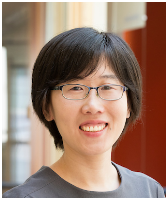
Advances in Federated Optimization: Efficiency, Resiliency, and Privacy

Yuejie Chi and Zhize Li



Carnegie Mellon University

ICASSP Tutorial
June 2023

Acknowledgements

Boyue Li
CMU→Apple

Haoyu Zhao
Princeton

This work is supported in part by NSF, ONR, and AFRL.

Introduction

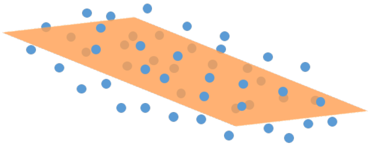
Empirical Risk Minimization (ERM)

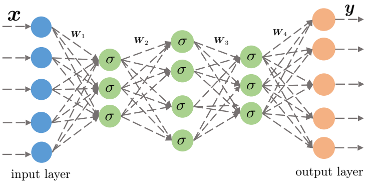
Given a set of data $\mathcal{M}$,

$$\text{minimize}_{\boldsymbol{x}} \quad f(\boldsymbol{x}) = \frac{1}{N} \sum_{\boldsymbol{z} \in \mathcal{M}} \ell(\boldsymbol{x}; \boldsymbol{z})$$

Here, $N =$ number of total samples.

- **convex:** least squares, logistic regression
- **non-convex:** PCA, training neural networks (focus of this tutorial)



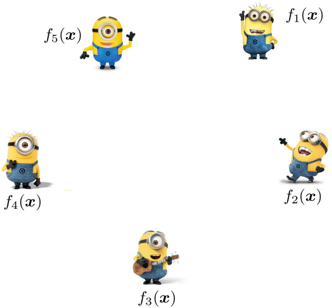


Let's go distributed

Distributed/Federated learning: due to privacy and scalability, data are distributed at multiple locations / workers / agents.

Let $\mathcal{M} = \cup_i \mathcal{M}_i$ be a data partition with equal splitting:

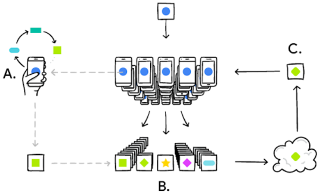
$$f(\boldsymbol{x}) := \frac{1}{n}\sum_{i=1}^{n} f_i(\boldsymbol{x}), \quad \text{where} \quad f_i(\boldsymbol{x}) := \frac{1}{(N/n)}\sum_{\boldsymbol{z}\in\mathcal{M}_i} \ell(\boldsymbol{x};\boldsymbol{z}).$$



$n =$ number of agents

$\underbrace{N/n}_{m} =$ number of local samples

Federated learning



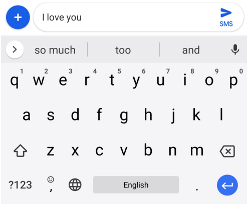
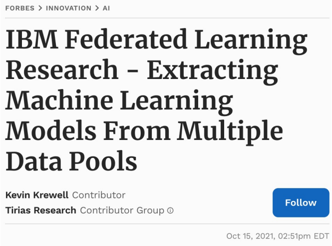


Image credit: Google


FORBES > INNOVATION > AI

IBM Federated Learning Research – Extracting Machine Learning Models From Multiple Data Pools

Kevin Krewell Contributor
Tirias Research Contributor Group

Follow

Oct 15, 2021, 02:51pm EDT


How Apple personalizes Siri without hoovering up your data

The tech giant is using privacy-preserving machine learning to improve its voice assistant while keeping your data on your phone.

By Karen Hao

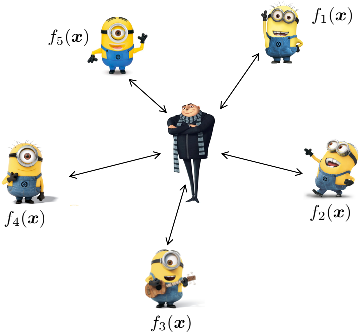
December 11, 2019

Federated learning is deployed nowadays by companies in many areas, e.g., on-device inference.

Multi-agent and distributed information processing

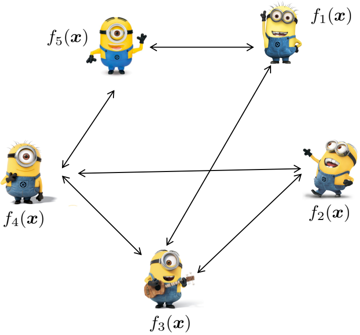
Decentralized processing without central coordination in wireless sensor networks, internet of things, swarms, ...

Two distributed schemes



Server/client model

PS coordinates *global* information sharing

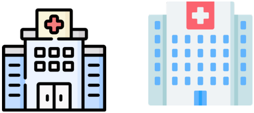


Network/decentralized model

agents share *local* information over a graph topology

Two data regimes

Entities



cross-silo

small n, large m

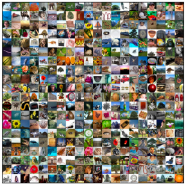
Devices

cross-device:

small m, large n

Challenges in federated/decentralized learning

- **Communication efficiency:** limited bandwidth, stragglers, ...
- **Heterogeneity:** non-iid data and systems across the agents
- **Privacy:** does not come for free without sharing data

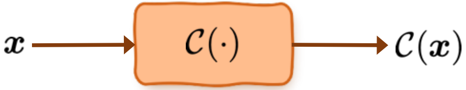


Communication efficiency

Communication cost $=$ Communication rounds $\times$ Cost per round

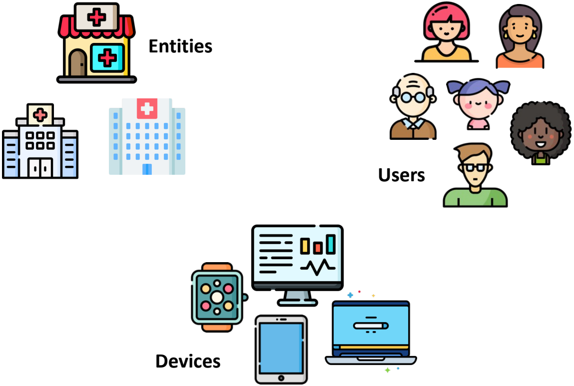
- **Local method:** perform more local computation to reduce communication rounds, e.g. FedAvg (McMahan et al., 2016).

- **Communication compression:** compress the message into fewer bits, e.g. sparsification or quantization (Alistarh et al., 2017).



— How to design communication-efficient algorithms?

Data heterogeneity

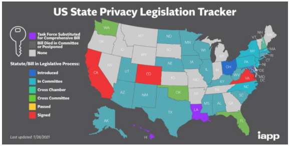


Heterogeneity measure

$$\textit{local objective} \quad \neq \quad \textit{global objective}$$

— Can we tame the data heterogeneity?

A little privacy, please



Privacy guarantees are becoming increasingly critical!

— Can we design privacy-preserving algorithms?

Tutorial outline

Part 0: Primer on centralized nonconvex optimization

Part 1: Efficient federated optimization via local methods

- Federated averaging
- SCAFFOLD: dealing with heterogeneity via variance reduction

Part 2: Communication-compressed federated optimization

- How do we compress? the role of error feedback
- Dealing with data heterogeneity

Part 3: Private federated optimization

- Differential privacy
- Understanding gradient clipping

Part 0: A Primer on Centralized Nonconvex Optimization

Unconstrained optimization

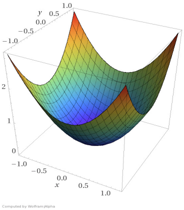
Consider an unconstrained optimization problem

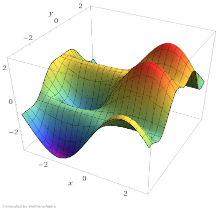
$$\text{minimize}_{\boldsymbol{x}} \qquad f(\boldsymbol{x})$$

Definition (first-order critical points)

A first-order critical point of f satisfies

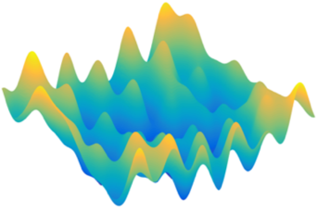
$$\nabla f(\boldsymbol{x}) = \mathbf{0}$$

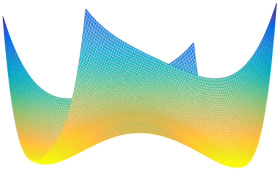




How do we converge to first-order critical points?

Smoothness





Definition (Smoothness)

A function $f(\boldsymbol{x})$ is L-smooth if

$$\|\nabla f(\boldsymbol{x}_1) - \nabla f(\boldsymbol{x}_2)\|_2 \leq L\|\boldsymbol{x}_1 - \boldsymbol{x}_2\|_2$$

Convergence of gradient descent (GD)

Gradient descent (GD):

$$\boldsymbol{x}_{t+1} = \boldsymbol{x}_t - \eta_t \nabla f(\boldsymbol{x}_t)$$

where η_t is the learning rate.

Theorem (Convergence of GD)

Suppose $f^\star = \min_{\boldsymbol{x}} f(\boldsymbol{x}) > -\infty$. *Setting* $\eta_t = \eta = 1/L$, *it satisfies*

$$\frac{1}{T}\sum_{t=0}^{T-1} \|\nabla f(\boldsymbol{x}_t)\|_2^2 \leq \frac{2L\Delta}{T},$$

where $\Delta = f(\boldsymbol{x}_0) - f^\star$.

