FEDERATED LEARNING CAN FIND FRIENDS THAT ARE ADVANTAGEOUS

Anonymous authors
Paper under double-blind review

ABSTRACT

In Federated Learning (FL), the distributed nature and heterogeneity of client data present both opportunities and challenges. While collaboration among clients can significantly enhance the learning process, not all collaborations are beneficial; some may even be detrimental. In this study, we introduce a novel algorithm that assigns adaptive aggregation weights to clients participating in FL training, identifying those with data distributions most conducive to a specific learning objective. We demonstrate that our aggregation method converges no worse than the method that aggregates only the updates received from clients with the same data distribution. Furthermore, empirical evaluations consistently reveal that collaborations guided by our algorithm outperform traditional FL approaches. This underscores the critical role of judicious client selection and lays the foundation for more streamlined and effective FL implementations in the coming years.

1 INTRODUCTION

Federated Learning (FL) introduces an innovative paradigm redefining traditional machine learning workflow. Instead of centrally pooling sensitive client data, FL allows for model training on decentralized data sources stored directly on client devices (Konečný et al., 2016; Zhang et al., 2021; Li et al., 2020a; Beznosikov et al., 2021). In this approach, rather than training Machine Learning (ML) models in a centralized manner, a shared model is distributed to all clients. Each client then performs local training, and model updates are exchanged between clients and the FL orchestrator (often referred to as the master server) (McMahan et al., 2017; Shokri & Shmatikov, 2015; Karimireddy et al., 2020).

Personalized Federated Learning (PFL). The concept of PFL (Collins et al., 2021; Hanzely et al., 2020; Sadiev et al., 2022; Almansoori et al., 2024; Borodich et al., 2021; Sadiev et al., 2022) has been gaining traction. In this framework, each client, often referred to as an agent, takes part in developing their own personalized model variant. This tailored training approach leverages local data distributions, aiming to design models that cater to the distinct attributes of each client's dataset (Fallah et al., 2020). In contrast, standard `Parallel SGD` (Zinkevich et al., 2010) often leads to models that generalize across all clients rather than personalize to the specific data distributions and unique characteristics of individual clients, potentially resulting in suboptimal performance on personalized tasks. However, a prominent challenge arises in this decentralized training landscape due to the data's non-IID (independent and identically distributed) nature across various clients. Data distributions that differ considerably can have a pronounced impact on the convergence and generalization capabilities of the trained models. While certain client-specific data distributions might strengthen model performance, others could prove detrimental, introducing biases or potential adversarial patterns. Additionally, within the personalized federated learning paradigm, the emphasis on crafting individualized models could inadvertently heighten these data disparities (Kairouz et al., 2021). Consequently, this may lead to models that deliver subpar or, in some cases, incorrect results when applied to wider or diverse datasets (Kulkarni et al., 2020).

Collaboration as a service. In this paper, we introduce a modified protocol for FL that deviates from a strictly personalized approach. Rather than focusing solely on refining individualized models, our approach seeks to harness the advantages of distinct data distributions, curb the detrimental effects of outlier clients, and promote collaborative learning. Through this innovative training mechanism, our algorithm discerns which clients are optimal collaborators to ensure faster convergence and potentially better generalization.

1.1 Setup

We assume that there are n clients participating in the training and consider the first one as a target client. The goal is to train the model for this client, i.e., we consider

$$\min_{x\in\mathbb{R}^d} \left\{f(x) \equiv f_1(x) := \mathbb{E}_{\xi_1\sim\mathcal{D}_1}[f_{\xi_1}(x)]\right\}, \tag{1}$$

where $f_{\xi_1} : \mathbb{R}^d \to \mathbb{R}$ is the loss function on sample ξ_1 and $f : \mathbb{R}^d \to \mathbb{R}$ is an expected loss. Other clients can also have data sampled from similar distributions, but we also allow adversarial participants, e.g., Byzantines (Lamport et al., 1982; Lyu et al., 2020). That is, some clients can be beneficial for the training in certain stages, but they are not assumed to be known apriori.

The considered target client scenario naturally arises in FL on medical image data. In such applications, different hospitals naturally have different data distributions (e.g., due to the differences in the equipment). Therefore, the data coming from one clinic can be useless to another clinic. At the same time, several clinics can have similar data distributions.

1.2 Contribution

Our main contributions are listed below.

- **New method: `MeritFed`.** We proposed a new method called Merit-based Federated Averaging for Diverse Datasets (`MeritFed`) that aims to solve (1). The key idea is to use the stochastic gradients received from the clients to adjust the weights of averaging through the inexact solving of the auxiliary problem of minimizing a validation loss as a function of aggregation weights.
- **Provable convergence under mild assumptions.** We prove that `MeritFed` converges not worse than `SGD` that averages only the stochastic gradients received from clients having the same data distribution (these clients are not known apriori) for smooth non-convex and Polyak-Lojasiewicz functions under standard bounded variance assumption.
- **Utilizing all possible benefits.** We numerically show that `MeritFed` can even benefit from collaboration with clients having different data distributions when these distributions are close to the target one. That is, `MeritFed` automatically detects beneficial clients at any stage of training. Moreover, we illustrate the Byzantine robustness of the proposed method even when Byzantine workers form a majority.

1.3 Related work

Federated optimization. Standard results in distributed/federated optimization focus on the problem:

$$\min_{x\in\mathbb{R}^d} \frac{1}{n}\sum_{i=1}^n f_i(x), \tag{2}$$

where $f_i(x)$ represents either expected or empirical loss on the client i. This problem significantly differs from (1), since one cannot completely ignore the updates from some clients to achieve a better solution. Typically, in this case, communication is the main bottleneck of the methods for solving such problems. To address this issue one can use communication compression (Alistarh et al., 2017; Stich et al., 2018; Mishchenko et al., 2019), local steps (Stich, 2018; Khaled et al., 2020; Kairouz et al., 2021; Wang et al., 2021; Mishchenko et al., 2022; Sadiev et al., 2022; Beznosikov et al., 2024), client importance sampling (Cho et al., 2020; Nguyen et al., 2020; Ribero & Vikalo, 2020; Lai et al., 2021; Luo et al., 2022; Chen et al., 2022d), or decentralized protocols (Lian et al., 2017; Song et al., 2022), or FL of graph neural network on graph data Tan et al. (2023). However, these techniques are orthogonal to what we focus on in our paper, though incorporating them into our algorithm is a prominent direction for future research.

Clustered FL. Another way of utilizing benefits from the other clients is the clustering of clients based on some information about their data or personalized models. Tang et al. (2021) propose a personalized formulation with ℓ_2-regularization that attracts a personalized model of a worker to the center of the cluster that this worker belongs to. A similar objective is studied by Ma et al. (2022). Ghosh et al. (2020) develop an algorithm that updates clusters's centers using the gradients of those clients that have the smallest loss functions at the considered cluster's center. It is worth mentioning that, in contrast to our work, the mentioned works modify the personalized objective to illustrate some benefits of collaboration while we focus on the pure personalized problem of the target client. Under the assumption that the data distributions of each client are mixtures of some finite set of underlying distributions, Marfoq et al. (2021) derive the convergence result for the Federated Expectation-Maximization algorithm. This is the closest work to our setup in the Clustered FL literature. However, in contrast to (Marfoq et al., 2021), we do not assume that the gradients are bounded and that the local loss functions have bounded gradient dissimilarity. Another close work to ours is (Fraboni et al., 2021), where the authors consider so-called clustered-based sampling. However, Fraboni et al. (2021) also make a non-standard assumption on the bounded dissimilarity of the local loss functions, while one of the key properties of our approach is its robustness to arbitrary clients' heterogeneity. (Li et al., 2020b) is also a relevant paper in the sense that not all workers are selected for aggregation at each communication round (due to the client sampling). However, this work focuses on weighted empirical risk minimization (with weights proportional to the dataset size), i.e., Li et al. (2020b) consider a different problem. Ma et al. (2023) addresses the "clustering collapse" issue with clustering rules based on the min-loss criterion and k-means style criterion. Bao et al. (2023) focus on optimizing collaboration in federated learning by grouping workers into clusters based on data similarity. Their method requires minimizing a score function for each pair of clients to measure the distance between their data. This clustering process involves computational efforts during the preprocessing stage, and the training within each cluster uses static aggregation weights.

Non-uniform averaging. There are also works studying the convergence of distributed `SGD`-type methods that use non-uniform (but fixed) weights of averaging. Ding & Wang (2022) propose a method to detect collaboration partners and adaptively learn "several" models for numerous heterogeneous clients. Directed graph edge weights are used to calculate group partitioning. Since the calculation of optimal weights in their approach is based on similarity measures between clients' data, it is unclear how to compute them in practice without sacrificing the users's data privacy. Even et al. (2022) develop and analyze another approach for personalized aggregation, where each client filters gradients and aggregates them using fixed weights. The optimal weights also require estimating the distance between distributions (or communicating empirical means among all clients and estimating effective dimensions). Both works do not consider weights evolving in time, which is one of the key features of our method.

Non-fixed weights are considered in (Wu & Wang, 2021), but the authors focus on non-personalized problem formulation. In particular, Wu & Wang (2021) propose the method called `FedAdp` that uses cosine similarity between gradients and the Gompertz function for updating aggregation weights. Under the strong bounded local gradient dissimilarity assumption[1], Wu & Wang (2021) derive a non-conventional upper bound (for the loss function at the last iterate of their algorithm) that does not necessarily imply convergence of the method. Zhang et al. (2020) introduce `FedFomo` that uses additional data to adjust the weights of aggregation in Federated Averaging. In this context, `FedFomo` is close to `MeritFed`. However, the weights selection formulas significantly differ from ours. In particular, Zhang et al. (2020) do not relate the proposed weights with the minimization problem from Line 7 of our method. In addition, there is no theoretical convergence analysis of `FedFomo`.

Bi-level optimization. Taking into account that we want to solve problem (1) using the information coming from not only the target client, it is natural to consider the following bi-level optimization (BLO) problem formulation:

$$\min_{w\in\Delta_1^n} \quad f(x^*(w)), \tag{3}$$

$$\text{s.t.} \quad x^*(w) \in \arg\min_{x\in\mathbb{R}^d} \textstyle\sum_{i=1}^n w_i f_i(x), \tag{4}$$

where Δ_1^n is a unit simplex in $\mathbb{R}^n$: $\Delta_1^n = \{w \in \mathbb{R}^n \mid \sum_{i=1}^n w_i = 1,\ w_i \geq 0\ \forall i \in [n]\}$. The problem in (3) is usually called the upper-level problem (UL), while the problem in (4) is the lower-level (LL) one. Since in our case $f(x) \equiv f_1(x)$, (3)-(4) is equivalent to (1). In the general case, this equivalence does not always hold and, in addition, function f is allowed to depend on w not only through x^*. All these factors make the general BLO problem hard to solve. The literature for this general class of problems is quite rich, and we cover only closely related works.

[1]Wu & Wang (2021) assume that there exist constants $A, B > 0$ such that $A\|\nabla f(x)\| \leq \|\nabla f_i(x)\| \leq B\|\nabla f(x)\|$ for every client $i \in [n]$ and any x, where $f(x) = \frac{1}{n}\sum_{i=1}^n f_i(x)$.

