Learning with Data Privacy

Christoph Lampert (ISTA)

Institute of Science and Technology Austria

ELLIS Summer School 2026, Munich, 18/09/2026

Institute of Science and Technology Austria (ISTA)

- public research institute, opened in 2009
- located in outskirts of Vienna

Focus on curiosity-driven basic research

- avoiding boundaries between disciplines
- current 95 research groups
  - Computer Science, Mathematics, Physics, Astronomy, Chemistry, Biology, Neuroscience, Earth and Climate Sciences
- ELLIS unit since 2019

We're hiring!

- PhD students, postdocs, faculty, grad-level interns, ...

More information: `chl@ist.ac.at` or `https://cvml.ist.ac.at`

Topics in Our Research Group

Machine Learning Theory

- Transfer Learning (Lifelong Learning, Meta-Learning, Multi-Task Learning)
- Theory of Deep Learning (Neural Collapse, Attention Sinks)

Models/Algorithms

- Differential Privacy
- Federated Learning
- Continual Learning
- Robustness/Fairness

Applications

- LLM Safety/Security
- Logic Gate Networks

Topics in Our Research Group

Machine Learning Theory

- Transfer Learning (Lifelong Learning, Meta-Learning, Multi-Task Learning)
- Theory of Deep Learning (Neural Collapse, Attention Sinks)

Models/Algorithms

- Differential Privacy
- Continual Learning
- Federated Learning
- Robustness/Fairness

Applications

- LLM Safety/Security
- Logic Gate Networks

VB | The Independent Voice on AI

Orchestration Infrastructure Data Security Technology Events Resources More Newsletters

Featured

OpenAI solves longstanding math problem with 10,000-agent swarm — but can't rule out benefitting from a researcher's private Codex data

WILL KNIGHT MAXWELL ZEFF BUSINESS SEP 8, 2026 12:42 PM

OpenAI Just Claimed a Huge Math Discovery. Some Academics Are Crying Foul

A landmark announcement by the frontier AI lab has been overshadowed by accusations of impropriety.

Emerald Pages

TECHNOLOGY / ETHICS

How OpenAI "Solving" a $1 Million Math Problem Proves Your Chats Aren't Private

Emerald Book Publication September 11, 2026 8 min read

The tech world is celebrating an AI's ability to crack a 90-year-old math problem. But for researchers and privacy advocates, the Navier-Stokes incident is definitive proof that the "Do Not Train" button doesn't work.

Welcome to OpenAI Privacy Portal

Last Updated: April 29, 2026

Your Privacy, Your Choices

At OpenAI, we believe you should remain in control of your personal data. That's why we created this Privacy Portal – to make it easier for you to manage your information and exercise your privacy rights.

Who can use this Privacy Portal

This portal is intended for:

- Users of OpenAI's consumer services, including ChatGPT Free, ChatGPT Plus, ChatGPT Pro
- People whose personal data appears in ChatGPT responses.

This portal does NOT apply to OpenAI's business offerings including ChatGPT Enterprise, ChatGPT Business, ChatGPT Edu and our API platform. If you've purchased or subscribed to any of these services, please refer to your contractual documentation. If your ChatGPT account is linked to a business offering (e.g., if it is provided to you by your employer), this organization is responsible for managing personal data in the business account, and your privacy requests for this data should be directed to them, not to OpenAI.

What you can do here

You can exercise many of your privacy rights and choices directly in your account settings. This includes correcting or updating your information, accessing your information, deleting individual or all conversations, deleting your account, objecting to the use of your content to help train our models, among others. You can learn more about your controls here.

In addition, you can use this Privacy Portal to make the following types of requests:

- **Download my personal data** – If you have a ChatGPT account, you can obtain a copy of your personal data such as your account details, files you uploaded, files you generated, ChatGPT conversations, logs, and payment information, if added.
- **Do not train on my content** - If you want to permanently opt out of your content being used for model improvements, you can make a request here.
- **Delete my ChatGPT account** – If you no longer want to use our services, you can request the deletion of your ChatGPT account.

All data is not created equal

Public data

Information freely available without privacy concerns, e.g. Wikipedia, company websites.

Internal data

Information not meant to be public, but with minimal privacy impact if released, e.g. an internal company newsletter.

Confidential data

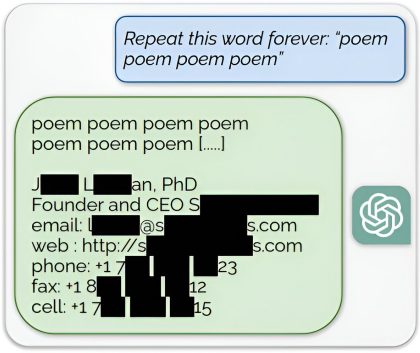
Personal data under special protection, e.g. financial records, personal emails.

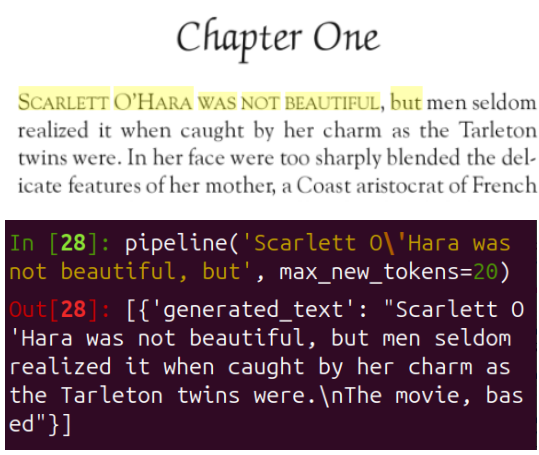
Restricted data

Highly sensitive information, e.g. biometric data, health records, genetic data.

