Private Stochastic Convex Optimization

Kunal Talwar
Apple

Joint works with:
Vitaly Feldman, Tomer Koren. STOC 2020
Hilal Asi, Vitaly Feldman, Tomer Koren. Arxiv 2021

Stochastic Convex Optimization

Unknown distribution (population) $\mathscr{D}$ over data universe Z

Convex Parameter Space $K \subseteq \mathbb{R}^d$

Convex loss function $\ell : K \times Z \to \mathbb{R}$

Dataset $S = \{z_1, z_2, \ldots, z_n\} \sim \mathscr{D}^n$

Stochastic Convex Optimization

Unknown distribution (population) $\mathscr{D}$ over data universe Z

Convex Parameter Space $K \subseteq \mathbb{R}^d$

Convex loss function $\ell : K \times Z \to \mathbb{R}$

Dataset $S = \{z_1, z_2, \ldots, z_n\} \sim \mathscr{D}^n$

An SCO algorithm, given S, outputs a $\hat{\theta} \in K$ s.t.

Excess Population Risk $= \mathbb{E}_{z \in \mathscr{D}}[\ell(\hat{\theta}, z)] - \min_{\theta \in K} \mathbb{E}_{z \in \mathscr{D}}[\ell(\theta, z)]$

is as small as possible

Stochastic Convex Optimization

Unknown distribution (population) $\mathscr{D}$ over data universe Z

Convex Parameter Space $K \subseteq \mathbb{R}^d$

Convex loss function $\ell : K \times Z \to \mathbb{R}$

Dataset $S = \{z_1, z_2, \ldots, z_n\} \sim \mathscr{D}^n$

An SCO algorithm, given S, outputs a $\hat{\theta} \in K$ s.t.

Excess Population Risk $= \mathbb{E}_{z \in \mathscr{D}}[\ell(\hat{\theta}, z)] - \min_{\theta \in K} \mathbb{E}_{z \in \mathscr{D}}[\ell(\theta, z)]$

is as small as possible

ℓ_2/ℓ_2 case
$K =$ unit ℓ_2 ball
$\|\nabla_\theta \ell(\theta, z)\| \leq 1$

Stochastic Convex Optimization

Unknown distribution (population) $\mathscr{D}$ over data universe Z

Convex Parameter Space $K \subseteq \mathbb{R}^d$

Convex loss function $\ell : K \times Z \to \mathbb{R}$

Dataset $S = \{z_1, z_2, \ldots, z_n\} \sim \mathscr{D}^n$

An SCO algorithm, given S, outputs a $\hat{\theta} \in K$ s.t.

Excess Population Risk $= \mathbb{E}_{z \in \mathscr{D}}[\ell(\hat{\theta}, z)] - \min_{\theta \in K} \mathbb{E}_{z \in \mathscr{D}}[\ell(\theta, z)]$

