Multi-party DP Write-up 2
Stochastic Gradient Descent

1 Notations and Settings

Let $D = [\mathbf{X}, \mathbf{y}]$ be a database of size n, with $\mathbf{X} \in \mathbb{R}^{n \times p}$ and $\mathbf{y} \in \mathbb{R}$. $[n]$ stands for the set $\{1, \ldots, n\}$. We use $z \sim N(\mu, \sigma)$ to denote a variable following the Gaussian distribution with mean μ and variance σ^2.

Definition 1 (ERM). *Given a data set $D = \{d_1, \ldots, d_n\}$ drawn from a universe $\mathcal{X} \subseteq \mathbb{R}^{n \times (p+1)}$, and a closed, convex set $\mathcal{C}$, the goal of Empirical Risk Minimization (ERM) is*

$$\min_{\theta \in \mathcal{C}} L(\theta; D) = \Sigma_{i=1}^{n} l(\theta; d_i) \tag{1}$$

The function l defines, for each data point d, a loss function $l(.; d)$ on $\mathcal{C}$. We will generally assume that $l(; d)$ is convex and L-Lipschitz for all $d \in \mathcal{X}$.

Definition 2 (Lipschitz Function). *$l : \mathcal{C} \to \mathbb{R}$ is L-Lipschitz (in the Euclidean norm) if, for all pairs $x, y \in \mathcal{C}$, we have $|l(x) - l(y)| \leq L|x - y|_2$.*

Definition 3 (Distributed Private ERM). *Let $A_1, \ldots A_t$ and B be $t + 1$ parties. Each A_i owns some of the data $\{d_j\}_{j \in S_i}$, where $S_i \subseteq [n]$ and $|S_i| = n_i$. For simplicity, we assume $\{S_1, \ldots, S_t\}$ partition $[n]$.*
Our goal is to find an algorithm that makes $\{A_1, \ldots A_t, B\}$ distributively compute θ under ERM model. such that the entire computation, including all the intermediate outputs and final output, is (ϵ, δ)-differential private.

Privacy & **Utility.** There is always a trade-off between privacy and utility. We use the concept of worst case (over input data) *expected excess risk* to evaluate the utility of our algorithm.

Definition 4. *Given loss function $L(\theta; D)$ over database D, its worst case expected excess risk is defined as*

$$E[L(\theta; D) - L(\theta^*; D)] \tag{2}$$

where θ is the output of the algorithm, θ^* is the optimal minimizer of the loss function and the randomness comes from the algorithm.

2 Stochastic Gradient Descent

We propose two distributed algorithms, both of which are based on the stochastic gradient descent algorithm in [1].

Algorithm SGD1.

1. B uniformly chooses the initial assignment $\theta_0 \in \mathcal{C}$, learning rate function $\eta : [n^2] \to \mathbb{R}$, a noise variance $\sigma^2 > 0$, and a sequence of $\pi = [\pi_1, \ldots \pi_{n^2}]$ with each π_j uniformly chosen from $[n]$. B broadcasts $(\theta_0, \eta, \sigma^2)$.
2. In each iteration $i = 1$ to n^2, B gives a computing permission to agent A_j who owns data $d_{\pi(i)}$.
3. Upon receiving permission, A_j will compute $\theta_{i+1} = \Pi_{\mathcal{C}}(\theta_i - \eta(i)G_i)$, where $G_i = n\Delta(l(\theta_i; d_{\pi(i)})) + z_i$ with $z_i \sim N(0, \sigma^2)^p$, and $\Pi_{\mathcal{C}}$ is the projection function onto set $\mathcal{C}$. A_j sends G_i to B.
4. B broadcast G_i together with a update permission to all other A_k, where $k \neq j$.
5. Each of the remaining A_k will update $\theta_{i+1} = \Pi_{\mathcal{C}}[\theta_i - \eta(i)G_i]$
6. After n^2 iterations, B returns $\theta = \theta_{n^2}$ as the final output.

