Weight Erosion: An Update Aggregation Scheme for Personalized Collaborative Machine Learning

Felix Grimberg(✉), Mary-Anne Hartley, Martin Jaggi, and Sai Praneeth Karimireddy

Ecole polytechnique fédérale de Lausanne, Lausanne, Switzerland
felix.grimberg@hotmail.com

Abstract. Background. In medicine and other applications, the copying and sharing of data is impractical for a range of well-considered reasons. With federated learning (FL) techniques, machine learning models can be trained on data spread across several locations without such copying and sharing. While good privacy guarantees can often be made, FL does not automatically incentivize participation and the resulting model can suffer if data is non-identically distributed (non-IID) across locations. Model personalization is a way of addressing these concerns. **Methods.** In this study, we introduce Weight Erosion: an SGD-based gradient aggregation scheme for personalized collaborative ML. We evaluate this scheme on a binary classification task in the Titanic data set. **Findings.** We demonstrate that the novel Weight Erosion scheme can outperform two baseline FL aggregation schemes on a classification task, and is more resistant to over-fitting and non-IID data sets.

1 Background

In medicine and other applications, data is siloed for a range of well-considered reasons including confidentiality and governmental regulations. This creates inequitable access to probabilistic medicine, where those with less accessible or smaller silos are under-represented in medical literature. Additionally, "ownership" of potential scientific results creates an environment of competition in which researchers are reluctant to share their intellectual property (IP). Moreover, delays in granting access can render time-sensitive epidemiological data substantially less relevant and can lead to outdated and poorly adaptive models.

Federated learning (FL) can transform access to highly sensitive data by jointly training a machine learning (ML) model through collaboration across silos without copying the data onto a central server [11]. However, it does not automatically protect IP, and blindly learning a single global model on possibly non-identically distributed (non-IID) data risks creating uninformative insights. One way of addressing these concerns is model personalization, e.g. through *Featurization*, *Multi-task learning*, or *Local fine-tuning* [7,9,10,14].

S. Albarqouni et al. (Eds.): DART 2020/DCL 2020, LNCS 12444, pp. 160–169, 2020.
https://doi.org/10.1007/978-3-030-60548-3_16

The *Weight Erosion* scheme presented here optimizes a personalized model for one silo, as opposed to conventional personalized FL methods which train personalized models for all silos simultaneously. While it is conceptually related to local fine-tuning, the *Weight Erosion* scheme is optimized to discard contributions from unhelpful[1] silos as early as possible in the training process.

2 Setting and Objective

We consider a network of agents i collecting samples from the underlying distributions $\mathcal{D}_i$. One agent (agent 0) is called the user and wishes to perform an inference task on $\mathcal{D}_0$. The task of *personalized collaborative ML*, in a broad sense, is to give agent 0 a training algorithm which discriminates between the available agents in some way to minimize the true loss of the resulting model on $\mathcal{D}_0$.

Example Clinical Setting. The agents could be individual hospitals dispersed across one or several regions. The aetiology of common, generic symptoms such as fever is highly dependent on geographic location. For instance, rural populations suffer more vector borne diseases such as malaria, while fevers in the urban setting tend to be related to respiratory disease. A clinician (agent 0) might want to train a ML model to help diagnose the patients admitted to their urban hospital. The number N_0 of samples collected at their hospital, however, is too limited to yield a satisfactory model, so agent 0 considers using FL to leverage data from other hospitals. Knowing that rural populations suffer from fairly different problems than agent 0's urban patients, agent 0 pre-selects only other urban hospitals – in a way, agent 0 performs manual model personalization based on prior medical knowledge. However, agent 0 suspects the presence of other confounding variables that could cause the samples collected by *some* of the other urban hospitals to negatively affect the diagnostic accuracy of the trained model on new patients treated by agent 0.

