INTRODUCTION TO FEDERATED LEARNING

Aurélien Bellet (Inria)

Federated Learning Winter School
November 24, 2020

OUTLINE OF THE TALK

1. What is Federated Learning?

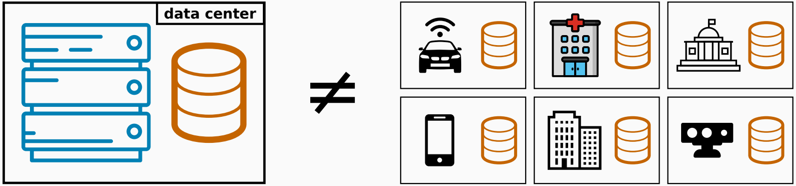
2. A baseline algorithm: FedAvg

3. Some challenges in Federated Learning

4. Wrapping up

What is Federated Learning?

- The standard setting in Machine Learning (ML) considers a centralized dataset processed in a tightly integrated system
- But in the real world data is often decentralized across many parties

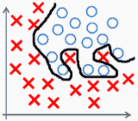


WHY CAN'T WE JUST CENTRALIZE THE DATA?

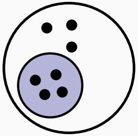
1. Sending the data may be too costly
    - Self-driving cars are expected to generate several TBs of data a day
    - Some wireless devices have limited bandwidth/power
2. Data may be considered too sensitive
    - We see a growing public awareness and regulations on data privacy
    - Keeping control of data can give a competitive advantage in business and research

HOW ABOUT EACH PARTY LEARNING ON ITS OWN?

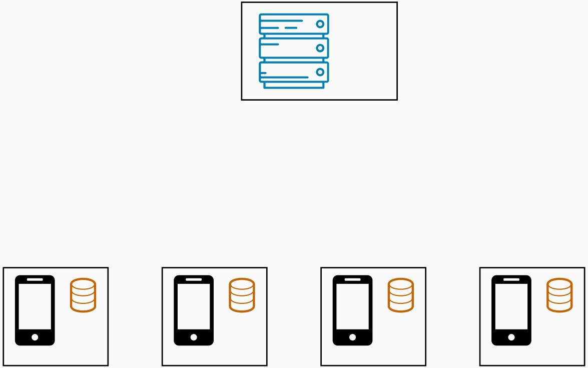
1. The local dataset may be too small
   - Sub-par predictive performance (e.g., due to overfitting)
   - Non-statistically significant results (e.g., medical studies)



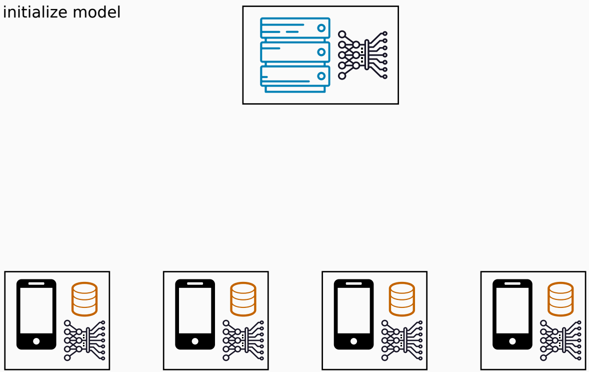
2. The local dataset may be biased
   - Not representative of the target distribution



- **Federated Learning (FL)** aims to collaboratively train a ML model while keeping the data decentralized

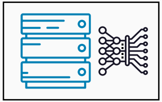


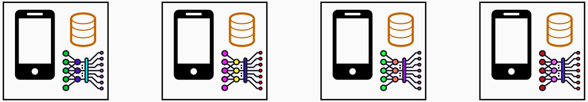
- **Federated Learning (FL)** aims to collaboratively train a ML model while keeping the data decentralized



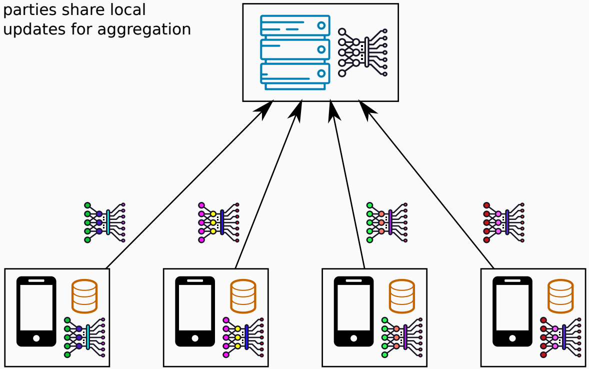
- **Federated Learning (FL)** aims to collaboratively train a ML model while keeping the data decentralized

