ADAPTIVE LOCAL TRAINING IN FEDERATED LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Federated Learning is a machine learning paradigm where multiple clients collaboratively train a global model by exchanging their locally trained model weights instead of raw data. In the standard setting, every client trains the local model for the same number of epochs. We introduce ALT (Adaptive Local Training), a simple yet effective feedback mechanism that could be introduced at the client side to limit unnecessary and degrading computations. ALT dynamically adjusts the number of training epochs for each client based on the similarity between their local representations and the global one, ensuring that well-aligned clients can train longer without experiencing client drift. We evaluated ALT on federated partitions of the CIFAR-10 and TinyImageNet datasets, demonstrating its effectiveness in improving model convergence and stability.

1 INTRODUCTION

Federated learning (FL) (McMahan et al., 2017) has emerged as a machine learning approach prioritizing privacy while fostering collaborative training, avoiding centralized data storage concerns.

Typically, in FL every client performs training for the same number of local epochs (Karimireddy et al., 2020; Li et al., 2021; Shenaj et al., 2023). However, different clients could have different computational capabilities, and communication speeds, and the server might request updates when the training is not yet concluded. In addition to that, when the clients' data distribution is very heterogeneous, training each client for a fixed pre-defined number of steps leads to client drift and complicates the aggregation.

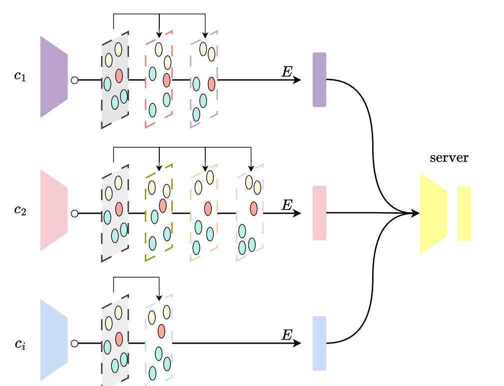
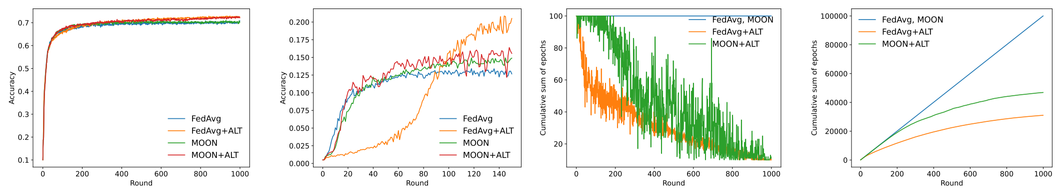


Figure 1: Overview of the proposed Federated learning strategy with dynamic local training epochs.

For this reason, we train for a variable number of local epochs across clients and training rounds, similarly to (Li et al., 2020; Michieli et al., 2022). In particular, we study the effect of dynamic local epochs on the client side and propose a client-side control strategy to mitigate representation drift, reduce communication costs, and improve performances.

Related Work. FedProx (Li et al., 2020) introduces a flexible framework that allows clients to perform variable amounts of local training, assuming that some clients may not complete their updates within a fixed time window. Additionally, it employs a dynamic regularizer in the local objective to mitigate the impact of inconsistent local updates.

Similarly, SCAFFOLD (Karimireddy et al., 2020) tackles client drift by estimating and correcting update directions for both the server and clients, ensuring more stable local updates. However, while these methods offer improvements, they fail to achieve significant performance gains over FedAvg in



(a) Accuracy curve on Cifar-10. (b) Accuracy curve on TinyImagenet. (c) Total number of epochs per round on Cifar-10. (d) Cumulative epochs per round on Cifar-10.

deep neural networks training, particularly when applied in realistic computer vision applications. To address this limitation, MOON (Li et al., 2021) proposes a model-based contrastive learning approach to enhance local training in non-IID settings. By enforcing similarity between the current local model and the incoming global model, while discouraging similarity with the previous local model, MOON effectively stabilizes training in deep networks.

Building on these insights, we introduce a simple yet effective mechanism to control the length of local training, opening new research opportunities in adaptive and dynamic federated learning.

2 Method

Let us assume a set of clients $\mathcal{K}$, where each client $k \in \mathcal{K}$ has access to a local set of samples $(\mathcal{X}_k, \mathcal{Y}_k)$ from a dataset $\mathcal{D}_k$ with $|\mathcal{D}_k| = n_k$. At each communication round $r \in \{1, \dots, R\}$, the server selects a subset of clients $S \subset \mathcal{K}$ and sets an adaptive threshold $T_h(r) = a + \frac{b \cdot r}{R}$, that linearly increases during the training (we set $a = 0.1$ and $b = 0.8$, see Appendix B for more details). Each client $s \in S$ then initializes its local model θ_s^r with the global model θ^r and trains for E_s^r local epochs, where E_s^r varies across clients and rounds.

At each training step, as usual in neural networks training, the local model is updated using gradient descent on mini-batches $\mathcal{B} \subset \mathcal{D}_s$, i.e., $\theta_s^r \leftarrow \theta_s^r - \eta \nabla \mathcal{L}_s(\theta_s^r; \mathcal{B})$.