For personalized collaborative ML, formally, we are given:

- A set of agents $i \in \mathcal{U} = \{0, 1, \ldots, N\}$
  - Each agent i has collected a set of samples (called *data set*), on the domain $\mathcal{X} \times \mathcal{Y}$: $\mathcal{S}_i = \left\{\mathbf{x}_i^{(n)}, y_i^{(n)}\right\}_{n=1,\ldots,N_i}$, each from an (unknown) underlying distribution $\mathcal{D}_i$: $\left(\mathbf{x}_i^{(n)}, y_i^{(n)}\right) \overset{i.i.d.}{\sim} \mathcal{D}_i$.
  - We assume that the label y_0 is not independent of the features $\mathbf{x}_0$ under $\mathcal{D}_0 : p_0(y|\mathbf{x}) \neq p_0(y)$
- A class of models $\mathcal{M}$ s.t. $f : \mathcal{X} \to \mathcal{Y}\ \forall f \in \mathcal{M}$. For instance, $\mathcal{M}$ could be the class of linear models where $f(\mathbf{x}) = \mathbf{w}^\top \mathbf{x}$, $\mathbf{w} \in \mathbb{R}^D$.

[1] The samples stored in other silos can be unhelpful for various reasons. One such reason is intentionally byzantine behaviour, which can be successfully addressed with existing byzantine-robust FL methods [5] and local fine-tuning. However, this approach does not cover cases where silos contain data sampled from a different distribution as laid out in Sect. 2.

– A loss function: $\ell(y, \hat{y})$. For instance, the loss function could be the mean squared error (MSE).

We measure the performance of a model $f \in \mathcal{M}$ by its *expected loss* on $\mathcal{D}_0$, defined in Eq. 1.

$$\mathcal{L}_{\mathcal{D}_0}(f) = \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}_0}\left[\ell\left(y, f\left(\mathbf{x}\right)\right)\right] \tag{1}$$

Our goal is to find a training algorithm $\mathcal{A}$ which, given $\mathcal{U}$, $\mathcal{M}$, and ℓ, will produce the model $f = \mathcal{A}(\mathcal{U}, \mathcal{M}, \ell) \in \mathcal{M}$ which minimizes the personalized true loss $\mathcal{L}_{\mathcal{D}_0}(f)$ on $\mathcal{D}_0$ (Eq. 2). This is opposed to standard federated learning methods $\mathcal{A}^{FL}_{standard}$, which consider that all non-malicious agents have samples from the same underlying distribution. Applying such a method in the present setting would minimize the loss on a weighted sum of the agents' distributions $\mathcal{D}_i$, $i \in \mathcal{U}$ instead, as shown in Eq. 3. The weights λ_i depend on the specific learning method, but they are often either $\lambda_i = \frac{1}{|\mathcal{U}|} \forall i \in \mathcal{U}$ or $\lambda_i = \frac{|\mathcal{S}_i|}{\sum_{i\in\mathcal{U}}|\mathcal{S}_i|}$.

$$\textbf{Personalized coll. ML:} \quad find: \quad \mathcal{A} = \arg\min_{\mathcal{A}'} \mathcal{L}_{\mathcal{D}_0}\left(\mathcal{A}'\left(\mathcal{U}, \mathcal{M}, \ell\right)\right) \tag{2}$$

$$\textbf{Standard FL result:} \quad \mathcal{A}^{FL}_{standard} = \arg\min_{\mathcal{A}'} \sum_{i\in\mathcal{U}} \lambda_i \mathcal{L}_{\mathcal{D}_i}\left(\mathcal{A}'\left(\mathcal{U}, \mathcal{M}, \ell\right)\right) \tag{3}$$

3 The Weight Erosion Aggregation Scheme

We present a novel adaptation of federated training algorithms based on *robust aggregation rules* such as in [1,5,13]. Briefly, each agent i is initially given a weight $\alpha_i^0 = 1$. At each round, they compute a mini-batch gradient[2] $\mathbf{g}_i$ and the relative distance $d^{rel}_{i,0}$ between $\mathbf{g}_i$ and $\mathbf{g}_0$ (Eq. 4). The weight α_i is then decreased by a small amount that depends on the distance $d^{rel}_{i,0}$ (Eq. 5). The weights α_i are finally used to compute a personalized weighted average of the gradient vectors $\mathbf{g}_i$ (Eq. 6).