each party makes an update using its local dataset

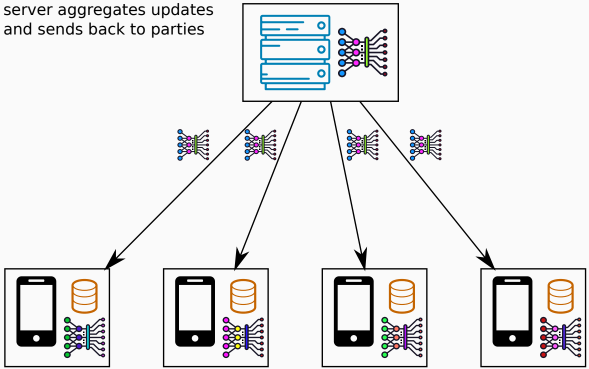




- **Federated Learning (FL)** aims to collaboratively train a ML model while keeping the data decentralized

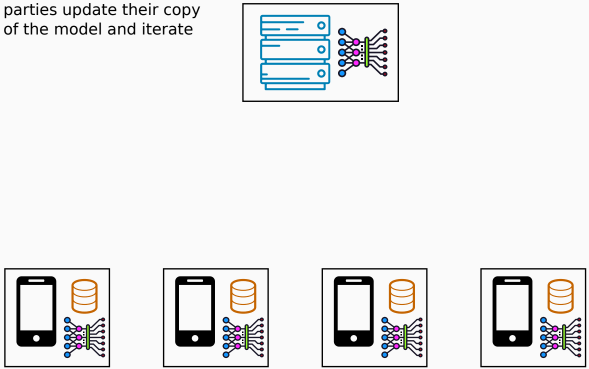


- **Federated Learning (FL)** aims to collaboratively train a ML model while keeping the data decentralized



A BROAD DEFINITION OF FEDERATED LEARNING

- **Federated Learning (FL)** aims to collaboratively train a ML model while keeping the data decentralized



- We would like the final model to be as good as the centralized solution (ideally), or at least better than what each party can learn on its own

KEY DIFFERENCES WITH DISTRIBUTED LEARNING

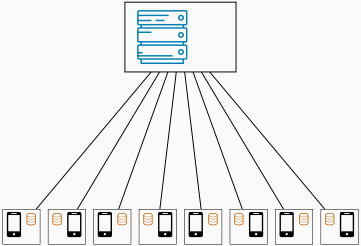
Data distribution

- In distributed learning, data is centrally stored (e.g., in a data center)
  - The main goal is just to train faster
  - We control how data is distributed across workers: usually, it is distributed uniformly at random across workers
- In FL, data is naturally distributed and generated locally
  - Data is **not** independent and identically distributed (non-i.i.d.), and it is imbalanced

Additional challenges that arise in FL

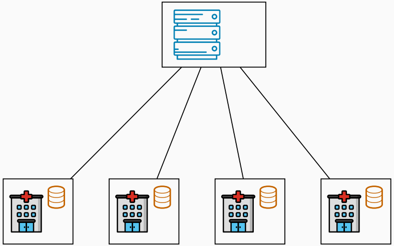
- Enforcing privacy constraints
- Dealing with the possibly limited reliability/availability of participants
- Achieving robustness against malicious parties
- ...

Cross-device FL



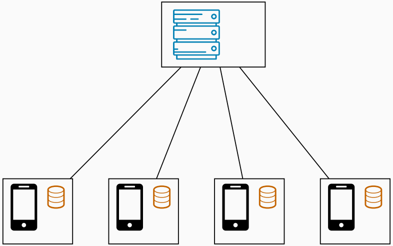
- Massive number of parties (up to 10^{10})
- Small dataset per party (could be size 1)
- Limited availability and reliability
- Some parties may be malicious

Cross-silo FL



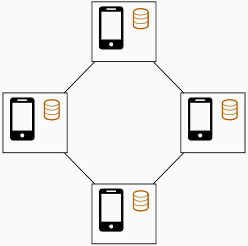
- 2-100 parties
- Medium to large dataset per party
- Reliable parties, almost always available
- Parties are typically honest

Server-orchestrated FL



- Server-client communication
- Global coordination, global aggregation
- Server is a single point of failure and may become a bottleneck

Fully decentralized FL



- Device-to-device communication
- No global coordination, local aggregation
- Naturally scales to a large number of devices

FEDERATED LEARNING IS A BOOMING TOPIC

- 2016: the term FL is first coined by Google researchers; 2020: more than 1,000 papers on FL in the first half of the year (compared to just 180 in 2018)[1]
- We have already seen some real-world deployments by companies and researchers
- Several open-source libraries are under development: PySyft, TensorFlow Federated, FATE, Flower, Substra...
- FL is highly multidisciplinary: it involves machine learning, numerical optimization, privacy & security, networks, systems, hardware...