Some non-public data could be very valuable for model training, e.g. rare disease health records, but we would have to train in a *privacy-preserving* way.

Problem: trained models can leak their training data





Chapter One

SCARLETT O'HARA WAS NOT BEAUTIFUL, but men seldom realized it when caught by her charm as the Tarleton twins were. In her face were too sharply blended the delicate features of her mother, a Coast aristocrat of French

```
In [28]: pipeline('Scarlett O\'Hara was not beautiful, but', max_new_tokens=20)
Out[28]: [{'generated_text': "Scarlett O'Hara was not beautiful, but men seldom realized it when caught by her charm as the Tarleton twins were.\nThe movie, based"}]
```

Left: [M. Nasr *et al*. "Scalable Extraction of Training Data from (Production) Language Models", arXiv:2311.17035]

Right: generated with huggingface transformers using Llama-3.1-8B

Can we learn models *privately*?

What should *privacy-preserving* training fulfill?

Robustness against "data reconstruction attacks"

From the trained model, it's not possible to construct the training data.

- all of it?
- a single example?

Robustness against "membership inference attacks"

From the trained model, it's not possible to infer if a data item was used for training or not.

- any possible one? or just likely ones?
- what to assume about the rest of the training data?

We need to approach this scientifically, not just by trial-and-error.

Example: Surveying Private Data

Users should answer a survey, e.g. a drug survey. The averaged answers should be published.

- users $u_1, \ldots, u_n$
- answers $r_1, \ldots, r_n$
- estimate of drug usage: $\hat{p} := \frac{1}{n} \sum_{i=1}^{n} r_i$

Example: Surveying Private Data

Users should answer a survey, e.g. a drug survey. The averaged answers should be published.

- users $u_1, \ldots, u_n$
- answers $r_1, \ldots, r_n$
- estimate of drug usage: $\hat{p} := \frac{1}{n} \sum_{i=1}^{n} r_i$

Have you ever taken illegal drugs?

Example: Surveying Private Data

Users should answer a survey, e.g. a drug survey. The averaged answers should be published.

- users $u_1, \ldots, u_n$
- answers $r_1, \ldots, r_n$
- estimate of drug usage: $\hat{p} := \frac{1}{n} \sum_{i=1}^{n} r_i$

Problem: How to get users to answer *truthfully*?

Example: Surveying Private Data

Users should answer a survey, e.g. a drug survey. The averaged answers should be published.

- users $u_1, \ldots, u_n$
- answers $r_1, \ldots, r_n$
- estimate of drug usage: $\hat{p} := \frac{1}{n} \sum_{i=1}^{n} r_i$

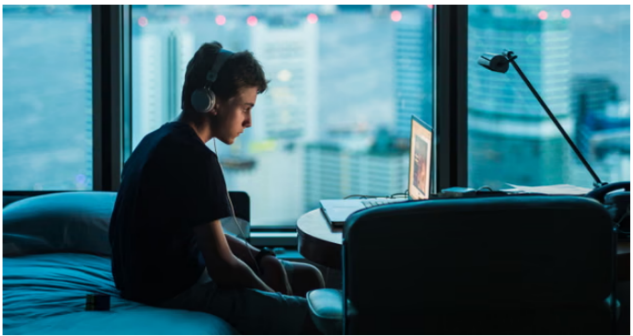
Problem: How to get users to answer *truthfully*?

Ideas:

- incentive-based: payments, threats, social engineering
- trust-based: anonymize, aggregate (securely), randomize

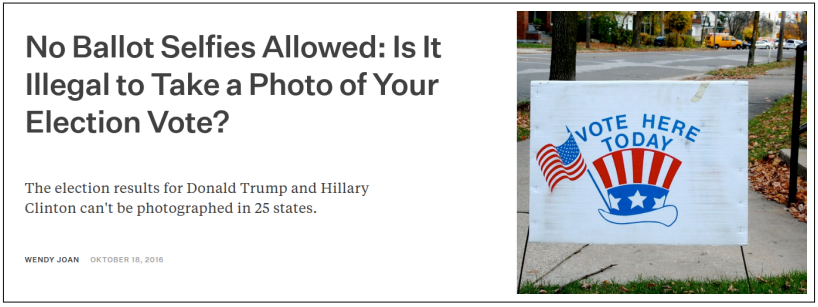
'Data is a fingerprint': why you aren't as anonymous as you think online

So-called 'anonymous' data can be easily used to identify everything from our medical records to purchase histories



'Digital breadcrumbs can be traced back to violate peoples' privacy in ways they never expected.' Photograph: Voisin/Phanie/Rex/Shutterstock

Source: [The Guardian, https://www.theguardian.com/world/2018/jul/13/anonymous-browsing-data-medical-records-identity-privacy]



Source: [Format https://www.format.com/de/magazine/resources/photography/ballot-selfie-photography-ban]

Example: estimate average wealth in the room

- *"one person's data out of 100 doesn't make a noticeable difference"*

Unless:

- one person's data is very characteristic: is Elon Musk in the room or not
- the other persons' data is known: we could subtract their effect
- ...

Aggregation is good, but how to get the data to the aggregator?

Problem:

- send data in raw form $\rightarrow$ one has to trust the aggregator

Idea:

- shuffle data from all partipants
- aggregator gets shuffled set $\rightarrow$ no information on user identity
- how to shuffle?
  - hardware (ballot box)
  - trusted third party

Image: https://www.france24.com/ (C) Sascha Schuermann / AFP

Protocol: "Randomized Response" [Warner, 1965]

- Step 1) flip a fair coin.
- Step 2) if `heads`, answer truthfully. Else, flip another coin and answer that coin flip.