$$d^{rel}_{i,0} = \frac{\|\mathbf{g}_i - \mathbf{g}_0\|}{\|\mathbf{g}_0\|} \geq 0 \tag{4}$$

$$\alpha_i^r = \max\left\{0, \alpha_i^{r-1} - \left(1 + p_s\left\lfloor\frac{(r-1)b}{|\mathcal{S}_i|}\right\rfloor\right) p_d \cdot d^{rel}_{i,0}\right\} \tag{5}$$

$$\bar{\mathbf{g}} \leftarrow \frac{\sum_{i\in\mathcal{U}} \alpha_i^r \mathbf{g}_i}{\sum_{i\in\mathcal{U}} \alpha_i^r} \tag{6}$$

As the number of rounds increases, samples from smaller data sets will be seen more often than samples from larger data sets, because each agent uses the same number of samples per round. To balance this over-representation, the change in α_i (Eq. 5) is made to depend also on the average number of times each sample in $\mathcal{S}_i$ has been used. In Eq. 5, b stands for the batch-size, while p_d is the distance penalty factor and p_s is the size penalty factor. The values of

[2] Alternatively, the agents can compute local SGD updates, used in the same way.

Algorithm 1: WEIGHT EROSION

Data: A set of agents $i \in \mathcal{U}$, with associated data sets $\mathcal{S}_i$, a model class $\mathcal{M}$, and a gradient-based machine learning algorithm $\mathcal{A}$.
Result: A personalized machine learning model f.
Set a number of rounds r_{max}, a batch size b, a distance penalty factor p_d, and a size penalty factor p_s ;
Initialize $\alpha_i^0 \leftarrow 1 \; \forall i \in \mathcal{U}$;
Randomly initialize the model $f \in \mathcal{M}$;
for r *from* 1 *to* r_{max} **do**
  for $i \in \mathcal{U}$, *starting with* $i = 0$ **do**
    Select a batch of size b from $\mathcal{S}_i$ and compute a gradient $\mathbf{g}_i$;
    Compute the distance $d_{i,0}^{rel}$ (Equation 4) and α_i^r (Equation 5) ;
  end
  $\bar{\mathbf{g}} \leftarrow \frac{\sum_{i \in \mathcal{U}} \alpha_i^r \mathbf{g}_i}{\sum_{i \in \mathcal{U}} \alpha_i^r}$;
  Update the model f based on $\bar{\mathbf{g}}$;
end

these penalty factors should be picked by the user, as they control the degree of personalization. Selecting a large value for p_d leads to a rapid decay of all agent weights, limiting the collaborative scope of the training. Conversely, if p_d is very low, available agents' contributions are used equally regardless of their $d_{i,0}^{rel}$. Finally, p_s controls how strongly small data sets are penalized to counteract their over-representation.
We next discuss some salient properties of our algorithm.

Privacy and Robustness. Weight Erosion can be seamlessly integrated into the privacy-preserving FL protocol proposed in [5], which leverages secure multiparty computations (MPC) to obtain distances while keeping the individual gradients private at all times. Replacing the Byzantine robust aggregation rule by Weight Erosion maintains the strong privacy guarantees, together with a weaker form of robustness since any Byzantine agent i would likely see their weight α_i decline very fast.

Incentives and IP. Agents could be rewarded for their data collection efforts according to how many rounds they participate in, or according to their weight (summed over all rounds). Intuition tells us that the result of Algorithm 1 lies somewhere between the global model and agent 0's local model (cf. Appendix A). Each agent i can therefore verify before participating, that the other agents contribute enough data to dilute the influence of $\mathcal{S}_i$ and protect agent i's IP.

Interpretability. With this approach, the user does not learn why a certain subset of the available data sets were selected, even if they understand the selection process. Indeed, the selection is only revealed to them *after* the model has been trained.

Resilience. Like [5], Algorithm 1 does not break if users appear or disappear between subsequent rounds. When a new agent appears after the first round,

their weight should not be initialized to 1, but rather to the mean or median weight of all agents.