This is all a bit hard to keep up with!

[1] https://www.forbes.com/sites/robtoews/2020/10/12/the-next-generation-of-artificial-intelligence/

A baseline algorithm: FedAvg

BASIC NOTATIONS

- We consider a set of K parties (clients)
- Each party k holds a dataset $\mathcal{D}_k$ of n_k points
- Let $\mathcal{D} = \mathcal{D}_1 \cup \cdots \cup \mathcal{D}_K$ be the joint dataset and $n = \sum_k n_k$ the total number of points
- We want to solve problems of the form $\min_{\theta \in \mathbb{R}^p} F(\theta; \mathcal{D})$ where:

$$F(\theta; \mathcal{D}) = \sum_{k=1}^{K} \frac{n_k}{n} F_k(\theta; \mathcal{D}_k) \quad \text{and} \quad F_k(\theta; \mathcal{D}_k) = \sum_{d \in \mathcal{D}_k} f(\theta; d)$$

- $\theta \in \mathbb{R}^p$ are model parameters (e.g., weights of a logistic regression or neural network)
- This covers a broad class of ML problems formulated as empirical risk minimization

FEDAVG (AKA LOCAL SGD) [McMahan et al., 2017]

Algorithm FedAvg (server-side)

Parameters: client sampling rate ρ

initialize θ
for each round $t = 0, 1, \dots$ **do**
  $\mathcal{S}_t \leftarrow$ random set of $m = \lceil \rho K \rceil$ clients
  for each client $k \in \mathcal{S}_t$ in parallel **do**
    $\theta_k \leftarrow \text{ClientUpdate}(k, \theta)$
  $\theta \leftarrow \sum_{k \in \mathcal{S}_t} \frac{n_k}{n} \theta_k$

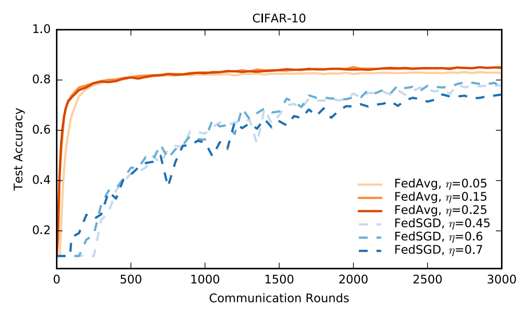
Algorithm ClientUpdate(k, θ)

Parameters: batch size B, number of local steps L, learning rate η

for each local step $1, \dots, L$ **do**
  $\mathcal{B} \leftarrow$ mini-batch of B examples from $\mathcal{D}_k$
  $\theta \leftarrow \theta - \frac{n_k}{B} \eta \sum_{d \in \mathcal{B}} \nabla f(\theta; d)$
send θ to server

- For $L = 1$ and $\rho = 1$, it is equivalent to classic parallel SGD: updates are aggregated and the model synchronized at each step
- For $L > 1$: each client performs multiple local SGD steps before communicating

FEDAVG (AKA LOCAL SGD) [McMahan et al., 2017]



- FedAvg with $L > 1$ allows to reduce the number of communication rounds, which is often the bottleneck in FL (especially in the cross-device setting)
- It empirically achieves better generalization than parallel SGD with large mini-batch
- Convergence to the optimal model can be guaranteed for i.i.d. data [Stich, 2019] [Woodworth et al., 2020] but issues arise in strongly non-i.i.d. case (more on this later)

FULLY DECENTRALIZED SETTING

- We can derive algorithms similar to FedAvg for the fully decentralized setting, where parties do not rely on a server for aggregating updates
- Let $G = (\{1, \dots, K\}, E)$ be a connected undirected graph where nodes are parties and an edge $\{k, l\} \in E$ indicates that k and l can exchange messages
- Let $W \in [0, 1]^{K \times K}$ be a symmetric, doubly stochastic matrix such that $W_{k,l} = 0$ if and only if $\{k, l\} \notin E$
- Given models $\Theta = [\theta_1, \dots, \theta_K]$ for each party, $W\Theta$ corresponds to a weighted aggregation among neighboring nodes in G:

$$[W\Theta]_k = \sum_{l \in \mathcal{N}_k} W_{k,l}\theta_l, \quad \text{where } \mathcal{N}_k = \{l : \{k, l\} \in E\}$$

Fully decentralized (local) SGD [Lian et al., 2017, Koloskova et al., 2020b]

