Distributed Machine Learning Systems

DATA INTEGRATION AND LARGE SCALE ANALYSIS — WS 2025/26

DAVID WEISSTEINER, LUCAS IACONO

16/01/2026

Bundesministerium Innovation, Mobilität und Infrastruktur
Bundesministerium Wirtschaft, Energie und Tourismus
Das Land Steiermark
FFG
Stadt Wien
Tirol
wirtschaftsagentur wien
Für die Stadt Wien
SFG

I am going to talk about Distributed Machine Learning Systems

What is Distributed Machine Learning?

What are Distributed Machine Learning Systems?

Who is ‘I’?

> whoami

David Weissteiner

Master Student @ Graz University of Technology
Junior Researcher @ Know Center Research GmbH

Interests: Distributed ML Systems, Federated Learning

Website: https://ywcb00.ywcb.org
Email: david@weissteiner.net
Phone: +39 349 655 5365

What is Distributed Machine Learning?

"[...] is an approach to large-scale ML tasks where workloads are spread across **multiple devices** or processors instead of running on a single computer." [1]

"[...] refers to **multinode** machine learning algorithms and systems that are designed to improve performance, increase accuracy, and scale to larger input data sizes." [2]

"[...] involves training machine learning models across **multiple devices or servers**, sharing the workload, and enhancing computational efficiency." [3]

[1] Belcic, I., Stryker, C. What Is Distributed Machine Learning? | IBM. https://www.ibm.com/think/topics/distributed-machine-learning

[2] Galakatos, A., Crotty, A., & Kraska, T. (2018). Distributed Machine Learning. Encyclopedia of Database Systems.

[3] IntellicoWorks. Distributed Machine Learning: Algorithms and Frameworks. https://intellicoworks.com/distributed-machine-learning/

Why Distributed ML?

- Increasingly large datasets
  - Terabytes/Petabytes of data records
- Increasingly large models
  - More layers, more parameters

Too slow to process on a single machine

Cannot fit on a single machine

Example GPT-3

- 175 billion parameters
- Trained on 45 Terabytes of text data

Adapted from Matthias Boehm, Data Integration and Large-scale Analysis WiSe2024/25, TU Berlin, https://mboehm7.github.io/teaching/ws2425_dia/13_MLSystems.pdf

What is a Distributed ML System?

- Simple answer: System to perform Distributed Machine Learning

- Better answer?
  - One step back: ML System

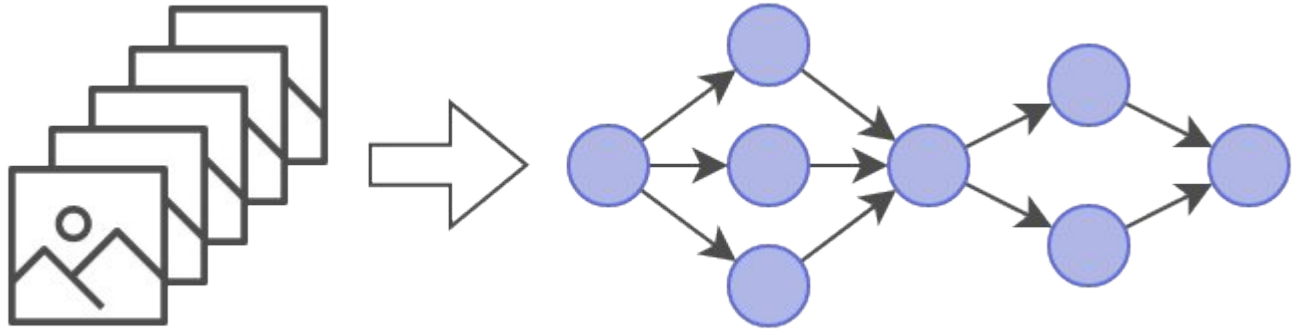
Recap: ML System

We have:

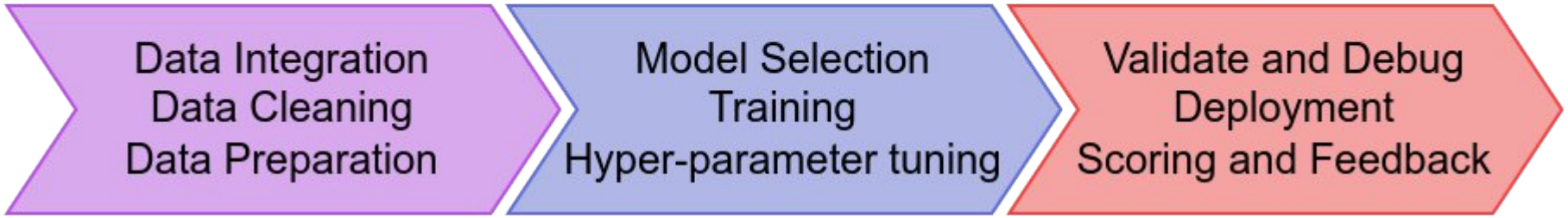
- Training Data
- ML Model

Task:

- Train model on data



Data Science Lifecycle



Distributed ML System

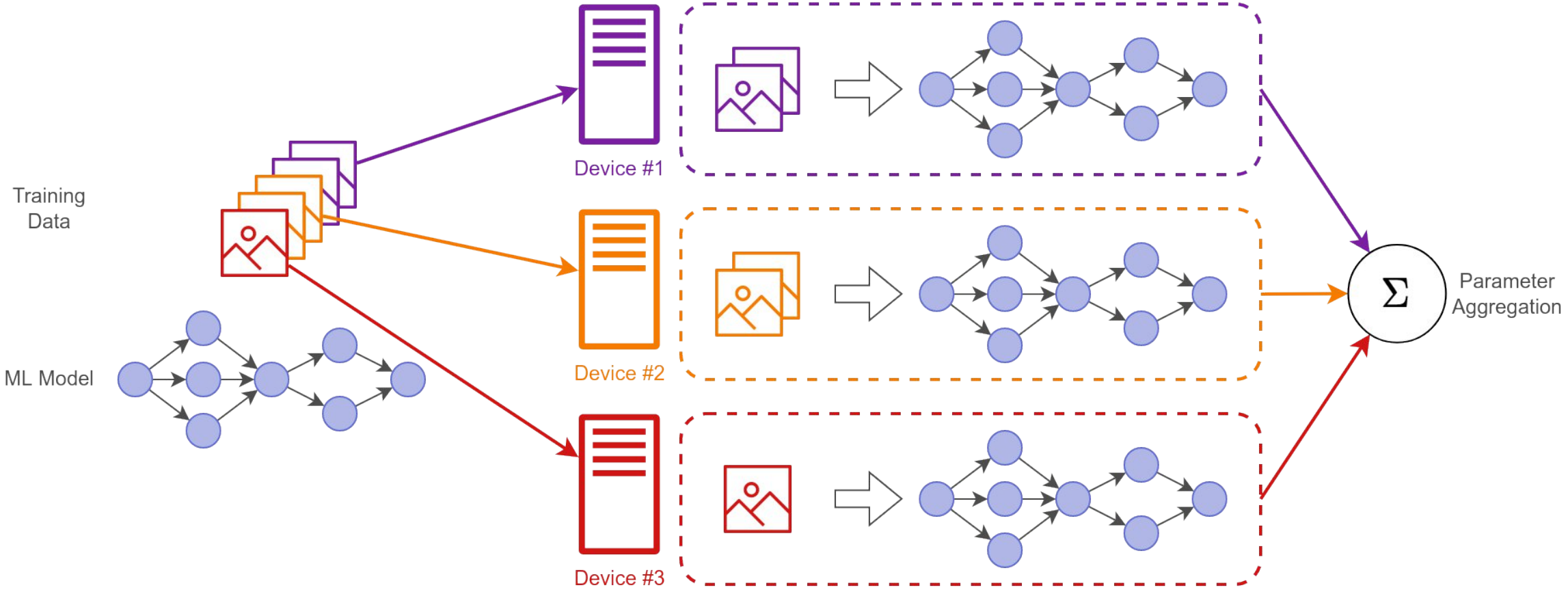
Multiple devices collaborate on a single task

Types of Distributed Machine Learning [1]:

- **Data** parallelism
  - Splitting the dataset across multiple devices
- **Model** parallelism
  - Partitioning a model across multiple devices
- **Hybrid** parallelism
  - Combining aspects of data and model parallelism

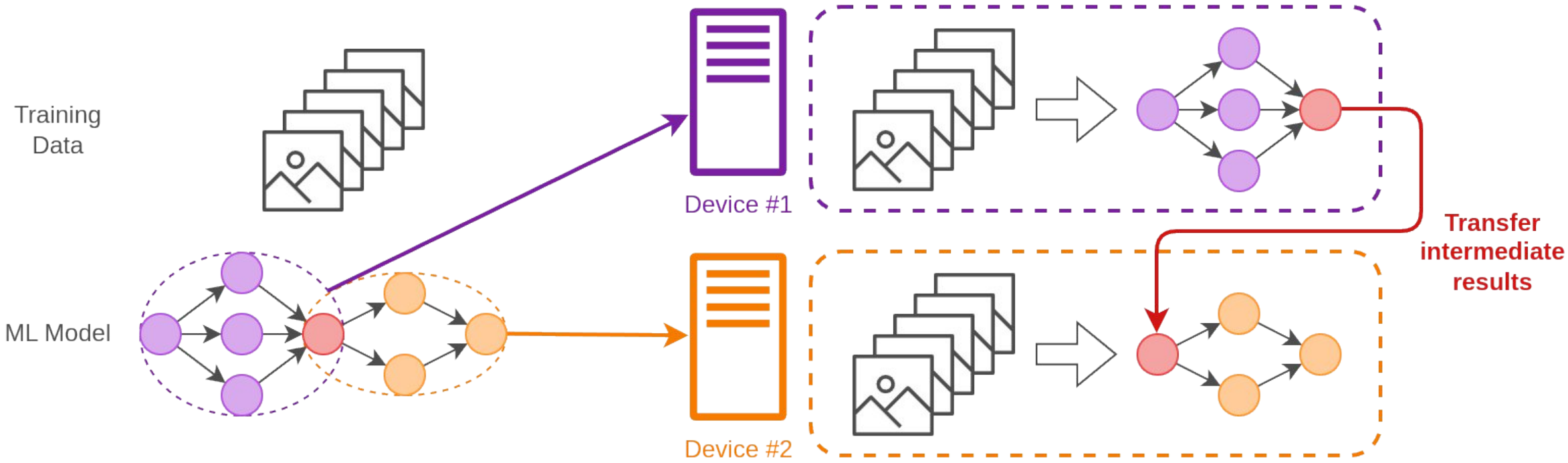
[1] Belcic, I., Stryker, C. What Is Distributed Machine Learning? | IBM. https://www.ibm.com/think/topics/distributed-machine-learning

