A Novel Approach for Improving Generalization in Federated Learning Through Corrective Gradient Weights

Ali Mousazadeh
polytechnic university of turin
Turin, Italy
Ali.Mousazadeh@studenti.polito.it

Hossein Khodadadi
polytechnic university of turin
Turin, Italy
Hossein.Khodadadi@studenti.polito.it

Sohrab Salehi
polytechnic university of turin
Turin, Italy
Sohrab.Salehi@studenti.polito.it

December 31, 2023

Abstract

Federated Learning (FL) is a framework for training a machine learning model in which the data exists in clients and cannot be accessed directly. The model is sent to these clients and is trained locally. Then, these trained models are aggregated by a server into a global model. In this scenario, the data amongst clients may vary greatly in terms of distribution, which could pose a challenge in the aggregation phase since the global model will have a tendency to lack proper generalization. This, in turn, could lead to poor performance on previously unseen data. In order to address this problem, we propose a method which takes advantage of the gradients of clients with respect to their local data in order to estimate the possible effect of each client's model on other clients. We formulate a constrained optimization problem in order to find the optimal correction terms to be added to the FedAvg baseline model, which adjusts the global model such that clients that have a better estimated loss reduction on other clients will have an increased say on the direction the global model will take. We test the performance of our method against the FedAvg baseline on the Federated EMNIST dataset. We find that our method outperforms the FedAvg baseline both in terms of convergence speed and mean accuracy.

1 Introduction

In this work, we propose a novel method that takes advantage of the gradients of clients in order to estimate the similarity of clients to each other. Using these gradients, we define a constrained optimization problem which tries to maximize the reduction in loss of each client given an update which combines the gradients of all clients. Our goal with this method is to address the differences in client models which may result from different distributions of data. The aim is the correct the average model in such a way that more "useful" clients with better generalization capabilities get an increased say in the direction of the global model.

2 Methodology

In this section, we describe the proposed method and argue for its benefits in addressing data heterogeneity

issues and achieving better generalization.

Consider the baseline FedAvg scenario in which during each round, some clients are randomly selected. They then receive the global model, train it using their local data, and send their models to the server. Let the loss function for client i be defined as:

$$L_i(W_i, t, y) = \frac{1}{M}\sum_{p=1}^{M}\Lambda(t_p, y_p) \tag{1}$$

where Λ is some non-negative criterion such as cross-entropy and M is the number of samples inside client i. In our method, we ask each client to calculate the gradient of its loss function with respect to its parameters given the local model it has trained during this round and its data. We denote this gradient by $\frac{\partial L_i}{\partial W_i}$ where i indicates the ith client selected during this round and W_i is a flattened vector containing all the trainable parameters of the client's model. We then consider a first-order Taylor expansion for the loss of client i in a small neighborhood around W_i:

$$\Delta L_i = \frac{\partial L_i}{\partial W_i}.(W - W_i) = \frac{\partial L_i}{\partial W_i}.\Delta W \tag{2}$$

Where . denotes the dot product. We set:

$$\Delta W = -\sum_{j=1}^{n} x_j \frac{\partial L_j}{\partial W_j} \tag{3}$$

where x_j is a weight assigned to the gradient of the jth client and n is the number of clients per round. This leads to:

$$\Delta L_i = -\frac{\partial L_i}{\partial W_i}.(\sum_{j=1}^{n} x_j \frac{\partial L_j}{\partial W_j}) \tag{4}$$

which is an estimation of the change in the loss of client i given the weighted sum of gradients of all clients. If we set the same value for ΔW across all clients, the sum of the changes in the losses of all clients becomes:

$$\Delta L = -\sum_{k=1}^{n}(\frac{\partial L_k}{\partial W_k}.(\sum_{j=1}^{n} x_j \frac{\partial L_j}{\partial W_j})) \tag{5}$$

rearranging the terms, we get:

$$\begin{aligned}\Delta L = -\sum_{i=1}^{n} x_i \frac{\partial L_i}{\partial W_i}.(\sum_{j=1}^{n} \frac{\partial L_j}{\partial W_j}) = \\ -\sum_{i=1}^{n} x_i (D_i.D) = -\sum_{i=1}^{n} x_i m_i\end{aligned} \tag{6}$$