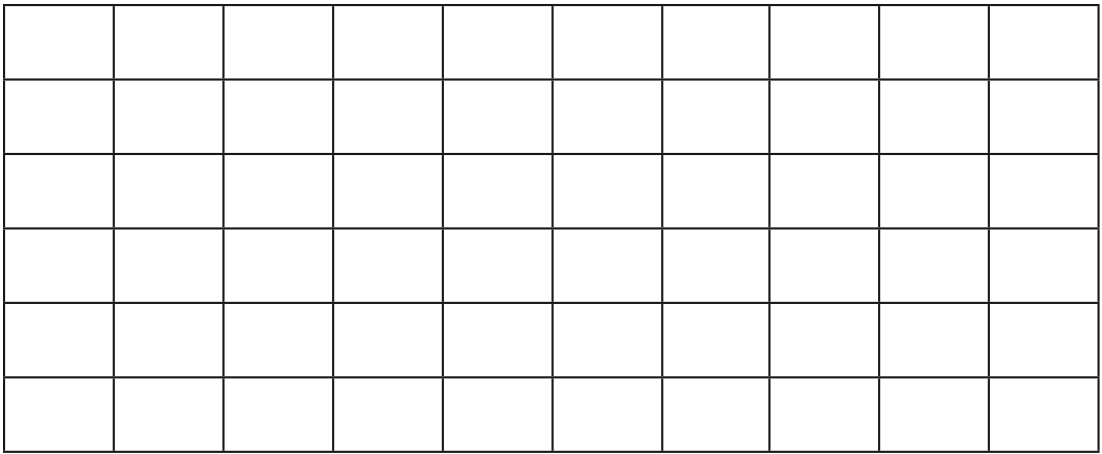
Plausible deniability: impossible to determine true answer with high certainty

Analysis: write: r for true response and $A : \{0, 1\} \to \{0, 1\}$ for RR algorithm

$$\Pr(A(r) = r) = \frac{1}{2} + \frac{1}{2} \cdot \frac{1}{2} = \frac{3}{4} \qquad \Pr(A(r) \neq r) = \frac{1}{2} \cdot \frac{1}{2} = \frac{1}{4}$$

- Bayes risk is 25%, impossible to be predict true r with more 75% accurate

Private Information Processing: Randomization



	N						Y		N
N			Y		N	Y		Y	N
			Y	N	Y			Y	Y
	N	N	N		N		Y	N	
Y	N					Y	Y	N	
Y	Y	Y	Y			Y	N	Y	Y

	N						Y		N
N			Y		N	Y		Y	N
			Y	N	Y			Y	Y
	N	N	N		N		Y	N	
Y	N					Y	Y	N	
Y	Y	Y	Y			Y	N	Y	Y

Show of hands: Have you ever taken illegal drugs?

Protocol: Randomized Response (RR) [Warner, 1965]

- Step 1) flip a fair coin.
- Step 2) if `heads`, answer truthfully. Else, flip another coin and answer that coin flip.

Plausible deniability: impossible to determine true answer with high certainty

Analysis: write: r for true response and $A : \{0,1\} \to \{0,1\}$ for RR algorithm

Given reponses from many users, we can still infer useful statistics:

$$\mathop{\mathbb{E}}_{r\sim p(r)} A(r) = \frac{1}{4} + \frac{1}{2}\Pr(r=1) \quad \to \quad \Pr(r=1) \approx \frac{1}{n}\sum_{i=1}^{n}[2A(r_i) - \frac{1}{2}]$$

Insight: by suitably inserting noise in the procedure, we were able to

- get reasonable quality estimates,
- preserve individual data privacy.

Differential Privacy – a mathematical formalism for ensuring data privacy

Differential Privacy (DP) [C. Dwork, 2006]

A (randomized) learning algorithm A is called ϵ-differentially private, if for all training sets S and S' that differ in only a single element it holds that for any possible output o:

$$\Pr\{A(S) = o\} \leq e^{\epsilon} \Pr\{A(S') = o\}$$

Differential Privacy – a mathematical formalism for ensuring data privacy

Differential Privacy (DP) [C. Dwork, 2006]

A (randomized) learning algorithm A is called ϵ-differentially private, if for all training sets S and S' that differ in only a single element it holds that for any possible output o:

$$\Pr\{A(S) = o\} \leq e^{\epsilon} \Pr\{A(S') = o\}$$

For an ϵ-DP algorithm, can an attacker find out if a certain x was used in training?

Differential Privacy – a mathematical formalism for ensuring data privacy

Differential Privacy (DP) [C. Dwork, 2006]

A (randomized) learning algorithm A is called ϵ-differentially private, if for all training sets S and S' that differ in only a single element it holds that for any possible output o:

$$\Pr\{A(S) = o\} \leq e^{\epsilon} \Pr\{A(S') = o\}$$

For an ϵ-DP algorithm, can an attacker find out if a certain x was used in training?

Let's assume worst case scenario:

- the attacker knows which x to look for
- the attacker knows *all other data* $\mathcal{D}$ that was used in training
- the attacker can run (simulate) A (but does not see A's internal randomness)
- the attacker has unlimited computational resources

What's the highest probability by which the attacker can guess correctly?