Distributed ML System — Data Parallelism



Adapted from Mike Freedman, Distributed Systems, Princeton University, https://www.cs.princeton.edu/courses/archive/spr22/cos418/docs/L22-ml.pdf

Distributed ML System — Model Parallelism

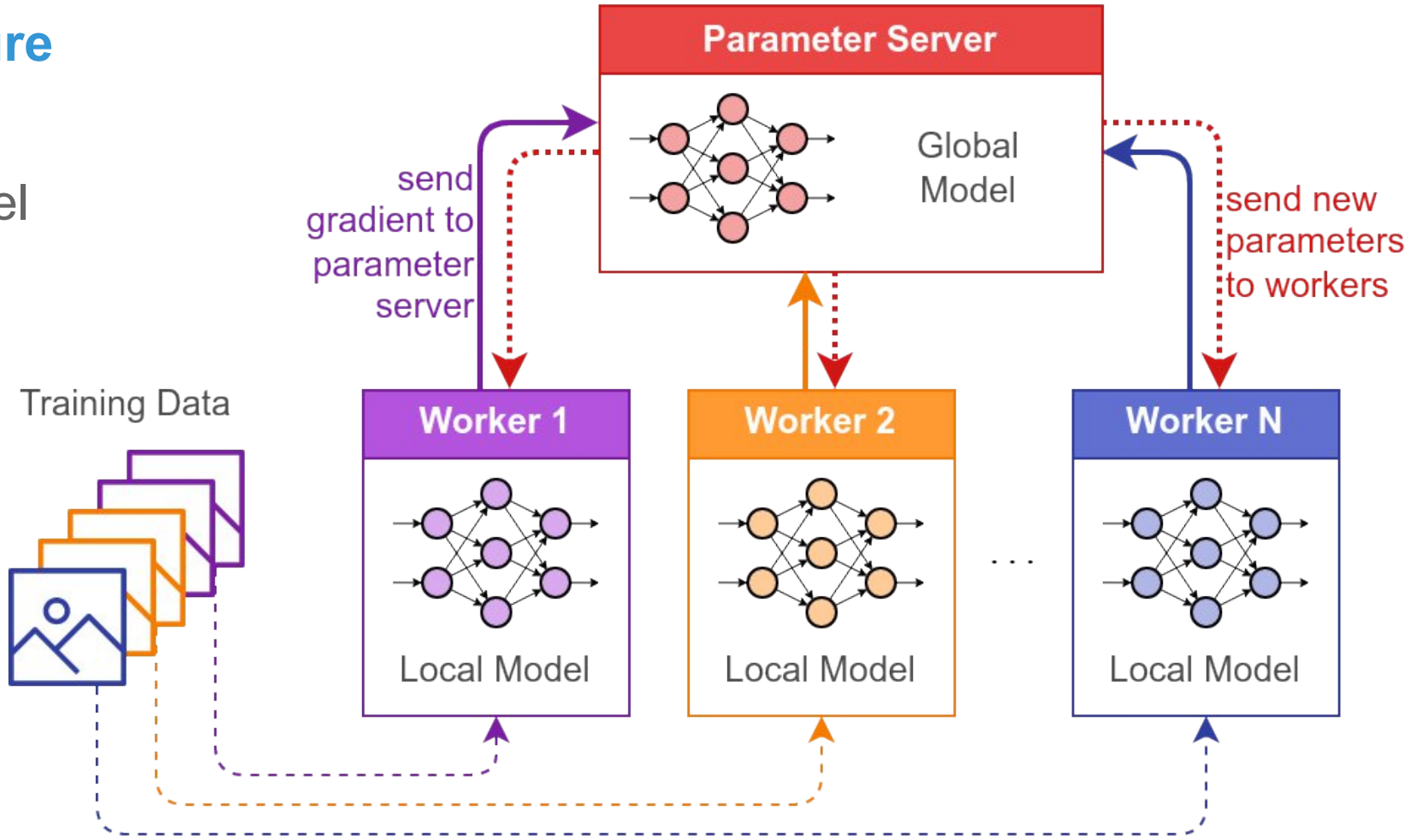


Adapted from Mike Freedman, Distributed Systems, Princeton University, https://www.cs.princeton.edu/courses/archive/spr22/cos418/docs/L22-ml.pdf

The Parameter Server

Architecture

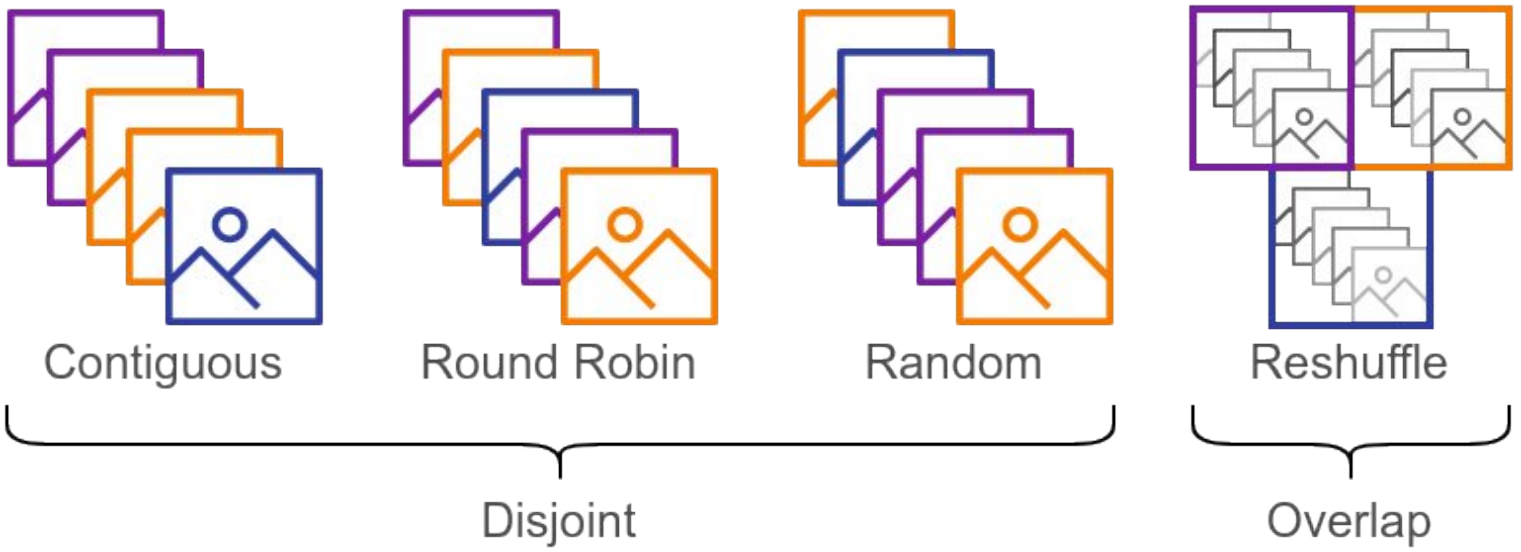
Data-parallel Parameter Server



Adapted from Christopher De Sa, Principles of Large-Scale Machine Learning Systems — Spring 2019, Cornell University, https://www.cs.cornell.edu/courses/cs4787/2019sp/notes/lecture22.pdf

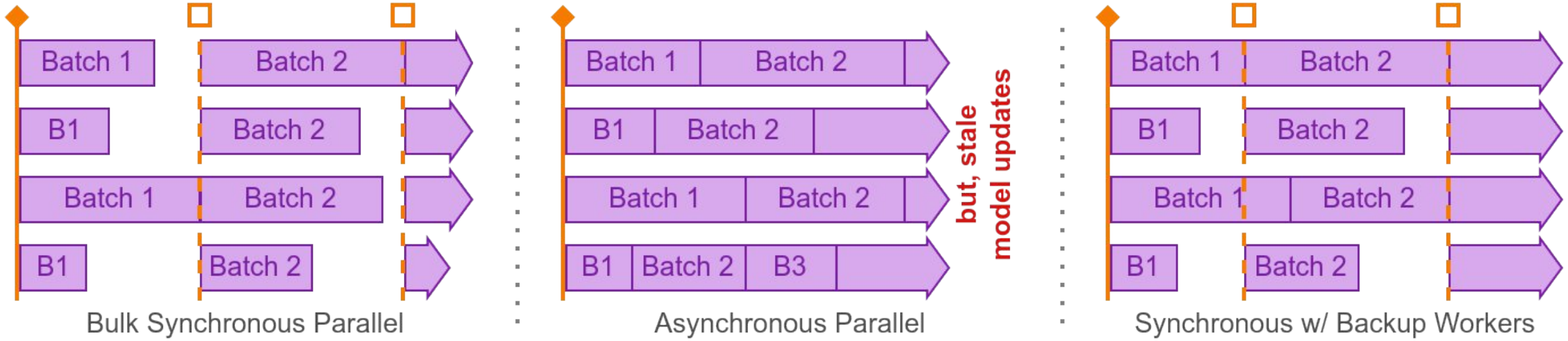
Key Techniques — Data Partitioning

- Disjoint Contiguous, Round Robin, Random
- Overlap Reshuffling
  - Each worker gets a full copy of the dataset but reshuffled



Adapted from Matthias Boehm, Architecture of ML Systems Summer 2025, TU Berlin, https://mboehm7.github.io/teaching/ss25_amls/06_ParameterServers.pdf

Key Techniques — Update Strategies

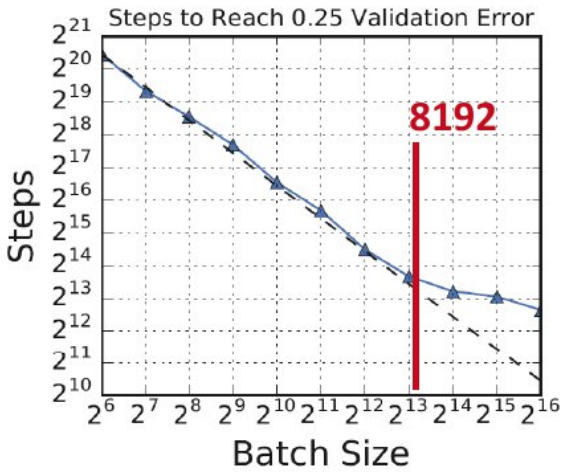


Adapted from Matthias Boehm, Architecture of ML Systems Summer 2025, TU Berlin, https://mboehm7.github.io/teaching/ss25_amls/06_ParameterServers.pdf

Key Techniques contd.