- GD converges at the rate $O(1/T)$ in terms of the average squared gradient norm.
- For finite-sum problems of size n, the IFO complexity of GD is $O(n\varepsilon^{-1})$ to reach $\mathbb{E}\|\nabla f(\boldsymbol{x}^{\text{output}})\|_2^2 \leq \varepsilon$.

Convergence of GD under smoothness

- By smoothness,

$$\begin{aligned} f(\boldsymbol{x}_{t+1}) - f(\boldsymbol{x}_t) &= f\big(\boldsymbol{x}_t - \eta \nabla f(\boldsymbol{x}_t)\big) - f(\boldsymbol{x}_t) \\ &\leq \langle \nabla f(\boldsymbol{x}_t), -\eta \nabla f(\boldsymbol{x}_t) \rangle + \frac{L}{2} \big\| \eta \nabla f(\boldsymbol{x}_t) \big\|_2^2 \\ &= -\Big(\eta - \frac{\eta^2 L}{2}\Big) \big\| \nabla f(\boldsymbol{x}_t) \big\|_2^2 \\ &\leq -\frac{\eta}{2} \big\| \nabla f(\boldsymbol{x}_t) \big\|_2^2 \end{aligned}$$

  as long as $\eta \leq 1/L$.

- Telescoping $t = 0, 1, \ldots, T-1$ gives

$$\frac{\eta}{2} \sum_{t=0}^{T-1} \big\| \nabla f(\boldsymbol{x}_t) \big\|_2^2 \leq \sum_{t=0}^{T-1} \big(f(\boldsymbol{x}_{t+1}) - f(\boldsymbol{x}_t) \big) = f(\boldsymbol{x}_0) - f(\boldsymbol{x}_T) \leq \Delta.$$

  Setting $\eta = 1/L$ finishes the proof.

Stochastic gradient descent (SGD)

Stochastic gradient descent (SGD):

$$\boldsymbol{x}_{t+1} = \boldsymbol{x}_t - \eta_t \nabla \ell(\boldsymbol{x}_t; \boldsymbol{z}_t), \qquad \boldsymbol{z}_t \sim \mathcal{M}$$

where η_t is the learning rate.

- **Unbiasedness:**

$$\mathbb{E}_{\boldsymbol{z}}[\nabla \ell(\boldsymbol{x}; \boldsymbol{z})] = \nabla f(\boldsymbol{x}).$$

- Additional assumption is needed for convergence analysis.

Definition (Bounded gradient assumption)

For any $\boldsymbol{x}, \boldsymbol{z} \in \mathbb{R}^d$, there exists some $G > 0$ such that

$$\|\nabla \ell(\boldsymbol{x}, \boldsymbol{z})\|_2 \leq G.$$

Convergence of SGD under bounded gradient

Theorem (Convergence of SGD)

Suppose $f^\star = \min_{\boldsymbol{x}} f(\boldsymbol{x}) > -\infty$ *and* $\|\nabla \ell(\boldsymbol{x}, \boldsymbol{z})\|_2 \leq G$ *for any* $\boldsymbol{x}, \boldsymbol{z} \in \mathbb{R}^d$. *Setting* $\eta = \sqrt{\frac{2\Delta}{G^2 L T}}$, *it satisfies*

$$\frac{1}{T} \sum_{t=0}^{T-1} \mathbb{E}\big\|\nabla f(\boldsymbol{x}_t)\big\|_2^2 \leq G\sqrt{\frac{2L\Delta}{T}}.$$

- SGD converges at the rate $O(1/\sqrt{T})$ in terms of the expected average squared gradient norm, which is slower than GD.
- For finite-sum problems of size n, the IFO complexity of SGD is $O(\varepsilon^{-2})$ to reach $\mathbb{E}\|\nabla f(\boldsymbol{x}^{\text{output}})\|_2^2 \leq \varepsilon$.

Convergence of SGD

- By smoothness,

$$\begin{aligned} f(\boldsymbol{x}_{t+1}) - f(\boldsymbol{x}_t) &= f\big(\boldsymbol{x}_t - \eta \nabla \ell(\boldsymbol{x}_t; \boldsymbol{z}_t)\big) - f(\boldsymbol{x}_t) \\ &\le \langle \nabla f(\boldsymbol{x}_t), -\eta \nabla \ell(\boldsymbol{x}_t; \boldsymbol{z}_t) \rangle + \frac{L}{2} \big\| \eta \nabla \ell(\boldsymbol{x}_t; \boldsymbol{z}_t) \big\|_2^2 \\ &\le -\eta \langle \nabla f(\boldsymbol{x}_t), \nabla \ell(\boldsymbol{x}_t; \boldsymbol{z}_t) \rangle + \frac{\eta^2 G^2 L}{2}. \end{aligned}$$

- Taking conditional expectation at the t-th iterate,

$$\mathbb{E}_t f(\boldsymbol{x}_{t+1}) - f(\boldsymbol{x}_t) \le -\eta \big\| \nabla f(\boldsymbol{x}_t) \big\|_2^2 + \frac{\eta^2 G^2 L}{2}.$$

- Telescoping $t = 0, 1, \ldots, T-1$ gives

$$\frac{1}{T} \mathbb{E} \sum_{t=0}^{T-1} \big\| \nabla f(\boldsymbol{x}_t) \big\|_2^2 \le \frac{(f(\boldsymbol{x}_0) - \mathbb{E} f(\boldsymbol{x}_T))}{\eta T} + \frac{\eta G^2 L}{2} \le \frac{\Delta}{\eta T} + \frac{\eta G^2 L}{2}.$$

Setting $\eta = \sqrt{\frac{2\Delta}{G^2 L T}}$ finishes the proof.

Bounded variance assumption

Definition (Bounded variance assumption)

For any $\boldsymbol{x} \in \mathbb{R}^d$, there exists some $\sigma > 0$ such that

$$\mathbb{E}_{\boldsymbol{z}} \|\nabla \ell(\boldsymbol{x}, \boldsymbol{z}) - \nabla f(\boldsymbol{x})\|_2^2 \le \sigma^2.$$

- Under unbiasedness, this assumption is equivalent to

$$\mathbb{E}_{\boldsymbol{z}} \|\nabla \ell(\boldsymbol{x}, \boldsymbol{z})\|_2^2 \le \|\nabla f(\boldsymbol{x})\|_2^2 + \sigma^2.$$

The convergence of SGD can be established under the relaxed bound variance assumption (Ghadimi and Lan, 2013):

$$\mathbb{E}\|\nabla f(\boldsymbol{x}^{\text{output}})\|_2^2 \lesssim \frac{L\Delta}{T} + \sigma\sqrt{\frac{\Delta}{LT}}$$

- By picking large enough batch size to make σ sufficiently small, the rate matches that of GD.

Can we achieve faster rate?

Variance reduction: perform SGD with a carefully designed stochastic gradient (SG) $\boldsymbol{g}_t$:

$$\boldsymbol{x}_{t+1} = \boldsymbol{x}_t - \eta_t \boldsymbol{g}_t$$

SVRG (Johnson and Zhang, 2013) assumes $(\boldsymbol{x}_0, \nabla f(\boldsymbol{x}_0))$ is a reference point,

$$\boldsymbol{g}_t = \underbrace{\nabla \ell(\boldsymbol{x}_t; \boldsymbol{z}_t)}_{\text{SG at } \boldsymbol{x}_t} - \underbrace{\underbrace{\nabla \ell(\boldsymbol{x}_0; \boldsymbol{z}_t)}_{\text{SG at } \boldsymbol{x}_0} + \underbrace{\nabla f(\boldsymbol{x}_0)}_{\text{FG at } \boldsymbol{x}_0}}_{\text{zero-mean}}$$

- Unbiased: $\mathbb{E}[\boldsymbol{g}_t] = \nabla f(\boldsymbol{x}^t)$;
- Variance:

  $$\boldsymbol{g}_t - \nabla f(\boldsymbol{x}_t) = [\nabla \ell(\boldsymbol{x}_t; \boldsymbol{z}_t) - \nabla f(\boldsymbol{x}_t)] - [\nabla \ell(\boldsymbol{x}_0; \boldsymbol{z}_t) - \nabla f(\boldsymbol{x}_0)]$$

  if the two terms are positively correlated, then variance reduction occurs, i.e. $\text{Var}[\boldsymbol{g}_t] \ll \text{Var}[\nabla \ell(\boldsymbol{x}_t; \boldsymbol{z}_t)]$.
- Update the reference $\boldsymbol{x}_0$ periodically.

Can we achieve faster rate?

Variance reduction: perform SGD with a carefully designed stochastic gradient (SG) $\boldsymbol{g}_t$:

$$\boldsymbol{x}_{t+1} = \boldsymbol{x}_t - \eta_t \boldsymbol{g}_t$$

SAGA (Defazio et al., 2014) maintains a table of stochastic gradient $\boldsymbol{g}(\boldsymbol{z})$ at each sample $\boldsymbol{z}$:

$$\boldsymbol{g}_t = \underbrace{\nabla \ell(\boldsymbol{x}_t; \boldsymbol{z}_t)}_{\text{SG at } \boldsymbol{x}_t} - \underbrace{\underbrace{\boldsymbol{g}(\boldsymbol{z}_t)}_{\text{old SG at } \boldsymbol{z}_t} + \underbrace{\frac{1}{n}\sum_{z \in \mathcal{M}} \boldsymbol{g}(\boldsymbol{z})}_{\text{average of old SGs}}}_{\text{zero-mean}},$$

$$g(\boldsymbol{z}_t) \leftarrow \nabla \ell(\boldsymbol{x}_t; \boldsymbol{z}_t)$$

For finite-sum problems of size n, the IFO complexity of SVRG/SAGA achieves the rate $O(n + n^{2/3}\varepsilon^{-1})$ to reach $\mathbb{E}\|\nabla f(\boldsymbol{x}^{\text{output}})\|_2^2 \leq \varepsilon$, which is sub-optimal.

