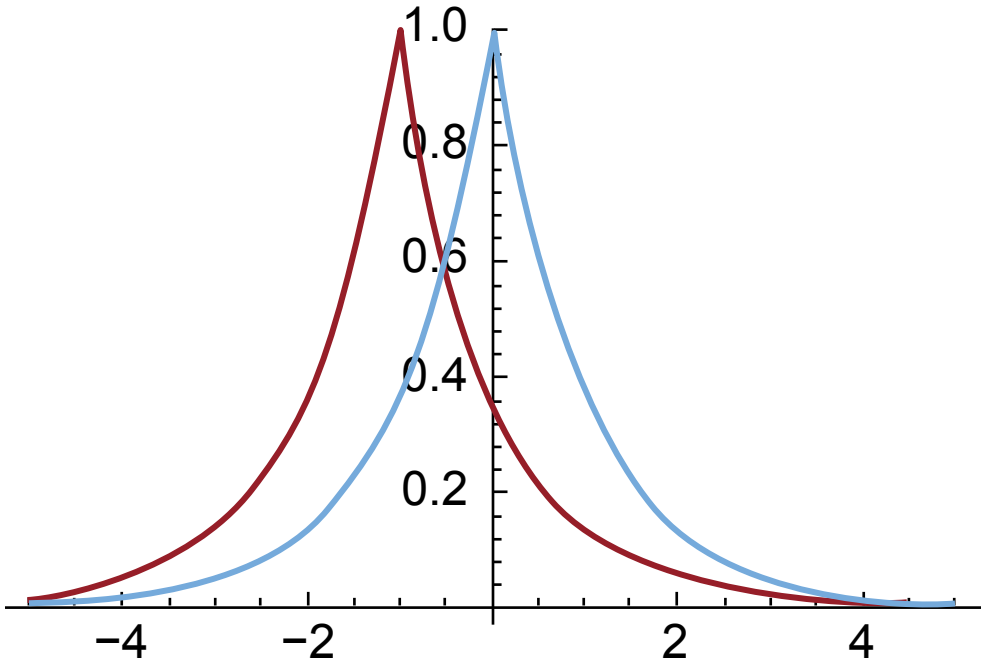
$$stdev(Lap(b)) = b\sqrt{2}$$



Laplace Mechanism for DP

Laplace($x, f^{np}: X^n \rightarrow R^k, \varepsilon$)

1. Let $\Delta_{f^{np}}$ be the ℓ_1, sensitivity of f^{np}
2. For $i = 1$ to k: Let $z_i \sim \text{Lap}(\frac{\Delta_{f^{np}}}{\varepsilon})$
3. Output $f^{np}(x) + (z_1, ..., z_k)$



Theorem: Laplace(.) satisfies (ϵ, 0) − DP.

Proof: See Aaron Roth's lecture notes.

Privacy-Accuracy Trade-Off

- The noise distribution depends on 1/epsilon and L1-sensitivity of the computation, **not** on the database or its size!
- This means that a DP computation based on the Laplace mechanism will be more accurate when sensitivity is small and epsilon is large.

Privacy-Accuracy Trade-Off

- The noise distribution depends on 1/epsilon and L1-sensitivity of the computation, **not** on the database or its size!
- This means that a DP computation based on the Laplace mechanism will be more accurate when sensitivity is small and epsilon is large.
- Example: “How many people in a room have brown eyes?” Sensitivity = **1**

$$z \sim Lap(\frac{1}{\varepsilon}): \quad E[|z|] = \frac{1}{\varepsilon} \quad \Pr[|z| \geq t\frac{1}{\varepsilon}] = e^{-t}$$

$$\xrightarrow{\varepsilon = 1}$$

$$E[|z|] = 1 \quad \Pr[|z| \geq t] = e^{-t} \quad \Pr[|z| \geq 7] < 0.1\%$$

- For a small value of the function (small group of people), +/−7 matters.
- But for a large count (e.g., 1,000), +/−7 doesn’t matter!
- So, the larger the database, the less the noise will impact accuracy.

Gaussian Mechanism

(Dwork, 2006)

Gaussian Mechanism for DP

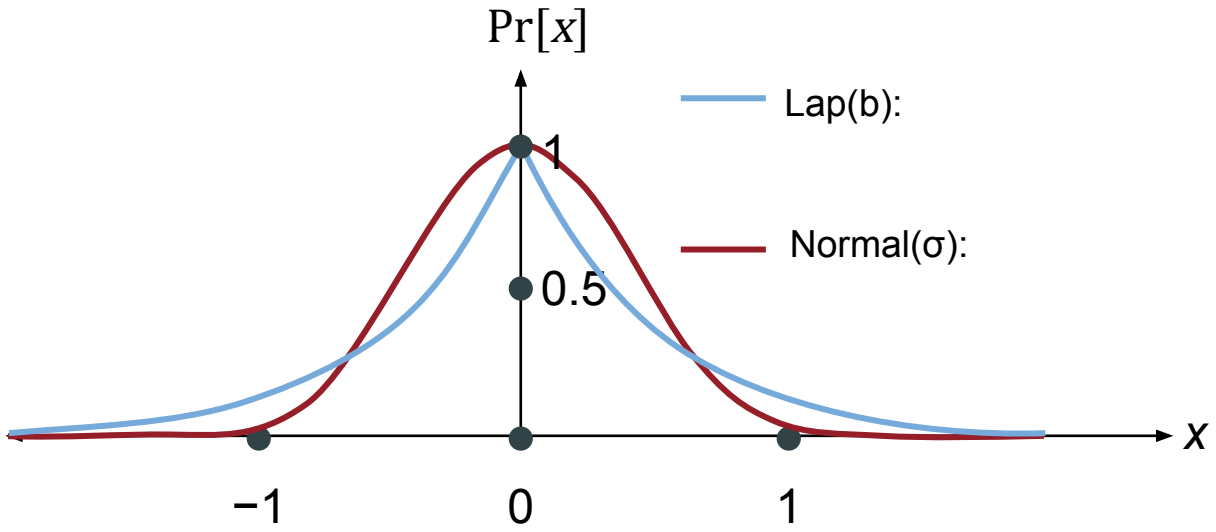
- Enforces **(ϵ, δ) − DP** by perturbing the output of a real-valued function with noise drawn from a normal (a.k.a. Gaussian) distribution with mean 0 and standard deviation dependent on the L2-sensitivity (denoted here as Δ_2) of the function:

$$\sigma = \frac{\Delta_2}{\varepsilon}\sqrt{2\ln\left(\frac{1.25}{\delta}\right)}$$

- L2-sensitivity is defined similarly to L1, but we take the maximum 2-norm (Euclidian distance) between *f*(*x*) and *f*(*x'*).

Laplace vs. Gaussian Mechanism

- The Laplace distribution (blue) has a better variance, but the tail of the Gaussian distribution (red) decreases faster.
- This is why Laplace gives pure DP but Gaussian needs the delta greater than 0 to account for its sharp cut of the tails.
- Thus, Laplace gives better semantic but you can get better utility from Gaussian.



When Laplace/Gaussian Don't Work

- What if we have a nonnumeric function?
  - "What's the most common eye color in the room?"
- What if the perturbed answer isn't "almost as good as" the exact answer?
  - "Which price would bring the most money from a set of buyers?"

Exponential mechanism
(McSherry, 2007)

- What if L1/L2-sensitivity is large?
  - "What's the median salary in a salary database?"

Smooth sensitivity
(Nissim, 2007)
(and other mechanisms)

Cited References (No Particular Order)

(Dwork, 2006) Dwork, McSherry, Nissim, & Smith. *Calibrating noise to sensitivity in private date analysis.* TCC, 2006.

(McSherry, 2007) McSherry & Talwar. *Mechanism design via differential privacy.* FOCS, 2007.

(Nissim, 2007) Nissim, Raskhodnikova, & Smith. *Smooth sensitivity and sampling in private data analysis.* STOC, 2007.

Questions

Determine L1-sensitivities of the following statistics:

- Number of families at zipcode 10027 and number of families in NYC (zipcode 10027 is in NYC).
  a. 1
  b. 2
  c. 3

$$\Delta_f = \max_{x,x':d(x,x')\leq 1} \| f(x) - f(x') \|_1$$

Questions

Determine L1-sensitivities of the following statistics:

- Counts of families per NYC zip code.
  a. 1
  b. 2
  c. number of zipcodes in NYC

$$\Delta_f = \max_{x,x':d(x,x')\leq 1} \| f(x) - f(x') \|_1$$

Questions

Determine L1-sensitivities of the following statistics:

- Average age of N people in a group (assume age is 0-120).
  a. 1
  b. 2
  c. 1/N
  d. 120/N
  e. N

$$\Delta_f = \max_{x,x':d(x,x')\leq 1} \| f(x) - f(x') \|_1$$

The End

Composite DP Algorithms

DP Algorithms

- There are some DP libraries that implement basic algorithms for statistical data analysis.
  - Google's statistics library: basic statistics (Wilson, 2020)
  - IBM's library: *k*-means, regression, PCA (Holohan, 2019)
  - TensorFlow privacy, Pytorch Opacus: stochastic gradient descent (SGD) (McMahan, 2018)
- We'll discuss some algorithms from Google's statistics library.
  - But best to read their code!

Bounded Sum

- Code
- Goal: compute sum over a set of values bounded in range [L,U]
  - L, U assumed to be public knowledge or obtained through the "approximate bounds" DP algorithm (also implemented in Google's lib)
- Method
  - Laplace mechanism with sensitivity max(|L|,|U|)
  - If L and U need to be computed privately too, then use part of the privacy budget for the approximate bounds algorithm, and the remainder for the Laplace mechanism (per the composition property of DP)

Bounded Mean

- Code
- Goal: compute mean over a set of values bounded in range [L,U]
- One option
  - Compute DP sum: S (with half the budget)
  - Compute DP count: N (with half the budget)
  - Return S/N, but sum has sensitivity $\max(|U|,|L|)$
- Observe
  - For fixed N, the mean has L1-sensitivity $1/N * \max(|U|,|L|)$
  - So, it's tempting to compute the mean and apply Laplace for that sensitivity
  - That assumes the count is public information but often it isn't

Bounded Mean (cont.)

- Google's implementation rewrites the average in terms relative to the middle of the interval [*L*,*U*] $\left(middle = \frac{L+U}{2}\right)$
- That enables calculating the sum of all input values with lower sensitivity than it would take if doing noisy sum/noisy count: $|U - L|/2$

$$average = \frac{\sum x_i}{N} = middle + \frac{\sum (x_i - middle)}{N}$$

Function 1, sensitivity: |U-L|/2

Function 2, sensitivity: 1

$$noisy\_average = middle + \frac{\sum x_i - N \cdot middle + Lap(|U - L|/\varepsilon)}{N + Lap(2/\varepsilon)}$$

Bounded Variance

- Code
- Goal: compute variance over a set of values bounded in $[L,U]$
- Method
  - Variance can be written as [Mean of squares − Mean squared].

$$\text{var}iance = \frac{\sum (x_i - \mu)^2}{n} = \frac{\sum x_i^2}{N} - \left(\frac{\sum x_i}{N}\right)^2$$

  - DP variance can therefore be computed by applying the preceding DP **bounded mean** algorithm twice, each time with half of the budget.

Approximate Bounds

- Code
- Goal: establish approximate [*L,U*] range over a set of values
- Method (described for non-negatives)
  - Organize data into **logarithmic histogram bins** (e.g., [0,1], (1,2], (2,4], (4,8],…)
  - Each bin keeps a **DP count** of the number of values in its range. Bin *i* holds the DP count of values in range $\left(scale \cdot base^{i-1}, scale \cdot base^{i}\right]$.
  - Given a confidence parameter *c*, we choose a **threshold *t*** after which to declare a bin as non-empty with probability >=*c*. The formula for *t* is:

$$t = \frac{1}{\varepsilon}\log\left(1 - c^{\frac{1}{base-1}}\right) \quad \text{(Wilson, 2020)}$$

  - ***L*** = leftmost bin with DP count greater than *t*. ***U*** = rightmost bin with DP count greater than *t*

Approximate Bounds (cont.)