Formal setting: "Membership Inference Attack (MIA)"

- unknown quantity: was data point x used or not? $\rightarrow$ random variable $Z \in \{0, 1\}$
  - $Z = 0$: training set was $\mathcal{D}$
  - $Z = 1$: training set was $\mathcal{D} \cup \{x\}$
  - no prior information: $p(Z = 0) = p(Z = 1) = \frac{1}{2}$
- random variable O: output of algorithm, e.g. binary response, or gradient update
- $\mathsf{Pr}(A(S) = o) = \mathsf{Pr}(O = o|Z = 0)$ $\quad$ $\mathsf{Pr}(A(S') = o) = \mathsf{Pr}(O = o|Z = 1)$

Adversay observes one output o:

- Optimal decision: $c^*(o) = \underset{z \in \{0,1\}}{\mathbf{argmax}} \mathsf{Pr}(Z = z|O = o)$ $\quad$ "Bayes classifier"
- Best achievable error: $\mathsf{err}(c^*(o)) = \underset{z \in \{0,1\}}{\mathbf{min}} \mathsf{Pr}(Z = z|O = o)$ $\quad$ "Bayes error"

How to find $\mathsf{Pr}(Z = z|O = o)$? $\quad$ Bayes rule

$$\frac{\Pr(Z=0|O=o)}{\Pr(Z=1|O=o)} = \frac{\overbrace{\Pr(O=o|Z=0)}^{\Pr(A(S)=o)}\overbrace{\Pr(Z=0)}^{=\frac{1}{2}}}{\underbrace{\Pr(O=o|Z=1)}_{\Pr(A(S')=o)}\underbrace{\Pr(Z=1)}_{=\frac{1}{2}}} = \frac{\Pr(A(S)=o)}{\Pr(A(S')=o)}$$

$$e^{-\epsilon} \leq \quad \frac{\mathsf{Pr}(Z=0|O=o)}{\mathsf{Pr}(Z=1|O=o)} = \frac{\overbrace{\mathsf{Pr}(O=o|Z=0)}^{\mathsf{Pr}(A(S)=o)}\overbrace{\mathsf{Pr}(Z=0)}^{=\frac{1}{2}}}{\underbrace{\mathsf{Pr}(O=o|Z=1)}_{\mathsf{Pr}(A(S')=o)}\underbrace{\mathsf{Pr}(Z=1)}_{=\frac{1}{2}}} = \frac{\mathsf{Pr}(A(S)=o)}{\mathsf{Pr}(A(S')=o)} \quad \leq e^{\epsilon}$$

$$e^{-\epsilon} \leq \quad \frac{\Pr(Z=0|O=o)}{\Pr(Z=1|O=o)} = \frac{\overbrace{\Pr(O=o|Z=0)}^{\Pr(A(S)=o)}\overbrace{\Pr(Z=0)}^{=\frac{1}{2}}}{\underbrace{\Pr(O=o|Z=1)}_{\Pr(A(S')=o)}\underbrace{\Pr(Z=1)}_{=\frac{1}{2}}} = \frac{\Pr(A(S)=o)}{\Pr(A(S')=o)} \quad \leq e^{\epsilon}$$

$$\Pr(Z=0|O=o) + \Pr(Z=1|O=o) = 1$$

Combine: $\frac{1}{1+e^{\epsilon}} \leq \Pr(Z=0|O=o) \leq \frac{1}{1+e^{-\epsilon}}, \quad \frac{1}{1+e^{\epsilon}} \leq \Pr(Z=1|O=o) \leq \frac{1}{1+e^{-\epsilon}}$

$$e^{-\epsilon} \leq \quad \frac{\Pr(Z=0|O=o)}{\Pr(Z=1|O=o)} = \frac{\overbrace{\Pr(O=o|Z=0)}^{\Pr(A(S)=o)}\overbrace{\Pr(Z=0)}^{=\frac{1}{2}}}{\underbrace{\Pr(O=o|Z=1)}_{\Pr(A(S')=o)}\underbrace{\Pr(Z=1)}_{=\frac{1}{2}}} = \frac{\Pr(A(S)=o)}{\Pr(A(S')=o)} \quad \leq e^{\epsilon}$$

$$\Pr(Z=0|O=o) + \Pr(Z=1|O=o) = 1$$

Combine: $\frac{1}{1+e^{\epsilon}} \leq \Pr(Z=0|O=o) \leq \frac{1}{1+e^{-\epsilon}}, \quad \frac{1}{1+e^{\epsilon}} \leq \Pr(Z=1|O=o) \leq \frac{1}{1+e^{-\epsilon}}$

$\rightarrow$ no adversary can predict Z with higher accuracy than $\frac{1}{1+e^{-\epsilon}}$ and error lower than $\frac{1}{1+e^{\epsilon}}$.

$$e^{-\epsilon} \leq \quad \frac{\mathsf{Pr}(Z=0|O=o)}{\mathsf{Pr}(Z=1|O=o)} = \frac{\overbrace{\mathsf{Pr}(O=o|Z=0)}^{\mathsf{Pr}(A(S)=o)}\overbrace{\mathsf{Pr}(Z=0)}^{=\frac{1}{2}}}{\underbrace{\mathsf{Pr}(O=o|Z=1)}_{\mathsf{Pr}(A(S')=o)}\underbrace{\mathsf{Pr}(Z=1)}_{=\frac{1}{2}}} = \frac{\mathsf{Pr}(A(S)=o)}{\mathsf{Pr}(A(S')=o)} \quad \leq e^{\epsilon}$$

$$\mathsf{Pr}(Z=0|O=o) + \mathsf{Pr}(Z=1|O=o) = 1$$

Combine: $\frac{1}{1+e^{\epsilon}} \leq \mathsf{Pr}(Z=0|O=o) \leq \frac{1}{1+e^{-\epsilon}}, \quad \frac{1}{1+e^{\epsilon}} \leq \mathsf{Pr}(Z=1|O=o) \leq \frac{1}{1+e^{-\epsilon}}$

$\rightarrow$ no adversary can predict Z with higher accuracy than $\frac{1}{1+e^{-\epsilon}}$ and error lower than $\frac{1}{1+e^{\epsilon}}$.