Can we achieve the optimal rate?

Variance reduction: perform SGD with a carefully designed stochastic gradient (SG) $\boldsymbol{g}_t$:

$$\boldsymbol{x}_{t+1} = \boldsymbol{x}_t - \eta_t \boldsymbol{g}_t$$

SARAH/Spider (Nguyen et al., 2017; Fang et al., 2019) assumes $(\boldsymbol{x}_0, \nabla f(\boldsymbol{x}_0))$ is a reference point,

$$\boldsymbol{g}_t = \nabla \ell(\boldsymbol{x}_t; \boldsymbol{z}_{i_t}) - \nabla \ell(\boldsymbol{x}_{t-1}; \boldsymbol{z}_{i_t}) + \boldsymbol{g}_{t-1}$$

where $\boldsymbol{g}_0 = \nabla f(\boldsymbol{x}^0)$.

- Biased: $\mathbb{E}[\boldsymbol{g}_t] \neq \nabla f(\boldsymbol{x}_t)$;
- The stochastic gradient is recursive.

For finite-sum problems of size n, the IFO complexity of SARAH/Spider achieves the optimal rate $O(n + n^{1/2}\varepsilon^{-1})$ to reach $\mathbb{E}\|\nabla f(\boldsymbol{x}^{\text{output}})\|_2^2 \leq \varepsilon$.

Summary

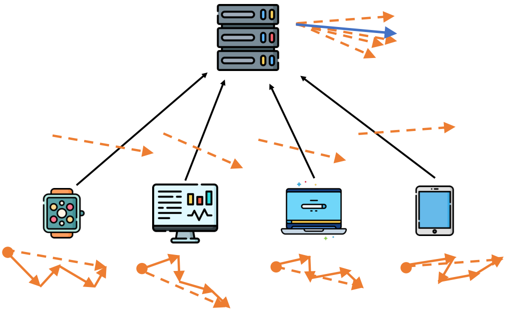
The IFO complexity to reach $\mathbb{E}\|\nabla f(\boldsymbol{x}^{\text{output}})\|_2^2 \le \varepsilon$ under smoothness.

Method	Complexity	Additional assumption
GD	$\frac{n}{\varepsilon}$	none
SGD	$\frac{1}{\varepsilon^2}$	bounded gradient or bounded variance
SVRG/SAGA	$n + \frac{n^{2/3}}{\varepsilon}$	none
SARAH/Spider	$n + \frac{n^{1/2}}{\varepsilon}$	none
Lower bound	$n + \frac{n^{1/2}}{\varepsilon}$	none

Part 1: Communication-efficient Federated Optimization via Local Methods

FedAvg

Federated Averaging (FedAvg): the first FL algorithm (McMahan et al., 2016) that alternates between local updates and global averaging.



- Also known as local SGD: the number of local updates $= E$.
- When $E = 1$, reduces to distributed SGD:

$$\boldsymbol{x}_{t+1} = \boldsymbol{x}_t - \eta \frac{1}{n} \sum_{i=1}^{n} \nabla f_i(\boldsymbol{x}^t)$$

Convergence guarantees of FedAvg

Definition (Bounded gradient dissimilarity)

There exist constants $G \geq 0$ and $B \geq 1$ such that for all $\boldsymbol{x} \in \mathbb{R}^d$:

$$\frac{1}{n}\sum_{i=1}^{n} \|\nabla f_i(\boldsymbol{x})\|^2 \leq G^2 + B^2 \|\nabla f(\boldsymbol{x})\|^2.$$

- Treating $f_i(\boldsymbol{x})$ as sampling $f(\boldsymbol{x})$, this assumption mimics the bounded variance assumptions. When $B = 1$,

$$\begin{aligned}\mathbb{E}_{i\sim[n]} \|\nabla f_i(\boldsymbol{x}) - \nabla f(\boldsymbol{x})\|_2^2 &= \frac{1}{n}\sum_{i=1}^{n} \|\nabla f_i(\boldsymbol{x})\|^2 - \|\nabla f(\boldsymbol{x})\|^2 \\ &\leq G^2.\end{aligned}$$

When $f_i = f$, set $G = 0$ and $B = 1$.

Convergence guarantees of FedAvg

Theorem (Karimireddy et al., 2019)

To achieve $\mathbb{E}\|\nabla f(\boldsymbol{x}^{\text{output}})\|^2 \leq \varepsilon$, *FedAvg takes at most an order of*

$$\frac{\sigma^2}{mE\varepsilon^2} + \frac{G}{\varepsilon^{3/2}} + \frac{B^2}{\varepsilon}$$

iterations, where σ^2 *is the local sampling variance.*

- $\frac{\sigma^2}{mE\varepsilon^2}$: error due to local stochasticity
- $\frac{G}{\varepsilon^{3/2}} + \frac{B^2}{\varepsilon}$: error due to client heterogeneity

FedAvg is sensitive to data heterogeneity

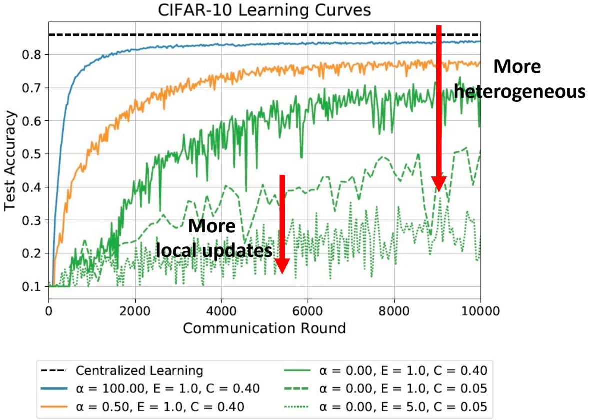
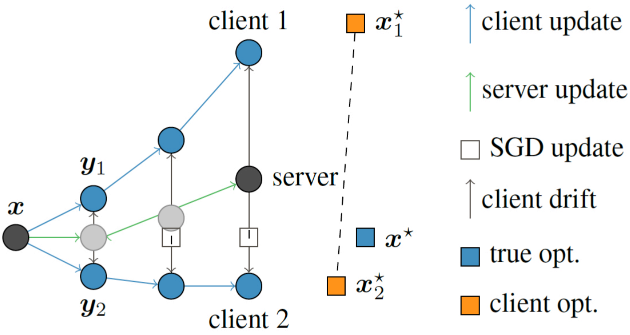


Figure credit: (Hsu et al., 2019)

The performance gets worse with more local updates for heterogenous data.

FedAvg is sensitive to data heterogeneity

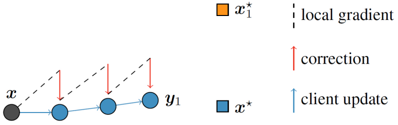


Client drift: the average of the local optima is not the global optimum!

How to design better algorithms that are more resilient to heterogeneous data?

SCAFFOLD: leveraging variance reduction

SCAFFOLD (Karimireddy et al., 2019): federated SAGA (Defazio et al., 2014)



- Client i performs K steps of SGD using local control variate c^i

$$y_t^i \leftarrow y_t^i - \eta(g_i(y_t^i) + \underbrace{c - c^i}_{\text{correction}})$$

- c: estimated update direction for server
- c_i: estimated update direction for client i

Algorithm 1 SCAFFOLD: Stochastic Controlled Averaging for federated learning

server input: initial x and c, and global step-size η_g
client i's input: c_i, and local step-size η_l
for each round $r = 1, \dots, R$ **do**
  $\boxed{\text{sample clients } \mathcal{S} \subseteq \{1, \dots, N\}}$
  communicate (x, c) to all clients $i \in \mathcal{S}$
  on client $i \in \mathcal{S}$ **in parallel do**
    initialize local model $y_i \leftarrow x$
    for $k = 1, \dots, K$ **do**
      compute mini-batch gradient $g_i(y_i)$
      $y_i \leftarrow y_i - \eta_l \boxed{(g_i(y_i) - c_i + c)}$
    end for
    $c_i^+ \leftarrow$ (i) $g_i(x)$, or (ii) $c_i - c + \frac{1}{K\eta_l}(x - y_i)$
    communicate $(\Delta y_i, \Delta c_i) \leftarrow (y_i - x, c_i^+ - c_i)$
    $c_i \leftarrow c_i^+$
  end on client
  $(\Delta x, \Delta c) \leftarrow \frac{1}{|\mathcal{S}|} \sum_{i \in \mathcal{S}} (\Delta y_i, \Delta c_i)$
  $x \leftarrow x + \eta_g \Delta x$ and $c \leftarrow c + \frac{|\mathcal{S}|}{N} \Delta c$
end for

Convergence of SCAFFOLD

Theorem (Karimireddy et al., 2019)

To achieve $\mathbb{E}\|\nabla f(\boldsymbol{x}^{\text{output}})\|^2 \le \varepsilon$, *SCAFFOLD takes at most an order of*

$$\frac{\sigma^2}{mE\varepsilon^2} + \frac{1}{\varepsilon}$$

iterations, where σ^2 *is the local sampling variance.*

- Handles arbitrary data heterogeneity: does not require the bounded dissimilarity assumption!
- Also allows client sampling; details in the paper.