Algorithm 1 `MeritFed`: Merit-based Federated Learning for Diverse Datasets

1: **Input:** Starting point $x^0 \in \mathbb{R}^d$, stepsize $\gamma > 0$
2: **for** $t = 0, \dots$ **do**
3: server sends x^t to each worker
4: **for all workers** $i = 1, \dots, n$ **in parallel do**
5: compute stochastic gradient $g_i(x^t, \boldsymbol{\xi}_i)$ from local data and **send** $g_i(x^t, \boldsymbol{\xi}_i)$ to the server
6: **end for**
7: $w^{t+1} \approx \underset{w \in \Delta_1^n}{\arg\min} f\left(x^t - \gamma \sum\limits_{i=1}^n w_i g_i(x^t, \boldsymbol{\xi}_i)\right)$
8: $x^{t+1} = x^t - \gamma \sum_{i=1}^n w_i^{t+1} g_i(x^t, \boldsymbol{\xi}_i)$.
9: **end for**

The closest works to ours are (Chen et al., 2021a), which propose so-called Target-Aware Weighted Training (`TAWT`), and its extension to the federated setup (Huang et al., 2022). Their analysis relies on the existence of weights w, such that $\mathrm{dist}(\sum_{i=1}^n w_i \mathcal{D}_i, \mathcal{D}_{\text{target}}) = 0$ in terms so-called representation-based distance (Chen et al., 2021a), which is also zero in our case, or existence of identical neighbors. However, the analysis is based on BLO's techniques and requires a hypergradient estimation, i.e., $\nabla_w f(x^*(w), w)$, which is usually hard to compute. To avoid the hypergradient calculation, (Chen et al., 2021a) also propose a heuristic based on the usage of cosine similarity between the clients' gradients, which makes the implementation of the algorithm similar to `FedAdp` (Wu & Wang, 2021).

In fact, there are two major difficulties in estimating hypergradient. The first one is that the optimal solution $x^*(w)$ of the lower problem for every given w needs to be estimated. The known approaches iteratively update the lower variable x multiple times before updating w, which causes high communication costs in a distributed setup. A lot of methods (Ghadimi & Wang, 2018; Hong et al., 2020; Chen et al., 2021b; Ji et al., 2021; 2022) are proposed to effectively estimate $x^*(w)$ before updating w, but anyway the less precise estimate slowdowns the convergence. The second obstacle is that hypergradient calculation requires second-order derivatives of $f_i(w, x)$. Many existing methods (Chen et al., 2022c; Dagréou et al., 2022) use an explicit second-order derivation of $f_i(w, x)$ with a major focus on efficiently estimating its Jacobian and inverse Hessian, which is computationally expensive itself, but also dramatically increases the communication cost in a distributed setup. A number of methods (Chen et al., 2022c; Li et al., 2022; Dagréou et al., 2022) avoid directly estimating its second-order computation and only use the first-order information of both upper and lower objectives, but they still have high communication costs and do not exploit our assumptions. For a more detailed review of BLO, we refer to (Zhang et al., 2023; Liu et al., 2021; Chen et al., 2022a).

2 MeritFed: Merit-Based Federated Learning For Diverse Datasets

Recall that the primary objective the target client seeks to solve is given by (1) where n workers are connected with a parameter-server. Standard Parallel `SGD`

$$x^{t+1} = x^t - \tfrac{\gamma}{n} \sum_{i=1}^n g_i(x^t, \boldsymbol{\xi}_i), \tag{5}$$

where $g_i(x^t, \boldsymbol{\xi}_i)$ denotes a stochastic gradient (unbiased estimate of $\nabla f_i(x^t)$) received from client i, cannot solve problem (1) in general, since workers $\{2, \dots, n\}$ do not necessarily have the same data distribution as the target client. This issue can be solved if we modify the method as follows:

$$x^{t+1} = x^t - \tfrac{\gamma}{|\mathcal{G}|} \sum_{i \in \mathcal{G}} g_i(x^t, \boldsymbol{\xi}_i), \tag{6}$$

where $\mathcal{G}$ denotes the set of workers that have the same data distribution as the target worker. However, the group $\mathcal{G}$ is not known in advance. This aspect makes the method from (6) impractical. Moreover, this method ignores potentially useful vectors received from the workers having different yet similar data distributions.

2.1 The Proposed Method

We develop Merit-based Federated Learning for Diverse Datasets (`MeritFed`; see Algorithm 1) aimed at solving (1) and safely gathering all potential benefits from collaboration with other clients.

As in Parallel SGD all clients are required to send the stochastic gradients to the server. However, in contrast to uniform averaging of the received stochastic gradients, MeritFed uses the weights w^t from the unit simplex Δ_1^n that are updated at each iteration. In particular, the new vector of weights $w^{t+1} \in \mathbb{R}^n$ at iteration t approximates $\arg\min_{w\in\Delta_1^n} f(x^t - \gamma \sum_{i=1}^n w_i g_i(x^t, \boldsymbol{\xi}_i))$. Then, the server uses the obtained weights for averaging the stochastic gradients and updating x^t.

2.2 Auxiliary Problem in Line 7

In general, solving the problem in Line 7 is not easier than solving the original problem (1). Therefore, we present three particular approaches for efficiently addressing this problem.

Approach 1: use fresh data. Let us assume that the target client can obtain new samples from distribution $\mathcal{D}_1$ at any moment in time. To avoid any risk of compromising clients' privacy, the target client dataset should be stored only on the target client, and stochastic gradients received from other clients cannot be directly sent to the target client. To satisfy these requirements, one can approximate

$$\arg\min_{w\in\Delta_1^n} \left\{\varphi_t(w) \equiv f\left(x^t - \gamma \textstyle\sum_{i=1}^n w_i g_i(x^t, \boldsymbol{\xi}_i)\right)\right\} \tag{7}$$

using *zeroth-order*[2] Mirror Descent (or its accelerated version) (Duchi et al., 2015; Shamir, 2017; Gasnikov et al., 2022b):

$$w^{k+1} = \arg\min_{w\in\Delta_1^n} \left\{\alpha\langle \tilde{g}^k, w\rangle + D_r(w, w^k)\right\}, \tag{8}$$

where $\alpha > 0$ is the stepsize, $\tilde{g}^k$ is a finite-difference approximation of the directional derivative of sampled function

$$\varphi_{t,\xi^k}(w) \overset{\text{def}}{=} f_{\xi^k}\left(x^t - \gamma \textstyle\sum_{i=1}^n w_i g_i(x^t, \boldsymbol{\xi}_i)\right), \tag{9}$$

where ξ^k is a fresh sample from the distribution $\mathcal{D}_1$ independent from all previous steps of the method, e.g., one can use $\tilde{g}^k = \frac{n(\varphi_{t,\xi^k}(w^k+he) - \varphi_{t,\xi^k}(w^k-he))}{2h}$ for $h > 0$ and e being sampled from the uniform distribution on the unit Euclidean sphere, and $D_r(w, w^k) = r(w) - r(w^k) - \langle \nabla r(w^k), w - w^k\rangle$ is the Bregman divergence associated with a 1-strongly convex function r. Although, typically, the oracle complexity bounds for gradient-free methods have $\mathcal{O}(n)$ dependence on the problem dimension (Gasnikov et al., 2022a), one can get just $\mathcal{O}(\log^2(n))$, in the case of the optimization over the probability simplex (Shamir, 2017; Gasnikov et al., 2022b). More precisely, if f is M_2-Lipschitz w.r.t. ℓ_2-norm and convex, then one can achieve $\mathbb{E}[\varphi_t(w) - \varphi_t(w^*)] \le \delta$ using $\mathcal{O}(M_2^2 \log^2(n)/\delta^2)$ computations of φ, where R is ℓ_1-distance between the starting point and the solution (Gasnikov et al., 2022b) and prox-function $r(w) = \sum_{i=1}^n w_i \log(w_i)$, which is 1-strongly convex w.r.t. ℓ_1-norm.

Approach 2: use additional validation data. Alternatively, one can assume that the target client has an additional validation dataset $\widehat{D}$ sampled from $\mathcal{D}_1$. Then, instead of function f in Line 7, one can approximately minimize

$$\widehat{f}(x) = \tfrac{1}{|\widehat{D}|} \textstyle\sum_{\xi\in\widehat{D}} f_\xi(x), \tag{10}$$

which under certain conditions provably approximates the original function $f(x)$ with any predefined accuracy if the dataset $\widehat{D}$ is sufficiently large (Shalev-Shwartz et al., 2009; Feldman & Vondrak, 2019). More precisely, the worst-case guarantees (e.g., (Liu & Tong, 2024)) imply that to guarantee $\mathbb{E}[f(\widehat{x}^*) - f(x^*)] \le \delta$, where $\widehat{x}^* \in \arg\min_{x\in\mathbb{R}^d} \widehat{f}(x)$ and $x^* \in \arg\min_{x\in\mathbb{R}^d} f(x)$, the validation dataset should be of the size $|\widehat{\mathcal{D}}| \sim \max\{L/\mu, 1/\mu\delta\}$ under the assumption that $f_\xi(x)$ is μ-strongly convex. However, as we observe in our experiments, MeritFed works well even with a relatively small size of the validation dataset for non-convex problems.

Approach 3: use training data. This approach utilizes the existing training dataset and replaces f in Line 7 with the training loss function. This method leverages the training data directly to validate the model, allowing the model to be evaluated against the same dataset it was trained on. This approach is particularly effective when data is limited or when acquiring additional datasets is not feasible. Moreover, in our experiments, this approach works not worse than the above ones.

[2]In this case, the server can ask the target client to evaluate loss values at the required points without sending the stochastic gradients received from other workers.

Memory usage. It is also worth mentioning that `MeritFed` requires the server to store n vectors at each iteration for solving the problem in Line 7. While standard `SGD` does not require such a memory, closely related methods — `FedAdp` and `TAWT` — also require the server to store n vectors for the computation of the weights for aggregation. However, for modern servers, this is not an issue.

3 Convergence Analysis

In our analysis, we rely on the standard assumptions for non-convex optimization literature.

Assumption 3.1. *For all $i \in \mathcal{G}$ the stochastic gradient $g_i(x, \boldsymbol{\xi}_i)$ is an unbiased estimator of $\nabla f_i(x)$ with bounded variance, i.e., $\mathbb{E}_{\boldsymbol{\xi}_i}[g_i(x, \boldsymbol{\xi}_i)] = \nabla f_i(x)$ and for some $\sigma \geq 0$*

$$\mathbb{E}_{\boldsymbol{\xi}_i} \|g_i(x, \boldsymbol{\xi}_i) - \nabla f_i(x)\|^2 \leq \sigma^2. \tag{11}$$

The above assumption is known as the bounded variance assumption. It is classical for the analysis of stochastic optimization methods, e.g., see (Nemirovski et al., 2009; Juditsky et al., 2011).

Next, we assume the smoothness of the objective.

Assumption 3.2. *f is L-smooth, i.e., $\forall\, x, y \in \mathbb{R}^d$*

$$f(x) \leq f(y) + \langle \nabla f(y), x - y \rangle + \tfrac{L}{2}\|x - y\|^2. \tag{Lip}$$

We also make the following (optional) assumption called Polyak-Łojasiewicz (PŁ) condition (Polyak, 1963; Lojasiewicz, 1963).