Algorithm Fully decentralized SGD (run by party k)

Parameters: batch size B, learning rate η, sequence of matrices $W^{(t)}$

initialize $\theta_k^{(0)}$

for each round $t = 0, 1, \dots$ **do**

$\mathcal{B} \leftarrow$ mini-batch of B examples from $\mathcal{D}_k$

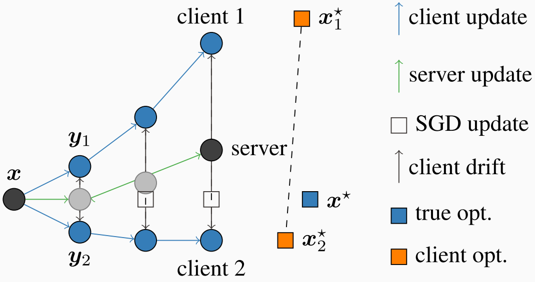
$\theta_k^{(t+\frac{1}{2})} \leftarrow \theta_k^{(t)} - \frac{n_k}{B}\eta \sum_{d \in \mathcal{B}} \nabla f(\theta_k^{(t)}; d)$

$\theta_k^{(t+1)} \leftarrow \sum_{l \in \mathcal{N}_k^{(t)}} W_{k,l}^{(t)} \theta_l^{(t+\frac{1}{2})}$

- Decentralized SGD alternates between local updates and local aggregation
- Doing multiple local steps is equivalent to choosing $W^{(t)} = I_n$ in some of the rounds
- The convergence rate depends on the topology (the more connected, the faster)

Some challenges in Federated Learning

1. Dealing with non-i.i.d. data



(Figure taken from [Karimireddy et al., 2020])

- When local datasets are non-i.i.d., FedAvg suffers from client drift
- To avoid this drift, one must use fewer local updates and/or smaller learning rates, which hurts convergence

THEORETICAL CONVERGENCE RATES FOR FEDAVG

- Analyzing the convergence rate of FL algorithms on non-i.i.d. data involves some assumption about how the local cost functions $F_1, \ldots, F_k$ are related
- For instance, one can assume that there exists constants $G \geq 0$ and $B \geq 1$ such that

$$\forall \theta : \quad \frac{1}{K} \sum_{k=1}^{K} \|\nabla F_k(\theta; \mathcal{D}_k)\|^2 \leq G^2 + B^2 \|\nabla F(\theta; \mathcal{D})\|^2$$

- FedAvg without client sampling reaches ϵ accuracy with $O(\frac{1}{KL\epsilon^2} + \frac{G}{\epsilon^{3/2}} + \frac{B^2}{\epsilon})$, which is slower than the $O(\frac{1}{KL\epsilon^2} + \frac{1}{\epsilon})$ of parallel SGD with large batch [Karimireddy et al., 2020]

SCAFFOLD: CORRECTING LOCAL UPDATES [KARIMIREDDY ET AL., 2020]

Algorithm Scaffold (server-side)

Parameters: client sampling rate ρ, global learning rate η_g

initialize θ, $c = c_1, \ldots, c_K = 0$

for each round $t = 0, 1, \ldots$ **do**

$\mathcal{S}_t \leftarrow$ random set of $m = \lceil \rho K \rceil$ clients

for each client $k \in \mathcal{S}_t$ in parallel **do**

$(\Delta\theta_k, \Delta c_k) \leftarrow \text{ClientUpdate}(k, \theta, \text{c})$

$\theta \leftarrow \theta + \frac{\eta_g}{m} \sum_{k \in \mathcal{S}_t} \Delta\theta_k$

$c \leftarrow c + \frac{1}{K} \sum_{k \in \mathcal{S}_t} \Delta c_k$

Algorithm ClientUpdate(k, θ, c)

Parameters: batch size B, # of local steps L, local learning rate η_l

Initialize $\theta_k \leftarrow \theta$

for each local step $1, \ldots, L$ **do**

$\mathcal{B} \leftarrow$ mini-batch of B examples from $\mathcal{D}_k$

$\theta_k \leftarrow \theta_k - \eta_l(\frac{n_k}{B} \sum_{d \in \mathcal{B}} \nabla f(\theta; d) - c_k + c)$

