CS208: Applied Privacy for Data Science
Machine Learning & Optimization under DP: Theory

James Honaker, Priyanka Nanayakkara, Salil Vadhan
School of Engineering & Applied Sciences
Harvard University

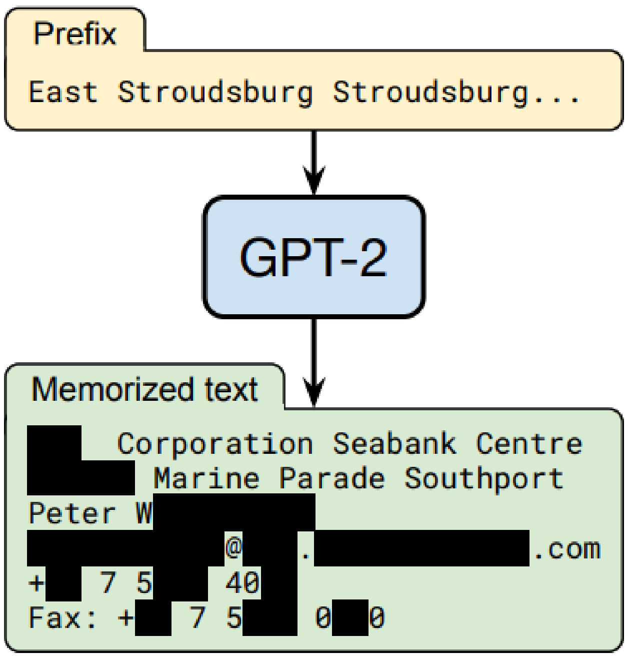
March 26, 2025

More Responses to Midterm Feedback

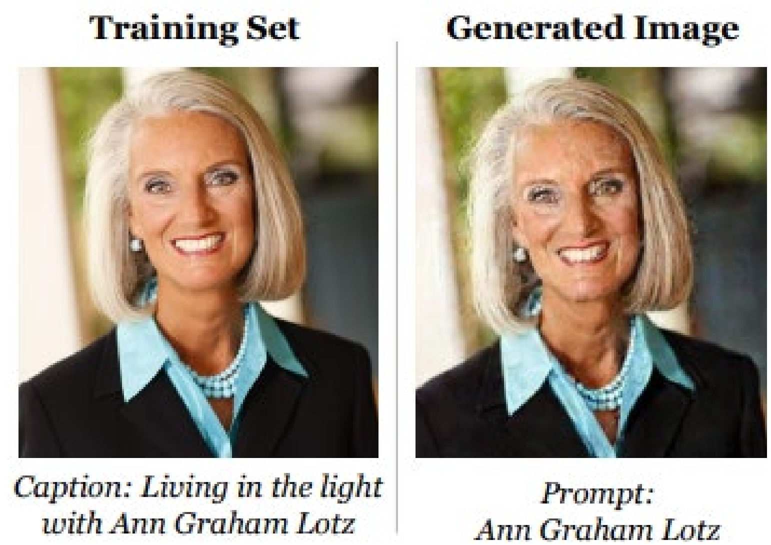
(see also 3/12 slides)

- Median time spent on readings: 1.75hrs
- Median time spent on psets: 8hrs
  - Come discuss if you're regularly spending 12+hrs
  - Should be lower in the rest of the course (to leave time for project work)
- Discussions:
  - Most enjoying, depends on who you are with
  - Request for more technical discussions, more TA involvement
- Section times inconvenient, need more OH
  - Have added Tue eve section, more OH
- Pset solutions should now have been released for past psets.

Why ML with DP?
ML models memorize training data



[Carlini, Tramèr, Wallace et al. 2021]



[Carlini, Hayes, Nasr et al. 2023]