Let us denote with $p_s = f(w_s, \mathcal{B})$ and $p_g = f(w_g, \mathcal{B})$ the feature embeddings of the local and global models for the considered batch: the local training halts as soon as the similarity condition $\cos(p_s, p_g) < T_h(r)$ is met, i.e., when the difference between the embeddings is smaller than the threshold. If the condition is never met, it will stop as usual after the maximum number of local epochs E is reached. Note that the threshold increases with time, i.e., the criteria becomes more and more strict while the difference w.r.t. the starting model typically increases.

After local training, the client models are sent to the server, which aggregates them using standard federated averaging, i.e.,: $\theta^{r+1} = \sum_{s \in S} \frac{n_s}{n} \theta_s^r$. The process is repeated for R rounds. The algorithm is detailed in Appendix C.

3 Results

We evaluate the performance of our algorithm on the CIFAR-10 (Krizhevsky et al., 2009), and Tiny-Imagenet (Le & Yang, 2015) datasets. As a strong baseline for local training, we consider MOON. We consider $|\mathcal{K}| = 100$ clients (with 10% participation), and the data is partitioned according to the Dirichlet distribution with the concentration parameter $\alpha = 100$. By looking at the Figure, we can notice that our method allows to reduce substantially carbon footprint by reducing the cumulative epochs per round (sum of all clients epochs), and leads also to improved performance. In particular, we notice that FedAvg + ALT (FedALT) could work similarly or better then MOON, while being much more efficient.

4 Conclusion

In this work, we introduced a representation learning feedback mechanism to control the number of local epochs reducing energy consumption and communication costs which can be seamlessly integrated into FL algorithms.

REFERENCES

Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank Reddi, Sebastian Stich, and Ananda Theertha Suresh. Scaffold: Stochastic controlled averaging for federated learning. In *International conference on machine learning*, pp. 5132–5143. PMLR, 2020.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Ya Le and Xuan Yang. Tiny imagenet visual recognition challenge. *CS 231N*, 7(7):3, 2015.

Qinbin Li, Bingsheng He, and Dawn Song. Model-contrastive federated learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10713–10722, 2021.

Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. *Proceedings of Machine learning and systems*, 2:429–450, 2020.

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial Intelligence and Statistics (AISTATS)*, pp. 1273–1282. PMLR, 2017.

Umberto Michieli, Marco Toldo, and Mete Ozay. Federated learning via attentive margin of semantic feature representations. *IEEE Internet of Things Journal*, 10(2):1517–1535, 2022.

Donald Shenaj, Giulia Rizzoli, and Pietro Zanuttigh. Federated learning in computer vision. *IEEE Access*, 2023.

APPENDIX

A IMPLEMENTATION DETAILS

We adopt as feature extractor f a simple CNN with 2 convolutional layers for CIFAR-10, and ResNet-50 for Tiny-Imagenet. We use the SGD optimizer with a learning rate 0.01 for all approaches. The SGD weight decay is set to 0.00001 and SGD momentum is set to 0.9. The batch size is set to 64. The maximum number of local epochs E is set to 10, and the number of communication rounds R is 1000 for Cifar-10 and 150 for TinyImagenet.

B SELECTION OF THE THRESHOLD FUNCTION

We experimented different threshold functions $T_h(r)$, including 3 simple approaches:

- Linear Increasing: $T_h(r) = 0.1 + 0.8 \cdot \frac{r}{R}$
- Linear Decreasing: $T_h(r) = 0.9 - 0.8 \cdot \frac{r}{R}$
- Fixed: $T_h(r) = c$, where c is a constant.

Among these strategies, the most effective was the linear increasing threshold. This approach enables a 'slow start,' where early training rounds allow for more flexibility in local updates before gradually enforcing stricter similarity constraints. This progressive tightening helps balance exploration and convergence, leading to improved overall model performance. However, in practice, any thresholding (including a fixed rule) is useful to reduce carbon footprint.

C ALGORITHM

Input: Initialize model parameters θ
    Initialize maximum rounds R
    Initialize threshold parameters a, b
Server executes:
for $r = 1$ **to** R **do**
    $S \leftarrow$ Random subset of clients
    $T_h(r) \leftarrow a + \frac{b*r}{R}$
    for each client $i \in S$ **do**
        $\theta_i \leftarrow$ ClientUpdate($i, \theta, r, T_h(r)$)
    end for
    $\theta \leftarrow \sum_{i \in S} \frac{n_i}{n} \theta_i$
end for
ClientUpdate($i, \theta, r, T_h(r)$)**:**
if $r = 1$:
    $\theta_i \leftarrow \theta$
    $\theta_i := \{w_i, v_i\}$
$\theta_g \leftarrow \theta$
$\theta_g := \{w_g, v_g\}$
stop $\leftarrow False$
for $j = 1, 2, ..., E$ and stop $= False$ **do**
    for each batch $\mathcal{B}$ in $\mathcal{D}_i$ **do**
        $p_i \leftarrow f(w_i, \mathcal{B})$
        $p_g \leftarrow f(w_g, \mathcal{B})$
        if cos$(p_i, p_g) < T_h(r)$: stop $\leftarrow True$
        $\theta_g \leftarrow \theta_g - \eta \nabla \mathcal{L}(\theta_g; \mathcal{B})$
$\theta_i \leftarrow \theta_g$
return θ_i

Algorithm 1: Implementation of FedAvg with the ALT stopping criteria (FedALT)