is as small as possible

ℓ_2/ℓ_2 case
$K =$ unit ℓ_2 ball
$\|\nabla_\theta \ell(\theta, z)\| \leq 1$

Optimal Excess Risk $\frac{1}{\sqrt{n}}$

Private Stochastic Convex Optimization

Unknown distribution (population) $\mathscr{D}$ over data universe Z

Convex Parameter Space $K \subseteq \mathbb{R}^d$

Convex loss function $\ell : K \times Z \to \mathbb{R}$

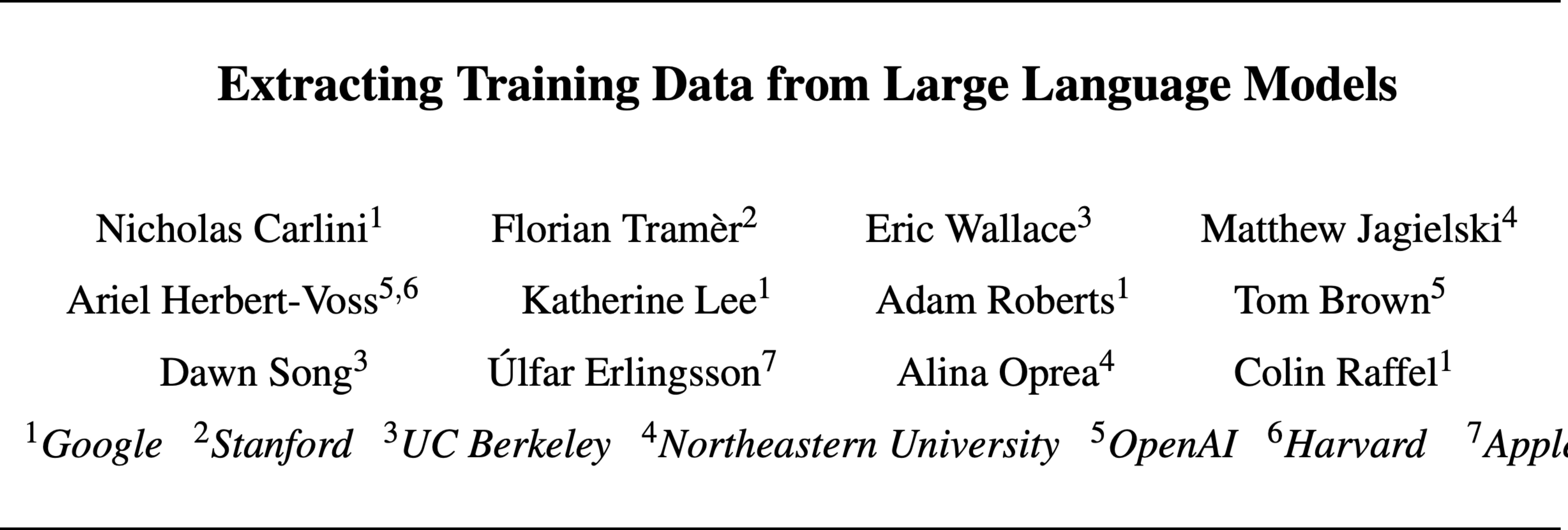
Dataset $S = \{z_1, z_2, \ldots, z_n\} \sim \mathscr{D}^n$

An (ε, δ)-DP SCO algorithm, given S, outputs a $\hat{\theta} \in K$ s.t.

Excess Population Risk $= \mathbb{E}_{z \in \mathscr{D}}[\ell(\hat{\theta}, z)] - \min_{\theta \in K} \mathbb{E}_{z \in \mathscr{D}}[\ell(\theta, z)]$

is as small as possible

Memorization Risk is real



Extracting Training Data from Large Language Models

Nicholas Carlini[1] Florian Tramèr[2] Eric Wallace[3] Matthew Jagielski[4]
Ariel Herbert-Voss[5,6] Katherine Lee[1] Adam Roberts[1] Tom Brown[5]
Dawn Song[3] Úlfar Erlingsson[7] Alina Oprea[4] Colin Raffel[1]
[1]*Google* [2]*Stanford* [3]*UC Berkeley* [4]*Northeastern University* [5]*OpenAI* [6]*Harvard* [7]*Apple*

- Extracted contact details of Peter W
  - Exist only 6 times in Google search results