ML Inputs and Loss Functions

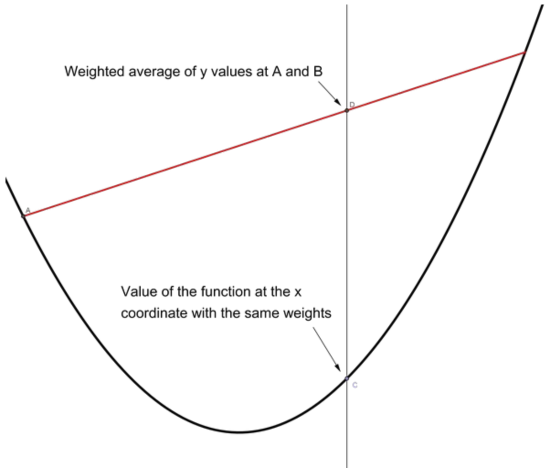
- Data: $(x_1, y_1), \dots, (x_n, y_n) \sim \mathcal{P}$
  - Examples $x_i \in \mathcal{X}$: d-dimensional, discrete or continuous
  - Labels $y_i \in \mathcal{Y}$: 1-dimensional, discrete or continuous
  - $\mathcal{P}$ typically unknown
- A loss function:
  - $\ell : \Theta \times \mathcal{X} \times \mathcal{Y} \rightarrow \mathbb{R}$ $\quad \ell(\theta | x_i, y_i)$ measures ``loss"
  - Define $L: \Theta \rightarrow \mathbb{R}$ $\quad L(\theta) = \frac{1}{n} \sum_{i=1}^{n} \ell(\theta | x_i, y_i)$
  - E.g. squared loss $\ell(\theta | x_i, y_i) = |(\theta_1 x_i + \theta_0) - y_i|^2$.
- Goal: output $\hat{\theta} \in \Theta$ s.t.

$$L(\hat{\theta}) \approx \min L(\theta)$$

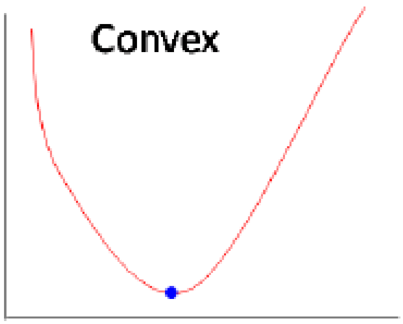
Convexity

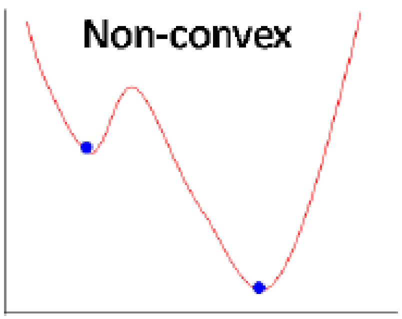
- Def: L is convex if for all points $\vec{a}, \vec{b}$, we have

$$L\left(\frac{\vec{a}+\vec{b}}{2}\right) \leq \frac{L(\vec{a})+L\left(\vec{b}\right)}{2}.$$



- Convex functions have no local minima



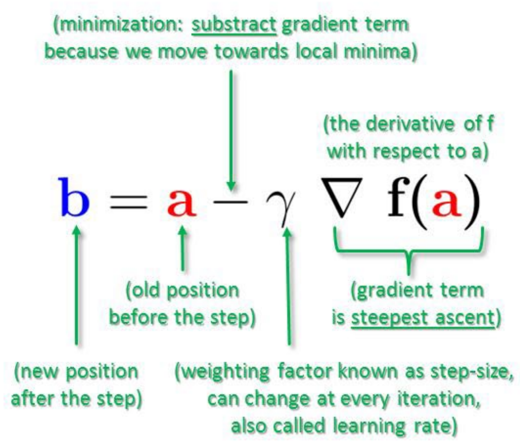


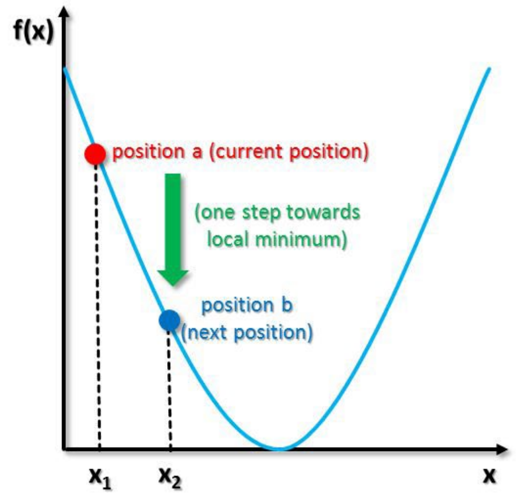
- Loss function for logistic regression is convex
  - No closed form solution for minimum, but it is easy to find