4 Application Case Study

We set up a collaborative ML simulation using the publicly available Titanic data set [3]. This set collects 14 features on passengers of the cruise ship Titanic, including their survival status. We train a prediction model for the survival status of passengers based on the following features: *fare*, *passenger class*, *port of embarkation*, travelling *alone or accompanied*, *sex*, *age*, and being either *adult or minor* (aged 16 or less). The pre-processing procedure is largely aligned with [12].

Implementation. A generic `Python` framework for efficient FL simulations on a single machine with `JAX` and `Haiku` [2,6] was created and made publicly available by the authors [8]. The Weight Erosion aggregation rule (Algorithm 1) is implemented in an altered version of this framework. All code (including the pre-processing) is made publicly available on GitHub [4]. Due to the small data set size, the number of agents is limited by splitting it into four subsets (agents 0–3) based on the variable *age*. The four-way split is undertaken in two ways to simulate IID and non-IID settings, as follows:

`AGE_STRICT`: Samples are strictly segregated into groups based on their age: agent 0: patients aged 0–20 years old / agent 1: 21–35 years old / agent 2: 36+ years old / agent 3: age unknown. In this data set, the *age*, *adult or minor* and some other features are strongly correlated with age. This feature skew should be sufficient to affect the conditional probability $p(y|\mathbf{x})$.

`AGE_SOME`: Agents 0 and 1 randomly partition the patients aged 0–35 years old among themselves, while agent 2 has all patients aged 36+ years old / agent 3: age unknown. This should make agents 0 and 1 more helpful for each other than agents 2 and 3.

5 Results

The collaborative training of a prediction model by four agents is simulated on a single machine for the data sets and splits detailed in Sect. 4. Each simulation is repeated four times, such that each agent serves as a user once. Every time, the user's data set is split into a test set and a training set of equal sizes, whereas 100% of the other agents' data sets are used for training. A classification model, consisting of a 2-output linear regression layer followed by log-softmax, is trained using federated SGD with three different gradient aggregation schemes:

`FedAvg:`	Train on all agents' training sets without Weight Erosion.
`Weight Erosion:`	Train on all agents' training sets with Weight Erosion.
`Local:`	Train only on the user's training set.

At each communication round, each model's accuracy is measured on the user's test set and reported (along with the weights α_i in the Weight Erosion scheme).

5.1 Predicting Survival of the Titanic with AGE_STRICT Split

In Fig. 1 (top), we can see the compensating effect of p_s at work, as agent 1's weight α_1 experiences the slowest decrease in all three graphs because agent 1 has collected one batch more than the other agents. Nevertheless, the difference between α_1 and the other agent weights is only very marked when agent 2 is the user, showing that the impact of p_d rivals that of p_s.

When splitting the Titanic data set into agents by age, we notice that the survival rate (Fig. 1 top: lower of the two red dotted lines in each graph) is fairly similar across all age groups. This implies that the *age* feature may not be particularly useful in predicting the survival of passengers. However, the survival rate is much lower among the group of passengers whose age is unknown. This could simply be a sampling bias, as it would seem easier to research the age of survivors than that of the deceased.

Another debatable feature is whether the passenger is a child (`Passenger is minor`), which is false across all data of agents 1 and 2, and unknown in the data set of agent 3. We observe that agent 0 stands out by the fact that the `Weight Erosion` model clearly out-performs the `Local` model. Nevertheless, the difference in prediction accuracy does not come from the `Local` model overfitting on the `Passenger is minor` feature, since its weights are nearly identical in both models as we see in Fig. 2 (top left, top center).

`Travelling alone` is strongly correlated with death in both models, indicating that passengers aged 0–20 years old were more likely to die when travelling alone. Notably, the same feature is weakly correlated with survival in the `FedAvg` model, as well as in the very accurate `Local` and `Weight Erosion` models trained with agent 2 as user (cf. Fig. 2: bottom left, bottom center).