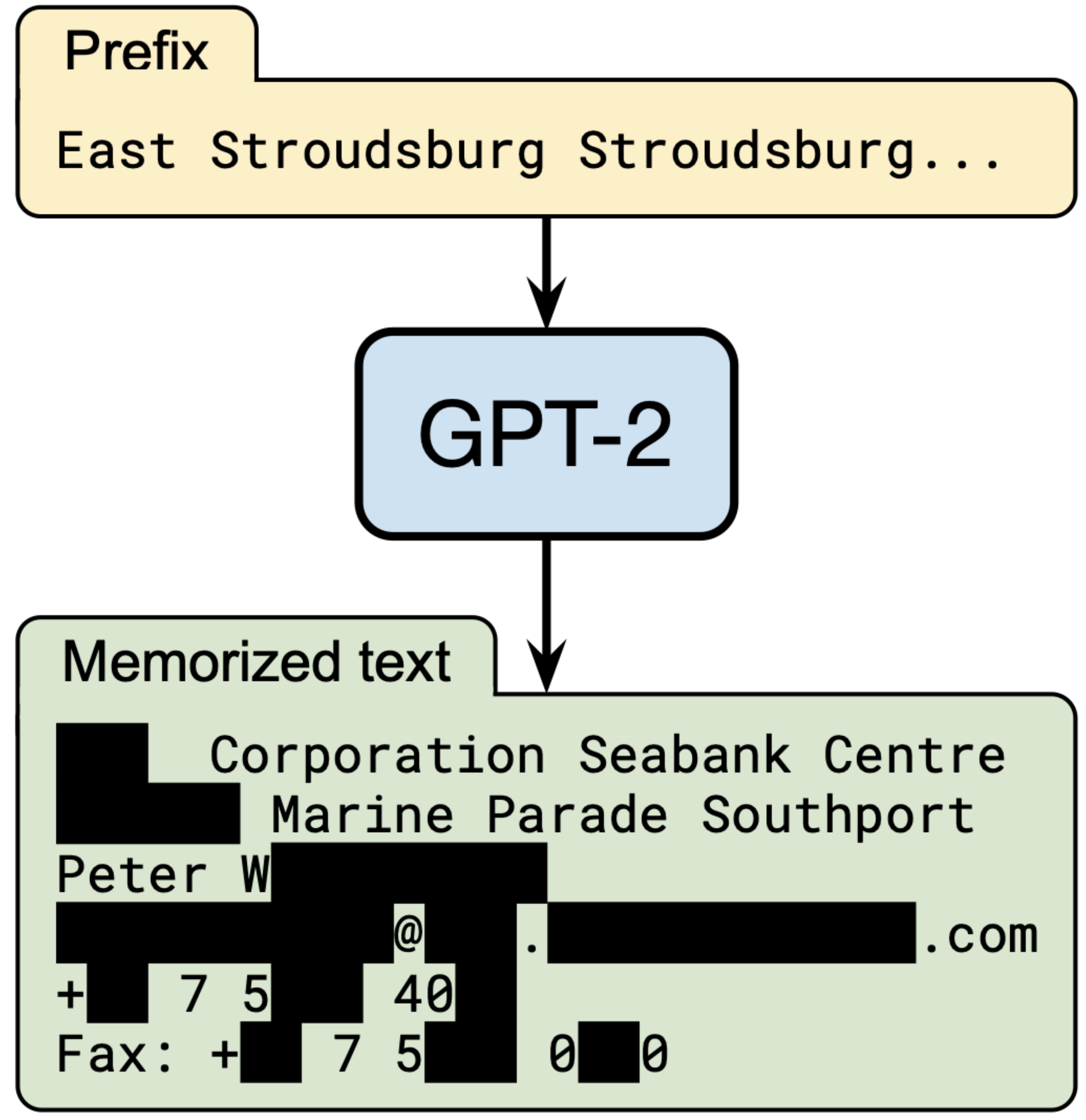

Prefix
East Stroudsburg Stroudsburg...
GPT-2
Memorized text
[redacted] Corporation Seabank Centre
Marine Parade Southport
Peter W[redacted]
[redacted]@[redacted].[redacted].com
+[redacted] 7 5[redacted] 40[redacted]
Fax: +[redacted] 7 5[redacted] 0[redacted]0

Figure 1: **Our extraction attack.** Given query access to a neural network language model, we extract an individual person's name, email address, phone number, fax number, and physical address. The example in this figure shows information that is all accurate so we redact it to protect privacy.

Private SCO vs Private ERM

Closely related problem: Private Empirical Risk Minimization

Given S, outputs a $\hat{\theta} \in K$ s.t.