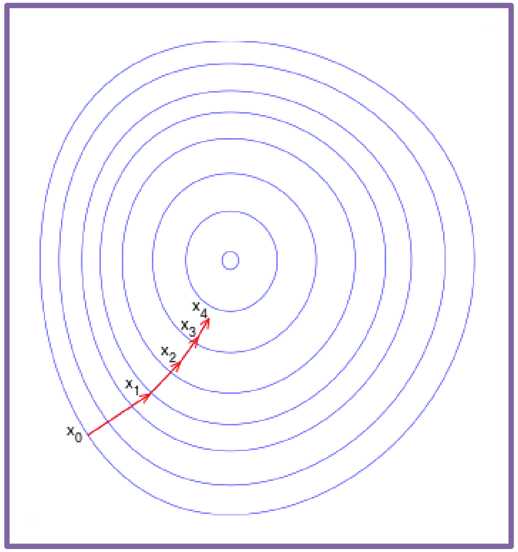
[slide modified from Adam Smith, BU CS 591 Fall 2018]

Gradient Descent

- Proceed in steps
- Start from (carefully chosen) initial parameters $\hat{\theta}_0$
- At each step, move in direction opposite to the gradient of the loss $\nabla L(\hat{\theta})$.
- Gradient is the vector of partial derivatives







[slide modified from Adam Smith, BU CS 591 Fall 2018]

Gradient Descent

- Specify
  - Number of steps T
  - Learning rate η
- Pick initial point $\hat{\theta}_0 \in \Theta$
- For $t = 1$ to T
  - Compute gradient

$$g_t = \nabla L(\hat{\theta}_{t-1}) = \frac{1}{n} \sum_i \nabla \ell(\hat{\theta}_{t-1} | x_i, y_i)$$

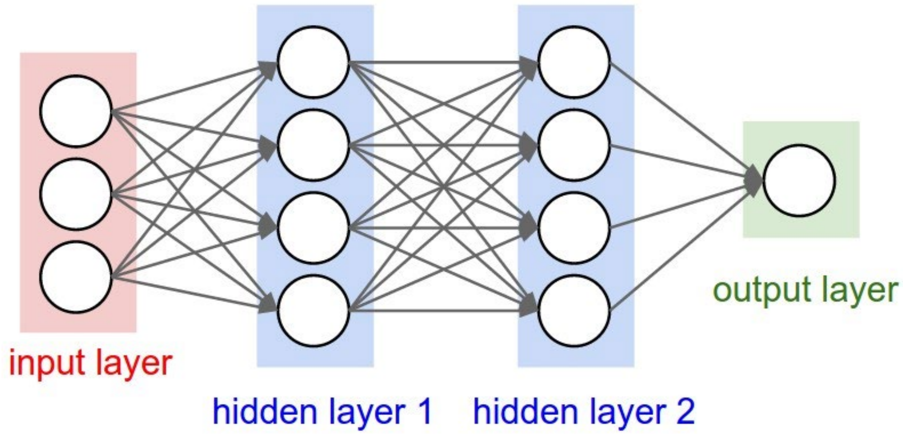
  - $\hat{\theta}_t = \hat{\theta}_{t-1} - \eta \cdot g_t$
- Output $\hat{\theta} = \sum_{t=1}^{T} \hat{\theta}_t$ or $\hat{\theta}_T$

Average iterate

Last iterate

[slide modified from Adam Smith, BU CS 591 Fall 2018]

