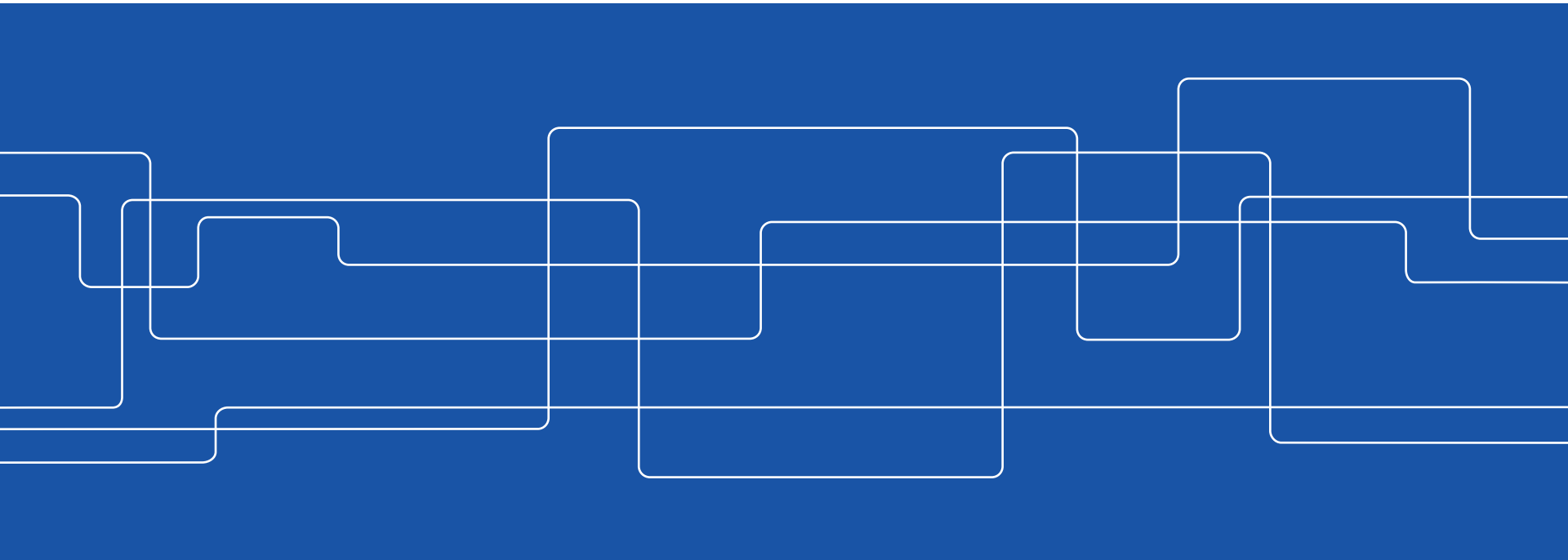
Lecture MX4: Distributed Machine Learning (II)

Ming Xiao

2020.10.1



Recap of convex optimization and distributed machine learning

In last lecture, we study:

- Canonical opt. problem in DML: $\min_{w} \frac{1}{N} \sum_{i \in [N]} f_i(w)$
- Efficient distributed solvers (such as D-GD, ADMM)
- Network structure and parallel approaches for distributed machine learning (DML)

Outline

- Background
- Main Challenges
- Federated Learning (FL) Principles
- FL Structure
- FedAvg Algorithm
- Communication Efficiency
- System Heterogneity
- Privacy
- Application Examples
- Recent Development

Background

- Data breaches may seriously threaten user data privacy. In 2019, millions of Facebook user records are exposed on Amazon cloud[1]
- The security requirement on training data is increasingly high due to various policies and regulations such as GDPR in Europe.
- The distributed property of DML makes data vulnerable.
- We might want a network of nodes (e.g., sensors, users, cameras, medical devices) to provide updates for our model but not transmit their raw data over networks.
- The term Federated Learning was coined by Google AI in a paper[2] (published in 2017)
- Since then, it has been an area of active research as evidenced by papers (500+) published on arXiv[3]

1. Hundreds of millions of facebook user records were exposed on amazon cloud server, 2019. URL https://www.cbsnews.com/news/millions-facebook-user-records-exposed-amazon-cloud-server/.
2. McMahan B, Moore E, Ramage D, Hampson S, Arcas BAy. 2017 Communication-efficient learning of deep networks from decentralized data. See http://proceedings.mlr.press/v54/mcmahan17a.html.
3. https://arxiv.org/search/?searchtype=all&query=%22federated+learning%22&abstracts=show&size=200&order=-announced_date_first

Main Motivations of FL

1. Expensive communication costs (different from data center)
   - communication may be more expensive than computing (e.g., mobile nodes)
   - Communication links may be unstable/unreliable
2. Systems Heterogeneity
   - Heterogeneous hardware
   - Heterogenous links (straggler devices in the network)
3. Data Heterogeneity

   Non-I.I.D. Data: Data generated from personalized or device-specific users. May not follow popular distribution.
4. Strong privacy requirement
   - Raw data may be very sensitive. Better share model updates, .e.g., gradient information.
   - Secure multiparty computation or differential privacy may reduce model performance or system efficiency.

Main Motivations of FL (cont')

5. Synchronization problems

- May be more difficult due to heterogeneous connection (than distributed ML with stable connection).

6. Users have control over their device and data

Only limited information will be shared and thus limits the models. Server cannot control worker nodes, contrary to classic server-slave models.

7. The amount of data may be quite imbalanced:

some nodes may have much more while others much smaller.

8. Massively distributed: May be very large amount of devices participation (large-scale machine learning)

[1] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. Aguera y Arcas, "Communication-Efficient Learning of Deep Networks from Decentralized Data," in AISTATS, 2017.
[2] J. Konevcny, H. McMaha and D. Ramage, "Federated Optimization: Distributed Optimiztion Beyond DataCenter, " *arXiv preprint arXiv:1511.03575*, 2015.

Federated Learning Principles

What is FL?

FL is a *privacy-preserving* and *communication-efficient* model training in *heterogeneous*, *distributed* networks often in mobile networks (Collaborative Machine Learning without Centralized Training Data or Computing).

Reference [1] first coins the term Federated Learning.

Problem formulation

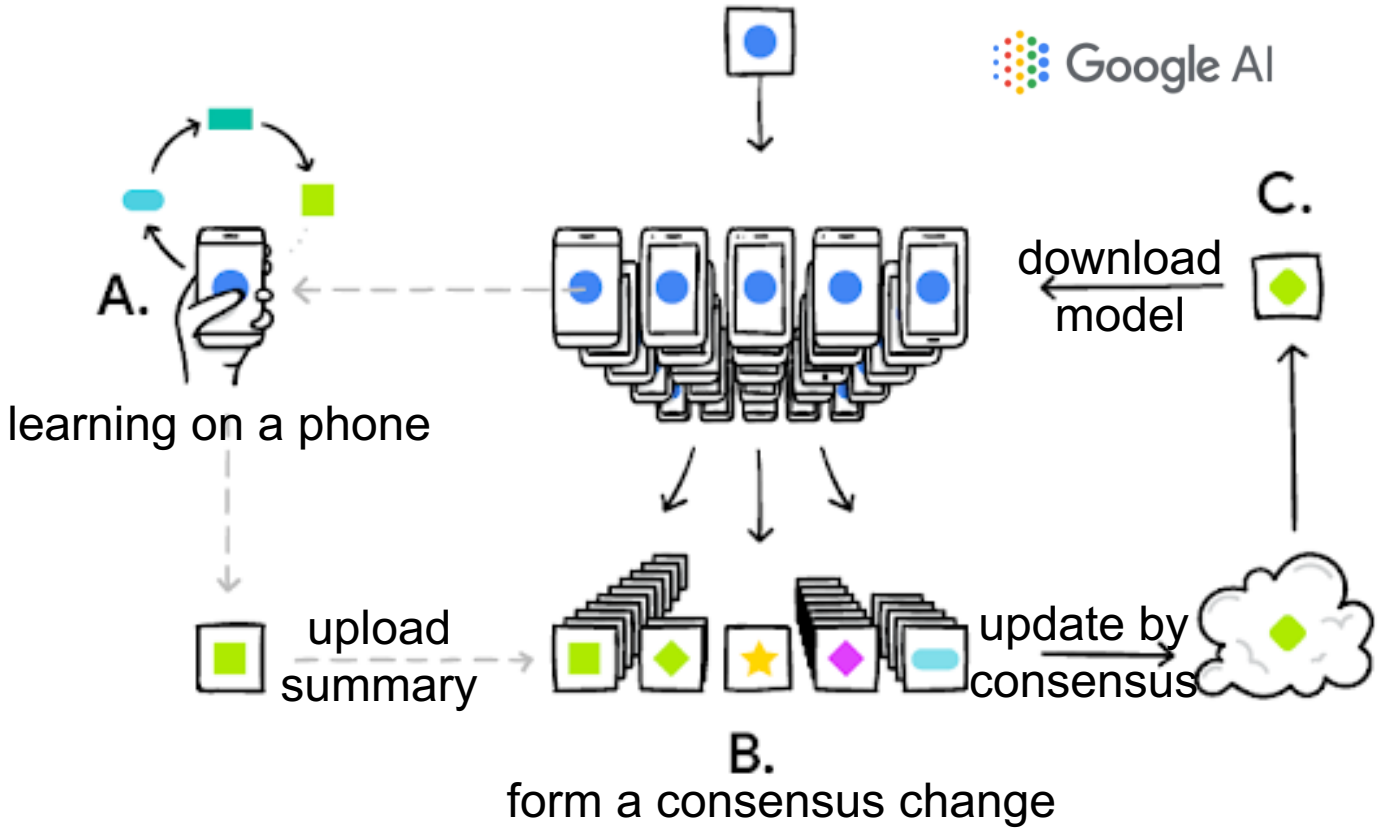
The canonical federated learning problem involves learning a single, global statistical model from data stored on tens to potentially millions of remote devices. In particular, the objective is typically to minimize the following objective function:

$$\min_w F(w), \text{ where } F(w) := \sum_{k=1}^{m} p_k F_k(w)$$

Here m is the total number of devices, F_k is the local objective function for the kth device, and p_k specifies the relative impact of each device with $p_k \geq 0$ and $\sum_{k=1}^{m} p_k = 1$.

[1] H. B. McMahan, E. Moore, D. Ramage, S. Hampson and B. Aguera y Arcas, "Communication-efficient learning of deep networks from decentralized data," In *Proceedings* of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS), 2017.

Federated Learning Principles (example in mobile)



Mobile phone personalizes the model locally, based on your usage (A) Many user updates are aggregated (B) to form a consensus change (C) to the shared model, after which the procedure is repeated.

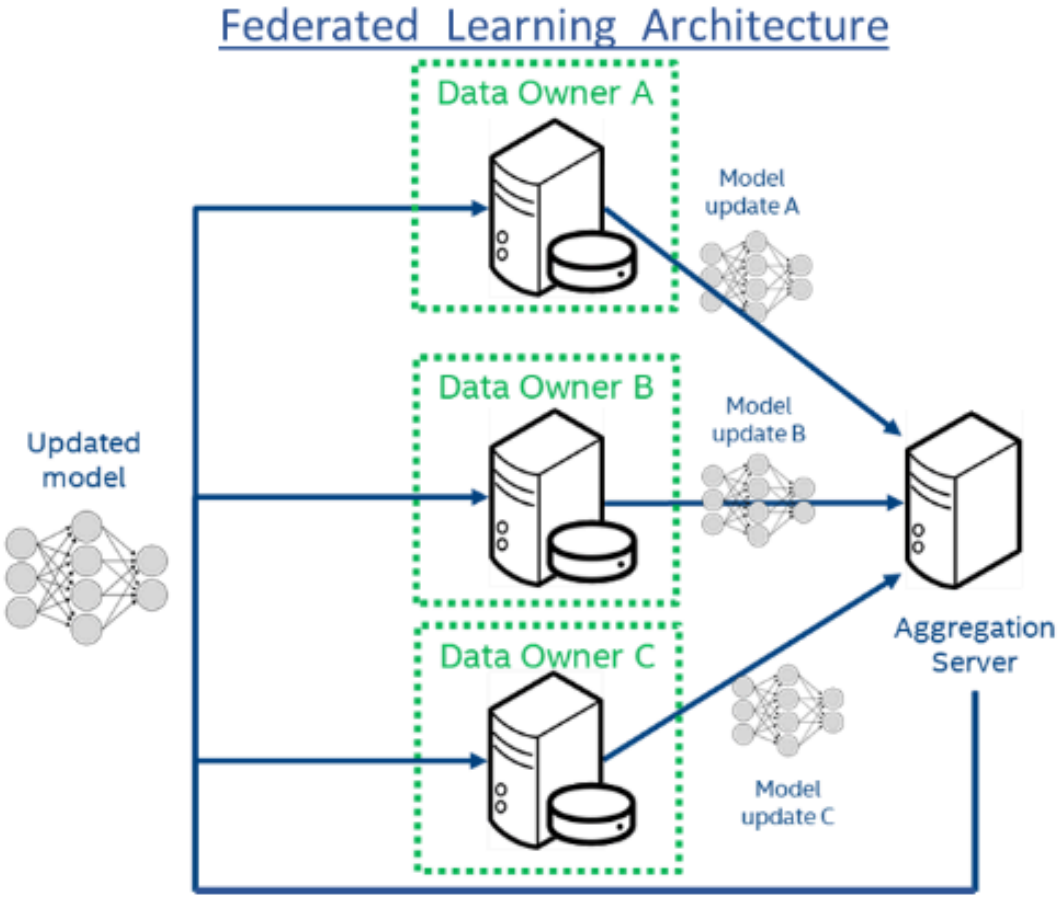
[3] https://ai.googleblog.com/2017/04/federated-learning-collaborative.html

FL Principle:
FL vs. Edge Computing vs. Centralized Learning

	Centralized Learning	Edge Computing	Federated Learning
Privacy	X	X	✓
Bandwidth/ Comm. efficient	X	✓	✓
Latency	X	✓	✓
Cost/Feasibility	X	X	✓

- User data do not leave the local device, only updates of the model are transmitted with security protection
- The updates are smaller than the original user data. Consequently the overall workload needed is lower in FL than in cloud based architectures or in edge computing. This makes FL cheaper and more convenient
- The model updates are computed close to the user device, allowing for real time inferences with no latency problems.

FL Structure: Master-Slave Structure



The encrypted model is sent to the individual data owners, which decrypt (?) and then train using the local data

Only the model updates are shared with the central model aggregator server. This provides protection to both the model and the data.

The raw data do not leave data owners. This does not only add privacy but also prevent a large amount of data transmitted on the network.

One Classical algorithm- Federated Averaging (*FedAvg*) Algorithm

- K: total # clients
- k: index of clients
- n_k: # data samples available during training for client k
- w_t: model weight vector on client k, at the **federated (comm.) round** t
- $l(w; b)$: loss function for weight w and batch b
- E: # local epochs

Algorithm 1 `FederatedAveraging`. The K clients are indexed by k; B is the local minibatch size, E is the number of local epochs, and η is the learning rate.

Server executes:

initialize w_0

for each round $t = 1, 2, \dots$ **do**

$m \leftarrow \max(C \cdot K, 1)$

$S_t \leftarrow$ (random set of m clients)

for each client $k \in S_t$ **in parallel do**

$w_{t+1}^k \leftarrow \text{ClientUpdate}(k, w_t)$

$w_{t+1} \leftarrow \sum_{k=1}^{K} \frac{n_k}{n} w_{t+1}^k$

ClientUpdate(k, w): *// Run on client k*

$\mathcal{B} \leftarrow$ (split $\mathcal{P}_k$ into batches of size B)

for each local epoch i from 1 to E **do**

for batch $b \in \mathcal{B}$ **do**

$w \leftarrow w - \eta \nabla \ell(w; b)$

return w to server

[1] H. Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Proceedings* of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS), 2017.

Convergence Rate for *FedAvg* algorithm

Convergence rate on Non-I.I.D data (w.r.t. Local Epochs): $O\left(\frac{1}{T}\right)$, T is **total** iterations [4]

Make following assumptions on the local loss function $F_1, \ldots F_N$:

Assumption 1: *L-Smooth* $F_k(v) \leq F_k(w) + (v-w)^T \nabla F_k(w) + \frac{L}{2}\|v-w\|_2^2$

Assumption 2: μ-*strongly* convex $F_k(v) \geq F_k(w) + (v-w)^T \nabla F_k(w) + \frac{\mu}{2}\|v-w\|_2^2$

Assumption 3 & 4: *Bounded variance and gradient* $E\left\|\nabla F_k\left(w_t^k, \xi_t^k\right) - \nabla F_k\left(w_t^k\right)\right\|^2 \leq \sigma_k^2$ and $E\left\|\nabla F_k\left(w_t^k, \xi_t^k\right)\right\|^2 \leq G^2$

Theorem 1. *Let Assumptions 1 to 4 hold and* L, μ, σ_k, G *be defined therein. Choose* $\kappa = \frac{L}{\mu}$, $\gamma = \max\{8\kappa, E\}$ *and the learning rate* $\eta_t = \frac{2}{\mu(\gamma+t)}$. *Then* `FedAvg` *with full device participation satisfies*

$$\mathbb{E}\left[F(\mathbf{w}_T)\right] - F^* \leq \frac{2\kappa}{\gamma + T}\left(\frac{B}{\mu} + 2L\|\mathbf{w}_0 - \mathbf{w}^*\|^2\right),$$

where

$$B = \sum_{k=1}^{N} p_k^2 \sigma_k^2 + 6L\Gamma + 8(E-1)^2 G^2.$$

[4] L. Xiang, et al. "On the convergence of fedavg on non-i.i.d data." arXiv preprint arXiv:1907.02189 (2019).