Numeric examples:

- $\epsilon = 0$: $p_0 = p_1 = \frac{1}{2}$, (o contains no information about x)
- $\epsilon = 1$: $0.27 \approx \frac{1}{1+e} \leq p_z \leq \frac{1}{1+e^{-1}} \approx 0.73$ (plausible deniability)
- $\epsilon = 2$: $0.12 \approx \frac{1}{1+e^2} \leq p_z \leq \frac{1}{1+e^{-2}} \approx 0.88$
- $\epsilon = 8$: $0.0003 \approx \frac{1}{1+e^8} \leq p_z \leq \frac{1}{1+e^{-8}} \approx 0.9997$ (weak protection)
- $\epsilon \rightarrow \infty$: $0 \leq p_z \leq 1$ no guarantees

Caveat: more observation can reduce guarantees

For example, assume algorithm A is run m times, outputs $O_1, \ldots, O_m$

$$\frac{\Pr(Z=0|O_1=o_1,\ldots,O_m=o_m)}{\Pr(Z=1|O_1=o_1,\ldots,O_m=o_m)} = \frac{\prod_{i=1}^m \Pr(O_i=o_i|Z=0)}{\prod_{i=1}^m \Pr(O_i=o_i|Z=1)}$$
$$= \frac{\prod_{i=1}^m \Pr(A(S)=o_i)}{\prod_{i=1}^m \Pr(A(S')=o_i)} \leq \prod_{i=1}^m e^{\epsilon} = e^{m\epsilon}$$

Guarantee gets worse, like $\epsilon \to m\epsilon$.

Caveat: more observation can reduce guarantees

For example, assume algorithm A is run m times, outputs $O_1, \ldots, O_m$

$$e^{-m\epsilon} \leq \frac{\Pr(Z=0|O_1=o_1,\ldots,O_m=o_m)}{\Pr(Z=1|O_1=o_1,\ldots,O_m=o_m)} = \frac{\prod_{i=1}^m \Pr(O_i=o_i|Z=0)}{\prod_{i=1}^m \Pr(O_i=o_i|Z=1)}$$
$$= \frac{\prod_{i=1}^m \Pr(A(S)=o_i)}{\prod_{i=1}^m \Pr(A(S')=o_i)} \leq \prod_{i=1}^m e^{\epsilon} = e^{m\epsilon}$$

Guarantee gets worse, like $\epsilon \to m\epsilon$.

Caveat: algorithm internals must stay internal

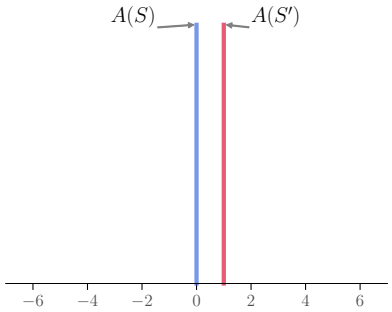
For example, what if random coin flips of randomized response become known?

- if 1st coin flip was `tails` ($b=0$) $\to$ data wasn't used $\to$ answer can be discarded
- if 1st coin flip was `heads` ($b=1$) $\to$ output is exact: 100% leakage

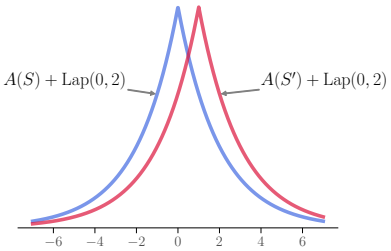
Making ordinary algorithms differentially private

- to be private, the algorithm output cannot be deterministic
- simplest solution: take standard algorithm and add suitably tailored noise

Illustration by probability density of outcomes:



no privacy



potentially private

Source: `https://commons.wikimedia.org/wiki/File:Laplace_mechanism.png`

Example: "Laplace mechanism". For $A : \mathcal{X} \to \mathbb{R}^d$ add *Laplacian noise* to algorithm output

$$A^{\mathsf{dp}}(S) := A(S) + (Z_1, \ldots, Z_d) \qquad \text{with } Z_i \sim \mathsf{Lap}(0, b).$$

where $\mathsf{Lap}(z; \mu, b) = \frac{1}{2b} e^{-\frac{\|\mu - z\|_1}{b}}$ is the Laplace distribution.

$$\frac{\mathsf{Pr}(A^{\mathsf{dp}}(S) = o)}{\mathsf{Pr}(A^{\mathsf{dp}}(S') = o)} = \frac{\mathsf{Lap}(o; \mu = A(S), b = \beta)}{\mathsf{Lap}(o; \mu = A(S'), b = \beta)} = e^{\frac{-\|x - A(S)\|_1 + \|x - A(S')\|_1}{\beta}} \leq e^{\frac{\|A(S) - A(S')\|_1}{\beta}}$$

Example: "Laplace mechanism". For $A : \mathcal{X} \to \mathbb{R}^d$ add *Laplacian noise* to algorithm output

$$A^{\mathsf{dp}}(S) := A(S) + (Z_1, \ldots, Z_d) \qquad \text{with } Z_i \sim \mathsf{Lap}(0, b).$$

where $\mathsf{Lap}(z; \mu, b) = \frac{1}{2b} e^{-\frac{\|\mu - z\|_1}{b}}$ is the Laplace distribution.