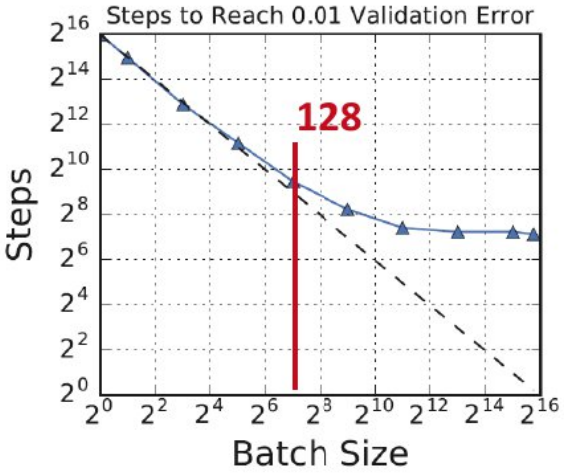
- Batch size strategies
  - Fixed batch size
    - Optimal batch size?
    - Dependent on data redundancy and model complexity
  - Dynamic methods
    - ❖ Increase the batch size instead of decaying the learning rate
    - ❖ Combine batch and mini-batch algorithms (full batch + n online updates)

ResNet-50 on ImageNet



VS

Simple CNN on MNIST



[Samuel L. Smith, Pieter-Jan Kindermans, Chris Ying, Quoc V. Le: Don't Decay the Learning Rate, Increase the Batch Size. **ICLR 2018**]

[Ashok Cutkosky, Róbert Busa-Fekete: Distributed Stochastic Optimization via Adaptive SGD. **NeurIPS 2018**]

Adapted from Matthias Boehm, Architecture of ML Systems Summer 2025, TU Berlin, https://mboehm7.github.io/teaching/ss25_amls/06_ParameterServers.pdf

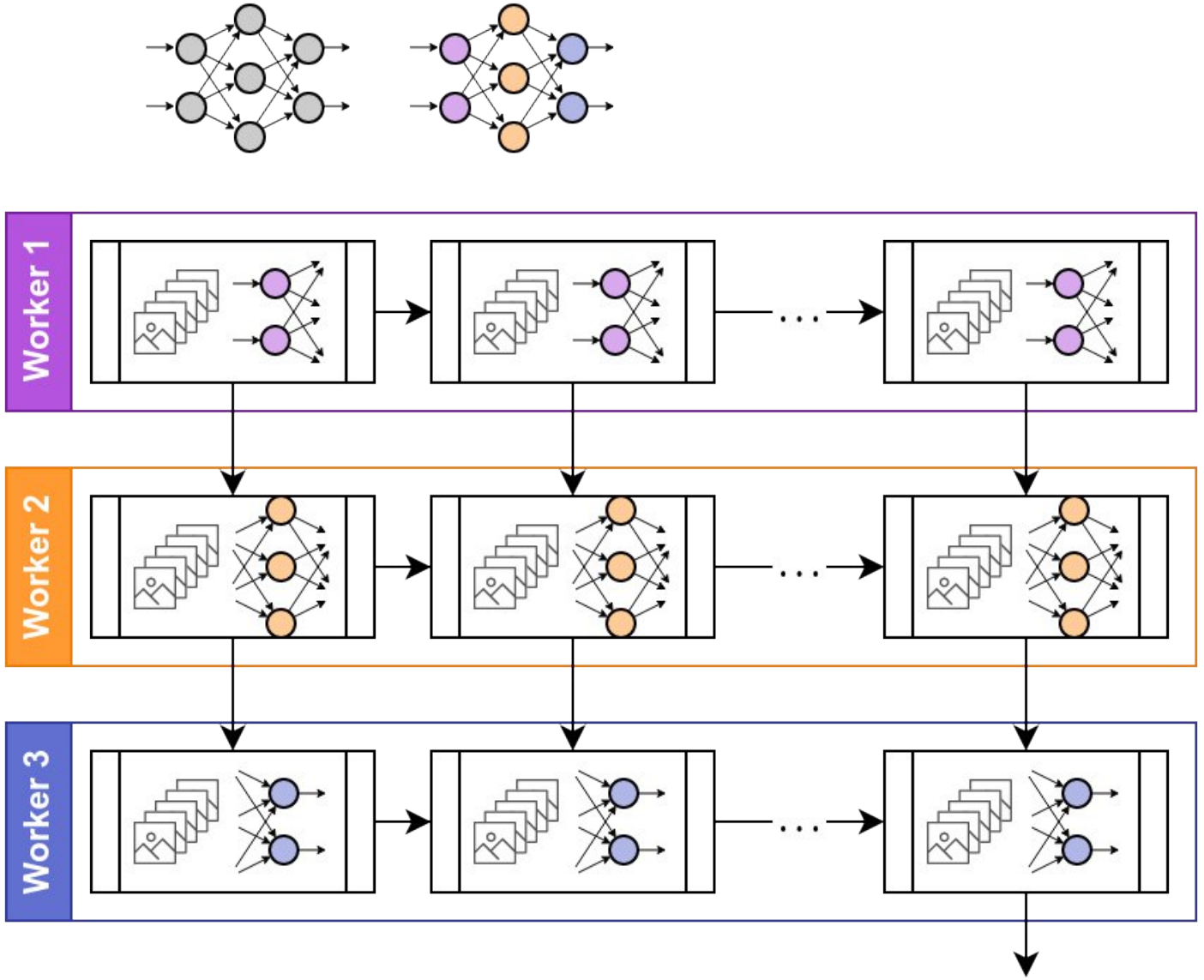
Types

Data-Parallel Parameter Server

- Data partitioned to the workers
- Problem: Need to fit entire model

Model-Parallel Parameter Server

- Nodes act as workers and parameter servers
- Workers responsible for disjoint partitions of model
- Pipeline Parallelism

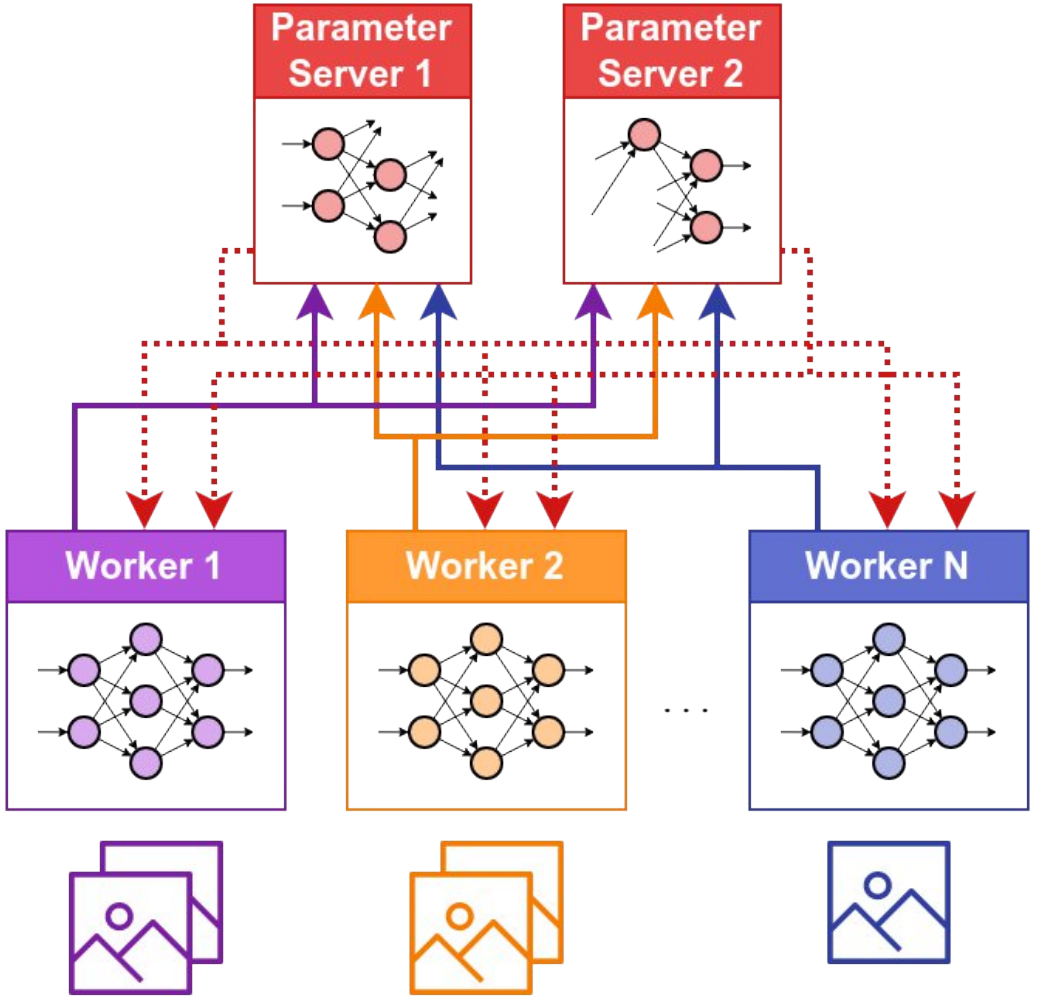


Adapted from Matthias Boehm, Architecture of ML Systems Summer 2025, TU Berlin, https://mboehm7.github.io/teaching/ss25_amls/06_ParameterServers.pdf

