Federated Optimization for Heterogeneous Networks

Tian Li *[1], Anit Kumar Sahu*[1], Manzil Zaheer[2], Maziar Sanjabi[3], Ameet Talwalkar[1, 4], and Virginia Smith[1]

[1]Carnegie Mellon University
[2]Google Research
[3]University of Southern California
[4]Determined AI

Abstract

Federated learning involves training machine learning models in massively distributed networks. While Federated Averaging (`FedAvg`) is the leading optimization method for training non-convex models in this setting, its behavior is not well understood in realistic federated settings when learning across statistically heterogeneous devices, i.e., where each device collects data in a non-identical fashion. In this work, we introduce a framework to tackle statistical heterogeneity, `FedProx`, which encompasses `FedAvg` as a special case. We provide convergence guarantees for `FedProx` through a device dissimilarity assumption that allows us to characterize heterogeneity in the network. Finally, we perform a detailed empirical evaluation across a suite of federated datasets, validating our theoretical analysis and demonstrating the improved robustness and stability of the generalized `FedProx` framework relative to `FedAvg` for learning in heterogeneous networks.

1 Introduction

Large networks of remote devices, such as phones, vehicles, and wearable sensors, generate a wealth of data each day. Due to user privacy concerns and systems constraints (e.g., communication costs, device-level computational constraints, and low availability amongst devices), federated learning has emerged as an attractive paradigm to push the training of models in such networks to the edge [19].

Optimization methods that allow for local updating and low participation have become the de facto solvers for federated learning [19, 25]. These methods perform a variable number of local updates on a subset of devices to enable flexible and efficient communication, e.g., compared to traditional distributed gradient descent or stochastic gradient descent (SGD). Of current federated optimization methods, `FedAvg` [19] has become state-of-the-art for non-convex federated learning. `FedAvg` works simply by running some number of epochs, E, of SGD on a subset $K \ll N$ of the total devices N at each communication round, and then averaging the resulting model updates via a central server.

However, `FedAvg` was not designed to tackle the *statistical heterogeneity* inherent in federated settings; namely, that data may be non-identically distributed across devices. In realistic statistically heterogeneous settings, `FedAvg` has been shown to diverge empirically [e.g., 19, Sec 3], and it also lacks theoretical convergence guarantees. Indeed, recent works exploring convergence guarantees are limited to unrealistic scenarios,

*Authors contributed equally.

e.g., where (1) the data is either shared across devices or distributed in an IID (independent and identically distributed) manner, or (2) all devices are involved in communication at each round [27, 29, 30, 31, 34, 38]. While these assumptions simplify the analyses, they also violate key properties of realistic federated networks.

Contributions. In this work, we ask the following two questions: (1) Can we gain a principled understanding of `FedAvg` in realistic, statistically heterogeneous federated settings? (2) Can we devise an improved federated optimization algorithm, both theoretically and empirically? To this end, we propose a federated optimization framework for heterogeneous networks, `FedProx`, which encompasses `FedAvg`. In order to characterize the convergence behavior of `FedProx` as a function of statistical heterogeneity, we introduce a *device dissimilarity* assumption in the network. Under this assumption, we provide the first convergence guarantees for `FedProx` in practical federated settings with heterogeneous data. Furthermore, through a set of experiments on numerous real-world federated datasets, we demonstrate that our theoretical assumptions reflect empirical performance, and that `FedProx` can improve the robustness and stability of convergence in comparison to `FedAvg` when data is heterogeneous across devices.

2 Related Work

Large-scale machine learning, particularly in data center settings, has motivated the development of numerous distributed optimization methods in the past decade [see, e.g., 3, 6, 7, 16, 21, 22, 24, 26, 35, 36]. However, as computing substrates such as phones, sensors, and wearable devices grow both in power and in popularity, it is increasingly attractive to learn statistical models directly over networks of distributed devices, as opposed to moving the data to the data center. This problem, known as federated learning, requires tackling novel challenges with privacy, heterogeneous data and devices, and massively distributed computational networks.

Recent optimization methods have been proposed that are tailored to the specific challenges in the federated setting. These methods have shown significant improvements over traditional distributed approaches like ADMM [3] or mini-batch methods [7] by allowing both for inexact local updating in order to balance communication vs. computation in large networks, and for a small subset of devices to be active at any communication round [19, 25]. For example, Smith et al. [25] proposes a communication-efficient primal-dual optimization method that learns separate but related models for each device through a multi-task learning framework. Despite the theoretical guarantees and practical efficiency of the proposed method, such an approach is not generalizable to non-convex problems, e.g., deep learning, where strong duality is no longer guaranteed. In the non-convex setting, Federated Averaging (`FedAvg`), a heuristic method based on averaging local Stochastic Gradient Descent (SGD) updates in the primal, has instead been shown to work well empirically [19].

Unfortunately, `FedAvg` is quite challenging to analyze due to its local updating scheme, the fact that few devices are active at each round, and the issue that data is frequently distributed in a heterogeneous nature in the network. In particular, as each device generates its own local data, *statistical heterogeneity* is common with data being non-identically distributed between devices. Recent works have made steps towards analyzing `FedAvg` in simpler, non-federated settings. For instance, parallel SGD and related variants [18, 21, 24, 27, 29, 31, 35, 38], which make local updates similar to `FedAvg`, have been studied in the IID setting. However, the results rely on the premise that each local solver is a copy of the same stochastic process (due to the IID assumption). This line of reasoning does not apply to the heterogeneous setting. Although some works [10, 13, 30, 34] have recently explored convergence guarantees in heterogeneous settings, they make the limiting assumption that all devices participate in each round of communication, which is often infeasible in realistic federated networks [19]. Further, they rely on specific solvers to be used on each device (either SGD or GD), as compared to the solver-agnostic framework proposed herein, and add additional assumptions of convexity [30] or uniformly bounded gradients [34] to their analyses.

There are also heuristic approaches that aim to tackle statistical heterogeneity, either by sharing the local device data or some server-side proxy data [11, 12, 37]. However, these methods may be unrealistic: in addition to imposing burdens on network bandwidth, sending local data to the server [12] violates the key privacy assumption of federated learning, and sending globally-shared proxy data to all devices [11, 37] requires effort to carefully generate or collect such auxiliary data.

In this work, inspired by `FedAvg`, we explore a broader framework, `FedProx`, that is capable of handling heterogeneous federated data while maintaining similar privacy and computational benefits. We analyze the convergence behavior of the framework under a local dissimilarity assumption between local functions. Our assumption is inspired by the Kaczmarz method for solving linear system of equations [14], a similar assumption of which has been used to analyze variants of SGD in other settings [see, e.g., 23, 28, 33]. Our proposed framework allows for improved robustness and stability of convergence in heterogeneous federated networks.

Finally, we note that two aspects of our proposed work—the proximal term in `FedProx` and the bounded dissimilarity assumption used in our analysis—have been previously studied in the optimization literature, though often with very different motivations and in non-federated settings. For completeness, we provide a further discussion in Appendix E on this background work.

3 Federated Optimization: Algorithms

In this section, we introduce the key ingredients behind recent methods for federated learning, including `FedAvg`, and then outline our proposed framework, `FedProx`. Federated learning methods [e.g., 19, 25] are designed to handle multiple devices collecting data and a central server coordinating the global learning objective across the network. In particular, the aim is to minimize:

$$\min_w \; f(w) = \sum_{k=1}^{N} p_k F_k(w) = \mathbb{E}_k[F_k(w)], \tag{1}$$

where N is the number of devices, $p_k \geq 0$, and $\sum_k p_k = 1$. In general, the local objectives measure the local empirical risk over possibly differing data distributions $\mathcal{D}_k$, i.e., $F_k(w) := \mathbb{E}_{x_k \sim \mathcal{D}_k}[f_k(w; x_k)]$, with n_k samples available at each device k. Hence, we can set $p_k = \frac{n_k}{n}$, where $n = \sum_k n_k$ is the total number of data points. In this work, we consider $F_k(w)$ to be possibly non-convex.

To reduce communication and handle systems constraints, a common technique in federated optimization methods is that on each device, a *local objective function* based on the device's data is used as a surrogate for the global objective function. At each outer iteration, a subset of the devices are selected and *local solvers* are used to optimize the local objective functions on each of the selected devices. The devices then communicate their local model updates to the central server, which aggregates them and updates the global model accordingly. The key to allowing flexible performance in this scenario is that each of the local objectives can be solved *inexactly*. This allows the amount of local computation vs. communication to be tuned based on the number of local iterations that are performed. We introduce this notion formally below, as it will be utilized throughout the paper.

Definition 1 (γ-inexact solution)**.** For a function $h(w; w_0) = F(w) + \frac{\mu}{2}\|w - w_0\|^2$, and $\gamma \in [0, 1]$, we say w^* is a γ-inexact solution of $\min_w h(w; w_0)$ if $\|\nabla h(w^*; w_0)\| \leq \gamma \|\nabla h(w_0; w_0)\|$, where $\nabla h(w; w_0) = \nabla F(w) + \mu(w - w_0)$. Note that a smaller γ corresponds to higher accuracy.

For full generality, we use γ-inexactness in our analysis (Section 4) to measure the amount of local computation from each local solver. However, in our experiments (Section 5) we simply run an iterative local solver

for some number of local epochs, E, as in `FedAvg` (Algorithm 1). The number of local epochs can be seen as a proxy for γ-inexactness, and it is straightforward (albeit notationally burdensome) to extend our analysis to directly cover this case by allowing γ to vary by iteration and device, similar to the analysis in [25].

3.1 Federated Averaging (`FedAvg`)

In Federated Averaging (`FedAvg`) [19], the local surrogate of the global objective function at device k is $F_k(\cdot)$, and the local solver is stochastic gradient descent (SGD), with the same learning rate and the number of local epochs across devices. At each round, a subset $K \ll N$ of the total devices are selected and run SGD locally for E number of epochs, and then the resulting model updates are averaged. The details of `FedAvg` are summarized in Algorithm 1.