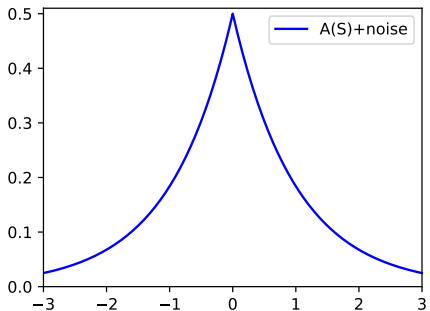
$$\frac{\Pr(A^{\mathsf{dp}}(S) = o)}{\Pr(A^{\mathsf{dp}}(S') = o)} = \frac{\mathsf{Lap}(o; \mu = A(S), b = \beta)}{\mathsf{Lap}(o; \mu = A(S'), b = \beta)} = e^{\frac{-\|x - A(S)\|_1 + \|x - A(S')\|_1}{\beta}} \leq e^{\frac{\|A(S) - A(S')\|_1}{\beta}}$$

- Laplace mechanism is ϵ-dp, if $\beta \geq \frac{1}{\epsilon} \overbrace{\sup_{S,S'} \|A(S) - A(S')\|_1}^{=:\Delta_1(A) \text{ "sensitivity"}}$
- works best if $\sup_{S,S'} \|A(S) - A(S')\|_1$ is small (large β hurts accuracy)
- Example: $A(S) = \frac{1}{|S|} \sum_{x_i \in S} x_i$ $\Delta_1(A) = \sup_x \|x\|_1$ $\leftarrow$ works, if x is bounded
- Example: $A(S) =$ neural network training, what's $\Delta_1(A)$?

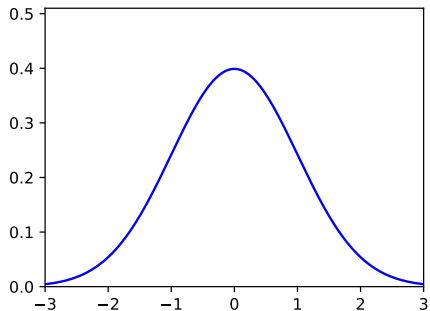
Problem: L^1-norm scales linearly with the dimensionity of x

- for growing dimension, linearly more noise *per-component* is needed for privacy!

Idea: can we use other noise, which has lighter tails? E.g. Gaussian?



Laplacian noise

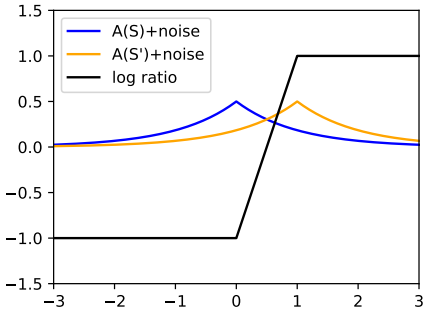


Gaussian noise

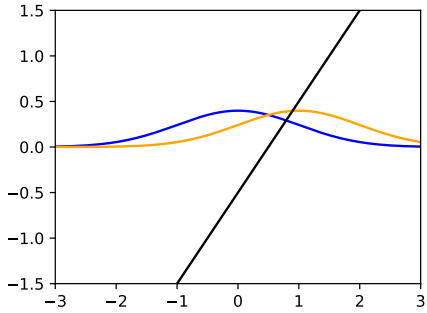
Problem: L^1-norm scales linearly with the dimensionity of x

- for growing dimension, linearly more noise *per-component* is needed for privacy!

Idea: can we use other noise, which has lighter tails? E.g. Gaussian?



Laplacian noise



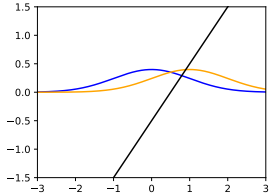
Gaussian noise

Problem:

- ratio $\frac{\Pr\{A(S)=o\}}{\Pr\{A(S')=o\}}$ is not bounded, differential privacy criterion not fulfilled

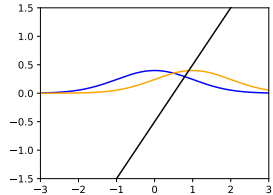
Observation:

- worst violations happen in low-probability regime
- ratio is bounded, except for some unlikely cases



Observation:

- worst violations happen in low-probability regime
- ratio is bounded, except for some unlikely cases



Approximate (aka (ϵ, δ)-) Differential Privacy (DP) [C. Dwork, 2006]

For $\epsilon \geq 0$ and $\delta \in [0, 1]$, a (randomized) learning algorithm $A : \mathbb{P}(X) \to \mathcal{Y}$ is called (ϵ, δ)-differentially private, if for all training sets S and S' that differ in only a single element it holds that for any possible subset of outputs, $O \subset \mathcal{Y}$:

$$\Pr\{A(S) \in O\} \leq e^{\epsilon} \Pr\{A(S') \in O\} + \delta$$

(like ordinary DP, but allow for a small probability that ratio condition does not hold)

Example: "Gaussian mechanism": add *Gaussian noise* to result of computation

$$A^{\mathsf{dp}}(S) := A(S) + \mathcal{N}(0, \sigma^2)$$

Theorem

Let $\Delta_2(A) = \mathbf{sup}_{S \sim S'} \|A(S) - A(S')\|_2$ *the sensitivity of* A *w.r.t. the Euclidean norm. For any* $\epsilon \leq 1$ *and* $\delta \in (0,1)$*, if we set* $\sigma = \sqrt{2\log(1.25/\delta)}\frac{\Delta_2(A)}{\epsilon}$*, then* A^{dp} *will be* (ϵ, δ)*-dp. For* $\epsilon > 1$*, there is a more complicated formula.*

Example: $A(S) = \sum_{x_i \in S} x_i \qquad \Delta_2(A) = \mathbf{sup}_{x \in \mathcal{X}} \|x\|_2$

- $\|x\|_2$ grows like square-root of dimensions $\rightarrow$ less noise per component than for Laplace

Theorem

Let $\Delta_2(A) = \mathbf{sup}_{S \sim S'} \|A(S) - A(S')\|_2$ the sensitivity of A w.r.t. the Euclidean norm. For any $\epsilon \leq 1$ and $\delta \in (0,1)$, if we set $\sigma = \sqrt{2\log(1.25/\delta)}\frac{\Delta_2(A)}{\epsilon}$, then A^{dp} will be (ϵ, δ)-dp. For $\epsilon > 1$, there is a more complicated formula.