Proof Outline

Main steps:

- Step 1: define $\overline{g_t} = \sum_{k=1}^{N} p_k \nabla F_k(w_t^k)$, $g_t = \sum_{k=1}^{N} p_k \nabla F_k(w_t^k, \xi_t^k)$, first bound the variance: $\mathrm{E}\|g_t - \overline{g_t}\|^2 \leq \sum_{k=1}^{N} p_k^2 \sigma_k^2$
- Step 2: bound the divergence of $\{w_t^k\}$, we get $\mathrm{E}\left[\sum_{k=1}^{N} p_k \left\|\overline{w_t} - w_k^t\right\|^2\right] \leq 4\eta_t^2 (E-1)^2 G^2$
- Step 3: define $\overline{w_t} = \sum_{k=1}^{N} p_k w_t^k$, $\Delta_t = \mathrm{E}\,\|w^* - \overline{w_t}\|^2$, it is easy to verify that $\Delta_{t+1} \leq (1 - \eta_t \mu)\Delta_t + \eta_t^2 B$
- Step 4: with L-smoothness of $F(\cdot)$, we derive $\mathrm{E}[F(\overline{w_t})] - F^* \leq \frac{L}{2}\Delta_t \leq \frac{L}{2}\frac{v}{\gamma+t}$, where $v = max\left\{\frac{\beta^2 B}{\beta\mu - 1}, (\gamma+1)\Delta_1\right\} \leq \frac{\beta^2 B}{\beta\mu - 1} + (\gamma+1)\Delta_1 \leq \frac{4B}{\mu^2} + (\gamma+1)\Delta_1$. This complete the proof

Convergence Rate for *FedAvg* Algorithm

Communication round = *T/E*

$$\frac{T_\epsilon}{E} \propto \left(1+\frac{1}{K}\right) EG^2 + \frac{\sum_{k=1}^{N} p_k^2 \sigma_k^2 + L\Gamma + \kappa G^2}{E} + G^2$$

Communication cost reduces with increasing *E* at first and then incease with overlage *E*. There is optimal *E*.

[4] Li, Xiang, et al. "On the convergence of fedavg on non-i.i.d data." arXiv preprint arXiv:1907.02189 (2019).

Performance

FedAvg

share updated parameters

FedSGD

share local gradients

baseline algorithm for FedAvg

special case of FedAvg:

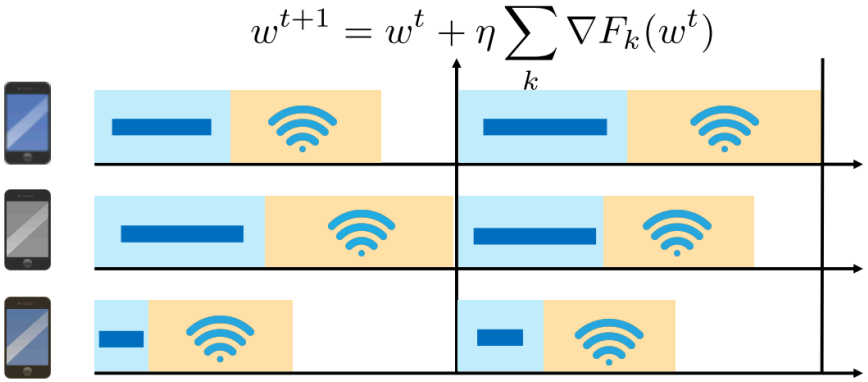
- Single local batch ($B = \infty$)
- Single local epoch ($E = 1$)

Communication-efficiency

Several general directions:

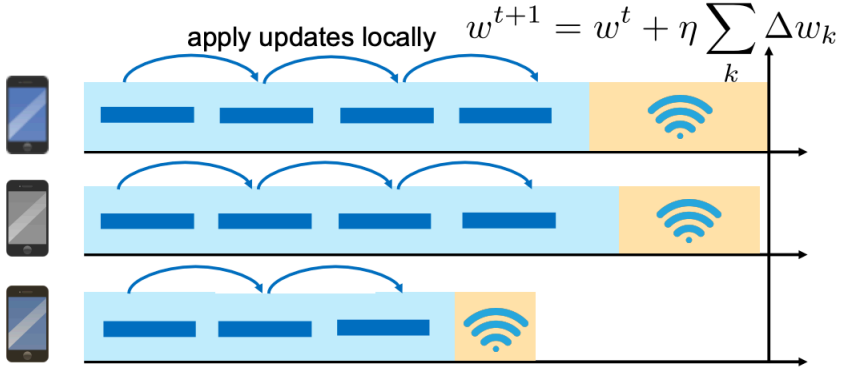
Local-updating (computing) reduce commu.

1. More local updating
   - Reduce the total ***number of communication rounds***



Left: FedSGD

Each device k computes gradients from a mini-batch of data points to approximate $\nabla F_k(w)$, and the aggregated mini-batch updates are applied on the server

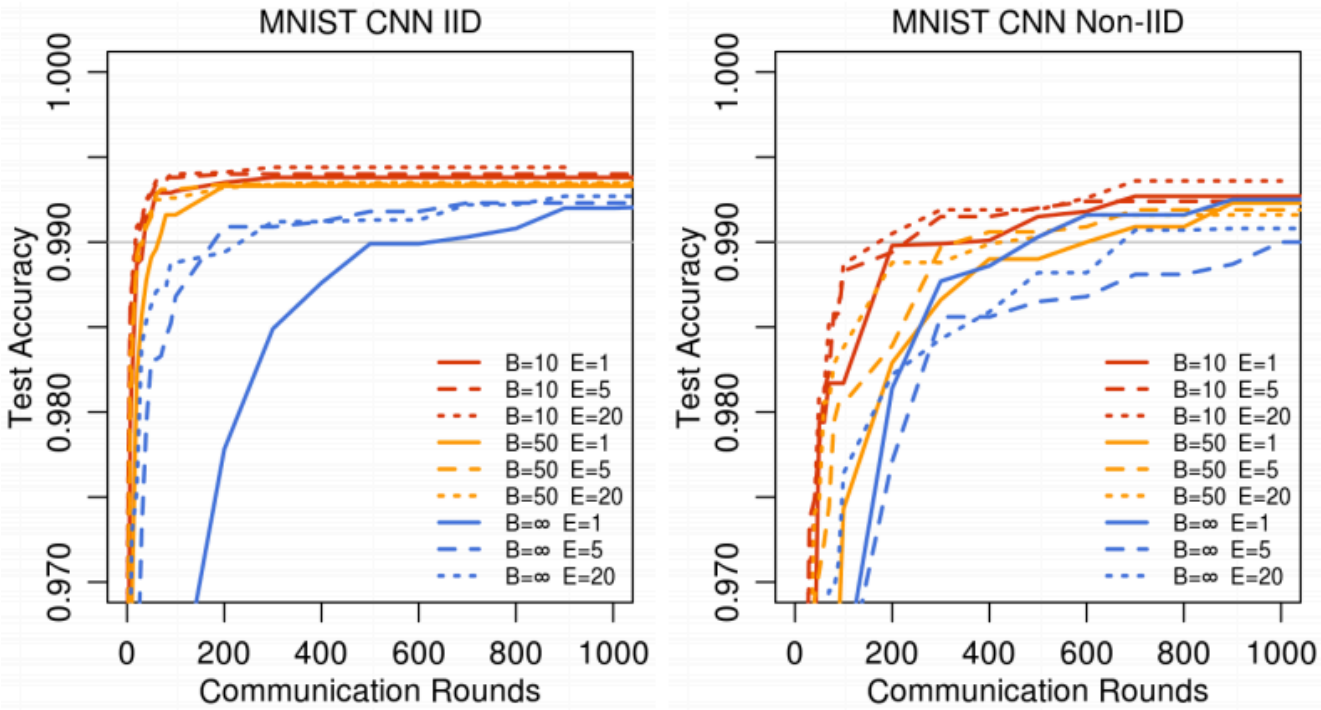


Right: Local updating schemes. FedAvg

Each device immediately applies local updates, e.g., gradients, after they are computed and a server performs a global aggregation after a variable number of local updates.