In Fig. 2 (top left, top center), three features have smaller weights in the `Weight Erosion` model than in the `Local` model: `Travelling alone`, boarding in `Queenstown` (as opposed to Southampton or Cherbourg), and travelling in `Second Class`. These differences could be a sign that the local model is overfitting, especially if they apply to a small portion of the passengers, as is the case with `Queenstown` and presumably `Travelling alone`.

Interestingly, the `Weight Erosion` aggregation scheme can produce a model that is not merely a weighted average of the `FedAvg` and `Local` models. Indeed, in Fig. 2 (top), the weights of some features such as `Sex`, `Age`, or boarding in `Queenstown`, differ more between the `FedAvg` model and the `Weight Erosion` model, than they do between the `FedAvg` model and the `Local` model.

In Fig. 2 (bottom left, bottom center), we investigate the weights of the models trained with `Local` and `Weight Erosion` schemes when agent 2 is the user, since these models perform exceptionally well ($> 90\%$) despite the unexceptional survival probability of 40%. We observe that weights with absolute values < 0.25 differ between the models, without affecting their performance, while larger weights are very consistent. Further, two main features stand out: Firstly, the weights for `Sex` are spectacularly large (with absolute values between 1.1 and 1.4). Given the models' extraordinary test accuracies, we conclude that most survivors aged 36+ were female. Secondly, the weights of the `First Class` and

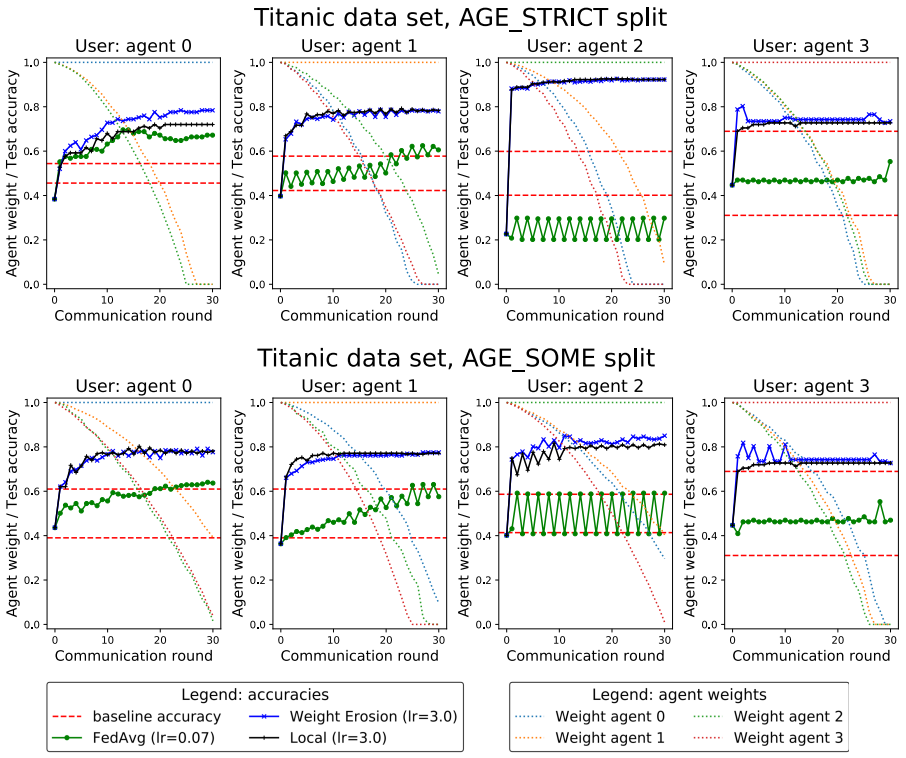

Fig. 1. Predicting survival of the Titanic. The data set is split across 4 agents by age. Full lines represent each model's accuracy on the user's test set. Displayed in red (dashed) is the accuracy obtained by predicting always 1 or always 0. The learning rates were tuned independently for each aggregation scheme (in legend: lr). Training on one batch per round and agent. $p_d = 0.01$, $p_s = 0.2$, seed = 278.
Top: `AGE_STRICT` split. Batch size: 161 (agent 1: 3 batches, others: 2 each).
Bottom: `AGE_SOME` split. Batch size: 132 (agent 3: 2 batches, others: 3 each).