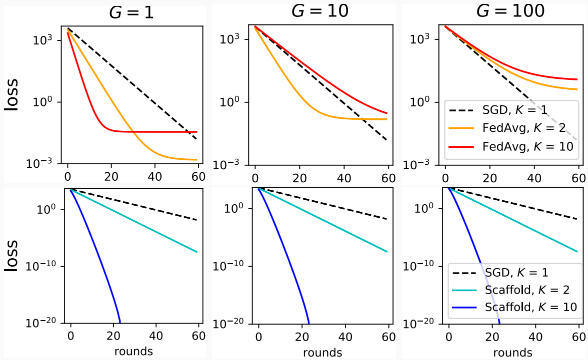
$c_k^+ \leftarrow c_k - c + \frac{1}{L\eta_l}(\theta - \theta_k)$

send $(\theta_k - \theta, c_k^+ - c_k)$ to server

$c_k \leftarrow c_k^+$

- Correction terms $c_1, \ldots, c_K$ approximate an ideal unbiased update
- Can show convergence rates which beat parallel SGD

SCAFFOLD: CORRECTING LOCAL UPDATES [Karimireddy et al., 2020]



- FedAvg becomes slower than parallel SGD for strongly non-i.i.d. data (large G)
- Scaffold can often do better in such settings
- Other relevant approach: FedProx [Li et al., 2020b]

- Learning from non-i.i.d. data is difficult/slow because each party wants the model to go in a particular direction
- If data distributions are very different, learning a single model which performs well for all parties may require a very large number of parameters
- Another direction to deal with non-i.i.d. data is thus to lift the requirement that the learned model should be the same for all parties ("one size fits all")
- Instead, we can allow each party k to learn a (potentially simpler) personalized model θ_k but design the objective so as to enforce some kind of collaboration

Personalized models from a "meta" model

- [Hanzely et al., 2020] propose to regularize personalized models to their mean:

$$F(\theta_1, \dots, \theta_K; \mathcal{D}) = \frac{1}{K}\sum_{k=1}^{K} F_k(\theta_k; \mathcal{D}_k) + \frac{\lambda}{2K}\sum_{k=1}^{K}\left\|\theta_k - \frac{1}{K}\sum_{l=1}^{K}\theta_l\right\|^2$$

- Inspired by meta-learning, [Fallah et al., 2020] propose to learn a global model which easily adapts to each party:

$$F(\theta; \mathcal{D}) = \frac{1}{K}\sum_{k=1}^{K} F_k(\theta - \alpha\nabla F_k(\theta); \mathcal{D}_k)$$

- These formulations are actually related to each other (and to the FedAvg algorithm)
- Other formulations exist, see e.g., the bilevel approach of [Dinh et al., 2020]

- Inspired by multi-task learning, [Smith et al., 2017, Vanhaesebrouck et al., 2017] propose to regularize personalized models using (learned) relationships between tasks:

$$F(\theta_1, \dots, \theta_K, W; \mathcal{D}) = \frac{1}{K} \sum_{k=1}^{K} F_k(\theta_k; \mathcal{D}_k) + \sum_{k<l} W_{k,l} \|\theta_k - \theta_l\|^2$$

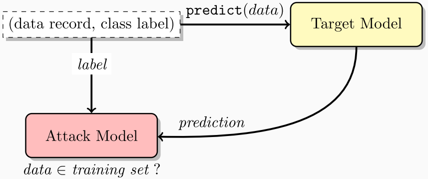
- This formulation naturally lends itself to alternating optimization schemes
- It is also well suited to the fully decentralized setting, since W can be seen as a graph of relationships over parties [Vanhaesebrouck et al., 2017]
- [Zantedeschi et al., 2020] propose to learn W to be a sparse graph of relationships and exchange messages only between pairs of related parties when updating the models

Some challenges in Federated Learning

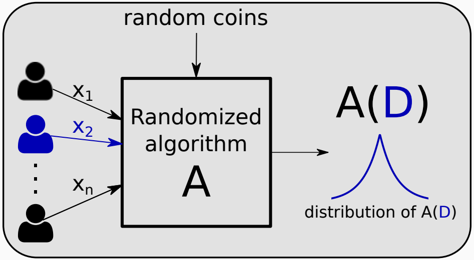
2. Preserving Privacy

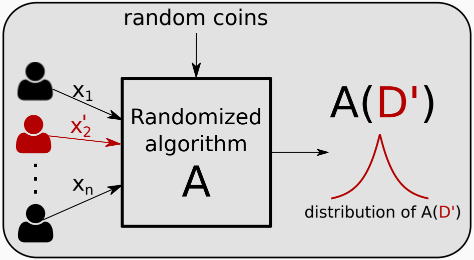
PRIVACY ISSUES IN (FEDERATED) ML

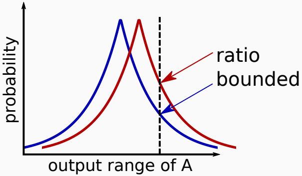
- ML models are susceptible to various attacks on data privacy
- Membership inference attacks try to infer the presence of a known individual in the training set, e.g., by exploiting the confidence in model predictions [Shokri et al., 2017]