Numerical Results from Experiment

Convergence performance

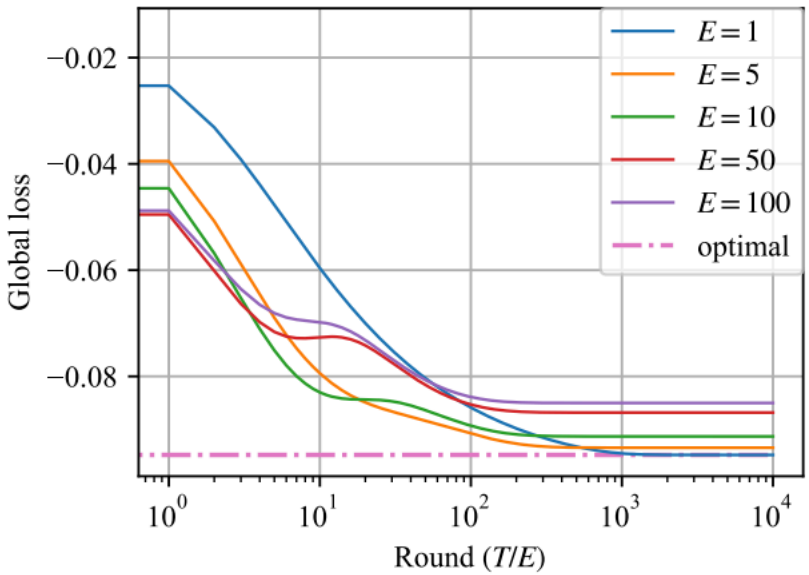


Test accuracy vs. # communication rounds for the MNIST CNN (IID and Non-IID)

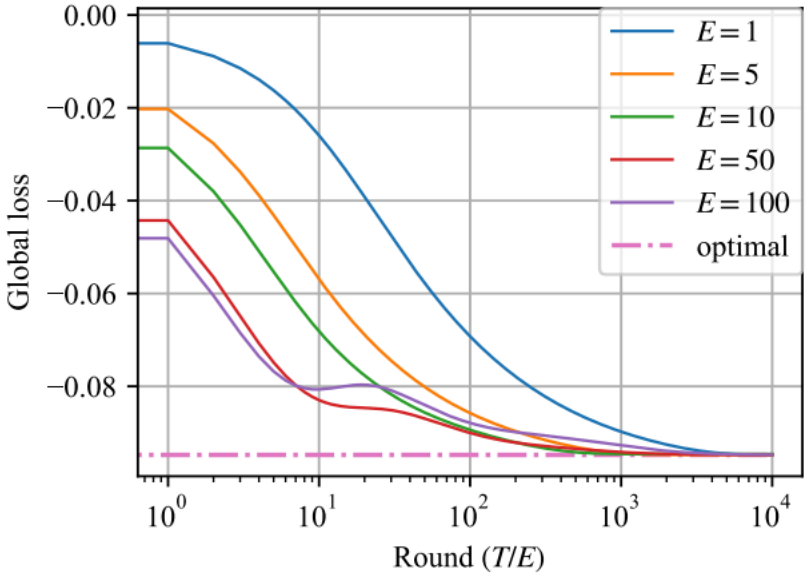
[1] H. B. McMahan, et. al. “Communication-efficient learning of deep networks from decentralized data,” In *Proceedings* of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS), 2017.

Numerical Results from Experiment

Learning rates (step size)



(a) Fixed learning rates



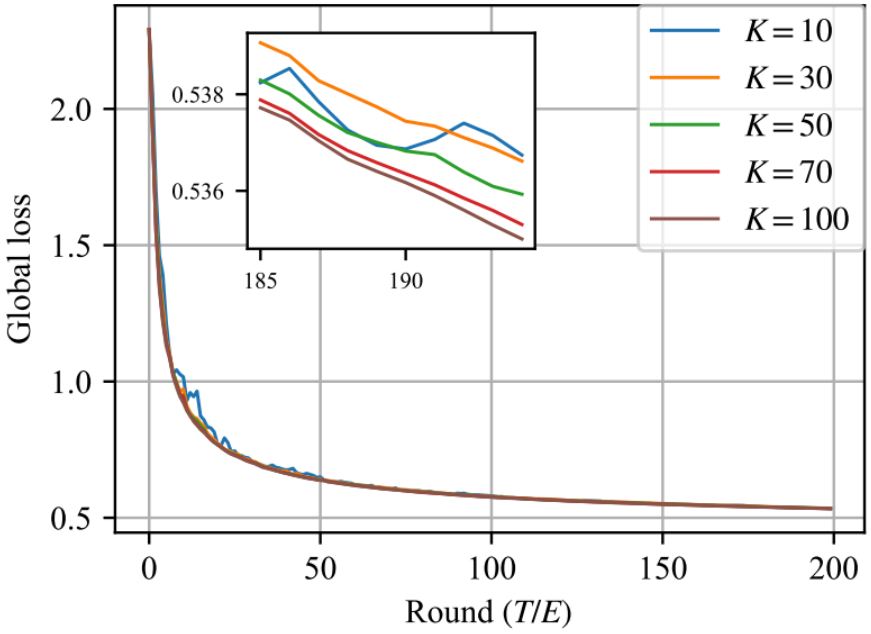
(b) Decayed learning rates

The left figure shows that the global objective value that FedAvg converges to is suboptimal unless $E = 1$. Once we decay the learning rate, FedAvg can converge to the optimal even if $E > 1$.

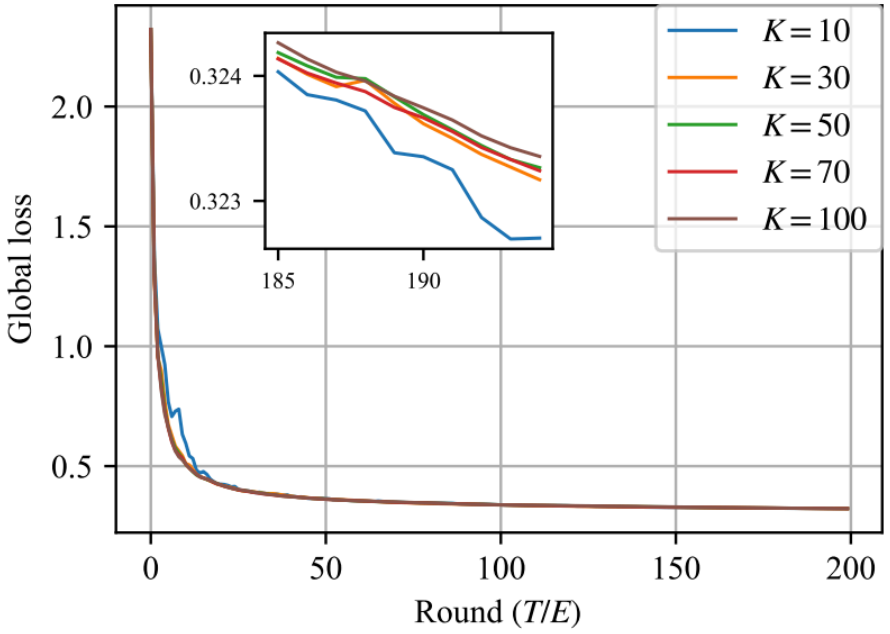
[4] Li, Xiang, et al. "On the convergence of fedavg on non-iid data." arXiv preprint arXiv:1907.02189 (2019).

Numerical Results from Experiment

Balance vs unbalanced data



(a) Balanced MNIST



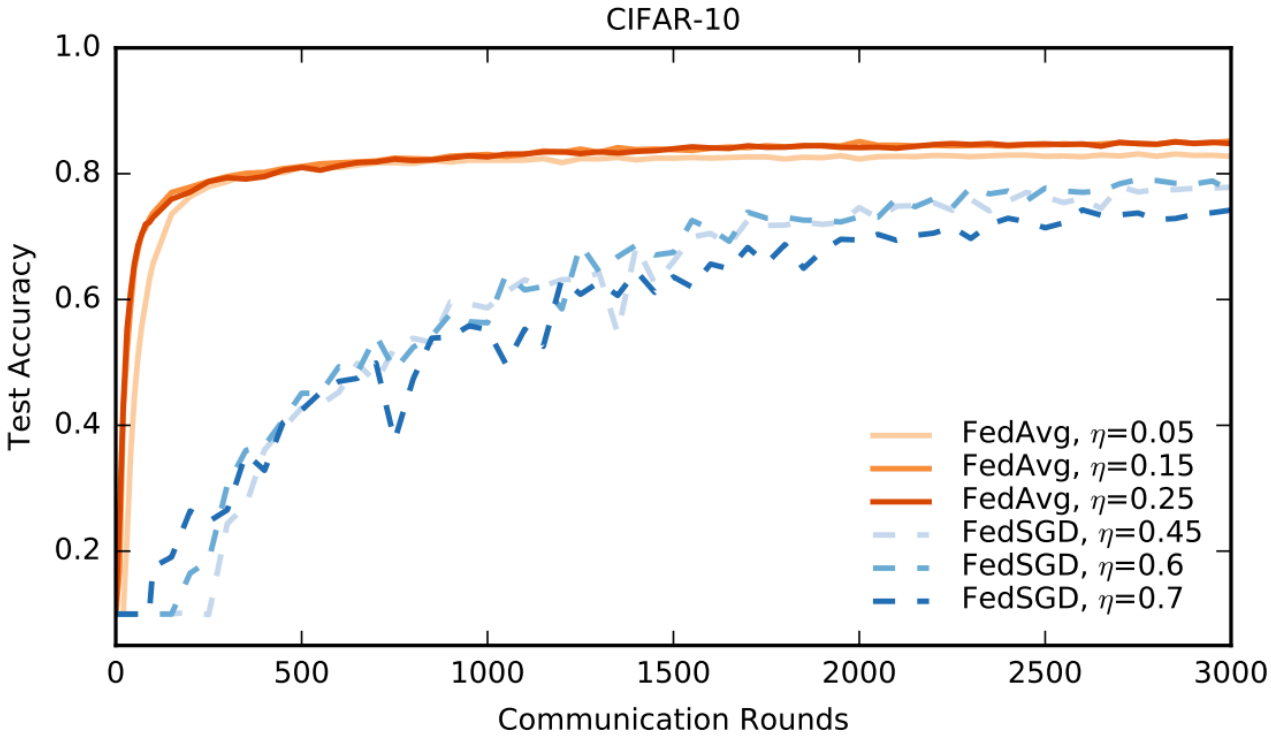
(b) Unbalanced MNIST

The impact of K on MNIST datasets.

