Federated Learning with Unbiased Gradient Aggregation and FedMeta

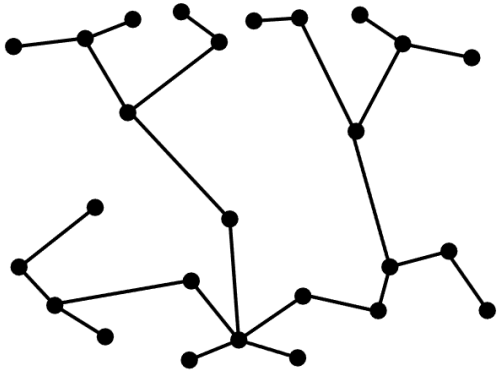
Decentralized learning

The group of nodes is not coordinated by any centralized server. Each node locally holds f_i and exchanges information only with its immediate neighbors.

$$\min_{x_1,\ldots,x_m\in\mathbb{R}^d} \sum_{i=1}^{m} f_i(x_i)$$
$$\text{s.t. } x_1 = \ldots = x_m.$$

The optimal point in the decentralized sense should be consensual and optimal, i.e.

$$x_1 = \ldots = x_m = x^* = \arg\min_{x\in\mathbb{R}^d} \frac{1}{m}\sum_{i=1}^{m} f_i(x).$$



Problem statement

The article focuses on two key issues in federated learning:

1. **Gradient Biases**: The multiple steps of local model updating on edge devices can lead to gradient biases, affecting the quality of the aggregated global model.
2. **Inconsistent Optimization Objectives**: Selecting a subset of clients for computation in each round may result in an inconsistency between the optimization objectives and the real target data distribution, particularly problematic in non-IID (non-identically distributed) FL settings.

Problem statement

The article focuses on two key issues in federated learning:

1. **Gradient Biases** -> Unbiased Gradient Aggregation (UGA)
2. **Inconsistent Optimization Objectives** -> Controllable FedMeta Updating

Problem statement

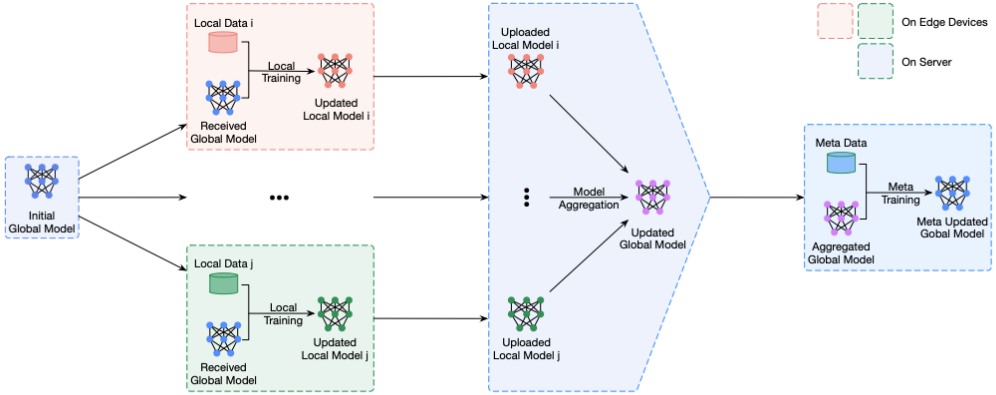


Figure 1: A typical round in FedMeta: model distributing, local training with either FedAvg [25] or UGA (Section 3.1), model aggregating and meta updating. (Better viewed in color)

Unbiased Gradient Aggregation (UGA)

This technique leverages a keep-trace gradient descent and gradient evaluation strategy to calculate gradients against the initial global model parameters, effectively reducing gradient biases:

Algorithm 1 Unbiased Gradient Aggregation

Server Executes:

Initialize ω_0
for each round $t = 0, 1, \ldots$ **do**
  $m \leftarrow \max(C \cdot K, 1)$
  $S_t \leftarrow$ (random set of m clients)
  for each client $k \in S_t$ **do in parallel**
    $g_t^k \leftarrow$ **ClientUpdate**(k, ω_t)
  end for
  $\omega_{t+1} \leftarrow \omega_t - \eta_g \sum_{k \in S_t} \frac{n_k}{n_{S_t}} g_t^k$ ▷ *Equation (14)*
end for

