Pisces: Efficient Federated Learning via Guided Asynchronous Training

THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

UNIVERSITY OF TORONTO

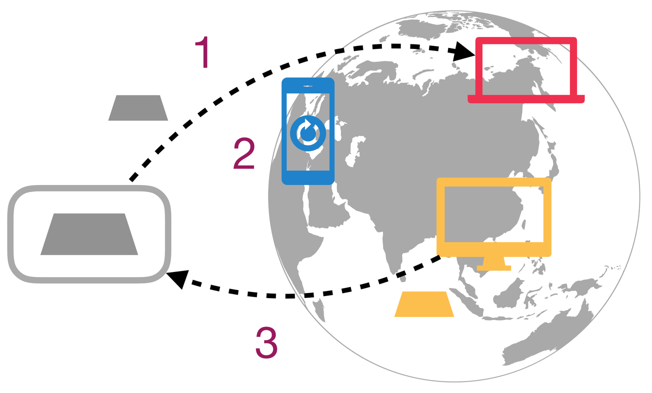
Zhifeng Jiang[1], Wei Wang[1], Baochun Li[2], Bo Li[1]

[1]HKUST [2]University of Toronto

FL & Synchronous FL

Federated learning allows multiple clients to collaboratively train a global model with their private data locked in local storage.

In **synchronous** FL, the server advances the global model on a round basis.

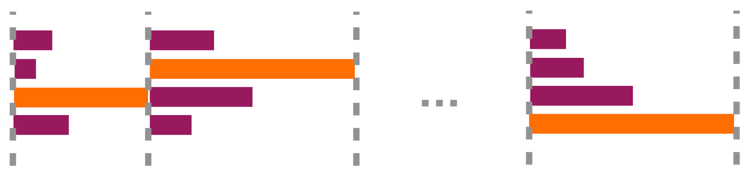


1. Participant selection
2. Local training
3. Model aggregation

We primarily care about the **time-to-accuracy** performance, i.e., the elapsed time for the global model to reach a target accuracy.

Motivation for Asynchronous FL

SyncFL

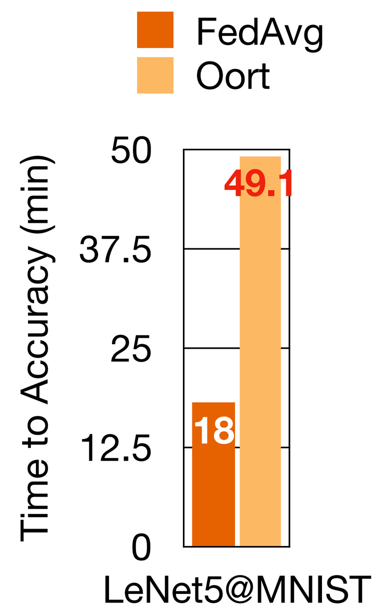


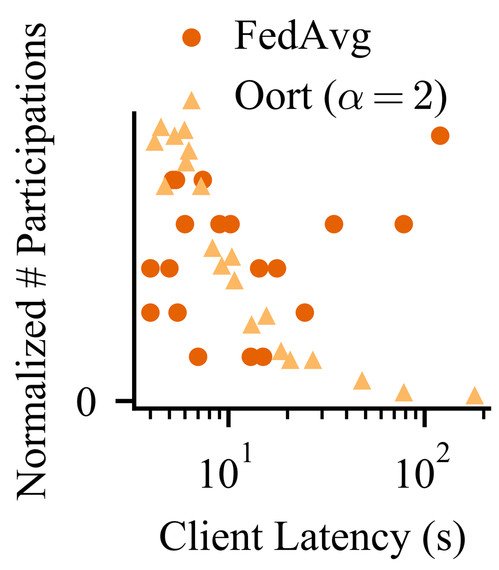
In vanilla sync FL[1], up to 57% of the training time is spent on waiting for **stragglers**.

To accelerate, Oort[2] **selects** participants who have high speeds and high-quality data.

$$U_i^{Oort} = \underbrace{|B_i|\sqrt{\frac{1}{|B_i|}\sum_{k\in B_i} Loss(k)^2}}_{Data\ quality} \times \underbrace{\left(\frac{T}{t_i}\right)^{\mathbb{1}(T<t_i)\times\alpha}}_{Speed}$$

However, when data quality and speeds are **at odds**, sync FL has to **trade** one for the other.