Distributed Parameter Servers

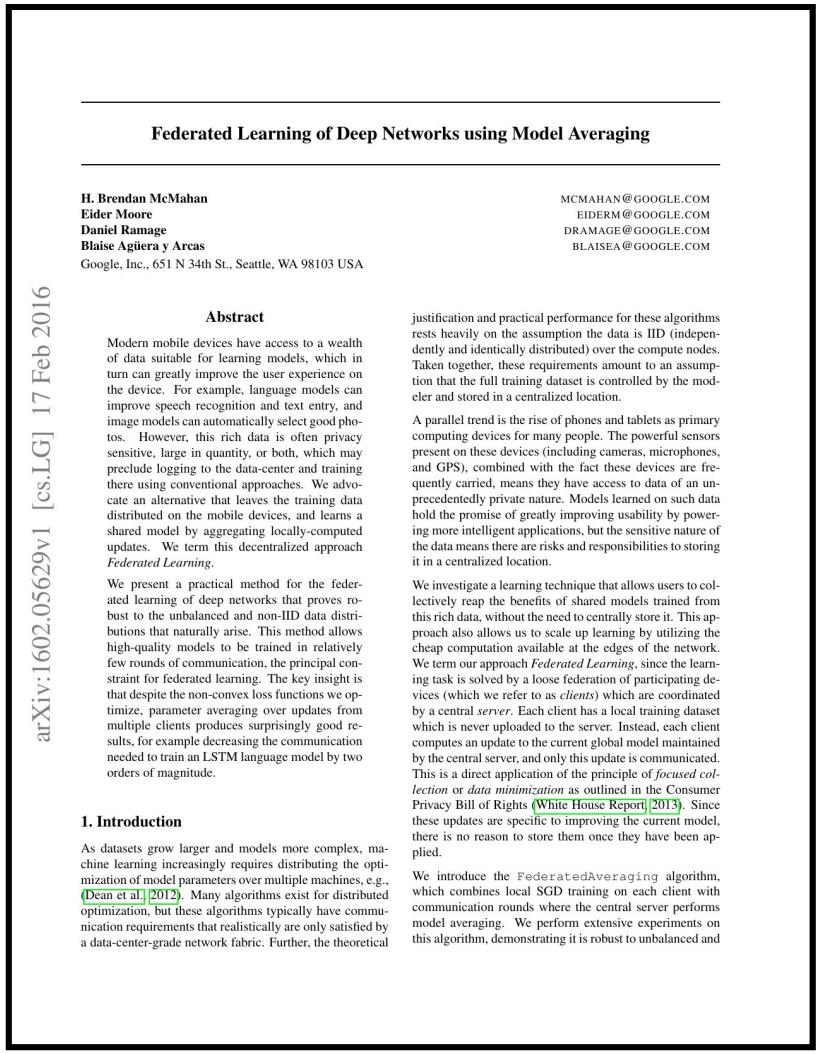
Distributed Parameter Servers

- Multiple parameter servers, multiple workers
- Multiple models or partitioned models



Adapted from Matthias Boehm, Data Integration and Large-scale Analysis WiSe2024/25, TU Berlin, https://mboehm7.github.io/teaching/ws2425_dia/13_MLSystems.pdf

Federated Learning



Federated Learning of Deep Networks using Model Averaging

H. Brendan McMahan MCMAHAN@GOOGLE.COM
Eider Moore EIDERM@GOOGLE.COM
Daniel Ramage DRAMAGE@GOOGLE.COM
Blaise Agüera y Arcas BLAISEA@GOOGLE.COM
Google, Inc., 651 N 34th St., Seattle, WA 98103 USA

Abstract

Modern mobile devices have access to a wealth of data suitable for learning models, which in turn can greatly improve the user experience on the device. For example, language models can improve speech recognition and text entry, and image models can automatically select good photos. However, this rich data is often privacy sensitive, large in quantity, or both, which may preclude logging to the data-center and training there using conventional approaches. We advocate an alternative that leaves the training data distributed on the mobile devices, and learns a shared model by aggregating locally-computed updates. We term this decentralized approach *Federated Learning*.

We present a practical method for the federated learning of deep networks that proves robust to the unbalanced and non-IID data distributions that naturally arise. This method allows high-quality models to be trained in relatively few rounds of communication, the principal constraint for federated learning. The key insight is that despite the non-convex loss functions we optimize, parameter averaging over updates from multiple clients produces surprisingly good results, for example decreasing the communication needed to train an LSTM language model by two orders of magnitude.

1. Introduction

As datasets grow larger and models more complex, machine learning increasingly requires distributing the optimization of model parameters over multiple machines, e.g., (Dean et al., 2012). Many algorithms exist for distributed optimization, but these algorithms typically have communication requirements that realistically are only satisfied by a data-center-grade network fabric. Further, the theoretical justification and practical performance for these algorithms rests heavily on the assumption the data is IID (independently and identically distributed) over the compute nodes. Taken together, these requirements amount to an assumption that the full training dataset is controlled by the modeler and stored in a centralized location.

A parallel trend is the rise of phones and tablets as primary computing devices for many people. The powerful sensors present on these devices (including cameras, microphones, and GPS), combined with the fact these devices are frequently carried, means they have access to data of an unprecedentedly private nature. Models learned on such data hold the promise of greatly improving usability by powering more intelligent applications, but the sensitive nature of the data means there are risks and responsibilities to storing it in a centralized location.

We investigate a learning technique that allows users to collectively reap the benefits of shared models trained from this rich data, without the need to centrally store it. This approach also allows us to scale up learning by utilizing the cheap computation available at the edges of the network. We term our approach *Federated Learning*, since the learning task is solved by a loose federation of participating devices (which we refer to as *clients*) which are coordinated by a central *server*. Each client has a local training dataset which is never uploaded to the server. Instead, each client computes an update to the current global model maintained by the central server, and only this update is communicated. This is a direct application of the principle of *focused collection* or *data minimization* as outlined in the Consumer Privacy Bill of Rights (White House Report, 2013). Since these updates are specific to improving the current model, there is no reason to store them once they have been applied.

We introduce the `FederatedAveraging` algorithm, which combines local SGD training on each client with communication rounds where the central server performs model averaging. We perform extensive experiments on this algorithm, demonstrating it is robust to unbalanced and

McMahan, H.B., Moore, E., Ramage, D., & Arcas, B.A. (2016). Federated Learning of Deep Networks using Model Averaging. ArXiv, abs/1602.05629.

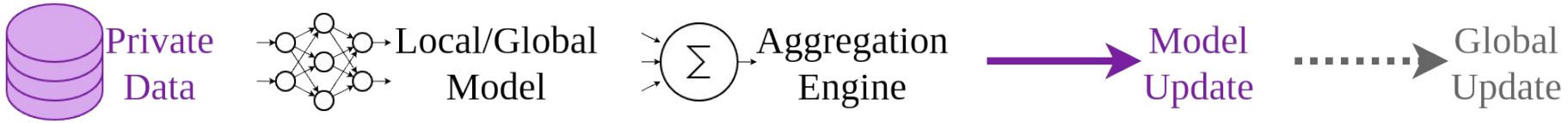
Motivation

- Train model on decentralized data from multiple client devices (federated workers)
- w/o consolidating the data
- w/ privacy constraints (?)

Example: Gboard on Android [1]

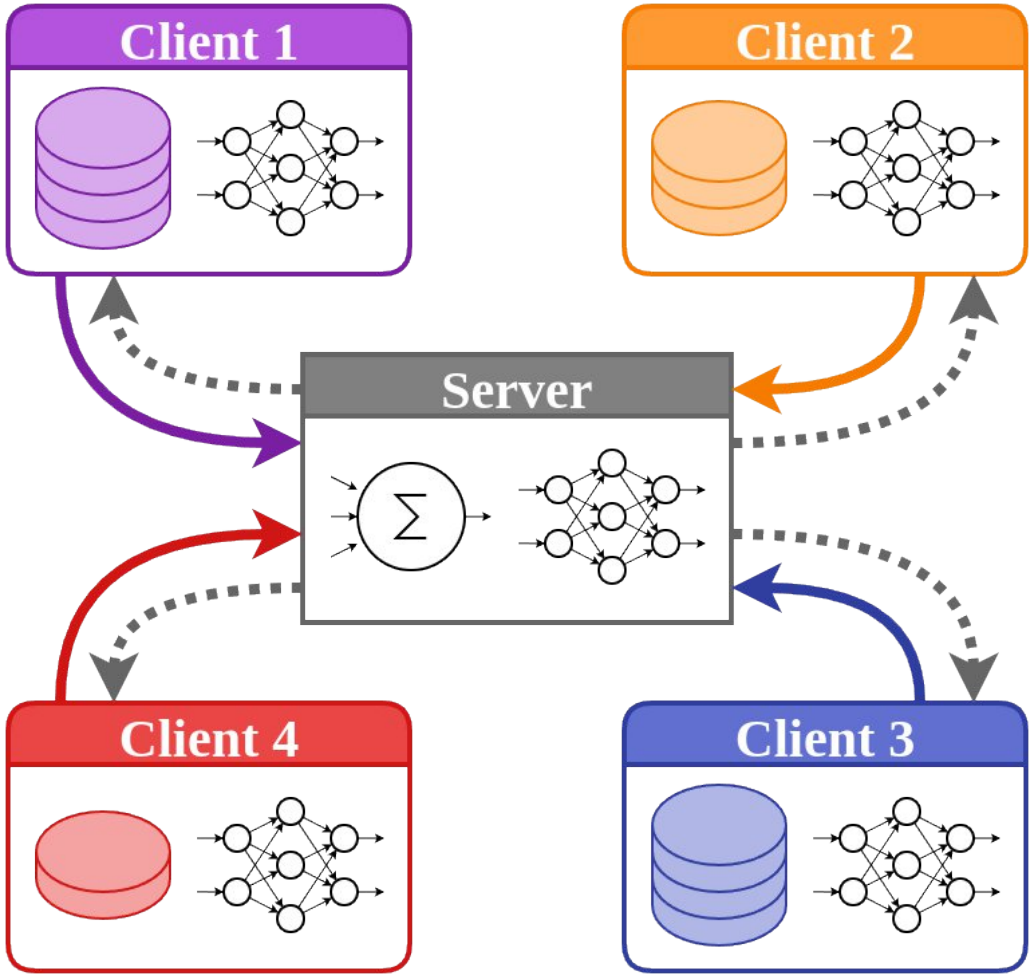
- Phone locally stores information about the current context and whether you clicked the suggestion
- Processes the data on-device to suggest improvements to the next iteration of Gboard's query suggestion model

[1] McMahan and Daniel Ramage, "Federated Learning: Collaborative Machine Learning without Centralized Training Data", April 2017, https://research.google/blog/federated-learning-collaborative-machine-learning-without-centralized-training-data/



Overview

- Architecture
  - Central Server / Coordinator
    - Aggregation Engine
    - Global Model
  - Client / Federated Worker
    - Local Model
    - Local Data
- Concept
  1. Server coordinates the workers
  2. Workers compute and send local updates
  3. Server aggregates updates
  4. Server broadcasts new model



Types

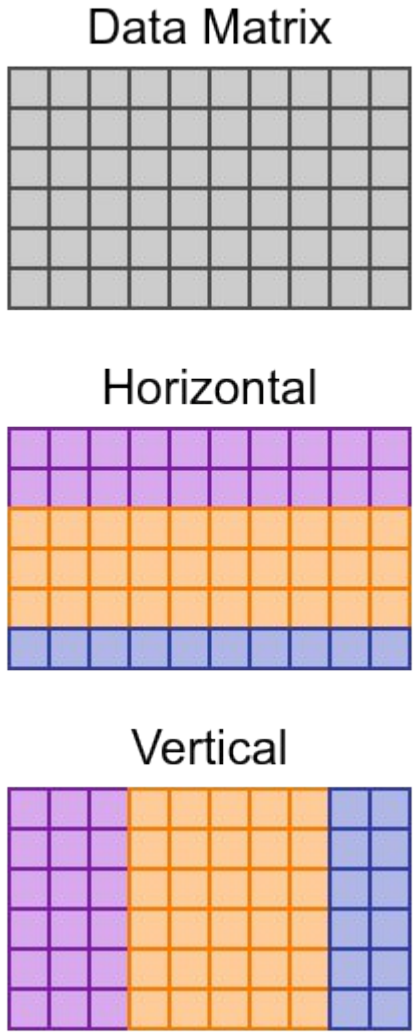
General

- **Centralized** Federated Learning (now)
- **Decentralized** Federated Learning (later)

