USER SELECTION IN FEDERATED LEARNING

Dipayan Mitra, **Ashish Khisti**

Department of Electrical and Computer Engineering, University of Toronto

Outlines

INTRODUCTION

Goal

K number of users wish to train a state-of-the-art machine learning model with their respective data $\{ \mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_K \}$.

Goal

K number of users wish to train a state-of-the-art machine learning model with their respective data $\{ \mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_K \}$.

Centralized Solution

Consolidate data from K users, as $\mathcal{D} = \mathcal{D}_1 \cup \mathcal{D}_2 \cup \cdots \cup \mathcal{D}_K$, into a server to train a global model.

Goal

K number of users wish to train a state-of-the-art machine learning model with their respective data $\{\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_K\}$.

Centralized Solution

Consolidate data from K users, as $\mathcal{D} = \mathcal{D}_1 \cup \mathcal{D}_2 \cup \cdots \cup \mathcal{D}_K$, into a server to train a global model.

Note: Assume the global accuracy to be A^c.

Is it really a nice way to train a global model?

Major Drawbacks

1. User data is centralized.
2. User data might contain private information.

Is there a way to train a model without centralizing users' data, i.e. ensuring 'privacy by default'?

Goal

K number of users wish to train a state-of-the-art machine learning model, collectively, without sharing their respective data $\mathcal{D}_i, \forall i \in 1, \ldots, K$; to other users.

Goal
K number of users wish to train a state-of-the-art machine learning model, collectively, without sharing their respective data $\mathcal{D}_i, \forall i \in 1, \ldots, K$; to other users.

Note: Assume the global accuracy to be A^f.

Goal

K number of users wish to train a state-of-the-art machine learning model, collectively, without sharing their respective data $\mathcal{D}_i, \forall i \in 1, \ldots, K$; to other users.

Note: Assume the global accuracy to be A^f.

In a practical federated learning setup, for $\delta \geq 0$,

$$|A^f - A^c| = \delta$$

Goal

K number of users wish to train a state-of-the-art machine learning model, collectively, without sharing their respective data $\mathcal{D}_i, \forall i \in 1, \ldots, K$; to other users.

Note: Assume the global accuracy to be A^f.

In a practical federated learning setup, for $\delta \geq 0$,

$$|A^f - A^c| = \delta$$

Note: A 'federation of users' participate in the learning process, hence the name federated learning.

Problem Formulation

Problem Formulation

1. K number of users participate in the learning process contributing to a total of n number of data points.

Problem Formulation

1. K number of users participate in the learning process contributing to a total of n number of data points.
2. Each user, k, has a set of data points $\mathbf{S}$ of size $n^{(k)} = |\mathbf{S}|$, i.e. $n = \sum_{k=1}^{K} n^{(k)}$.

Problem Formulation

1. K number of users participate in the learning process contributing to a total of n number of data points.

2. Each user, k, has a set of data points $\mathbf{S}$ of size $n^{(k)} = |\mathbf{S}|$, i.e. $n = \sum_{k=1}^{K} n^{(k)}$.

3. In order to train a machine learning model, with parameters w, on the labled data points $(\mathbf{x}, \mathbf{y})$ for each k, we consider a local objective function of $f^k(w) = \frac{1}{n^{(k)}} \sum_{i \in \mathbf{S}} l(x_i, y_i; w)$.

Problem Formulation

1. K number of users participate in the learning process contributing to a total of n number of data points.
2. Each user, k, has a set of data points $\mathbf{S}$ of size $n^{(k)} = |\mathbf{S}|$, i.e. $n = \sum_{k=1}^{K} n^{(k)}$.
3. In order to train a machine learning model, with parameters w, on the labled data points $(\mathbf{x}, \mathbf{y})$ for each k, we consider a local objective function of $f^k(w) = \frac{1}{n^{(k)}} \sum_{i \in \mathbf{S}} l(x_i, y_i; w)$.
4. In a federated setting, we can write the objective function $f^f(w)$ in the following form,