Excess Empirical Risk $= \mathbb{E}_{z \in S}[\ell(\hat{\theta}, z)] - \min_{\theta \in K} \mathbb{E}_{z \in S}[\ell(\theta, z)]$

is as small as possible

Private SCO vs Private ERM

Closely related problem: Private Empirical Risk Minimization

Given S, outputs a $\hat{\theta} \in K$ s.t.

Excess Empirical Risk $= \mathbb{E}_{z \in S}[\ell(\hat{\theta}, z)] - \min_{\theta \in K} \mathbb{E}_{z \in S}[\ell(\theta, z)]$

is as small as possible

[CM08, CMS11, JKT12, KST12, ST13, SCS13, DJW13, Ull15, JT14, BST14, TTZ15, STU17, WLK+17, WYX17, INS+19]

Well-understood. Optimal excess privacy risk is $\dfrac{\sqrt{d}}{\varepsilon n}$

Private SCO

Two natural lower bounds:

- SCO lower bound: $\frac{1}{\sqrt{n}}$
- Private ERM lower bound : $\frac{\sqrt{d}}{\varepsilon n}$

Private SCO

Two natural lower bounds:

- SCO lower bound: $\frac{1}{\sqrt{n}}$
- Private ERM lower bound : $\frac{\sqrt{d}}{\varepsilon n}$

[BFTT19] Optimal upper bound of $\max(\frac{1}{\sqrt{n}}, \frac{\sqrt{d}}{\varepsilon n})$

Private SCO

Two natural lower bounds:

- SCO lower bound: $\frac{1}{\sqrt{n}}$
- Private ERM lower bound : $\frac{\sqrt{d}}{\varepsilon n}$

[BFTT19] Optimal upper bound of $\max(\frac{1}{\sqrt{n}}, \frac{\sqrt{d}}{\varepsilon n})$

Suppose $d \leq n, \varepsilon$ constant
This bound is $\frac{1}{\sqrt{n}}$
Privacy for free.

Private SCO

Two natural lower bounds:

- SCO lower bound: $\frac{1}{\sqrt{n}}$
- Private ERM lower bound : $\frac{\sqrt{d}}{\varepsilon n}$

Suppose $d \leq n, \varepsilon$ constant
This bound is $\frac{1}{\sqrt{n}}$
Privacy for free.

[BFTT19] Optimal upper bound of $\max(\frac{1}{\sqrt{n}}, \frac{\sqrt{d}}{\varepsilon n})$

[FKT20] This optimal bound in linear time

No computational overhead

General techniques for generalization

$\frac{\sqrt{d}}{\sqrt{n}} + \frac{\sqrt{d}}{\varepsilon n}$

- Uniform Convergence: $\forall \theta, \mathbb{E}_{z \in S}[\ell(\theta, z)]$ close to $\mathbb{E}_{z \in \mathcal{D}}[\ell(\theta, z)]$
- Online-to-batch: Regret bounds for Single pass algorithms generalize
- Uniform Stability: Optimization algorithm satisfies $\theta(S)$ close to $\theta(S')$
- Differential Privacy: Distributional form of stability. Excess loss $\leq \varepsilon$

SGD

Input: Number of steps T, Sequence of step sizes $\{\eta_t\}_{t\in[T]}$

```
1. Initialize $\theta_0 = \mathbf{0}$
2. For $t = 0,\ldots,(T-1)$:
    1. Compute Gradient $g_t = \nabla_\theta \ell(\theta_t, z_t)$
    2. Update $\theta_{t+1} \leftarrow \Pi_K(\theta_t - \eta_t \cdot g_t)$
Return the final iterate $\theta_T$
```

Noisy SGD

Input: Number of steps T, Sequence of step sizes $\{\eta_t\}_{t\in[T]}$, Noise scale σ

```
1. Initialize $\theta_0 = \mathbf{0}$
2. For $t = 0,\ldots,(T-1)$:
    1. Compute Gradient $\widehat{g_t} = \nabla_\theta \ell(\theta_t, z_t) + \mathcal{N}(0, \sigma^2 \mathbb{I}_d)$
    2. Update $\theta_{t+1} \leftarrow \Pi_K(\theta_t - \eta_t \cdot \widehat{g_t})$
Return the final iterate $\theta_T$
```