FedAvg versus SCAFFOLD

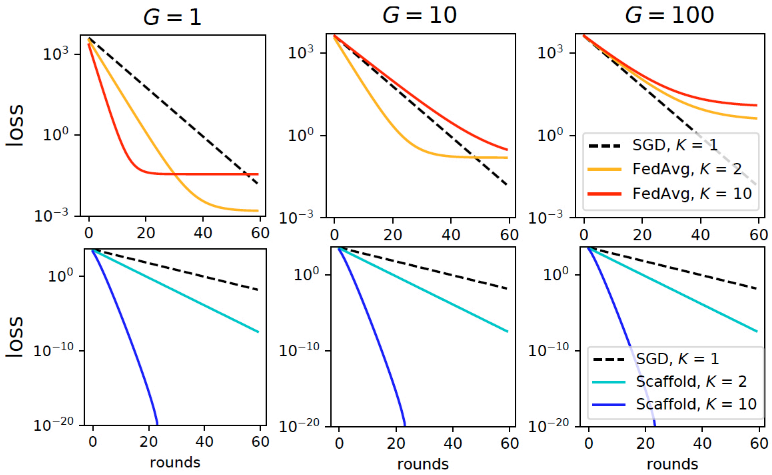


Figure credit: (Karimireddy et al., 2019)

Key takeaways and further pointers

Key takeaways:

- Local updates help improve communication efficiency
- FedAvg is sensitive to data heterogeneity
- Leverage variance reduction to deal with heterogeneity

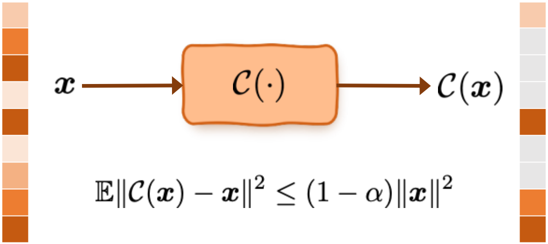
Further pointers:

- Client sampling

Part 2: Communication-compressed federated optimization

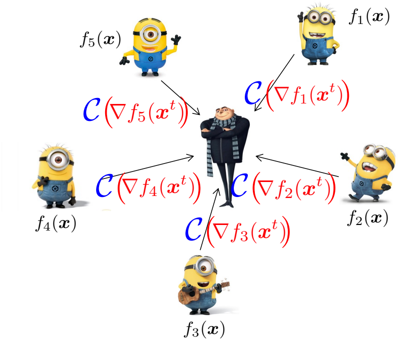
Communication compression

Communication compression is a popular approach to reduce communication cost (e.g., (Alistarh et al., 2017); (Koloskova et al., 2019)).



- **random sparsification:** $\alpha = k/d$ measures the compression ratio.
- Other examples: random quantization, top-k quantization, etc....

A prelude: what should we compress?



What about

$$\boldsymbol{x}^{t+1} = \boldsymbol{x}^t - \eta \frac{1}{n} \sum_{i=1}^{n} \mathcal{C}(\nabla f_i(\boldsymbol{x}^t))?$$

Somewhat surprisingly, *direct compression* may not work!

A counter-example

Consider $n = 3$ and let $f_i(x) = (\boldsymbol{a}_i^\top \boldsymbol{x})^2 + \frac{1}{2}\|\boldsymbol{x}\|^2$, where

$$\boldsymbol{a}_1 = (-4, 3, 3)^\top, \quad \boldsymbol{a}_2 = (3, -4, 3)^\top \quad \text{and} \quad \boldsymbol{a}_3 = (3, 3, -4)^\top.$$

- Let $\boldsymbol{x}^0 = (b, b, b)$, and the compressor be top$_1$,

$$\begin{aligned}
\nabla f_1(\boldsymbol{x}^0) = b(-15, 13, 13)^\top \quad &\longrightarrow \quad \mathcal{C}(\nabla f_1(\boldsymbol{x}^0)) = b(-15, 0, 0)^\top \\
\nabla f_2(\boldsymbol{x}^0) = b(13, -15, 13)^\top \quad &\longrightarrow \quad \mathcal{C}(\nabla f_2(\boldsymbol{x}^0)) = b(0, -15, 0)^\top \\
\nabla f_3(\boldsymbol{x}^0) = b(13, 13, -15)^\top \quad &\longrightarrow \quad \mathcal{C}(\nabla f_3(\boldsymbol{x}^0)) = b(0, 0, -15)^\top
\end{aligned}$$

- The next iteration

$$\boldsymbol{x}^1 = \boldsymbol{x}^0 - \eta \frac{1}{3} \sum_{i=1}^{3} \mathcal{C}(\nabla f_i(\boldsymbol{x}^0)) = (1 + 5\eta)\boldsymbol{x}^0,$$

and then $\boldsymbol{x}^t = (1 + 5\eta)^t \boldsymbol{x}^0$ diverges exponentially.

A better scheme: shift compression / error feedback

(Stich et al., 2018; Richtárik et al., 2021)

- The PS updates the model:

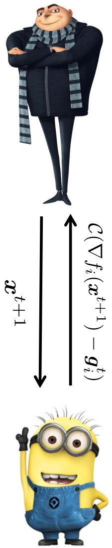
$$\boldsymbol{x}^{t+1} = \boldsymbol{x}^t - \frac{\eta}{n}\sum_{i=1}^{n} \boldsymbol{g}_i^t$$

— $\boldsymbol{g}_i^t$ is the compressed surrogate of $\nabla f_i(\boldsymbol{x}^t)$

- Clients update $\boldsymbol{g}_i^t$ with a shift compression:

$$\boldsymbol{g}_i^{t+1} = \boldsymbol{g}_i^t + \underbrace{\mathcal{C}(\nabla f_i(\boldsymbol{x}^{t+1}) - \boldsymbol{g}_i^t)}_{\text{difference compression}}$$

— $\boldsymbol{g}_i^t$ is constructed accumulatively over time



Let's revisit the example

- Let $\boldsymbol{x}^0 = (b, b, b)$, and the compressor be top_1, $\boldsymbol{g}_i^0 = \mathcal{C}(\nabla f_i(\boldsymbol{x}^0))$, and the first iteration is still $\boldsymbol{x}^1 = (1 + 5\eta)\boldsymbol{x}^0$.
- **Error feedback:**

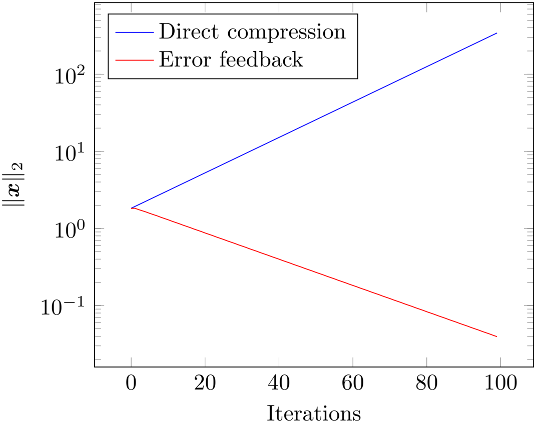
$$\nabla f_1(\boldsymbol{x}^1) - \boldsymbol{g}_1^0 = b \begin{bmatrix} -75\eta \\ 13(1+5\eta) \\ 13(1+5\eta) \end{bmatrix}$$

and as long as $\eta < 13/30$:

$$\mathcal{C}\left(\nabla f_1(\boldsymbol{x}^1) - \boldsymbol{g}_1^0\right) = b \begin{bmatrix} 0 \\ 13(1+5\eta) \\ 0 \end{bmatrix}$$

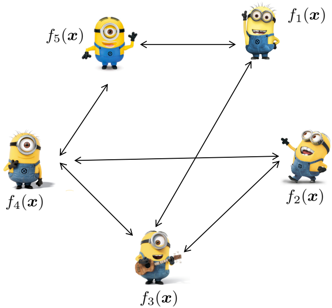
receiving information from coordinates other than the first one, leading to a better compressed gradient!

Let's revisit the example



We'll consider algorithms using shifted compression!

Case study: decentralized nonconvex optimization



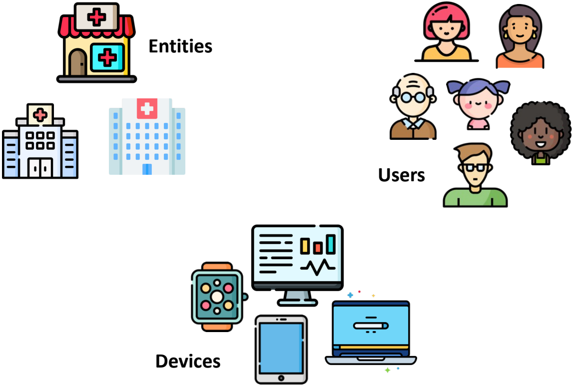
- The mixing of information is characterized by a mixing matrix $\boldsymbol{W} = [w_{ij}] \in \mathbb{R}^{n \times n}$ aligned with the network topology.
- The spectral quantity, which we call the *spectral gap*,

$$\rho \triangleq 1 - |\lambda_2(\boldsymbol{W})| \in (0, 1]$$

captures how fast information mixes over the network.

Goal: design fast-converging algorithms with communication compression

Data heterogeneity

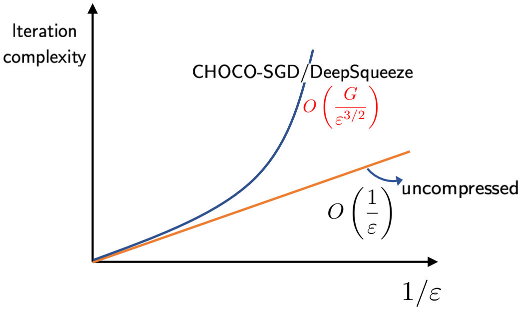


Heterogeneity measure

$$\mathbb{E}_i \| \underbrace{\nabla f_i(\boldsymbol{x})}_{\textit{local obj.}} - \underbrace{\nabla f(\boldsymbol{x})}_{\textit{global obj.}} \|^2 \leq G^2$$

— G can be unbounded!