Algorithm 1: Federated Averaging (`FedAvg`)

INPUT: K, T, η, E, w^0, N, p_k, $k = 1, \cdots, N$;
forall $t = 0, \cdots, T-1$ **do**
  Server selects a subset S_t of K devices at random (each device k is chosen with probability p_k);
  Server sends w^t to all chosen devices;
  Each device $k \in S_t$ updates w^t for E epochs of SGD on F_k with step-size η to obtain w_k^{t+1};
  Each chosen device $k \in S_t$ sends w_k^{t+1} back to the server;
  Server aggregates the w's as $w^{t+1} = \frac{1}{K}\sum_{k \in S_t} w_k^{t+1}$;

McMahan et al. [19] shows empirically that it is crucial to tune the optimization hyperparameters for `FedAvg` properly. In particular, carefully tuning the number of local epochs is critical for `FedAvg` to converge, as additional local epochs allow local models to move further away from the initial global model, potentially causing divergence. In a heterogeneous setting, where the local objectives may be quite different from the global, this issue is exacerbated, and it is beneficial to restrict the amount of local deviation through a more principled tool than heuristically limiting the number of local epochs of some iterative solver. This observation serves as inspiration for `FedProx`, introduced below.

3.2 Proposed Framework: `FedProx`

Our proposed framework, `FedProx` (Algorithm 2), is similar to `FedAvg` in that a subset of devices are selected at each round, local updates are performed, and these updates are then averaged to form a global update. However, instead of just minimizing the local function $F_k(\cdot)$, device k uses its local solver of choice to approximately minimize the following surrogate objective h_k:

$$\min_w h_k(w;\ w^t) = F_k(w) + \frac{\mu}{2}\|w - w^t\|^2 . \tag{2}$$

The proximal term in the above expression effectively limits the impact of local updates (by restricting them to be close to the initial model) without any need to manually tune the number of local epochs as in `FedAvg`. We note that proximal terms such as the one above are a popular tool utilized throughout the optimization literature; for completeness, we provide a more detailed discussion on this in Appendix E. An important distinction of the proposed usage is that we suggest, explore, and analyze such a term for the purpose of tackling statistical heterogeneity in federated networks. Our analysis (Section 4) is unique in considering solving such an objective in a distributed setting with: (1) non-IID partitioned data, (2) the use of any local

solver, (3) inexact updates on each device, and (4) a subset of devices being active at each round. These assumptions are critical to providing a characterization of such a framework in realistic federated scenarios.

Algorithm 2: `FedProx` (Proposed Framework)

INPUT: K, T, μ, γ, w^0, N, p_k, $k = 1, \cdots, N$;
forall $t = 0, \cdots, T-1$ **do**
  Server selects a subset S_t of K devices at random (each device k is chosen with probability p_k);
  Server sends w^t to all chosen devices;
  Each chosen device $k \in S_t$ finds a w_k^{t+1} which is a γ-inexact minimizer of:
  $w_k^{t+1} \approx \arg\min_w \; h_k(w;\; w^t) = F_k(w) + \frac{\mu}{2}\|w - w^t\|^2$;
  Each chosen device $k \in S_t$ sends w_k^{t+1} back to the server;
  Server aggregates the w's as $w^{t+1} = \frac{1}{K}\sum_{k \in S_t} w_k^{t+1}$;

In our experiments (Section 5.2), we see the modified local subproblem in `FedProx` results in more robust and stable convergence compared to vanilla `FedAvg` for heterogeneous datasets. In Section 4, we also see that the usage of the proximal term makes `FedProx` more amenable to theoretical analysis (i.e., the local objective may be more well-behaved). In particular, if μ is chosen accordingly, the Hessian of h_k may be positive semi-definite. Hence, when F_k is non-convex, h_k will be convex, and when F_k is convex, it becomes μ-strongly convex. Note that `FedAvg` is a special case of `FedProx` with $\mu = 0$, and where the local solver is specifically chosen to be SGD. `FedProx` is significantly more general in this regard, as any local (possibly non-iterative) solver can be used on each device.

4 `FedProx`: Convergence Analysis

`FedAvg` and `FedProx` are stochastic algorithms by nature: in each round, only a fraction of the devices are sampled to perform the update, and the updates performed on each device may be inexact. It is well known that in order for stochastic methods to converge to a stationary point, a decreasing step-size is required. This is in contrast to non-stochastic methods, e.g., gradient descent, that can find a stationary point by employing a constant step-size. In order to analyze the convergence behavior of methods with constant step-size (as is usually implemented in practice), we need to quantify the degree of dissimilarity among the local objective functions. This could be achieved by assuming the data to be IID, i.e., homogeneous across devices. Unfortunately, in realistic federated networks, this assumption is impractical. Thus, we propose a metric that specifically measures the dissimilarity among local functions (Section 4.1) and analyze `FedProx` under this assumption (Section 4.2).

4.1 Local dissimilarity

Here we introduce a measure of dissimilarity between the devices in a federated network, which is sufficient to prove convergence. This can also be satisfied via a simpler bounded variance assumption of the gradients (Corollary 8), which we explore in our experiments in Section 5.

Definition 2 (B-local dissimilarity)**.** The local functions F_k are B-locally dissimilar at w if $\mathbb{E}_k\left[\|\nabla F_k(w)\|^2\right] \leq \|\nabla f(w)\|^2 B^2$. We further define $B(w) = \sqrt{\frac{\mathbb{E}_k[\|\nabla F_k(w)\|^2]}{\|\nabla f(w)\|^2}}$ for[1] $\|\nabla f(w)\| \neq 0$.

[1] As an exception we define $B(w) = 1$ when $\mathbb{E}_k\left[\|\nabla F_k(w)\|^2\right] = \|\nabla f(w)\|^2$, i.e. w is a stationary solution that all the local functions F_k agree on.

Here $\mathbb{E}_k[\cdot]$ denotes the expectation over devices with masses $p_k{=}n_k/n$ and $\sum_{k=1}^{N} p_k{=}1$ (as in Equation 1). As a sanity check, when all the local functions are the same, we have $B(w) = 1$ for all w. If the samples on all the devices are sampled in an IID fashion, then as $\min_k\ n_k \to \infty$, it follows that $B(w) \to 1$ for every w as all the local functions converge to the same expected risk function in the large sample limit. Thus, $B(w) \geq 1$ and the larger the value of $B(w)$, the larger is the dissimilarity among the local functions. Interestingly, similar assumptions [e.g., 23, 28, 33] have been explored elsewhere but for differing purposes; we provide a discussion of these works in Appendix E.

Using Definition 2, we now state our formal dissimilarity assumption, which we use in our convergence analysis. This simply requires that the dissimilarity defined in Definition 2 is bounded.

Assumption 1 (Bounded dissimilarity)**.** *For some $\epsilon > 0$, there exists a B_ϵ such that for all the points $w \in \mathcal{S}_\epsilon^c = \{w \mid \|\nabla f(w)\|^2 > \epsilon\}$, $B(w) \leq B_\epsilon$.*

For most practical machine learning problems, there is no need to solve the problem to arbitrarily accurate stationary solutions, i.e., ϵ is typically not very small. Indeed, it is well-known that solving the problem beyond some threshold may even hurt generalization performance due to overfitting [32]. Although in practical federated learning problems the samples are not IID, they are still sampled from distributions that are not entirely unrelated (if this were the case, e.g., fitting a single global model w across devices would be ill-advised). Thus, it is reasonable to assume that the dissimilarity between local functions remains bounded throughout the training process. We measure the notion of dissimilarity empirically for several real datasets in Section 5.4.

4.2 `FedProx` Analysis

Using the bounded dissimilarity assumption (Assumption 1), we now analyze the amount of expected decrease in the objective when one step of `FedProx` is performed.

Theorem 3 (Non-convex `FedProx` Convergence: B-local dissimilarity)**.** *Let Assumption 1 hold. Assume the functions F_k are non-convex, L-Lipschitz smooth, and there exists $L_- > 0$, such that $\nabla^2 F_k \succeq -L_- \mathbf{I}$, with $\bar{\mu} := \mu - L_- > 0$. Suppose that w^t is not a stationary solution and the local functions F_k are B-dissimilar, i.e. $B(w^t) \leq B$. If μ, K, and γ in Algorithm 2 are chosen such that*

$$\rho{=}\left(\frac{1}{\mu}{-}\frac{\gamma B}{\mu}{-}\frac{B(1{+}\gamma)\sqrt{2}}{\bar{\mu}\sqrt{K}}{-}\frac{LB(1{+}\gamma)}{\bar{\mu}\mu}\ {-}\frac{L(1{+}\gamma)^2B^2}{2\bar{\mu}^2}{-}\frac{LB^2(1{+}\gamma)^2}{\bar{\mu}^2K}\left(2\sqrt{2K}{+}2\right)\right){>}0,$$

then at iteration t of Algorithm 2, we have the following expected decrease in the global objective:

$$\mathbb{E}_{S_t}\left[f(w^{t+1})\right]{\leq} f(w^t){-}\rho\|\nabla f(w^t)\|^2,$$

where S_t is the set of K devices chosen at iteration t.

We direct the reader to Appendix A for a detailed proof. The key steps include applying our notion of γ-inexactness for each subproblem and using the bounded dissimilarity assumption, while allowing for only K devices to be active at each round. This last step in particular introduces $\mathbb{E}_{S_t}$, an expectation with respect to the choice of devices, S_t, in round t.

Theorem 3 uses the dissimilarity in Definition 2 to identify sufficient decrease at each iteration for `FedProx`. In Appendix B, we provide a corollary characterizing the performance with a more common (though slightly

more restrictive) bounded variance assumption. This assumption is commonly employed, e.g., when analyzing methods such as SGD.

We next provide sufficient conditions that ensure $\rho > 0$ in Theorem 3 so that sufficient decrease is attainable after each round.

Remark 4. *For ρ in Theorem 3 to be positive, we need $\gamma B < 1$ and $\frac{B}{\sqrt{K}} < 1$. These conditions help to quantify the trade-off between dissimilarity (B) and the algorithm parameters (γ, K).*

Finally, we can use the above sufficient decrease to the characterize the rate of convergence to the set of approximate stationary solutions $\mathcal{S}_s = \{w \mid \mathbb{E}\left[\|\nabla f(w)\|^2\right] \leq \epsilon\}$ under the bounded dissimilarity assumption, Assumption 1. Note that these results hold for general non-convex $F_k(\cdot)$.