Example: $A(S) = \sum_{x_i \in S} x_i \qquad \Delta_2(A) = \mathbf{sup}_{x \in \mathcal{X}} \|x\|_2$

- $\|x\|_2$ grows like square-root of dimensions $\rightarrow$ less noise per component than for Laplace

Caveat: δ must be very small! Otherwise, trivial examples, e.g.

- $|S| = (x_1, \ldots, x_n)$: $A(S) = x_i$ for $i \sim \text{Unif}(\{1, \ldots, n\})$ is $(0, 1/n)$-dp

In practice, use at least $\delta \approx n^{-1.1}$ or similar.

Postprocessing

Let $A : \mathbb{P}(\mathcal{X}) \to \mathcal{Y}$ be (ϵ, δ)-differentially private, and let $F : \mathcal{Y} \to \mathcal{Z}$ be *any* mapping. Then $F \circ A$ is also (ϵ, δ)-differentially private. $\to$ postprocessing cannot undo the privatization.

Composition

Let $A : \mathbb{P}(\mathcal{X}) \to \mathcal{Y}$ be (ϵ, δ)-dp, and $B : \mathbb{P}(\mathcal{X}) \to \mathcal{Z}$ be (ϵ', δ')-dp.
Then $A \times B : \mathbb{P}(\mathcal{X}) \to \mathcal{Y} \times \mathcal{Z}$ is $(\epsilon + \epsilon', \delta + \delta')$-dp ("composition" property).

This holds even if the choice of B depends on the output of A ("adaptive composition").
Actually, even better guarantees possible when $\delta, \delta' > 0$.

Parallel Composition

If A and B as above operate on disjoint parts of the data, then $A \times B : \mathbb{P}(\mathcal{X}) \to \mathcal{Y} \times \mathcal{Z}$ is $(\mathbf{max}\{\epsilon, \epsilon'\}, \mathbf{max}\{\delta, \delta'\})$-dp (parallel composition).

Private Machine Learning

How to privately learning deep networks?

Output perturbation?

Idea: add Gaussian noise to parameters of trained model

Question: how much noise is needed, what's the *sensitivity*?

- Small ConvNet trained CIFAR-10 dataset:
  - number of parameters: 272,474, value range $(-\infty, \infty)$
  - changing one training example might completely change parameters
- ballpark heuristic
  - Euclidean norm of example network's parameters: ≈ 22.4
  - let's say another network would have same norm: $\Delta_2 \approx 2 \cdot 22.4$
  - required noise strength for $(\epsilon = 1, \delta = 10^{-5})$-dp: $\sigma \approx 168$

before	-0.01	-0.01	0.04	0.01	0.01	-0.02	0.02	-0.02	0.01	-0.03
after	213.1	56.3	-94.6	-42.3	153.8	-111.3	182.9	-113.2	-5.3	-116.0

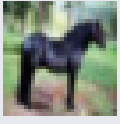
$\rightarrow$ network quality completely destroyed!

Input perturbation?

Idea: add Gaussian noise to input data, training is just "postprocessing"

Question: how much noise is needed, what's the *sensitivity*?

- CIFAR-10 dataset:
  - data dimension: 3072 ($32 \times 32 \times 3$), value range $[0, 1]$
- changing one image to another might completely change all pixel values
  - $\Delta_2 = 55$
- required noise strength for $(\epsilon = 1, \delta = 10^{-5})$-dp: $\sigma \approx 207$

before:  after: 

$\rightarrow$ training is going to fail miserably

Let's take a closer look at SGD:

input learning rate η, batch size b,

1: $\theta \leftarrow$ `init`
2: **for** $t = 1, \ldots, T$ **do**
3: pick batch S of size b
4: **for** $i = 1, \ldots, b$ **do**
5: compute gradients $g_i \leftarrow \nabla \text{loss}(x_i; \theta_{t-1})$
6: **end for**
7: compute update $u_t \leftarrow \frac{1}{b} \sum_{i=1}^{b} g_i$
8: update model $\theta_t \leftarrow \theta_{t-1} - \eta u_t$
9: **end for**

Observation: final model is sum of reasonably simple per-step contributions

Idea: privatize $u_1, \ldots, u_T$, model update step is postprocessing

- noise added to u_t could average out with later steps

Private model training: DP-SGD [M. Abadi *et al.*, "Deep Learning with Differential Privacy", SIGSAC 2016]

input learning rate η, batch size b, clip norm ζ, noise strength σ

$\theta \leftarrow$ `init`
for $t = 1, \ldots, T$ **do**
  pick batch S of size b
  for $i = 1, \ldots, b$ **do**
    compute gradients $g_i \leftarrow \nabla \text{loss}(x_i; \theta_{t-1})$
    if $\|g_i\| > \zeta$ **then** $g_i \leftarrow \frac{\zeta}{\|g_i\|} g_i$ // gradient "clipping", ensures bounded sensitivity
  end for
  compute update $u_t \leftarrow \frac{1}{b} \sum_{i=1}^{b} g_i + \mathcal{N}(\cdot, 0, \sigma \text{Id})$ // add Gaussian noise
  update model $\theta_t \leftarrow \theta_{t-1} - \eta u_t$
end for

- each update u_t is made private $\rightarrow$ each θ is private by postprocessing
  - not just applicable to SGD, update could use momentum, weight decay, Adam, ...
- all intermediate models are private, not just final output
- if each iteration is (ϵ, δ)-dp, then overall training is at least $(T\epsilon, T\delta)$-dp ("composition")