Prior art



CHOCO-SGD (Koloskova et al., 2019) / DeepSqueeze (Tang et al., 2019):

- slow convergence rates (need more communication rounds) and
- Incompatible with heterogeneity

Can we converge at the rate $O\left(\frac{1}{\varepsilon}\right)$ under arbitrary heterogeneity?

Yes, by using gradient tracking!

Detour: DGD with gradient tracking

Centralized Gradient Descent (GD):

$$\boldsymbol{x}^t = \boldsymbol{x}^{t-1} - \eta \nabla f(\boldsymbol{x}^{t-1})$$

Constant step size, linear convergence for strongly convex problems.

Decentralized Gradient Descent (DGD):

$$\boldsymbol{x}_i^t = \underbrace{\sum\nolimits_j w_{ij} \boldsymbol{x}_j^{t-1}}_{\text{mixing}} - \eta \underbrace{\nabla f_i(\boldsymbol{x}_i^{t-1})}_{\text{local gradient}}$$

Constant step size, does not converge!

At optimal point $\boldsymbol{x}^\star$: $\nabla f(\boldsymbol{x}^\star) = \mathbf{0}$, but $\nabla f_i(\boldsymbol{x}^\star) \neq \mathbf{0}$

How do we fix this?

DGD with gradient tracking

Use dynamic average consensus (Zhu and Martinez, 2010) to track the global gradient $\boldsymbol{s}_i^t$:

$$\boldsymbol{x}_i^t = \underbrace{\sum\nolimits_j w_{ij} \boldsymbol{x}_j^{t-1}}_{\text{mixing}} - \eta \boldsymbol{s}_i^t$$

$$\boldsymbol{s}_i^t = \underbrace{\sum\nolimits_j w_{ij} \boldsymbol{s}_i^{t-1}}_{\text{mixing}} + \underbrace{\nabla f_i(\boldsymbol{x}_i^t) - \nabla f_i(\boldsymbol{x}_i^{t-1})}_{\text{gradient tracking}}$$

This trick, and other alternatives, have been used extensively to fix the non-convergence issue in decentralized optimization.

- EXTRA (Shi, Ling, Wu and Yin, 2015); NEXT (Di Lorenzo and Scutari, 2016); NIDS (Li, Shi, Yan, 2017); ADD-OPT (Xi, Xin, and Khan, 2017); DIGING (Nedic, Olshevsky, and Shi, 2017); DGD (Qu and Li, 2018);
- many, many more...

BEER: gradient tracking + shift compression

$\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_n]$: local models.

$\nabla F(\boldsymbol{X}) = [\nabla f_1(\boldsymbol{x}_1), \nabla f_2(\boldsymbol{x}_2), \cdots, \nabla f_n(\boldsymbol{x}_n)]$: local gradients.

- **model update:**

$$\boldsymbol{X}^{t+1} = \boldsymbol{X}^t + \gamma \underbrace{\boldsymbol{H}^t(\boldsymbol{W} - \boldsymbol{I})}_{\text{mixing}} - \eta \underbrace{\boldsymbol{V}^t}_{\text{gradient}}$$

  where $\boldsymbol{H}^t$ is the accumulated compressed surrogate of $\boldsymbol{X}^t$, and $\boldsymbol{V}^t$ is the global gradient estimates across the agents.

- **gradient tracking:**

$$\boldsymbol{V}^{t+1} = \boldsymbol{V}^t + \gamma \underbrace{\boldsymbol{G}^t(\boldsymbol{W} - \boldsymbol{I})}_{\text{mixing}} + \underbrace{\nabla F(\boldsymbol{X}^{t+1}) - \nabla F(\boldsymbol{X}^t)}_{\text{gradient tracking}},$$

  where $\boldsymbol{G}^t$ is the accumulated compressed surrogate of $\boldsymbol{V}^t$.

- Both $\boldsymbol{H}^t$ and $\boldsymbol{G}^t$ are updated using **shift compression**.

Theoretical convergence of BEER

Theorem (Zhao et al., NeurIPS 2022)

To achieve $\mathbb{E}\|\nabla f(\boldsymbol{x}^{\text{output}})\|^2 \le \varepsilon$, *BEER requires at most*

$$O\left(\frac{1}{\rho^3 \alpha \varepsilon}\right)$$

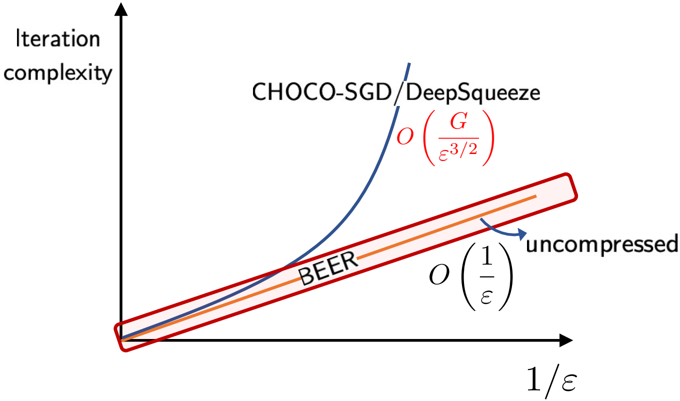
communication rounds, without the bounded heterogeneity assumption. Here, α *is the compression ratio,* β *is the spectral gap of the network.*

- Assuming constant α and ρ, the convergence rate of BEER is

$$O\left(\frac{1}{\varepsilon}\right).$$

- Can also be extended to using stochastic gradients.

Theoretical convergence of BEER



BEER converges at the rate $O\left(\frac{1}{\varepsilon}\right)$ under arbitrary heterogeneity!

BEER vs CHOCO-SGD

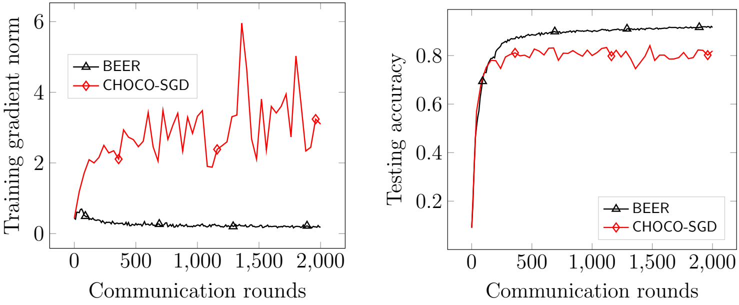


Figure: Training gradient norm and testing accuracy against communication rounds for classification on the *unshuffled* MNIST dataset using a simple neural network. Both BEER and CHOCO-SGD employ the biased gsgd_b compression with $b = 20$.

Key takeaways and further pointers

Key takeaways:

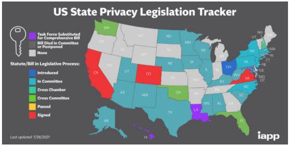
- Compression can greatly improve communication efficiency without hurting performance
- Compressing the error, not the gradient
- Accelerating decentralized optimization via gradient tracking

Further pointers:

- Biased versus unbiased compression
- Uplink and downlink compression

Part 3: Private federated optimization

A little privacy, please



Privacy guarantees are becoming increasingly critical!

Differential privacy

Differential privacy (Dwork, 2006) is a popular approach for preserving privacy in practice, and widely adopted by Google, Apple, US Census, etc.

Definition (Differential privacy (DP))

A randomized mechanism $\mathcal{M} : \mathcal{Z} \to \mathcal{R}$ satisfies (ϵ, δ)-DP, if for any two neighboring dataset $\boldsymbol{Z}, \boldsymbol{Z}_i \in \mathcal{Z}$ and any outputs $\mathbb{R} \subseteq \mathcal{R}$, it holds that

$$\mathbb{P}\big(\mathcal{M}(\boldsymbol{Z}) \in \mathbb{R}\big) \le e^{\epsilon}\mathbb{P}\big(\mathcal{M}(\boldsymbol{Z}_i) \in \mathbb{R}\big) + \delta.$$

The neighboring datasets are defined as $\boldsymbol{Z} = \{\boldsymbol{z}_1, \ldots, \boldsymbol{z}_n\}$ and $\boldsymbol{Z}_i = \{\boldsymbol{z}_1, \ldots, \boldsymbol{z}_i', \ldots, \boldsymbol{z}_n\}$, which means $\boldsymbol{Z}$ and $\boldsymbol{Z}_i$ are only different at one sample.

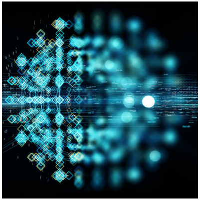
- Probabilistic definition: making it hard to tell if a data sample is used or not.
- Suitable to protect the privacy of individual records (cross-silo).

Gaussian mechanism

Gaussian mechanism: **add noise** to each sample gradient:

$$g_{\mathsf{DP}}(\boldsymbol{x}_t;\boldsymbol{z}) \leftarrow \nabla \ell(\boldsymbol{x}_t;\boldsymbol{z}) + \boldsymbol{w}_t,$$

where $\boldsymbol{w}_t \sim \mathcal{N}(0, \sigma_{\mathsf{DP}}^2 \boldsymbol{I})$.



- The noise level σ_{DP} depends on the size of $\nabla \ell(\boldsymbol{x}_t;\boldsymbol{z})$

  $\rightarrow$ requiring bounded gradient assumption.