- Example: base = 2, num_bins = 4, inputs = {0, 0, 0, 0, 1, 3, 7, 8, 8, 8}
  - The bins and (non-DP) counts are [0, 1]: 5 ; (1, 2]: 0 ; (2, 4]: 1 ; (4, 8]: 4
  - If success_probability = .9 and epsilon = 1, we get threshold $t = 3.5$
  - Since the count of bin [4, 8] > t, we would return max := 2^3 = 8
  - Since the count of bin [0, 1] > t, we would return min := 0
  - With DP counts, the procedure gives approximate values of course
- The parameters of this scheme—num_bins, base—are not trivial to set without some external knowledge of the rough range you want to capture

Cited References

(Holohan, 2019) Naoise Holohan, Stefano Braghin, Pól Mac Aonghusa, & Killian Levacher. (2019). *Diffprivlib: The IBM differential privacy library.* arXiv:1907.02444.

(McMahan, 2018) H. Brendan McMahan, Galen Andrew, Ulfar Erlingsson, Steve Chien, Ilya Mironov, Nicolas Papernot, & Peter Kairouz. (2018). *A general approach to adding differential privacy to iterative training procedures.* arXiv:1812.06210.

(Wilson, 2020) Royce J. Wilson, Celia Yuxin Zhang, William Lam, Damien Desfontaines, Daniel Simmons-Marengo, & Bryant Gipson. (2020). *Differentially private SQL with bounded user contribution.* PETs.

Task for Home

- Inspect the code in Google's statistics library corresponding to:
  - Bounded sum
  - Bounded mean
  - Bounded variance
  - Approximate bounds
  - And any other function you wish.

The End

DP SGD Algorithm

Stochastic Gradient Descent (SGD)

- DP version was described in (Abadi, 2016) and implemented in several libraries, including Tensorflow Privacy and Pytorch Opacus
- Without privacy, SGD is:
  - Repeated
    - Sample a batch from input dataset
    - Calculating $\nabla\theta$ with respect to the loss function
    - $\theta := \theta + \nabla\theta$
- With privacy, at each iteration you clip and noise the gradients $\nabla\theta$ using the Gaussian mechanism

Differentially Private SGD (Outline)

Input: Examples $\{x_1, \ldots, x_N\}$, loss function $\mathcal{L}(\theta) = \frac{1}{N}\sum_i \mathcal{L}(\theta, xi)$. Parameters: learning rate η_t, noise scale σ, batch size L, gradient norm bound C

Initialize θ_0 randomly

for $t \in [T]$ **do**

Take a random sample L_t with sampling probability L/N

Compute gradient

For each $i \in L_t$, compute $\mathbf{g}_t(x_i) \leftarrow \nabla_{\theta_t} \mathcal{L}(\theta_t, xi)$

Clip gradient

$$\bar{\mathbf{g}}_t(x_i) \leftarrow \mathbf{g}_t(x_i) / \max\left(1, \frac{\|\mathbf{g}_t(x_i)\|_2}{C}\right)$$

Add noise

$$\tilde{\mathbf{g}}_t \leftarrow \frac{1}{L}\left(\sum_i \bar{\mathbf{g}}_t(x_i) + \mathcal{N}(0, \sigma^2 C^2 \mathbf{I})\right)$$

Descent

$$\theta_{t+1} \leftarrow \theta_t - \eta_t \tilde{\mathbf{g}}_t$$

Output θ_T and compute the overall privacy cost (ε, δ) using a privacy accounting method

Differentially Private SGD (Outline)

Input: Examples $\{x_1, \ldots, x_N\}$, loss function $\mathcal{L}(\theta) = \frac{1}{N}\sum_i \mathcal{L}(\theta, xi)$. Parameters: learning rate η_t, noise scale σ, batch size L, gradient norm bound C

Initialize θ_0 randomly

for $t \in [T]$ **do**

Take a random sample L_t with sampling probability L/N

Compute gradient

For each $i \in L_t$, compute $\mathbf{g}_t(x_i) \leftarrow \nabla_{\theta_t} \mathcal{L}(\theta_t, xi)$

Clip gradient

$\bar{\mathbf{g}}_t(x_i) \leftarrow \mathbf{g}_t(x_i) / \max\left(1, \frac{\|\mathbf{g}_t(x_i)\|_2}{C}\right)$

Add noise

$\tilde{\mathbf{g}}_t \leftarrow \frac{1}{L}\left(\sum_i \bar{\mathbf{g}}_t(x_i) + \mathcal{N}(0, \sigma^2 C^2 \mathbf{I})\right)$

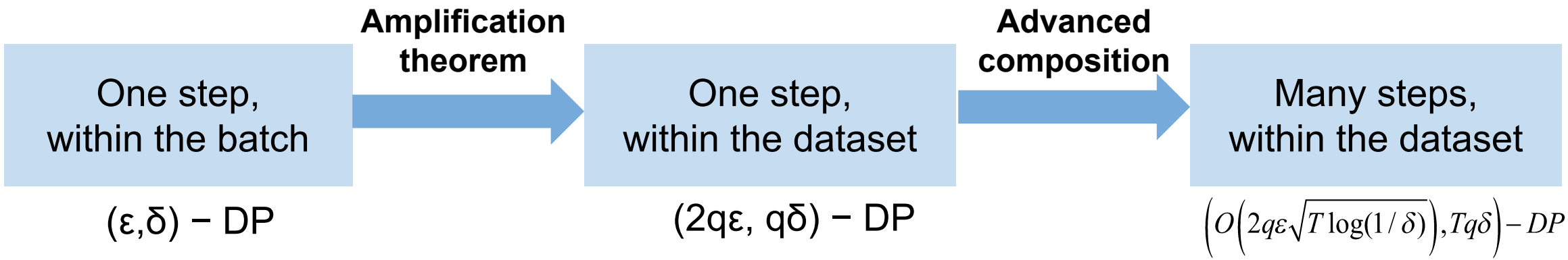
Descent

$\theta_{t+1} \leftarrow \theta_t - \eta_t \tilde{\mathbf{g}}_t$

Output θ_T and compute the overall privacy cost (ε, δ) using a privacy accounting method

Privacy Analysis

- The preceding algorithm receives as arguments a noise scale factor σ, gradient norm C, batch size L, and number of iterations T.
- Because each step uses the Gaussian mechanism, the gradient at each step is (ε,δ) − DP with respect to the batch. ε,δ are determined from σ.
- Question: What's the DP guarantee after many steps for the gradient with regard to the dataset? Answer: Apply the **amplification theorem** and **composition**.

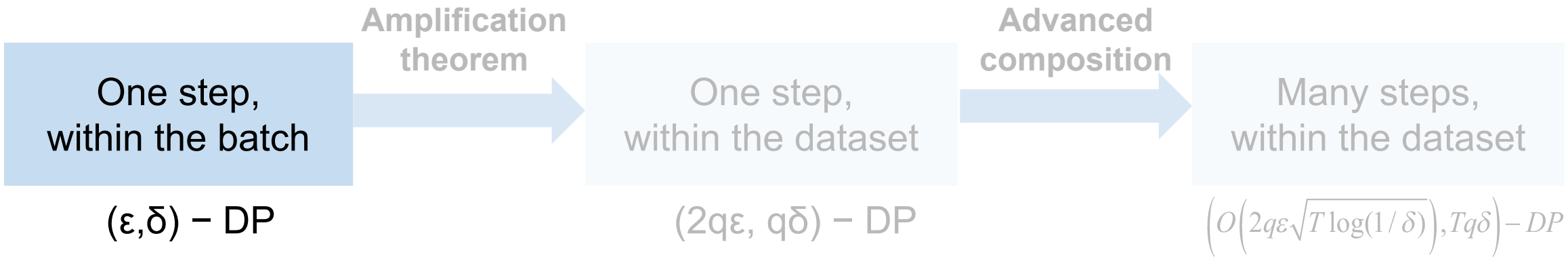


Determining ε,δ From σ

- To make a real-valued function DP with the Gaussian mechanism, we add noise from a normal distribution with standard deviation shown to the right.
- In the preceding algorithm, we added noise with standard deviation σC, where $C = \Delta_2$.
- Fixing δ to something reasonable, we can now compute ε based on σ, as shown on the right.

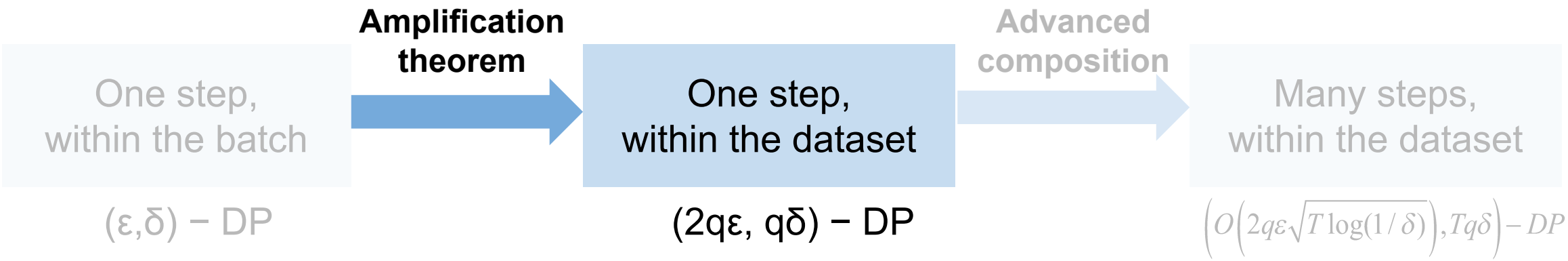
$$\frac{\Delta_2}{\varepsilon}\sqrt{2\ln\left(\frac{1.25}{\delta}\right)}$$

$$\varepsilon = \frac{1}{\sigma}\sqrt{2\ln\left(\frac{1.25}{\delta}\right)}$$



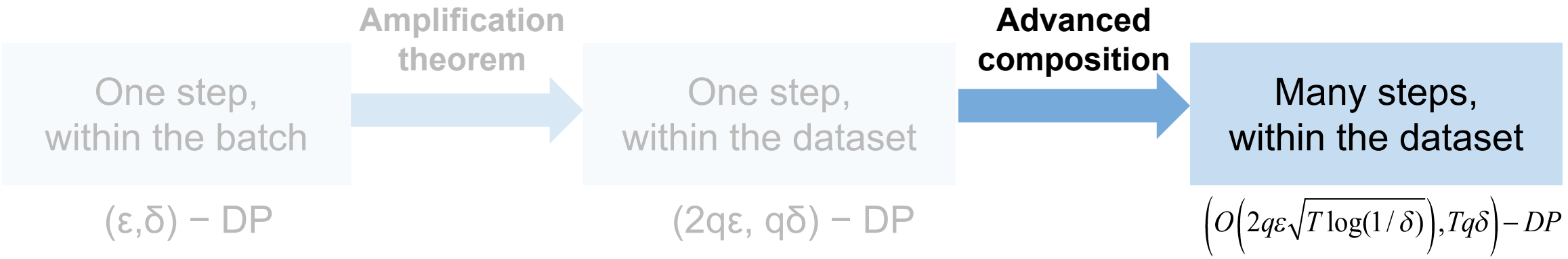
Amplification Theorem

- N: data size; L: size of each batch
- Let $q = L/N$
- **Amplification theorem:** if gradient is (ε,δ) − DP within the batch, then it is (2qε, qδ) − DP within the dataset



Advanced Composition

- Applying the same (ε,δ) − DP algorithm *T* times will give an $\left(O\left(\varepsilon\sqrt{T\log(1/\delta)}\right), T\delta\right)$ − DP algorithm.



Cited References

(Abadi, 2016) Martín Abadi, Andy Chu, Ian Goodfellow, H. Brendan McMahan, Ilya Mironov, Kunal Talwar, & Li Zhang. (2016). *Deep learning with differential privacy.* CCS.