$$\min_{w} f^f(w) = \sum_{k=1}^{K} p_k f^k(w) = \mathbb{E}_k[f^k(w)]$$

where $p_k = \frac{n^{(k)}}{n}$, $p_k \geq 0$ & $\sum_k p_k = 1$

SYSTEM LEVEL COMPARISON

Figure: Centralized learning

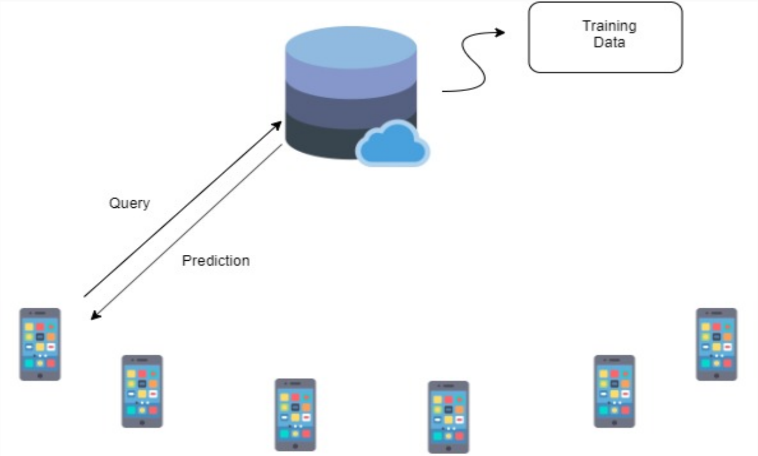

Figure: Centralized learning

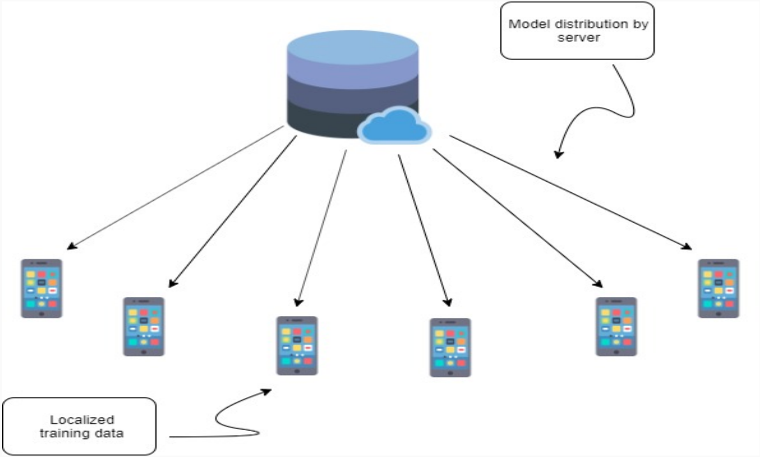

Figure: Federated learning

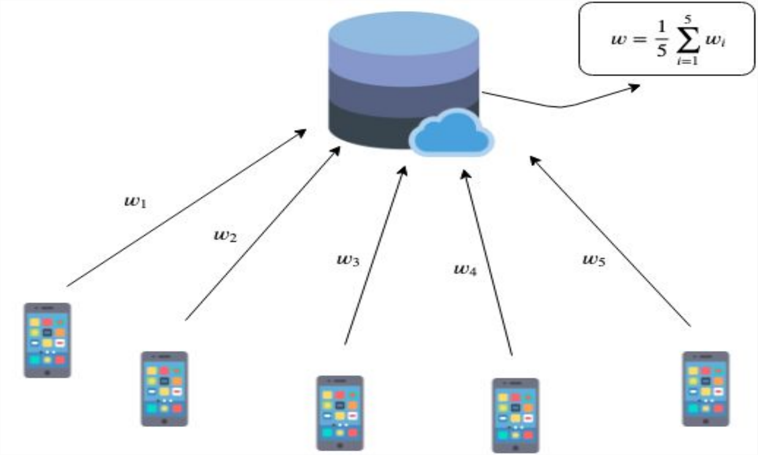

Figure: Federated learning

Federated Averaging ($Fed - Avg$) Algorithm [1]

Until Convergence:

Server:

1. Select K number of users randomly.
2. Send w_t, i.e. parameter update at t^{th} iteration, to all K users.

   User:

   1. Download parameter update w_t from the server.
   2. Run SGD locally, for E epochs, and obtain w_t^k.
   3. Upload $w_t - w_t^k$ to the server.

3. $w_{t+1} = w_t +$ weighted average of the parameter updates by K users.

[1] McMahan *et al.*, 'Communication-efficient Learning of Deep Networks from Decentralized Data', AISTATS, 2017.

User level advantages:

User level advantages:

1. Lesser communication with the server

Advantages

User level advantages:

1. Lesser communication with the server $\longrightarrow$ Saves bandwidth & battery.

Advantages

User level advantages:

1. Lesser communication with the server $\longrightarrow$ Saves bandwidth & battery.
2. Lesser user data sent to the could.