- or, clip the gradient before adding the noise

$$g_{\mathsf{DP}}(\boldsymbol{x}_t;\boldsymbol{z}) \leftarrow \mathsf{Clip}_\tau(\nabla \ell(\boldsymbol{x}_t;\boldsymbol{z})) + \boldsymbol{w}_t$$

$\rightarrow$ harder to analyze due to clipping!

A baseline: single-machine DP-SGD

Differentially private SGD (Abadi et al., 2016) in a single-machine setting:

$$g_{\mathsf{DP}}(\boldsymbol{x}_t; \boldsymbol{z}) \leftarrow \mathsf{Clip}_\tau(\nabla \ell(\boldsymbol{x}_t; \boldsymbol{z})) + \boldsymbol{w}_t$$

Theorem (Abadi et al., 2016)

Assume the bounded gradient assumption holds. DP-SGD achieves (ϵ, δ)-DP, and the utility

$$\frac{1}{T}\sum_{t=1}^{T} \mathbb{E}\|\nabla f(\boldsymbol{x}^t)\|_2^2 \lesssim \frac{\sqrt{d \log(1/\delta)}}{m\epsilon} =: \phi_m$$

within $T \asymp \frac{m\epsilon}{\sqrt{d\log(1/\delta)}} = \phi_m^{-1}$ rounds.

- Base utility $\phi_m = \frac{\sqrt{d\log(1/\delta)}}{m\epsilon}$: lower is better.
- Stronger privacy, worse utility (accuracy), less communication.
- $\sigma_{\mathsf{DP}} \asymp \frac{G\phi_m\sqrt{T}}{d}$, G is the gradient norm: add more noise when running the algorithm longer.

Local differential privacy

Local differential privacy (McMahan et al., 2018) protect from leaking info to other agents.

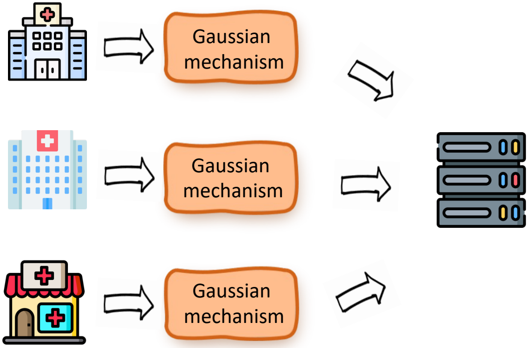
Definition (Local differential privacy (LDP))

A randomized mechanism $\mathcal{M} : \mathcal{Z} \to \mathcal{R}$ satisfies (ϵ, δ)-LDP for client i, if for any two neighboring dataset $\boldsymbol{Z}, \boldsymbol{Z}_i \in \mathcal{Z}$ and any outputs $\mathbb{R} \subseteq \mathcal{R}$, it holds that

$$\mathbb{P}\big(\mathcal{M}(\boldsymbol{Z}) \in \mathbb{R}\big) \leq e^{\epsilon}\mathbb{P}\big(\mathcal{M}(\boldsymbol{Z}_i) \in \mathbb{R}\big) + \delta.$$

The neighboring datasets are defined as $\boldsymbol{Z} = \{\boldsymbol{z}_1, \dots, \boldsymbol{z}_n\}$ and $\boldsymbol{Z}_i = \{\boldsymbol{z}_1, \dots, \boldsymbol{z}_i', \dots, \boldsymbol{z}_n\}$, which means $\boldsymbol{Z}$ and $\boldsymbol{Z}_i$ are only different at agent i.

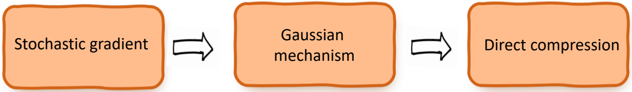
Protecting privacy via Gaussian mechanism



Introducing local differential privacy to guarantee the client privacy

— used by Google, Apple, etc in products

Warm-up: a direct compression approach (CDP-SGD)



Theorem (Li et al., NeurIPS 2022)

Assume the bounded gradient assumption holds. CDP-SGD achieves (ϵ, δ)*-LDP, and the utility*

$$\frac{1}{T}\sum_{t=1}^{T} \mathbb{E}\|\nabla f(\boldsymbol{x}_t)\|_2^2 \lesssim \frac{1}{\sqrt{\alpha n}} \cdot \phi_m,$$

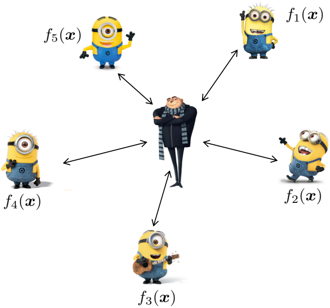
within communication complexity on the order of

$$dn^{3/2}\alpha^{3/2}\phi_m^{-1} + \alpha n d \phi_m^{-2}.$$

- Larger $\phi_m = \frac{\sqrt{d \log(1/\delta)}}{m\epsilon}$ gives stronger privacy, worse accuracy, fewer communication.
- Caveat: the communication complexity is $O(\phi_m^{-2})$ when the local data size m is dominating.

Better compression and compute: a unified framework?

- **Compression:** shift compression with many options, e.g. sparsification or quantization
- **Computation:** stochastic local gradient estimators with many options, e.g. SGD, SVRG or SAGA



Can we develop a unified framework for private FL with compression, with a characterization of the privacy-utility-communication trade-off?

SoteriaFL: a unified framework for compressed private FL

Local gradient estimator ⇨ Gaussian mechanism ⇨ Shift compression ⇨ Shift update

Highlights of SoteriaFL:

- Flexible local gradient estimators
- Protect local data privacy
- State-of-the-art shift compression scheme
- Privacy-utility-communication trade-offs

Performance of SoteriaFL

Theorem (Li et al., NeurIPS 2022)

Assume the bounded gradient assumption holds. When $n \geq 1/\alpha^3$, SoteriaFL—with SGD, GD, SVRG, SAGA—achieves (ϵ, δ)-LDP, and the utility

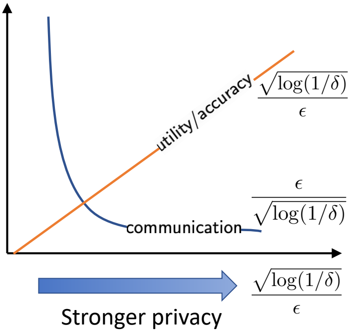
$$\frac{1}{T}\sum_{t=1}^{T} \mathbb{E}\|\nabla f(\boldsymbol{x}^t)\|_2^2 \lesssim \frac{1}{\sqrt{\alpha n}} \cdot \phi_m$$

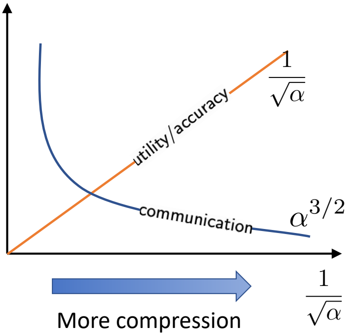
with communication complexity on the order of

$$d n^{3/2} \alpha^{3/2} \phi_m^{-1}.$$

- Communication complexity is *linear* in ϕ_m^{-1}, better than CDP-SGD!
- This analysis applies to unbiased compressions, and adapts to other gradient estimators too.

Privacy-utility-communication trade-off





- Stronger privacy, worse accuracy, fewer communication
- More compression, worse accuracy, fewer communication

Numerical experiments

Compression preserves privacy at a better communication complexity.

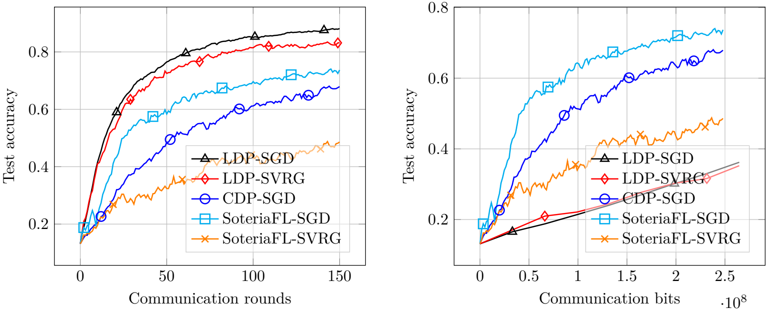
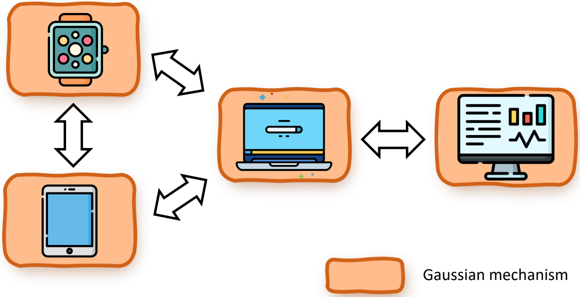


Figure: Shallow NN training on the MNIST dataset under $(1, 10^{-3})$-LDP.

