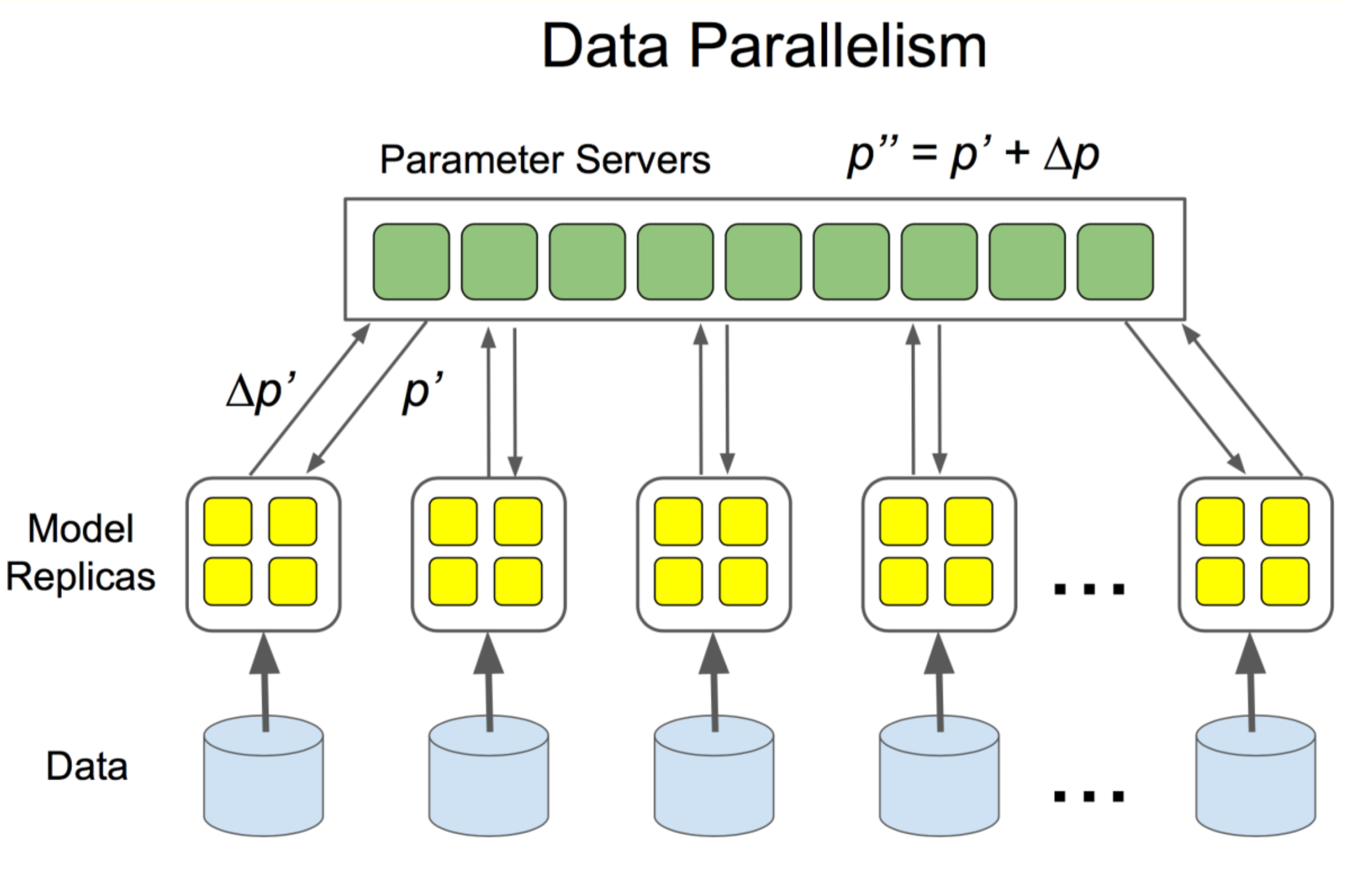
FEDERATED LEARNING: SYSTEM AND ALGORITHM