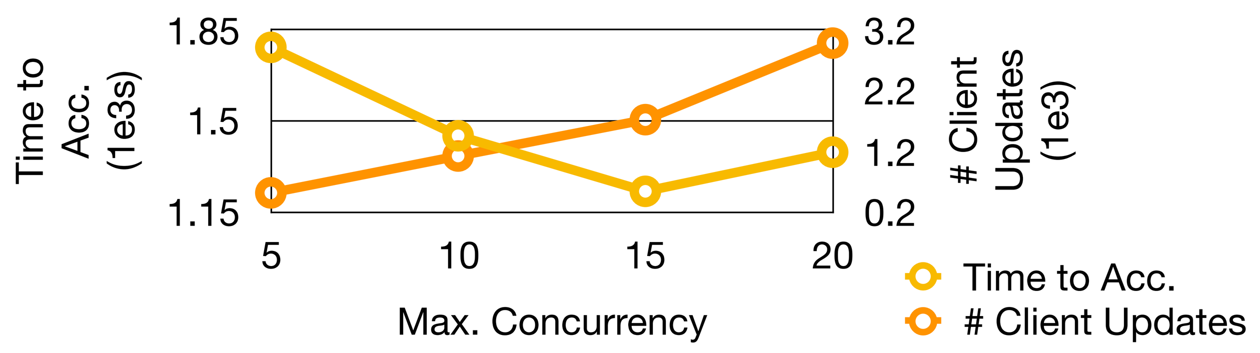


Tolerance for slow clients: Low ——▲——————▼—— High

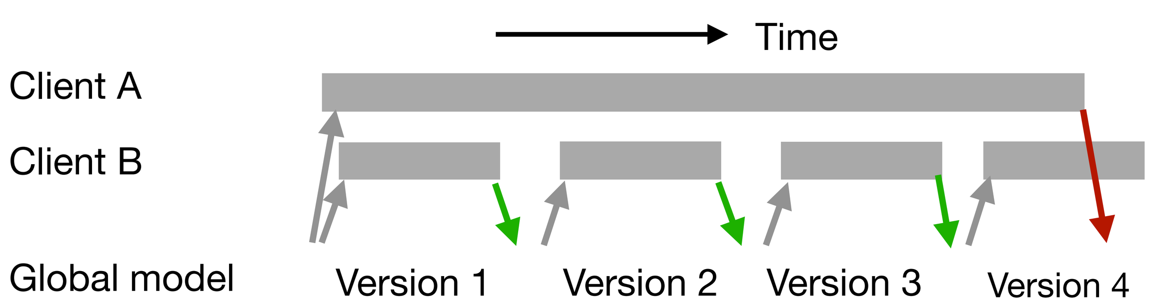
AsyncFL Offers **freedom** in participant selection and model aggregation.

Challenges in Async FL

1. Freedom in selection solicits high concurrency which can hurt **resource efficiency**.



2. Freedom in aggregation solicits **stale** local updates, which can hurt **model convergence**.



Optimizing Async FL w/ Pisces

1. Pisces selects clients with **high data quality and low chance to generate stale updates**.

$$U_i^{Pisces} = \underbrace{|B_i|\sqrt{\frac{1}{|B_i|}\sum_{k\in B_i} Loss(k)^2}}_{Data\ quality} \times \underbrace{\frac{1}{(\tilde{\tau}_i+1)^{\beta}}}_{Staleness},$$

- To avoid being misled by corrupted clients, we detect outlier losses via **clustering**.
- As clients' staleness evolve steadily, we estimate it based on the **moving average**.