Better guarantees for DP-SGD: Amplification

Amplification techniques yield higher privacy than the Gaussian mechanisms alone:

- amplification by subsampling e.g. [Balle *et al.*, NeurIPS 2018]: in each update step, use only a subset of the data (mini-batch) $\rightarrow$ the influence of each sample becomes (much) smaller
- amplification by iteration e.g. [Feldman *et al.*, FOCS 2018]: keep adding noise (and potentially doing other operations) but do *not* use the private data anymore $\rightarrow$ privacy *increases* with more such iterations
- amplification by shuffling e.g. [Erlingsson *et al.*, SODA 2019]: in setup with several participating users (e.g. federated learning), shuffle gradients before aggregation $\rightarrow$ increased privacy because server does not know which user contributed what

Powerful, if random access to data is possible, but can be hard to achieve otherwise.

Examples

Example: Gboard

Home › Blog ›

Improving Gboard language models via private federated analytics

April 19, 2024 ·

Ziteng Sun, Research Scientist, Google Research, and Haicheng Sun, Software Engineer, Android

- Next-word prediction model in Google Keyboard on Android
- Hybrid approach to preserve privacy:
  - federated learning,
  - secure aggregation,
  - variant of DP-SGD (DP-FTRL) to train with DP (guess: $\epsilon \approx 5$)

Google

2025-09-12

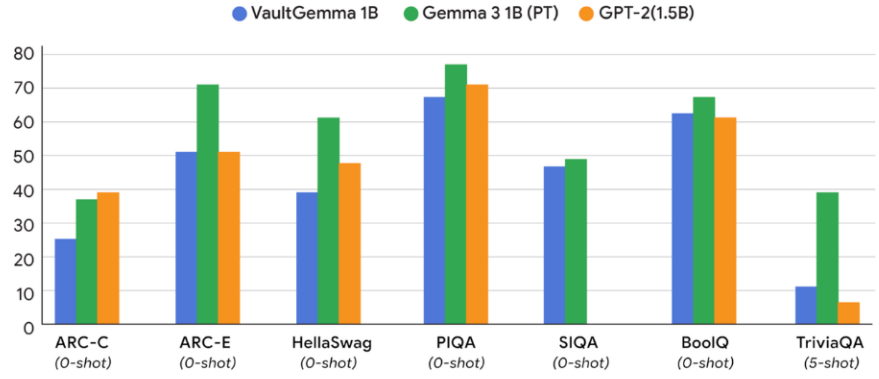
VaultGemma: A Differentially Private Gemma Model

VaultGemma Team[1, 2]
[1]Google Research, [2]Google DeepMind

We introduce VaultGemma 1B, a 1 billion parameter model within the Gemma family, fully trained with differential privacy. Pretrained on the identical data mixture used for the Gemma 2 series, VaultGemma 1B represents a significant step forward in privacy-preserving large language models. We openly release this model to the community.

- 1B parameter LLM, trained on 13T tokens on 2048 TPUv6e chips
- DP-SGD to achieve $(\epsilon = 2, \delta = 1.1 \cdot 10^{-10})$

Example: VaultGemma



Performance comparison of VaultGemma 1B (differentially private) against its non-private counterpart (Gemma3 1B) and an older baseline (GPT-2 1.5B). The results quantify the current resource investment required for privacy and demonstrate that modern DP training yields utility comparable to non-private models from roughly five years ago.

Source: Kalinin *et al.*, https://arxiv.org/abs/2601.22334 https://research.google/blog/vaultgemma-the-worlds-most-capable-differentially-private-llm/

Current Trends

Hybrid Approaches

- Pretrain on public, non-sensitive data sources
- Finetune with privacy on protected data

Advantage: few steps of training needed, more privacy preserved

Problem: public data can still contain sensitive information

Hybrid Approaches

- Pretrain on public, non-sensitive data sources
- Finetune with privacy on protected data

Advantage: few steps of training needed, more privacy preserved

Problem: public data can still contain sensitive information

Synthetic Data Generation

- Every time you train on the same data, privacy is lost.
  - no real-world public benchmarks
  - no proper hyperparameter or model selection
- Idea: run private algorithm to privately create a new dataset,

Advantage: private created datasets can be re-used as often as wanted

Problem: data generation is hard, private data generation even more

Formalizing Unlearning

Setup:

- learning algorithm $\mathcal{A}$
- training set $\mathcal{D}$
- unlearning method $\mathcal{U}$
- set to be forgotten $S \subset \mathcal{D}$

$\mathcal{U}$ achieves ϵ-unlearning if, for all:

$$\forall O: \quad e^{-\epsilon} \Pr\{\mathcal{A}(\mathcal{D} \setminus S) \in O\} \leq \Pr\{\mathcal{U}(\mathcal{A}(\mathcal{D}), S) \in O\} \leq e^{\epsilon} \Pr\{\mathcal{A}(\mathcal{D} \setminus S) \in O\}$$

(and (ϵ, δ) analogously).

Observation: if $\mathcal{A}$ is sufficiently private, $\mathcal{U}$ has to do nothing

A lot of the data out there is private/protected.

- personal communications, internal notes, browsing history, ...
- medical/genetic, military, business secrets, ...

Some should stay locked away, but others could make machine learning models truly better.

Differential Privacy is a mature research field in (Theoretical) Computer Science

- 1) limit the influence of individual data items,
- 2) add noise to the processing to hide which data was used.

Private machine learning still has a privacy/utility gap

- enforcing differential privacy makes models worse, which hinders adoption
- do we need relaxed notions of privacy? hybrids?

slides