Theorem 5 (Convergence rate: `FedProx`)**.** *Given some $\epsilon > 0$, assume that for $B \geq B_\epsilon$, μ, γ, and K the assumptions of Theorem 3 hold at each iteration of `FedProx`. Moreover, $f(w^0) - f^* = \Delta$. Then, after $T = O(\frac{\Delta}{\rho\epsilon})$ iterations of `FedProx`, we have $\frac{1}{T}\sum_{t=0}^{T-1} \mathbb{E}\left[\|\nabla f(w^t)\|^2\right] \leq \epsilon$.*

While the results thus far hold for non-convex $F_k(\cdot)$, we can also characterize the convergence for the special case of convex loss functions with exact minimization in terms of local objectives (Corollary 6). A proof is provided in Appendix C.

Corollary 6 (Convergence: Convex Case)**.** *Let the assertions of Theorem 3 hold. In addition, let $F_k(\cdot)$ be convex and $\gamma = 0$, i.e., all the local problems are solved exactly. If $1 \ll B \leq 0.5\sqrt{K}$, then we can choose $\mu \approx 6LB^2$ from which it follows that $\rho \approx \frac{1}{24LB^2}$.*

Remark 7 (Comparison with SGD)**.** *Note that `FedProx` achieves the same asymptotic convergence guarantee as SGD: Under the bounded variance assumption, for small ϵ, if we replace B_ϵ with its upper-bound in Corollary 8 and choose μ large enough, the iteration complexity of `FedProx` when the subproblems are solved exactly and $F_k(\cdot)$'s are convex is $O(\frac{L\Delta}{\epsilon} + \frac{L\Delta\sigma^2}{\epsilon^2})$, the same as SGD [8].*

To help provide context for the rate in Theorem 5, we compare it with SGD in the convex case in Remark 7. Note that small ϵ in Assumption 1 translates to larger B_ϵ. Corollary 6 suggests that, in order to solve the problem with increasingly higher accuracies using `FedProx`, one needs to increase μ appropriately. Moreover, in Corollary 6, if we plug in the upper bound for B_ϵ, under a bounded variance assumption (Corollary 8), the number of required steps to achieve accuracy ϵ is $O(\frac{L\Delta}{\epsilon} + \frac{L\Delta\sigma^2}{\epsilon^2})$. Our analysis helps to characterize the performance of `FedProx` and similar methods when local functions are dissimilar. As a future direction, it would be interesting to quantify lower bounds for the convergence of methods such as `FedProx`/`FedAvg` in heterogeneous settings.

5 Experiments

We now present empirical results for `FedProx`. In Section 5.2, we study the effect of statistical heterogeneity on the convergence of `FedAvg` and `FedProx`. We explore properties of the `FedProx` framework (the effect of μ and the local epochs E) in Section 5.3. Finally, in Section 5.4, we show how empirical convergence is related to the bounded dissimilarity assumption (Assumption 1, Corollary 8). We provide thorough details of the experimental setup in Section 5.1 and Appendix D. All code, data, and experiments are publicly available at github.com/litian96/FedProx.

5.1 Experimental Details

We evaluate `FedProx` on diverse tasks, models, and real-world federated datasets. In order to characterize statistical heterogeneity and study its effect on convergence, we also evaluate on a set of synthetic data, which allows for more precise manipulation of heterogeneity.

Synthetic data. To generate synthetic data, we follow a similar setup to that in [24], additionally imposing heterogeneity among devices. In particular, for each device k, we generate samples (X_k, Y_k) according to the model $y = argmax(\text{softmax}(Wx + b))$, $x \in \mathbb{R}^{60}, W \in \mathbb{R}^{10\times 60}, b \in \mathbb{R}^{10}$. We model $W_k \sim \mathcal{N}(u_k, 1)$, $b_k \sim \mathcal{N}(u_k, 1)$, $u_k \sim \mathcal{N}(0, \alpha)$; $x_k \sim \mathcal{N}(v_k, \Sigma)$, where the covariance matrix Σ is diagonal with $\Sigma_{j,j} = j^{-1.2}$. Each element in the mean vector v_k is drawn from $\mathcal{N}(B_k, 1), B_k \sim N(0, \beta)$. Therefore, α controls how much local models differ from each other and β controls how much the local data at each device differs from that of other devices. We vary α, β to generate three heterogeneous distributed datasets, denoted Synthetic (α, β), as shown in Figure 1. We also generate one IID dataset by setting the same W, b on all devices and setting X_k to follow the same distribution. Our goal is to learn a global W and b. Full details are given in Appendix D.1.

Real data. We also explore five real datasets; statistics are summarized in Table 1 in Appendix D.1. These datasets are curated from prior work in federated learning [4, 19]. We study two convex models on partitioned MNIST [15], Federated Extended MNIST [4, 5] (FEMNIST), and FMNIST*. We study two non-convex models on Sentiment140 [9] (Sent140) and *The Complete Works of William Shakespeare* [19] (Shakespeare). Details of datasets, models, and workloads are in Appendix D.1.

Setup. For each experiment, we tune the learning rate and ratio of active devices per round on `FedAvg`. For each comparison, we fix selected devices and mini-batch order across all runs. We report all metrics based on the global objective $f(w)$. We implement Algorithm 1 and 2 in Tensorflow [1]; full details are provided in Appendix D.2. Note that `FedAvg` ($\mu = 0$) and `FedProx` ($\mu \geq 0$) perform the same amount of work at each round when the number of local epochs, E, is the same—we therefore report results in terms of rounds rather than FLOPs or wall-clock time.

5.2 Effect of Statistical Heterogeneity

In Figure 1, we study how statistical heterogeneity affects convergence using four synthetic datasets (fixing E to be 20). From left to right, as data become more heterogeneous, convergence becomes worse for `FedProx`

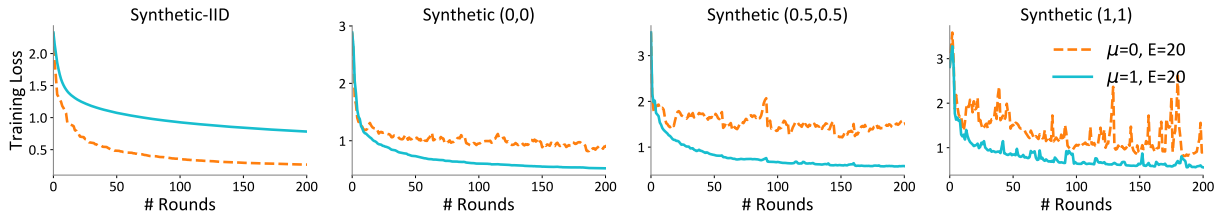

Figure 1: Effect of data heterogeneity on convergence. We show training loss (see testing accuracy and dissimilarity metric in Appendix D.3, Figure 8) on four synthetic datasets whose heterogeneity increases from left to right. Note that the method with $\mu = 0$ corresponds to `FedAvg`. Increasing heterogeneity leads to worse convergence, but setting $\mu > 0$ can help to combat this.

with μ=0 (i.e., `FedAvg`). Though it may slow convergence for IID data, we see that setting $\mu > 0$ is particularly useful in heterogeneous settings. This indicates that the modified subproblem introduced in `FedProx` can benefit practical federated settings with varying statistical heterogeneity. For perfect IID data which rarely occurs in practice, some heuristics such as decreasing μ if the loss continues to decrease may help avoid the deceleration of convergence (see Figure 7 in Appendix D.4). In the sections to follow, we see similar results in our non-synthetic experiments.

5.3 Properties of `FedProx` Framework

The key parameters of `FedProx` that affect performance are the number of local epochs, E, and the proximal term scaled by μ. Intuitively, large E may cause local models to drift too far away from the initial starting point, thus leading to potential divergence. Alternatively, large μ can restrict the trajectory of the iterates by constraining the iterates to be closer to that of the global model, potentially slowing convergence. We study `FedProx` under different values of E and μ using the federated datasets described in Table 1 in Appendix D.1.

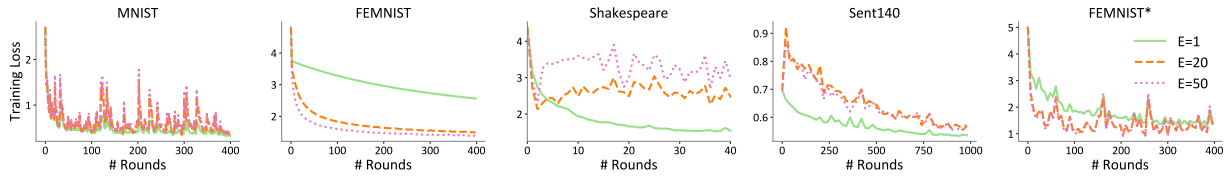

Figure 2: Effect of increasing E on real federated datasets where $\mu = 0$ (corresponds to `FedAvg`). Too many local updates can cause divergence or instability for heterogeneous datasets. Note that FEMNIST* is a more heterogeneous variant of FEMNIST.

Dependence on E. We explore the effect of E in Figure 2. For each dataset, we set E to be 1, 20, and 50 while keeping $\mu = 0$ (`FedProx` reduces to `FedAvg` in this case) and show the convergence in terms of the training loss. We see that large E leads to divergence or instability on MNIST and Shakespeare. On FEMNIST and Sent140, nevertheless, larger E speeds up the convergence. Based on conclusions drawn from Figure 1, we hypothesize this is due to the fact that the distributed FEMNIST and Sent140 datasets lack significant heterogeneity. We validate this hypothesis by observing instability on FEMNIST*, which is a skewed variant of the FEMNIST dataset. Moving forward, we therefore demonstrate the impact of μ using FEMNIST* instead of FEMNIST.