The End

Demo: DP Neural Network Training

Demo

- Demo link (accessible to Columbia Lionmail accounts)

Homework 2 Overview

(CA walks through HW2 notebook, posted on courseworks)

This section: not for midterm!
Its purpose: broaden your perspective on privacy and its connections to other properties

Broader View on DP

- definition variations
- units of protection
- other mechanisms
- broader connections to other properties of statistical analyses

Broader View on DP

- **definition variations**
- units of protection
- other mechanisms
- broader connections to other properties of statistical analyses

Why *This* Definition?

A randomized computation $f : X \to Y$ is ε-DP if
$\forall x, x' \in X . d(x, x') \leq 1, \forall S \subseteq Y :$

$$\Pr(f(x) \in S) \leq e^{\varepsilon} \Pr(f(x') \in S).$$

- DP stipulates that the **distance** between the probability distributions over the outputs shall be small when the **distance** between the input datasets is small.
- But why those particular choices of **distance functions**: hamming for inputs, multiplicative measure of distance between output probability distributions (with optional additive factor)?

One (Bad!) Alternative

- What if we changed the multiplicative measure to statistical distance:

A randomized computation $f : X \to Y$ is "δ-alternative-DP" if $\forall x, x' \in X . d(x, x') \leq 1, \forall S \subseteq Y$:

$$\Pr(f(x) \in S) \leq \Pr(f(x') \in S) + \delta.$$

- **Not a well-behaved definition!**
  - Depending on delta, it either does not permit useful computations or permits mechanisms with poor privacy.

- When $\delta \leq 1/2n$, with proba ½ we get an answer independent of the dataset.
- When $\delta \geq 1/2n$, the mechanism "with proba ½ output a random row of the dataset" satisfies the definition. This semantic, we claimed, isn't acceptable privacy.

One (Bad!) Alternative

- What if we changed the multiplicative measure to statistical distance:

A randomized computation $f : X \to Y$ is "δ-alternative-DP" if $\forall x, x' \in X . d(x, x') \leq 1, \forall S \subseteq Y :$

$$\Pr(f(x) \in S) \leq \Pr(f(x') \in S) + \delta.$$

- **Not a well-behaved definition!**
  - Depending on delta, it either does not permit useful computations or permits mechanisms with poor privacy.

- On the other hand, (ε,δ)-DP is a well-behaved and very popular definition!
  - Are there more?

Why Need Other Definitions?

- Multiple definitions of privacy have been introduced.
- Some introduce different distance functions between the output distributions.
- Primary reason is to be able to analyze certain randomness distributions better (with tighter bounds):
  - E.g., Gaussian mechanism cannot be analyzed with epsilon-DP, it needs a delta>0.
- Another reason is to analyze composition more tightly.
- Example: advanced composition with (epsilon, delta)-DP.

Advanced Composition with (ε,δ)-DP

- **Basic composition** (akin to pure DP):

$$\text{If } \mathrm{f}_i, i = 1..k \text{ are } (\varepsilon,\delta)\text{-DP, then } \mathrm{x} \mapsto (\mathrm{f}_1(x),...,\mathrm{f}_k(x)) \text{ is}$$

$$(k\varepsilon, k\delta) - \text{DP}.$$

- **Advanced composition** (gives only (ε,δ>0)-DP, whether f_i are pure or not pure DP):

$$\text{If } \mathrm{f}_i, i = 1...k \text{ are } (\varepsilon,\delta)\text{-DP, then } \mathrm{x} \mapsto (\mathrm{f}_1(x),...,\mathrm{f}_k(x)) \text{ is}$$

$$(\varepsilon\sqrt{2k\log(1/\delta')} + k\varepsilon(e^{\varepsilon} - 1), k\delta + \delta') - \text{DP for any } \delta' > 0.$$

When Is Advanced Better than Basic?

- Wlog assume pure DP functions f_i (δ=0) and fix δ'= 10^{-6}.
- Advanced composition is better than basic composition when:

$$\varepsilon\sqrt{2k\log(1/\delta')}+k\varepsilon(e^{\varepsilon}-1)<k\varepsilon \Leftrightarrow k>2\log(1/\delta')/(e^{\varepsilon}-1)^2$$

- Another good way to look at this is the **global privacy budgeting** perspective, of which you'll hear in some papers we'll read in the second part of course.

Global Privacy Budgeting

- Think of the parameters (ε_g, δ_g) of the computation $x \mapsto (f_1(x),...,f_k(x))$ as fixed. They represent the *global guarantee* you want to enforce across all computations you do on the dataset, k computations in total.
- Then, asymptotically as k grows, basic composition requires allocating **ε~O(1/k)** for each computation f_i, while advanced composition requires **ε~O(1/sqrt(k))** foreach computation f_i.
- This scalability benefit becomes critical when you do a lot of computations, for example as part of a big iterative process like stochastic gradient descent (SGD).

Example

- Suppose the number of functions f_i is fairly large (e.g., k=50K) and the global privacy parameters are not too big (e.g., ε_g=1 and δ_g=10^{-6}). And let's assume δ=0 (f_i *are* all pure DP*)*, hence δ'=10^{-6}.
- For basic composition, we need ε=ε_g/k=2e-5 for each f_i to maintain ε_g=1.
- For advanced composition, we can calculate that ε=7e-4 suffices for each f_i in order to stay within ε_g=1.
- Here, advanced composition increases the per-function allocation of the privacy budget by a **factor of 35** over basic composition. This means less noise for each computation, so better accuracy.

Other Common Definitions

- **Renyi-DP**: Distance between output distributions is defined in terms of Renyi divergence.
  - Analyzes composition with simpler arithmetic and eve a bit tighter than (epsilon, delta)-DP.
- **Zero-concentrated DP**: Similar to Renyi-DP, used in Census 2020.
- **Gaussian DP**

Renyi-DP

- Define:

$$\text{PrivacyLoss}(y, \mathcal{D}, \mathcal{D}') = \log\left(\frac{P(\mathcal{A}(\mathcal{D}) = y)}{P(\mathcal{A}(\mathcal{D}') = y)}\right).$$

- Then, epsilon-DP is equivalent to \forall y, D, D':

  | PrivacyLoss(y,D,D') | <= epsilon

- Renyi DP defines privacy loss in terms of renyi divergence:

$$\text{PrivacyLoss}_\alpha(\mathcal{D}, \mathcal{D}') = \frac{1}{\alpha - 1} \log \mathop{\mathbb{E}}_{y \sim \mathcal{A}(\mathcal{D})} \left(\frac{P(\mathcal{A}(\mathcal{D}) = y)}{P(\mathcal{A}(\mathcal{D}') = y)}\right)^\alpha.$$

- (alpha, epsilon)-RDP is: \forall D, D':

  | PrivacyLoss_alpha(D,D') | <= epsilon

Renyi-DP Properties

- Very simple arithmetic for composition (additive in epsilon parameter), but in such a way that it scales with sqrt(k)!
  - So the benefit of (epsilon, delta)-DP but without the ugly math.
- Can always translate from Renyi DP to (epsilon, delta)-DP:

$$\varepsilon_{DP} = \varepsilon + \frac{\log(1/\delta)}{\alpha - 1}.$$

for <u>any</u> alpha>1!!

- Any alpha will work, so you can try a bunch and take the one that gives the best epsilon-DP (this means, the tightest bound)!

- For these reasons, some DP-SGD implementations internally compose steps with Renyi-DP and eventually translate the guarantee into the best (epsilon, delta)-DP guarantee they can arrive at.

Broader View on DP

- definition variations
- **units of protection**
- other mechanisms
- broader connections to other properties of statistical analyses

Another Direction of Variation: DB Distance

- d(D, D') <= 1: Hamming distance is standard and has interpretation of "adding or removing one entry or user."
- But L1 distance is also possible: "modifying one entry"
- But what is "one entry"? A user? A data item of a user?...

Units of Protection

- **User-level DP**: all data points corresponding to a user are “hidden” at once.
  - Need to bound user’s data contribution to the statistical analysis.
- **Event-level DP**: individual data point -- whatever that may be: a click, a location, a message… -- is “hidden” separately.
  - Much weaker, but easier to maintain without losing utility.
  - Over time, more data comes in in streaming settings.
  - But adversary can still learn user patterns from outputs.
- **User-time DP**: user-level guarantee on a time-window basis.
  - E.g., all users’ activities during a week are all “hidden” at once.
  - A.k.a., “resetting privacy budgets.”
  - Stronger semantic than event-level, more realistic to maintain compared to user-level DP.
  - Apple adopts this.

Broader View on DP

- definition variations
- units of protection
- **other mechanisms**
- broader connections to other properties of statistical analyses

When Laplace/Gaussian Don't Work

- What if we have a non-numeric function?
  - "What's most common eye color in the room?"
- What if the perturbed answer isn't "almost as good as" the exact answer?
  - "Which price would bring the most money from a set of buyers?"

Exponential Mechanism

- What if L1-sensitivity is large?
  - "What's the median salary in a salary database?"

Smooth Sensitivity (and other mechanisms)

Exponential Mechanism

(McSherry, 2007)

- A mechanism $M : X^n \rightarrow Y$ for some **abstract range Y**, e.g.:
  - Y={Red, Blue, Green, Brown, Hazel}
  - Y={$1.00, $1.01, $1.02, …}

- Paired with a non-private, real-valued, **quality scoring function**:

$$q : X^n \times Y \rightarrow R$$

(q(x,y) represents how good output y is for database x.)

Exponential Mechanism

(McSherry, 2007)

$$\text{Exponential}(x, Y, q: X^n \times Y \to R, \varepsilon):$$

1. Let Δ_q be the ℓ_1 – sensitivity of q.
2. Output $y \sim Y$ with probability:

$$\Pr[y] = \frac{\exp\left(\frac{\varepsilon \cdot q(x,y)}{2\Delta_q}\right)}{\sum_{y' \in Y} \exp\left(\frac{\varepsilon \cdot q(x,y')}{2\Delta_q}\right)}$$

Make high-quality outputs exponentially more likely at a rate that depends on:

1. the sensitivity of the quality score, and
2. the privacy parameter.

Exponential Mechanism

(McSherry, 2007)

Theorem: The Exponential mechanism is $(\epsilon, 0)$-DP.

Proof: See Aaron Roth's lecture notes:
https://www.cis.upenn.edu/~aaroth/courses/slides/Lecture3.pdf

When Does Exponential Make Sense?

When Does Exponential Make Sense?

Define: $\mathrm{OPT}_q(x) = \max_{y \in Y} q(x,y)$ $\quad Y_{OPT} = \{y \in Y : q(x,y) = \mathrm{OPT}_q(x)\}$

$$y^* = Exponential(x, Y, q, \varepsilon)$$

When Does Exponential Make Sense?

Define: $\mathrm{OPT}_q(x) = \max_{y \in Y} q(x,y)$ $\qquad Y_{OPT} = \{y \in Y : q(x,y) = \mathrm{OPT}_q(x)\}$

$$y^* = Exponential(x, Y, q, \varepsilon)$$

Theorem:

$$\Pr\left[q(y^*) \leq \mathrm{OPT}_q(x) - \frac{2\Delta_q}{\varepsilon}\left(\log\left(\frac{|Y|}{|Y_{OPT}|} \right) + t \right) \right] \leq e^{-t}$$

(Proofs in Aaron Roth's lecture notes: https://www.cis.upenn.edu/~aaroth/courses/slides/Lecture3.pdf)

When Does Exponential Make Sense?

Define: $\mathrm{OPT}_q(x) = \max_{y \in Y} q(x,y)$ $\quad Y_{OPT} = \{y \in Y : q(x,y) = \mathrm{OPT}_q(x)\}$

$y^* = Exponential(x, Y, q, \varepsilon)$

Theorem: $$\Pr\left[q(y^*) \le \mathrm{OPT}_q(x) - \frac{2\Delta_q}{\varepsilon}\left(\log\left(\frac{|Y|}{|Y_{OPT}|} \right) + t \right) \right] \le e^{-t}$$

Corollary: $$\mathrm{E}\left[q(y^*) \right] \ge OPT_q(x) - \frac{2\Delta_q}{\varepsilon}(\log(|Y|) + \log(OPT_q(x))) - 1$$

(Proofs in Aaron Roth's lecture notes: https://www.cis.upenn.edu/~aaroth/courses/slides/Lecture3.pdf)

When Does Exponential Make Sense?