LDP meets decentralized ML



Introducing local differential privacy in BEER to guarantee client privacy

PORTER: BEER meets differential privacy

Theorem (Li and Chi, 2023)

Assuming bounded gradient assumption holds. PORTER *achieves* (ϵ, δ)*-LDP, and the utility*

$$\frac{1}{T}\sum_{t=1}^{T}\mathbb{E}\|\nabla f(\boldsymbol{x}_t)\|_2^2 \lesssim \frac{1}{(1-\alpha)^{8/3}\rho^{4/3}} \cdot \phi_m$$

within communication complexity on the order of ϕ_m^{-2}*. Here,* α *is the compression ratio,* β *is the spectral gap of the network.*

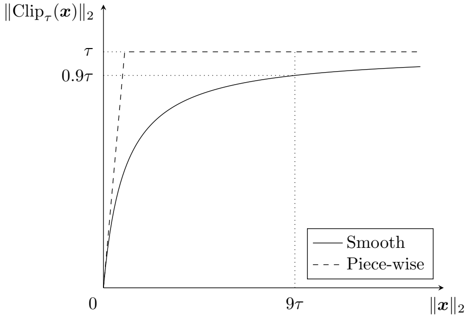
- Captures the trade-off with network connectivity.
- Communication complexity degenerates to ϕ_m^{-2}, due to dealing with the decentralized setting.

Getting rid of the bounded gradient assumption?

Definition (Clipping operator)

$$\mathsf{Clip}_\tau(\boldsymbol{x}) = \frac{\tau}{\tau + \|\boldsymbol{x}\|_2}\boldsymbol{x}$$

- The norm of a clipped vector is bounded by τ, i.e. $\left\|\mathsf{Clip}_\tau(\boldsymbol{x})\right\|_2 \leq \tau$.
- Can also use a hard thresholding operator for clipping.



Clipping is widely used in practice

One stone, two birds: clipping is widely used for two reasons (and they differ when using mini-batches).

- Privacy-preserving via per-sample clipping:

$$g_{\mathsf{DP}}(\boldsymbol{x}_t; \boldsymbol{z}) \leftarrow \frac{1}{|\mathcal{I}_t|} \sum_{\boldsymbol{z} \in \mathcal{I}_t} \mathsf{Clip}_\tau(\nabla \ell(\boldsymbol{x}_t; \boldsymbol{z})) + \boldsymbol{w}_t$$

- Stabilize training via per-batch clipping:

$$g_{\mathsf{GC}}(\boldsymbol{x}_t; \boldsymbol{z}) \leftarrow \mathsf{Clip}_\tau\left(\frac{1}{|\mathcal{I}_t|} \sum_{\boldsymbol{z} \in \mathcal{I}_t} \nabla \ell(\boldsymbol{x}_t; \boldsymbol{z})\right)$$

How does clipping impact the performance of federated optimization?

Let's take a detour to understand clipping!

Understanding gradient clipping with batch gradient

Clipping only impacts the size of the gradient, but not the direction.

$$\boldsymbol{x}_{t+1} = \boldsymbol{x}_{t+1} - \eta_t \mathsf{Clip}_\tau(\nabla f(\boldsymbol{x}_t))$$

- Define $\delta_t = \frac{\tau}{\tau + \|\nabla f(\boldsymbol{x}_t)\|_2}$.

$$\begin{aligned} f(\boldsymbol{x}_{t+1}) - f(\boldsymbol{x}_t) &= f\Big(\boldsymbol{x}_t - \eta_t \mathsf{Clip}_\tau(\nabla f(\boldsymbol{x}_t))\Big) - f(\boldsymbol{x}_t) \\ &\le \langle \nabla f(\boldsymbol{x}_t), -\eta_t \mathsf{Clip}_\tau(\nabla f(\boldsymbol{x}_t)) \rangle + \frac{L}{2}\big\|\eta_t \mathsf{Clip}_\tau(\nabla f(\boldsymbol{x}_t))\big\|_2^2 \\ &= -\eta_t \delta_t \langle \nabla f(\boldsymbol{x}_t), \nabla f(\boldsymbol{x}_t) \rangle + \frac{\eta_t^2 \delta_t^2 L}{2}\big\|\nabla f(\boldsymbol{x}_t)\big\|_2^2 \\ &= -\Big(\eta_t \delta_t - \frac{\eta_t^2 \delta_t^2 L}{2}\Big)\big\|\nabla f(\boldsymbol{x}_t)\big\|_2^2 \\ &\le -\frac{\eta_t \delta_t}{2}\big\|\nabla f(\boldsymbol{x}_t)\big\|_2^2 \end{aligned}$$

as long as $\eta_t \delta_t < 1/L$.

Gradient clipping: a contradiction argument

When $\|\nabla f(\boldsymbol{x}_t)\|_2 \geq \nu$,

$$\begin{aligned}
\frac{\delta_t}{2}\left\|\nabla f(\boldsymbol{x}_t)\right\|_2^2 &= \frac{1}{2}\frac{\tau\left\|\nabla f(\boldsymbol{x}_t)\right\|_2^2}{\tau + \|\nabla f(\boldsymbol{x}_t)\|_2} \\
&\overset{\text{(i)}}{\geq} \frac{\tau}{\tau+\nu} \cdot \frac{\nu^2}{2} \\
&\geq \frac{\tau}{\max\{\tau,\nu\}} \cdot \frac{\nu^2}{4}
\end{aligned}$$

where (i) holds since $h(x) = \frac{x^2}{c+x}$ is convex and increases monotonically when $x \geq 0$. Then, choose any $\tau \geq \nu$, the function value decrease can be bounded by

$$f(\boldsymbol{x}_{t+1}) - f(\boldsymbol{x}_t) \leq -\frac{\nu^2}{4L},$$

which can not decrease for more than $T = O(\frac{L\Delta}{\nu^2})$ iterations.

Gradient clipping in the decentralized setting

- Let's consider a toy example:
  - Number of agents $n = 3$, problem dimension $d = 1$
  - Local models are $x_1 = x_2 = x_3 = x^\star$
  - Local gradients are $g_1 = 8, g_2 = -2, g_3 = -6$
- The global gradient is

$$g = \tfrac{1}{3}(g_1 + g_2 + g_3) = 0.$$

- Apply $\mathrm{Clip}_2(\cdot)$, the global gradient becomes

$$g' = \tfrac{1}{3}\big(\mathrm{Clip}_2(g_1)+\mathrm{Clip}_2(g_2)+\mathrm{Clip}_2(g_3)\big) = \tfrac{1}{3}(1.6-1-1.25) = -0.22.$$

Definition (Bounded dissimilarity)

The local and global objectives satisfy the following:

$$\|\nabla f_i(\boldsymbol{x}) - \nabla f(\boldsymbol{x})\|_2 \le \frac{1}{12}\|\nabla f(\boldsymbol{x})\|_2.$$

PORTER with per-batch clipping

Theorem (Li and Chi, 2023)

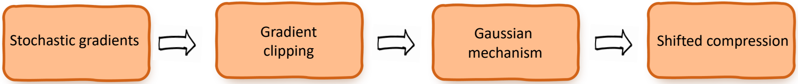
Assuming bounded local gradient variance and bounded dissimilarity assumptions hold. PORTER with gradient clipping achieves

$$\min_{t \in [T]} \mathbb{E}\big\|\nabla f(\boldsymbol{x}_t)\big\|_2 \lesssim \frac{1}{(1-\alpha)^{\frac{4}{3}} \rho^{\frac{2}{3}}} \cdot \frac{1}{T^{1/2}}$$

under appropriate parameter choices and large enough batch size.

- Matches the rate $O(1/T^{1/2})$ of centralized SGD as long as the mini-batch size is large enough and the local datasets are not too dissimilar.
- First convergence guarantee of decentralized optimization with gradient clipping and communication compression.

PORTER with per-sample clipping



Theorem (Li and Chi, 2023)

Assuming bounded local gradient variance and bounded dissimilarity assumptions hold. PORTER achieves (ϵ, δ)-LDP, and the utility

$$\min_{t \in [T]} \mathbb{E}\|\nabla f(\boldsymbol{x}_t)\|_2 \lesssim \frac{1}{(1-\alpha)^{8/3}\rho^{4/3}} \cdot \phi_m^{1/2}$$

within communication rounds ϕ_m^{-2}.

- Dependencies on mixing rate and compression match previous results.

Numerical experiments

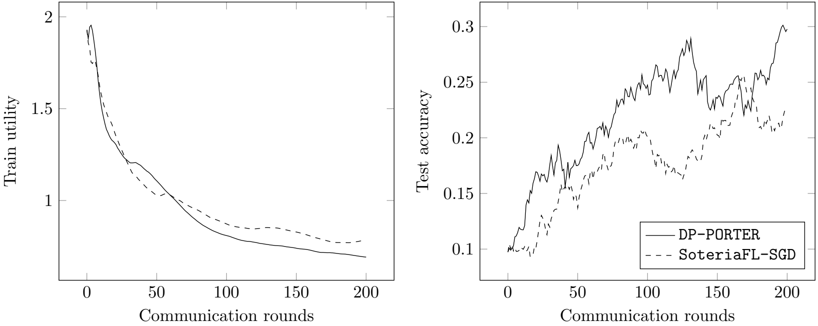
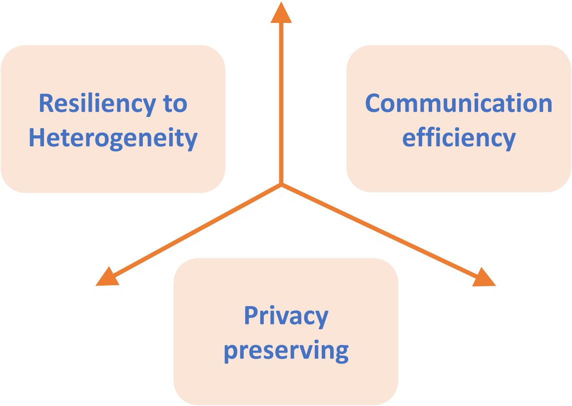


Figure: Shallow NN training on the MNIST dataset under $(10^{-2}, 10^{-3})$-LDP. Both PORTER and SoteriaFL-SGD employ random$_{2583}$ compression.