`Second Class` features show that, in this age group, only First-Class passengers were much more likely to survive than Third-Class passengers.

5.2 Predicting Survival of the Titanic with `AGE_SOME` Split

As expected, we observe that agents 0 and 1, whose samples are drawn from the same age group, have the highest weight in each other's model, while their weights are similar to each other in the two other agents' models (Fig. 1, bottom). The models trained by these agents achieve the same test accuracy as with the `AGE_STRICT` split, save for the spectacular accuracy obtained for agent 2 in Fig. 1 (top).

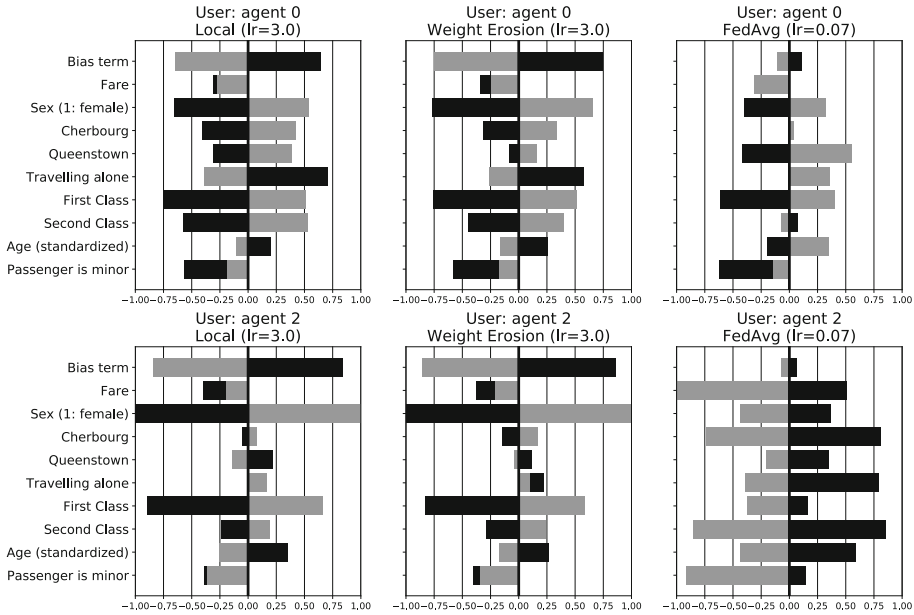

Fig. 2. Top: Parameters of the models trained in Fig. 1 (top left). Bottom: Parameters of the models trained in Fig. 1 (top, agent 2). Black: Weights for the first output of the linear layer. Gray: Weights for the second output, e.g.:
 : The feature is strongly correlated with death.
 : The feature is strongly correlated with survival.

6 Conclusion

In this paper, we introduce a novel method of model personalization in collaborative ML: Weight Erosion. Our application case study demonstrates that it can outperform two baseline schemes (FedAvg and local training), by converging to a better model that is not a linear combination of the local and global models.

Further Work. Additional refinement is needed to address in a more equitable way the under-representation of samples from larger data sets, e.g. by introducing a different weight β_i that is unrelated to α_i. Equally importantly, the Weight Erosion scheme should be tested on different ML tasks, such as image classification. For instance, the need for model personalization could arise as clinicians train a risk stratification model for COVID-19 pneumonia to identify pathological patterns in lung ultrasound images that warrant hospitalization. Personalized collaborative ML would allow them to leverage existing lung ultrasound acquired in other hospitals without being negatively affected by geographic variations in aggravating factors (such as diabetes or exposure to tuberculosis and indoor air pollution), nor by other confounding variables. Additionally, the patterns of pathology that warrant hospitalization may be assessed differently in light of local resources and thus vary between subsets of agents.

