Dynamic Policies on Differentially Private Learning

Junyuan Hong
Michigan State University
Machine Learning in Our Life

Local model training
Federated Learning

Federated Learning

Local model training

Privacy leakage

Reverse engineering

Federated Learning

Differential Privacy

Local model training


Federated Learning

Private Learning:
- Publish knowledge (model) rather than data
- Control the privacy loss
- Guarantee the convergence

Differential Privacy
Private Learning

Algorithm

Convergence theory and dynamic policy
Learning by Gradient Descent

\[ \nabla t = \frac{1}{N} \sum_{n=1}^{N} \nabla f(\theta; x_n) \]

- \( \rho \) privacy measure
- \( \pi \) projection: AdaGrad, etc

Private sample (to protect)
Privacy attack

- **2019Grad**: Deep Leakage from Gradients, Zhu et al.: $x = \text{arg min} \| \nabla f(x) - \nabla_t \|_2$
- **2017MIA**: Membership Inference Attacks, Shokri et al.: $P(x \in D_{\text{train}}) = h(f(x; \theta))$ where $h()$ is a trained attack.
- **2017GAN**: Info Leakage from Collaborative Deep Learning, Hitaj et al. 2017: $x = G(z)$ where $z = \max f(G(z); \theta)$
- **2015MI**: Model Inversion, Fredrikson et al.: $x = \text{arg max} f(x)$ (statistical model)
Quantify privacy

If privacy cost is over a budget, we stop and publish model.
Quantify privacy: Differential Privacy (DP)
Differential Privacy

\[ \mathcal{A}(D) \]

\[ \mathcal{A}(D') \]

Adversary
Differential Privacy

Privacy loss at $y$

$$Z(y) \triangleq \log \left( \frac{p(\mathcal{A}(D) = y)}{p(\mathcal{A}(D') = y)} \right)$$

where $y \sim \mathcal{A}(D)$ and $D, D'$ are adjacent (differing at one sample)
Differential Privacy

Privacy loss at $y$

$$Z(y) \triangleq \log \left( \frac{p(\mathcal{A}(D) = y)}{p(\mathcal{A}(D') = y)} \right)$$

where $y \sim \mathcal{A}(D)$

$\mathcal{A}$ is $\epsilon$-DP: $Z \leq \epsilon$ or $P(Z > \epsilon) = 0$

$\mathcal{A}$ is $(\epsilon, \delta)$-DP: $P(Z > \epsilon) = \delta$

$\mathcal{A}$ is $\rho$-$z$CDP: $P(Z > t + \rho) \leq e^{-t^2/(4\rho)}$ for $t \geq 0$

$\mathcal{A}$ is $(\rho, \omega)$-tCDP: $P(Z > t + \rho) \leq e^{-(\omega-1)^2\rho} \cdot e^{-(\omega-1)t}$ for $t \in [0, 2\rho(\omega - 1)]$

$P(Z > t \rho) \leq e^{(\omega-1)^2\rho} \cdot e^{-(\omega-1)t}$ for $t \in (2\rho(\omega - 1), \infty)$

Quantify privacy: Accumulate privacy loss

Compose **dynamic** privacy parameter

**Lemma 3.5.** (Composition) Suppose two mechanisms \( M, M' : \mathcal{D}^n \rightarrow \mathbb{R}^d \) satisfy \( \rho_1 \)-zCDP and \( \rho_2 \)-zCDP, then their composition satisfies \( (\rho_1 + \rho_2) \)-zCDP.

Note: zCDP allows \( \rho_1 \) and \( \rho_2 \) to be different, but DP does not. For DP, an additional privacy cost has to be paid.

Quantify privacy

\[ \nabla_t = \frac{1}{N} \sum_{n=1}^{N} \nabla f(\theta; x_n) \]

\( \pi \) is **deterministic** which is non-private

\[ \rho \] privacy measure

\[ \pi \] projection: AdaGrad, etc

Private sample (to protect)
Privatize Gradients

Algorithm 1 Privatizing gradients

**Input:** Private gradient $\nabla_t$ summed from $[\nabla_t^{(1)}, \ldots, \nabla_t^{(n)}]$, residual privacy budget $R_t$

1. $\nabla_t \leftarrow \frac{1}{N} \sum_{n=1}^{N} \nabla_t^{(n)} \min\{1, C_t/\left\|\nabla_t^{(n)}\right\|\}$ \hspace{1cm} $\triangleright$ Sensitivity constraint
2. $\rho_t \leftarrow 1/\sigma_t^2$
3. **if** $\rho_t < R_t$ **then**
   4. $R_{t+1} \leftarrow R_t - \rho_t$ \hspace{1cm} $\triangleright$ Budget request
5. $g_t \leftarrow \nabla_t + C_t \sigma_t \nu_t / N, \nu_t \sim \mathcal{N}(0, I)$ \hspace{1cm} $\triangleright$ Privacy noise
6. **return** $\eta_t g_t, R_{t+1}$ \hspace{1cm} $\triangleright$ Utility projection
7. **else**
8. Terminate