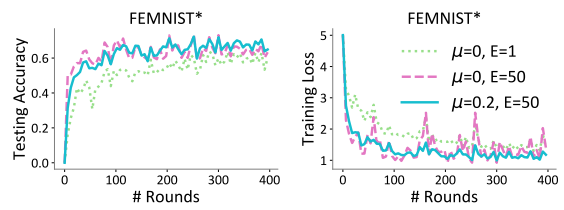

Figure 3: `FedProx` can provide faster and more stable convergence in communication-constrained environments (those requiring large E) with an appropriate μ.

We note that a potential approach to handle the divergence or instability of `FedAvg` with a large number of local epochs would be to *just keep E small*. However, this precludes the possibility of exactly solving the local subproblems. A large number of local epochs, E, may also be particularly useful in practice when communication is expensive (which is common in federated networks). Indeed, in such situations increasing μ can improve stability. In Figure 3, we show that `FedProx` with a large E (E=50) and an appropriate μ (μ=0.2) leads to faster and more stable convergence compared with E=1, μ=0 (slow convergence) and E=50, μ=0 (unstable convergence).

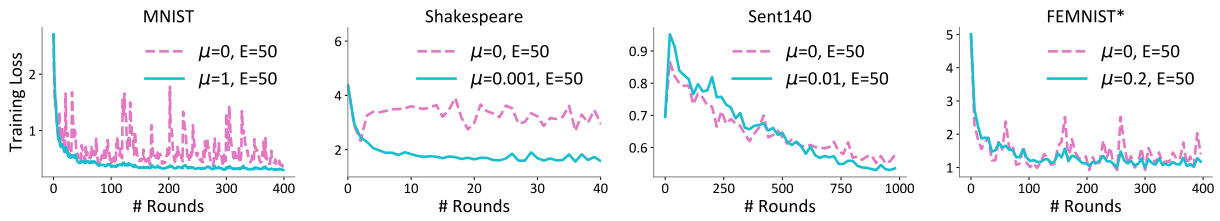

Figure 4: Effect of μ on real datasets. The setting $\mu = 0$ corresponds to `FedAvg`. `FedProx` with $\mu > 0$ leads to more stable convergence and enables otherwise divergent methods to converge.

Dependence on μ. We consider the effect of μ on convergence in Figure 4. For each experiment, we compare the results between $\mu = 0$ and the best μ. For all datasets, we observe that the appropriate μ can increase the stability for unstable methods and can force divergent methods to converge. It also increases the accuracy in most cases (see Figure 8 and Figure 10 in Appendix D.3). In practice, μ can be adaptively chosen based on the current performance of the model. For example, one simple heuristic is to increase μ when seeing the loss increasing and decreasing μ when seeing the loss decreasing. We provide additional experiments demonstrating the effectiveness of this approach in Appendix D.4. Future work includes developing methods to automatically tune this parameter for heterogeneous datasets, based, e.g., on the theoretical groundwork provided here.

5.4 Dissimilarity Measurement and Divergence

Finally, in Figure 5, we demonstrate that our B-local dissimilarity measurement in Definition 2 captures the heterogeneity of datasets and is therefore an appropriate proxy of performance. In particular, we track the variance of gradients on each device, $E_k[\|\nabla F_k(w) - \nabla f(w)\|^2]$, which is lower bounded by B_ϵ (see Bounded Variance Equivalence Corollary 8). Empirically, either decreasing E (Figure 9 in Appendix D.3) or increasing μ (Figure 5) leads to smaller dissimilarity among local functions. We also observe that the dissimilarity metric is consistent with the training loss. Therefore, smaller dissimilarity indicates better convergence, which can be enforced by setting μ appropriately. Full results tracking B (for all experiments performed) are provided in Appendix D.3.

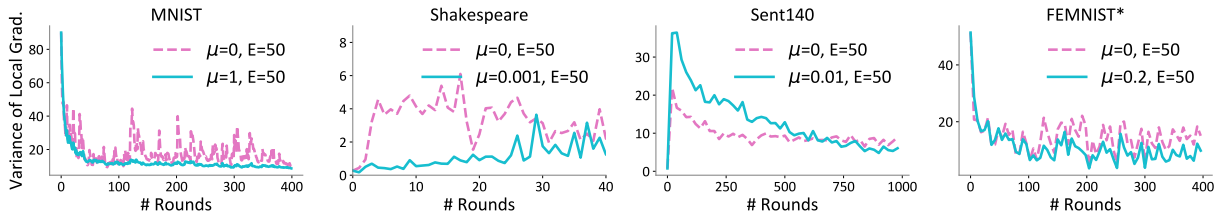

Figure 5: The dissimilarity measurement (variance of gradients) on four federated datasets. This metric captures statistical heterogeneity and is consistent with training loss (Figure 4). Smaller dissimilarity indicates better convergence.

6 Conclusion

In this work, we have proposed `FedProx`, a distributed optimization framework that tackles the statistical heterogeneity inherent in federated networks. We have formalized statistical heterogeneity through a *device dissimilarity* assumption which allowed us to characterize the convergence of `FedProx`. Our empirical evaluation across a suite of federated datasets has validated our theoretical analysis and demonstrated that the `FedProx` framework can improve convergence behavior in realistic heterogeneous federated networks.

Acknowledgement. We thank Jakub Konečný, Brendan McMahan, Nathan Srebro, and Jianyu Wang for their helpful discussions. This work was supported in part by DARPA FA875017C0141, the National Science Foundation grants IIS1705121 and IIS1838017, an Okawa Grant, a Google Faculty Award, an Amazon Web Services Award, a Carnegie Bosch Institute Research Award, and the CONIX Research Center. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of DARPA, the National Science Foundation, or any other funding agency.

References

[1] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. K. Kudlur, J. Levenberg, R. Monga, S. Moore, D. G. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, and X. Zheng. Tensorflow: A system for large-scale machine learning. In *Operating Systems Design and Implementation*, pages 265–283, 2016.

[2] Z. Allen-Zhu. How to make the gradients small stochastically: Even faster convex and nonconvex sgd. In *Advances in Neural Information Processing Systems*, pages 1157–1167, 2018.

[3] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2010.

[4] S. Caldas, P. Wu, T. Li, J. Konečnỳ, H. B. McMahan, V. Smith, and A. Talwalkar. Leaf: A benchmark for federated settings. *arXiv preprint arXiv:1812.01097*, 2018.

[5] G. Cohen, S. Afshar, J. Tapson, and A. van Schaik. Emnist: an extension of mnist to handwritten letters. *arXiv preprint arXiv:1702.05373*, 2017.

[6] J. Dean, G. Corrado, R. Monga, K. Chen, M. Devin, Q. V. Le, M. Mao, M. Ranzato, A. Senior, P. Tucker, K. Yang, and A. Ng. Large scale distributed deep networks. In *Advances in Neural Information Processing Systems*, pages 1223–1231, 2012.

[7] O. Dekel, R. Gilad-Bachrach, O. Shamir, and L. Xiao. Optimal Distributed Online Prediction Using Mini-Batches. *Journal of Machine Learning Research*, 13:165–202, 2012.

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

[9] A. Go, R. Bhayani, and L. Huang. Twitter sentiment classification using distant supervision. *CS224N Project Report, Stanford*, 1(12), 2009.

[10] Y. Hao, J. Rong, and Y. Sen. On the linear speedup analysis of communication efficient momentum sgd for distributed non-convex optimization. In *International Conference on Machine Learning*, 2019.

[11] L. Huang, Y. Yin, Z. Fu, S. Zhang, H. Deng, and D. Liu. Loadaboost: Loss-based adaboost federated machine learning on medical data. *arXiv preprint arXiv:1811.12629*, 2018.

[12] E. Jeong, S. Oh, H. Kim, J. Park, M. Bennis, and S.-L. Kim. Communication-efficient on-device machine learning: Federated distillation and augmentation under non-iid private data. *arXiv preprint arXiv:1811.11479*, 2018.

[13] P. Jiang and G. Agrawal. A linear speedup analysis of distributed deep learning with sparse and quantized communication. In *Advances in Neural Information Processing Systems*, pages 2525–2536, 2018.

[14] S. Kaczmarz. Approximate solution of systems of linear equations. *International Journal of Control*, 57(6): 1269–1271, 1993.

[15] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[16] M. Li, D. G. Andersen, A. J. Smola, and K. Yu. Communication efficient distributed machine learning with the parameter server. In *Advances in Neural Information Processing Systems*, pages 19–27, 2014.

[17] M. Li, T. Zhang, Y. Chen, and A. J. Smola. Efficient mini-batch training for stochastic optimization. In *Conference on Knowledge Discovery and Data Mining*, pages 661–670, 2014.

[18] T. Lin, S. U. Stich, and M. Jaggi. Don't use large mini-batches, use local sgd. *arXiv preprint arXiv:1808.07217*, 2018.

[19] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y. Arcas. Communication-efficient learning of deep networks from decentralized data. In *International Conference on Artificial Intelligence and Statistics*, 2017.

[20] J. Pennington, R. Socher, and C. Manning. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing*, pages 1532–1543, 2014.

[21] S. J. Reddi, J. Konečnỳ, P. Richtárik, B. Póczós, and A. Smola. Aide: Fast and communication efficient distributed optimization. *arXiv preprint arXiv:1608.06879*, 2016.

[22] P. Richtárik and M. Takáč. Distributed coordinate descent method for learning with big data. *Journal of Machine Learning Research*, 17:1–25, 2016.

[23] M. Schmidt and N. L. Roux. Fast convergence of stochastic gradient descent under a strong growth condition. *arXiv preprint arXiv:1308.6370*, 2013.

[24] O. Shamir, N. Srebro, and T. Zhang. Communication-efficient distributed optimization using an approximate newton-type method. In *International Conference on Machine Learning*, pages 1000–1008, 2014.

[25] V. Smith, C.-K. Chiang, M. Sanjabi, and A. S. Talwalkar. Federated multi-task learning. In *Advances in Neural Information Processing Systems*, pages 4424–4434, 2017.

[26] V. Smith, S. Forte, C. Ma, M. Takac, M. I. Jordan, and M. Jaggi. Cocoa: A general framework for communication-efficient distributed optimization. *Journal of Machine Learning Research*, 18:1–47, 2018.