Assumption 3.3. *f satisfies Polyak-Łojasiewicz (PŁ) condition with parameter μ, i.e., for $\mu \geq 0$*

$$f^* \geq f(x) - \tfrac{1}{2\mu}\|\nabla f(x)\|^2, \quad \forall\, x \in \mathbb{R}^d. \tag{PL}$$

This assumption belongs to the class of structured non-convexity assumptions allowing linear convergence for first-order methods such as Gradient Descent (Necoara et al., 2019).

The main result for `MeritFed` is given below (see the proof in Appendix B).

Theorem 3.4. *Let Assumptions 3.1 and 3.2 hold. Then after T iterations, `MeritFed` with $\gamma \leq \frac{1}{2L}$ outputs x^i, $i = 0, \cdots, T-1$ such that*

$$\tfrac{1}{T}\textstyle\sum_{t=0}^{T-1} \mathbb{E}\|\nabla f(x^t)\|^2 \leq \frac{2\left(f\left(x^0\right) - f(x^*)\right)}{T\gamma} + \frac{2\sigma^2\gamma L}{G} + \frac{2\delta}{\gamma},$$

where δ is the accuracy of solving the problem in Line 7 and $G = |\mathcal{G}|$. Moreover if Assumption 3.3 additionally holds, then after T iterations of `MeritFed` with $\gamma \leq \frac{1}{2L}$ outputs x^T such that

$$\mathbb{E} f\left(x^T\right) - f^* \leq (1 - \gamma\mu)^T \left(f\left(x^0\right) - f^*\right) + \frac{\sigma^2\gamma L}{\mu G} + \frac{\delta}{\gamma\mu}.$$

If δ is sufficiently small, then the above result matches the known results for Parallel `SGD` (Ghadimi & Lan, 2013; Karimi et al., 2016; Khaled & Richtárik, 2022) that uniformly averages only the workers from the group $\mathcal{G}$, i.e., those workers that have data distribution $\mathcal{D}_1$ (see the method in (6)). More precisely, we see a linear speed-up of $1/G$ in the obtained convergence rates. However, `MeritFed` does not require knowing which workers share the same distribution. Moreover, as our numerical experiments show, `MeritFed` can converge even better when there exist workers with distinct yet close data distributions, and it is not necessary to solve the problem in Line 7 with high precision.

4 Experiments

Since the literature on FL is very rich, we focus only on the closely related methods, i.e., the methods that satisfy two criteria: (i) they solve the same problem as we consider in our work 1, and (ii) have theoretical convergence guarantees. That is, we evaluate the performance of proposed methods in comparison with `FedAdp` (Wu & Wang, 2021), `TAWT` (Chen et al., 2021a), and `FedProx` (Li et al., 2020b) (`FedProx` reduces to `FedAvg` if there are no local steps, that is the setup for

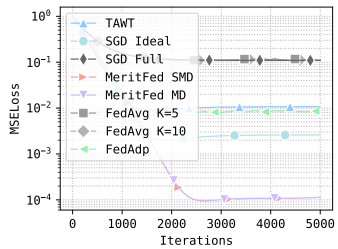

Figure 1: Mean Estimation: $\mu = 0.001$, MD learning rate = 3.5.

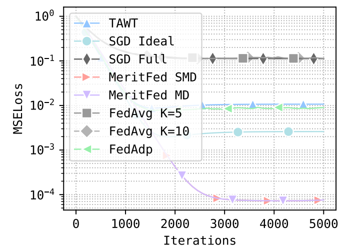

Figure 2: Mean Estimation: $\mu = 0.01$, MD learning rate = 4.5.

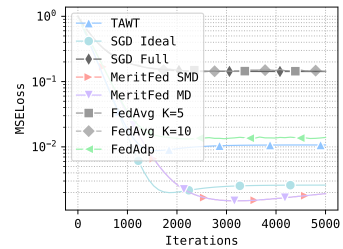

Figure 3: Mean Estimation: $\mu = 0.1$, MD learning rate = 12.5.

MeritFed). We also compare standard SGD with uniform weights (labeled as SGD Full[3]), SGD that accumulates only gradients from clients with the target distribution (SGD Ideal) and two versions of our algorithm. The first one, labeled as MeritFed SMD, samples gradient for the Mirror Descent subroutine in contrast to the other one, labeled as MeritFed MD, that uses the full dataset (additional or train) to calculate gradient for Mirror Descent step. We use only 10 Mirror Descent steps for solving the auxiliary problem from Line 7 since it was sufficient to achieve good enough results in our experiments. In addition, we present the evolution of weighs (if applicable) using heatmap plots. In the main text, we show the results for the case when the additional validation dataset is available for the problem in Line 7. Additional experiments with the usage of train data for the problem in Line 7, with the presence of Byzantine participants and with more workers, are provided in the appendix. Our code is available at https://anonymous.4open.science/r/86315. We use a cluster with the following hardware: AMD EPYC 7552 48-Core CPU, 512GiB RAM, NVIDIA A100 80GB GPU, 200Gb storage space.

Mean estimation. We start with the mean estimation problem, i.e., finding such a vector that minimizes the mean squared distance to the data samples. More formally, the goal is to solve

$$\min_{x\in\mathbb{R}^d} \mathbb{E}_{\xi\sim\mathcal{D}_1}\|x-\xi\|^2,$$

that has the optimum at $x^* = \mathbb{E}_{\xi\sim\mathcal{D}_1}[\xi]$. We consider $\mathcal{D}_1 = \mathcal{N}(0, \boldsymbol{I})$ and also two other distributions from where some clients also get samples: $\mathcal{D}_2 = \mathcal{N}(\mu\mathbf{1}, \boldsymbol{I})$ and $\mathcal{D}_3 = \mathcal{N}(e, \boldsymbol{I})$, where $\mathbf{1} = (1, 1, \ldots, 1)^\top \in \mathbb{R}^d$, $\mu > 0$ is a parameter, and e is some vector that we obtain in advance via sampling uniformly at random from the unit Euclidean sphere. We consider 150 clients with data distributed as follows: the first 5 workers have data from $\mathcal{D}_1$ (the first group of clients), the next 95 workers have data from $\mathcal{D}_2$ (the second group of clients), and the remaining 50 clients have data from $\mathcal{D}_3$ (the third group of clients). Each client has 1000 samples from the corresponding distribution, and the target client has additional 1000 samples for validation, i.e., for solving the problem in Line 7. The dimension of the problem is $d = 10$. Parameters that are the same for all experiments: number of peers $= 150$, number of samples $= 1000$, batch size $= 100$, learning rate $= 0.01$, number of steps for Mirror Descent $= 50$. For FedAvg, the number of sampled clients K is chosen from the set $\{5, 10\}$.

We consider three cases: $\mu = 0.001, 0.01, 0.1$. The smaller μ is, the closer $\mathcal{D}_2$ is to $\mathcal{D}_1$ and, thus, the more beneficial the samples from the second group are. Therefore, for small μ, we expect to see that MeritFed outperforms SGD Ideal. Moreover, since the workers from the third group have quite different data distribution, SGD Full is expected to work worse than other baselines.

The results are presented in Figures 1-3. They fit the described intuition and our theory well: the workers from the second group are beneficial (since their distributions are close enough to the distribution of the target client). Indeed, MeritFed achieves better optimization error (due to the smaller variance because of the averaging with more workers). However, when the dissimilarity between distributions is large the second group becomes less useful for the training, and MeritFed has comparable performance to SGD Ideal and consistently outperforms other methods.

Image classification: CIFAR10 + ResNet18. This part is devoted to image classification on the CIFAR10 (Krizhevsky et al., 2009) dataset using ResNet18 (He et al., 2016) model and cross-entropy loss. We consider 20 clients with data distributed as follows: the first worker has data from $\mathcal{D}_1$ (the first group of clients), the next 10 workers have data from $\mathcal{D}_2$ (the second group of clients), and the

[3] Although, FedProx and SGD Full are designed for standard empirical risk minimization, we consider these methods as standard baselines.

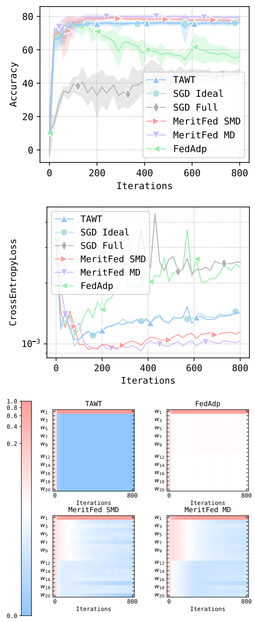

Figure 4: CIFAR10 (extra val.): $\alpha = 0.5$

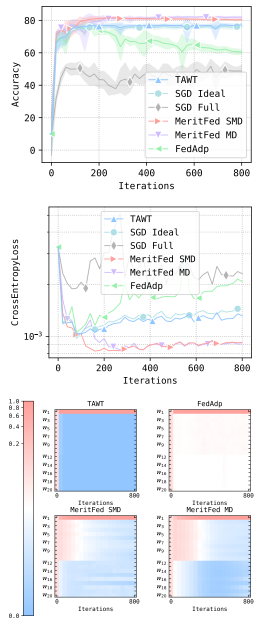

Figure 5: CIFAR10 (extra val.): $\alpha = 0.7$

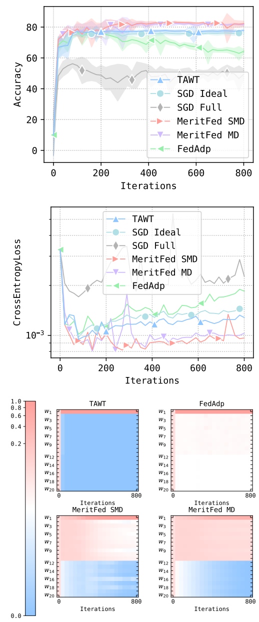

Figure 6: CIFAR10 (extra val.): $\alpha = 0.9$

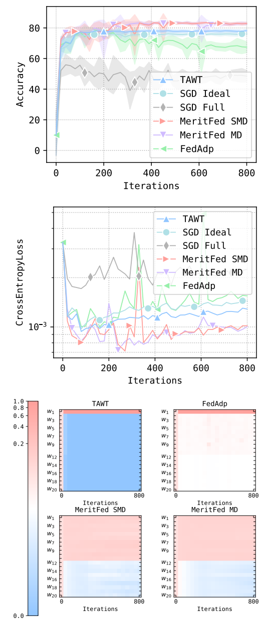

Figure 7: CIFAR10 (extra val.): $\alpha = 0.99$.