Concluding remarks

Summary



Federated optimization: let's make it efficient, resilient and private!

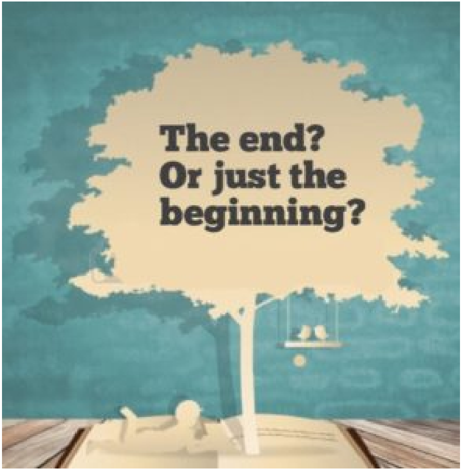
Key algorithmic pillars and trade-offs

It's all about trade-offs:

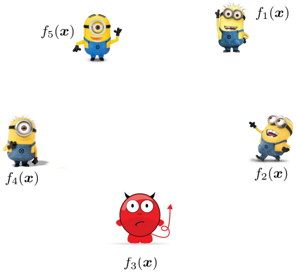
- Computation
- Communication
- Privacy
- Performance

Algorithmic ideas to probe the trade-offs:

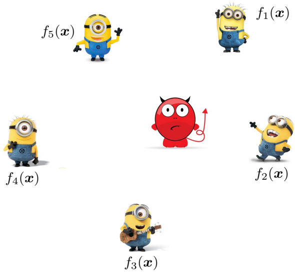
- Local updates
- Compression
- Variance reduction
- Error feedback
- Gradient tracking
- Differential privacy
- ...


The end?
Or just the
beginning?

Robustness to adversary



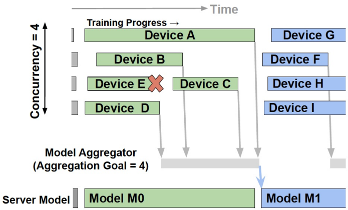
adversarial client



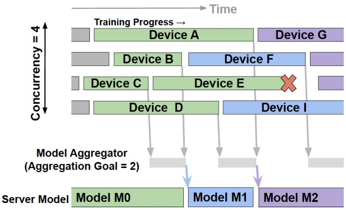
Man-in-the-middle

Robust algorithms that are oblivious to adversarial clients/attack?

Asynchronous updates



Synchronous update

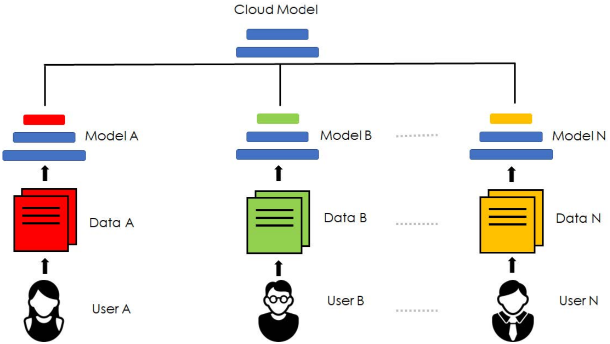


Asynchronous update

Credit: (Huba et al., 2022)

Asynchronous updates to the rescue!

Personalization

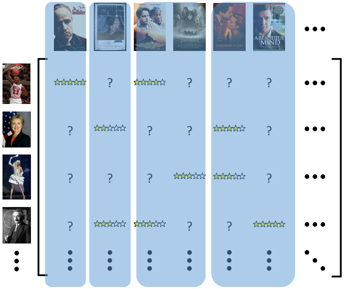


Credit: (Arivazhagan et al., 2019)

Shared the representation, personalize the prediction

Vertical FL

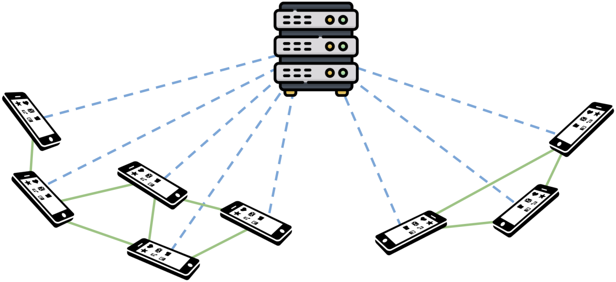
Horizontal FL
sample-distributed



Vertical FL
feature-distributed

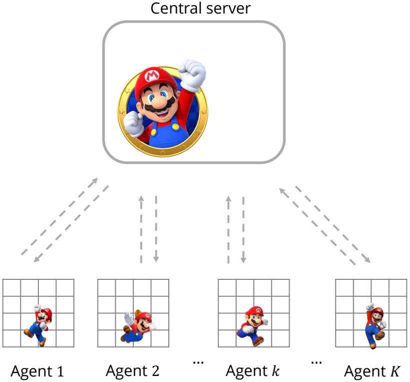
How to design efficient algorithms for feature-distributed data?

Semi-decentralized topology



Can we combining the best of worlds?

RL meets federated learning



Federated reinforcement learning: enables multiple agents to collaboratively learn a global model without sharing datasets.

Reference I

Disclaimer: this straw-man list is by no means exhaustive (in fact, it is quite the opposite given the fast pace of the field), and biased towards materials most related to this tutorial; readers are invited to further delve into the references therein to gain a more complete picture.

Monographs:

- Kairouz, Peter, et al. "Advances and open problems in federated learning." Foundations and Trends in Machine Learning 14.1–2 (2021): 1-210.
- Konečný, Jakub, et al. "Federated optimization: Distributed machine learning for on-device intelligence." arXiv preprint arXiv:1610.02527 (2016).
- Li, Tian, et al. "Federated learning: Challenges, methods, and future directions." IEEE Signal Processing Magazine 37.3 (2020): 50-60.
- Wang, Jianyu, et al. "A field guide to federated optimization." arXiv preprint arXiv:2107.06917 (2021).

Primer on nonconvex optimization:

- Ghadimi, Saeed, and Guanghui Lan. "Stochastic first-and zeroth-order methods for nonconvex stochastic programming." SIAM Journal on Optimization 23.4 (2013): 2341-2368.

Reference II

- Bottou, Léon, Frank E. Curtis, and Jorge Nocedal. "Optimization methods for large-scale machine learning." SIAM review 60.2 (2018): 223-311.
- Reddi, Sashank J., et al. "Stochastic variance reduction for nonconvex optimization." International conference on machine learning. PMLR, 2016.
- Fang, Cong, et al. "Spider: Near-optimal non-convex optimization via stochastic path-integrated differential estimator." Advances in Neural Information Processing Systems 31 (2018).

Local methods:

- McMahan, Brendan, et al. "Communication-efficient learning of deep networks from decentralized data." Artificial Intelligence and Statistics, 2017.
- Li, Xiang, et al. "On the Convergence of FedAvg on Non-IID Data." International Conference on Learning Representations.
- Woodworth, Blake, et al. "Is local SGD better than minibatch SGD?." International Conference on Machine Learning, 2020.
- Karimireddy, Sai Praneeth, et al. "Scaffold: Stochastic controlled averaging for federated learning." International Conference on Machine Learning, 2020.

Reference III

- Zhao, Haoyu, Zhize Li, and Peter Richtárik. "FedPAGE: A fast local stochastic gradient method for communication-efficient federated learning." arXiv preprint arXiv:2108.04755 (2021).

Communication compression:

- Bernstein, Jeremy, et al. "signSGD: Compressed optimisation for non-convex problems." International Conference on Machine Learning. PMLR, 2018.
- Karimireddy, Sai Praneeth, et al. "Error feedback fixes signSGD and other gradient compression schemes." International Conference on Machine Learning. PMLR, 2019.
- Richtárik, Peter, Igor Sokolov, and Ilyas Fatkhullin. "EF21: A new, simpler, theoretically better, and practically faster error feedback." Advances in Neural Information Processing Systems 34 (2021): 4384-4396.
- Tang, Hanlin, Xiangru Lian, Ming Yan, Ce Zhang, and Ji Liu. "D^2: Decentralized training over decentralized data." In International Conference on Machine Learning, pp. 4848-4856. PMLR, 2018.
- Koloskova, Anastasia, Sebastian Stich, and Martin Jaggi. "Decentralized stochastic optimization and gossip algorithms with compressed communication." In International Conference on Machine Learning, pp. 3478-3487. PMLR, 2019.

Reference IV

- Zhao, Haoyu, et al. "BEER: Fast $O(1/T)$ Rate for Decentralized Nonconvex Optimization with Communication Compression." Advances in Neural Information Processing Systems, 2022.

Differentially-private federated optimization:

- Abadi, Martin, et al. "Deep learning with differential privacy." Proceedings of the 2016 ACM SIGSAC conference on computer and communications security. 2016.
- Agarwal, Naman, et al. "cpSGD: Communication-efficient and differentially-private distributed SGD." Advances in Neural Information Processing Systems 31 (2018).
- McMahan, H. Brendan, et al. "Learning Differentially Private Recurrent Language Models." International Conference on Learning Representations.
- Li, Zhize, et al. "SoteriaFL: A Unified Framework for Private Federated Learning with Communication Compression." Advances in Neural Information Processing Systems, 2022.
- Li, Boyue, and Yuejie Chi. "Convergence and Privacy of Decentralized Nonconvex Optimization with Gradient Clipping and Communication Compression." arXiv preprint arXiv:2305.09896 (2023).

Thank you!

https://users.ece.cmu.edu/~yuejiec