Automating FaaS-based Federated Learning

Matt Baughman

ParslFest
October 20, 2023

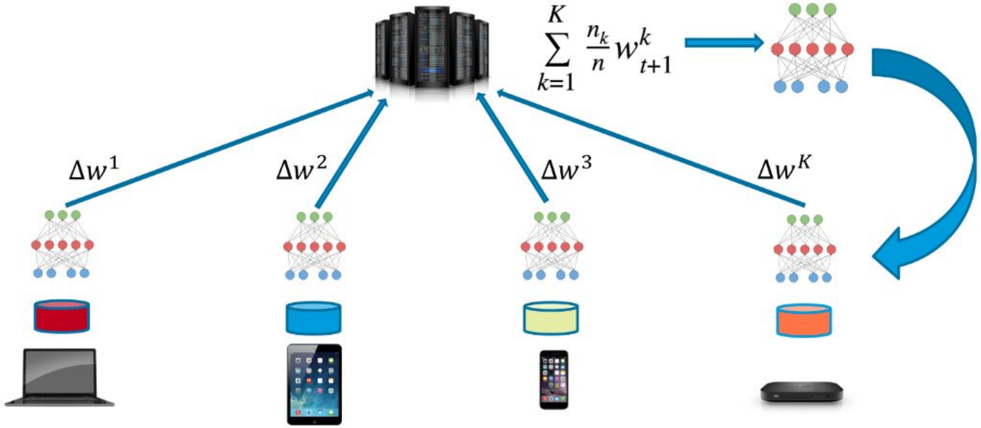
A few main ideas…

The Application: ML/FL

The Challenges: Workload Balancing, Tuning Hyperparameters, Robustness

What is Federated Learning?

- Problems with traditional ML
  - Data locality
  - Resource distribution
  - Privacy concerns
- Distributed data sources
  - Training at those sources
  - No raw data is communicated or shared
- Configurable aggregation
- Assists in security
- Use of distributed resources



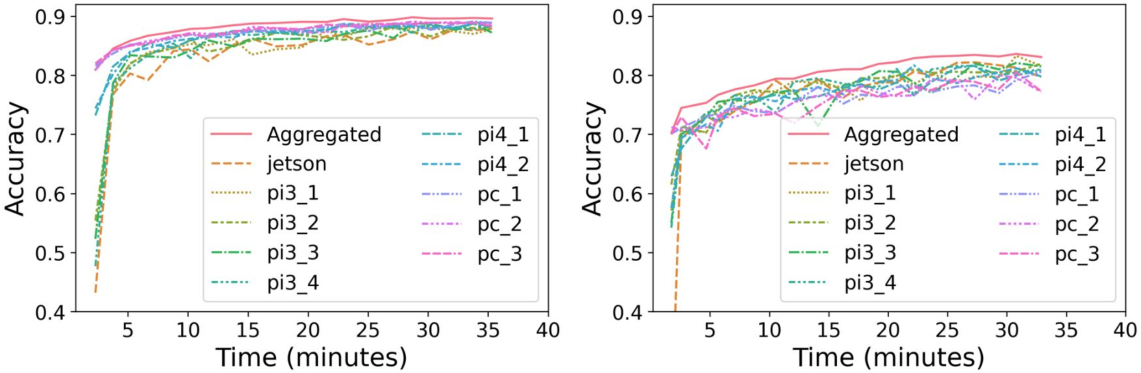
Why serverless is the answer…

- Portability and interoperability
  - Functions when and where they are needed
- Modularity
  - Register functions and replace function IDs as needed
- Fire-and-forget
  - We do not need constant contact between resources
  - Excellent for weak networks
- Needs to be easier than home-spun solution
  - Very simple FL is trivially easy
    - np.mean(list_of_weights)
  - Widespread adoption requires undercutting ease at every stage
- We have put together **FLoX–Federated Learning on funcX**

*final logo design pending

Workload balancing

- Serverless enables computing on many different devices
  - Many different devices are imbalance
- Remove compute bottlenecks by altering endpoint workloads
- Demonstrated effective FL while varying both samples and epochs



(a) Balancing samples (b) Balancing epochs

Autotuning: an ongoing effort

- Users shouldn't have to configure experiments either!
- Understanding workload balance and aggregation
  - Frequency of aggregation
    - Every epoch to once per experiment
  - Comparing workload balance methods
    - Balancing on epochs, samples, or both
    - Epochs seems like the parameter to sacrifice
      - Much more testing needed
- Currently testing on sensitivity to dropped endpoints