[27] S. U. Stich. Local sgd converges fast and communicates little. In *International Conference on Learning Representations*, 2019.

[28] S. Vaswani, F. Bach, and M. Schmidt. Fast and faster convergence of sgd for over-parameterized models (and an accelerated perceptron). In *International Conference on Artificial Intelligence and Statistics*, 2019.

[29] J. Wang and G. Joshi. Cooperative sgd: A unified framework for the design and analysis of communication-efficient sgd algorithms. *arXiv preprint arXiv:1808.07576*, 2018.

[30] S. Wang, T. Tuor, T. Salonidis, K. K. Leung, C. Makaya, T. He, and K. Chan. Adaptive federated learning in resource constrained edge computing systems. *IEEE Journal on Selected Areas in Communications*, 2019.

[31] B. E. Woodworth, J. Wang, A. Smith, B. McMahan, and N. Srebro. Graph oracle models, lower bounds, and gaps for parallel stochastic optimization. In *Advances in Neural Information Processing Systems*, pages 8496–8506, 2018.

[32] Y. Yao, L. Rosasco, and A. Caponnetto. On early stopping in gradient descent learning. *Constructive Approximation*, 26(2):289–315, 2007.

[33] D. Yin, A. Pananjady, M. Lam, D. Papailiopoulos, K. Ramchandran, and P. Bartlett. Gradient diversity: a key ingredient for scalable distributed learning. In *International Conference on Artificial Intelligence and Statistics*, pages 1998–2007, 2018.

[34] H. Yu, S. Yang, and S. Zhu. Parallel restarted sgd for non-convex optimization with faster convergence and less communication. In *AAAI Conference on Artificial Intelligence*, 2018.

[35] S. Zhang, A. E. Choromanska, and Y. LeCun. Deep learning with elastic averaging sgd. In *Advances in Neural Information Processing Systems*, pages 685–693, 2015.

[36] Y. Zhang, J. C. Duchi, and M. J. Wainwright. Communication-efficient algorithms for statistical optimization. *Journal of Machine Learning Research*, 14:3321–3363, 2013.

[37] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra. Federated learning with non-iid data. *arXiv preprint arXiv:1806.00582*, 2018.

[38] F. Zhou and G. Cong. On the convergence properties of a k-step averaging stochastic gradient descent algorithm for nonconvex optimization. In *International Joint Conference on Artificial Intelligence*, pages 3219–3227, 2018.

A Proof of Theorem 3

Proof. Using our notion of γ-inexactness for each local solver (Definition 1), we can define e_k^{t+1} such that:

$$\begin{aligned}&\nabla F_k(w_k^{t+1}) + \mu(w_k^{t+1} - w^t) - e_k^{t+1} = 0,\\ &\|e_k^{t+1}\| \leq \gamma \|\nabla F_k(w^t)\|.\end{aligned} \tag{3}$$

Now let us define $\bar{w}^{t+1} = \mathbb{E}_k\big[w_k^{t+1}\big]$. Based on this definition, we know

$$\bar{w}^{t+1} - w^t = \frac{-1}{\mu}\mathbb{E}_k\big[\nabla F_k(w_k^{t+1})\big] + \frac{1}{\mu}\mathbb{E}_k\big[e_k^{t+1}\big]. \tag{4}$$

Let us define $\bar{\mu} = \mu - L_- > 0$ and $\hat{w}_k^{t+1} = \arg\min_w \ h_k(w; w^t)$. Then, due to the $\bar{\mu}$-strong convexity of h_k, we have

$$\|\hat{w}_k^{t+1} - w_k^{t+1}\| \leq \frac{\gamma}{\bar{\mu}}\|\nabla F_k(w^t)\|. \tag{5}$$

Note that once again, due to the $\bar{\mu}$-strong convexity of h_k, we know that $\|\hat{w}_k^{t+1} - w^t\| \leq \frac{1}{\bar{\mu}}\|\nabla F_k(w^t)\|$. Now we can use the triangle inequality to get

$$\|w_k^{t+1} - w^t\| \leq \frac{1+\gamma}{\bar{\mu}}\|\nabla F_k(w^t)\|. \tag{6}$$

Therefore,

$$\begin{aligned}&\|\bar{w}^{t+1} - w^t\| \leq \mathbb{E}_k\big[\|w_k^{t+1} - w^t\|\big] \leq \frac{1+\gamma}{\bar{\mu}}\mathbb{E}_k\big[\|\nabla F_k(w^t)\|\big]\\ &\leq \frac{1+\gamma}{\bar{\mu}}\sqrt{\mathbb{E}_k[\|\nabla F_k(w^t)\|^2]} \leq \frac{B(1+\gamma)}{\bar{\mu}}\|\nabla f(w^t)\|,\end{aligned} \tag{7}$$

where the last inequality is due to the bounded dissimilarity assumption.

Now let us define M_{t+1} such that $\bar{w}^{t+1} - w^t = \frac{-1}{\mu}\big(\nabla f(w^t) + M_{t+1}\big)$, i.e. $M_{t+1} = \mathbb{E}_k\big[\nabla F_k(w_k^{t+1}) - \nabla F_k(w^t) - e_k^{t+1}\big]$. We can bound $\|M_{t+1}\|$:

$$\|M_{t+1}\| \leq \mathbb{E}_k\big[L\|w_k^{t+1} - w_k^t\| + \|e_k^{t+1}\|\big] \leq \left(\frac{L(1+\gamma)}{\bar{\mu}} + \gamma\right) \times \mathbb{E}_k\big[\|\nabla F_k(w^t)\|\big] \tag{8}$$

$$\leq \left(\frac{L(1+\gamma)}{\bar{\mu}} + \gamma\right) B\|\nabla f(w^t)\|, \tag{9}$$

where the last inequality is also due to bounded dissimilarity assumption. Based on the L-Lipschitz smoothness of f and Taylor expansion, we have

$$\begin{aligned}&f(\bar{w}^{t+1}) \leq f(w^t) + \langle \nabla f(w^t), \bar{w}^{t+1} - w^t\rangle + \frac{L}{2}\|\bar{w}^{t+1} - w^t\|^2\\ &\leq f(w^t) - \frac{1}{\mu}\|\nabla f(w^t)\|^2 - \frac{1}{\mu}\langle \nabla f(w^t), M_{t+1}\rangle + \frac{L(1+\gamma)^2B^2}{2\bar{\mu}^2}\|\nabla f(w^t)\|^2\\ &\leq f(w^t) - \left(\frac{1-\gamma B}{\mu} - \frac{LB(1+\gamma)}{\bar{\mu}\mu} - \frac{L(1+\gamma)^2B^2}{2\bar{\mu}^2}\right) \times \|\nabla f(w^t)\|^2.\end{aligned} \tag{10}$$

From the above inequality it follows that if we set the penalty parameter μ large enough, we can get a decrease in the objective value of $f(\bar{w}^{t+1}) - f(w^t)$ which is proportional to $\|\nabla f(w^t)\|^2$. However, this is not the way that the algorithm works. In the algorithm, we only use K devices that are chosen randomly to approximate $\bar{w}^t$. So, in order to find the $\mathbb{E}\big[f(w^{t+1})\big]$, we use local Lipschitz continuity of the function f.

$$f(w^{t+1}) \leq f(\bar{w}^{t+1}) + L_0\|w^{t+1} - \bar{w}^{t+1}\|, \tag{11}$$

where L_0 is the local Lipschitz continuity constant of function f and we have

$$\begin{aligned} L_0 &\leq \|\nabla f(w^t)\| + L\max(\|\bar{w}^{t+1} - w^t\|, \|w^{t+1} - w^t\|) \\ &\leq \|\nabla f(w^t)\| + L(\|\bar{w}^{t+1} - w^t\| + \|w^{t+1} - w^t\|). \end{aligned} \tag{12}$$

Therefore, if we take expectation with respect to the choice of devices in round t we need to bound

$$\mathbb{E}_{S_t}\big[f(w^{t+1})\big] \leq f(\bar{w}^{t+1}) + Q_t, \tag{13}$$

where $Q_t = \mathbb{E}_{S_t}\big[L_0\|w^{t+1} - \bar{w}^{t+1}\|\big]$. Note that the expectation is taken over the random choice of devices to update.

$$\begin{aligned} Q_t &\leq \mathbb{E}_{S_t}\left[\Big(\|\nabla f(w^t)\| + L(\|\bar{w}^{t+1} - w^t\| + \|w^{t+1} - w^t\|)\Big) \times \|w^{t+1} - \bar{w}^{t+1}\|\right] \\ &\leq \Big(\|\nabla f(w^t)\| + L\|\bar{w}^{t+1} - w^t\|\Big)\mathbb{E}_{S_t}\big[\|w^{t+1} - \bar{w}^{t+1}\|\big] + L\mathbb{E}_{S_t}\big[\|w^{t+1} - w^t\| \cdot \|w^{t+1} - \bar{w}^{t+1}\|\big] \\ &\leq \Big(\|\nabla f(w^t)\| + 2L\|\bar{w}^{t+1} - w^t\|\Big)\mathbb{E}_{S_t}\big[\|w^{t+1} - \bar{w}^{t+1}\|\big] + L\mathbb{E}_{S_t}\big[\|w^{t+1} - \bar{w}^{t+1}\|^2\big] \end{aligned} \tag{14}$$

