Asynchronous Federated Learning Strategy

Nitin Nikamanth Appiah Balaji

Introduction/Motivation:

Traditional centralized training requires the collection of data in a data center for training. This causes privacy and security concerns as most individuals will not be willing to expose their confidential information. The best solution for this is to train local models on the private data present on individual edge devices considering it to be a local optimization problem and aggregating it to generate a superior better performing model. This technique of training model on different secluded devices in a collaborative fashion is called Federated Learning.

Federated learning architecture can be constructed in three different ways: Synchronous, Asynchronous and Semi-Synchronous. In synchronous learning, the aggregation happens only after all devices have completed their training; in asynchronous learning, the aggregation step is done whenever a device sends the trained model and in semi-synchronous training, the aggregation happens after a portion of the devices complete their training. The synchronous federated learning approach which is commonly implemented is inefficient as all the devices must wait between each aggregation step if one of the devices takes a long time to complete training. This wastes the compute power of the edge devices and wastes a lot of time.

Problem Statement:

Construct an architecture for efficiently train models with asynchronous and semi-synchronous learning settings. Validate different aggregation strategies and design an efficient aggregator that helps to reach the optimal model weights as quickly as possible. Analyze the performance and time improvements of the three federated learning strategies with respect to the centralized training paradigm.

Approach:

A real-world like environment can be simulated considering different devices with unequal data samples taking unequal training time. This can be implemented by allocating randomized unequal, unique data subsample to each node from a large dataset. Simulate the presence of delay and use only one node’s model for aggregation for the asynchronous learning scenario. For semi-synchronous training, introduce a hyperparameter K to wait for the first K devices to send their trained model and then apply the aggregation function. Implement different aggregators for each of the three federated learning scenarios and compare the performance.

The basic task of image classification can be taken for experimentation. The standard datasets such as CIFAR100 and ImageNet can be used for benchmarking. Performance trends of popular CNN architectures such as AlexNet, VGG, ResNet, ResNeXt can be analyzed for the three federated learning implementations as well as a centralized training scenario.

Expected Outcome:

An improvement in training time is expected in the case of asynchronous and semi-synchronous federated learning strategies as compared to synchronous. We expect a slight performance degradation in the case of asynchronous training case, but a logical selection of aggregation function and optimization strategies will give on par results with centralized training. As federated learning implicitly does data parallelism, a drastic reduction in training time and the time taken is expected in this order: Asynchronous < Semi-synchronous < Synchronous < Centralized training.