by setting $D_i = \frac{\partial L_i}{\partial W_i}$, $D = \sum_{j=1}^{n} \frac{\partial L_j}{\partial W_j}$, and $m_i = D_i.D$. Ideally, our goal is to minimize ΔL or equivalently maximize:

$$J(x) = -\Delta L = \sum_{i=1}^{n} x_i m_i \tag{7}$$

However, since (7) is a linear function of x, its maximum is infinity without some sort of limit on x. Furthermore, it is important to consider the validity of the first order Taylor approximation in the desired neighborhood. If the gradient of each client with respect to its model parameters is bounded and well-behaved, the parameters of the model are bounded, the neighborhood defined by ΔW is sufficiently small, and the loss function in this neighborhood behaves quite linearly with respect to small changes in model parameters, then it would be reasonable to assume that the linear approximation is quite accurate, although some error would still remain. Of these conditions, the ones concerning the linear and smooth behavior of the loss function within some small neighborhood of the trained state can be addressed to some extent through choosing appropriate training hyper parameters such as batch size [1], learning-rate, and momentum. Tuning these hyper parameters in such a way that would lead to better generalization and smoother minima makes the loss function more well-behaved, which in turn helps the first-order Taylor approximation to be more reliable. If the well-behaved conditions were to be explicitly satisfied, we'd need the loss function of each client to be Lipschitz differentiable [3] having $K \leq ||D_i||$ where K is the Lipschitz constant. This is a rather strong assumption and furthermore, it is not easy to measure the Lipschitz constant of neural networks in practice [2] specially for more complex networks. Generally it's more convenient to set appropriate hyper parameters which lead to better convergence rather than

checking if the loss functions are Lipschitz differentiable for each client in advance.

We now consider the condition regarding the neighborhood spanned by ΔW to be small. Ideally, we would like to define a constraint which limits the span of this neighborhood, whilst simultaneously leading to an easy constrained optimization problem that can be solved relatively quickly and preferably in closed form. We should also consider that after finding a solution for a constrained optimization problem, we must also check if the solution is a maximum, minimum, or saddle point. It would be ideal if the chosen constraint in combination with (7) would lead to an easily verifiable maximum as well. Furthermore, it would be desirable if the values of x_i vary between clients and do not remain constant, as constant weights would mean taking simple full-batch steps which typically aren't good for generalization.

After defining an appropriate constraint and finding the optimal values for x_i, the way we create the new global model for the next round is to average the corrected client models which would lead to:

$$W_{global} = W_{FedAvg} - \sum_{i=1}^{n} x_i D_i \tag{8}$$

Initially, one could consider the most direct constraint which enforces the span of the neighborhood, which is:

$$||\Delta W||^2 - c^2 = 0 \tag{9}$$

where c is some chosen small constant. If the conditions for the linear approximation being fairly accurate are satisfied, we have the following inequality which results from (2):

$$||\Delta L_i|| < ||\Delta W||.||D_i|| \tag{10}$$

Therefore given some desired upper bound on the change in loss denoted by ϵ, we have:

$$||\Delta W|| < \frac{\epsilon}{||D_i||} \tag{11}$$

It would be reasonable to choose ϵ in such a way that would scale with the value of the loss function at the point of estimation. Leading to:

$$||\Delta W|| < \frac{\alpha L_i}{||D_i||} \tag{12}$$

where $\alpha \in [0, 1]$ in this case determines the percentage of the desired upper bound of change in loss considering the loss of clients at the point of estimation. And since this must be considered for each client, we can pick:

$$c = min \frac{\alpha L_i}{||D_i||} \ \forall i \tag{13}$$

Therefore, the constrained optimization task becomes:

$$max f(x) = \sum_{i=1}^{n} x_i m_i \tag{14}$$

such that:

$$g(x) = ||\Delta W||^2 - c^2 = 0 \tag{15}$$

The Lagrangian is:

$$\mathcal{L} = f(x) - \lambda g(x) = \sum_{i=1}^{n} x_i m_i - \lambda(||\Delta W||^2 - c^2) \tag{16}$$

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial x_i} &= m_i - 2\lambda D_i(\sum_{j=1}^{n} x_j D_j) \\ &= D_i(\sum_{j=1}^{n} D_j) - 2\lambda D_i(\sum_{j=1}^{n} x_j D_j) = 0 \end{aligned} \tag{17}$$