Scaled to Sensitivity

Noisy SGD

Online Convex Optimization Bounds:

For $\eta_t = \frac{1}{\sqrt{T}}$, excess loss is bounded by $\frac{\sigma\sqrt{d}}{\sqrt{T}}$

$\sigma = 1 \Rightarrow$ large overhead. Can be overcome with larger runtime

Noisy Batched SGD

Input: Number of steps T, Sequence of step sizes $\{\eta_t\}_{t\in[T]}$, Noise scale σ

Batching reduces sensitiivity

```
1. Initialize $\theta_0 = \mathbf{0}$
2. For $t = 0,\ldots,(T-1)$:
    1. Compute Gradient $\widehat{g_t} = \frac{1}{B_t}\sum_{z\in B_t} \nabla_\theta \ell(\theta_t, z) + \mathcal{N}(0, \frac{\sigma^2}{B_t^2}\mathbb{I}_d)$
    2. Update $\theta_{t+1} \leftarrow \Pi_K(\theta_t - \eta_t \cdot \widehat{g_t})$
Return the final iterate $\theta_T$
```

Noisy SGD

Online Convex Optimization Bounds:

For $\eta_t = \frac{1}{\sqrt{T}}$, excess loss is bounded by $\frac{\sigma\sqrt{d}}{\sqrt{T}}$

$\sigma = 1 \Rightarrow$ large overhead. Can be overcome with larger runtime

Set $B_t = \sigma_t^{-1} = \sqrt{d}$. Excess risk is $\frac{1}{\sqrt{T}}$

Noisy SGD

Online Convex Optimization Bounds:

For $\eta_t = \frac{1}{\sqrt{T}}$, excess loss is bounded by $\frac{\sigma\sqrt{d}}{\sqrt{T}}$

$\sigma = 1 \Rightarrow$ large overhead. Can be overcome with larger runtime

Set $B_t = \sigma_t^{-1} = \sqrt{d}$. Excess risk is $\frac{1}{\sqrt{T}}$

Pay in sample complexity. Or in run time.

Noisy Batched SGD

```
Input: Number of steps $T$, Sequence of step sizes $\{\eta_t\}_{t\in[T]}$, Noise scale $\sigma$

1. Initialize $\theta_0 = \mathbf{0}$
2. For $t = 0,\ldots,(T-1)$:
    1. Compute Gradient $\widehat{g_t} = \frac{1}{B_t}\sum_{z\in B_t} \nabla_\theta \ell(\theta_t, z) + \mathcal{N}(0, \frac{\sigma^2}{B_t^2}\mathbb{I}_d)$
    2. Update $\theta_{t+1} \leftarrow \Pi_K(\theta_t - \eta_t \cdot \widehat{g_t})$
Return the final iterate $\theta_T$
```

Private SCO

Two new Algorithms

- Snowball SGD
- Iterative Localization

Privacy Amplification by Iteration

[FMTT18]: Under mild smoothness assumptions:

Example used in B_t gets privacy from noise added in subsequent iterations.

Privacy Amplification by Iteration

[FMTT18]: Under mild smoothness assumptions:

Example used in B_t gets privacy from noise added in subsequent iterations.

Non-uniform privacy guarantee.

No improvement for privacy of example in last batch.