[4] Li, Xiang, et al. "On the convergence of fedavg on non-iid data." arXiv preprint arXiv:1907.02189 (2019).

Numerical Results from Experiment

Communication analysis [1]



Test accuracy vs. # communication for the CI-FAR10 experiments. *FedSGD* uses a learning-rate decay of 0.9934 per round; *FedAvg* uses $B = 50$, learning-rate decay of 0.99 per round, and $E = 5$.

Further improvement on communication-efficiency

1. Compression

- Model compression schemes (such as *sparsification*, *subsampling* and *quantization*) can significantly reduce the ***size of messages*** communicated at each round

Sparsification

- The key idea of a sparsification is to sparsify a high-dimensional vector by randomly selecting some coordinates and setting the information of these coordinates to zero
- E.g., Sparsification gradient: sends a sparse vector with a subset of important values from the full gradient

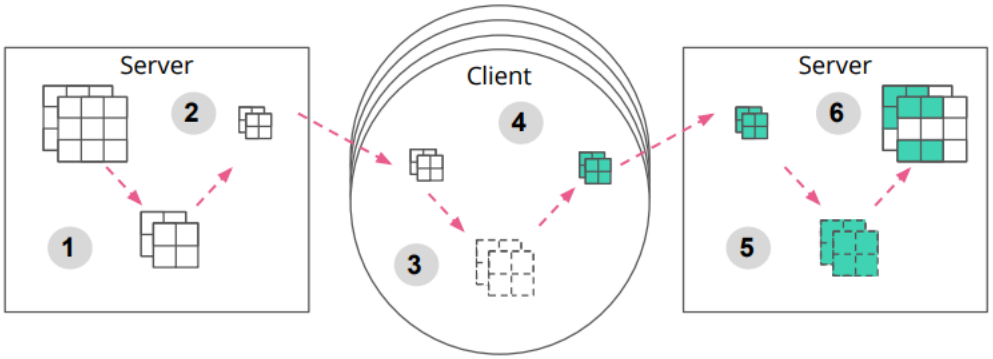
Subsampling

- The key idea is to analyze a fraction of samples (in terms of model parameters)

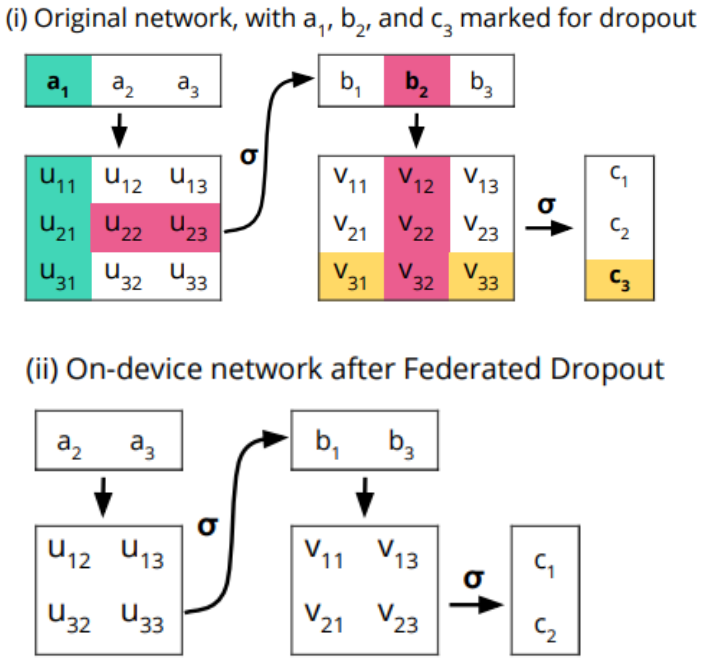
Quantization

- The key idea of quantizer aims to compress a high-dimensional vector by limiting the number of bits that represent floating point number (with low-precision vector)
- E.g., signed gradient

Example: Lossy compression via federated dropout



1) Construct a sub-model via *Federated Dropout*
2) Lossily compress the resulting object
3) Decompress and train it using local data
4) Compress the final update
5) Decompress the update
6) Aggregate it into the global model



Federated Dropout applied to two fully-connected neural layers.

[5] S. Caldas, J. Konecny, H. B. McMahan, and A. Talwalkar, Expanding the reach of federated learning by reducing client resource requirements. 2018. [Online]. Available: arXiv:1812.07210, 2018.

Further improvement on communication-efficiency (cont'd)

2. Decentralized Training

- Devices only communicate with their neighbours, e.g., the right figure

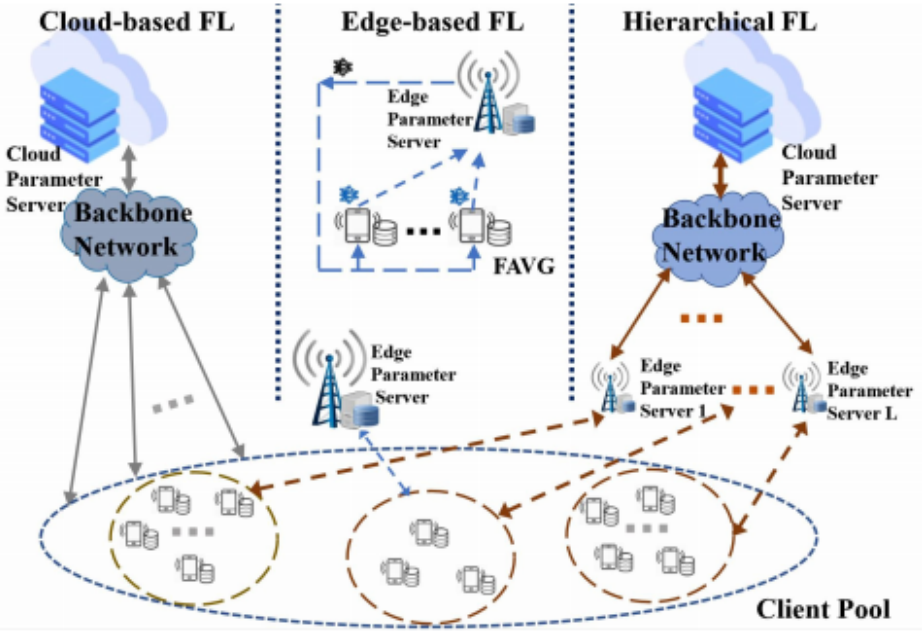
Centralized vs. decentralized topologies

[6] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," https://arxiv.org/abs/1908.07873.

Further improvement on communication-efficiency (cont'd)

3. Hierarchical communication patterns

- e.g., Client-edge-cloud hierarchical FL system
  - Train faster and better communication-computation trade-offs



By first leveraging edge servers to aggregate the updates from edge devices and then relying on a cloud server to aggregate updates from edge servers.

[7] L. Liu, J. Zhang, S. Song, and K. B. Letaief. Edge-assisted hierarchical federated learning with non-iid data. arXiv preprint arXiv:1905.06641, 2019.