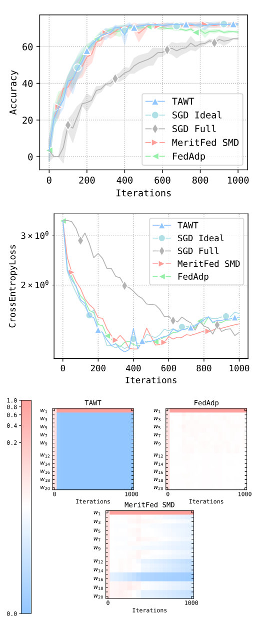

Figure 8: GoEmotions (extra val.): $\alpha = 0.5$

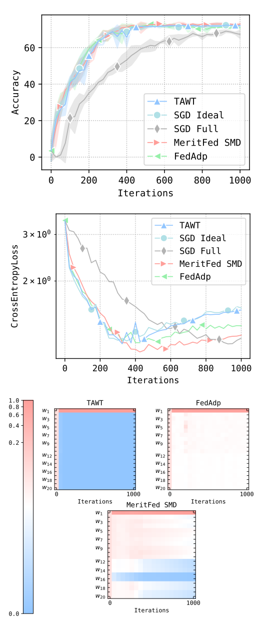

Figure 9: GoEmotions (extra val.): $\alpha = 0.7$

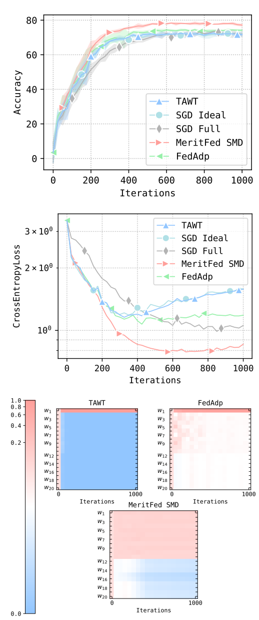

Figure 10: GoEmotions (extra val.): $\alpha = 0.9$

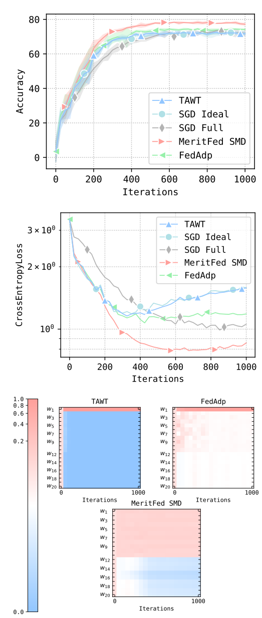

Figure 11: GoEmotions (extra val.): $\alpha = 0.99$

remaining 9 clients have data from $\mathcal{D}_3$ (the third group of clients). Specifically, the target client's objective is to classify the first three classes: 0, 1, and 2. This client possesses data with these three labels. The following ten workers (second group) also have datasets where a proportion, denoted by $\alpha \in (0, 1]$, consists of classes from the set $0, 1, 2$, while the remaining $1 - \alpha$ portion includes classes from the set $3, 4, 5$. The remaining clients (third group) have data from the rest, e.g., $6, 7, 8, 9$ labeled. The data is randomly distributed among clients without overlaps, adhering to the aforementioned label restrictions. For `MeritFed` each worker calculates stochastic gradient using a batch size of 75; then the server performs 10 steps of Mirror Descent (or its stochastic version) with a batch-size of 90 (in case of stochastic version) and a learning rate of 0.1 to update weights of aggregation, and then performs a model parameters update with a learning rate of 0.01. We normalize images (similarly to (Horváth & Richtárik, 2020)). Since an additional validation dataset can be used by MeritFed, we cut 300 samples of each target class $(0, 1, 2)$ off from the test data. Accuracy and loss are calculated on the rest of the test data, including labels 0, 1, and 2, modeling the case when the target client aims to classify samples with these labels.

The results are provided in Figures 4-7, where we show how accuracy and cross-entropy loss change for different methods and different values of α, which measures the similarity between data distributions of the target client and the second group of clients, and the evolution of the aggregation weights. In all settings, MeritFed outperforms SGD Ideal and other baselines regardless of α. In all cases, the weights are almost the same for all workers during the few initial steps (even if workers have quite different distributions like for the last nine clients). This phenomenon can be explained as follows: if we have two different convex functions with different optima (e.g., two quadratic functions), then for a far enough starting point, the gradients of those functions will point roughly in the same direction. Therefore, during a few initial steps, both gradients are useful and the method gives noticeable weights to both. However, once the method comes closer to the optima, the gradients become noticeably different, and after a certain stage, the gradient of the second function no longer points closely towards the optimum of the first function. Therefore, starting from this stage, MeritFed assigns a smaller weight to the gradient of the second function. Going back to Figures 4-7, we see a similar behavior: for $\alpha = 0.5$, the advantages of collaboration with clients 2-11 disappear after a certain stage since the method reaches the region where two distributions become noticeably different. In contrast, when $\alpha = 0.99$, those workers have a very close distribution to the target worker, and therefore, their stochastic gradients remain useful during the whole learning process. FedAdp is biased to the target client and assigns almost identical weights to either clients with similar or dissimilar distributions, which results in an accuracy decrease at the end of the training, in contrast to MeritFed, which tracks and maintains less weights to non-beneficial clients. TAWT is much more biased to the target client, which makes it almost identical to SGD Ideal.

Texts classification: GoEmotions + BERT. The next problem we consider is devoted to fine-tuning pretrained BERT (Devlin et al., 2018) model for emotions classification on the GoEmotions dataset (Demszky et al., 2020). The dataset consist of texts labeled with one or more of 28 emotions. First of all, we form "truncated dataset" by cutting the dataset so that its each entry has the only label. Then we use Ekman mapping (Ekman, 1992) to split the data between clients. According to the mapping, 28 emotions can be mapped to 7 basic emotions. That is, we simulate a situation when the target client classifies only basic emotions, e.g., the target client has only emotions belonging to "joy" class and namely includes only "joy", "amusement", "approval", "excitement", "gratitude", "love", "optimism", "relief", "pride", "admiration", "desire", "caring". The distribution of these sub-emotions is kept to be the same as the distribution of the truncated train dataset. Clients, that data are suppose to have similar distribution (second group – next 10 clients), also has texts from base class "joy" and are labeled as one of the sub-emotion belonging to "joy". The distribution of sub-emotions is also the same as the distribution of the truncated train dataset. These texts constitute an α portion of the total client's data. The other $1 - \alpha$ portion of the texts is taken from "neutral" class. The rest of clients (third group – next 9 clients) are supposed to have different distribution and their data consist of either texts belonging to one of the other basic emotion, either mixed with neutral (if there is not enough texts to have a desired number of samples) or texts from "neutral" class only. Again, the distribution of sub-emotions is the same as the distribution of the truncated train dataset. For MeritFed each worker calculates stochastic gradient using a batch size of 40; then the server performs 10 steps of Mirror Descent (or its stochastic version) with a batch-size of 30 (in case of stochastic version) and a learning rate of 0.1 to update weights of aggregation, and then performs a model parameters update with a learning rate of 0.01. The plots are averaged over 3 runs with different seeds. Additionally, accuracy plots show standard deviation. The results are presented in Figures 8-11. The target client benefits from collaborating with clients from the second group and achieves better accuracy using MeritFed. In general, the results are similar to the ones obtained for image classification.

MedMNIST. We apply MeritFed to enhance the classification of medical images, as introduced in the MedMNIST dataset (Yang et al., 2021). MedMNIST offers medical image datasets, including three datasets featuring images of internal organs (Organ{A,C,S}MNIST) with identical labels. These datasets can be collectively utilized during training to improve accuracy. A potential method involves aggregating gradients computed from these three datasets. However, due to the diverse nature of the data, some datasets may have limited contributions to the training. We anticipate that adaptive aggregation, provided by MeritFed, will improve the model's performance. For

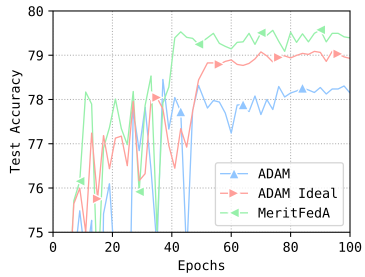

Figure 12: Test Accuracy for OrgansMNIST

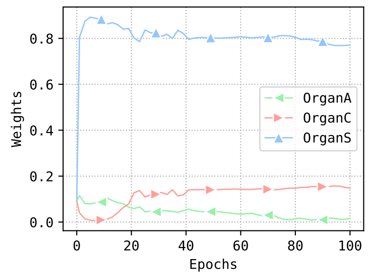

Figure 13: Evolution of Relevant Weights

empirical justification, we assume that each worker possesses one MedMNIST dataset. Importantly, `MeritFed` does not restrict the setup to only three workers and accommodates additional clients with irrelevant data, aligning with real-world scenarios. To demonstrate this, we introduce a nuisance worker handling data from other MedMNIST datasets. See Appendix C.2 for the detailed description.

OrganSMNIST worker is the target one. For the `ADAM Ideal` baseline, we use only the gradients from the target client and ignore the others. Moreover, we employ the same hyperparameters as specified in (Yang et al., 2021). See For `ADAM` baseline, we aggregate gradients uniformly from the first three workers, then proceed with the Adam step. For `MeritFed`, we maintain the same parameters but adjust the learning rate schedule to reduce after 40 and 75 epochs. The mirror descent learning rate is set at 0.1, with five iterations. To enable a fair comparison, we incorporate our adaptive aggregation technique into Adam optimizer, obtaining `MeritFedA`. It adaptively aggregates gradients before performing the Adam update. The gradient with respect to the weights is obtained by deriving the Adam update formula, where the gradient is replaced with its weighted counterpart. This derived gradient is then used to update the weights of aggregation via Mirror Descent. The experimental results, depicted in Figures 12 and 13 demonstrate the superior performance of `MeritFed`. They also highlight its capability to identify workers that are beneficial for training.

5 Conclusion

In this paper, we introduced a novel algorithm called Merit-based Federated Learning (`MeritFed`) to address the challenges posed by the heterogeneous data distributions in federated learning (FL) via the adaptive selection of the aggregation weights through solving an auxiliary minimization problem at each iteration. We demonstrated that `MeritFed` can effectively harness the advantages of distinct data distributions, control the detrimental effects of outlier clients, and promote collaborative learning. Our approach assigns adaptive aggregation weights to clients participating in FL training, allowing for faster convergence and potentially better generalization. `MeritFed` stands in contrast to `TAWT`, which depends on computationally intensive hypergradient estimations, and `FedAdp`, which utilizes cosine similarity for weight calculation. In addition, we incorporate zero-order mirror descent (MD) to enhance privacy. The key contributions of this paper include the development of `MeritFed`, provable convergence under mild assumptions, and the ability to utilize benefits from collaborating with clients having different but similar data distributions.

However, our work has some limitations. Firstly, (in theory) `MeritFed` relies on the fact that the objective from the problem in Line 7 gives a good enough approximation of the expected risk f, which in some situations may require the availability of additional data on the target client to solve the problem (though in all of our experiments, it was not the case and `MeritFed` worked well even without additional data). Collecting and maintaining extra data may not always be practical or efficient. Secondly, the experiments used a limited number of clients and a dataset of moderate size. Extending `MeritFed` to large-scale FL with a substantial number of clients and massive datasets may pose scalability challenges. Addressing these limitations is part of our plan for future work.

Furthermore, `MeritFed` serves as a foundation for numerous extensions and enhancements. Future research can explore topics such as acceleration techniques, adaptive or scaled optimization methods (e.g., variants akin to `Adam`) on the server side, communication compression strategies, and the efficient implementation of similar collaborative learning approaches for all clients simultaneously. These directions will contribute to the continued development of federated learning methods, making them more efficient, robust, and applicable to a wide range of practical scenarios.