- So if Y = {Red, Blue, Green, Brown, Hazel}, then we can answer "What is the most common eye color in this room?" with a color that is shared by at least $OPT - \frac{2}{\varepsilon}\left(\log(5)+3\right) > OPT - \frac{7.4}{\varepsilon}$ people, except with probability ≤ e^{-3} < .05.

- *Independent* of the number of people in the room.
- Hence, again **small relative error if # of people is large**.

Laplace vs. Exponential?

- The exponential mechanism is based on the vector:

$$\hat{q} = \left(q(x, y_1), q(x, y_2), \ldots, q(x, y_{|Y|}) \right)$$

- We could have computed each of these values DP using the **Laplace mechanism**, and then through post-processing, we could have chosen the exact argmax from this vector while still preserving DP.
- The question of whether Exponential vs. Laplace would do better in this case depends on the sensitivity of $\hat{q}$, which we denote $\Delta_{\hat{q}}$.
- In some cases (like the price example), $\Delta_{\hat{q}}$ could be as large as $|Y| \Delta_q$.
- In contrast, the exponential mechanism only depends on Δ_q.

When Neither Laplace nor Exponential Make Sense

- They both depend on the ℓ_1-sensitivity of the computation, which is a **worst-case, database-independent** notion of how much impact an entry could have on the computation's output.
- When that's big, what do we do? Example: "What's the median salary in a salary database?"
- For such cases, other mechanisms exist, which rely on sensitivity notions that are *more tuned to the database*.
- One such mechanism is smooth sensitivity.

Initial Idea

- **L1-sensitivity (a.k.a. global sensitivity):** $\Delta_f = \max_{x,x':d(x,x')\leq 1} \| f(x) - f(x') \|_1 .$

- For median, L1-sensitivity would be the **whole range of the output domain**, so for Laplace, you'd add noise proportional to the range. That's terrible!
- But I have a fixed database, couldn't I use noise proportional to the sensitivity of the computation w.r.t. *my database*? Maybe that's much lower!

- **Local sensitivity:** $Local\Delta_f(x) = \max_{x':d(x,x')\leq 1} \| f(x) - f(x') \|_1 .$
- Problem with using local sensitivity to calibrate the noise is that **local sensitivity can vary dramatically between neighboring datasets**, so the noise distribution could become the distinguisher between two neighboring datasets.

Example

- Consider the median over n points $\{x_1,..x_n\}$ in [0,1].
- If “my database” happens to have (n+1)/2 entries that are 0, and (n-1)/2 entries that are 1, then the median is 0. But if one entry changes from 0 to 1, the median becomes 1. So local sensitivity of median for “my database” is 1.
- But if “my database” happens to have *many* data points near the medium (e.g., two thirds of the entries are 0 and one third is 1), then no individual entry’s change can change the output. In this case, the local sensitivity of “my database” is 0.
- Could we hope to scale the noise by 0 in the latter case????!!!!
- Tempting, but no, for the reason mentioned before and illustrated even better in the next example…

Counter-Example

- Imagine a function whose value is 0 on all databases except for a very specific database, x, on which it is 10M.
- In that case, local sensitivity of a (any) database neighboring x is 0 while local sensitivity on x is 10M.
- If you happen to have the database x, then applying noise proportional to 10M **will leak a lot** about that entry that differentiates x from every other neighboring database.
- So the draw of noise will leak information that your database is very likely x.

Smooth Sensitivity

(Nissim, et.al, 2007)

- **Smooth sensitivity:** $Smooth\Delta_f^{\varepsilon}(x) = \max_{x' \in X^n}(Local\Delta_f(x') \cdot e^{-\varepsilon d(x,x')})$.

- Intuitively, we are smoothing out the local sensitivity around our database, x, so that it doesn't change much between neighboring datasets.
- **Theorem:** Adding noise from Cauchy Distribution scaled by $O(Smooth\Delta_f^{\varepsilon}(x)/\varepsilon)$ gives (ϵ, 0)-DP. This is called the **smooth sensitivity mechanism**.
- For median, as well as some graph statistics, there are ways to compute smooth sensitivity efficiently (Nissim, et.al, 2007).

Other Mechanisms

- When global sensitivity is too high and computing smooth sensitivity is hard, there are other methods people have proposed:
  - Propose-test-release
  - Privately bounding local sensitivity
  - Releasing stable values

- I recommend reading about these algorithms in (Vadhan, 2016).

Broader View on DP

- definition variations
- units of protection
- other mechanisms
- **broader connections to other properties of statistical analyses**

Broader Connections

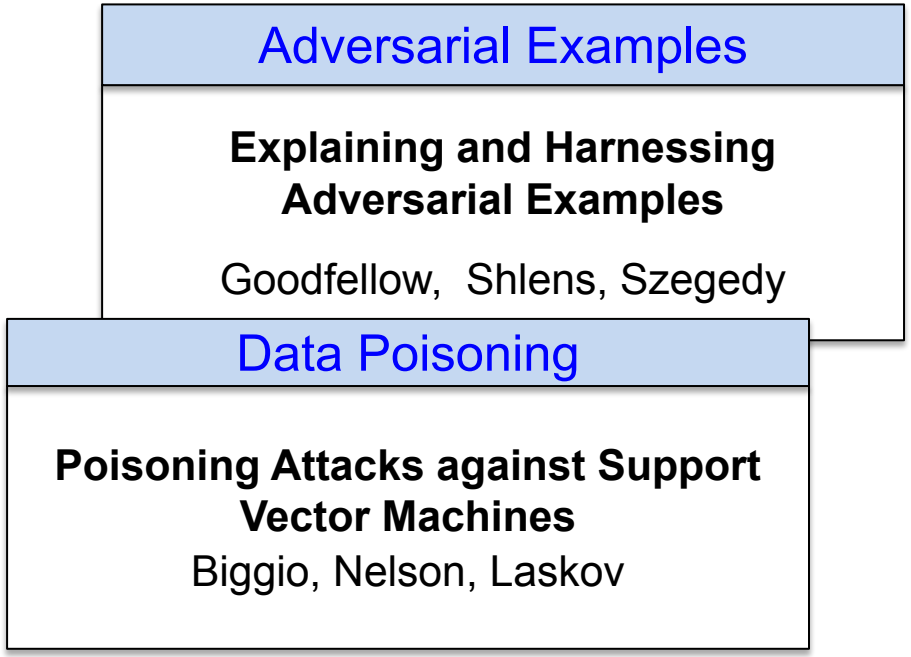
- Connections exist between privacy and other desirable properties of ML
- In theory, this could mean that technologies for one property could be useful for other properties
- Practical approaches to exploit these connections are still being researched

Myriad of ML Concerns

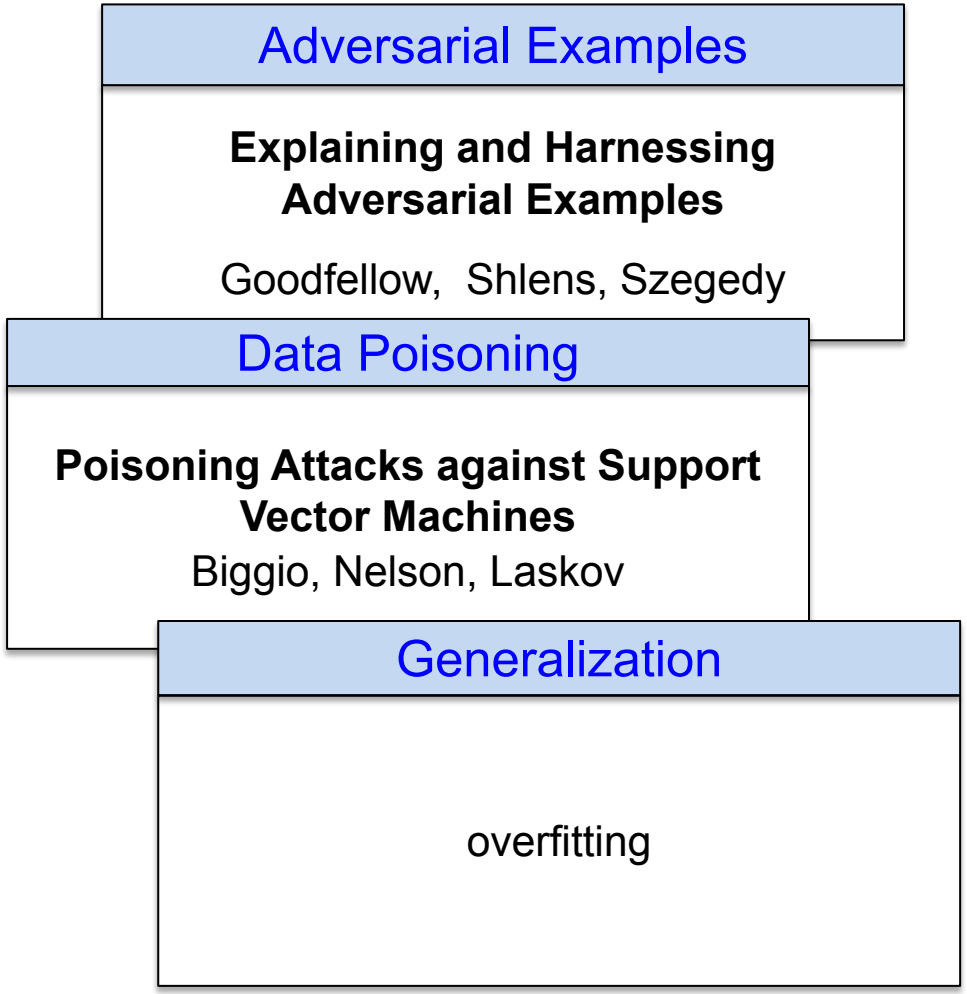
Myriad of ML Concerns

Adversarial Examples
Explaining and Harnessing Adversarial Examples Goodfellow, Shlens, Szegedy

Myriad of ML Concerns



Myriad of ML Concerns



Myriad of ML Concerns

Adversarial Examples

Explaining and Harnessing Adversarial Examples

Goodfellow, Shlens, Szegedy

Privacy Loss

The Secret Sharer: Evaluating and Testing Unintended Memorization in Neural Networks

Carlini, Liu, Erlingsson, Kos, Song

Data Poisoning

Poisoning Attacks against Support Vector Machines

Biggio, Nelson, Laskov

Generalization

overfitting

Myriad of ML Concerns

Adversarial Examples

Explaining and Harnessing Adversarial Examples

Goodfellow, Shlens, Szegedy

Privacy Loss

The Secret Sharer: Evaluating and Testing Unintended Memorization in Neural Networks

Carlini, Liu, Erlingsson, Kos, Song

Data Poisoning

Poisoning Attacks against Support Vector Machines

Biggio, Nelson, Laskov

Bias, Discrimination

Man is to Computer Programmer as Woman is to Homemaker? Debiasing Word Embeddings

Bolukbasi, Chang, Zou, Saligrama, Kalai

Generalization

overfitting

Myriad of ML Concerns

Adversarial Examples

Explaining and Harnessing Adversarial Examples

Goodfellow, Shlens, Szegedy

Privacy Loss

The Secret Sharer: Evaluating and Testing Unintended Memorization in Neural Networks

Carlini, Liu, Erlingsson, Kos, Song

Data Poisoning

Poisoning Attacks against Support Vector Machines

Biggio, Nelson, Laskov

Bias, Discrimination

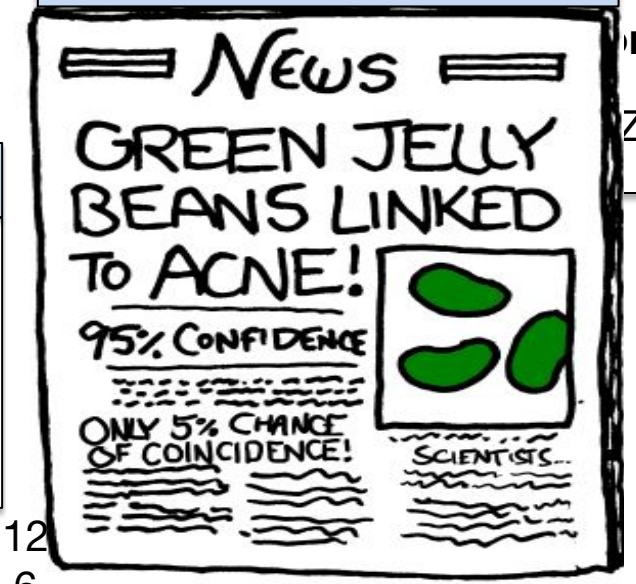
ter Programmer as Homemaker?
rd Embeddings