User level advantages:

1. Lesser communication with the server $\longrightarrow$ Saves bandwidth & battery.
2. Lesser user data sent to the could.

Developer level advantage:

User level advantages:

1. Lesser communication with the server $\longrightarrow$ Saves bandwidth & battery.
2. Lesser user data sent to the could.

Developer level advantage:

1. Localized data leading to new product opportunities.

User level advantages:

1. Lesser communication with the server $\longrightarrow$ Saves bandwidth & battery.
2. Lesser user data sent to the could.

Developer level advantage:

1. Localized data leading to new product opportunities.

Security:

User level advantages:

1. Lesser communication with the server $\longrightarrow$ Saves bandwidth & battery.
2. Lesser user data sent to the could.

Developer level advantage:

1. Localized data leading to new product opportunities.

Security:

1. More privacy preserving, assuming an honest but curious server.

USER SELECTION

Key Criterion

1. With the increase in K, communication bottleneck becomes a challenge for distributing model parameter updates

Key Criterion

1. With the increase in K, communication bottleneck becomes a challenge for distributing model parameter updates$\longleftarrow$ Communication-based challenge

Key Criterion

1. With the increase in K, communication bottleneck becomes a challenge for distributing model parameter updates$\longleftarrow$ Communication-based challenge

2. For a very large K, unreliability increase due to the chance of device drop-out during each iteration

1. With the increase in K, communication bottleneck becomes a challenge for distributing model parameter updates$\longleftarrow$ Communication-based challenge
2. For a very large K, unreliability increase due to the chance of device drop-out during each iteration$\longleftarrow$ Communication & hardware-based challenge

1. With the increase in K, communication bottleneck becomes a challenge for distributing model parameter updates$\longleftarrow$ Communication-based challenge
2. For a very large K, unreliability increase due to the chance of device drop-out during each iteration$\longleftarrow$ Communication & hardware-based challenge
3. With an increase in network size challenges like, fault tolerance, straggler mitigation, arise

1. With the increase in K, communication bottleneck becomes a challenge for distributing model parameter updates$\longleftarrow$ Communication-based challenge

2. For a very large K, unreliability increase due to the chance of device drop-out during each iteration$\longleftarrow$ Communication & hardware-based challenge

3. With an increase in network size challenges like, fault tolerance, straggler mitigation, arise $\longleftarrow$ Communication-based challenge

1. With the increase in K, communication bottleneck becomes a challenge for distributing model parameter updates$\longleftarrow$ Communication-based challenge
2. For a very large K, unreliability increase due to the chance of device drop-out during each iteration$\longleftarrow$ Communication & hardware-based challenge
3. With an increase in network size challenges like, fault tolerance, straggler mitigation, arise $\longleftarrow$ Communication-based challenge

Communication capacity of each user is one of the most used user selection criteria.

Key Criterion

How about statistical heterogeneity of users' data?

How about statistical heterogeneity of users' data?

1. With the increase in K, individual users generate data in a non-i.i.d. manner

How about statistical heterogeneity of users' data?

1. With the increase in K, individual users generate data in a non-i.i.d. manner $\longrightarrow$ Most of the distributed optimization algorithms become impractical.

How about statistical heterogeneity of users' data?

1. With the increase in K, individual users generate data in a non-i.i.d. manner $\longrightarrow$ Most of the distributed optimization algorithms become impractical.

2. Understanding of the underline structure relating the data distributions of various users have remains an open area of research

How about statistical heterogeneity of users' data?

1. With the increase in K, individual users generate data in a non-i.i.d. manner $\longrightarrow$ Most of the distributed optimization algorithms become impractical.
2. Understanding of the underline structure relating the data distributions of various users have remains an open area of research

Is this convincing reason to investigate user selection based on statistical heterogeneity?

WEIGHT DIVERGENCE

Weight Divergence on non-i.i.d Data [2]

How is non-i.i.d defined in this problem setup?

[2] Zhao *et al.*, 'Federated learning with non-i.i.d data', *Pre-print.* Available: https://arxiv.org/abs/1806.00582

Setup:

Number of Users: $K = 10$.

Dataset: MNIST and CIFAR-10, each having 10 classes, i.e. $C = 10$.

i.i.d. Case: Uniform distribution over 10 classes are randomly assigned to each user, i.e. k.

Non-i.i.d. Case:

1. Each user is assigned data partition from a single class $\longleftarrow$ Defined as '1-class non-IID'.

2. Data is sorted into 20 partitions and each user receives 2 partitions from 2 classes $\longleftarrow$ Defined as '2-class non-IID'.