System Heterogeneity

Important topics:

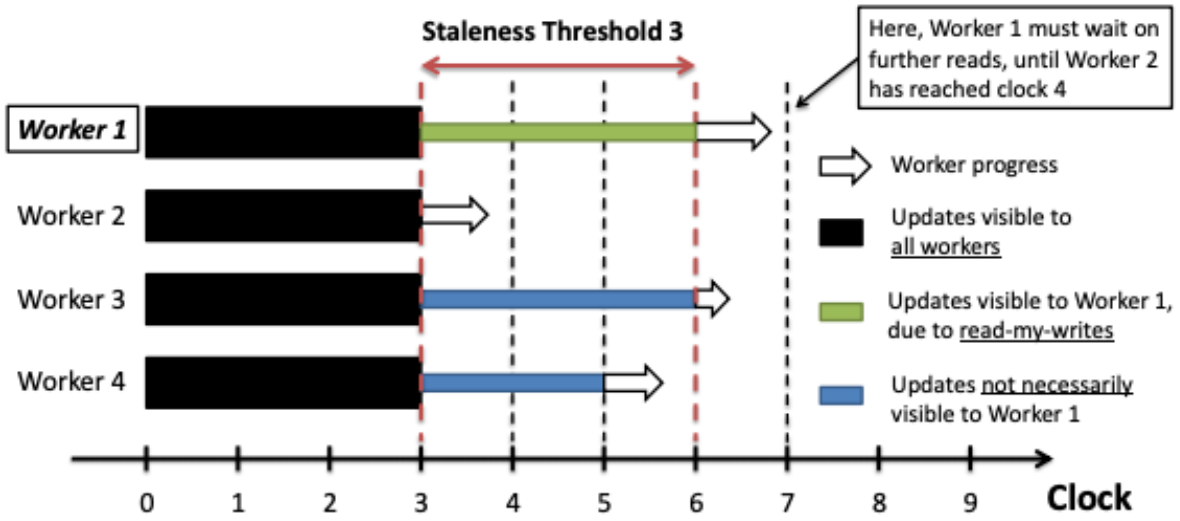
1. **Asynchronous Communication**
2. **Fault Tolerance**

Asynchronous Communication

Synchronous Schemes: Simple but more susceptible to stragglers in the face of device variation

Asynchronous Schemes: Mitigate stragglers in heterogenous environments

Example: SSP: Stale Synchronous Parallel



Bounded Staleness under the SSP Model

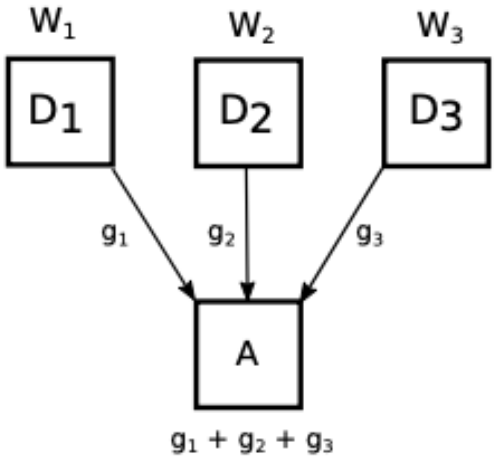
[8] Q. Ho, J. Cipar, H. Cui, S. Lee, J. K. Kim, P. B. Gibbons, G. A. Gibson, G. Ganger, and E. P. Xing. More effective distributed ML via a stale synchronous parallel parameter server. In Advances in Neural Information Processing Systems, 2013.

Fault Tolerance

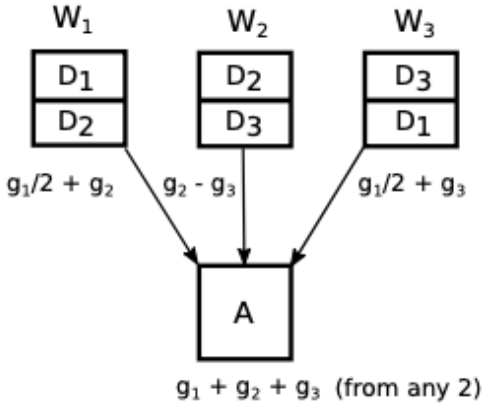
Some participating devices to drop out at some point before the completion of the given training iteration

One practical strategy to simply ignore such device failure, which may introduce bias into the device sampling scheme

Coded computation by introducing algorithmic redundancy



(a) Naive synchronous gradient descent



(b) Gradient coding: The vector $g_1 + g_2 + g_3$ is in the span of *any two* out of the vectors $g_1/2 + g2$, $g_2 - g_3$ and $g_1/2 + g_3$.

[9] R. Tandon, Q. Lei, A. G. Dimakis, and N. Karampatziakis. Gradient coding: Avoiding stragglers in distributed learning. In International Conference on Machine Learning, 2017.

Statistical Heterogeneity

Several general research topics in FL:

It is challenging to learn with non-I.I.D data.

Convergence Guarantees for non-I.I.D data

Convergence Guarantees for Non-I.I.D Data

- Statistical heterogeneity also presents novel challenges in terms of analysing the **convergence** behaviour in FL settings.
- Methods such as *FedAvg* can be divergent in practice
- Also some heuristic approaches that aim to tackle statistical heterogeneity, either by sharing local device data or some server-side proxy data
- However, these methods may be unrealistic: in addition to imposing burdens on network bandwidth, sending local data to the server violates the key privacy assumption of FL
- Method *FedProx* was proposed to help improve convergence, both theoretically and practically

FedProx Algorithm

Different from *FedAvg:*

Introduce *proximal* term: $\min_w h_k(w; w^t) = F_k(w) + \frac{\mu}{2}\|w - w^t\|^2$

Algorithm 2 `FedProx` (Proposed Framework)

Input: $K, T, \mu, \gamma, w^0, N, p_k, k = 1, \cdots, N$
for $t = 0, \cdots, T-1$ **do**
  Server selects a subset S_t of K devices at random (each device k is chosen with probability p_k)
  Server sends w^t to all chosen devices
  Each chosen device $k \in S_t$ finds a w_k^{t+1} which is a γ_k^t-inexact minimizer of: $w_k^{t+1} \approx \arg\min_w h_k(w; w^t) = F_k(w) + \frac{\mu}{2}\|w - w^t\|^2$
  Each device $k \in S_t$ sends w_k^{t+1} back to the server
  Server aggregates the w's as $w^{t+1} = \frac{1}{K}\sum_{k \in S_t} w_k^{t+1}$
end for

[10] T. Li, A. Kumar Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," 2018, arXiv:1812.06127. [Online]. Available: http://arxiv.org/abs/1812.06127.

Privacy

- Privacy concerns often motivate the need to keep raw data on each device local in FL settings
- However, sharing other information such as model updates as part of the training process can also leak sensitive user information since model parameters are functions (e.g., linear or poly.) of raw data.
- For instance, one can extract sensitive text patterns from a recurrent neural network trained on user language data.

Main strategies for improving privacy in FL:

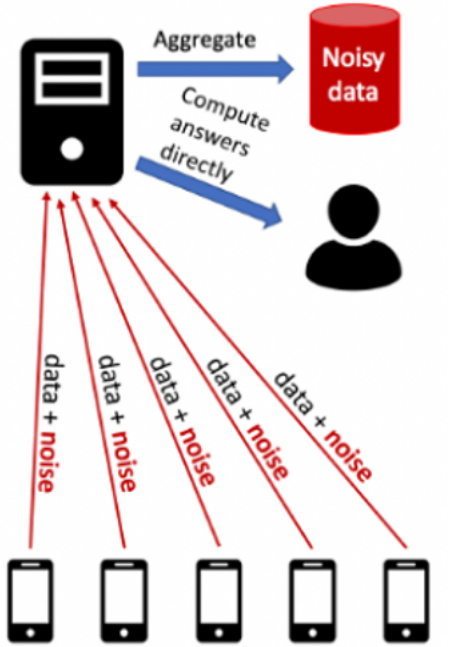
- **Differential privacy** (DP) to communicate noisy data (weights)
- **Homomorphic encryption** (HE) to operate on encrypted data
- Secure multiparty computation (**SMC**)

Differential Privacy (DP)

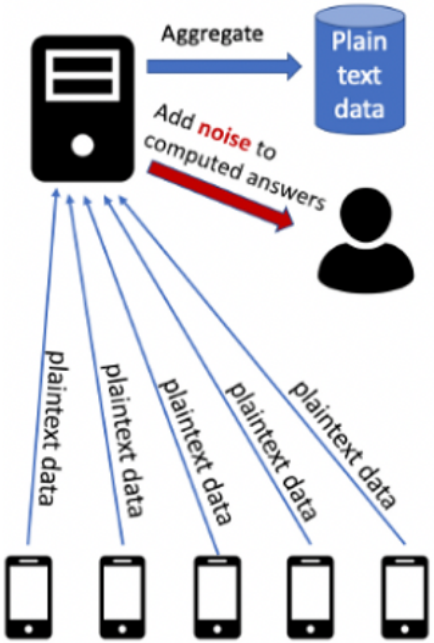
- Differential privacy is the statistical approach of trying to learn **as mush as possible about a group** while learning **as little as possible about any individual in it**
- Maximize the accuracy and minimize the opportunity of recognizing the individual by adding artificial noise.
- There exists a **trade-off** between differential privacy and model accuracy, as adding more noise results in greater privacy, buy may compromise accuracy significantly

DP: Global privacy and local privacy

- Global privacy: the model updates are private to **all untrusted third parties** other than the central parameter-server
- Local privacy: the model updates are also **private to the server**