- $\rho$ privacy measure
- $\pi$ projection: AdaGrad, etc
- $\sigma$ noise schedule
- $\mathcal{N}$ noise distribution
- $C$ sensitivity constraint
Privatize Gradients

**Lemma 3.1 (L2 sensitivity).** Given mapping from a n-element dataset domain to d-dimensional real space \( f : \mathcal{D}^n \rightarrow \mathbb{R}^d \), the L2 sensitivity of \( f \), denoted by \( \Delta_2(f) \), is defined as:

\[
\Delta_2(f) = \max_{D,D'} \| f(D) - f(D') \|_2,
\]

where \( D, D' \) are adjacent datasets.

**Algorithm 1** Privatizing gradients

**Input:** Private gradient \( \nabla_t \) summed from \( \{\nabla^{(1)}_t, \ldots, \nabla^{(n)}_t\} \), residual privacy budget \( R_t \)

1. \( \nabla_t \leftarrow \frac{1}{N} \sum_{i=1}^{N} \nabla^{(i)}_t \min\{1, C_t/\|\nabla^{(i)}_t\|\} \) \( \triangleright \) Sensitivity constraint
2. \( \rho_t \leftarrow 1/\sigma_t^2 \)
3. **if** \( \rho_t < R_t \) **then**
4. \( \rho_t \leftarrow R_t - \rho_t \)
5. \( g_t \leftarrow \nabla_t + C_t\sigma_t \nu_t/N, \nu_t \sim \mathcal{N}(0, I) \)
6. **return** \( \rho_t, g_t, R_{t+1} \) \( \triangleright \) Privacy noise
7. **else**
8. **Terminate**

**Control the influence of a sample**

\( \rho \) privacy measure

\( \pi \) projection: AdaGrad, etc

\( \sigma \) noise schedule

\( \mathcal{N} \) noise distribution

\( C \) sensitivity constraint
Differentially Private Learning

A deterministic function

Algorithm 1 Privatizing gradients

**Input:** Private gradient $\nabla_t$ summed from $[\nabla_t^{(1)}, \ldots, \nabla_t^{(n)}]$, residual privacy budget $R_t$

1: $\hat{\nabla}_t \leftarrow \frac{1}{N} \sum_{i=1}^{N} \nabla_i^{(n)} \min\{1, C_i/\|\nabla_i^{(n)}\|\}$  
\text{▷ Sensitivity constraint}

2: $\rho_t \leftarrow 1/\sigma_t^2$  

3: If $\rho_t < R_t$ then

4: $R_{t+1} \leftarrow R_t - \rho_t$  

5: $g_t \leftarrow \hat{\nabla}_t + C_t \sigma_t \nu_t / N$, $\nu_t \sim \mathcal{N}(0, I)$  

6: return $\eta_t g_t$, $R_{t+1}$  

\text{▷ Privacy noise}

\text{▷ Utility projection}

7: Else

8: Terminate

**Lemma 3.4.** The Gaussian mechanism, which returns $\mu(D) + \sigma v$ satisfies $\Delta_2(f)^2 / (2\sigma^2) - zCDP$.  

**$\rho$** privacy measure  

**$\pi$** projection: AdaGrad, etc  

**$\sigma$** noise schedule  

**$\mathcal{N}$** noise distribution  

**$C$** sensitivity control
Differentially Private Learning

If gradients are a stochastic mini-batch, e.g., sampled by q-probability, the privacy cost is $\propto q^2 \rho$ for DP metric, e.g., tCDP.

![Diagram showing the process of Privatizing gradients]

Algorithm 1 Privatizing gradients

```
Input: Private gradient $\nabla_t$ summed from $[\nabla_t^{(1)}, \ldots, \nabla_t^{(n)}]$ residual privacy budget $R_t$

1: $\bar{\nabla}_t \leftarrow \frac{1}{N} \sum_{i=1}^{N} \nabla_t^{(i)} \min\{1, C_i/\left\|\nabla_t^{(i)}\right\|\}$

2: $\rho_t \leftarrow 1/\sigma_t^2$

3: if $\rho_t < R_t$ then

4: $R_{t+1} \leftarrow R_t - \rho_t$

5: $g_t \leftarrow \bar{\nabla}_t + C_i \sigma_t \nu_t / N, \nu_t \sim N(0, I)$

6: return $g_t, R_{t+1}$

7: else

8: Terminate
```