REFERENCES

Dan Alistarh, Demjan Grubic, Jerry Li, Ryota Tomioka, and Milan Vojnovic. Qsgd: Communication-efficient sgd via gradient quantization and encoding. *Advances in neural information processing systems*, 30, 2017.

Abdulla Jasem Almansoori, Samuel Horváth, and Martin Takáč. Collaborative and efficient personalization with mixtures of adaptors. *arXiv*, 2024.

Wenxuan Bao, Haohan Wang, Jun Wu, and Jingrui He. Optimizing the collaboration structure in cross-silo federated learning. In *International Conference on Machine Learning*, pp. 1718–1736. PMLR, 2023.

Moran Baruch, Gilad Baruch, and Yoav Goldberg. A little is enough: Circumventing defenses for distributed learning, 2019.

Aleksandr Beznosikov, Vadim Sushko, Abdurakhmon Sadiev, and Alexander Gasnikov. Decentralized personalized federated min-max problems. *arXiv preprint arXiv:2106.07289*, 2021.

Aleksandr Beznosikov, Martin Takác, and Alexander Gasnikov. Similarity, compression and local steps: three pillars of efficient communications for distributed variational inequalities. *Advances in Neural Information Processing Systems*, 36, 2024.

Ekaterina Borodich, Aleksandr Beznosikov, Abdurakhmon Sadiev, Vadim Sushko, Nikolay Savelyev, Martin Takáč, and Alexander Gasnikov. Decentralized personalized federated min-max problems. *arXiv preprint arXiv:2106.07289*, 2021.

Can Chen, Xi Chen, Chen Ma, Zixuan Liu, and Xue Liu. Gradient-based bi-level optimization for deep learning: A survey. *arXiv preprint arXiv:2207.11719*, 2022a.

Fengwen Chen, Guodong Long, Zonghan Wu, Tianyi Zhou, and Jing Jiang. Personalized federated learning with a graph. In Lud De Raedt (ed.), *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*, pp. 2575–2582. International Joint Conferences on Artificial Intelligence Organization, 7 2022b. doi: 10.24963/ijcai.2022/357. URL `https://doi.org/10.24963/ijcai.2022/357`. Main Track.

Shuxiao Chen, Koby Crammer, Hangfeng He, Dan Roth, and Weijie J Su. Weighted training for cross-task learning. *arXiv preprint arXiv:2105.14095*, 2021a.

Tianyi Chen, Yuejiao Sun, and Wotao Yin. Closing the gap: Tighter analysis of alternating stochastic gradient methods for bilevel problems. *Advances in Neural Information Processing Systems*, 34: 25294–25307, 2021b.

Tianyi Chen, Yuejiao Sun, Quan Xiao, and Wotao Yin. A single-timescale method for stochastic bilevel optimization. In *International Conference on Artificial Intelligence and Statistics*, pp. 2466–2488. PMLR, 2022c.

Wenlin Chen, Samuel Horváth, and Peter Richtárik. Optimal client sampling for federated learning. *Transactions on Machine Learning Research*, 2022d. ISSN 2835-8856. URL `https://openreview.net/forum?id=8GvRCWKHIL`.

Yae Jee Cho, Jianyu Wang, and Gauri Joshi. Client selection in federated learning: Convergence analysis and power-of-choice selection strategies. *arXiv preprint arXiv:2010.01243*, 2020.

Liam Collins, Hamed Hassani, Aryan Mokhtari, and Sanjay Shakkottai. Exploiting shared representations for personalized federated learning. In *International conference on machine learning*, pp. 2089–2099. PMLR, 2021.

Mathieu Dagréou, Pierre Ablin, Samuel Vaiter, and Thomas Moreau. A framework for bilevel optimization that enables stochastic and global variance reduction algorithms. *arXiv preprint arXiv:2201.13409*, 2022.

Dorottya Demszky, Dana Movshovitz-Attias, Jeongwoo Ko, Alan Cowen, Gaurav Nemade, and Sujith Ravi. GoEmotions: A Dataset of Fine-Grained Emotions. In *58th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2020.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Shu Ding and Wei Wang. Collaborative learning by detecting collaboration partners. *Advances in Neural Information Processing Systems*, 35:15629–15641, 2022.

John C Duchi, Michael I Jordan, Martin J Wainwright, and Andre Wibisono. Optimal rates for zero-order convex optimization: The power of two function evaluations. *IEEE Transactions on Information Theory*, 61(5):2788–2806, 2015.

Paul Ekman. An argument for basic emotions. *Cognition & emotion*, 6(3-4):169–200, 1992.

Mathieu Even, Laurent Massoulié, and Kevin Scaman. On sample optimality in personalized collaborative and federated learning. *Advances in Neural Information Processing Systems*, 35: 212–225, 2022.

Alireza Fallah, Aryan Mokhtari, and Asuman Ozdaglar. Personalized federated learning: A meta-learning approach. *arXiv preprint arXiv:2002.07948*, 2020.

Vitaly Feldman and Jan Vondrak. High probability generalization bounds for uniformly stable algorithms with nearly optimal rate. In *Conference on Learning Theory*, pp. 1270–1279. PMLR, 2019.

Yann Fraboni, Richard Vidal, Laetitia Kameni, and Marco Lorenzi. Clustered sampling: Low-variance and improved representativity for clients selection in federated learning. In *International Conference on Machine Learning*, pp. 3407–3416. PMLR, 2021.

Alexander Gasnikov, Darina Dvinskikh, Pavel Dvurechensky, Eduard Gorbunov, Aleksander Beznosikov, and Alexander Lobanov. Randomized gradient-free methods in convex optimization. *arXiv preprint arXiv:2211.13566*, 2022a.

Alexander Gasnikov, Anton Novitskii, Vasilii Novitskii, Farshed Abdukhakimov, Dmitry Kamzolov, Aleksandr Beznosikov, Martin Takac, Pavel Dvurechensky, and Bin Gu. The power of first-order smooth optimization for black-box non-smooth problems. In *International Conference on Machine Learning*, pp. 7241–7265. PMLR, 2022b.

Saeed Ghadimi and Guanghui Lan. Stochastic first-and zeroth-order methods for nonconvex stochastic programming. *SIAM Journal on Optimization*, 23(4):2341–2368, 2013.

Saeed Ghadimi and Mengdi Wang. Approximation methods for bilevel programming. *arXiv preprint arXiv:1802.02246*, 2018.

Avishek Ghosh, Jichan Chung, Dong Yin, and Kannan Ramchandran. An efficient framework for clustered federated learning. *Advances in Neural Information Processing Systems*, 33:19586–19597, 2020.

Filip Hanzely, Slavomír Hanzely, Samuel Horváth, and Peter Richtárik. Lower bounds and optimal algorithms for personalized federated learning. *Advances in Neural Information Processing Systems*, 33:2304–2315, 2020.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Mingyi Hong, Hoi-To Wai, Zhaoran Wang, and Zhuoran Yang. A two-timescale framework for bilevel optimization: Complexity analysis and application to actor-critic. *arXiv preprint arXiv:2007.05170*, 2020.

Samuel Horváth and Peter Richtárik. A better alternative to error feedback for communication-efficient distributed learning. *arXiv preprint arXiv:2006.11077*, 2020.

Yankun Huang, Qihang Lin, Nick Street, and Stephen Baek. Federated learning on adaptively weighted nodes by bilevel optimization. *arXiv preprint arXiv:2207.10751*, 2022.

Kaiyi Ji, Junjie Yang, and Yingbin Liang. Bilevel optimization: Convergence analysis and enhanced design. In *International conference on machine learning*, pp. 4882–4892. PMLR, 2021.

Kaiyi Ji, Mingrui Liu, Yingbin Liang, and Lei Ying. Will bilevel optimizers benefit from loops. *arXiv preprint arXiv:2205.14224*, 2022.

Anatoli Juditsky, Arkadi Nemirovski, and Claire Tauvel. Solving variational inequalities with stochastic mirror-prox algorithm. *Stochastic Systems*, 1(1):17–58, 2011.

Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. Advances and open problems in federated learning. *Foundations and Trends® in Machine Learning*, 14(1–2):1–210, 2021.

Hamed Karimi, Julie Nutini, and Mark Schmidt. Linear convergence of gradient and proximal-gradient methods under the polyak-łojasiewicz condition. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2016, Riva del Garda, Italy, September 19-23, 2016, Proceedings, Part I 16*, pp. 795–811. Springer, 2016.

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.

Ahmed Khaled and Peter Richtárik. Better theory for sgd in the nonconvex world. *Transactions on Machine Learning Research*, 2022.

Ahmed Khaled, Konstantin Mishchenko, and Peter Richtárik. Tighter theory for local sgd on identical and heterogeneous data. In *International Conference on Artificial Intelligence and Statistics*, pp. 4519–4529. PMLR, 2020. URL `http://proceedings.mlr.press/v108/bayoumi20a/bayoumi20a-supp.pdf`.

Jakub Konečný, H Brendan McMahan, Daniel Ramage, and Peter Richtárik. Federated optimization: Distributed machine learning for on-device intelligence. *arXiv preprint arXiv:1610.02527*, 2016.

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

Viraj Kulkarni, Milind Kulkarni, and Aniruddha Pant. Survey of personalization techniques for federated learning. In *2020 Fourth World Conference on Smart Trends in Systems, Security and Sustainability (WorldS4)*, pp. 794–797. IEEE, 2020.

Fan Lai, Xiangfeng Zhu, Harsha V Madhyastha, and Mosharaf Chowdhury. Oort: Efficient federated learning via guided participant selection. In *15th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 21)*, pp. 19–35, 2021.

Leslie Lamport, Robert Shostak, and Marshall Pease. The byzantine generals problem. *ACM Transactions on Programming Languages and Systems*, 4(3):382–401, 1982.

Junyi Li, Bin Gu, and Heng Huang. A fully single loop algorithm for bilevel optimization without hessian inverse. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 7426–7434, 2022.

Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. Federated learning: Challenges, methods, and future directions. *IEEE signal processing magazine*, 37(3):50–60, 2020a.

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, 2020b.

Xiangru Lian, Ce Zhang, Huan Zhang, Cho-Jui Hsieh, Wei Zhang, and Ji Liu. Can decentralized algorithms outperform centralized algorithms? a case study for decentralized parallel stochastic gradient descent. *Advances in Neural Information Processing Systems*, 30, 2017. URL `https://arxiv.org/pdf/1705.09056.pdf`.

Hongcheng Liu and Jindong Tong. New sample complexity bounds for (regularized) sample average approximation in several heavy-tailed, non-lipschitzian, and high-dimensional cases. *arXiv preprint arXiv:2401.00664*, 2024.

Risheng Liu, Jiaxin Gao, Jin Zhang, Deyu Meng, and Zhouchen Lin. Investigating bi-level optimization for learning and vision from a unified perspective: A survey and beyond. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(12):10045–10067, 2021.

Stanislaw Lojasiewicz. A topological property of real analytic subsets. *Coll. du CNRS, Les équations aux dérivées partielles*, 117(87-89):2, 1963.