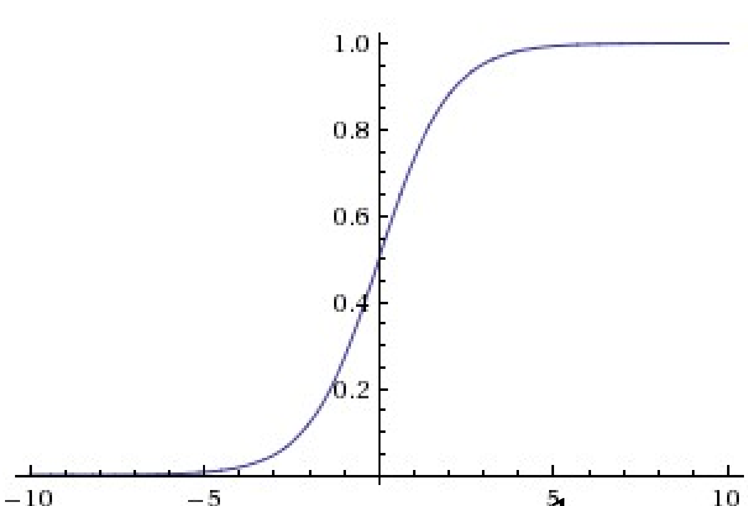
Gradient Descent for Neural Networks



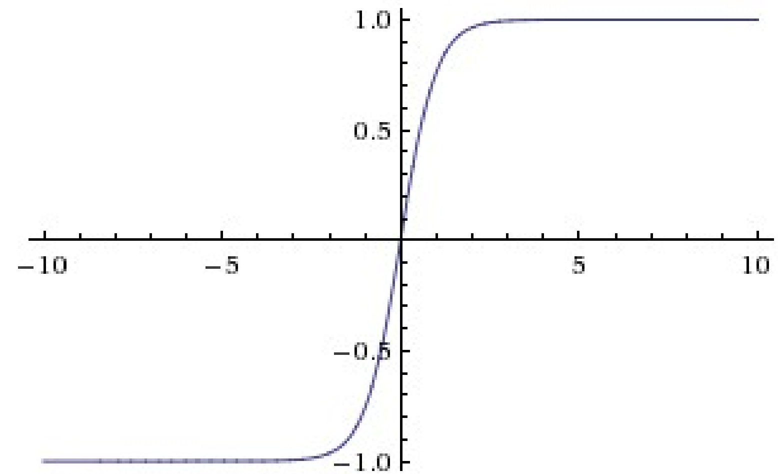
- Each node is a linear function of inputs (specified by θ) composed with a nonlinear “activation” function
- Gradient of Loss function can be computed quickly
  - Using chain rule (technique called “backpropagation”)
- But no longer convex, has many local minima
  - Can get stuck in a bad place
  - But works well in practice!

[slide modified from Adam Smith, BU CS 591 Fall 2018]

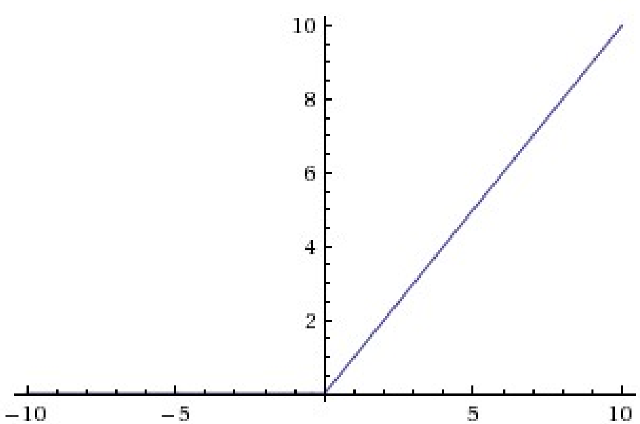
Common Activation Functions



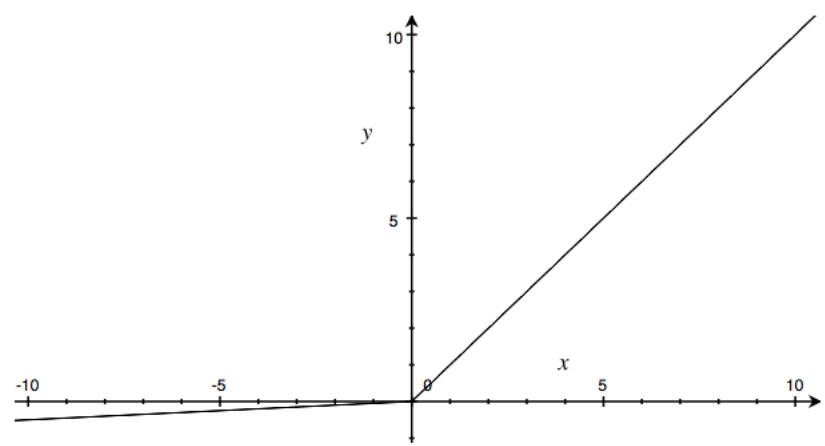
Sigmoid $\sigma(x) = \frac{1}{1+e^{-x}}$



$\tanh(x) = 2\sigma(2x) - 1$



$\text{ReLU}(x) = \max(0, x)$



$\text{Leaky ReLU}(x) = \max(0.05x, x)$

[slide modified from Adam Smith, BU CS 591 Fall 2018]