Xiaoyang Wang
University of Illinois

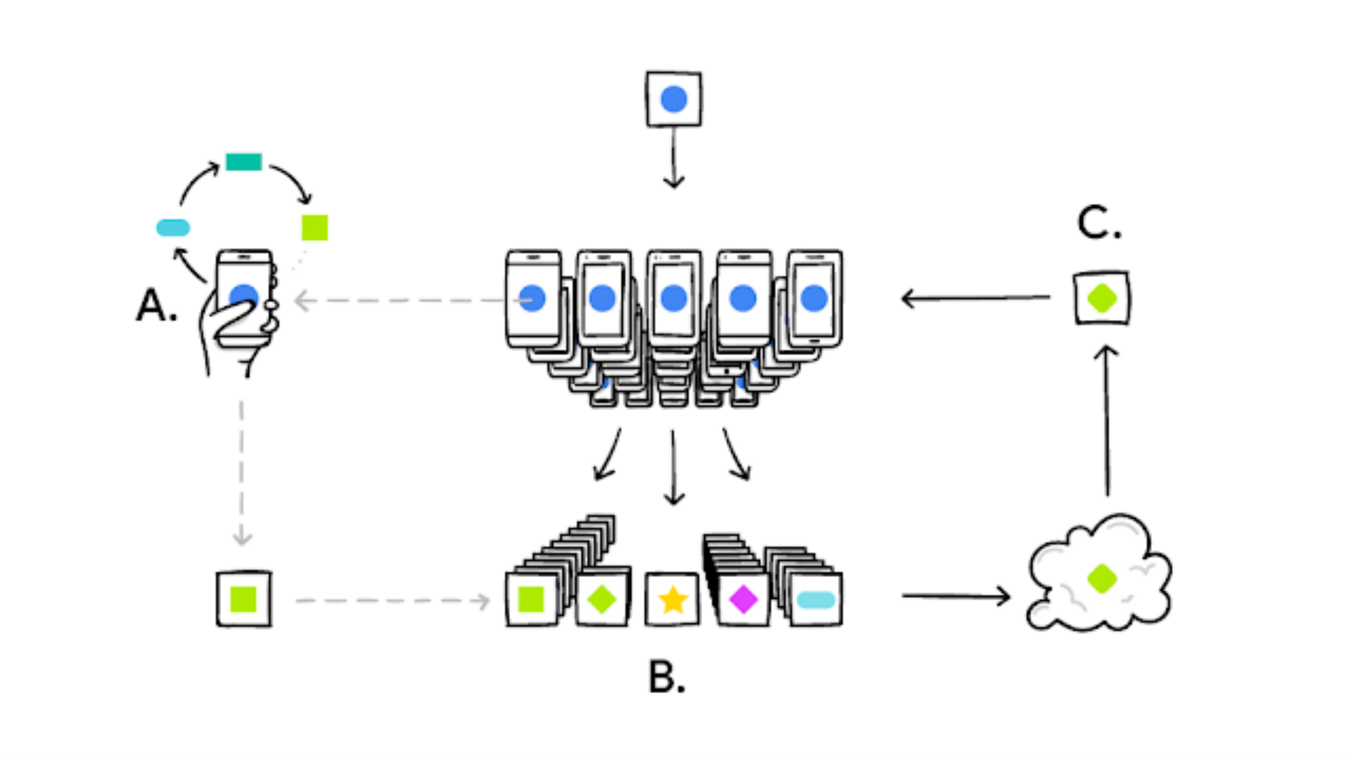
Distributed Machine Learning in Data Centers

- The data is stores in the data center.
- High quality network connection.
- IID data distribution.



Federated Learning

- The data is stored separately.
- Can’t share the data due to privacy.
- Limited network connection.
- Non-IID data distribution.



Federated Learning: Strategies for Improving Communication Efficiency

- Proposed two general approaches to reduce communication cost for gradient updates:
  - 1. Structured updates:
    - a. Low rank approximation: Express a matrix with size (m, n) with the product of two matrix with size (m, k) and (k, n).
    - b. Random mask: Restrict the gradient matrix to be a sparse matrix.
  - 2. Sketched updates:
    - a. Subsampling: Randomly sample a subset.
    - b. Quantization: Use less bit to store the gradient.

- General problem of these approaches:
  - They are lossy strategies.
  - We need a ground truth model to tell us the how much accuracy we lose.
    - If we have a ground truth model, why bother training a lossy one?
    - A accuracy bound is desirable.

Konečný, Jakub, et al. "Federated learning: Strategies for improving communication efficiency." *arXiv preprint arXiv:1610.05492* (2016).

Communication-Efficient Learning of Deep Networks from Decentralized Data

- Proposed *FederatedAveraging* algorithm.
- No gradient exchange.
- Average the weights from each client model to assemble a global model.
- Algorithm is evaluated empirically, no theorical analysis.