2. Pisces **adapts the aggregation interval** to currently slowest client's pace in a guided way.

- The **staleness** of each local update is proven to be **bounded** within any predefined limit b.
- This further enables us to prove the **convergence** in smooth non-convex settings:

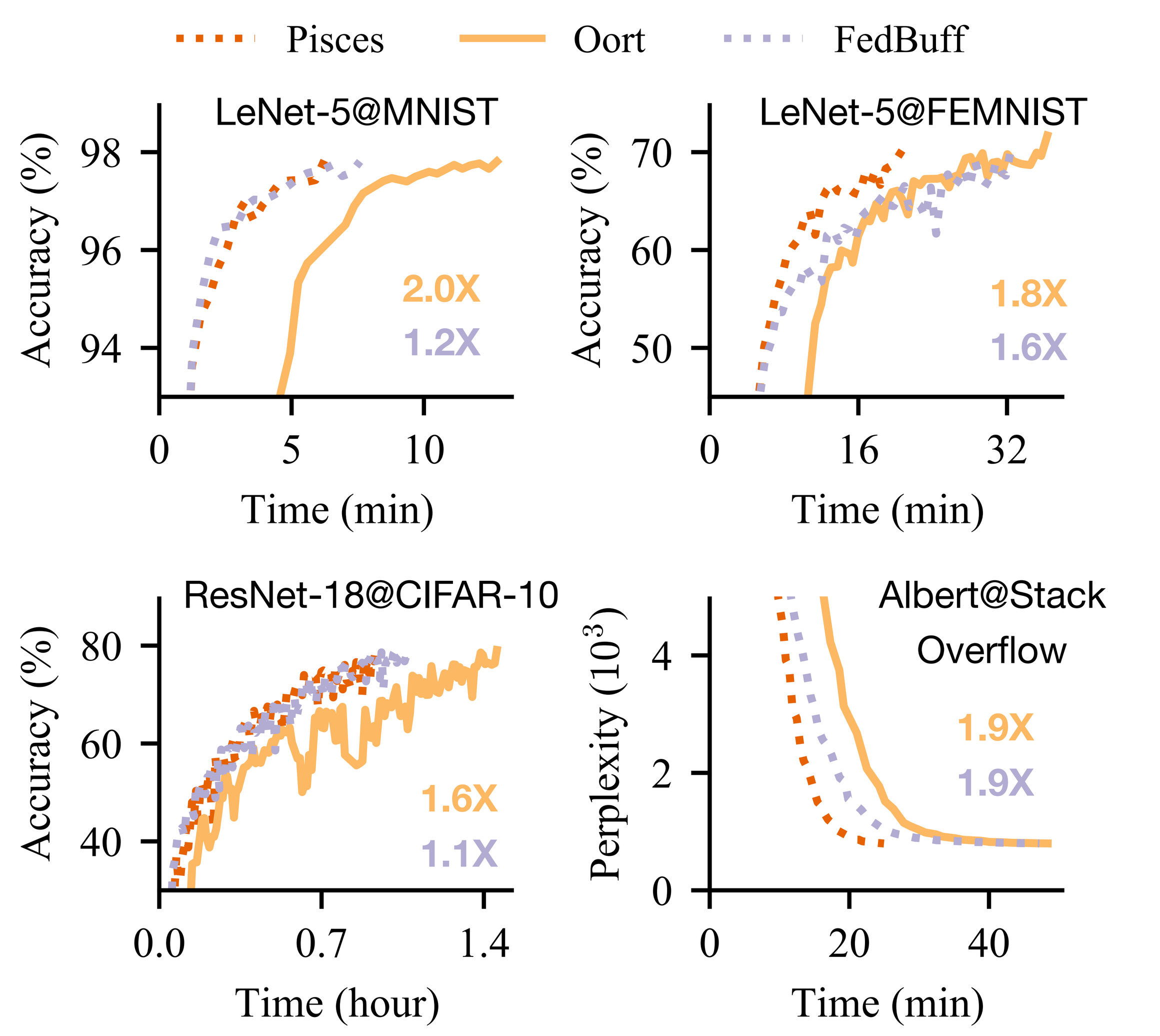
$$\frac{1}{T}\sum_{t=0}^{T-1}\left\|\nabla f(w^t)\right\|^2 \le \frac{2\left(f(w^0)-f^*\right)}{\alpha(Q)T} + \frac{L}{2}\frac{\beta(Q)}{\alpha(Q)}\sigma_\ell^2 + 3L^2 Q\beta(Q)\left(b^2+1\right)\left(\sigma_\ell^2+\sigma_g^2+G\right).$$