DP for Vector-Valued Functions

- Let $f : \mathcal{X}^n \to \mathbb{R}^k$, and $M(x) = f(x) + Z$ for noise $Z \in \mathbb{R}^k$.
- global ℓ_2-sensitivity of f is

$$\mathrm{GS}_{f,\ell_2} \stackrel{\text{def}}{=} \max_{x \sim x'} \|f(x) - f(x')\|_2.$$

$$\|z\|_2 = \left(\sum_j \left|z_j\right|^2\right)^{1/2}$$

- Gaussian Mechanism: $Z \sim \mathcal{N}\left(\vec{0}, 2\left(\frac{\mathrm{GS}_{f,\ell_2}}{\varepsilon}\right)^2 \cdot \ln\frac{1.25}{\delta} \cdot I_k\right)$
  - independent Gaussian noise per coordinate.

Robustness to Noise in Gradient Estimation

- For efficiency:

  Sample a minibatch $B \subseteq \{1, 2, \ldots, n\}$

  Gradient estimate $\tilde{g}_t = \frac{1}{|B|} \sum_{i \in B} \nabla \ell(\hat{\theta}_{t-1}, x_i, y_i)$

  Stochastic Gradient Descent (SGD)!

- For privacy:

  Add Gaussian Noise $\tilde{g}_t = g_t + \mathcal{N}(0, \sigma^2 I)$

In both cases, $\tilde{g}_t$ is an unbiased estimate of g_t: $E[\tilde{g}_t] = g_t$

DP Gradient Descent

[Williams-McSherry`10, …]

- Specify
  - Number of steps T
  - Learning rate η
  - Privacy parameters ε, δ
  - Clipping parameter C. Write $[\vec{z}]_C = \vec{z} \cdot \max\left(1, \frac{C}{\|\vec{z}\|_2}\right)$.
  - Noise variance $\sigma^2 = \mathrm{TBD}(T, \varepsilon, \delta, C)$.
- Pick initial point $\hat{\theta}_0$
- For $t = 1$ to T
  - Estimate gradient as noisy average of clipped gradients
    $\hat{g}_t = \frac{1}{n} \sum_i \left[\nabla \ell\left(\hat{\theta}_{t-1} | x_i, y_i\right)\right]_C + \mathcal{N}(0, \sigma^2 I)$
  - $\hat{\theta}_t = \hat{\theta}_{t-1} - \eta \cdot \hat{g}_t$
- Output $\hat{\theta} = \sum_{t=1}^{T} \hat{\theta}_t$ or $\hat{\theta}_T$

[slide modified from Adam Smith, BU CS 591 Fall 2018]

Privacy Analysis

- Proof idea: Show releasing $(\hat{g}_1, \hat{g}_2, \ldots, \hat{g}_T)$ satisfies DP
  - Each step (releasing $\hat{g}_t$) satisfies (ϵ, δ)-DP
  - Adaptive composition across T steps

Privacy Analysis

- By Gaussian Mechanism, each iteration is $(\varepsilon_0, \delta_0)$-DP if

$$\sigma^2 = 2\left(\frac{C}{\varepsilon_0 n}\right)^2 \cdot \ln\frac{1.25}{\delta_0}$$

- By Advanced Composition, overall algorithm is (ε, δ)-DP for:

$$\varepsilon = O\left(\varepsilon_0 \cdot \sqrt{T \ln(2/\delta)}\right)$$
$$\delta = 2T \cdot \delta_0$$

- Solving, suffices to use noise variance

$$\sigma^2 = O\left(\left(\frac{C}{\varepsilon n}\right)^2 \cdot T \cdot \ln\frac{T}{\delta} \cdot \ln\frac{1}{\delta}\right)$$

[slide modified from Adam Smith, BU CS 591 Fall 2018]

Improved Analysis with "Concentrated DP"

[Dwork-Rothblum `16, Bun-Steinke `16]

- By Gaussian Mechanism, each iteration is ε_0^2 -zCDP if

$$\sigma^2 = \frac{1}{2}\left(\frac{C}{\varepsilon_0 n}\right)^2 \cdot \cancel{\ln \frac{1.25}{\delta_0}}$$