Zou, Saligrama, Kalai

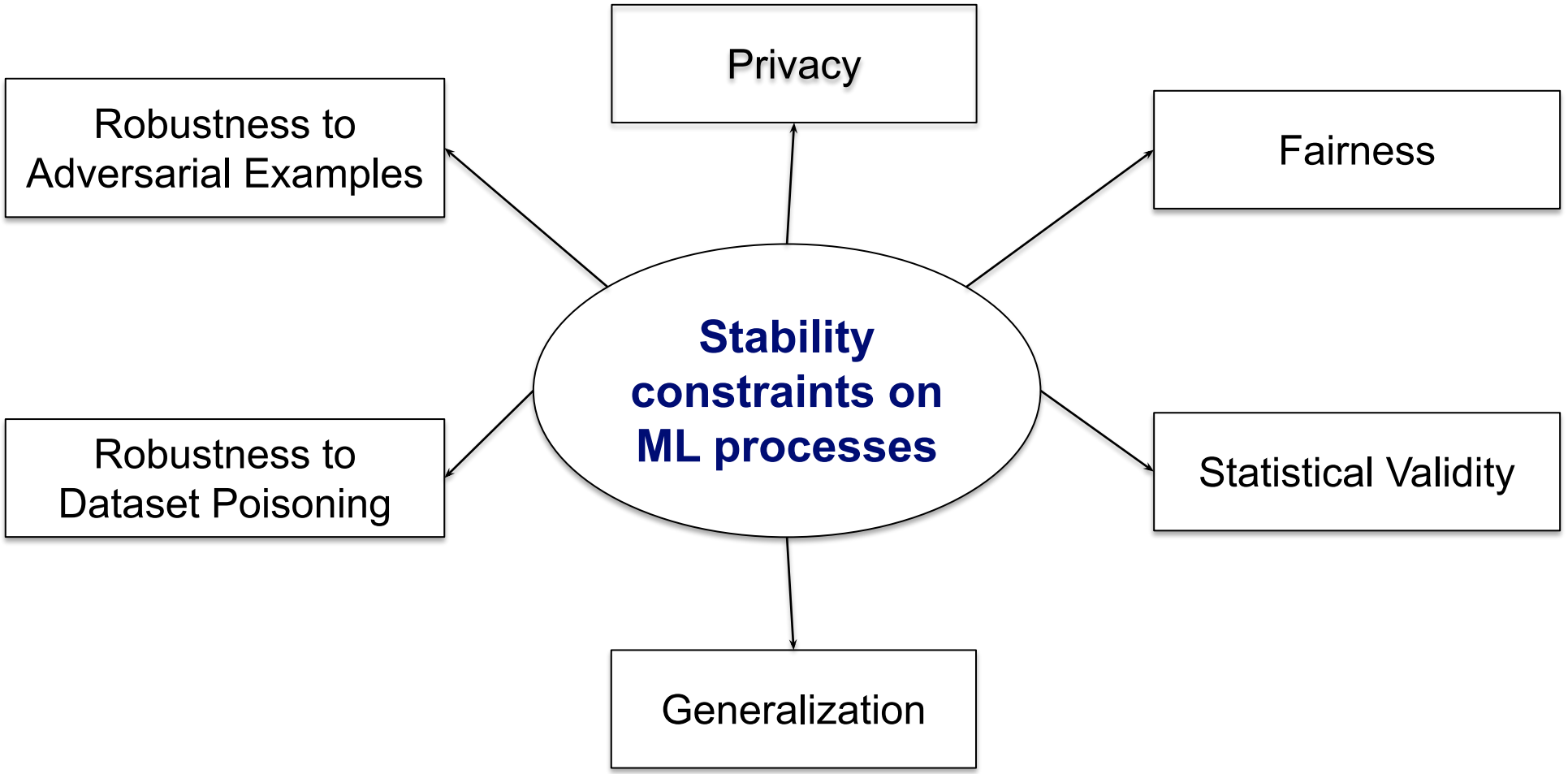
Generalization

overfitting

False Discoveries



Many Concerns Are Related



[Hardt-16]

Example: DP Improves More than Privacy

- DP is a **strong stability constraint** on computations running on datasets: it requires that no single data point in an input dataset has significant influence over the output

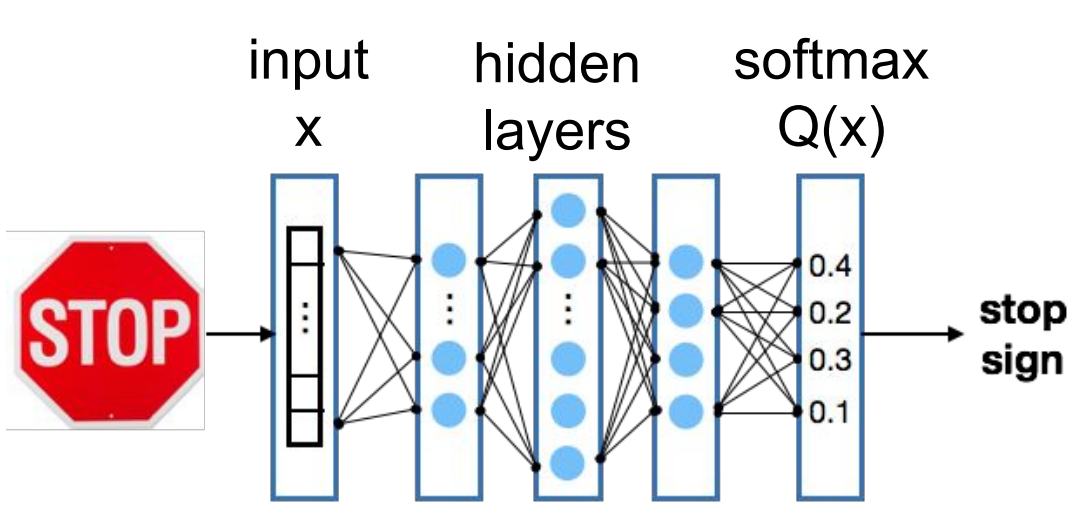
- It has been been shown to improve a variety of desirable ML properties beyond privacy, e.g.:
  - DP for Adversarial Robustness (Lecuyer+19)
  - DP for Generalization (Hardt-16, Bassily+16)
  - DP for Fairness (Dwork+13)
  - DP for Statistical Validity (Dwork+15)

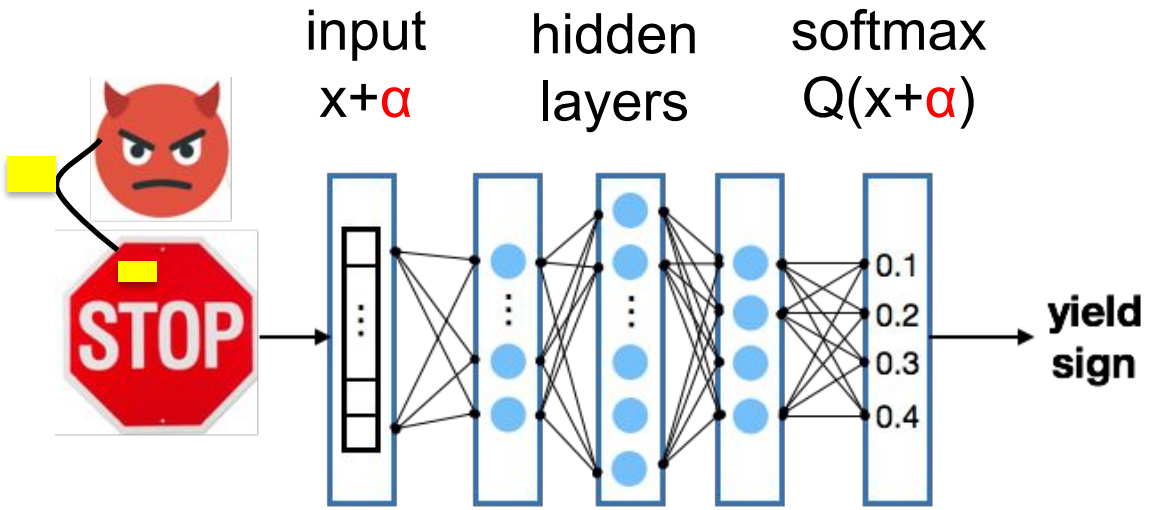
DP for Adversarial Robustness

(Lecuyer+19)

Adversarial Examples

- Adversary finds a tiny perturbation to a correctly classified input that causes misclassification



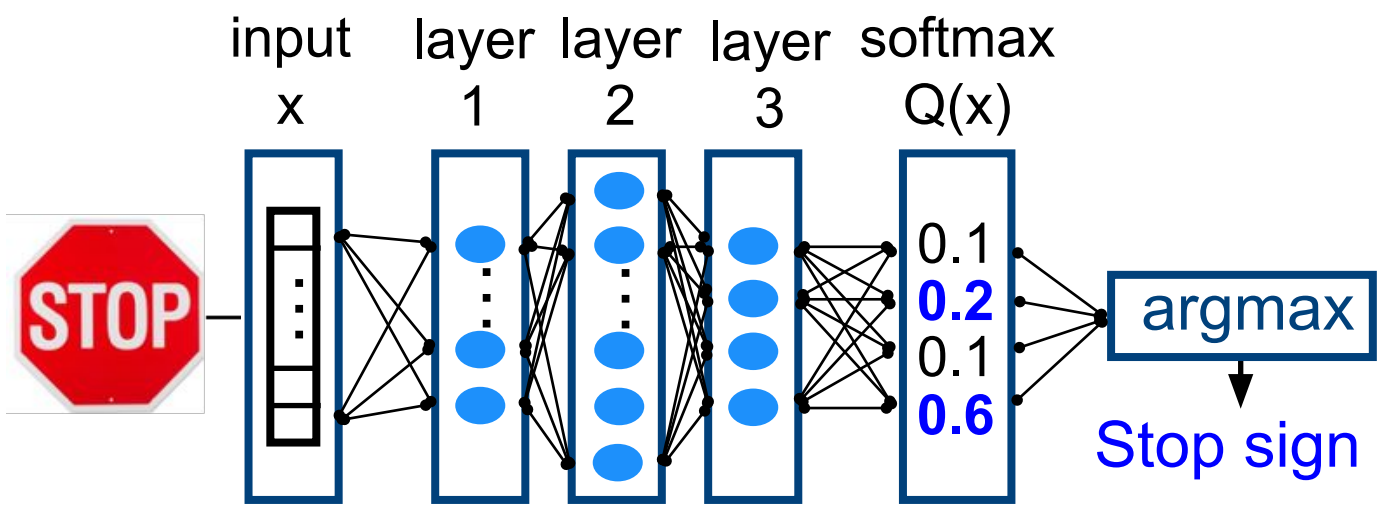


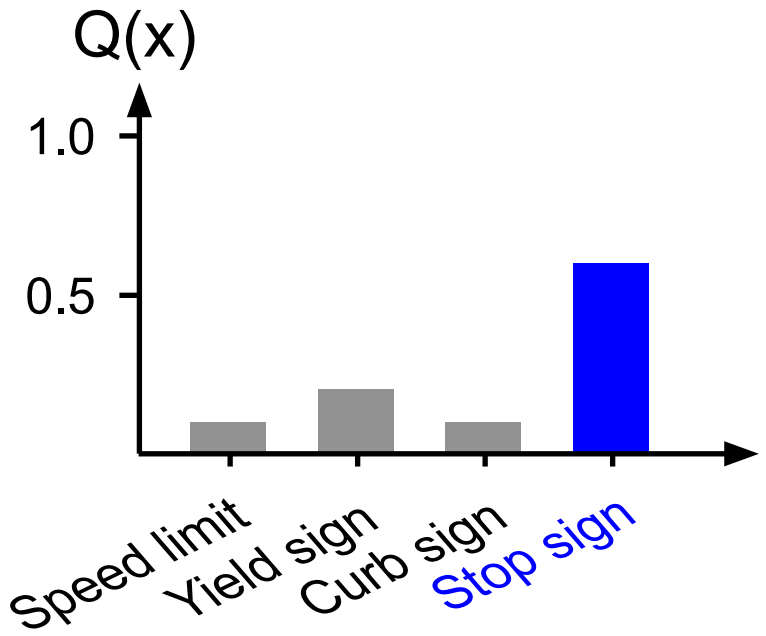
DP for Adversarial Examples

- Problem: small input changes create large score changes
- Approach: **make prediction function DP**