ClientUpdate(k, ω_t)**:** ▷ *Run on client* k
for i in the total steps of the first $E - 1$ epochs **do**
  $\omega_t^{k(i)} \leftarrow \omega_t^{k(i-1)} - \eta g_t^{k(i)}$ ▷ with *Keep-trace GD*
end for
$g_t^k = \nabla_{\omega_t} \mathcal{L}(\omega_t^k; \mathcal{D}_k)$ ▷ *Equation (13)*
return g_t^k to server

Controllable FedMeta Updating

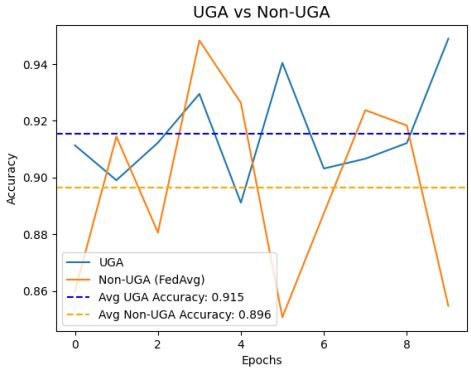
This approach introduces an additional meta updating procedure using a small set of data samples after model aggregation in each round:

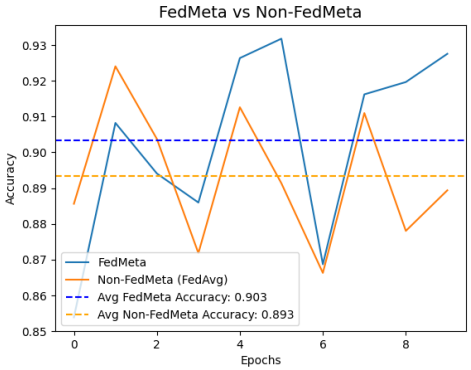
Algorithm 2 FedMeta

Server Executes:

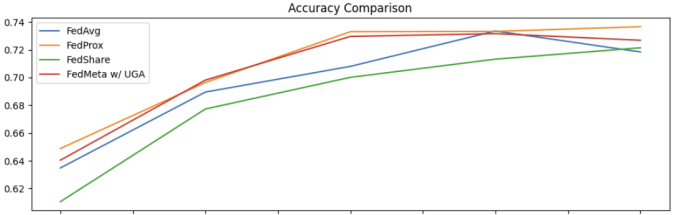
Initialize ω_0
for each round $t = 0, 1, \ldots$ **do**
  $m \leftarrow \max(C \cdot K, 1)$
  $S_t \leftarrow$ (random set of m clients)
  for each client $k \in S_t$ **do in parallel**
    $g_t^k \leftarrow$ **ClientUpdate**(k, ω_t)
   // Compatible with both FedAvg and Algorithm 1
  end for
  $\omega_{t+1} \leftarrow \omega_t - \eta_g \sum_{k \in S_t} \frac{n_k}{n_{S_t}} g_t^k$ ▷ *Equation (14)*
  $\omega_{t+1}^{meta} = \omega_{t+1} - \eta_{meta} \nabla_{\omega_{t+1}} \mathcal{L}(\omega_{t+1}; \mathcal{D}_{meta})$
 // Equation (20)
end for