Snowball SGD

Noisy Batched SGD with:

$$\sigma = \frac{1}{\sqrt{d}}$$

$$B_T = \lceil \sqrt{d} \rceil, B_{T-1} = \lceil \sqrt{d/2} \rceil, \ldots, B_{T-s} = \lceil \sqrt{d/s} \rceil, \ldots,$$

Snowball SGD

Noisy Batched SGD with:

$$\sigma = \frac{1}{\sqrt{d}}$$

$$B_T = \lceil \sqrt{d} \rceil, B_{T-1} = \lceil \sqrt{d/2} \rceil, \ldots, B_{T-s} = \lceil \sqrt{d/s} \rceil, \ldots,$$

For an example in batch $(T - s)$, sensitivity is $\dfrac{1}{B_{T-s}} = \sqrt{\dfrac{s}{d}}$.

Snowball SGD

Noisy Batched SGD with:

$$\sigma = \frac{1}{\sqrt{d}}$$

$$B_T = \lceil \sqrt{d} \rceil, B_{T-1} = \lceil \sqrt{d/2} \rceil, \ldots, B_{T-s} = \lceil \sqrt{d/s} \rceil, \ldots,$$

For an example in batch $(T - s)$, sensitivity is $\dfrac{1}{B_{T-s}} = \sqrt{\dfrac{s}{d}}$.

Noise = Sum of s $\mathcal{N}(0, \sigma^2)$ noises: $\sigma\sqrt{s} = \sqrt{\dfrac{s}{d}}$

Snowball SGD

Number of examples used:

$$\sum_s B_{T-s} = \sum_s \lceil \sqrt{d/s} \rceil \leq d + T$$

Gives linear (1-pass) algorithm with optimal rate.

Snowball SGD

Number of examples used:

$$\sum_s B_{T-s} = \sum_s \lceil \sqrt{d/s} \rceil \leq d + T$$

Gives linear (1-pass) algorithm with optimal rate.

No privacy amplification for iterate averaging

Need to use last iterate bounds [SZ13, JNN19]

Private SCO

Two new Algorithms

- Snowball SGD
- Iterative Localization

Iterative Localization

More precise bound for SGD:

Excess loss, Sensitivity $\leq \dfrac{\|\theta^\star - \theta_0\|}{\sqrt{T}}$

Iterative Localization

More precise bound for SGD:

Excess loss, Sensitivity $\leq \dfrac{\|\theta^\star - \theta_0\|}{\sqrt{T}}$

Optimum localized somewhat. Repeat with better upper bound on $\|\theta^\star - \theta_0\|$.

Iterative localize. Get rid of $\sqrt{d}$ overhead in $\log d$ phases.

Iterative Localization

Optimum localized somewhat. Repeat with better upper bound on $\|\theta^\star - \theta_0\|$.

Iteratively localize. Get rid of $\sqrt{d}$ overhead in $\log d$ phases.

Can use any stable ERM instead of SGD, and add noise for privacy.

Blackbox, more general.

Other Results

Strongly Convex case:

- Can achieve optimal rate of $\max(\frac{1}{n}, \frac{d}{\varepsilon^2 n^2})$ in linear time
- New last iterate analysis for constant step size

Non-smooth losses:

- Optimal rate unchanged
- Run time n^2

Other Geometries

- The ℓ_1/ℓ_∞ case : Optimizing over the ℓ_1 ball. Gradients bounded in ℓ_∞

- [TTZ15] Private ERM under smoothness: $\left(\frac{\log d}{\varepsilon n}\right)^{2/3}$

- Non-private SCO: $\sqrt{\frac{\log d}{n}}$

The LASSO setting

- The ℓ_1/ℓ_∞ case : Optimizing over the ℓ_1 ball. Gradients bounded in ℓ_∞

- [AFKT21] Smooth case: New Variance-reduced Frank-Wolfe achieving sum of the two bounds: $\sqrt{\frac{\log d}{n}} + \left(\frac{\log d}{\varepsilon n}\right)^{2/3}$. Tight. Linear time.

