# DecentraLearn: Robust Learning in Decentralized Environments using Best-Effort Aggregation

Martijn de Vos
EPFL, Switzerland

Rishi Sharma
EPFL, Switzerland

Anne-Marie Kermarrec
EPFL, Switzerland

## Abstract

**Motivation.** Federated Learning (FL) has emerged as a popular approach for training machine learning models. With FL, model updates are communicated between user devices and a central server while personal data remains on users' devices [2]. Performing FL at scale involves deploying a centralized server that, amongst other tasks, selects participants during a round of training, aggregates model updates, and tracks the capabilities and availability of devices [1].

We identify three challenges associated with centralized coordination in FL setups: (1) a central server poses a single point of failure since it is a critical component to ensure training progression; any downtime can lead to significant disruptions in the training process, (2) the infrastructure and maintenance costs associated with a centralized architecture can be substantial, rendering the deployment of FL infrastructure at scale prohibitive for organizations with constrained budgets, and (3) the need to obtain approval from a particular organization to run learning tasks at scale opens the door to censorship, as this organization can arbitrarily deny model engineers from deploying their models.

We argue for a decentralized learning approach that avoids critical dependency on a central server. Such an approach is highly desirable as it addresses censorship concerns while removing a single point of failure. At the same time, we acknowledge that a central aggregation operation can speed up model convergence compared to existing decentralized learning approaches and reduce the communication cost compared to an all-to-all communication pattern. In summary, we seek a learning architecture where *system progression does not critically depend on the availability of a single server* but still benefits from the services provided by such a server if it is available.

**Approach.** We design a robust and novel learning architecture named DecentraLearn that avoids critical dependencies on a single, central aggregation service. DecentraLearn has three types of actors: model engineers, end-user devices, and aggregator services. Model engineers can prepare learning tasks and make them available to end users.[1] End-users download a mobile application to participate in DecentraLearn and can select the learning tasks they want to participate in, similar to task selection in volunteer computing. Aggregator services are responsible for aggregating the updated models in individual learning tasks.

End-user devices participating in DecentraLearn join a *meta-overlay* for the decentralized discovery of learning tasks, where available learning tasks are continuously gossiped amongst peers. Upon joining a particular learning task, users join a *sub-overlay* where they communicate exclusively with other participants working on the same learning task. This technique partitions the network into isolated sub-overlays for each learning task to avoid all-to-all communication and reduce network usage. Users can join or leave these overlays subject to power and network availability.

Within each sub-overlay, peers train a local model by requesting a model from an aggregator service for that learning task, if available, or by requesting a model from another peer otherwise. A peer then locally updates the model using its private training data and either sends the updated model back to the aggregator service, if available, or to a random peer otherwise. We call this mechanism *best-effort aggregation* as central aggregation only happens when a server is available. This approach ensures that the network follows a more centralized learning approach and fallbacks to a random interaction pattern when the central server is unavailable or faulty. *In DecentraLearn, aggregators are only used to speed up model convergence and are not critical to ensure the progression of the training process.* Additionally, when the aggregator service recovers from a crash, the peers merge the model sent by the aggregators with their models using an exponentially decaying weighting scheme based on model age. Model engineers can set up in-house aggregators for the learning task if they have the resources to do so, use the aggregator service of other organizations, or rely solely on device-to-device communication. Through this approach, DecentraLearn provides an open infrastructure to publish and work on learning tasks, tolerates churn by peers, and also tolerates the unavailability of aggregator services.

**Evaluation.** We plan to evaluate our mechanism with realistic models using real-world traces with information on mobile device availability, CPU speed, and network capacities. Our evaluation will focus on model convergence speed under churn and server failures and quantifies the additional network and computation overhead compared to a centralized setup.

## References

[1] Bonawitz et al. 2019. Towards federated learning at scale: System design. *Proceedings of machine learning and systems* 1 (2019), 374–388.

[2] McMahan et al. 2017. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*.

[1] We assume that learning tasks are pre-approved, e.g., using an open peer-review process, to prevent abuse.