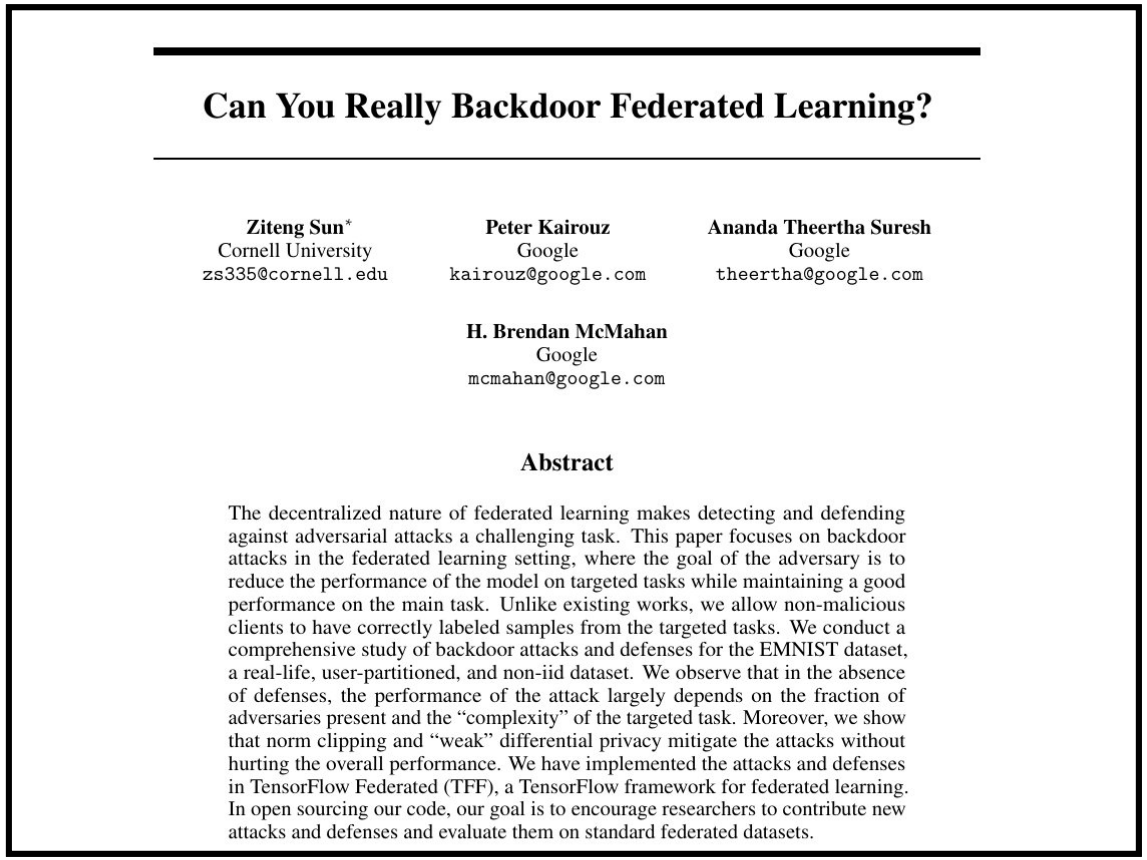
Data Partitioning

- **Horizontal** Federated Learning
  - I.I.D. Assumption
- **Vertical** Federated Learning

Federated **Transfer** Learning



Privacy?



Can You Really Backdoor Federated Learning?

Ziteng Sun*
Cornell University
zs335@cornell.edu

Peter Kairouz
Google
kairouz@google.com

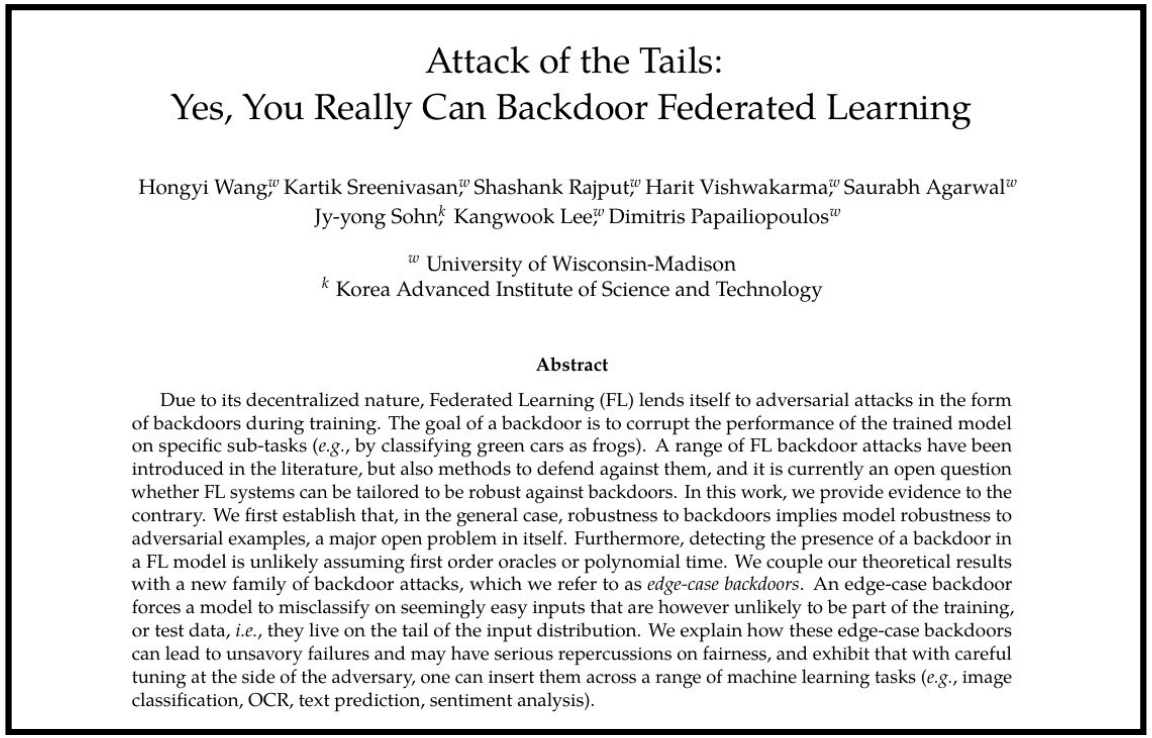
Ananda Theertha Suresh
Google
theertha@google.com

H. Brendan McMahan
Google
mcmahan@google.com

Abstract

The decentralized nature of federated learning makes detecting and defending against adversarial attacks a challenging task. This paper focuses on backdoor attacks in the federated learning setting, where the goal of the adversary is to reduce the performance of the model on targeted tasks while maintaining a good performance on the main task. Unlike existing works, we allow non-malicious clients to have correctly labeled samples from the targeted tasks. We conduct a comprehensive study of backdoor attacks and defenses for the EMNIST dataset, a real-life, user-partitioned, and non-iid dataset. We observe that in the absence of defenses, the performance of the attack largely depends on the fraction of adversaries present and the "complexity" of the targeted task. Moreover, we show that norm clipping and "weak" differential privacy mitigate the attacks without hurting the overall performance. We have implemented the attacks and defenses in TensorFlow Federated (TFF), a TensorFlow framework for federated learning. In open sourcing our code, our goal is to encourage researchers to contribute new attacks and defenses and evaluate them on standard federated datasets.

Sun, Z., Kairouz, P., Suresh, A.T., & McMahan, H.B. (2019). Can You Really Backdoor Federated Learning? ArXiv, abs/1911.07963.



Attack of the Tails: Yes, You Really Can Backdoor Federated Learning

Hongyi Wang[w] Kartik Sreenivasan[w] Shashank Rajput[w] Harit Vishwakarma[w] Saurabh Agarwal[w]
Jy-yong Sohn[k] Kangwook Lee[w] Dimitris Papailiopoulos[w]

[w] University of Wisconsin-Madison
[k] Korea Advanced Institute of Science and Technology