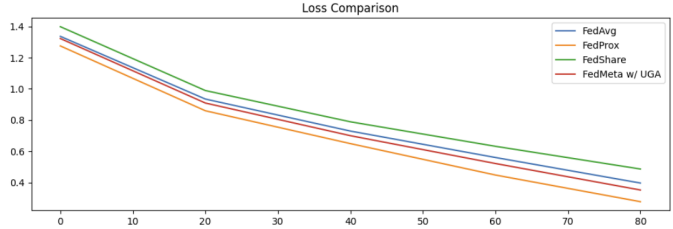
UGA and FedMeta implementation

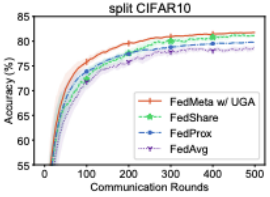


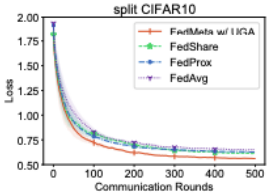


Experiments reproducing - CIFAR-10 dataset

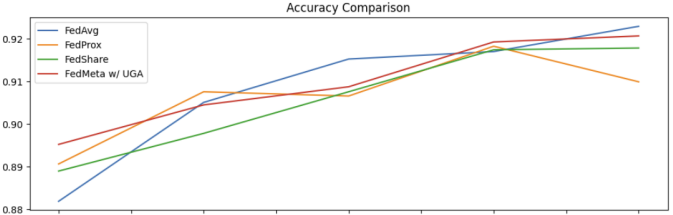


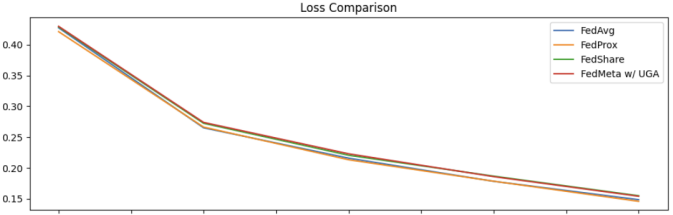


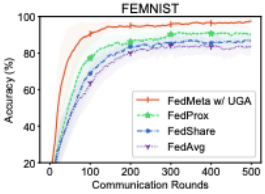


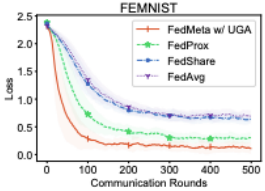


Experiments reproducing - FEMNIST dataset

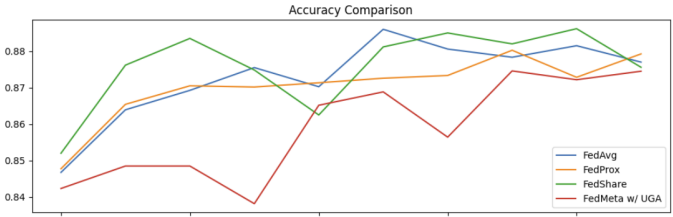


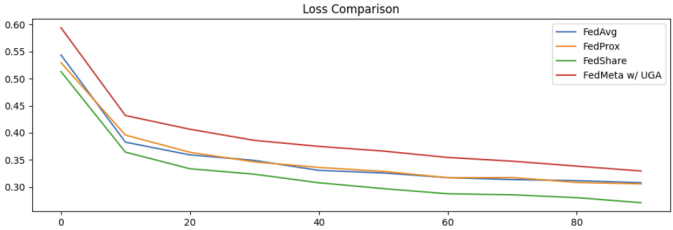


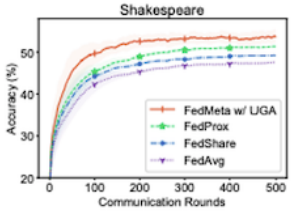


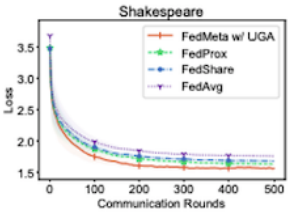


Experiments reproducing - Shakespeare dataset

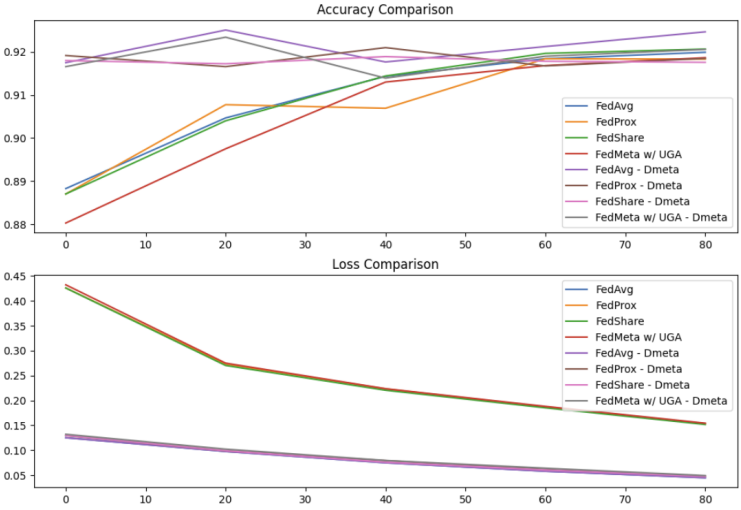








Experiments reproducing - more experiments



Questions?