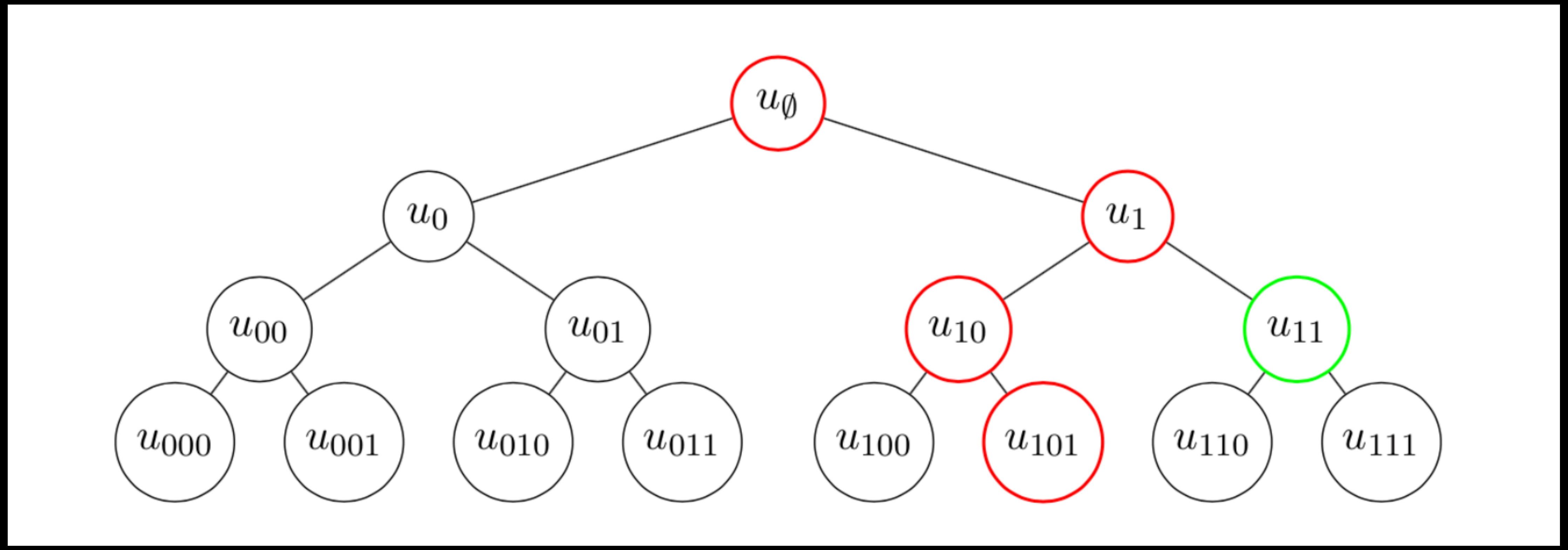
- [AFKT21] Non-smooth case: Iterative localization + regularized mirror descent gives $\sqrt{\frac{\log d}{n}} + \frac{\sqrt{d}}{\varepsilon n}$. Tight. Extends to $\ell_p, p \in [1,2]$

Variance Reduction reduces Sensitivity

- In SGD: noise added to gradient proportional to gradient norm
- Variance reduction: $\hat{g}(\theta_t) = \hat{g}(\theta_{anchor}) + \left(g_t(\theta_t) - g_t(\theta_{anchor})\right)$
- Smooth loss $\Rightarrow \|g_t(\theta_t) - g_t(\theta_{anchor})\| \leq \|\theta_t - \theta_{anchor}\|$
- If e.g. $\theta_{anchor} = \theta_{t-1},$ sensitivity reduced by a factor of $\eta \approx \frac{1}{\sqrt{T}}$
- Thus a lot less noise needed to ensure privacy!