- $\rho$ privacy measure
- $\pi$ projection: AdaGrad, etc
- $\sigma$ noise schedule
- $\mathcal{N}$ noise distribution
- $C$ sensitivity control
Privatize Gradients

\[ \rho_t - zCDP \]

\[ \theta_t \xrightarrow{\text{Protector}} \theta_{t+1} \]

\[ \rho \]

\[ \pi \]

\[ \sigma \]

\[ \mathcal{N} \]

\[ C \]

\[ \rho \]

privacy measure

projection: AdaGrad, etc

noise schedule

noise distribution

sensitivity constraint

Algorithm 1 Privatizing gradients

**Input:** Private gradient \( \nabla_t \) summed from \( [\nabla_t^{(1)}, \ldots, \nabla_t^{(n)}] \), residual privacy budget \( R_t \)

1: \( \tilde{\nabla}_t \leftarrow \frac{1}{N} \sum_{n=1}^{N} \nabla_t^{(n)} \min\{1, C_t / \| \nabla_t^{(n)} \| \} \)

\( \rhd \) Sensitivity constraint

2: \( \rho_t \leftarrow 1 / \sigma_t^2 \)

3: if \( \rho_t < R_t \) then

4: \( R_{t+1} \leftarrow R_t - \rho_t \)

5: \( g_t \leftarrow \tilde{\nabla}_t + C_t \sigma_t \nu_t / N, \nu_t \sim \mathcal{N}(0, I) \)

6: return \( \eta_t g_t, R_{t+1} \)

\( \rhd \) Privacy noise

7: else

8: Terminate
Differentially Private Learning

Algorithm 1 Privatizing gradients

Input: Private gradient $\nabla_t$ summed from $[\nabla_t^{(1)}, \ldots, \nabla_t^{(n)}]$, residual privacy budget $R_t$

1: $\tilde{\nabla}_t \leftarrow \frac{1}{N} \sum_{n=1}^{N} \nabla_t^{(n)} \min\{1, C_t/\|\nabla_t^{(n)}\|\}$  \hspace{1cm} $\triangleright$ Sensitivity constraint
2: $\rho_t \leftarrow 1/\sigma_t^2$
3: if $\rho_t < R_t$ then
4: \hspace{1cm} $R_{t+1} \leftarrow R_t - \rho_t$
5: \hspace{1cm} $g_t \leftarrow \tilde{\nabla}_t + C_t \sigma_t \nu_t / N$, $\nu_t \sim \mathcal{N}(0, I)$
6: \hspace{1cm} return $\eta_t g_t$, $R_{t+1}$  \hspace{1cm} $\triangleright$ Privacy noise
7: \hspace{1cm} else
8: \hspace{2cm} Terminate  \hspace{1cm} $\triangleright$ Utility projection
Private Learning

Algorithm

Convergence theory and dynamic policy
Does private learning converge?

• Not converge to the optimal
  • Finite iteration
  • Noise
• Improve the final iterate loss given a privacy budget:
  \[ \text{EER} = \mathbb{E}_\nu[f(\theta_{T+1})] - f(\theta^*) \]
• The upper bound of EER
Why study convergence upper bound?

• Bound the worst case.
• Find a way to speed up optimization algorithm
• To study the impact of privacy operations, e.g., noise magnitude, clipping norm, etc.
• To compare different algorithms: convergence rate
Assumptions

- $G$-Lipschitz continuous loss,
  \[ \| f(x) - f(x') \| \leq G \| x - x' \| \Leftrightarrow \| f'(x) \| \leq G \text{ if } \hat{f} \text{ is differentiable.} \]

- $M$-Lipschitz continuous gradient or $M$-smooth loss:
  \[ \| \nabla f(x) - \nabla f(x') \| \leq M \| x - x' \| \]

- $\mu$-Polyak-Lojasiewicz (PL) condition $< \mu$-strongly convex
  \[ \| \nabla f(\theta) \|^2 \geq 2\mu(f(\theta) - f(\theta^*)) \]
Convergence

\textbf{Algorithm 1} Privatizing gradients

\textbf{Input}: Private gradient $\nabla_t$ summed from $[\nabla_t(1), \ldots, \nabla_t(n)]$, residual privacy budget $R_t$