Bing Luo, Wenli Xiao, Shiqiang Wang, Jianwei Huang, and Leandros Tassiulas. Tackling system and statistical heterogeneity for federated learning with adaptive client sampling. In *IEEE INFOCOM 2022-IEEE Conference on Computer Communications*, pp. 1739–1748. IEEE, 2022.

Lingjuan Lyu, Han Yu, Xingjun Ma, Lichao Sun, Jun Zhao, Qiang Yang, and Philip S Yu. Privacy and robustness in federated learning: Attacks and defenses. *arXiv preprint arXiv:2012.06337*, 2020.

Jie Ma, Guodong Long, Tianyi Zhou, Jing Jiang, and Chengqi Zhang. On the convergence of clustered federated learning. *arXiv preprint arXiv:2202.06187*, 2022.

Jie Ma, Tianyi Zhou, Guodong Long, Jing Jiang, and Chengqi Zhang. Structured federated learning through clustered additive modeling. *Advances in Neural Information Processing Systems*, 36: 43097–43107, 2023.

Othmane Marfoq, Giovanni Neglia, Aurélien Bellet, Laetitia Kameni, and Richard Vidal. Federated multi-task learning under a mixture of distributions. *Advances in Neural Information Processing Systems*, 34:15434–15447, 2021.

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*, pp. 1273–1282. PMLR, 2017.

Konstantin Mishchenko, Eduard Gorbunov, Martin Takáč, and Peter Richtárik. Distributed learning with compressed gradient differences. *arXiv preprint arXiv:1901.09269*, 2019.

Konstantin Mishchenko, Grigory Malinovsky, Sebastian Stich, and Peter Richtárik. Proxskip: Yes! local gradient steps provably lead to communication acceleration! finally! In *International Conference on Machine Learning*, pp. 15750–15769. PMLR, 2022.

I Necoara, Yu Nesterov, and F Glineur. Linear convergence of first order methods for non-strongly convex optimization. *Mathematical Programming*, 175:69–107, 2019.

Arkadi Nemirovski, Anatoli Juditsky, Guanghui Lan, and Alexander Shapiro. Robust stochastic approximation approach to stochastic programming. *SIAM Journal on optimization*, 19(4):1574–1609, 2009.

Hung T Nguyen, Vikash Sehwag, Seyyedali Hosseinalipour, Christopher G Brinton, Mung Chiang, and H Vincent Poor. Fast-convergent federated learning. *IEEE Journal on Selected Areas in Communications*, 39(1):201–218, 2020.

Boris T Polyak. Gradient methods for the minimisation of functionals. *USSR Computational Mathematics and Mathematical Physics*, 3(4):864–878, 1963.

Monica Ribero and Haris Vikalo. Communication-efficient federated learning via optimal client sampling. *arXiv preprint arXiv:2007.15197*, 2020.

Abdurakhmon Sadiev, Ekaterina Borodich, Aleksandr Beznosikov, Darina Dvinskikh, Saveliy Chezhegov, Rachael Tappenden, Martin Takáč, and Alexander Gasnikov. Decentralized personalized federated learning: Lower bounds and optimal algorithm for all personalization modes. *EURO Journal on Computational Optimization*, 10:100041, 2022.

Shai Shalev-Shwartz, Ohad Shamir, Nathan Srebro, and Karthik Sridharan. Stochastic convex optimization. In *COLT*, volume 2, pp. 5, 2009.

Ohad Shamir. An optimal algorithm for bandit and zero-order convex optimization with two-point feedback. *The Journal of Machine Learning Research*, 18(1):1703–1713, 2017.

Reza Shokri and Vitaly Shmatikov. Privacy-preserving deep learning. In *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*, pp. 1310–1321, 2015.

Zhuoqing Song, Weijian Li, Kexin Jin, Lei Shi, Ming Yan, Wotao Yin, and Kun Yuan. Communication-efficient topologies for decentralized learning with $o(1)$ consensus rate. *Advances in Neural Information Processing Systems*, 35:1073–1085, 2022.

Sebastian U Stich. Local sgd converges fast and communicates little. In *International Conference on Learning Representations*, 2018.

Sebastian U Stich, Jean-Baptiste Cordonnier, and Martin Jaggi. Sparsified sgd with memory. *Advances in Neural Information Processing Systems*, 31, 2018.

Yue Tan, Yixin Liu, Guodong Long, Jing Jiang, Qinghua Lu, and Chengqi Zhang. Federated learning on non-iid graphs via structural knowledge sharing. In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pp. 9953–9961, 2023.

Xueyang Tang, Song Guo, and Jingcai Guo. Personalized federated learning with contextualized generalization. *arXiv preprint arXiv:2106.13044*, 2021.

Jianyu Wang, Zachary Charles, Zheng Xu, Gauri Joshi, H Brendan McMahan, Maruan Al-Shedivat, Galen Andrew, Salman Avestimehr, Katharine Daly, Deepesh Data, et al. A field guide to federated optimization. *arXiv preprint arXiv:2107.06917*, 2021.

Jeremy West, Dan Ventura, and Sean Warnick. Spring research presentation: A theoretical foundation for inductive transfer. *Brigham Young University, College of Physical and Mathematical Sciences*, 1(08), 2007.

Hongda Wu and Ping Wang. Fast-convergent federated learning with adaptive weighting. *IEEE Transactions on Cognitive Communications and Networking*, 7(4):1078–1088, 2021. doi: 10.1109/TCCN.2021.3084406.

Cong Xie, Sanmi Koyejo, and Indranil Gupta. Fall of empires: Breaking byzantine-tolerant sgd by inner product manipulation, 2019.

Jiancheng Yang, Rui Shi, and Bingbing Ni. Medmnist classification decathlon: A lightweight automl benchmark for medical image analysis. In *IEEE 18th International Symposium on Biomedical Imaging (ISBI)*, pp. 191–195, 2021.

Chen Zhang, Yu Xie, Hang Bai, Bin Yu, Weihong Li, and Yuan Gao. A survey on federated learning. *Knowledge-Based Systems*, 216:106775, 2021.

Chunxu Zhang, Guodong Long, Tianyi Zhou, Zijian Zhang, Peng Yan, and Bo Yang. Gpfedrec: Graph-guided personalization for federated recommendation. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 4131–4142, 2024.

Michael Zhang, Karan Sapra, Sanja Fidler, Serena Yeung, and Jose M Alvarez. Personalized federated learning with first order model optimization. *arXiv preprint arXiv:2012.08565*, 2020.

Yihua Zhang, Prashant Khanduri, Ioannis Tsaknakis, Yuguang Yao, Mingyi Hong, and Sijia Liu. An introduction to bi-level optimization: Foundations and applications in signal processing and machine learning. *arXiv preprint arXiv:2308.00788*, 2023.

Martin Zinkevich, Markus Weimer, Lihong Li, and Alex Smola. Parallelized stochastic gradient descent. *Advances in neural information processing systems*, 23, 2010.

CONTENTS

A Extended Related work

A.1 Relation to Transfer Learning

While our approach resembles transfer learning (West et al., 2007), where a model trained on one dataset is then enhanced/fine-tuned on another related dataset, `MeritFed` differs significantly in both motivation and framework. Unlike transfer learning, which involves adapting a pre-trained model to new data, `MeritFed` enhances the training process itself. Transfer learning can be theoretically viewed as training with "better" initialization, while `MeritFed` decides on the fly what dataset to use and to what extent.

That is, `MeritFed` performs adaptive aggregation and benefits from clients having data with the same distribution. It promotes collaborative learning, which is particularly applicable in cross-silo federated learning (scenarios such as medical imaging).

Furthermore, in situations where datasets are unrelated, traditional transfer learning may not yield performance improvements. In contrast, `MeritFed` performs not worse than `SGD Ideal` under such conditions. Additionally, `MeritFed` provides robustness against Byzantine attacks, further distinguishing it from conventional transfer learning methods.

Exploring whether `MeritFed` can outperform transfer learning techniques in specific applications remains a valuable direction for future research but outside the scope of our work.

A.2 Personalized FL by Graph-based aggregation

Another related direction in FL more accurately addresses client clustering by constructing a clients' relation graph. Chen et al. (2022b) does a graph-based model aggregation (k-hop) based on an adaptively learned Graph Convolution Net (GCN). Zhang et al. (2024) also uses GCN to perform graph-guided aggregation but focuses on recommendations. Both works lack theoretical analysis and require solving a subproblem (similar to BLO) of learning GCN at each iteration. This subproblem has a higher computation cost than MeritFed has for adaptive aggregation.

A.3 Weights Update for `TAWT` and `FedAdp`

TAWT. A faithful implementation of `TAWT` (Chen et al., 2021a) would require a costly evaluation of the inverse of the Hessian matrix $\sum_{t=1}^{T} w_t \nabla^2 f(x^k)$ to calculate an approximation of hyper-gradient g^k. Then g^k is supposed to be used to run one step of Mirror Descent (with step size η^k) to update the weights:

$$w_t^{k+1} = \frac{w_t^k \exp\{-\eta^k g_t^k\}}{\sum_{t'=1}^{T} w_{t'}^k \exp\{-\eta^k g_{t'}^k\}}. \tag{12}$$

In practice, Chen et al. (2021a) advise bypassing this step by replacing the Hessian-inverse-weighted dissimilarity measure with a cosine-similarity-based measure, i.e., to approximate g_t^k by $-c \times \mathcal{S}(\nabla f_0(x^k), \nabla f_t(x^k))$, where

$$\mathcal{S}(a, b) = \arccos \frac{\langle a, b\rangle}{\|a\|\|b\|} \tag{8}$$

denotes the cosine similarity between two vectors.

FedAdp. `FedAdp` (Wu & Wang, 2021) uses a similar update rule for weights, but it additionally uses a non-linear mapping function (*Gompertz function*)

$$\mathcal{G}(\xi) = \alpha \left(1 - e^{-e^{-\alpha\xi}}\right)$$

where ξ is the *smoothed angle* in *radian*, e denotes the exponential constant and α is a constant. By denoting $\mathcal{S}_t^k = \mathcal{S}(\nabla f_0(x^k), \nabla f_t(x^k))$ one can obtain `FedAdp` weights update rule in the form

$$w_t^k = \frac{e^{\mathcal{G}(\mathcal{S}_t^k)}}{\sum_{t'=1}^{n} e^{\mathcal{G}(\mathcal{S}_t^k)}}.$$

B PROOF OF THEOREM 3.4

Theorem B.1. *Let Assumptions 3.1 and 3.2 hold. Then after T iterations of* `MeritFed` *with $\gamma \leq \frac{1}{2L}$ outputs x^i, $i = 0, \cdots, T-1$ such that*

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\left\|\nabla f(x^t)\right\|^2 \leq \frac{2\left(f(x^0) - f(x^*)\right)}{T\gamma} + \frac{2\sigma^2\gamma L}{G} + \frac{2\delta}{\gamma}, \tag{13}$$

where δ is the accuracy of solving the problem in Line 7 and $G = |\mathcal{G}|$. Moreover if Assumption 3.3 additionally holds, then after T iterations of `MeritFed` *with $\gamma \leq \frac{1}{2L}$ outputs x^T such that*

$$\mathbb{E}f(x^T) - f^* \leq (1-\gamma\mu)^T\left(f(x^0) - f^*\right) + \frac{\sigma^2\gamma L}{\mu G} + \frac{\delta T}{\gamma\mu}. \tag{14}$$