From (7), we have that $\|\bar{w}^{t+1} - w^t\| \leq \frac{B(1+\gamma)}{\bar{\mu}}\|\nabla f(w^t)\|$. Moreover,

$$\mathbb{E}_{S_t}\big[\|w^{t+1} - \bar{w}^{t+1}\|\big] \leq \sqrt{\mathbb{E}_{S_t}[\|w^{t+1} - \bar{w}^{t+1}\|^2]} \tag{15}$$

and

$$\begin{aligned} \mathbb{E}_{S_t}\big[\|w^{t+1} - \bar{w}^{t+1}\|^2\big] &\leq \frac{1}{K}\mathbb{E}_k\big[\|w_k^{t+1} - \bar{w}^{t+1}\|^2\big] \\ &\leq \frac{2}{K}\mathbb{E}_k\big[\|w_k^{t+1} - w^t\|^2\big], \quad (\text{as } \bar{w}^{t+1} = \mathbb{E}_k\big[w_k^{t+1}\big]) \\ &\leq \frac{2}{K}\frac{(1+\gamma)^2}{\bar{\mu}^2}\mathbb{E}_k\big[\|\nabla F_k(w^t)\|^2\big] \quad (\text{from (6)}) \\ &\leq \frac{2B^2}{K}\frac{(1+\gamma)^2}{\bar{\mu}^2}\|\nabla f(w^t)\|^2, \end{aligned} \tag{16}$$

where the first inequality is a result of K devices being chosen randomly to get w^t and the last inequality is due to bounded dissimilarity assumption. If we replace these bounds in (14) we get

$$Q_t \leq \left(\frac{B(1+\gamma)\sqrt{2}}{\bar{\mu}\sqrt{K}} + \frac{LB^2(1+\gamma)^2}{\bar{\mu}^2 K}\left(2\sqrt{2K} + 2\right)\right)\|\nabla f(w^t)\|^2 \tag{17}$$

Combining (10), (13), (11) and (17) and using the notation $\alpha = \frac{1}{\mu}$ we get

$$\begin{aligned} \mathbb{E}_{S_t}\big[f(w^{t+1})\big] \leq f(w^t) - \Bigg(&\frac{1}{\mu} - \frac{\gamma B}{\mu} - \frac{B(1+\gamma)\sqrt{2}}{\bar{\mu}\sqrt{K}} - \frac{LB(1+\gamma)}{\bar{\mu}\mu} \\ &- \frac{L(1+\gamma)^2B^2}{2\bar{\mu}^2} - \frac{LB^2(1+\gamma)^2}{\bar{\mu}^2 K}\left(2\sqrt{2K} + 2\right)\Bigg)\|\nabla f(w^t)\|^2. \end{aligned}$$

□

B Proof for Bounded Variance

Theorem 3 uses the dissimilarity in Definition 2 to identify sufficient decrease at each iteration for `FedProx`. Here we provide a corollary characterizing the performance with a more common (though slightly more restrictive) bounded variance assumption. This assumption is commonly employed, e.g., when analyzing methods such as SGD.

Corollary 8 (Bounded Variance Equivalence)**.** *Let Assumption 1 hold. Then, in the case of bounded variance, i.e.,* $\mathbb{E}_k\left[\|\nabla F_k(w) - \nabla f(w)\|^2\right] \leq \sigma^2$*, for any* $\epsilon > 0$ *it follows that* $B_\epsilon \leq \sqrt{1 + \frac{\sigma^2}{\epsilon}}$*.*

Proof. We have,

$$\begin{aligned}
&E_k[\|\nabla F_k(w) - \nabla f(w)\|^2] = E_k[\|\nabla F_k(w)\|^2] - \|\nabla f(w)\|^2 \leq \sigma^2 \\
&\Rightarrow E_k[\|\nabla F_k(w)\|^2] \leq \sigma^2 + \|\nabla f(w)\|^2 \\
&\Rightarrow B_\epsilon = \sqrt{\frac{E_k[\|\nabla F_k(w)\|^2]}{\|\nabla f(w)\|^2}} \leq \sqrt{1 + \frac{\sigma^2}{\epsilon}}.
\end{aligned}$$

With Corollary 8 in place, we can restate the main result in Theorem 3 in terms of the bounded variance assumption.

Theorem 9 (Non-Convex `FedProx` Convergence: Bounded Variance)**.** *Let the assertions of Theorem 3 hold. In addition, let the iterate* w^t *be such that* $\|\nabla f(w^t)\|^2 \geq \epsilon$*, and let* $\mathbb{E}_k\left[\|\nabla F_k(w) - \nabla f(w)\|^2\right] \leq \sigma^2$ *hold instead of the dissimilarity condition. If* μ*,* K *and* γ *in Algorithm 2 are chosen such that*

$$\begin{aligned}
\rho = &\left(\frac{1}{\mu} - \left(\frac{\gamma}{\mu} + \frac{(1+\gamma)\sqrt{2}}{\bar{\mu}\sqrt{K}} + \frac{L(1+\gamma)}{\bar{\mu}\mu}\right)\sqrt{1+\frac{\sigma^2}{\epsilon}}\right. \\
&\left. - \left(\frac{L(1+\gamma)^2}{2\bar{\mu}^2} + \frac{L(1+\gamma)^2}{\bar{\mu}^2 K}\left(2\sqrt{2K}+2\right)\right)\left(1+\frac{\sigma^2}{\epsilon}\right)\right) > 0,
\end{aligned}$$

then at iteration t *of Algorithm 2, we have the following expected decrease in the global objective:*

$$\mathbb{E}_{S_t}\left[f(w^{t+1})\right] \leq f(w^t) - \rho\|\nabla f(w^t)\|^2,$$

where S_t *is the set of* K *devices chosen at iteration* t*.*

The proof of Theorem 9 follows from the proof of Theorem 3 by noting the relationship between the bounded variance assumption and the dissimilarity assumption as portrayed by Corollary 8.

C Proof of Corollary 6

In the convex case, where $L_- = 0$ and $\bar{\mu} = \mu$, if $\gamma = 0$, i.e., all subproblems are solved accurately, we can get a decrease proportional to $\|\nabla f(w^t)\|^2$ if $B < \sqrt{K}$. In such a case if we assume $1 << B \leq 0.5\sqrt{K}$, then we can write

$$\mathbb{E}_{S_t}\left[f(w^{t+1})\right] \lessapprox f(w^t) - \frac{1}{2\mu}\|\nabla f(w^t)\|^2 + \frac{3LB^2}{2\mu^2}\|\nabla f(w^t)\|^2. \tag{18}$$

In this case, if we choose $\mu \approx 6LB^2$ we get

$$\mathbb{E}_{S_t}\left[f(w^{t+1})\right] \lessapprox f(w^t) - \frac{1}{24LB^2}\|\nabla f(w^t)\|^2. \tag{19}$$

Note that the expectation in (19) is a conditional expectation conditioned on the previous iterate. Taking expectation of both sides, and telescoping, we have that the number of iterations to at least generate one solution with squared norm of gradient less than ϵ is $O(\frac{LB^2\Delta}{\epsilon})$.

D Experimental Details

D.1 Datasets and Models

Here we provide full details on the datasets and models used in our experiments. We curate a diverse set of non-synthetic datasets, including those used in prior work on federated learning [19], and some proposed in LEAF, a benchmark for federated settings [4]. We also create synthetic data to directly test the effect of heterogeneity on convergence, as in Section 5.1.

- **Synthetic:** We set (α, β)=(0,0), (0.5,0.5) and (1,1) respectively to generate three non-identical distributed datasets (Figure 1). In the IID data, we set the same $W, b \sim \mathcal{N}(0, 1)$ on all devices and X_k to follow the same distribution $\mathcal{N}(v, \Sigma)$ where each element in the mean vector v is zero and Σ is diagonal with $\Sigma_{j,j} = j^{-1.2}$. For all synthetic datasets, there are 30 devices in total and the number of samples on each device follows a power law.
- **MNIST:** We study image classification of handwritten digits 0-9 in MNIST [15] using multinomial logistic regression. To simulate a heterogeneous setting, we distribute the data among 1000 devices such that each device has samples of only 2 digits and the number of samples per device follows a power law. The input of the model is a flattened 784-dimensional (28×28) image, and the output is a class label between 0 and 9.
- **FEMNIST:** We study an image classification problem on the 62-class EMNIST dataset [5] using multinomial logistic regression. Each device corresponds to a writer of the digits/characters in EMNIST. We call this *federated* version of EMNIST *FEMNIST*. The input of the model is a flattened 784-dimensional (28×28) image, and the output is a class label between 0 and 61.
- **Shakespeare:** This is a dataset built from *The Complete Works of William Shakespeare* [19]. Each speaking role in a play represents a different device. We use a two-layer LSTM classifier containing 100 hidden units with an 8D embedding layer. The task is next character prediction, and there are 80 classes of characters in total. The model takes as input a sequence of 80 characters, embeds each of the characters into a learned 8-dimensional space and outputs one character per training sample after 2 LSTM layers and a densely-connected layer.
- **Sent140:** In non-convex settings, we consider a text sentiment analysis task on tweets from Sentiment140 [9] (Sent140) with a two layer LSTM binary classifier containing 256 hidden units with pretrained 300D GloVe embedding [20]. Each twitter account corresponds to a device. The model takes as input a sequence of 25 characters, embeds each of the character into a 300-dimensional space by looking up Glove and outputs one character per training sample after 2 LSTM layers and a densely-connected layer.
- **FEMNIST*:** We generate FEMNIST* by subsampling 26 lower case characters from FEMNIST and distributing only 20 classes to each device. There are 200 devices in total. The model is the same as the one used on FEMNIST.

D.1.1 Statistics of federated datasets

We report the total number of devices, samples, and the mean and standard deviation of samples per device of real federated datasets in Table 1.

Table 1: Statistics of Federated Datasets

Dataset	Devices	Samples	Samples/device	
			mean	stdev
MNIST	1,000	69,035	69	106
FEMNIST	900	305,654	340	107
Shakespeare	143	517,706	3,620	4,115
Sent140	5,726	215,829	38	19
FEMNIST*	200	79,059	395	873

D.2 Implementation Details

(**Implementation**) In order to draw a fair comparison with `FedAvg`, we use SGD as a local solver for `FedProx`, and adopt a slightly different device sampling scheme than that in Algorithms 1 and 2: sampling devices uniformly and averaging updates with weights proportional to the number of local data points (as originally proposed in [19]). While this sampling scheme is not supported by our analysis, we observe similar relative behavior of `FedProx` vs. `FedAvg` whether or not it is employed (Figure 6). Interestingly, we also observe that the sampling scheme proposed herein results in more stable performance for both methods. This suggests an added benefit of the proposed framework.