Number of data-points: Amount of data $n^{(k)}$ for each client, resulting $n = \sum_{k=1}^{K} n^{(k)}$

[2] Zhao *et al.*, 'Federated learning with non-i.i.d data', *Pre-print.* Available: https://arxiv.org/abs/1806.00582

1. C class classification problem defined over a compact space X and label space Y. So, $Y = [C]$, where $[C] = \{1, 2, \ldots, C\}$.
2. $f : X \to S$, where probability simplex $S = \left\{ z | \sum_{i=1}^{C} z_i = 1, z_i \geq 0, \forall i \in [C] \right\}$. f is parameterized over w, i.e. weights of the neural network.
3. Population loss with cross-entropy loss is defined as,

$$\ell(\boldsymbol{w}) = \mathbb{E}_{\boldsymbol{x}, y \sim p} \left[\sum_{i=1}^{C} \mathbb{1}_{y=i} \log f_i(\boldsymbol{x}, \boldsymbol{w}) \right] = \sum_{i=1}^{C} p(y = i) \mathbb{E}_{\boldsymbol{x}|y=i} \left[\log f_i(\boldsymbol{x}, \boldsymbol{w}) \right]$$

4. The learning problem (ignoring the generalization error for simplicity) by directly optimizing the population loss,

$$\min_{\boldsymbol{w}} \sum_{i=1}^{C} p(y = i) \mathbb{E}_{\boldsymbol{x}|y=i} \left[\log f_i(\boldsymbol{x}, \boldsymbol{w}) \right]$$

How to find $\boldsymbol{w}$?

How to find $\boldsymbol{w}$?

The optimization is iteratively solved using SGD.

Centralized learning:

Parameter $\boldsymbol{w}$ after t^{th} update, denoted as $\boldsymbol{w}_t^{(c)}$, obtained as:

$$\boldsymbol{w}_t^{(c)} = \boldsymbol{w}_{t-1}^{(c)} - \eta \nabla_{\boldsymbol{w}} \ell \left(\boldsymbol{w}_{t-1}^{(c)} \right) = \boldsymbol{w}_{t-1}^{(c)} - \eta \sum_{i=1}^{C} p(y=i) \nabla_{\boldsymbol{w}} \mathbb{E}_{\boldsymbol{x}|y=i} \left[\log f_i \left(\boldsymbol{x}, \boldsymbol{w}_{t-1}^{(c)} \right) \right]$$

Centralized learning:

Parameter $\boldsymbol{w}$ after t^{th} update, denoted as $\boldsymbol{w}_t^{(c)}$, obtained as:

$$\boldsymbol{w}_t^{(c)} = \boldsymbol{w}_{t-1}^{(c)} - \eta \nabla_{\boldsymbol{w}} \ell \left(\boldsymbol{w}_{t-1}^{(c)} \right) = \boldsymbol{w}_{t-1}^{(c)} - \eta \sum_{i=1}^{C} p(y=i) \nabla_{\boldsymbol{w}} \mathbb{E}_{\boldsymbol{x}|y=i} \left[\log f_i \left(\boldsymbol{x}, \boldsymbol{w}_{t-1}^{(c)} \right) \right]$$

Federated learning:

Each user $k \in [K]$ performs SGD locally to obtain $\boldsymbol{w}_t^{(k)}$ as:

$$\boldsymbol{w}_t^{(k)} = \boldsymbol{w}_{t-1}^{(k)} - \eta \sum_{i=1}^{C} p^{(k)}(y=i) \nabla_{\boldsymbol{w}} \mathbb{E}_{\boldsymbol{x}|y=i} \left[\log f_i \left(\boldsymbol{x}, \boldsymbol{w}_{t-1}^{(k)} \right) \right]$$

Considering the synchronization is performed at each T^{th} step, $m-th$ such synchronization in the server produces the following update:

$$\boldsymbol{w}_{mT}^{(f)} = \sum_{k=1}^{K} \frac{n^{(k)}}{\sum_{k=1}^{K} n^{(k)}} \boldsymbol{w}_{mT}^{(k)}$$