Proof. We write g_i^t or simply g_i instead of $g_i(x^t, \boldsymbol{\xi}_i^t)$ when there is no ambiguity. Then, the update rule in `MeritFed` can be written as

$$x^{t+1} = x^t - \gamma\sum_{i=0}^{n-1} w_i^{t+1} g_i(x^t),$$

where w^{t+1} is an approximate solution of

$$\min_{w\Delta_1^n} f\left(x^t - \gamma\sum_{i=0}^{n-1} w_i g_i(x^t)\right)$$

that satisfies

$$\mathbb{E}\left[f(x^{t+1})|x^t, \boldsymbol{\xi}^t\right] - \min_w f\left(x^t - \gamma\sum_{i=0}^{n-1} w_i g_i(x^t)\right) \leq \delta.$$

By definition of the minimum, we have

$$\begin{aligned}
&\min_{w\in\Delta_1^n} f\left(x^t - \gamma\sum_{i=0}^{n-1} w_i g_i(x^t)\right) \leq f\left(x^t - \frac{\gamma}{G}\sum_{i\in\mathcal{G}} g_i(x^t)\right)\\
&\overset{\text{(Lip)}}{\leq} \quad f(x^t) - \frac{\gamma}{G}\left\langle\nabla f(x^t), \sum_{i\in\mathcal{G}} g_i(x^t)\right\rangle + \frac{L\gamma^2}{2}\left\|\frac{1}{G}\sum_{i\in\mathcal{G}} g_i(x^t)\right\|^2\\
&\leq \quad f(x^t) - \frac{\gamma}{G}\left\langle\nabla f(x^t), \sum_{i\in\mathcal{G}} g_i(x^t)\right\rangle + \gamma^2 L\left\|\nabla f(x^t) - \frac{1}{G}\sum_{i\in\mathcal{G}} g_i(x^t)\right\|^2 + \gamma^2 L\left\|\nabla f(x^t)\right\|^2.
\end{aligned}$$

The last two inequalities imply

$$\begin{aligned}
&\mathbb{E}\left[f(x^{t+1})|x^t, \boldsymbol{\xi}^t\right]\\
&\leq \quad f(x^t) - \frac{\gamma}{G}\left\langle\nabla f(x^t), \sum_{i\in\mathcal{G}} g_i(x^t)\right\rangle + \gamma^2 L\left\|\nabla f(x^t) - \frac{\sum_{i\in\mathcal{G}} g_i(x^t)}{G}\right\|^2 + \gamma^2 L\left\|\nabla f(x^t)\right\|^2 + \delta.
\end{aligned}$$

Taking the full expectation we get

$$\begin{aligned}
\mathbb{E}[f(x^{t+1})] \quad &\leq \quad \mathbb{E}[f(x^t)] - \gamma(1-\gamma L)\mathbb{E}\left[\|\nabla f(x^t)\|^2\right] + \gamma^2 L\mathbb{E}\left[\left\|\nabla f(x^t) - \frac{\sum_{i\in\mathcal{G}} g_i(x^t)}{G}\right\|^2\right] + \delta\\
&\overset{\gamma\leq\frac{1}{2L}}{\leq} \quad \mathbb{E}[f(x^t)] - \frac{\gamma}{2}\mathbb{E}\left[\|\nabla f(x^t)\|^2\right] + \frac{\gamma^2 L}{G^2}\sum_{i\in\mathcal{G}}\mathbb{E}\left[\left\|\nabla f(x^t) - g_i(x^t)\right\|^2\right] + \delta\\
&\overset{(11)}{\leq} \quad \mathbb{E}[f(x^t)] - \frac{\gamma}{2}\mathbb{E}\left[\|\nabla f(x^t)\|^2\right] + \frac{\gamma^2 L\sigma^2}{G} + \delta.
\end{aligned} \tag{15}$$

The above is equivalent to

$$\frac{\gamma}{2}\mathbb{E}\left\|\nabla f\left(x^t\right)\right\|^2 \leq \mathbb{E}f\left(x^t\right) - \mathbb{E}f\left(x^{t+1}\right) + \frac{\sigma^2\gamma^2 L}{G} + \delta,$$

which concludes the first part of the proof.

Next, summing the inequality for $t \in \{0, 1, \ldots, T-1\}$ leads to

$$\begin{aligned}\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\left\|\nabla f\left(x^t\right)\right\|^2 &\leq \frac{2\left(f\left(x^0\right) - \mathbb{E}f\left(x^T\right)\right)}{T\gamma} + \frac{2\sigma^2\gamma L}{G} + \frac{2\delta}{\gamma} \\ &\leq 2\left(f\left(x^0\right) - f(x^*)\right) + \frac{2\sigma^2\gamma L}{G} + \frac{2\delta}{\gamma}.\end{aligned}$$

Combining (15) with (PL) gives

$$\mathbb{E}[f(x^{t+1}) - f^*] \quad \leq \quad (1-\gamma\mu)\mathbb{E}[f(x^t) - f^*] + \frac{\gamma^2 L\sigma^2}{G} + \delta.$$

Unrolling the above recurrence, we obtain (14). □

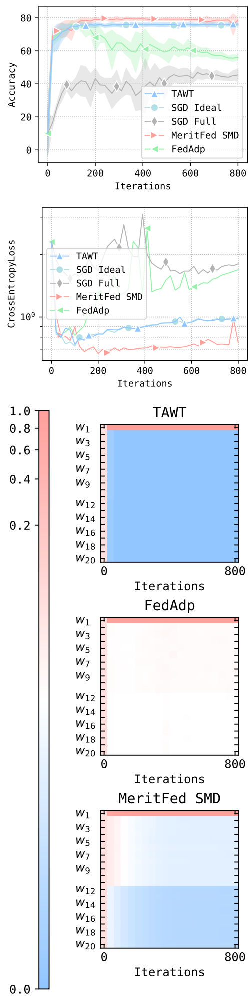

Figure 14: CIFAR10: $\alpha = 0.5$

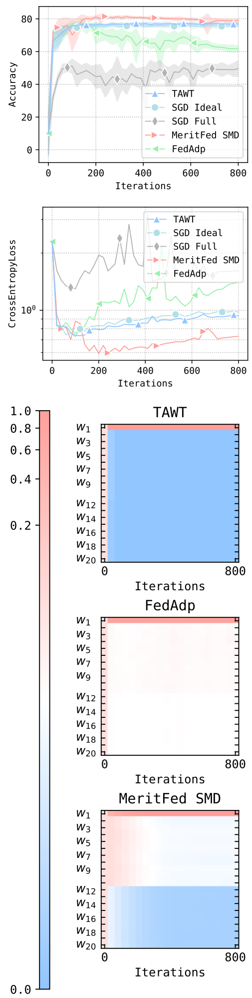

Figure 15: CIFAR10: $\alpha = 0.7$

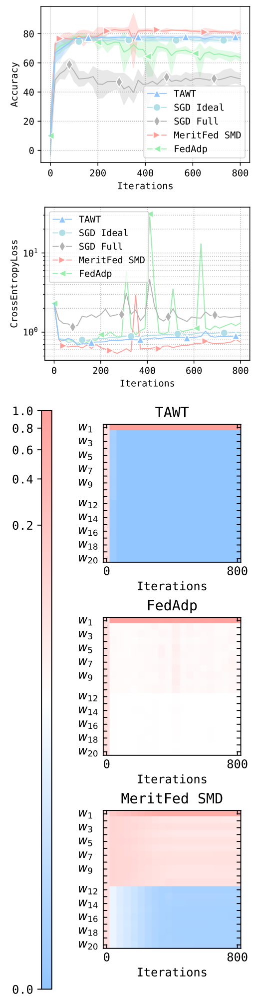

Figure 16: CIFAR10: $\alpha = 0.9$

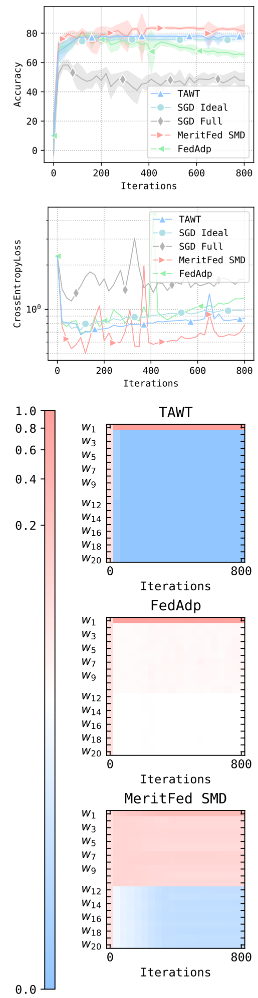

Figure 17: CIFAR10: $\alpha = 0.99$

C Additional Experiments

C.1 Results without Additional Validation Dataset

In this section, we provide experiments without an additional dataset. Instead, we use the target client's train dataset to approximately solve the problem in Line 7. The results are provided in Figures 14-17 (image classification) and Figures 18-21 (text classification). They show that `MeritFed`'s behavior with and without additional validation data is almost the same. Thus, these preliminary results give evidence that our method can be efficient in practice even when an extra validation dataset is unavailable.

C.2 Missing Details for MedMNIST Experiments

Complete dataset-worker mapping is OrganSMNIST, OrganAMNIST, OrganCMNIST, PathMNIST, DermaMNIST, OCTMNIST, PneumoniaMNIST, RetinaMNIST, BreastMNIST, BloodMNIST, TissueMNIST. OrganSMNIST worker is the target one.

We employ the same hyperparameters as specified in (Yang et al., 2021), including an input resolution of 28x28, ResNet-18 architecture, entropy loss, a batch size of 128, and the Adam optimizer with an