- Reconstruction attacks try to infer some of the points used to train the model, e.g., by differencing attacks [Paige et al., 2020]
- Federated Learning offers an additional attack surface because the server and/or other clients observe model updates (not only the final model) [Nasr et al., 2019]







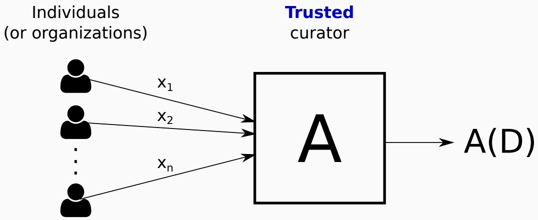
Definition ([Dwork et al., 2006], informal)

$\mathcal{A}$ is ε-differentially private (DP) if for all neighboring datasets $\mathcal{D} = \{x_1, x_2, \ldots, x_n\}$ and $\mathcal{D}' = \{x_1, x'_2, x_3, \ldots, x_n\}$ and all sets S:

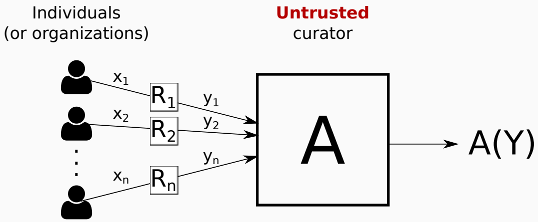
$$\Pr[\mathcal{A}(\mathcal{D}) \in S] \leq e^{\varepsilon} \Pr[\mathcal{A}(\mathcal{D}') \in S].$$

TWO SETTINGS: CENTRALIZED VS DECENTRALIZED

Centralized setting (also called global setting or trusted curator setting): $\mathcal{A}$ is differentially private wrt dataset $\mathcal{D}$



Decentralized/federated setting (also called local setting or untrusted curator setting): each $\mathcal{R}_k$ is DP wrt record x_k (or local dataset $\mathcal{D}_k$)



A KEY FUNCTIONALITY: DP AGGREGATION

- Most (server-orchestrated) FL algorithms follow the same high-level pattern:

  for $t = 1$ to T **do**
  At each party k: compute $\theta_k \leftarrow \textsc{LocalUpdate}(\theta, \theta_k)$, send θ_k to server
  At server: compute $\theta \leftarrow \frac{1}{K}\sum_k \theta_k$, send θ back to the parties

- Differentially private federated learning algorithms can thus be designed from a differentially private aggregation primitive and the composition property of DP

- In other words, given a private value x_k for each party k, we want to accurately estimate $x^{avg} = \frac{1}{K}\sum_k x_k$ under a DP constraint

APPROACHES TO DP AGGREGATION

- A standard approach in DP to add Gaussian noise calibrated to the sensitivity of the private value (and to the strength of the desired DP guarantee)
- In the decentralized setting, a baseline approach is to have each party k add the noise directly to x_k [Duchi et al., 2013]
- Unfortunately, the resulting average has poor accuracy unless K is very large: for a fixed privacy guarantee, the gap with the centralized setting is of $O(\sqrt{K})$
- Cryptographic primitives such as secure aggregation [Bonawitz et al., 2017] and secure shuffling [Balle et al., 2019] can be used to close this gap
- However their practical implementation poses important challenges when K is large

A simpler protocol for DP aggregation: GOPA [Sabater et al., 2020]

Algorithm GOPA protocol

Parameters: graph G, variances $\sigma_\Delta^2, \sigma_\eta^2 \in \mathbb{R}^+$

for all neighboring parties $\{k, l\}$ in G **do**
  k and l draw $y \sim \mathcal{N}(0, \sigma_\Delta^2)$
  set $\Delta_{k,l} \leftarrow y$, $\Delta_{l,k} \leftarrow -y$
for each party k **do**
  k draws $\eta_k \sim \mathcal{N}(0, \sigma_\eta^2)$
  k reveals $\hat{x}_k \leftarrow x_k + \sum_{l \sim k} \Delta_{k,l} + \eta_k$

1. All neighbors $\{k, l\}$ in G generate pairwise-canceling Gaussian noise
2. Each party k generate independent Gaussian noise
3. Party k reveals the sum of private value, pairwise and independent noise terms

- $\hat{x}^{avg} = \frac{1}{K} \sum_k \hat{x}_k$ can match the accuracy of the centralized setting (for sufficient σ_Δ^2)
- By choosing an appropriate graph G, each party communicates with only $O(\log K)$ other parties
- The approach is robust to collusions and drop outs (to some extent)