For $0 < \epsilon < 1$ and $0 < \delta < 1/n$, we set parameters in Algorithm SGD1 as the follows

$$\sigma^2 = O(\frac{L^2 n^2 \log(n/\delta) \log(1/\delta)}{\epsilon^2}) \tag{3}$$

$$\eta(t) = \frac{|\mathcal{C}|_2}{\sqrt{t(n^2 L^2 + p\sigma^2)}} \tag{4}$$

Theorem 1. *Algorithm SGD1 is $(\epsilon, 1 - (1 - \delta/2)^{n^2})(1 - \delta^{1/3}))$-Private. In particular, Algorithm SGD1 is $(\epsilon, n^2\delta/2)$ when $\delta = \omega(1/n^3)$,*

Proof. The entire output of Algorithm SGD1 consists of $\{G_1, \ldots, G_{n^2}\}$. We show adaptively outputting n^2 such G_i is $(\epsilon, (n^2 + 1)\delta)$-Private.

Let $G_i(D) = n\Delta(l(\theta_i; d_{\pi(i)})) + z_i$ be a random variable defined over the randomness of π, z_i and conditioned on θ_i. Fixing the randomness of π, it can be shown according to standard Gaussian Differential Privacy analysis that each G_i is $(\frac{\epsilon}{2\sqrt{\log(1/\delta)}}, \delta/2)$-DP [1]. (We just need to show with probability at least $1-\delta/2$, $|\log(\frac{G_i(D)}{G_i(D')})| \leq \frac{\epsilon}{2\sqrt{\log(1/\delta)}}$, where D, D' are a pair of neighboring databases)

It is obvious that π provides the same randomness as the uniform sampling from D. By the privacy amplification lemma in [1], each G_i is $(\frac{\epsilon}{n\sqrt{\log(1/\delta)}}, \delta/2)$-DP. We apply the strong composition theorem lemma [2] with $k = n^2$, $\delta' = \delta^{1/3}$. We have

$$n^2\epsilon^2 + \epsilon\sqrt{2n^2\log(1/\delta')} = \epsilon(\frac{\epsilon}{\log(1/\delta)} + \sqrt{2/3}) \leq \epsilon \tag{5}$$

Lemma 1 (Strong Composition Theorem)**.** *For any $1 > \epsilon > 0$, $\delta \in [0,1]$ and $\delta' \in [0,1]$ the class of (ϵ, δ)-differentially private mechanisms satisfies $(\epsilon', 1-(1-\delta)^k)(1-\delta')$-DP under k-fold adaptive composition for*

$$\epsilon' = \min\{k\epsilon, k\epsilon^2 + \epsilon\sqrt{2k\log(1/\delta')}\} \tag{6}$$

Algorithm SGD2.

1. B uniformly chooses the initial assignment $\theta_0 \in \mathcal{C}$, learning rate function $\eta : [n^2] \to \mathbb{R}$, a noise variance $\sigma^2 > 0$, and a sequence of $\pi = [\pi_1, \ldots \pi_t]$ with each π_j uniformly chosen from $[t]$. B broadcasts $(\theta_0, \eta, \sigma^2)$.

2. In each iteration $i = 1$ to t, B gives a computing permission to agent A_{π_i}.

3. Upon receiving permission, for $j = 1$ to n_{π_i}, A_{π_i} uniformly pick d_j from his dataset S_{π_i}. Then, A_{π_i} will compute $\theta_{j+1} = \Pi_{\mathcal{C}}(\theta_j - \eta(j)G_j)$, where $G_j = n\Delta(l(\theta_j; d_j)) + z_j$ with $z_j \sim N(0, \sigma^2)^p$, and $\Pi_{\mathcal{C}}$ is the projection function onto set $\mathcal{C}$. A_{π_i} sends $\{G_j\}_{j\in[n_{\pi_i}]}$ to B.

4. B broadcast $\{G_j\}_{j\in[n_{\pi_i}]}$ together with a update permission to all other A_k, where $k \neq j$.

5. Each of the remaining A_k will update n_{π_i} steps by $\theta_{j+1} = \Pi_{\mathcal{C}}[\theta_j - \eta(j)G_j]$

6. After n^2 iterations, B returns $\theta = \theta_{n^2}$ as the final output.

Theorem 2. *Algorithm SGD2 is*

3 References

1. http://www.cse.psu.edu/ ads22/pubs/BST14/2014-04-10-BST14-convex-opt.pdf
2. http://arxiv.org/pdf/1311.0776v2.pdf