DP for Adversarial Examples

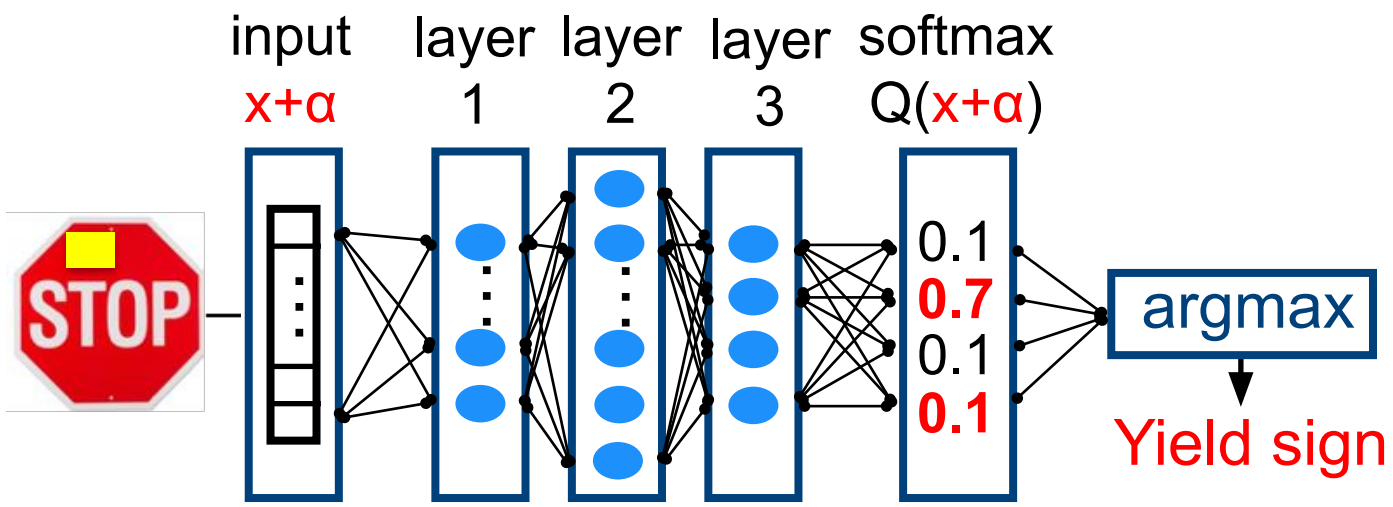
- Problem: small input changes create large score changes
- Approach: **make prediction function DP**

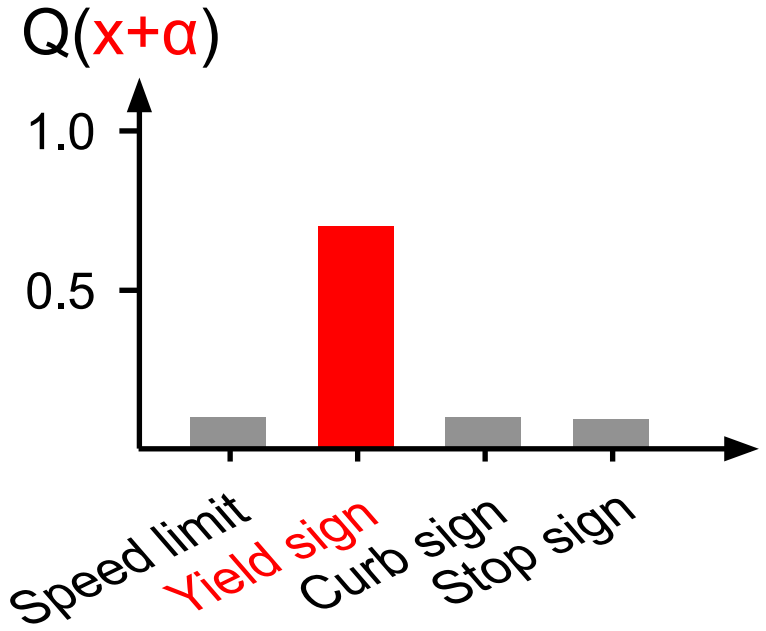




DP for Adversarial Examples

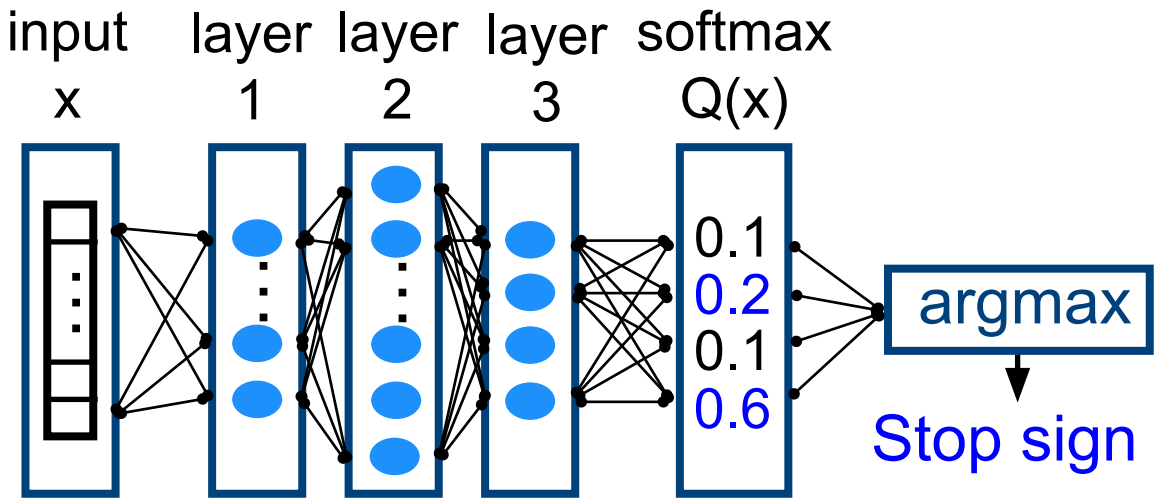
- Problem: small input changes create large score changes
- Approach: **make prediction function DP**





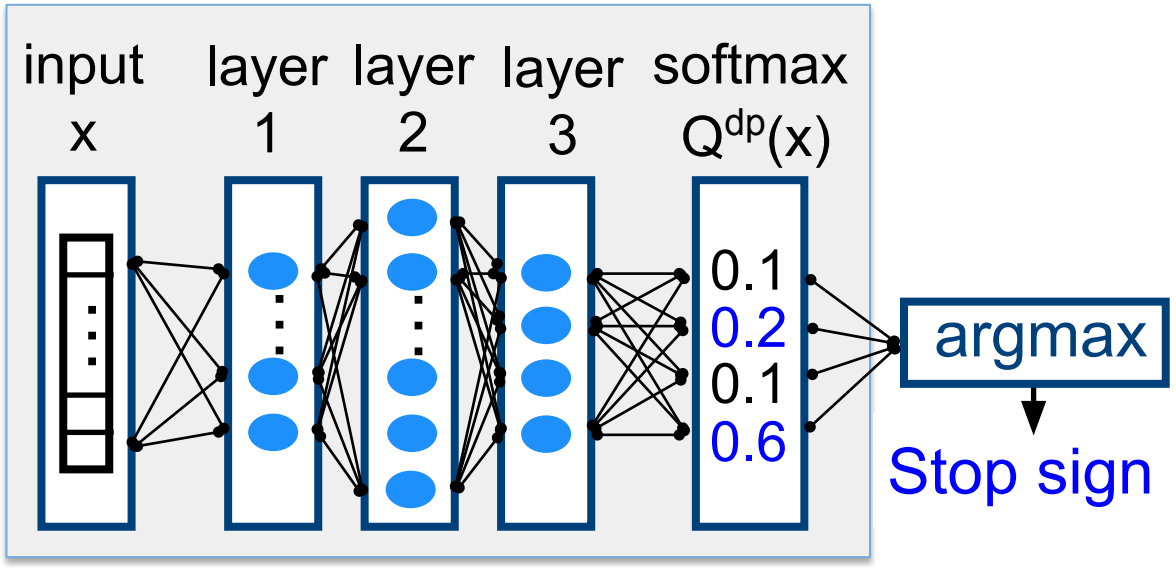
How It Works

1. Randomize prediction function to make it DP
2. Use expected scores to choose argmax
3. Use DP's stability bounds on expected scores to certify prediction on x



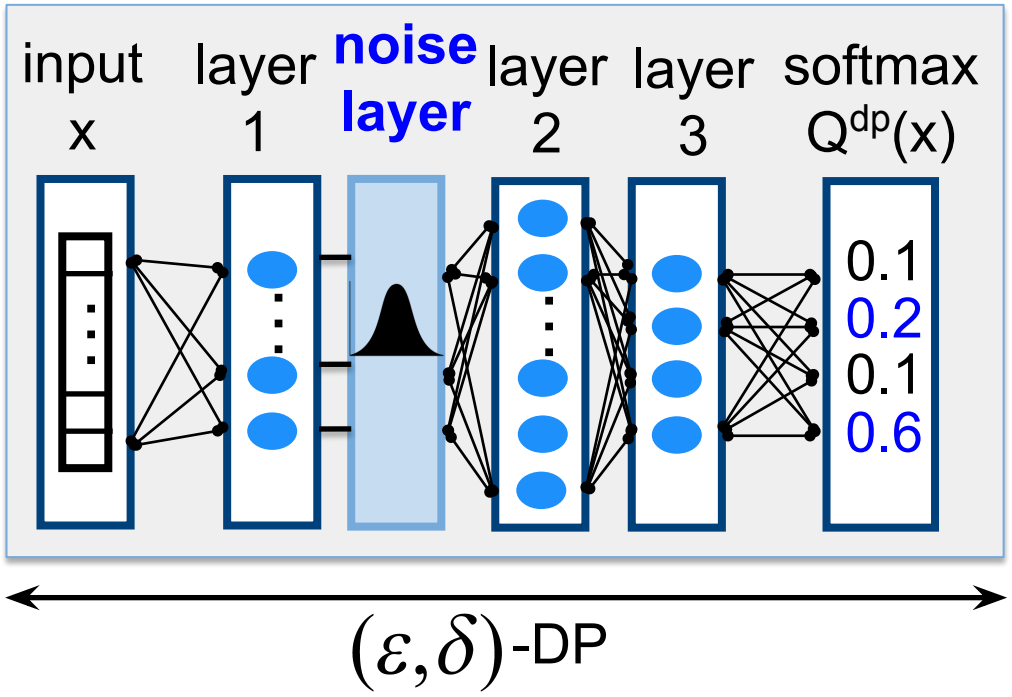
How It Works

1. Randomize prediction function to make it DP
2. Use expected scores to choose argmax
3. Use DP's stability bounds on expected scores to certify prediction on x