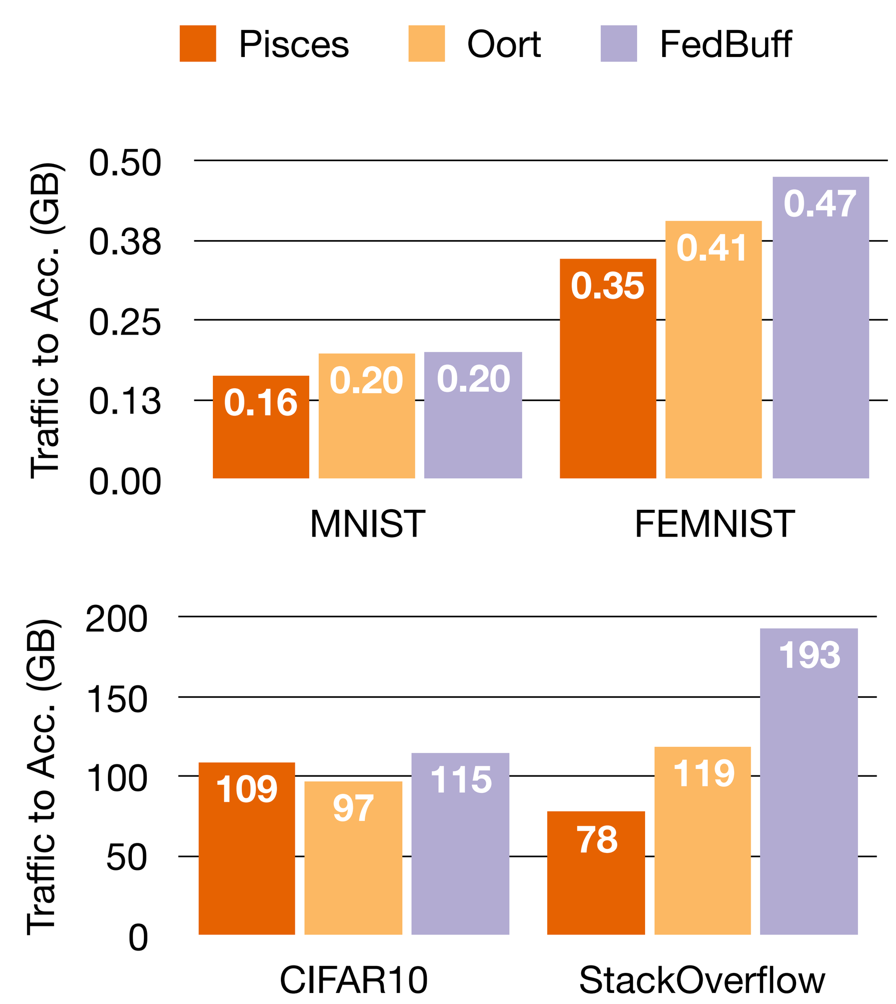
Evaluating Efficiency and Sensitivity

Experiment Setup

- Cluster: 200 clients in the AWS public cloud; 10% clients run concurrently.
- Heterogeneity: (1) Zipf's speeds; (2) native or synthetic non-IID data partitions.
- Baseline: Oort[2] and FedBuff[3], SOTA sync FL method and async FL

Pisces Outperforms in Time Performance and Network Footprint





Pisces is Insensitive to Training Environments or Learning Tasks

- Participation scales (100 to 400 clients)
- Corrupted client portion (0% to 20%)
- Staleness Penalty factors (0.2 to 0.8)
- Optimizers (SGD/Adam/FedProx[4]
- Model architectures (LeNet-5/ customized CNN/ResNet-18)

Reference

[1] Communication-efficient learning of deep networks from decentralized data. McMahan et al. *AISTATS*, 2017.

[2] Oort: Efficient federated learning via guided participant selection. Fan et al. *OSDI* 2021.

[3] Federated learning with buffered asynchronous aggregation, Nguyen et al. *AISTATS*, 2022.

[4] Federated optimization in heterogeneous networks. Li et al. *MLSys*, 2020.

Code available at:

Contact:
Zhifeng Jiang
zjiangaj@cse.ust.hk