A Analyzing Weight Erosion

Let us analyze what happens in a few examples, if $\mathbf{g}_i$ are indeed the full gradients (i.e., computed on the entire set $\mathcal{S}_i$), as opposed to stochastic gradients. Let us further assume that each training loss function $\mathcal{L}_{\mathcal{S}_i}(f)$ is convex in the model parameters $\mathbf{w}$.

Example 1. *Suppose the model has just been randomly initialized to f^0 and performs very sub-optimally for both $\mathcal{S}_i$ and $\mathcal{S}_0$. Then, the normalized distance between the gradients will be very low. Formally:*

$$\begin{aligned} &\textit{Let} \quad \min\left\{\mathcal{L}_{\mathcal{S}_0}\left(f^0\right), \mathcal{L}_{\mathcal{S}_i}\left(f^0\right)\right\} \gg \max\left\{\mathcal{L}_{\mathcal{S}_0}\left(f_{\mathcal{S}_i}\right), \mathcal{L}_{\mathcal{S}_i}\left(f_{\mathcal{S}_0}\right)\right\}, \quad f_{\mathcal{S}_i} = \underset{f\in\mathcal{M}}{\arg\min}\, \mathcal{L}_{\mathcal{S}_i}(f) \\ &\Rightarrow \qquad \|\mathbf{g}_i - \mathbf{g}_0\| \ll \|\mathbf{g}_0\| \\ &\Rightarrow \qquad d_{i,0}^{rel} \ll 1 \end{aligned}$$

Therefore, both agents i and 0 are fully included at this stage of the training process, because the distance $d_{i,0}^{rel}$ is close to 0.

However, as the training process progresses and the model gradually performs better, the distance between the gradients steadily increases. We can analyze the edge cases where f is either the global model or the local model:

Example 2. *Suppose f is the global model for 2 agents, 0 and i:*

$$\begin{aligned} &\textit{Let} \qquad f = \underset{f'\in\mathcal{M}}{\arg\min}\, \mathcal{L}_{\mathcal{S}_0\cup\mathcal{S}_i}(f'), \quad \mathcal{L}_{\mathcal{S}_0\cup\mathcal{S}_i}(f) = \frac{|\mathcal{S}_0|\mathcal{L}_{\mathcal{S}_0}(f) + |\mathcal{S}_i|\mathcal{L}_{\mathcal{S}_i}(f)}{|\mathcal{S}_0| + |\mathcal{S}_i|} \\ &\Rightarrow \qquad \mathbf{0} = \nabla_{\mathbf{w}}\mathcal{L}_{\mathcal{S}_0\cup\mathcal{S}_i}(f) = \frac{|\mathcal{S}_i|}{|\mathcal{S}_0| + |\mathcal{S}_i|}\left(\frac{|\mathcal{S}_0|}{|\mathcal{S}_i|}\mathbf{g}_0 + \mathbf{g}_i\right) \\ &\Rightarrow \qquad \mathbf{g}_i = -\frac{|\mathcal{S}_0|}{|\mathcal{S}_i|}\mathbf{g}_0 \\ &\Rightarrow \qquad d_{i,0}^{rel} = \frac{\|\mathbf{g}_i - \mathbf{g}_0\|}{\|\mathbf{g}_0\|} = \frac{\left\|\left(-\frac{|\mathcal{S}_0|}{|\mathcal{S}_i|} - 1\right)\mathbf{g}_0\right\|}{\|\mathbf{g}_0\|} = 1 + \frac{|\mathcal{S}_0|}{|\mathcal{S}_i|} \end{aligned}$$

In this case, whether (and to which degree) agent i should participate in the training depends on the sizes $|\mathcal{S}_0|$ and $|\mathcal{S}_i|$: Indeed, if $|\mathcal{S}_0| \ll |\mathcal{S}_i|$, then $d_{i,0}^{rel} \approx 1$, indicating that it would be useful to incorporate agent i further in the training process. This is sensible, considering that the much larger number of samples in $\mathcal{S}_i$ could help reduce the generalization error substantially. Inversely, if $|\mathcal{S}_0| \gg |\mathcal{S}_i|$, then it is not useful to include $\mathcal{S}_i$ in training and we are better off only using the (much more numerous) samples collected by agent 0. Correspondingly, this leads to $d_{i,0}^{rel} \gg 1$.