(**Machines**) We simulate the federated learning setup (1 server and N devices) on a commodity machine with 2 Intel® Xeon® E5-2650 v4 CPUs and 8 NVidia® 1080Ti GPUs. Note that `FedAvg` ($\mu = 0$) and `FedProx` ($\mu \geq 0$) perform the same amount of work at each round when the number of local epochs, E, is the same—we therefore report results in terms of rounds rather than FLOPs or wall-clock time.

(**Hyperparameters**) We randomly split the data on each local device into 80% training set and 20% testing set. For each dataset, we tune the ratio of active clients per round from $\{0.01, 0.05, 0.1\}$ on `FedAvg`. For synthetic datasets, roughly 10% of the devices are active at each round. For MNIST, FEMNIST, Shakespeare, Sent140 and FEMNIST*, the numbers of active devices (K) are 1%, 5%, 10%, 1%, and 5% respectively. We also do a grid search on the learning rate based on `FedAvg`. We do not decay the learning rate through all rounds. For all synthetic data experiments, the learning rate is 0.01. For MNIST, FEMNIST, Shakespeare, Sent140 and FEMNIST*, we use the learning rates of 0.03, 0.003, 0.8, 0.3 and 0.003. We use a batch size of 10 for all experiments.

(**Libraries**) All code is implemented in Tensorflow [1] Version 1.10.1. Please see github.com/litian96/FedProx for full details.

D.3 Full Experiments

We demonstrate effects of two sampling schemes (as mentioned earlier in Section D.2) in Figure 6. We present testing accuracy, training loss and dissimilarity measurements of all the experiments in Figure 8, Figure 9 and Figure 10.

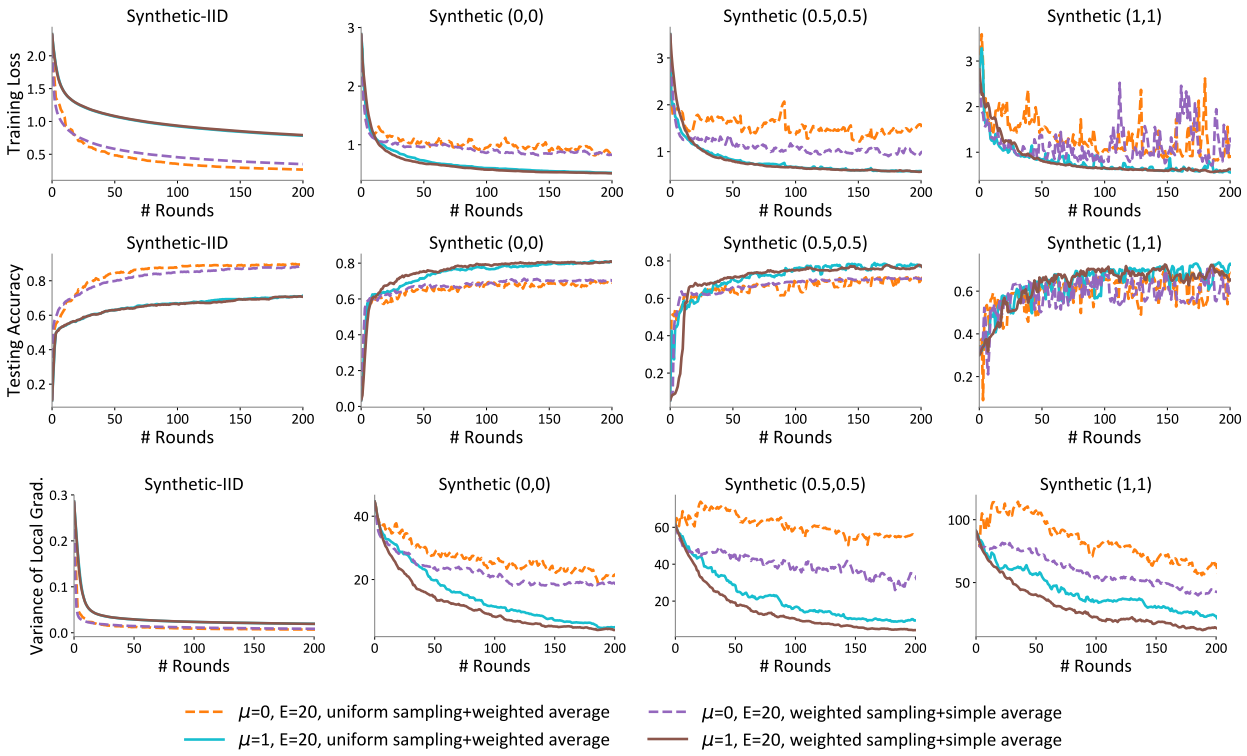

Figure 6: Differences between two sampling schemes in terms of training loss, testing accuracy, and dissimilarity measurement. Sampling devices with a probability proportional to the number of local data points and then simply averaging local models performs slightly better than uniformly sampling devices and averaging the local models with weights proportional to the number of local data points. Under either sampling scheme, the settings with $\mu = 1$ demonstrate more stable performance than settings with $\mu = 0$.

D.4 Adaptively setting μ

One of the key parameters of `FedProx` is μ. We demonstrate the results of a simple heuristic of adaptively setting μ on four synthetic datasets in Figure 7. For the IID dataset (Synthetic-IID), μ starts from 1, and for the other non-IID datasets, μ starts from 0. Such initialization is adversarial to our methods. We decrease μ by 0.1 when the loss continues to decrease for 5 rounds and increase μ by 0.1 when we see the loss increase. This heuristic allows for competitive performance. It could also alleviate the potential issue that $\mu > 0$ might slow down convergence on IID data, which rarely occurs in real federated settings.

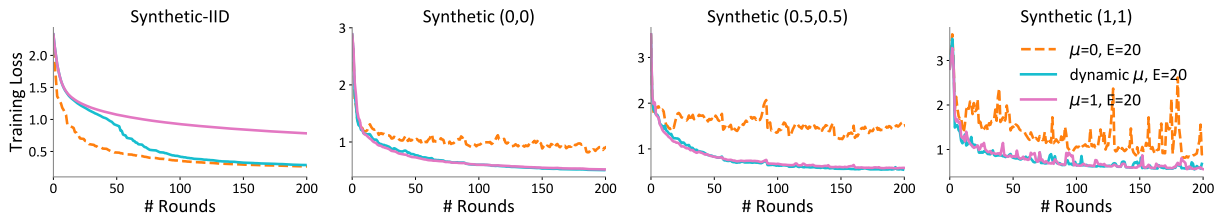

Figure 7: Results of choosing μ adaptively. We increase μ by 0.1 whenever the loss increases and decreases it by 0.1 whenever the loss decreases for 5 consecutive rounds. We initialize μ to 1 for the IID data (Synthetic-IID) (in order to be adversarial to our methods), and initialize it to 0 for the other three non-IID datasets. We observe that this simple heuristic works well in practice.

E Connections to other single-machine and distributed methods

Two aspects of the proposed work—the proximal term in `FedProx`, and the bounded dissimilarity assumption used in our analysis—have been previously studied in the optimization literature, but with very different motivations. For completeness, we provide a discussion below on our relation to these prior works.

Proximal term. The proposed modified objective in `FedProx` shares a connection with elastic averaging SGD (EASGD) [35], which was proposed as a way to train deep networks in the data center setting, and uses a similar proximal term in its objective. While the intuition is similar to EASGD (this term helps to prevent large deviations on each device/machine), EASGD employs a more complex moving average to update parameters, is limited to using SGD as a local solver, and has only been analyzed for simple quadratic problems. The proximal term we introduce has also been explored in previous optimization literature with different purposes, such as [2], to speed up (mini-batch) SGD training on a single machine, and in [17] for efficient SGD training both in a single machine and distributed settings. However, the analysis in [17] is limited to a single machine setting with different assumptions (e.g., IID data and solving the subproblem exactly at each round). Finally, DANE [24] and AIDE [21], distributed methods designed for the data center setting, propose a similar proximal term in the local objective function, but also augment this with an additional gradient correction term. Both methods assume that all devices participate at each communication round, which is impractical in federated settings. Indeed, due to the inexact estimation of full gradients (i.e., $\nabla\phi(w^{(t-1)})$ in [24, Eq (13)]) with device subsampling schemes and the staleness of the gradient correction term [24, Eq (13)], these methods are not directly applicable to our setting. Regardless of this, we explore a variant of such an approach in federated settings and see that the gradient direction term does not help in this scenario—performing uniformly worse than the proposed `FedProx` framework for heterogeneous datasets, despite the extra computation required (see Figure 11).

Bounded dissimilarity assumption. The bounded dissimilarity assumption we discuss in Assumption 1 has appeared in different forms, for example in [23, 28, 33]. In [33], the bounded similarity assumption is used in the context of asserting gradient diversity and quantifying the benefit in terms of scaling of the mean square error for mini-batch SGD for IID data. In [23, 28], the authors use a similar assumption, called *strong growth condition*, which is a stronger version of Assumption 1 with $\epsilon = 0$. They prove that some interesting practical problems satisfy such a condition. They also use this assumption to prove optimal and better convergence rates for SGD with constant step-sizes. Note that this is different from our approach as the algorithm that we are analyzing is not SGD, and our analysis is different in spite of the similarity in the assumptions.