Wrapping up

- Going beyond empirical risk minimization formulations: tree-based methods [Li et al., 2020a], online learning [Dubey and Pentland, 2020], Bayesian learning...
- Vertical data partitioning, where parties have access to different features about the same examples [Patrini et al., 2016]
- Compressing updates to reduce communication [Koloskova et al., 2020a]
- Fairness in FL [Mohri et al., 2020, Li et al., 2020c, Laguel et al., 2020]
- Security in FL: how to mitigate poisoning attacks [Bagdasaryan et al., 2020] [Blanchard et al., 2017], how to make local computation verifiable [Sabater et al., 2020]

KEEPING UP WITH ADVANCES IN FEDERATED LEARNING

Survey paper: Advances and Open Problems in FL [Kairouz et al., 2019]

- A large collaborative effort (50+ authors!)
- Should be updated by the end of the year

Online seminar: Federated Learning One World (FLOW)

`https://sites.google.com/view/one-world-seminar-series-flow/`

- Co-organized with D. Alistarh, V. Smith and P. Richtárik, started in May 2020
- Weekly talks (usually on Wednesdays, 1pm UTC) covering all aspects of FL
- The videos and slides of all previous talks are available online

[Bagdasaryan et al., 2020] Bagdasaryan, E., Veit, A., Hua, Y., Estrin, D., and Shmatikov, V. (2020).
How To Backdoor Federated Learning.
In *AISTATS*.

[Balle et al., 2019] Balle, B., Bell, J., Gascón, A., and Nissim, K. (2019).
The Privacy Blanket of the Shuffle Model.
In *CRYPTO*.

[Blanchard et al., 2017] Blanchard, P., Mhamdi, E. M. E., Guerraoui, R., and Stainer, J. (2017).
Machine learning with adversaries: Byzantine tolerant gradient descent.
In *NIPS*.

[Bonawitz et al., 2017] Bonawitz, K., Ivanov, V., Kreuter, B., Marcedone, A., McMahan, H. B., Patel, S., Ramage, D., Segal, A., and Seth, K. (2017).
Practical Secure Aggregation for Privacy-Preserving Machine Learning.
In *CCS*.

[Dinh et al., 2020] Dinh, C. T., Tran, N. H., and Nguyen, T. D. (2020).
Personalized Federated Learning with Moreau Envelopes.
In *NeurIPS*.

[Dubey and Pentland, 2020] Dubey, A. and Pentland, A. S. (2020).
Differentially-Private Federated Linear Bandits.
In *NeurIPS*.

[Duchi et al., 2013] Duchi, J. C., Jordan, M. I., and Wainwright, M. J. (2013).
Local privacy and statistical minimax rates.
In *FOCS*.

[Dwork et al., 2006] Dwork, C., McSherry, F., Nissim, K., and Smith, A. (2006).
Calibrating noise to sensitivity in private data analysis.
In *Theory of Cryptography (TCC)*.

[Fallah et al., 2020] Fallah, A., Mokhtari, A., and Ozdaglar, A. (2020).
Personalized Federated Learning with Theoretical Guarantees: A Model-Agnostic Meta-Learning Approach.
In *NeurIPS*.

[Hanzely et al., 2020] Hanzely, F., Hanzely, S., Horváth, S., and Richtarik, P. (2020).
Lower Bounds and Optimal Algorithms for Personalized Federated Learning.
In *NeurIPS*.

[Kairouz et al., 2019] Kairouz, P., McMahan, H. B., Avent, B., Bellet, A., Bennis, M., Bhagoji, A. N., Bonawitz, K., Charles, Z., Cormode, G., Cummings, R., D'Oliveira, R. G. L., Rouayheb, S. E., Evans, D., Gardner, J., Garrett, Z., Gascón, A., Ghazi, B., Gibbons, P. B., Gruteser, M., Harchaoui, Z., He, C., He, L., Huo, Z., Hutchinson, B., Hsu, J., Jaggi, M., Javidi, T., Joshi, G., Khodak, M., Konečný, J., Korolova, A., Koushanfar, F., Koyejo, S., Lepoint, T., Liu, Y., Mittal, P., Mohri, M., Nock, R., Özgür, A., Pagh, R., Raykova, M., Qi, H., Ramage, D., Raskar, R., Song, D., Song, W., Stich, S. U., Sun, Z., Suresh, A. T., Tramèr, F., Vepakomma, P., Wang, J., Xiong, L., Xu, Z., Yang, Q., Yu, F. X., Yu, H., and Zhao, S. (2019).
Advances and Open Problems in Federated Learning.
Technical report, arXiv:1912.04977.