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**
      $w^k_{t+1} \leftarrow \text{ClientUpdate}(k, w_t)$
    $w_{t+1} \leftarrow \sum_{k=1}^{K} \frac{n_k}{n} w^k_{t+1}$

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

McMahan, H. Brendan, et al. "Communication-efficient learning of deep networks from decentralized data." arXiv preprint arXiv:1602.05629 (2016). Screen Shot 2019-10-20 at 6.09.18 PM

Federated Machine Learning: Concept and Applications

- A review paper.
- Defines three type of federated learning:
  - Horizontal federated learning: Same feature space, same label space, different user ID.
  - Vertical federated learning: Different feature space, different label space, same user ID
  - Federated transfer learning: Different feature space, different label space, different user ID.
- Two papers from previous slides are Horizontal federated learning.

Yang, Qiang, et al. "Federated machine learning: Concept and applications." ACM Transactions on Intelligent Systems and Technology (TIST) 10.2 (2019): 12.

Federated Multi-Task Learning

- Showed the multi-task learning is naturally suited to handle challenges in federated learning.
- W is a matrix (d, m) whose *t*-th column is the weight vector for the *t*-th task
- The matrix Ω (m, m) models relationships amongst tasks

$$\min_{\mathbf{W},\mathbf{\Omega}} \left\{ \sum_{t=1}^{m} \sum_{i=1}^{n_t} \ell_t(\mathbf{w}_t^T \mathbf{x}_t^i, y_t^i) + \mathcal{R}(\mathbf{W}, \mathbf{\Omega}) \right\}$$

$$\mathcal{R}(\mathbf{W}, \mathbf{\Omega}) = \lambda_1 \, \mathrm{tr}\left(\mathbf{W}\mathbf{\Omega}\mathbf{W}^T\right) + \lambda_2 \|\mathbf{W}\|_F^2$$

Loss function

Smith, Virginia, et al. "Federated multi-task learning." *Advances in Neural Information Processing Systems*. 2017.

Federated Multi-Task Learning

- Observation 1: In general, the loss function is not jointly convex in W and Ω, and even in the cases where loss function is convex, solving for W and Ω simultaneously can be difficult.

- Observation 2: When fixing Ω, updating W depends on both the data X, which is distributed across the nodes, and the structure Ω, which is known centrally.

- Observation 3: When fixing W, optimizing for Ω only depends on W and not on the data X.

- Solution: update W locally and update Ω with a centralized node.

$$\min_{\mathbf{W},\mathbf{\Omega}} \left\{ \sum_{t=1}^{m} \sum_{i=1}^{n_t} \ell_t(\mathbf{w}_t^T \mathbf{x}_t^i, y_t^i) + \mathcal{R}(\mathbf{W}, \mathbf{\Omega}) \right\}$$

$$\mathcal{R}(\mathbf{W}, \mathbf{\Omega}) = \lambda_1 \operatorname{tr}\left(\mathbf{W}\mathbf{\Omega}\mathbf{W}^T\right) + \lambda_2 \|\mathbf{W}\|_F^2$$

Loss function

Federated Multi-Task Learning

- Tolerated straggler in distributed learning.
- Added a parameter theta to relax the consistency.

Definition 1 (Per-Node-Per-Iteration-Approximation Parameter)**.** At each iteration h, we define the accuracy level of the solution calculated by node t to its subproblem (4) as:

$$\theta_t^h := \frac{\mathcal{G}_t^{\sigma'}(\Delta\boldsymbol{\alpha}_t^{(h)}; \mathbf{v}^{(h)}, \boldsymbol{\alpha}_t^{(h)}) - \mathcal{G}_t^{\sigma'}(\Delta\boldsymbol{\alpha}_t^{\star}; \mathbf{v}^{(h)}, \boldsymbol{\alpha}_t^{(h)})}{\mathcal{G}_t^{\sigma'}(\mathbf{0}; \mathbf{v}^{(h)}, \boldsymbol{\alpha}_t^{(h)}) - \mathcal{G}_t^{\sigma'}(\Delta\boldsymbol{\alpha}_t^{\star}; \mathbf{v}^{(h)}, \boldsymbol{\alpha}_t^{(h)})}\,, \tag{5}$$

Federated Learning

- Other work:
  - Share a minimum amount of data over all users to address the non-i.i.d problem.
  - Privacy preserving learning.

Relationship of the Federated Learning Problem and the Research Areas

- Federated learning.

- Distributed optimization.
- Communication efficient algorithms.
- Async optimization.
- Decentralized optimization.
- Machine learning on non-i.i.d data.