Example 3. *Suppose now that f is the local model of agent 0:*

$$\begin{aligned} &\textit{Let} \qquad f = \underset{f'\in\mathcal{M}}{\arg\min}\, \mathcal{L}_{\mathcal{S}_0}(f') \\ &\Rightarrow \qquad \mathbf{g}_0 = \nabla_{\mathbf{w}}\mathcal{L}_{\mathcal{S}_0}(f) = \mathbf{0} \\ &\Rightarrow \quad \forall i \in \mathcal{U} : d_{i,0}^{rel} \textit{ undefined } (+\infty) \end{aligned}$$

Under the stated assumptions of convexity and full gradient descent, all distances $d_{i,0}^{rel}$ *grow without an upper bound as the trained model approaches the local model. In other words, when we move too close to the local model, all data sets start to appear very different from* $\mathcal{S}_0$*. Consequently, using any decreasing function of* $d_{i,0}^{rel}$ *as a similarity metric results in the danger of converging to the local model. One possible strategy to prevent this is to stop training before convergence. Intuitively, training should not be stopped as long as the gradients fit into a D-dimensional cone*[3]*.*

References

1. Blanchard, P., Guerraoui, R., Stainer, J., et al.: Machine learning with adversaries: byzantine tolerant gradient descent. In: Advances in Neural Information Processing Systems, pp. 119–129 (2017)
2. Bradbury, J., et al.: JAX: composable transformations of Python+NumPy programs (2018). http://github.com/google/jax
3. Harrell Jr., F.E.: Titanic dataset, October 2017. https://www.openml.org/d/40945
4. Grimberg, F., Jaggi, M.: Semester project on private and personalized ml (2020). https://github.com/epfl-iglobalhealth/coML-Personalized-v2
5. He, L., Karimireddy, S.P., Jaggi, M.: Secure byzantine-robust machine learning. arXiv:2006.04747 [cs, stat] (2020). http://arxiv.org/abs/2006.04747
6. Hennigan, T., Cai, T., Norman, T., Babuschkin, I.: Haiku: Sonnet for JAX (2020). http://github.com/deepmind/dm-haiku
7. Kairouz, P., McMahan, H.B., Avent, B., et al.: Advances and open problems in federated learning. arXiv:1912.04977 [cs, stat] (2019). http://arxiv.org/abs/1912.04977
8. Karimireddy, S.P.: Jax federated learning (2020). https://tinyurl.com/Karimireddy-Jax-FL
9. Mansour, Y., Mohri, M., Ro, J., Suresh, A.T.: Three approaches for personalization with applications to federated learning. arXiv:2002.10619 [cs, stat] (2020). http://arxiv.org/abs/2002.10619
10. Mansour, Y., Mohri, M., Rostamizadeh, A.: Domain adaptation: learning bounds and algorithms. arXiv preprint arXiv:0902.3430 (2009)
11. McMahan, H.B., Moore, E., Ramage, D., Hampson, S., Agüera y Arcas, B.: Communication-efficient learning of deep networks from decentralized data. In: Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS). http://arxiv.org/abs/1602.05629 (2016)
12. MNassri, B.: Titanic: logistic regression with python (2020). https://www.kaggle.com/mnassrib/titanic-logistic-regression-with-python?scriptVersionId=26445092
13. Pillutla, K., Kakade, S.M., Harchaoui, Z.: Robust aggregation for federated learning. arXiv preprint arXiv:1912.13445 (2019)
14. Smith, V., Chiang, C.K., Sanjabi, M., Talwalkar, A.S.: Federated multi-task learning. In: Advances in Neural Information Processing Systems, pp. 4424–4434 (2017)

[3] If all gradients fit into a cone, then it is possible to improve the loss on all data sets by taking a step in the opposite direction of the axis of the cone.