[Karimireddy et al., 2020] Karimireddy, S. P., Kale, S., Mohri, M., Reddi, S. J., Stich, S. U., and Suresh, A. T. (2020).
SCAFFOLD: Stochastic Controlled Averaging for On-Device Federated Learning.
In *ICML*.

[Koloskova et al., 2020a] Koloskova, A., Lin, T., Stich, S. U., and Jaggi, M. (2020a).
Decentralized Deep Learning with Arbitrary Communication Compression.
In *ICLR*.

[Koloskova et al., 2020b] Koloskova, A., Loizou, N., Boreiri, S., Jaggi, M., and Stich, S. U. (2020b).
A Unified Theory of Decentralized SGD with Changing Topology and Local Updates.
In *ICML*.

[Laguel et al., 2020] Laguel, Y., Pillutla, K., Malick, J., and Harchaoui, Z. (2020).
Device Heterogeneity in Federated Learning:A Superquantile Approach.
Technical report, arXiv:2002.11223.

[Li et al., 2020a] Li, Q., Wen, Z., and He, B. (2020a).
Practical Federated Gradient Boosting Decision Trees.
In *AAAI*.

[Li et al., 2020b] Li, T., Sahu, A. K., Zaheer, M., Sanjabi, M., Talwalkar, A., and Smith, V. (2020b).
Federated Optimization in Heterogeneous Networks.
In *MLSys*.

[Li et al., 2020c] Li, T., Sanjabi, M., Beirami, A., and Smith, V. (2020c).
Fair Resource Allocation in Federated Learning.
In *ICLR*.

[Lian et al., 2017] Lian, X., Zhang, C., Zhang, H., Hsieh, C.-J., Zhang, W., and Liu, J. (2017).
Can Decentralized Algorithms Outperform Centralized Algorithms? A Case Study for Decentralized Parallel Stochastic Gradient Descent.
In *NIPS*.

[McMahan et al., 2017] McMahan, H. B., Moore, E., Ramage, D., Hampson, S., and Agüera y Arcas, B. (2017).
Communication-efficient learning of deep networks from decentralized data.
In *AISTATS*.

[Mohri et al., 2020] Mohri, M., Sivek, G., and Suresh, A. T. (2020).
Agnostic Federated Learning.
In *ICML*.

[Nasr et al., 2019] Nasr, M., Shokri, R., and Houmansadr, A. (2019).
Comprehensive Privacy Analysis of Deep Learning: Passive and Active White-box Inference Attacks against Centralized and Federated Learning.
In *IEEE Symposium on Security and Privacy*.

[Paige et al., 2020] Paige, B., Bell, J., Bellet, A., Gascón, A., and Ezer, D. (2020).
Reconstructing Genotypes in Private Genomic Databases from Genetic Risk Scores.
In *International Conference on Research in Computational Molecular Biology RECOMB*.

[Patrini et al., 2016] Patrini, G., Nock, R., Hardy, S., and Caetano, T. S. (2016).
Fast Learning from Distributed Datasets without Entity Matching.
In *IJCAI*.

[Sabater et al., 2020] Sabater, C., Bellet, A., and Ramon, J. (2020).
Distributed Differentially Private Averaging with Improved Utility and Robustness to Malicious Parties.
Technical report, arXiv:2006.07218.

[Shokri et al., 2017] Shokri, R., Stronati, M., Song, C., and Shmatikov, V. (2017).
Membership Inference Attacks Against Machine Learning Models.
In *IEEE Symposium on Security and Privacy (S&P)*.

[Smith et al., 2017] Smith, V., Chiang, C.-K., Sanjabi, M., and Talwalkar, A. S. (2017).
Federated Multi-Task Learning.
In *NIPS*.

[Stich, 2019] Stich, S. U. (2019).
Local SGD Converges Fast and Communicates Little.
In *ICLR*.

[Vanhaesebrouck et al., 2017] Vanhaesebrouck, P., Bellet, A., and Tommasi, M. (2017).
Decentralized collaborative learning of personalized models over networks.
In *AISTATS*.

[Woodworth et al., 2020] Woodworth, B., Patel, K. K., Stich, S. U., Dai, Z., Bullins, B., McMahan, H. B., Shamir, O., and Srebro, N. (2020).
Is Local SGD Better than Minibatch SGD?
In *ICML*.

[Zantedeschi et al., 2020] Zantedeschi, V., Bellet, A., and Tommasi, M. (2020).
Fully Decentralized Joint Learning of Personalized Models and Collaboration Graphs.
In *AISTATS*.