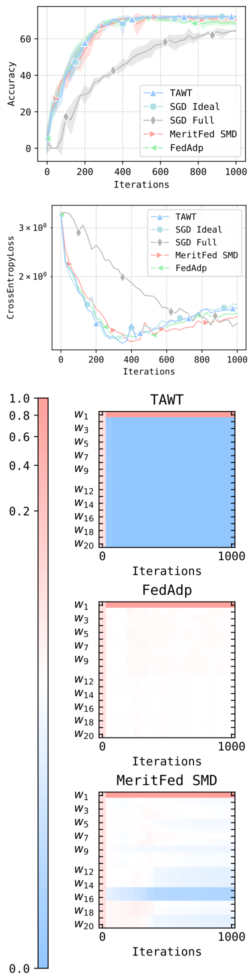

Figure 18: GoEmotions: $\alpha = 0.5$

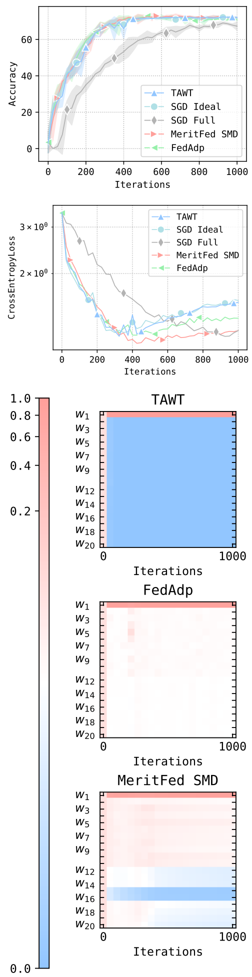

Figure 19: GoEmotions: $\alpha = 0.7$

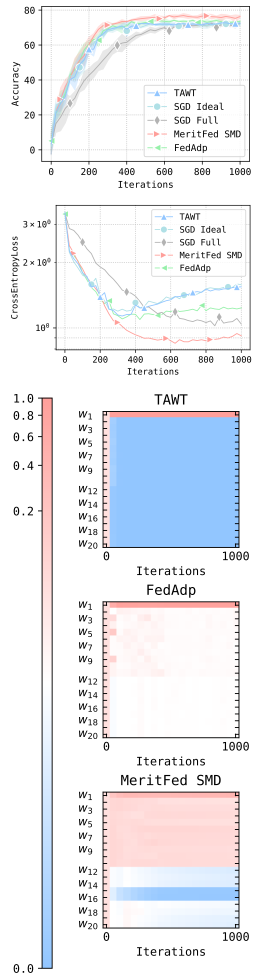

Figure 20: GoEmotions: $\alpha = 0.9$

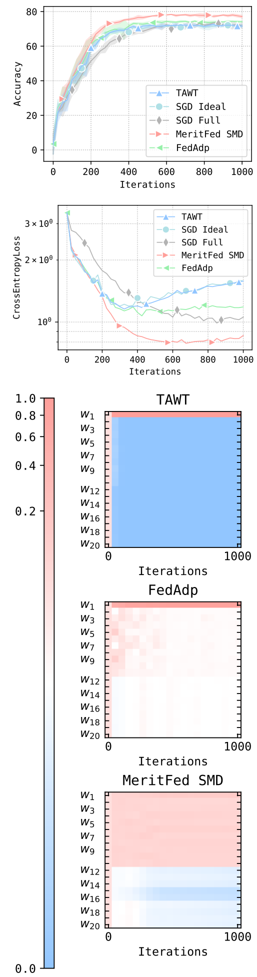

Figure 21: GoEmotions: $\alpha = 0.99$

initial learning rate of 0.001. This setup is run for 100 epochs, with the learning rate decreased by a factor of 0.1 after 50 and 75 epochs. Additionally, we expand the number of channels for grayscale images, as originally done by the authors.

C.3 Robustness against Byzantine Attacks

`MeritFed` is robust to Byzantine attacks since our proof of Theorem 3.4 does not make any assumptions on the vectors received from the workers having different data distribution than the target client. This means that any worker $i \notin \mathcal{G}$ can send arbitrary vectors at each iteration, and `MeritFed` will still be able to converge. Moreover, `MeritFed` can tolerate Byzantine attacks even if Byzantine workers form a majority, e.g., the method converges even if all clients are Byzantine except for the target one.

To test the Byzantine robustness of our method on the mean estimation problem, we chose the total number of peers equal to 55 with the 50 clients being malicious. Malicious clients know the target distribution of the first 5 client and use it for performing IPM (with parameter $\varepsilon_{\text{IPM}} = 0.1$) (Xie et al., 2019) and ALIE (with parameter $z_{\text{ALIE}} = 100$) (Baruch et al., 2019) attacks. We also consider the Bit

Flipping[4] (BF) and the Random Noise[5] (RN) attacks. The following choice of parameters is used: each client has 1000 samples from the corresponding distribution. The dimension of the problem is $d = 10$, learning rate $= 0.01$, number of steps for Mirror Descent $= 10$, learning rate for Mirror Descent $= 3.5$.

The results are presented in Figures 22-25. As expected, `SGD Full` does not converge under the considered attacks, and `SGD Ideal` shows the best results since, by design, it averages only with non-Byzantine workers. `FedAdp` has poor performance under ALIE attack and is quite unstable under RN attack. As in other experiments, `TAWT` is very biased towards the target client, which helps `TAWT` to tolerate Byzantine attacks, but it does not take extra advantage of averaging with clients having the same distribution. Finally, `MeritFed` consistently shows comparable results to `SGD Ideal`.

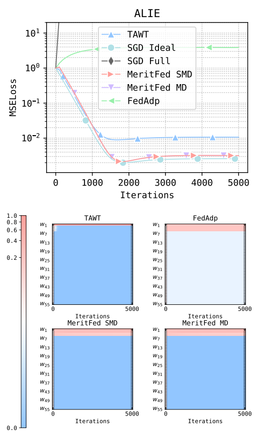

Figure 22: ALIE

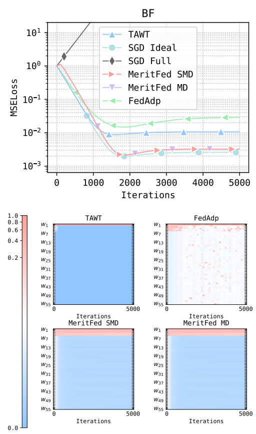

Figure 23: BF

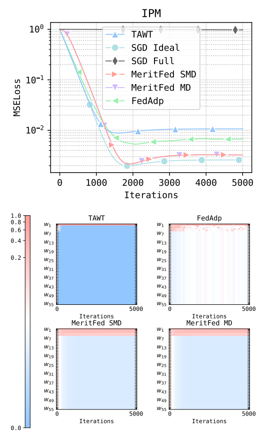

Figure 24: IPM

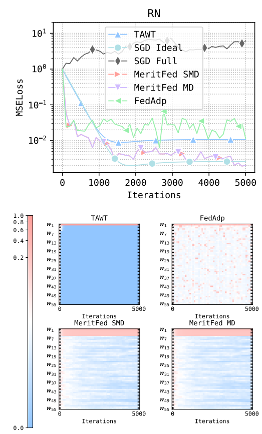

Figure 25: RN

C.4 ResNet18+CIFAR10: 40 workers

In the mean estimation problem, we generate the data and can control the number of workers. Therefore, for this problem we have many clients participating in the training.

However, for the other two tasks, datasets are fixed. Therefore, we limited the number of workers to 20 to have enough data on each client (given the splitting strategy) without repetition. That is, each data sample (image or tokens) from the original datasets belongs to no more than 1 client. Therefore, to run experiments with more workers we either need to have more data or allow repetitions in data on the clients.

In the additional experiments, we have 40 clients where the new 20 clients are just copies of the first 20 clients. The experimental setup follows the same data partitioning idea as presented in the paper and deals with for values of heterogeneity values across clients α. For `MeritFed` each worker calculates stochastic gradient using a batch size of 75; then the server uses Mirror Descent (or its stochastic version) with a batch-size of 90 (in case of stochastic version) and a learning rate of 0.1 to update weights of aggregation, and then performs a model parameters update with a learning rate of 0.01.

The results presented on Figures 26-29. Overall, the conclusions are consistent with what we have in the experiment with 20 workers, further supporting the scalability of `MeritFed`.

[4]Byzantine workers compute stochastic gradients g_i^k and send $-g_i^k$ to the server.

[5]Byzantine workers compute stochastic gradients g_i^k and send $g_i^k + \sigma\xi_i^k$ to the server, where $\xi_i^k \sim \mathcal{N}(0, \boldsymbol{I})$ and $\sigma = 1$.

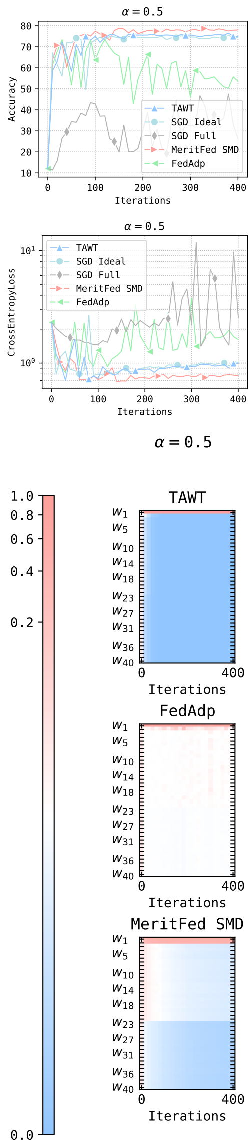

Figure 26: CIFAR10: $\alpha = 0.5$

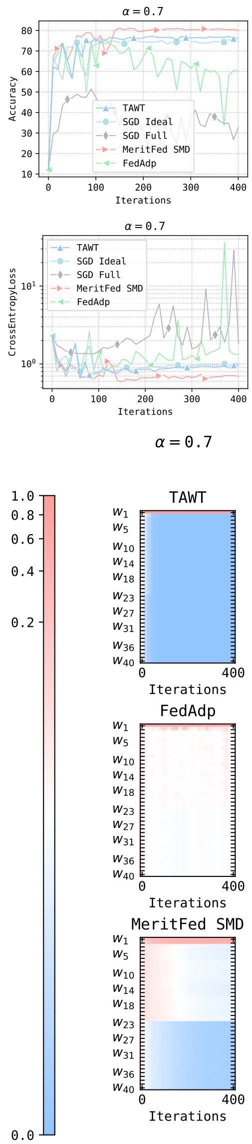

Figure 27: CIFAR10: $\alpha = 0.7$

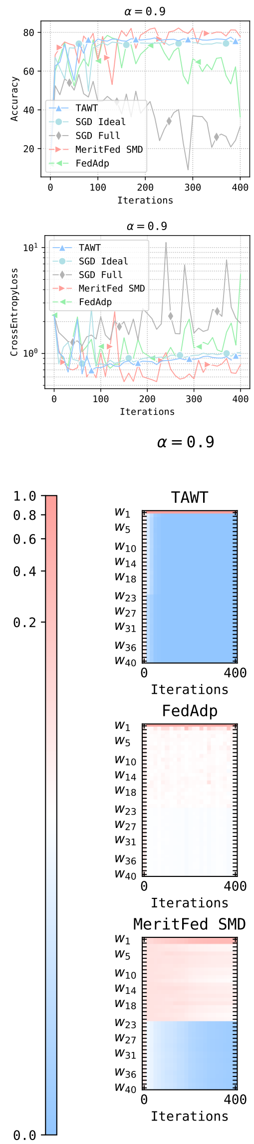

Figure 28: CIFAR10: $\alpha = 0.9$

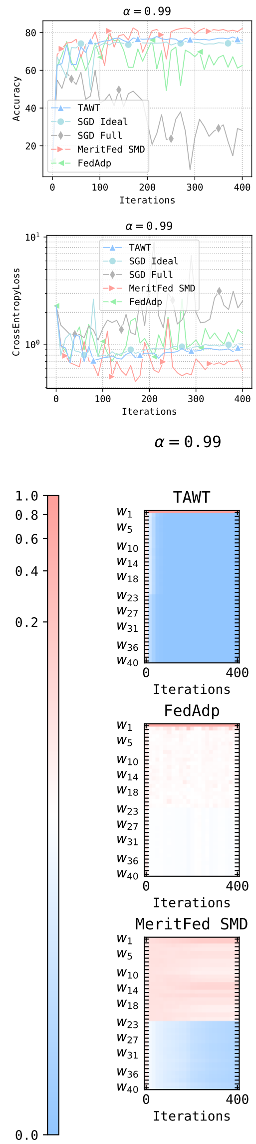

Figure 29: CIFAR10: $\alpha = 0.99$.