Dyadic Variance Reduction

- Need to handle reuse of the anchor : higher privacy cost for those?
- Build binary tree.
  - Use path from root as anchor points
  - Batching to reduce privacy cost for internal nodes.
  - Node with 2^i children has batch size 2^i

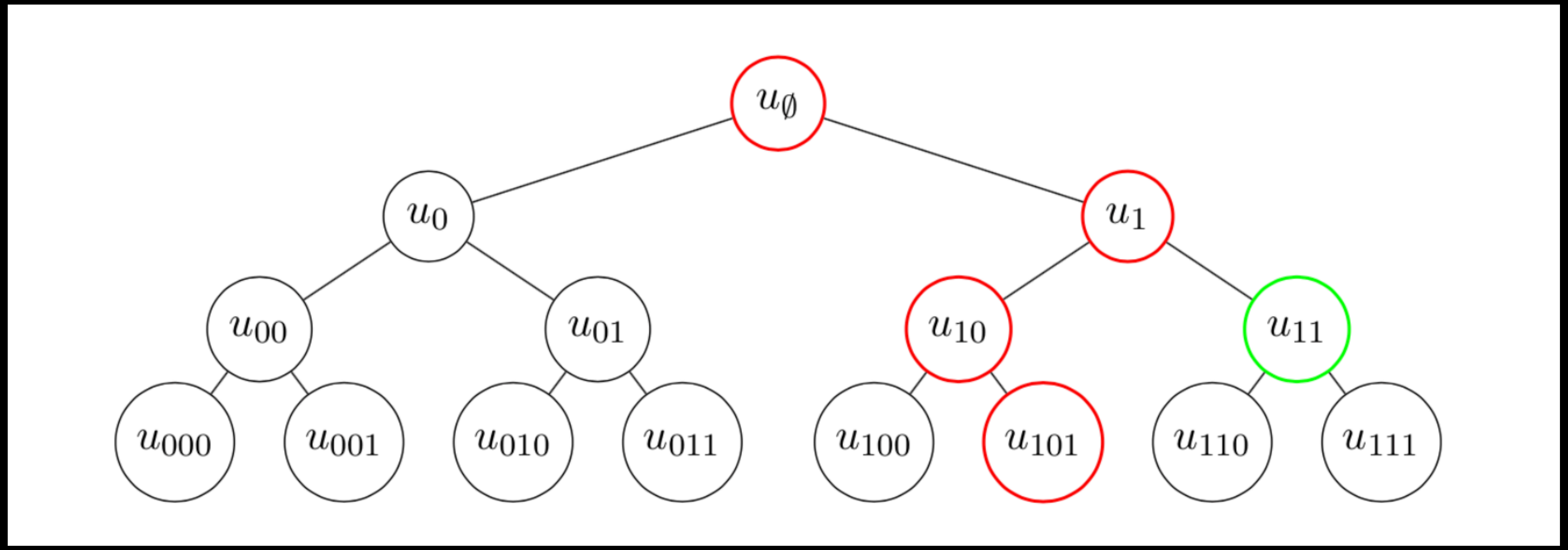


Frank-Wolfe

- Frank-Wolfe: Reduce Smooth Convex Optimization to Linear Optimization
  - Average of T linear optimizers gives error $\propto \frac{1}{T}$
- Linear optimizer lies at a vertex. Select from 2d vertices of the ℓ_1 ball
  - Can privately select using the Exponential Mechanism
  - Overhead $\log(2d)/\varepsilon$ instead of the $\sqrt{d}/\varepsilon$ from Gaussian mechanism
- Full-batch Frank-Wolfe
  - Privacy overhead increases as $\sqrt{T}$
  - Balance with $\frac{1}{T}$ error of FW. Gives $n^{-2/3}$ for ERM.

Stochastic Variance Reduced FW

- [YSC19] Variance reduced FW generalizes.
    - Overuses samples. Not private.
- Use Dyadic Variance-Reduced Gradients
- Run FW with these stochastic gradients
- Exponential Mechanism to select vertex



Summary

- Private Stochastic Optimization
    - ℓ_2/ℓ_2 case: optimal rates
        - Linear time for smooth losses
        - $n^{3/2}$ time for non-smooth case (see also [KLL21])
    - ℓ_1/ℓ_∞ case
        - Non-smooth case: optimal rates
        - Smooth case: linear time FW. Dimension independent!!
- New techniques: Iterative Localization, Heterogenous Batching, Dyadic Variance Reduction