Setting $x_j = \frac{1}{2\lambda}$ for all j, we get:

$$\frac{\partial \mathcal{L}}{\partial x_i} = D_i(\sum_{j=1}^{n} D_j) - D_i(\sum_{j=1}^{n} D_j) = 0 \tag{18}$$

Therefore:

$$x_i = x_j = \frac{1}{2\lambda} \tag{19}$$

Inserting this into (15), we get:

$$\frac{1}{4\lambda^2} \sum_{k=1}^{n} \sum_{p=1}^{n} D_k.D_p = \frac{1}{4\lambda^2} D.D = c^2 \tag{20}$$

$$\lambda = \pm \frac{\sqrt{D.D}}{2c} \tag{21}$$

Inserting this into (19) leads to:

$$x_i = \pm \frac{c}{\sqrt{D.D}} \tag{22}$$

where the positive and negative answers lead to the maximum and minimum of (15) respectively. Therefore:

$$x_i = \frac{c}{\sqrt{D.D}} \tag{23}$$

Due to the linear nature of the optimization function and the quadratic constraint, the solutions are mirrored by their sign which denote the maximum and the minimum. Therefore we do not need to check for saddle points. Technically the constraint (15) does not need to be an equality and in fact, it's < 0 since any point for which $||\Delta W||$ is sufficiently small can be a possible solution. However, since this is a maximization problem, the objective function is linear, the constraint is convex, and λ is positive at the maximum, that means relaxing the constraint would lead to a higher maximum. This in turn means the maximum value of the objective function for any $c_0^2 < c^2$ is going to be less than the maximum of the objective function given c^2. Therefore, we do not need to consider any point on the interior of the condition and we can replace < 0 with $= 0$ as we have done in (15).

While the solutions found in this formulation of the problem are valid, they give equal weight to all the clients regardless of the similarity of their gradients. This happens due to the nature of the constraint (15). This would effectively mean correcting the FedAvg model by the sum of the full-batch gradients of clients, which isn't desirable. We would like to keep the quadratic nature of the constraint since it is more likely to lead to an easily solvable problem, but we would also like the constraint to lead to a solution which considers the effects of the clients on each other.

We consider a different constraint of the following form:

$$g(x) = (\sum_{i=1}^{n} x_i^2 ||D_i||^2) - c^2 = 0 \tag{24}$$

where c is the same as in (13). Unlike (15), this constraint no longer enforces $||\Delta W||$ to be small enough such that the changes in the loss are of the order αL_i. It still limits $||\Delta W||$, however the hyper parameter α has lost the clear interpretation it had in the previous constraint. This constraint encourages giving smaller weights to clients with bigger local gradient norms, therefore it pushes the correction term in favor of the clients with harder to minimize losses. Furthermore, since it involves the term x_i^2, it allows the weight for a client to be negative as well. As we will see, this has the interesting outcome that if the estimated bad effect a client has on others outweighs the estimated good effect it has on its local loss, it will be given a negative weight in the correction term to run back a portion of its contribution to the FedAvg model.

Given (24), The Lagrangian becomes:

$$\mathcal{L} = \sum_{i=1}^{n} x_i m_i - \lambda (\sum_{i=1}^{n} x_i^2 ||D_i||^2 - c^2) \tag{25}$$

We have:

$$\frac{\partial \mathcal{L}}{\partial x_i} = m_i - 2\lambda x_i ||D_i||^2 = 0 \tag{26}$$

$$x_i = \frac{1}{2\lambda} \frac{m_i}{||D_i||^2} \tag{27}$$

Inserting (27) into (24) we get:

$$\lambda = \pm \frac{1}{2c} \sqrt{\sum_{j=1}^{n} (\frac{m_j}{||D_j||})^2} \tag{28}$$

Similar to the previous constraint, the positive value for λ is the one leading to a maximum. This leads to the following value for x_i:

$$x_i = \frac{c}{\sqrt{\sum_{j=1}^{n} (\frac{m_j}{||D_j||})^2}} \frac{m_i}{||D_i||^2} \tag{29}$$

Through inspecting (29) we see that for a solution to exist, we must have $\forall i : \ ||D_i|| \neq 0$. We also see that x_i relates linearly to α, therefore the hyper parameter α will linearly scale the weights of clients. Setting $\alpha = 0$ will reduce the correction term to 0 and therefore the global model will be updated with the FedAvg model. The appropriate value for α should

be set in the same manner as with any other hyper parameter, which would be through checking the convergence on the train data or the accuracy on the validation set.