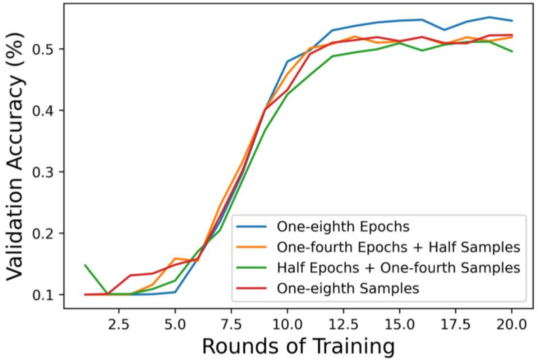
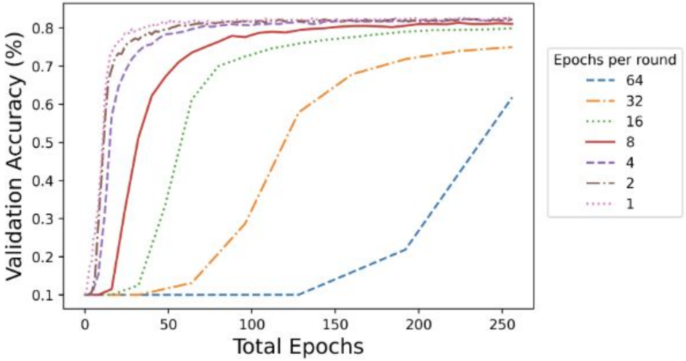
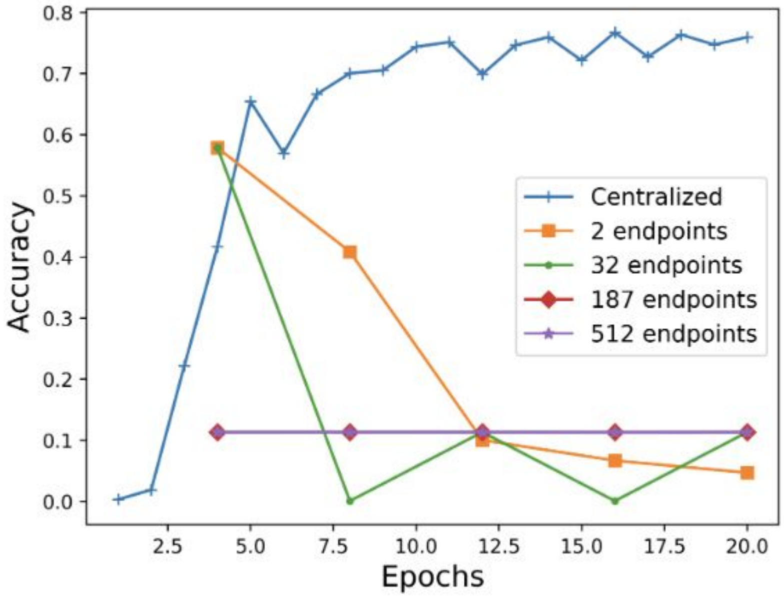


Figure 2: Comparison of balancing techniques to perform FL between two high powered endpoints and two additional endpoints with one-eighth the capabilities.

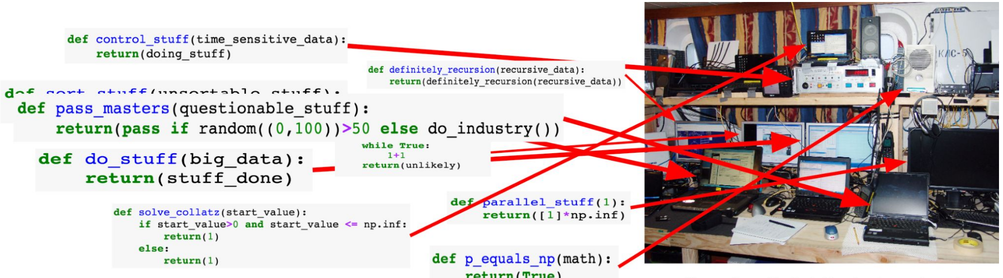


Autotuning: a practical use-case

- FL data is more prone to being sparse/ non-IID
  - More difficult to learn
- FL models are likely to be smaller and less able to learn complex features
- Result: extreme forgetting when learning sparse features across multiple endpoints
  - Anything learned is “averaged” away
- How to address this…
  - Must be automatic
    - See “ease of use” challenge
  - Algorithms
    - Tournament based pretraining
    - Advanced aggregation