Abstract

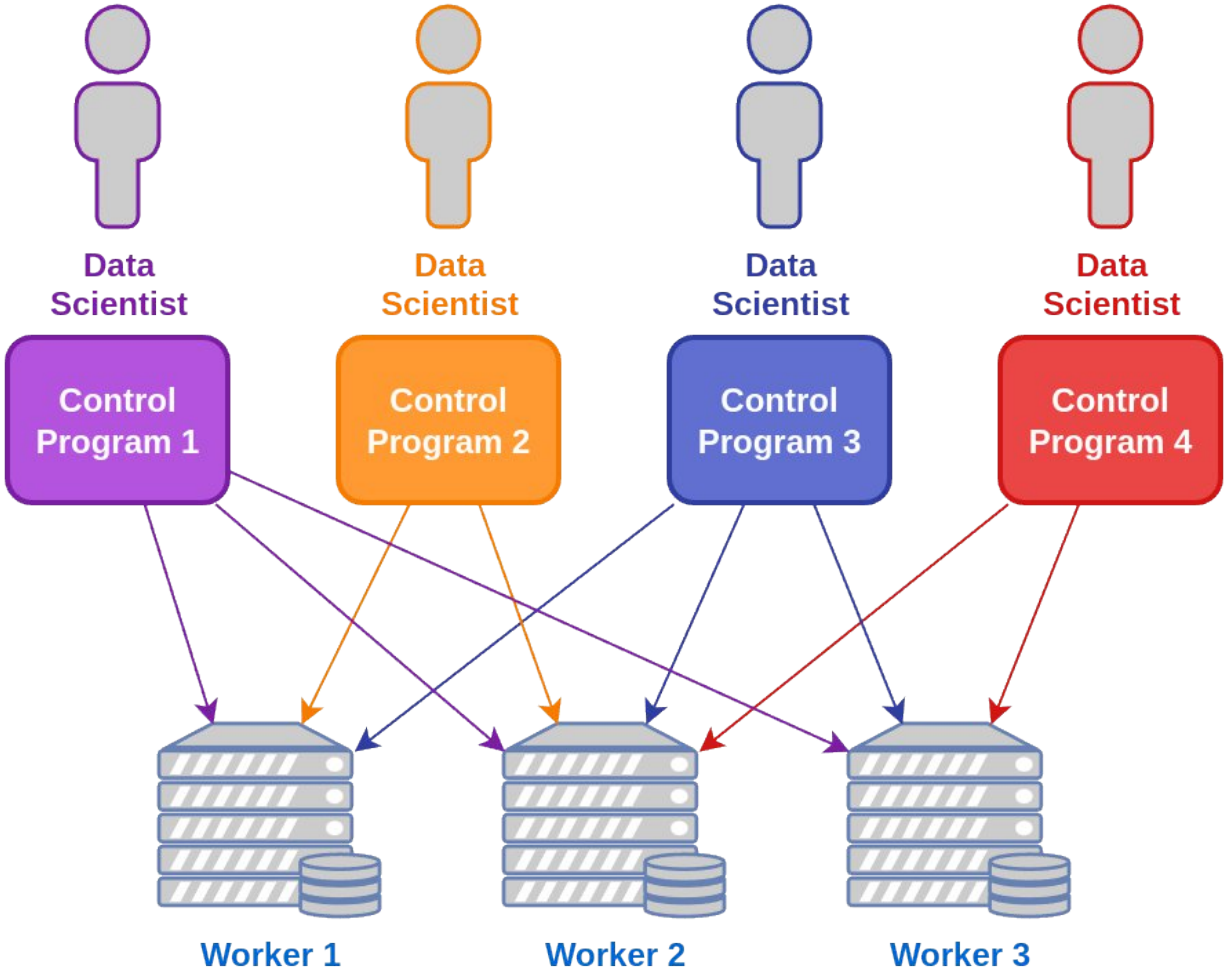
Due to its decentralized nature, Federated Learning (FL) lends itself to adversarial attacks in the form of backdoors during training. The goal of a backdoor is to corrupt the performance of the trained model on specific sub-tasks (*e.g.*, by classifying green cars as frogs). A range of FL backdoor attacks have been introduced in the literature, but also methods to defend against them, and it is currently an open question whether FL systems can be tailored to be robust against backdoors. In this work, we provide evidence to the contrary. We first establish that, in the general case, robustness to backdoors implies model robustness to adversarial examples, a major open problem in itself. Furthermore, detecting the presence of a backdoor in a FL model is unlikely assuming first order oracles or polynomial time. We couple our theoretical results with a new family of backdoor attacks, which we refer to as *edge-case backdoors*. An edge-case backdoor forces a model to misclassify on seemingly easy inputs that are however unlikely to be part of the training, or test data, *i.e.*, they live on the tail of the input distribution. We explain how these edge-case backdoors can lead to unsavory failures and may have serious repercussions on fairness, and exhibit that with careful tuning at the side of the adversary, one can insert them across a range of machine learning tasks (*e.g.*, image classification, OCR, text prediction, sentiment analysis).

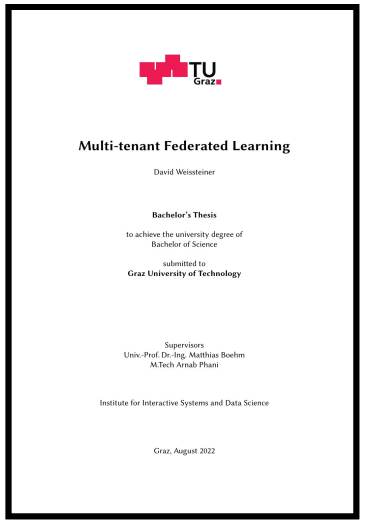
Wang, H., Sreenivasan, K.K., Rajput, S., Vishwakarma, H., Agarwal, S., Sohn, J., Lee, K., & Papailiopoulos, D. (2020). Attack of the Tails: Yes, You Really Can Backdoor Federated Learning. ArXiv, abs/2007.05084.

Challenges

- Security and Privacy
  - Model updates can still reveal sensitive information about the data
  - Possible solution: Differential privacy, homomorphic encryption
- Data Heterogeneity
  - Data partitions violate the I.I.D. assumption
  - Possible solutions: Data augmentation, client selection, regularization
- System Heterogeneity
  - Federated workers often differ in their computing, communication, and storage resources
  - Possible solutions: Asynchronous FL and semi-asynchronous FL
- Communication Bottleneck
  - All federated workers send their model to the central server
  - Possible solutions: Partial client participation, local updating, compression

Multi-tenant Federated Learning





Multi-tenant Federated Learning

David Weissteiner

Bachelor's Thesis

to achieve the university degree of
Bachelor of Science

submitted to
Graz University of Technology

Supervisors
Univ.-Prof. Dr.-Ing. Matthias Boehm
M.Tech Arnab Phani

Institute for Interactive Systems and Data Science

Graz, August 2022

Weissteiner, D. (2022), Multi-tenant Federated Learning, Bachelor's Thesis, Graz University of Technology

Federated Data Preparation, Learning, and Debugging in Apache SystemDS

Sebastian Baunsgaard, Matthias Boehm, Kevin Innerebner, Mito Kehayov, Florian Lackner, Olga Ovcharenko, Arnab Phani, Tobias Rieger, David Weissteiner, Sebastian Benjamin Wrede

ABSTRACT

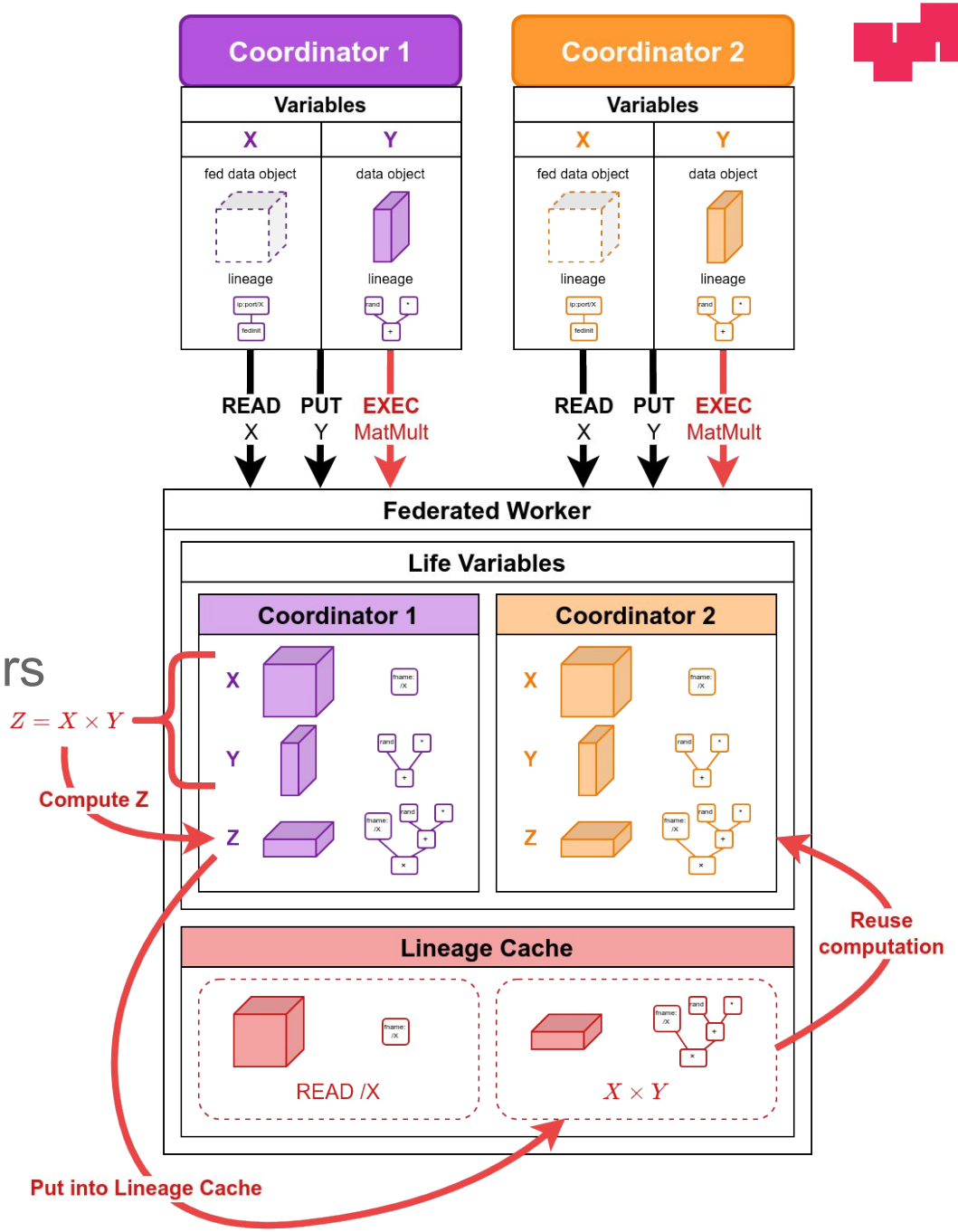
1 INTRODUCTION

Baunsgaard, S., et al. (2022). Federated Data Preparation, Learning, and Debugging in Apache SystemDS. International Conference on Information & Knowledge Management.

Multi-tenant Federated Learning contd.