Theorem: Given each user $k \in [K]$ with $n^{(k)}$ i.i.d samples following distribution $p^{(k)}$. If $\nabla_{\boldsymbol{w}} \mathbb{E}_{\boldsymbol{x}|y=i} \log f_i(\boldsymbol{x}, \boldsymbol{w})$ is $\lambda_{\boldsymbol{x}|y=i}$-Lipschitz for each class $i \in [C]$ and synchronization is performed at each each T step, the weight divergence after m^{th} synchronization follows the inequality shown below,

$$\begin{aligned}\left\|\boldsymbol{w}_{mT}^{(f)} - \boldsymbol{w}_{mT}^{(c)}\right\| \leq & \sum_{k=1}^{K} \frac{n^{(k)}}{\sum_{k=1}^{K} n^{(k)}} \left(a^{(k)}\right)^{T} \left\|\boldsymbol{w}_{(m-1)T}^{(f)} - \boldsymbol{w}_{(m-1)T}^{(c)}\right\| \\ & + \eta \sum_{k=1}^{K} \frac{n^{(k)}}{\sum_{k=1}^{K} n^{(k)}} \sum_{i=1}^{C} \left\|p^{(k)}(y=i) - p(y=i)\right\| \sum_{j=0}^{T-1} \left(a^{(k)}\right)^{j} g_{\max}\left(\boldsymbol{w}_{mT-1-k}^{(c)}\right)\end{aligned}$$

where $g_{\max}(\boldsymbol{w}) = \max_{i=1}^{C} \left\|\nabla_{\boldsymbol{w}} \mathbb{E}_{\boldsymbol{x}|y=i} \log f_i(\boldsymbol{x}, \boldsymbol{w})\right\|$ and $a^{(k)} = 1 + \eta \sum_{i=1}^{C} p^{(k)}(y=i)\lambda_{\boldsymbol{x}|y=i}$.

Key Takeaways:

Key Takeaways:

1. What are the main causes of divergence?

Key Takeaways:

1. What are the main causes of divergence?

   $\Longrightarrow$ Weight divergence caused by the $(m-1)^{th}$ update, i.e. $\left\|\boldsymbol{w}_{(m-1)T}^{(f)} - \boldsymbol{w}_{(m-1)T}^{(c)}\right\|$.

Key Takeaways:

1. What are the main causes of divergence?

   $\Longrightarrow$ Weight divergence caused by the $(m-1)^{th}$ update, i.e. $\left\|\boldsymbol{w}^{(f)}_{(m-1)T} - \boldsymbol{w}^{(c)}_{(m-1)T}\right\|$.
   $\Longrightarrow$ Distance between the data distribution on user k and the actual distribution for the whole population, i.e. $\sum_{i=1}^{C}\left\|p^{(k)}(y=i) - p(y=i)\right\|$.

Key Takeaways:

1. What are the main causes of divergence?

   $\Longrightarrow$ Weight divergence caused by the $(m-1)^{th}$ update, i.e. $\left\| \boldsymbol{w}^{(f)}_{(m-1)T} - \boldsymbol{w}^{(c)}_{(m-1)T} \right\|$.
   $\Longrightarrow$ Distance between the data distribution on user k and the actual distribution for the whole population, i.e. $\sum_{i=1}^{C} \left\| p^{(k)}(y=i) - p(y=i) \right\|$.
   $\Longrightarrow$ Divergence can be treated a proxy [3] to the accuracy, i.e. higher the divergence lower the accuracy.

[3] Although no theoretical analysis exists.

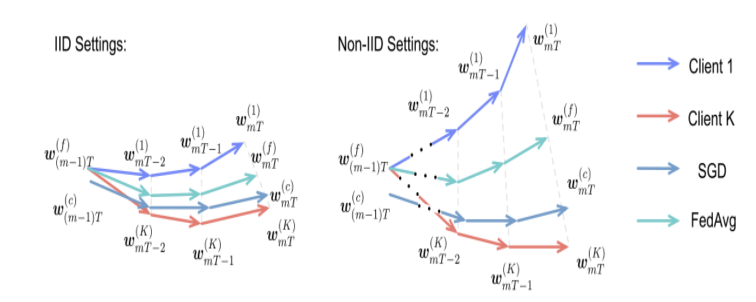

Figure: Comparison of weight divergence between i.i.d. and non-i.i.d. setup [3]

[3] Zhao *et al.*, 'Federated learning with non-i.i.d data', *Pre-print.* Available: https://arxiv.org/abs/1806.00582

How to mitigate the divergence?

How to mitigate the divergence?

$\Longrightarrow$ Select the users judiciously

How to mitigate the divergence?

$\Longrightarrow$ Select the users judiciously $\longrightarrow$ Select users with the highest amount of data and discard the rest.

How to mitigate the divergence?

$\Longrightarrow$ Select the users judiciously $\longrightarrow$ Select users with the highest amount of data and discard the rest.