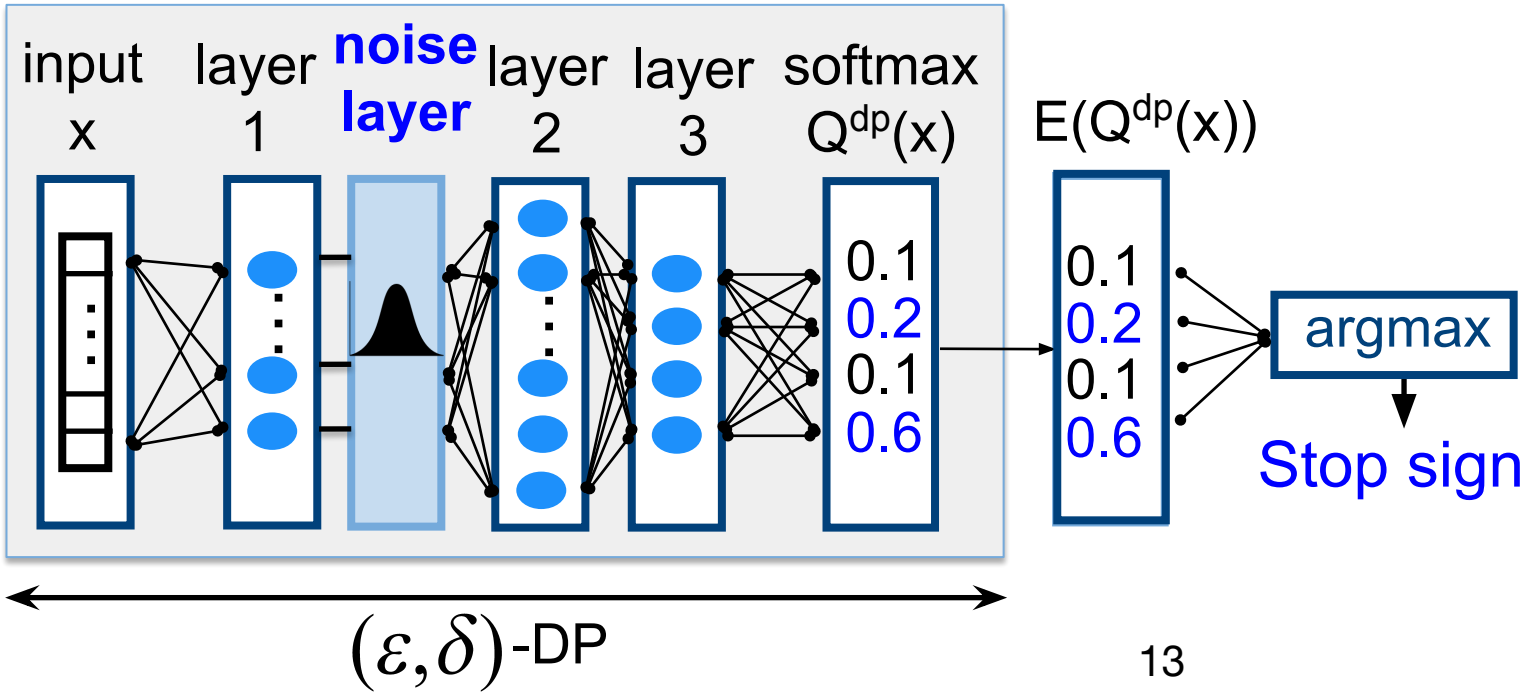
How It Works

1. Randomize prediction function to make it DP
2. Use expected scores to choose argmax
3. Use DP's stability bounds on expected scores to certify prediction on x



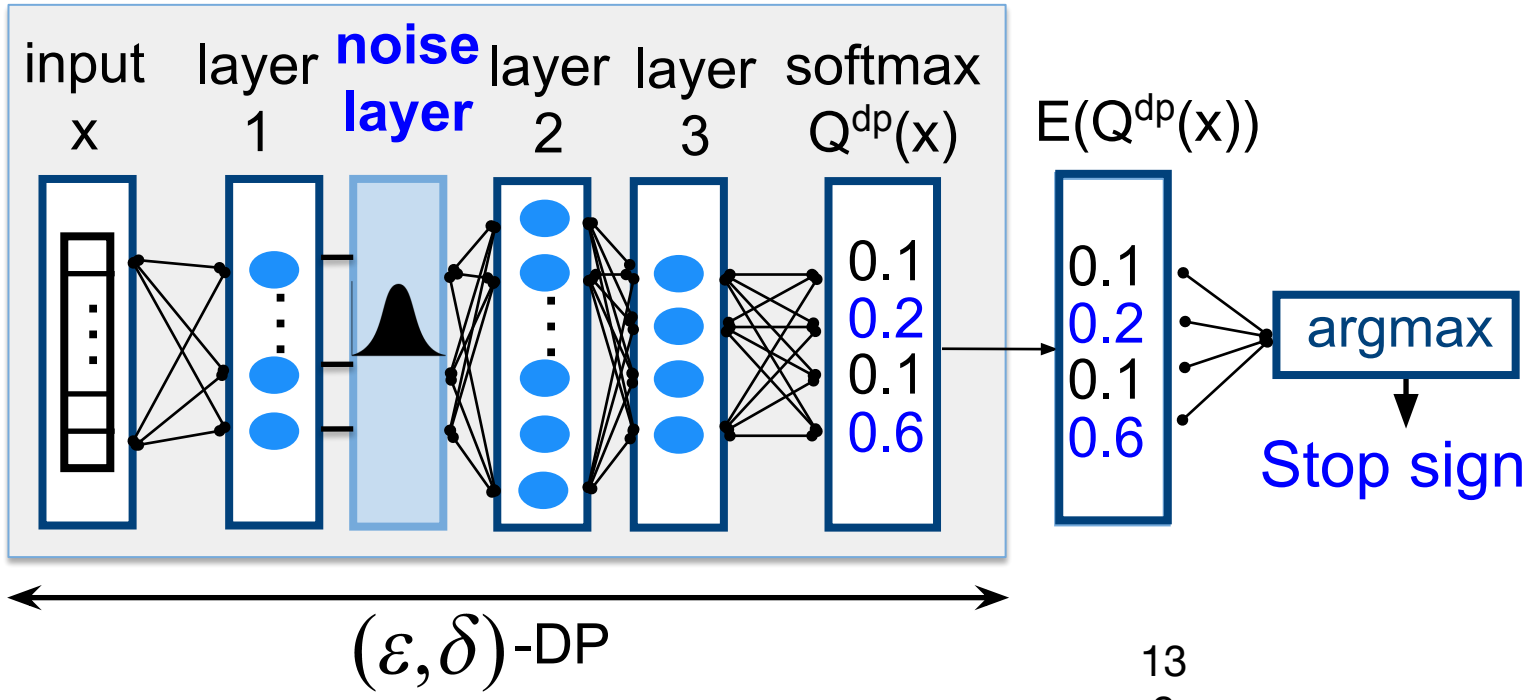
How It Works

1. Randomize prediction function to make it DP
2. Use expected scores to choose argmax
3. Use DP's stability bounds on expected scores to certify prediction on x



How It Works

1. Randomize prediction function to make it DP
2. Use expected scores to choose argmax
3. Use DP's stability bounds on expected scores to certify prediction on x



DP's stability bounds on expected scores:

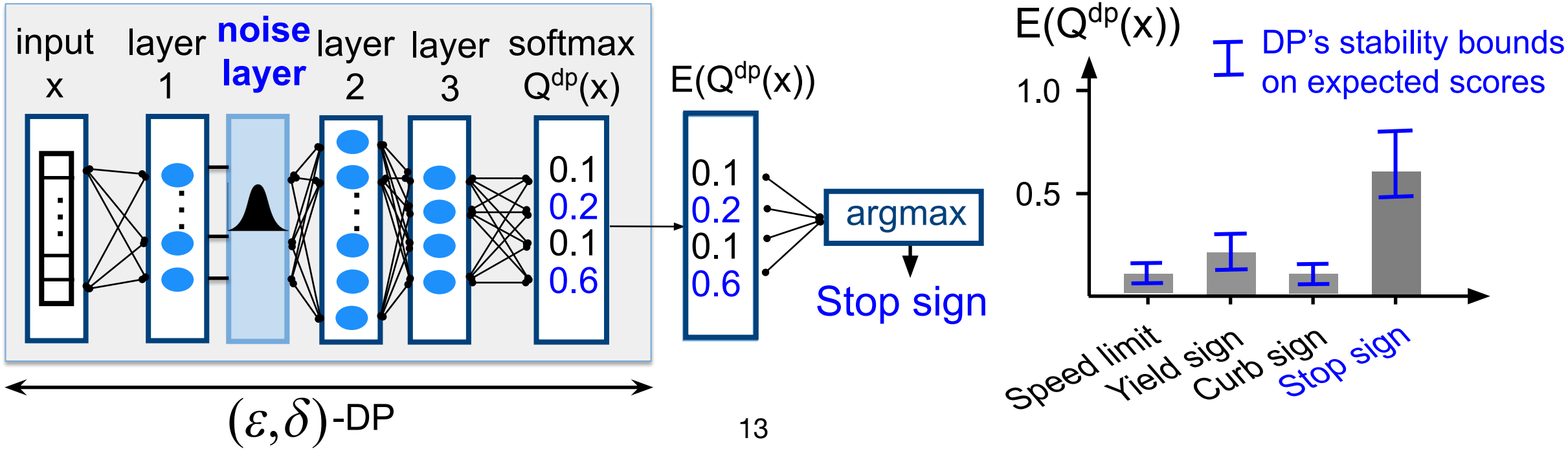
$$\text{for all } \alpha \text{ with } \|\alpha\|_2 = \sqrt{\sum \alpha_i^2} \le L:$$

$$\frac{E(Q^{dp}(x)) - \delta}{e^{\varepsilon}} \le E(Q^{dp}(x+\alpha))$$

$$E(Q^{dp}(x+\alpha)) \le e^{\varepsilon} E(Q^{dp}(x)) + \delta$$

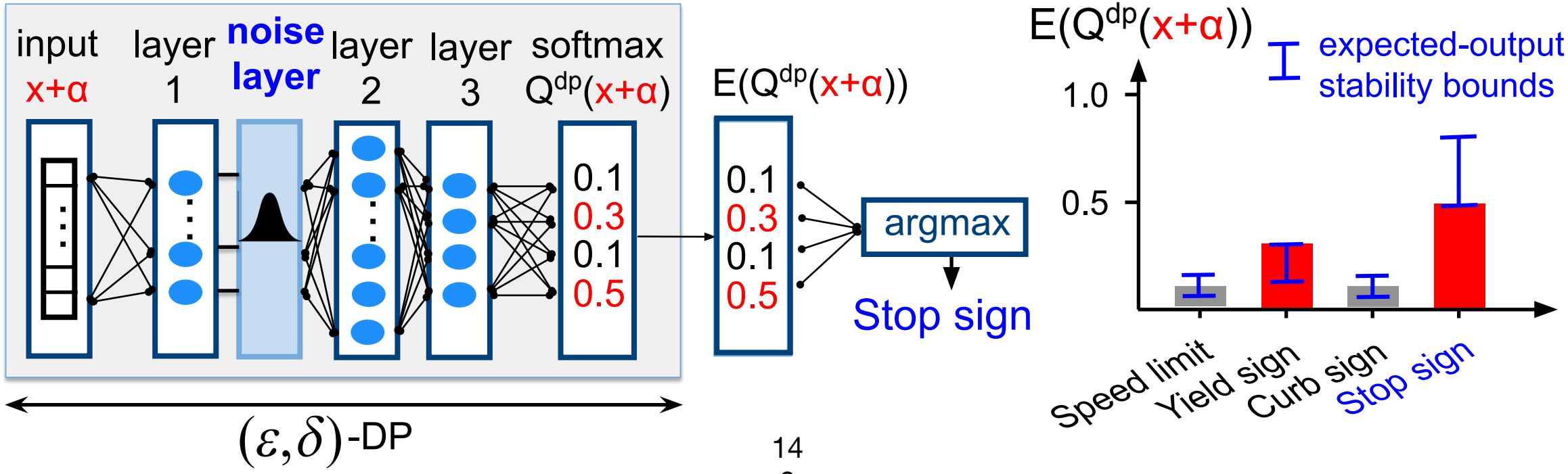
How It Works

1. Randomize prediction function to make it DP
2. Use expected scores to choose argmax
3. Use DP's stability bounds on expected scores to certify prediction on x