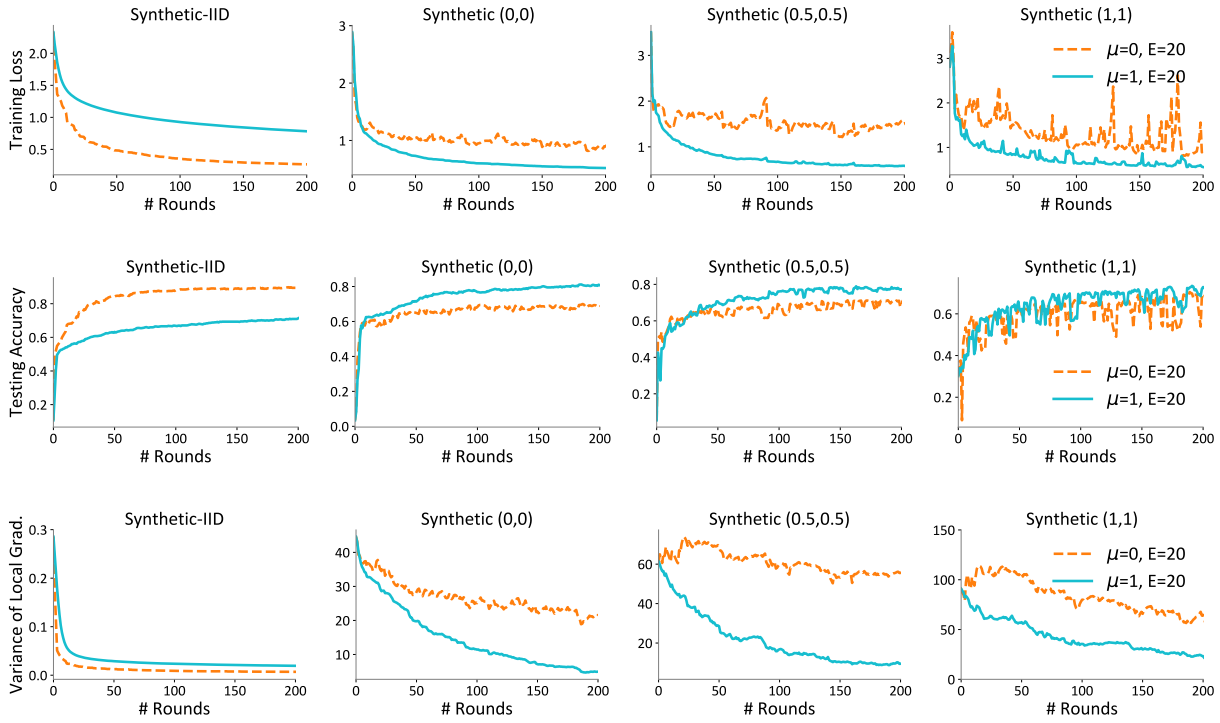

Figure 8: Training loss, testing accuracy and dissimilarity measurement for experiments in Figure 1

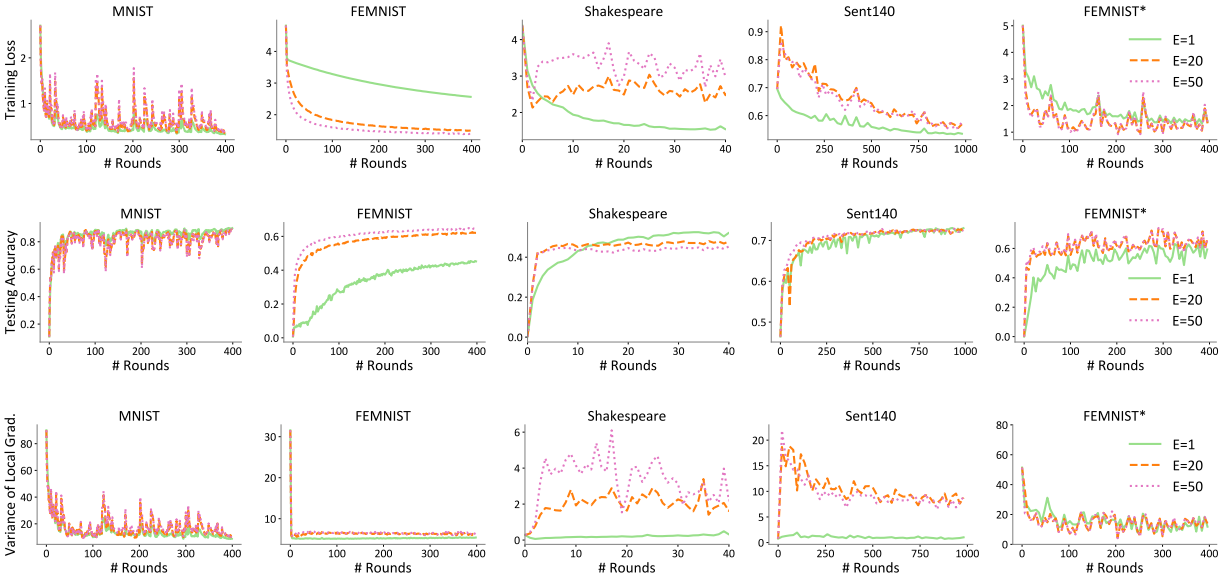

Figure 9: Training loss, testing accuracy and dissimilarity measurement for experiments in Figure 2

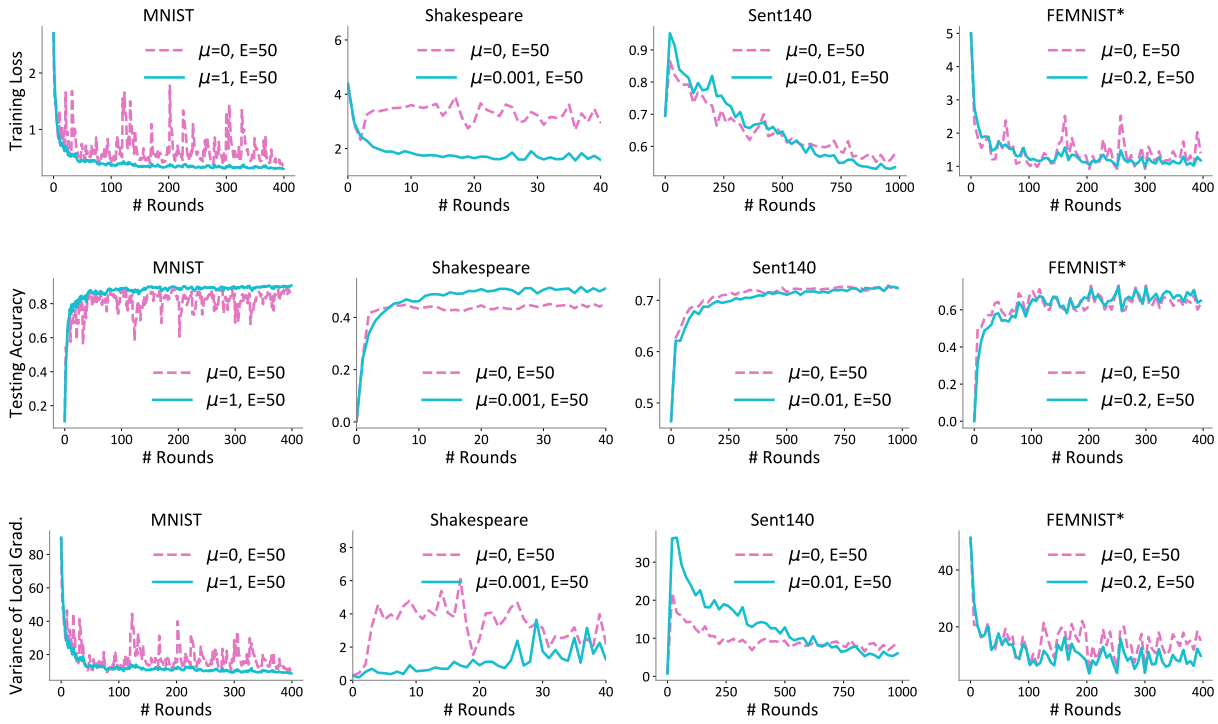

Figure 10: Training loss, testing accuracy and dissimilarity measurement for experiments in Figure 4

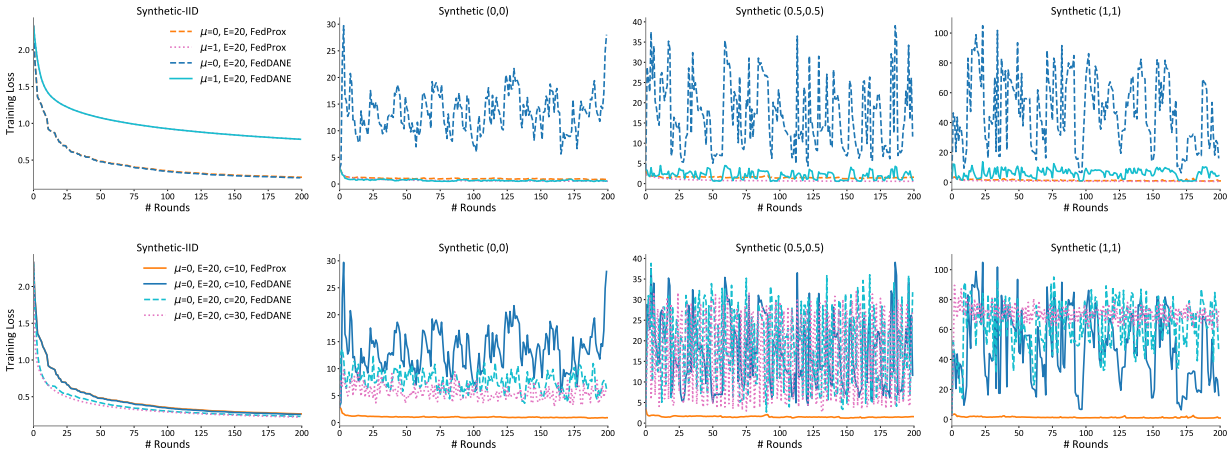

Figure 11: DANE and AIDE [21, 24] are methods proposed in the data center setting that use a similar proximal term as `FedProx` as well as an additional gradient correction term. We modify DANE to apply to federated settings by allowing for local updating and low participation of devices. We show the convergence of this modified method, which we call `FedDane`, on synthetic datasets. In the top figures, we subsample 10 devices out of 30 on all datasets for both `FedProx` and `FedDane`. While `FedDane` performs similarly as `FedProx` on the IID data, it suffers from poor convergence on the non-IID datasets. In the bottom figures, we show the results of `FedDane` when we increase the number of selected devices in order to narrow the gap between our estimated full gradient and the real full gradient (in the gradient correction term). Note that communicating with all (or most of the) devices is already unrealistic in practical settings. We observe that although sampling more devices per round might help to some extent, `FedDane` is still unstable and tends to diverge. This serves as additional motivation for the specific subproblem we propose in `FedProx`.