How is that beneficial?

How to mitigate the divergence?

$\Longrightarrow$ Select the users judiciously $\longrightarrow$ Select users with the highest amount of data and discard the rest.

How is that beneficial?

$\Longrightarrow \sum_{k=1}^{K} \frac{n^{(k)}}{\sum_{k=1}^{K} n^{(k)}}$ reduces, causing reduction in divergence.

Problem Formulation:

$[\mathbf{U}] \implies$ Set of all users.

$[\mathbf{U_S}] \implies$ Set of selected users. ($[\mathbf{U_S}] \subseteq [\mathbf{U}]$)

With τ being the threshold, for selecting users complying with $|n^{(k)}| \geq \tau$.

Assumption: N users (with $|n^{(k)}|$ above τ) are selected from a set of K users.

Assumption: N users (with $|n^{(k)}|$ above τ) are selected from a set of K users.

Modified weight divergence:

$$\left\| \boldsymbol{w}_{mT}^{(f)} - \boldsymbol{w}_{mT}^{(c)} \right\| \leq \sum_{k=1}^{N} \frac{n^{(k)}}{n} \left(a^{(k)} \right)^{T} \left\| \boldsymbol{w}_{(m-1)T}^{(f)} - \boldsymbol{w}_{(m-1)T}^{(c)} \right\|$$
$$+ \eta \sum_{k=1}^{N} \frac{n^{(k)}}{n} \sum_{i=1}^{C} \left\| p^{(k)}(y=i) - p(y=i) \right\| \sum_{j=0}^{T-1} \left(a^{(k)} \right)^{j} g_{\max} \left(\boldsymbol{w}_{mT-1-k}^{(c)} \right) + \tilde{c}$$

Where, $\tilde{c} = \left\| \sum_{k=N+1}^{K} \frac{n^{(k)}}{n} p^{(k)}(y=i) \nabla_{\boldsymbol{w}} \mathbb{E}_{\boldsymbol{x}|y=i} [\log f_i(\boldsymbol{x}, \boldsymbol{w}_{(m-1)T})] \right\|$

Key Takeaways:

Key Takeaways:

1. $\tilde{c}$ is a constant depending on selection of τ.

Key Takeaways:

1. $\tilde{c}$ is a constant depending on selection of τ.
2. $\left(\left\|\boldsymbol{w}_{mT}^{(f)} - \boldsymbol{w}_{mT}^{(c)}\right\|\right)_{\mathbf{U}} \geq \left(\left\|\boldsymbol{w}_{mT}^{(f)} - \boldsymbol{w}_{mT}^{(c)}\right\|\right)_{\mathbf{U_S}}$

What can possibly go wrong with such user selection technique?

What can possibly go wrong with such user selection technique?

The collectively learned model would be biased to a set of users having higher $n^{(k)}$.

Any solution?

MITIGATING BIAS IN LEARNING

Definition: [3] Given two trained models with parameters $\boldsymbol{w}$ and $\boldsymbol{w}'$, a more fair solution to the objective of the federated learning is obtained by model $\boldsymbol{w}$ when,

$$Var(A_1, A_2, \ldots, A_K) \leq Var(A'_1, A'_2, \ldots, A'_K)$$

where A_i & $A'_i, \forall i = 1, \ldots, K$, are the accuracy obtained by using model $\boldsymbol{w}$ and $\boldsymbol{w}'$ respectively.

[3] Li *et al.*, 'Fair Resource Allocation in Federated Learning', Pre-print, Available: https://arxiv.org/abs/1905.10497

Resembling α-fairness [4], for $q \geq 0$, a q-fair federated learning objective can be expressed as,

$$\min_{w} f_q^f(w) = \sum_{k=1}^{m} \frac{p_k}{q+1} f_k^{q+1}(w)$$

Note:

1. Hyper-parameter q is trained through an iterative algorithm.
2. $q = 0$ provides the classical definition of federated learning objective.

[4] T. Lan *et al.*, 'An axiomatic theory of fairness in network resource allocation', In *Conference on Information Communications*, pages 1343–1351, 2010.

[5] Li *et al.*, 'Fair Resource Allocation in Federated Learning', Pre-print, Available: https://arxiv.org/abs/1905.10497

FUTURE DIRECTIONS

Future Works

1. A better user selection strategy by minimizing the bias introduced by the model.
2. Combining communication-based techniques for user selection with a data-driven selection technique.
3. Developing a user-reward strategy based on game theoretic formulations, for example Stackelberg games.

BACKUP SLIDES