- By composition of zCDP, overall algorithm is $T \cdot \varepsilon_0^2$-zCDP.
- By zCDP-to-approx. DP conversion, overall algorithm is (ε, δ)-DP for:

$$\varepsilon = T \cdot \varepsilon_0^2 + 2\sqrt{T \cdot \varepsilon_0^2 \cdot \ln(1/\delta)}$$

- Solving, suffices to use noise variance

$$\sigma^2 = O\left(\left(\frac{C}{\varepsilon n}\right)^2 \cdot T \cdot \ln\frac{1}{\delta} \cdot \cancel{\ln\frac{T}{\delta}}\right)$$

[slide modified from Adam Smith, BU CS 591 Fall 2018]

DP Stochastic Gradient Descent (SGD)

[Jain-Kothari-Thakurta `12, Song-Chaudhuri-Sarwate `13, Bassily-Smith-Thakurta `14]

- Specify
  - Number of steps T, learning rate η, privacy parameters ε, δ, clipping parameter C.
  - Batch size $B \ll n$ (for efficiency)
  - Noise variance $\sigma^2 = \mathrm{TBD}(T, \varepsilon, \delta, \mathrm{C}, B)$.
- Pick initial point $\hat{\theta}_0$
- For $t = 1$ to T
  - Select a random batch $S_t \subseteq \{1, \ldots, n\}$ of size B.
  - Estimate gradient as noisy average of clipped gradients $\hat{g}_t = \frac{1}{B} \sum_{i \in S_t} \left[\nabla \ell\left(\hat{\theta}_{t-1} | x_i, y_i\right)\right]_C + \mathcal{N}(0, \sigma^2 I)$
  - $\hat{\theta}_t = \hat{\theta}_{t-1} - \eta \cdot \hat{g}_t$
- Output $\hat{\theta} = \sum_{t=1}^{T} \hat{\theta}_t$ or $\hat{\theta}_T$

[slide modified from Adam Smith, BU CS 591 Fall 2018]

DP SGD: Improved Privacy Analysis

[Bassily-Smith-Thakurta `14, Abadi-Chu-Goodfellow-McMahan-Mironov-Talwar-Zhang `17]

- Privacy amplification by subsampling:
  If $S : \mathcal{X}^n \to \mathcal{X}^B$ outputs a random subset of pn out of n rows and $M : \mathcal{X}^B \to \mathcal{Y}$ is (ε, δ)-DP, then
  $M'(x) = M(S(x))$ is $(\ln(1 + (e^{\epsilon} - 1)p), p\delta)$-DP.
  – Keep S_t secret; use its randomness

$\approx p\epsilon$

- Poisson sampling: choosing each point independently with prob. $p = B/n$.
- Choosing B points without replacement
- Choosing B points with replacement

[slide modified from Adam Smith, BU CS 591 Fall 2018]

DP SGD: Improved Privacy Analysis

[Bassily-Smith-Thakurta `14, Abadi-Chu-Goodfellow-McMahan-Mironov-Talwar-Zhang `17]

- Idea: Keep S_t secret; use its randomness

- Privacy amplification by subsampling:
  If $S : \mathcal{X}^n \to \mathcal{X}^B$ outputs a random subset of pn out of n rows and
  $M : \mathcal{X}^B \to \mathcal{Y}$ is (ε, δ)-DP, then
  $M'(x) = M(S(x))$ is $(\ln(1 + (e^\epsilon - 1)p), p\delta)$-DP.

$\approx p\epsilon$

- We can take $p = B/n$.
  - Unfortunately privacy amplification by subsampling does not hold for zCDP.
  - But similar analysis can be recovered using the "moments accountant" [Abadi et al. `17], "truncated zCDP" [Bun et al. `18], or f-DP [Dong et al. `19, Doroshenko et al. `22]

[slide modified from Adam Smith, BU CS 591 Fall 2018]

Neural Networks & Privacy

- Choice of the model architecture
  - Noise is proportional to the square root of number of parameters.
- Hyperparameter tuning
  - Run analyses on the training data with various hyperparameter settings, and choose the best one. Q: any problems?
  - Can do this privately, with additional cost in privacy.
  - Or can tune on a public dataset.
- Can pretrain with "public" dataset (e.g. use a foundation model) and then fine-tune using a sensitive dataset.
- State of Art as of 2022 [Bu-Mao-Xu]:
  - CIFAR-10 (finetuned after pretraining on ImageNet): 96.7% accuracy with $(1, 10^{-5})$-DP (vs. 99.7% w/o DP)
- Current analyses of DP-SGD are nearly tight if adversary sees all intermediate θ_t's [Nasr et al. `23]

Differentially Private Empirical Risk Minimization

Supervised ML Output

Primary Goal (risk minimization):

- Find $\theta \in \Theta$ minimizing $L(\theta) = \mathrm{E}_{(x,y)\sim\mathcal{P}}[\ell(\theta|x,y)]$.
- Difficulty: $\mathcal{P}$ unknown.