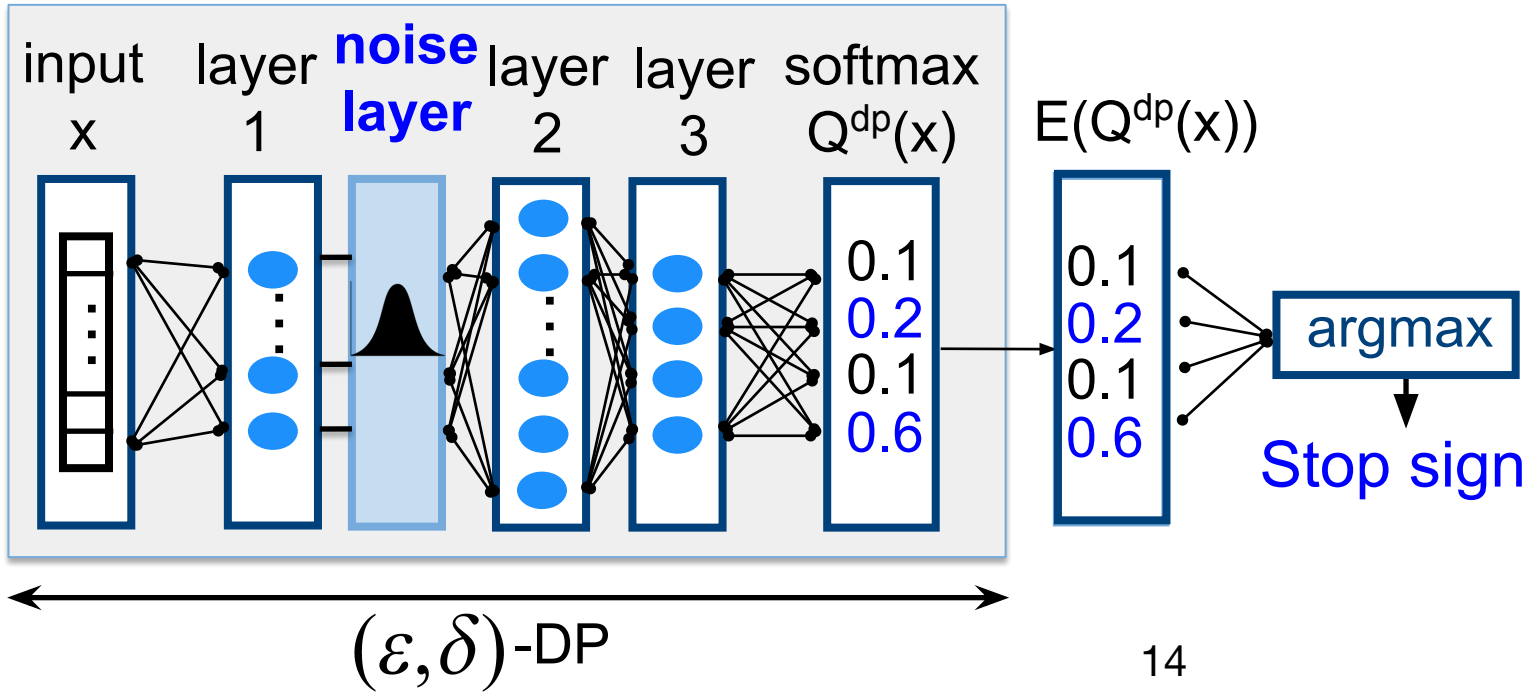
How It Works

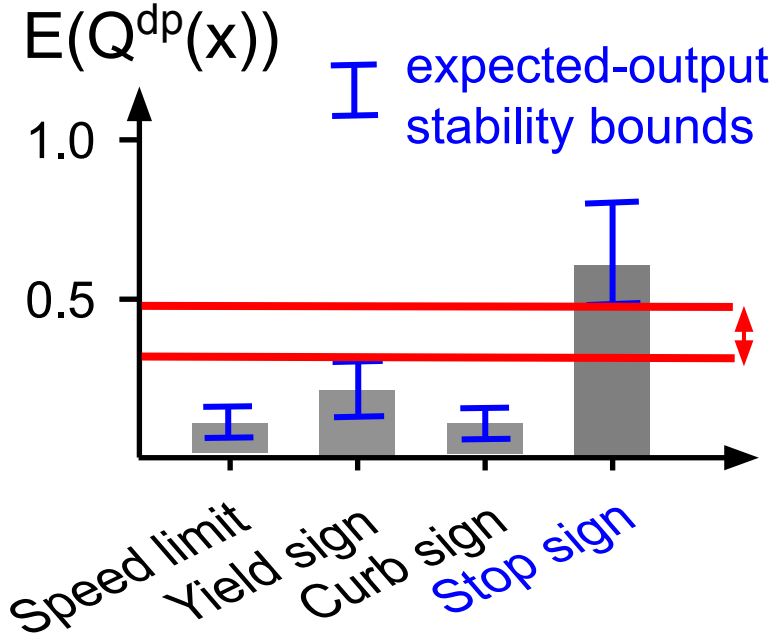
1. Randomize prediction function to make it DP
2. Use expected scores to choose argmax
3. Use DP's stability bounds on expected scores to certify prediction on x



How It Works

1. Randomize prediction function to make it DP
2. Use expected scores to choose argmax
3. Use DP's stability bounds on expected scores to certify prediction on x





DP for Generalization

(Hardt-16)

Generalization

- Central to ML is our ability to relate how a learning algorithm fares on a sample set to its performance on unseen instances. This is called **generalization**

Generalization

- Central to ML is our ability to relate how a learning algorithm fares on a sample set to its performance on unseen instances. This is called **generalization**

Risk (Out-of-sample Error)

$$R = \mathop{\mathrm{E}}_{z \sim D}\left[\ell(A(S), z)\right]$$

Empirical Risk (Train Error)

$$R_S = \frac{1}{n}\sum_{i=1}^{n}\ell(A(S), s_i)$$

Generalization Error

$$R - R_S$$

A= training function; D= input distribution; S= training set; n=|S|; ℓ = loss function

Generalization

- Central to ML is our ability to relate how a learning algorithm fares on a sample set to its performance on unseen instances. This is called **generalization**

Risk (Out-of-sample Error)	Empirical Risk (Train Error)	**Generalization Error**
$R = \mathop{\mathrm{E}}_{z\sim D}\left[\ell(A(S), z)\right]$	$R_S = \frac{1}{n}\sum_{i=1}^{n}\ell(A(S), s_i)$	$R - R_S$

A= training function; D= input distribution; S= training set; n=|S|; ℓ = loss function

- We care about R. If we manage to minimize R_S, all that matters is the **generalization error**. Many approaches exist that improve generalization error (mostly statistical)

Generalization ⇔ Stability

- **Thm: In expectation, generalization equals stability**
  - Proof in (Hardt-16)
- An algorithm is **stable** if its output doesn't change much if we perturb the input sample in a single point
- The theorem says that stability is **necessary and sufficient** for generalization

DP for Generalization

- DP is a strong stability constraint on algorithms
- DP thus provides an algorithmic approach to generalization in ML: **make the training function DP**
- It's been long known that adding randomness into training improves generalization
- The level of randomness added is likely insufficient to offer meaningful privacy, but the link DP<->generalization suggests that privacy isn't fundamentally at odds with functionality in ML

DP for Fairness
(Dwork+13)

Individual Fairness

- People who are similar from the perspective of the task at hand should be treated similarly
  - E.g., people with similar capabilities w.r.t. to a graduate program should all be either admitted or rejected
- But in ML, because of data biases and algorithmic amplification of them, small changes in people's relevant capabilities can lead to large changes in the predictions
- That's a sign of instability of the prediction function

DP for Individual Fairness

- Approach: **make the prediction function DP**
  - Similar to PixelDP, apply extension of DP to a distance metric among people with respect to their abilities for a task

- While in theory interesting, this approach is not very practical because it relies on a good distance metric among people, which is hard to define

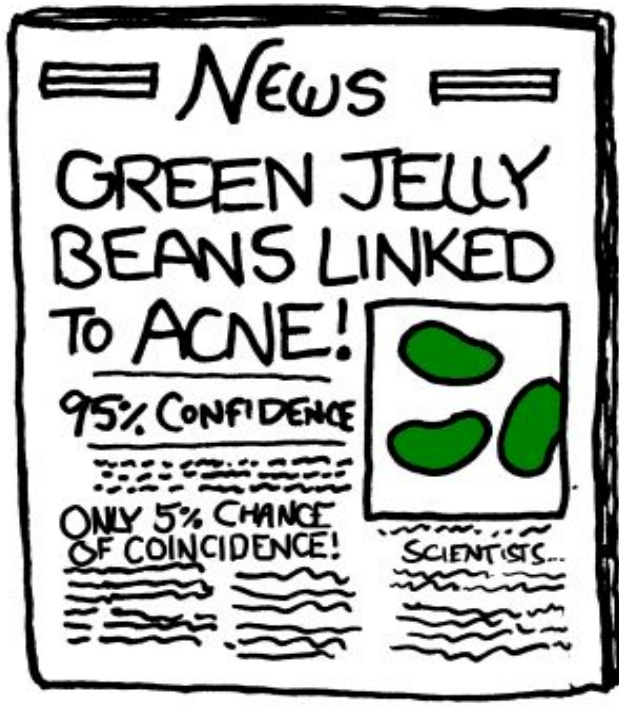
150

DP for Statistical Validity

(Dwork+15)

False Discoveries

- **Ideal scientific method:** Formulate your hypothesis, design your experiment to collect data, test your hypothesis on the data, report finding if statistically significant, and throw away the data.
- **In reality:** data is collected and reused to refine hypotheses, and the new hypotheses are tested on the same data, multiple times.
- Adaptive data reuse breaks assumptions of independence between hypotheses and test data, which hypothesis tests make to ensure statistical validity of the results. Referred to as p-hacking.



A Baseline Approach

- A baseline approach to allow statistical validity on top of a dataset collected from one study is to **split the dataset** into k components, where k is the number of hypotheses you anticipate testing on that dataset adaptively
- Each hypothesis runs on n/k points, so you can only run **k<<n adaptive hypothesis tests** on a dataset of size n
- Can we do better?

DP for Statistical Validity

- Problem: you're learning too much from the dataset, therefore your conclusions may overfit it and inherit its biases
- Approach: **make hypothesis tests DP and run on entire dataset**
- Recall DP supports adaptive composition. If you formulate a new hypothesis based on the results of a DP statistical test, and then you test again on the same dataset, you still have a bound on how much information you've extracted from your observations
- You can thus bound the number of tests you can perform while maintaining statistical validity. With advanced composition, the number of adaptive tests you can afford to run is **O(n^2)**

Take-Aways

- Many challenges in ML can be attributed to instability of some algorithm involved in learning: training, prediction, testing

- DP is a very strong stability constraint on algorithms. It thus has broad connections with many desirable properties in ML:
  - Training set privacy: make training function DP
  - Adversarial robustness: make prediction function DP
  - Generalization: make training function DP
  - Fairness: make prediction function DP
  - Statistical validity: make hypothesis test or model evaluation DP

- However, DP may be overly strong for some of these, and that impacts accuracy! Balance is needed, and future research may provide that

Cited References

(Bassily+16) R. Bassily, K. Nissim, A. Smith, T. Steinke, U. Stemmer, and J. Ullman. *Algorithmic stability for adaptive data analysis*. STOC 2016

(Dwork+15) C. Dwork, V. Feldman, M. Hardt, T. Pitassi, O. Reingold, A. Roth. *Preserving Statistical Validity in Adaptive Data Analysis*. STOC 2015

(Hardt-16) M. Hardt. *Stability as a foundation for machine learning.* Blog post, 2016

(Lecuyer+19) M. Lecuyer, V. Atlidakis, R. Geambasu, D. Hsu, S. Jana. *Certified Robustness to Adversarial Examples with Differential Privacy*. IEEE Security & Privacy, 2019

Cited References

(Vadhan, 2016) Vadhan. *The complexity of differential privacy.* https://privacytools.seas.harvard.edu/files/privacytools/files/complexityprivacy_1.pdf.

The End