1: $\bar{\nabla}_t \leftarrow \frac{1}{N} \sum_{n=1}^{N} \nabla_t^{(n)} \min\{1, C_t / \|\nabla_t^{(n)}\|\}$ \hspace{1cm} \triangleright \text{Sensitivity constraint}
2: $\rho_t \leftarrow 1/\sigma_t^2$
3: \textbf{if} $\rho_t < R_t$ \textbf{then}
4: \hspace{0.5cm} $R_{t+1} \leftarrow R_t - \rho_t$
5: \hspace{0.5cm} $g_t \leftarrow \bar{\nabla}_t + C_t \sigma_t \nu_t / N$, $\nu_t \sim N(0, I)$
6: \hspace{0.5cm} \textbf{return} $\eta_t g_t, R_{t+1}$ \hspace{1cm} \triangleright \text{Utility projection}
7: \textbf{else}
8: \hspace{0.5cm} Terminate

\textbf{Theorem 3.2}. Let $\alpha, \kappa$ and $\gamma$ be defined in Eq. (5), and $\eta_t = \frac{1}{\sqrt{M}}$. Suppose $f(\theta; x_t)$ is $G$-Lipschitz $M$-smooth and satisfies the Polyak-Łojasiewicz condition. If $C_t \leq G$, then clipping does not take place, i.e., $\bar{\nabla}_t = \nabla_t$ and the following holds:

$$E_E \left[ f(\theta_{T+1}) \right] - f(\theta^*) \leq \left( \gamma T + R \sum_{t=1}^{T} q_t \sigma_t^2 \right) (f(\theta_t) - f(\theta^*)),$$

where $q_t \triangleq \gamma T - t \kappa$. \hspace{1cm} (6)

$$\alpha_t \triangleq \frac{MD}{2R} \left( \frac{\eta_t C_t}{N} \right)^2 \frac{1}{f(\theta_t) - f(\theta^*)} > 0, \kappa \triangleq \frac{M}{\mu} \geq 1, \text{ and } \gamma \triangleq 1 - \frac{1}{\kappa} \in [0, 1).$$ \hspace{1cm} (5)
Convergence

**Theorem 3.2.** Let $\alpha$, $\kappa$ and $\gamma$ be defined in Eq. (5), and $\eta_t = \frac{1}{\sqrt{t}}$. Suppose $f(\theta; x_t)$ is $G$-Lipschitz $M$-smooth and satisfies the Polyak-Lojasiewicz condition. If $C_t \leq G$, then clipping does not take place, i.e., $\tilde{\nabla}_t = \nabla_t$ and the following holds:

\[
\text{EER} \leq \left( \gamma^T + R \sum_{t=1}^{T} q_t \sigma_t^2 \right) \left( f(\theta_i) - f(\theta^*) \right),
\]

where $q_t \overset{\text{def}}{=} \gamma^{T-t} \alpha_t$.  

Finite iteration  \hspace{1cm} Noise impact

- Schedule noise to
  - Extend iteration $T$
  - Reduce the effect of noise
Convergence

**Theorem 3.2.** Let $\alpha$, $\kappa$, and $\gamma$ be defined in Eq. (5), and $\eta_t = \frac{1}{\lambda t}$. Suppose $f(\theta; x_t)$ is $G$-Lipschitz $M$-smooth and satisfies the Polyak-Łojasiewicz condition. If $\tilde{C}_t \leq G$, then clipping does not take place, i.e., $\tilde{\nabla}_t = \nabla_t$ and the following holds:

$$
\text{EER} \leq \left( \gamma^T + R \sum_{t=1}^{T} q_t \sigma_t^2 \right) (f(\theta_t) - f(\theta^*)) ,
$$

where $q_t \triangleq \gamma^{T-t} \alpha$.  \hfill (6)

**Lemma 3.1** (Dynamic schedule). Suppose $\sigma_t$ satisfy $\sum_{t=1}^{T} \sigma_t^{-2} = R$. Given a positive sequence $\{q_t\}$, the following equation holds:

$$
\min_{\sigma} R \sum_{t=1}^{T} q_t \sigma_t^2 = \left( \sum_{t=1}^{T} \sqrt{q_t} \right)^2 , \text{ when } \sigma_t = \frac{1}{R} \sum_{i=1}^{T} \sqrt{\frac{q_i}{q_t}} .
$$

Influence of noise

Reduce noise impact

How much improvement can we achieve?
Advantage of dynamic schedule on optimal upper bound

Extend iter

stable when the loss curvature is sharp
Advantage of dynamic schedule

- Empirically check the $q_t$

$$
EER \leq \left( \gamma^T + R \sum_{t=1}^{T} q_t \sigma_i^2 \right) (f(\theta_1) - f(\theta^*)),
$$

where $q_t \triangleq \gamma^{T-t} \alpha_t$. 
Further reduce the noise by momentum