Subgoal 1 (empirical risk minimization (ERM)):

- Find $\theta \in \Theta$ minimizing $L(\theta) = \frac{1}{n}\sum_{i=1}^{n} \ell(\theta|x_i, y_i)$.
- Turns learning into optimization.
- Difficulty: overfitting*

Subgoal 2 (regularized ERM):

- Find $\theta \in \Theta$ minimizing $L(\theta) = \frac{1}{n}\sum_{i=1}^{n} \ell(\theta|x_i, y_i) + R(\theta)$.
- $R(\theta)$ typically penalizes "large" θ, can capture Bayesian prior.

*Fact: DP automatically helps prevent overfitting! [Dwork et al. `15]

Output Perturbation

[Chaudhuri-Monteleoni-Sarwate `11]

$$M(\vec{x}, \vec{y}) = \mathrm{argmin}_{\theta} \left(\frac{1}{n} \sum_{i=1}^{n} \ell(\theta | x_i, y_i) + R(\theta) \right) + \mathrm{Noise}$$

Challenge: bounding sensitivity of $\theta_{opt} = \mathrm{argmin}_{\theta}(\cdot)$

- Global sensitivity can be infinite (e.g. OLS regression)
- Global sensitivity can be bounded when ℓ is strictly convex, has bounded gradient (as a function of θ), and R is strongly convex. Even analyzing local sensitivity seems to require unique global optimum and using an optimizer that is guaranteed to succeed.

Objective Perturbation

[Chaudhuri-Monteleoni-Sarwate `11]

$$M(\vec{x}, \vec{y}) = \operatorname{argmin}_\theta \left(\frac{1}{n} \sum_{i=1}^{n} \ell(\theta | x_i, y_i) + R(\theta) + R_{\text{priv}}(\theta, \text{noise}) \right)$$

Challenge: how to put noise in the objective function?

- [CMS11] use $R_{\text{priv}}(\theta, v) = \langle \theta, v \rangle + c\|\theta\|^2$ where v is sampled with probability density $\propto \exp(-c'\varepsilon\|v\|)$.
- Privacy proven under similar assumptions on ℓ and R as before, plus ℓ having bounded Jacobian.
- Has better performance than output perturbation [CMS11].

Exponential Mechanism for ML

[Kasiwiswanathan-Lee-Nissim-Raskhodnikova-Smith `11]

Use score function

$$s\big((x,y),\theta\big) = -L(\theta|x,y) = -\frac{1}{n}\sum_{i=1}^{n}\ell(\theta|x_i,y_i) - R(\theta).$$

That is,

$$\Pr[M(\vec{x},\vec{y}) = \theta] \propto e^{-\frac{\varepsilon}{2}\sum_{i=1}^{n}\ell(\theta|x_i,y_i)-\frac{\varepsilon n}{2}R(\theta)}.$$