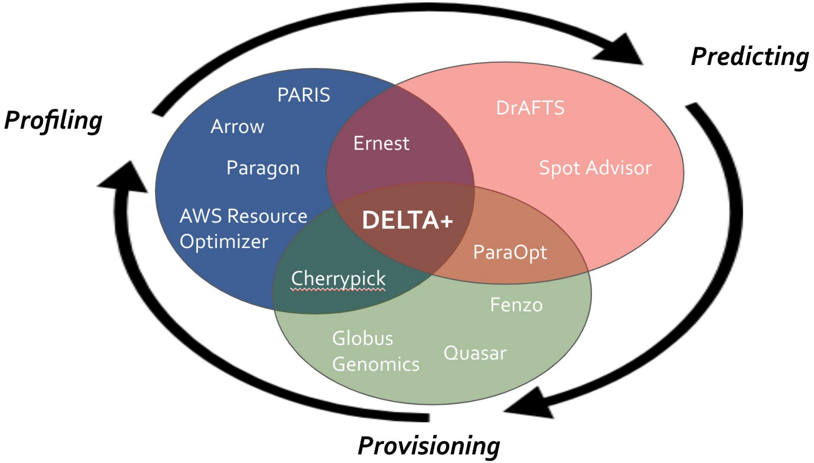


Let’s generalize…



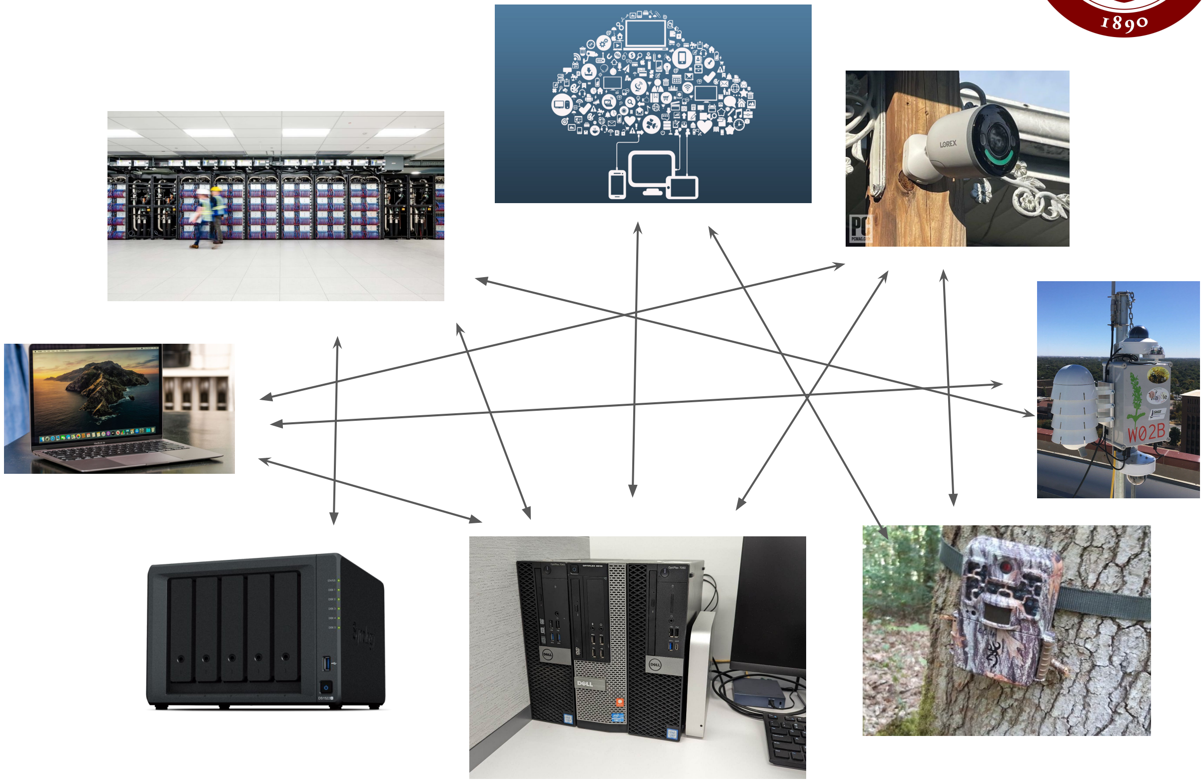
(“modern” distributed system)

- ❖ DELTA+
  - ➢ Automate placement of funcX/GC tasks across available endpoints
  - ➢ Work began ~2020
  - ➢ Minor improvements since
    - ■ Cloud provisioning
    - ■ ML-based placement
    - ■ Probabilistic scheduling
    - ■ Complex cost usage
- ❖ Potential applications in FL?



The Big Idea – Automate Distributed ML/FL

- Data all over the place
  - More data all the time
- Many endpoints
- Hybrid structure
  - Distributed ML on HPC
  - Ad-hoc clustering for hierarchical aggregation
  - Global workload coordination/consolidation
- "Compute where the data lives"
- Perhaps more cost effective to move the data?
  - "Train or transfer?"
    - Train *and* transfer!



Future Work

- Hybrid/hierarchical FL
  - FL principles as a distributed ML paradigm
  - Distributed ML on a resource, FL on multiple resources
- Decentralized FL
  - Endpoints initiate training rounds – event-driven FL?
- DELTA-Learn
  - Big resource management issues
    - Profiling endpoints/workloads - embeddings for resource characteristics?
    - Placing tasks - graph ML?
  - Consolidated multi-modal ML – using different data from different endpoints?
    - Extraction questions, data management, knowledge discovery, etc
  - FL for FL – self-adaptive ML/FL system (learn by doing)

Questions?

mbaughman@uchicago.edu