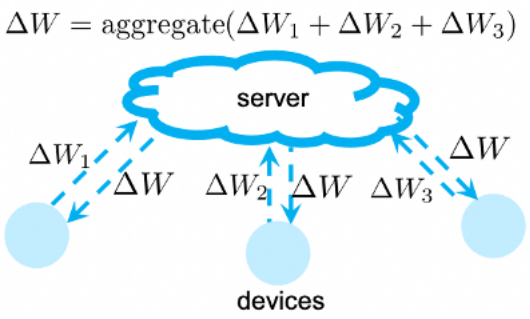


Local differential privacy

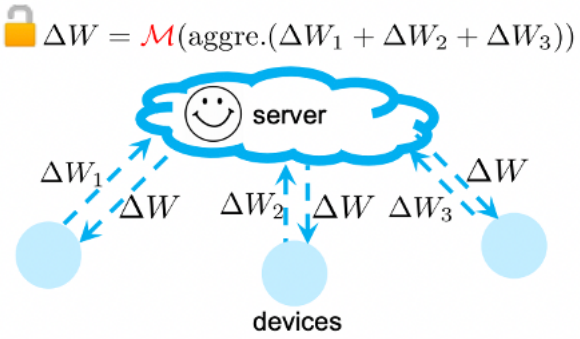


Global differential privacy

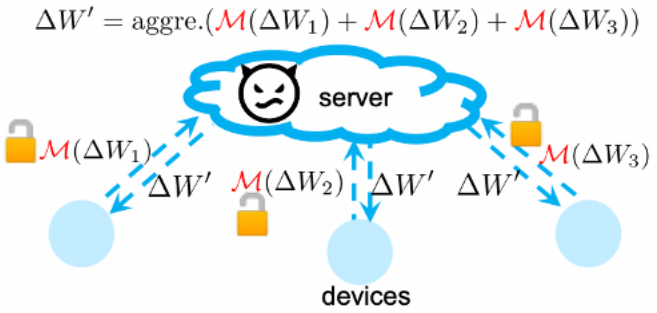
Example: Different Privacy-Preserving Mechanisms [6]



(a) FL without additional privacy protection mechanisms



(b) Global privacy, where a trusted server is assumed



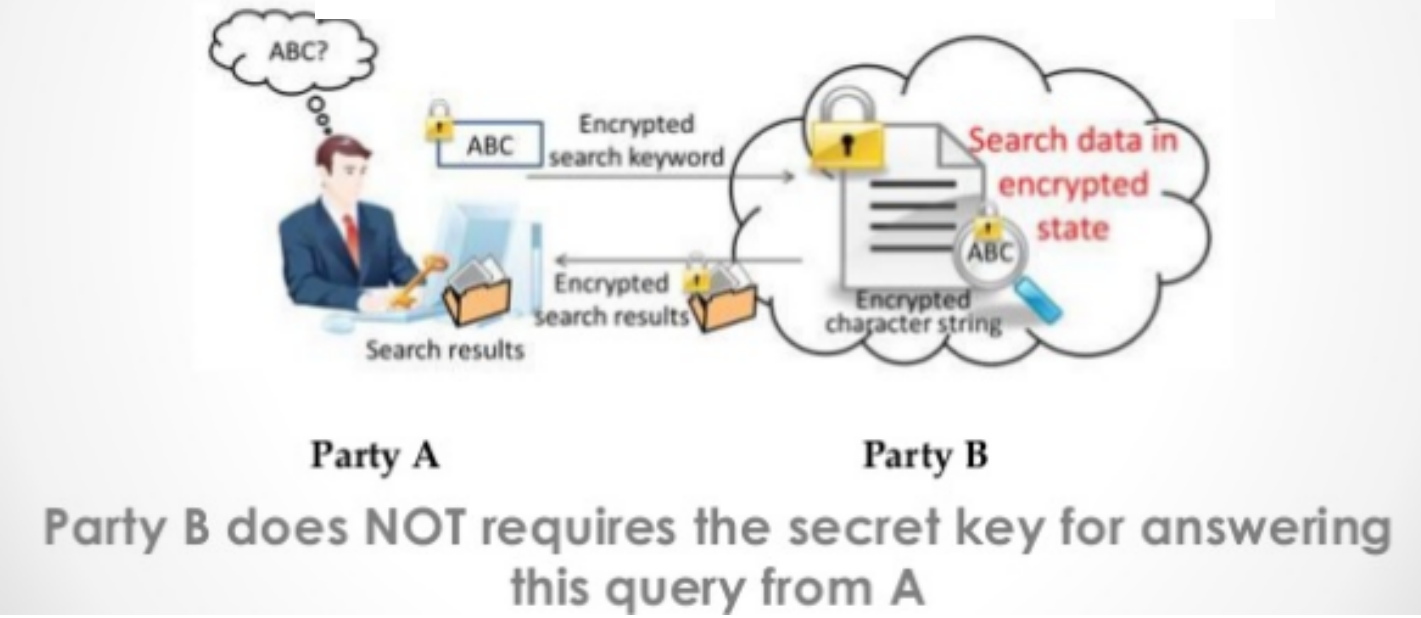
(c) Local privacy, where the central server might be malicious

An illustration of different privacy –enhancing mechanisms in one round of FL.
M denotes a randomized mechanism used to privatize the data. (a), no privacy protection is added.
(b), With global privacy, the model updates are private to all third parties other than a single trusted party (the central server). Training in worker node uses noisy version of aggregated weight: may lose accuracy.
(c), With local privacy, the individual model updates are also private to the server. Each worker adds local noise and use noisy version of aggregated weights for training: will also lose accuracy.

Homomorphic Encryption (HE)

- Secure the learning process by **computing/learning on encrypted data**
- **Without revealing the values of the decrypted data**
- Differ from other forms of encryptions, use an algebraic system to allow a variety of computations (or operations) on the encrypted data
- Has currently been applied in some settings, e.g., training linear models

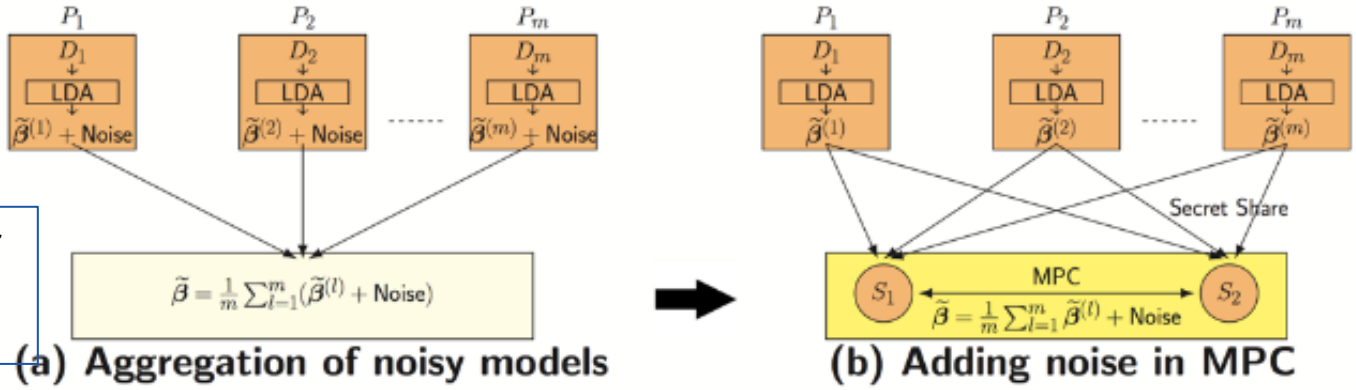
$$Enc(m_1) \circ Enc(m_2) = Enc(m_1 + m_2),$$



Secure multiparty computation (SMC)

- Sensitive datasets owned by different organizations
- This protocol enables **multiple parties to collaboratively compute an agreed-upon function** without leaking input information from any party except for what can be inferred from the output
- SMC can be **combined** with DP to achieve storing privacy guarantees
- SMC can also use secret sharing approach (multiple-server in FL, only sufficient number of servers leak, information can leak)
- Need to be carefully designed and implemented for large-scale system

P_i data owner
S_i party
$\tilde{\beta}$ model



[11] B. Jayaraman, L. Wang, D. Evans and Q. Gu, “Distributed Learning without Distress: Privacy-Preserving Empirical Risk Minimization,” 32nd Conference on Neural Information Processing Systems (NeurIPS). Montreal, Canada. December 2018.