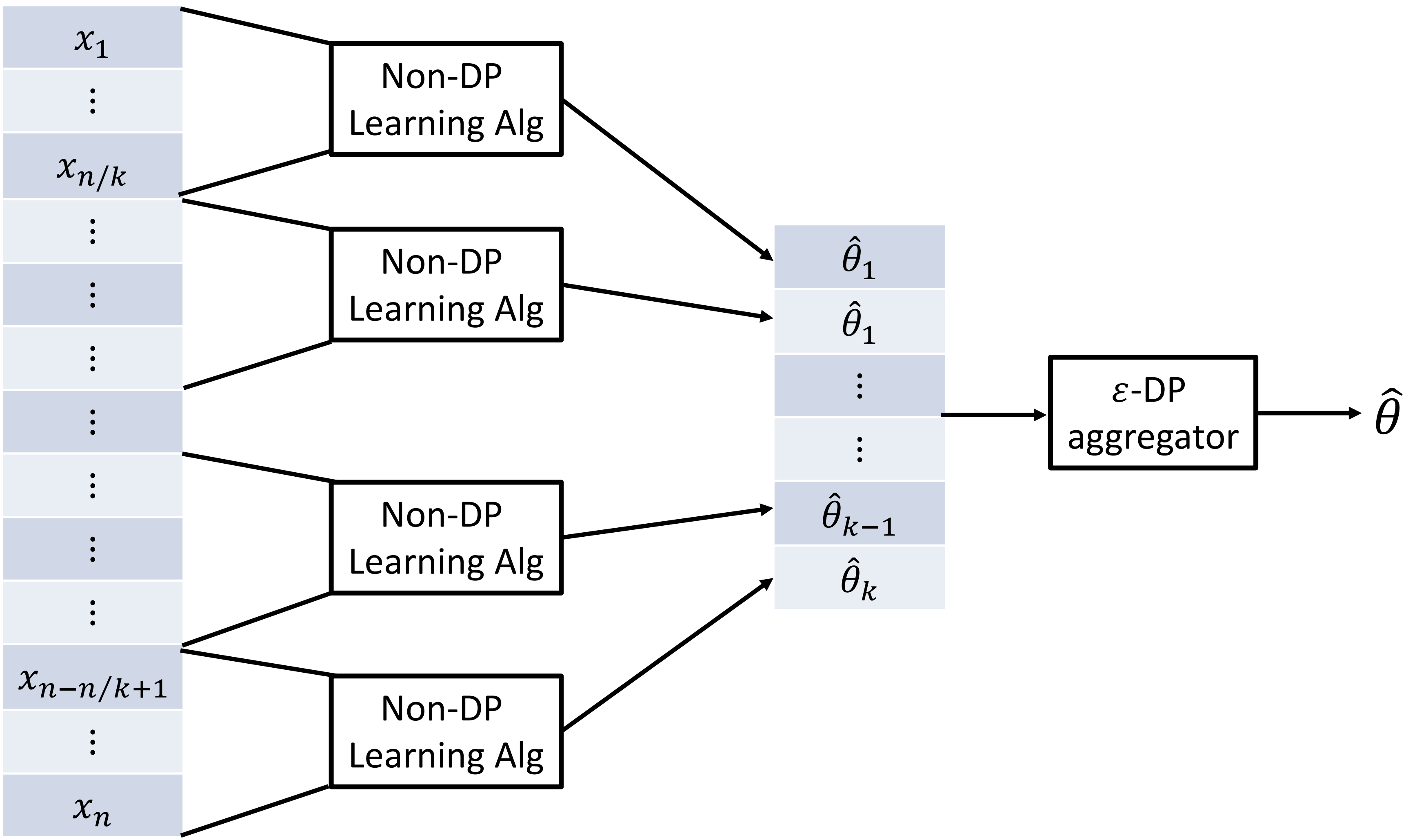
Is ε-DP if the loss functions are clipped to $[0,1]$. (why?)

Thm [KLNRS `11, informally stated]: anything learnable non-privately on a finite data universe is also learnable with DP (with larger n).

Problem: runtime often exponential in dimensionality of θ.

Subsample & Aggregate

[Nissim-Rakhodnikova-Smith `07, Smith `11]



Q: Why is this ε-DP?

Subsample & Aggregate

[Nissim-Rakhodnikova-Smith `07, Smith `11]

- Typical aggregators: DP (clipped) mean, DP median
- Benefits:
  - Use any non-private estimator as a black box
  - Can give optimal asymptotic convergence rates: for many statistical estimators, variance is asymptotically $c_\theta/(\text{sample size})$, so variance of DP mean $\hat{\theta}$ is
$$(1/k)\cdot(c_\theta\cdot k/n) + O(1/\varepsilon k)^2 = \big(1+o(1)\big)\cdot c_\theta/n$$
if $k=\omega(\sqrt{n})$.
- Drawbacks:
  - Dependence on dimension, model parameters, distribution can be bad.
  - Often takes very large sample size to kick in.
- PATE [PAE+17, PSM+18]: Use S&A just to label a public dataset

Modifying ML Algorithms

- Another approach: decompose existing ML/inference algorithms into steps that can be made DP, like Statistical Queries (estimating means of bounded functions)

- Example: linear regression
  - $S_{xx}/n\,, S_{xy}/n\,, \bar{x}, \bar{y}$ are all statistical queries