As briefly mentioned earlier, we have:

$$sign(x_i) = sign(m_i) = sign(D_i.D) \tag{30}$$

Therefore the sign of a client's weight is determined by the dot product of its gradient with the sum of the gradients of all clients. This means the contribution of a client to the FedAvg model is reversed to some extent determined by α if its gradient is dissimilar to the average gradients of the sampled clients during each round.

3 Experiments

In this section we discuss the results of some experiments on our proposed method. We compare the mean test accuracy of our method against the baseline FedAvg for different random seeds and values of α. In particular, we check how well these methods converge improve in terms of mean test accuracy, and how much their mean test accuracy differs when sufficient convergence has been reached.

We see in Table 1 and Table 2 that as we increase the value of α, we typically get better convergence in both the IID and non-IID cases. As the FedAvg ($\alpha = 0$) and the proposed method reach the final rounds of training, the gap between the two becomes smaller. Our experiments show that the gap between the FedAvg and centralized accuracy on the EMNIST dataset is quite small, therefore it is not easy to measure an improvement for a given federated learning method. Nonetheless, our experiments show that the proposed method does perform better, even if by a small margin.

In the non-IID case, higher values of α can be selected and the correction can be made more aggressively. However it's typically a better idea to go for smaller values of α in the IID case. While we get a boost in mean accuracy as we increase α, it starts to oscillate as we increase α by too much. The value for this hyper parameter depends on several factors, including but not limited to: model and data complexity, the distribution of labels and inputs inside clients, and uniform or non-uniform access to clients. A full study of the effects of all possible factors on α is beyond the scope of this report, however in any case its value can be tuned by checking the performance on the validation set.

Seed/α	0	0.1	0.2	0.3	0.4
0	78.57	78.88	79.01	**79.22**	79.02
	83.33	83.37	83.38	**83.41**	83.39
13	78.06	78.39	78.69	**78.92**	78.86
	83.31	83.29	83.31	83.31	**83.41**
42	78.29	78.67	78.93	79.20	**79.26**
	83.05	83.01	82.99	83.13	**83.25**

Table 1: Mean test accuracy of the proposed method on the non-IID FEMNIST dataset for different choices of α. Top number in each cell is the mean test accuracy of rounds 171-200, while the bottom number is the mean test accuracy of rounds 901-1000.

Seed/α	0	0.025	0.05	0.075	0.1
0	79.61	79.73	79.82	79.91	**79.99**
	83.21	83.21	83.27	83.29	**83.29**
13	79.93	79.99	80.05	80.14	**80.16**
	83.74	83.86	83.91	83.99	**84.06**
42	81.07	81.11	81.14	**81.18**	81.16
	83.97	84.00	84.03	84.06	**84.15**

Table 2: Mean test accuracy of the proposed method on the IID FEMNIST dataset for different choices of α. Top number in each cell is the mean test accuracy of rounds 6-15, while the bottom number is the mean test accuracy of rounds 41-50.

4 Conclusion

In this report, we introduced a novel method which corrects the FedAvg model by a weighted sum of the gradients of clients. These weights are calculated as the solution to an optimization problem which aims to minimize the local losses of clients through this correction, given a condition and hyper parameter

which limits the strength of the correction. We have shown that our method performs better than the FedAvg model on the Federated EMNIST dataset in terms of convergence and accuracy.

References

[1] Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. *arXiv preprint arXiv:1609.04836*, 2016. 2

[2] Aladin Virmaux and Kevin Scaman. Lipschitz regularity of deep neural networks: analysis and efficient estimation. *Advances in Neural Information Processing Systems*, 31, 2018. 2

[3] Han Wang, Siddartha Marella, and James Anderson. Fedadmm: A federated primal-dual algorithm allowing partial participation. In *2022 IEEE 61st Conference on Decision and Control (CDC)*, pages 287–294. IEEE, 2022. 2