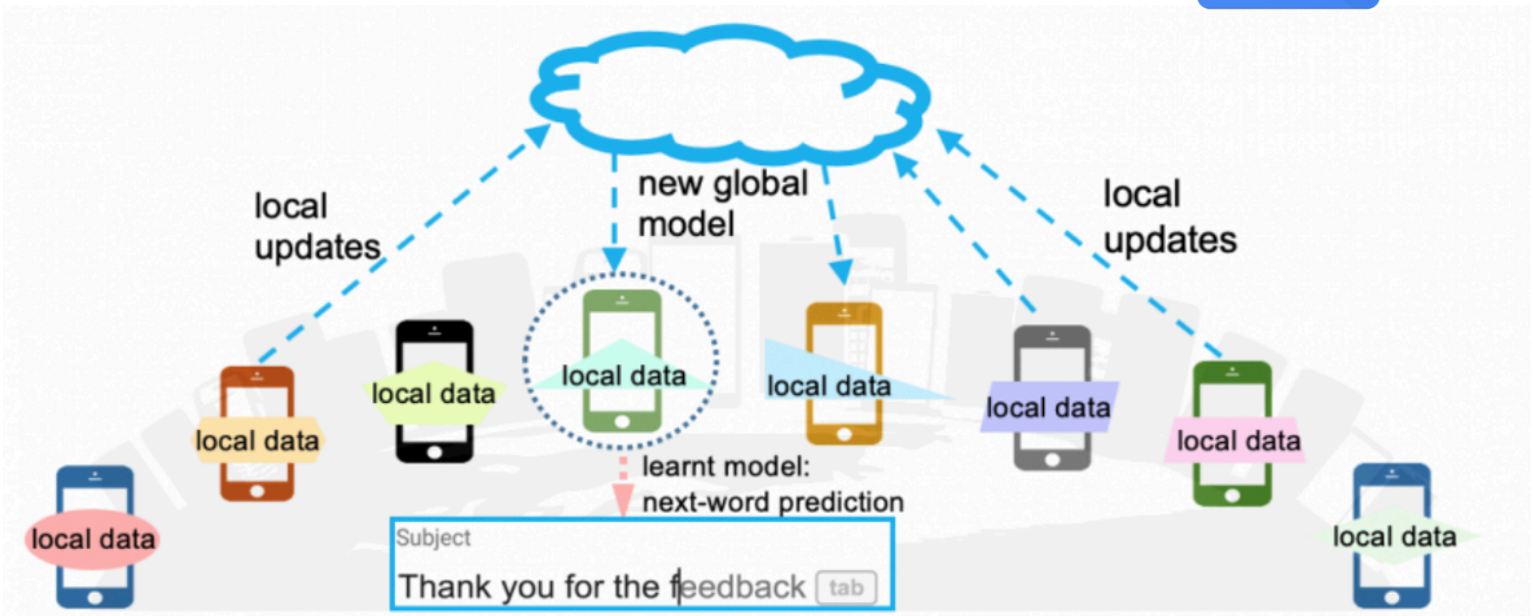
Application Examples

Two canonical applications:

- Learning over smart phones
- Learning across organizations

Learning over smart phones [6]

Text prediction on mobile phones (e.g., Gboard)

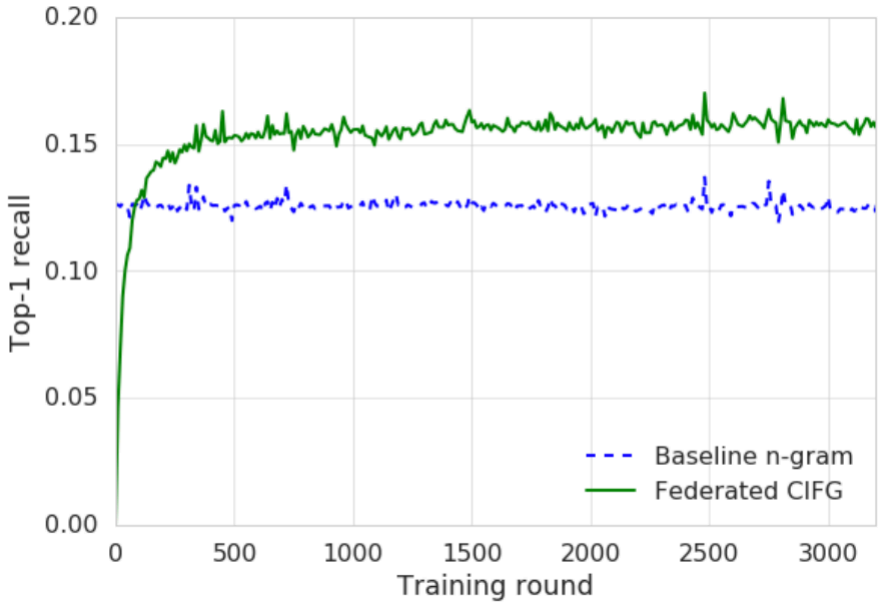


An example of FL for the task of next-word prediction on mobile phones. Devices communicate with a central server periodically to learn a global model. FL preserves user privacy and reduces comm. on the network by keeping data localized.

Gboard

- The first large-scale deployment of federated learning in the real world was as part of **Google keyboard application, Gboard**.
- The technique to improve word suggestions **without compromising user privacy.**
- Under the previous machine learning approach, developing better keyboard predictions would have been tremendously invasive – everything we typed, all of our private messages and strange Google searches would have to have been sent to a central server for analysis.
- Currently, Google use their federated learning for typing prediction. Because learning is on user devices, it is able to learn from the words that users type in, summarize the key information and then send it back to the server. **These summaries are then used to enhance Google's predictive text feature, which is then tested and pushed out to users**.

FL for next-word prediction



Top-1 recall (prediction accuracy) as a function of training round.

n-gram model is a type of probalistic language model for predicting the next item in a sequence.

[12] Hard, Andrew, et al. "Federated learning for mobile keyboard prediction." arXiv preprint arXiv:1811.03604 (2018).

FL for personal healthcare

Use FL across multiple hospitals to develop robust AI models **without** sharing personal data [6]

Another application of FL for personal healthcare via learning over heterogeneous electronic medical records distributed across multiple hospitals.

FL for personal healthcare (cont'd)

Data privacy and security are critical in healthcare. Many organizations store significant amounts of both sensitive and valuable patient data, which is also keenly sought by hackers.

In 2018, ***Intel*** partnered with the University of Pennsylvania's Center for Biomedical Image Computing and Analytics to demonstrate how federated learning could be applied to medical imaging as a proof of concept.

The collaboration revealed that under a federated learning approach, their particular deep learning model could be trained to be **99 percent** as accurate as the same model trained through traditional methods (centralized one).

[13] https://www.comparitech.com/blog/information-security/federated-learning/

FL for personal healthcare (cont'd)

Table: Performance of centralized, original federated and Federated-autonomous learning

Training method	AUCROC	AUCPR
Centralized learning	0.79	0.21
Original federated learning	0.75	0.16
Federated autonomous deep learning (FADL)	0.79	0.23

[14] Liu, Dianbo, et al. "Fadl: Federated-autonomous deep learning for distributed electronic health record." arXiv preprint arXiv:1811.11400 (2018).

Potential Research Direction on FL

- Most of the existing FL schemes consider exchanging the raw model parameters without any privacy guarantees
- Slave-worker model is the mainstream
  - Need a trusted server
  - may consider the **decentralized** architecture
- Main research directions are to improve the effectiveness, efficiency, and privacy (three key metrics)
  - Need to other research topics: e.g., **fairness and incentive** mechanisms

References

[1] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. Aguera y Arcas, “Communication-efficient learning of deep networks from decentralized data,” In *Proceedings* of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS), 2017.

[2] J. Konevcny, H. McMaha and D. Ramage, ”Federated Optimization: Distributed Optimiztion Beyond DataCenter, ” *arXiv preprint arXiv:1511.03575*, 2015.

[3] https://ai.googleblog.com/2017/04/federated-learning-collaborative.html

[4] L. Xiang, et al. "On the convergence of fedavg on non-i.i.d data." arXiv preprint arXiv:1907.02189 (2019).

[5] S. Caldas, J. Konecˇny, H. B. McMahan, and A. Talwalkar, Expanding the reach of federated learning by reducing client resource requirements. 2018. [Online]. Available: arXiv:1812.07210, 2018.

[6] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, “Federated learning: Challenges, methods, and future directions,” https://arxiv.org/abs/1908.07873.

[7] L. Liu, J. Zhang, S. Song, and K. B. Letaief. Edge-assisted hierarchical federated learning with non-i.i.d data. arXiv preprint arXiv:1905.06641, 2019.

[8] Q. Ho, J. Cipar, H. Cui, S. Lee, J. K. Kim, P. B. Gibbons, G. A. Gibson, G. Ganger, and E. P. Xing. More effective distributed ML via a stale synchronous parallel parameter server. In Advances in Neural Information Processing Systems, 2013.

[9] R. Tandon, Q. Lei, A. G. Dimakis, and N. Karampatziakis. Gradient coding: Avoiding stragglers in distributed learning. In International Conference on Machine Learning, 2017.

[10] T. Li, A. Kumar Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, “Federated optimization in heterogeneous networks,” 2018, arXiv:1812.06127. [Online]. Available: http://arxiv.org/abs/1812.06127.

[11] B. Jayaraman, L. Wang, D. Evans and Q. Gu, “Distributed Learning without Distress: Privacy-Preserving Empirical Risk Minimization,” 32nd Conference on Neural Information Processing Systems (NeurIPS). Montreal, Canada. December 2018.

[12] Hard, Andrew, et al. "Federated learning for mobile keyboard prediction." arXiv preprint arXiv:1811.03604 (2018).