Key Techniques

- READ, PUT, EXEC
- Federated unique coordinator identifier
- Isolated variable scopes at federated workers
- Optimization: READ reuse
- Optimization: Lineage-based reuse

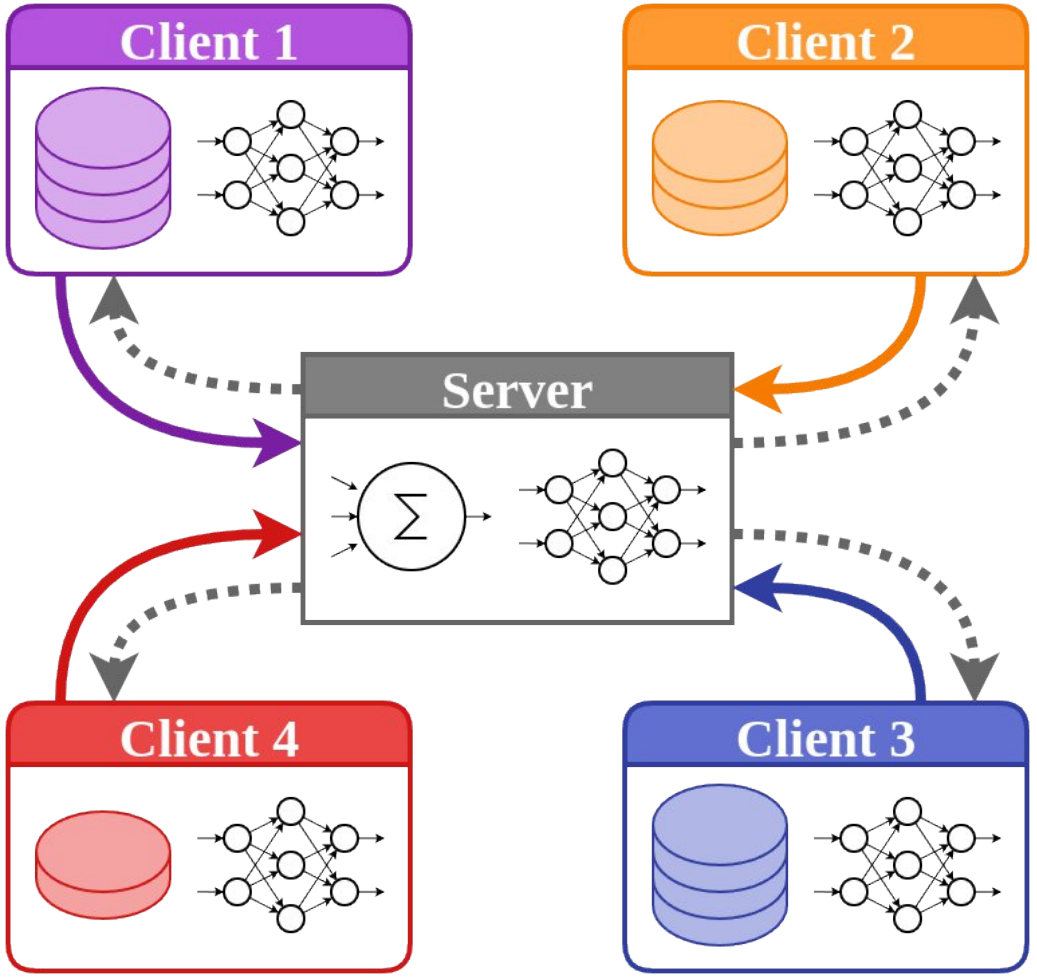


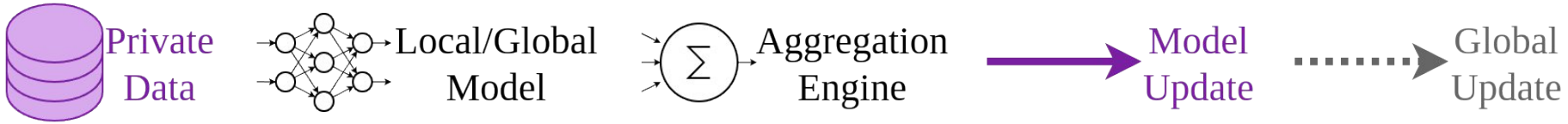
Decentralized Federated Learning

Why Decentralized Federated Learning?

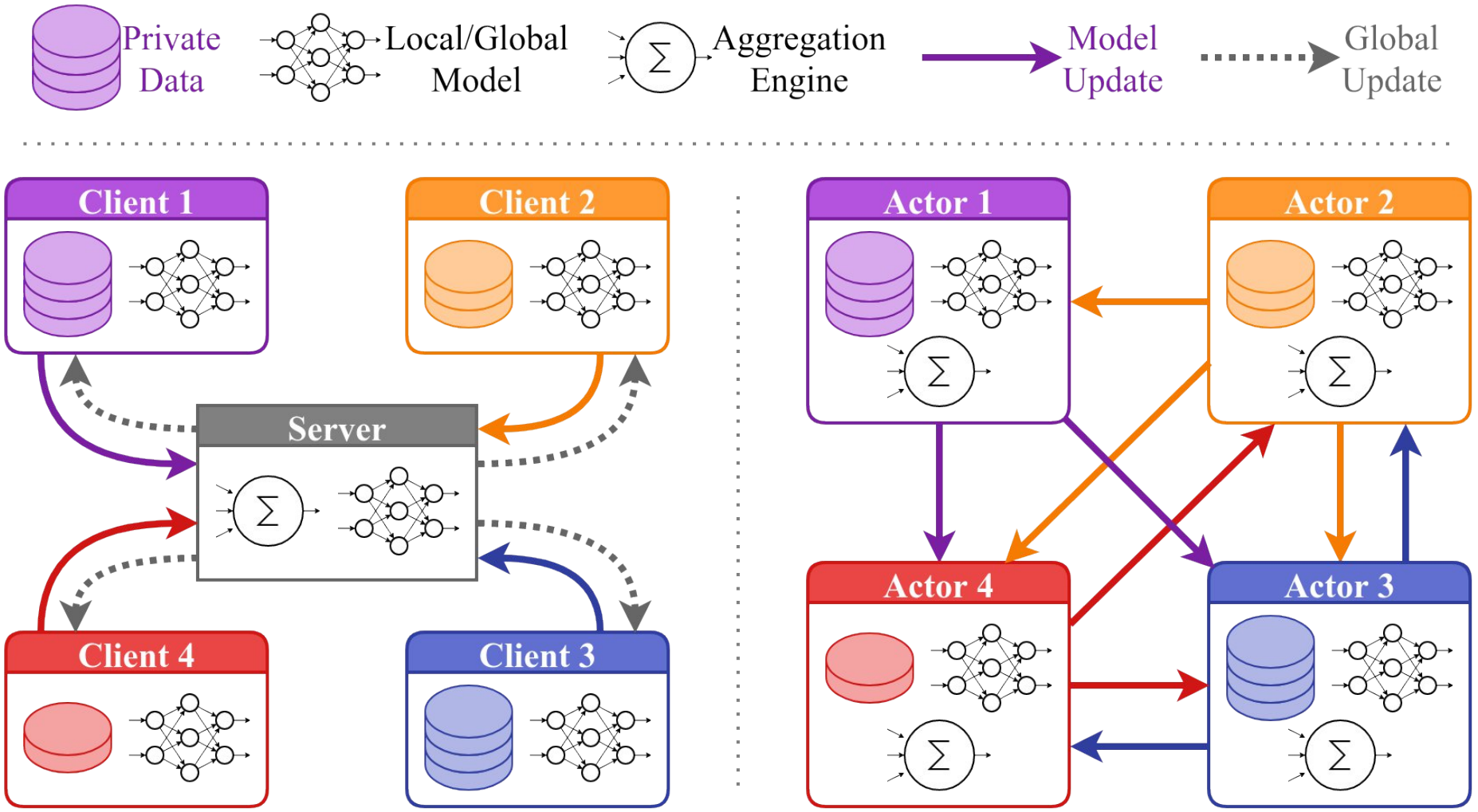
Problems w/ Centralized Federated Learning

- Communication bottleneck
- Single-point-of-failure
- Privacy and Security concerns





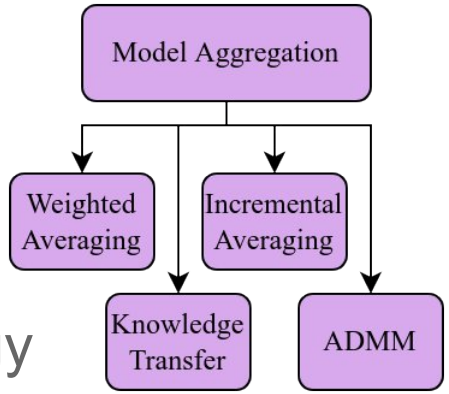
What is Decentralized Federated Learning?