Algorithm 2 Privatizing gradients with debiased momentum

**Input:** Private gradient $\nabla_t$ summed from $[\nabla_{t}^{(1)}, \ldots, \nabla_{t}^{(n)}]$, residual privacy budget $R_t$

1: $\tilde{\nabla}_t \leftarrow \frac{1}{N} \sum_{n=1}^{N} \nabla_{t}^{(n)} \min\{1, C_t/\|\nabla_{t}^{(n)}\|\}$  
2: $\rho_t \leftarrow 1/\sigma_t^2$  
3: **if** $\rho_t < R_t$ **then**  
4: $R_{t+1} \leftarrow R_t - \rho_t$  
5: $g_t \leftarrow \tilde{\nabla}_t + \nu_1, \nu_1 \sim \mathcal{N}(0,(C_t\sigma_t/N)^2 I)$  
6: $v_{t+1} = \beta v_t + (1 - \beta) g_t, v_1 = 0$  
7: $\hat{v}_{t+1} = v_{t+1}/(1 - \beta^t)$  
8: **return** $\eta_t \hat{v}_{t+1}, R_{t+1}$  
9: **else**  
10: Terminate

▷ Sensitivity constraint  
▷ Budget request  
▷ Privacy noise  
▷ Utility projection
Further reduce the noise by momentum

Theorem 3.4 (Convergence under PL condition). Suppose $f(\theta; x_i)$ is $M$-smooth, $G$-Lipschitz and satisfies the Polyak-Lojasiewicz condition. Let $\eta_t = \eta_0$. If $C_t \geq G$ which implies $\widetilde{\nabla}_t = \nabla_t$ (clipping does not take place), then the following holds:

$$EER \leq \gamma^T (f(\theta_1) - f(\theta^*)) + \frac{2\eta_0 D}{N^2} \sum_{t=1}^{T} q_t (C_t \sigma_t)^2 + \eta_0 \zeta \sum_{t=1}^{T} \gamma^{T-t} \|v_{t+1}\|^2$$

(16)

where $q_t = \frac{\beta^2 (T-t+1) - \gamma^{T-t+1}}{\beta^2 - \gamma}$, $\gamma = 1 - \eta_0 \mu$, $\zeta = \frac{4M^2 \beta \gamma}{(\gamma - \beta)^2 (1 - \beta)^3} \eta_0^2 + \frac{1}{2} M \eta_0 - 1$. (17)

Especially, when $\eta_0 \leq \frac{\beta (1-\beta)^3}{8M} \left[ \sqrt{\frac{1}{4} + \frac{16}{\beta (1-\beta)^3}} - 1 \right]$, the noise variance dominates the bound, i.e.,

$$EER = \mathcal{O} \left( \frac{2\eta_0 D}{N^2} \sum_{t=1}^{T} q_t (C_t \sigma_t)^2 \right).$$

A negative term if $\eta_0$ is small.

The GD noise

Proof partially based on (Zhu, et al., ArXiv 2020)
Beyond dynamic noise magnitude

- Learning to protect: Transfer the dynamic policies learned from auxiliary tasks to private task.
- AdaClip (Pichapati et al. 2019): Adaptively clipping the gradients
- Dynamic batch size (Feldman et al., 2019, STOC): Increase the batch size to improve non-convex convergence bound.
Beyond dynamic noise magnitude

• **Learning to protect:** Transfer the dynamic policies learned from auxiliary tasks to private task.

• AdaClip (Pichapati et al. 2019): Adaptively clipping the gradients

• Dynamic batch size (Feldman et al., 2019, STOC): Increase the batch size to improve non-convex convergence bound.

\[
\min_{\pi, \sigma, T} \mathbb{E} \left[ \tilde{F}(\sigma, \pi, T) \right], \text{ s.t. } h_T(\sigma; \rho_{\text{tol}}) = 0
\]
Beyond dynamic noise magnitude

• **Learning to protect**: Transfer the dynamic policies learned from auxiliary tasks to private task.

• AdaClip (Pichapati et al. 2019): Adaptively clipping the gradients

• Dynamic batch size (Feldman et al., 2019, STOC): Increase the batch size to improve non-convex convergence bound.

![Figure 2: Test performance (top) and training loss values (bottom) by varying ε of logistic and MLP classifiers on IPUMS and MNIST35 datasets. The error bar presents the size of standard deviations. For better visualization, some horizontal offsets are added to every point.](image)

\[
\min_{\pi, \sigma, T} E \left[ \tilde{F}(\sigma, \pi, T) \right], \quad \text{s.t.} \ h_T(\sigma; \rho_{\text{tot}}) = 0
\]
Thank you for your time!