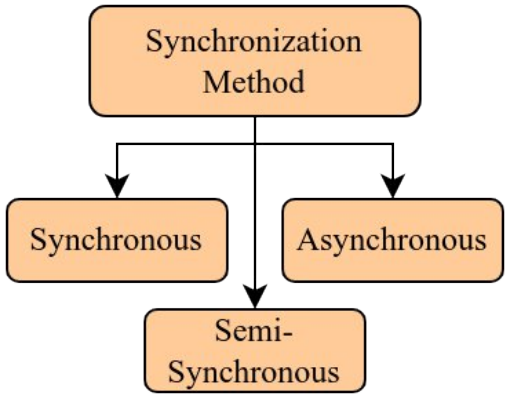


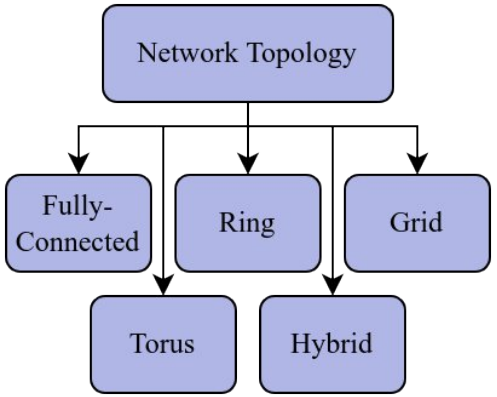
Key Techniques

Components

- Model Aggregation Strategy
- Synchronization Method
- Network Topology

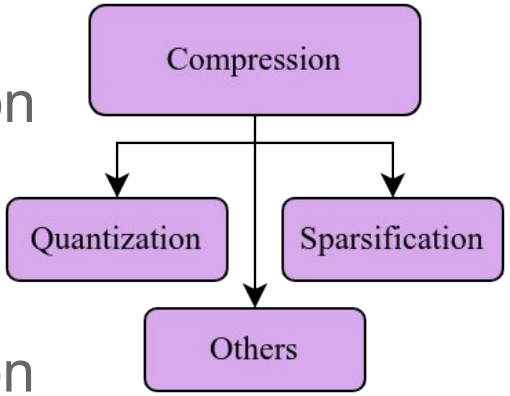


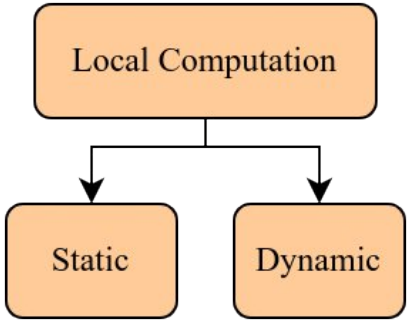


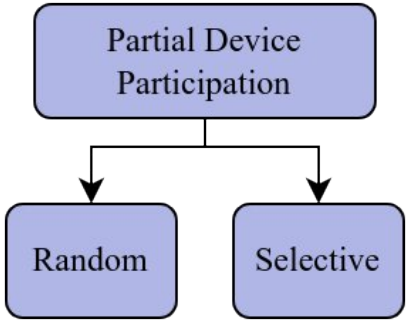


Communication Optimization

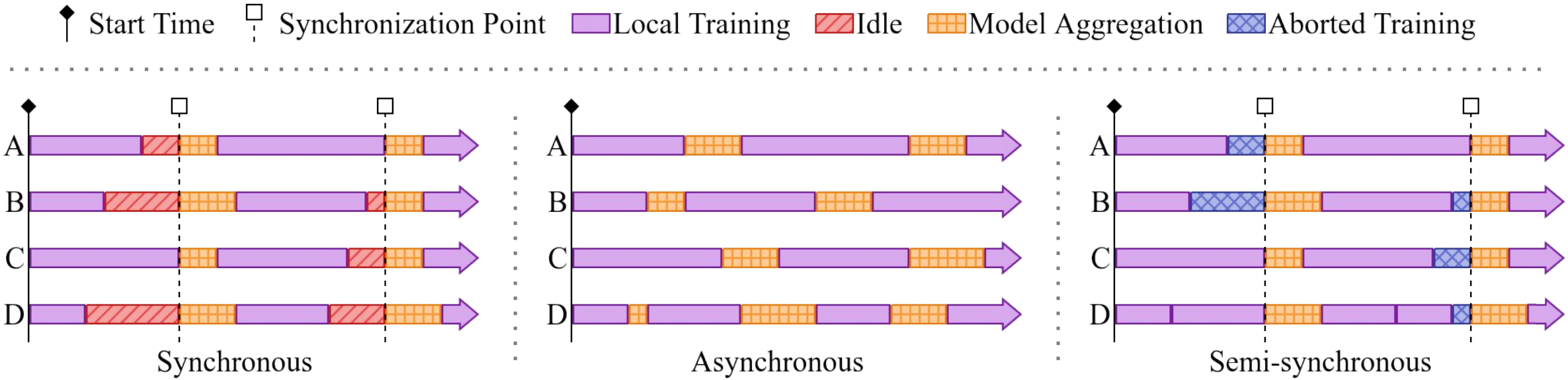
- Compression
- Local Computation
- Partial Device Participation



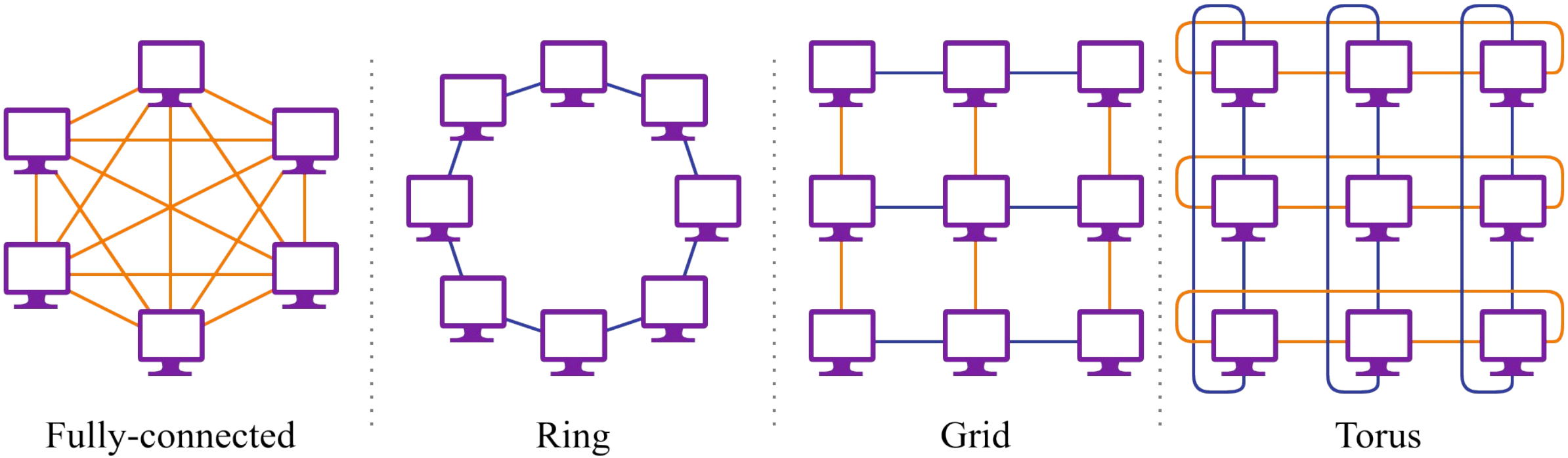




Key Components — Synchronization Method



Key Components — Network Topology



Fully-connected

N neighbors

Ring

2 neighbors

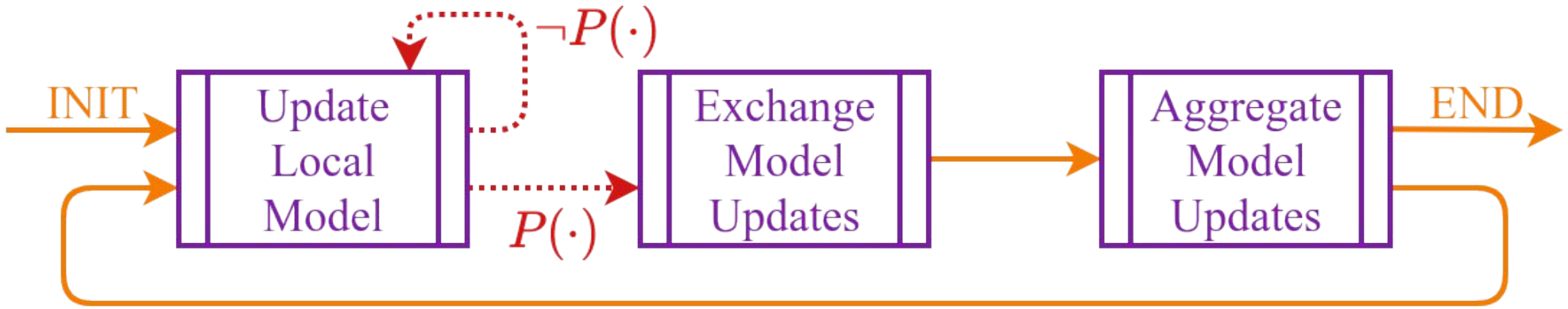
Grid

2—4 neighbors

Torus

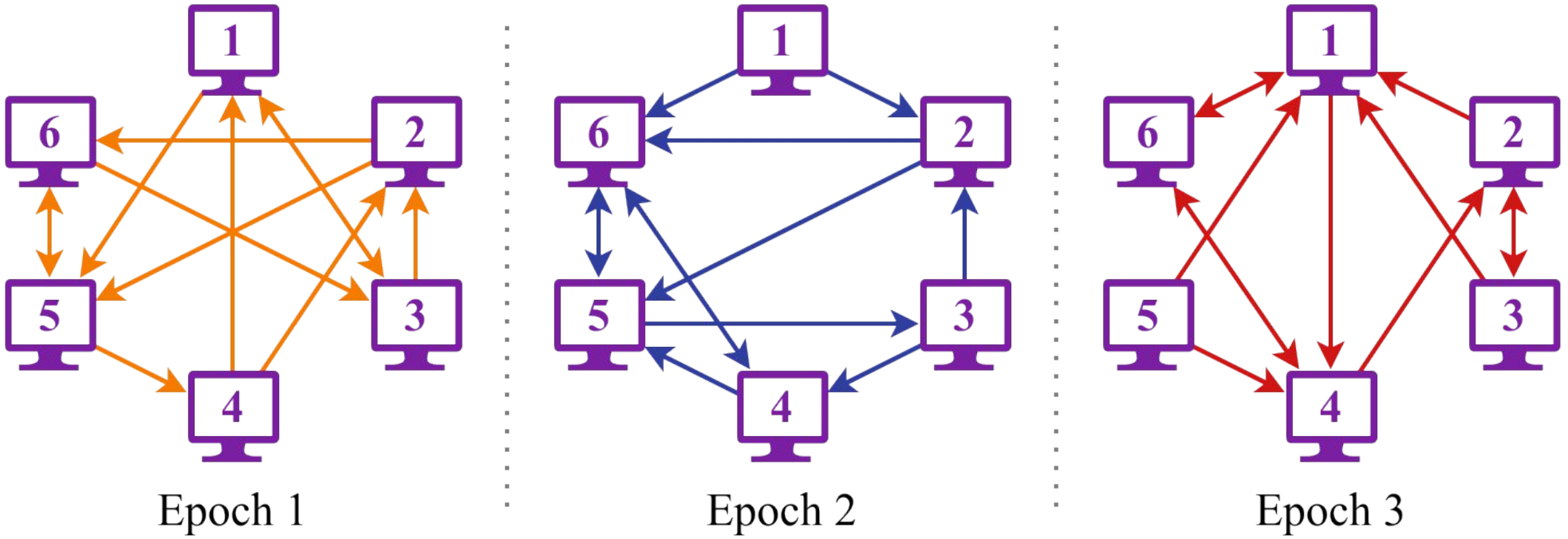
4 neighbors

Key Communication Optimizations — Local Computation



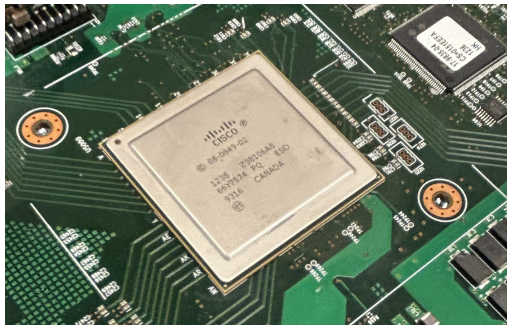
- Predicate function P
- Static vs. Dynamic

Key Communication Optimizations — Partial Device Participation



Hardware Accelerators for ML

Accelerators (GPUs, FPGAs, ASICs)



A packet processing ASIC inside an Ethernet switch

Memory-intensive vs. Compute-intensive

- CPU: dense/sparse, large mem, high mem-bandwidth, moderate compute
- GPU: dense, small mem, slow PCI, **very high** bandwidth/compute

- Graphics Processing Units (GPUs)
  - Extensively used for deep learning training and scoring
- Field-Programmable Gate Arrays (FPGAs)
  - Customizable HW accelerators for pre-filtering, compression, deep learning
- Application-Specific Integrated Circuits (ASIC)
  - Hardware customized for a specific use

Adapted from Matthias Boehm, Data Integration and Large-scale Analysis WiSe2024/25, TU Berlin, https://mboehm7.github.io/teaching/ws2425_dia/13_MLSystems.pdf

Summary

- Distributed Machine Learning
- Distributed ML Systems
- Parameter Server
- Federated Learning
- Decentralized Federated Learning
- Hardware Accelerators for ML

Questions?

David Weissteiner

Website: https://ywcb00.ywcb.org

Email: david@weissteiner.net

Phone: +39 349 655 5365