Federated Optimization for Heterogeneous Networks

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

Abstract

Federated learning involves training and effectively combining machine learning models from distributed partitions of data (i.e., tasks) on edge devices, and be naturally viewed as a multi-task learning problem. 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 the devices/tasks are statistically heterogeneous, i.e., where each device collects data in a non-identical fashion. In this work, we introduce a framework, called `FedProx`, to tackle statistical heterogeneity. `FedProx` encompasses `FedAvg` as a special case. We provide convergence guarantees for `FedProx` through a device dissimilarity assumption. Our empirical evaluation validates our theoretical analysis and demonstrates the improved robustness and stability of `FedProx` 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. *Federated learning* has emerged as an attractive paradigm to push the training of models in such networks to the edge (McMahan et al., 2017). In such settings, the goal is to jointly learn over distributed partitions of data/tasks, where statistical heterogeneity and systems constraints present significant challenges. Optimization methods that allow for local updating and low participation have become the de facto solvers for federated learning (McMahan et al., 2017; Smith et al., 2017). These methods perform a variable number of local updates on a subset of devices to enable flexible and efficient communication. Of current federated optimization methods, `FedAvg` (McMahan et al., 2017) has become state-of-the-art for non-convex federated learning. However, `FedAvg` was not designed to tackle the *statistical heterogeneity* which is 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 (McMahan et al., 2017, Sec 3), and it also lacks theoretical convergence guarantees. Indeed, recent works exploring convergence guarantees are limited to unrealistic scenarios, where (1) the data is either shared across devices or distributed in an IID (identically and independently distributed) manner, or (2) all devices are active at each communication round (Zhou & Cong, 2017; Stich, 2018; Wang & Joshi, 2018; Woodworth et al., 2018; Yu et al., 2018; Wang et al., 2018).

Due to the statistical heterogeneity of the data in federated networks, one can think of federated learning as a prime example of distributed multi-task learning, where each device corresponds to a task. However, the more common goal of federated learning—and the focus of this work—involves training a *single* global model on distributed data collected for these various tasks. We introduce and study a novel optimization framework in the federated setting. Our focus on its convergence behavior in the face of statistically heterogeneous data is closely related to the classical multi-task setting which involves jointly learning task-specific models from statistically heterogeneous data.

Contributions. We propose a federated optimization framework for heterogeneous networks, `FedProx`, which encompasses `FedAvg`. In order to characterize the convergence behavior of `FedProx`, we invoke a *device dissimilarity* assumption in the network. Under this assumption, we provide the first convergence guarantees for `FedProx`. Finally, we demonstrate that our theoretical assumptions reflect empirical performance, and that `FedProx` can improve the robustness and stability of convergence over `FedAvg` when data is heterogeneous across devices.

2. Related Work

Large-scale distributed machine learning has motivated the development of numerous distributed optimization meth-

[*]Equal contribution [1]Carnegie Mellon University [2]Bosch Center for Artificial Intelligence [3]Google AI [4]University of Southern California. Correspondence to: Tian Li <tianli@cmu.edu>.

Proceedings of the 1st Adaptive & Multitask Learning Workshop, Long Beach, California, 2019. Copyright 2019 by the author(s).

ods in the past decade (see, e.g., Dean et al., 2012; Zhang et al., 2013; Li et al., 2014a; Shamir et al., 2014; Reddi et al., 2016; Zhang et al., 2015; Richtárik & Takáč, 2016; Smith et al., 2018). However, it is increasingly attractive to learn statistical models directly over networks of distributed devices. This problem, known as federated learning, requires tackling novel challenges with privacy, heterogeneous data, and massively distributed 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 (Boyd et al., 2010) 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 (McMahan et al., 2017; Smith et al., 2017; Lin et al., 2018). For example, Smith et al. (2017) proposes a communication-efficient primal-dual optimization method that learns separate but related models for each device through a multi-task learning framework. However, such an approach does not generalize to non-convex problems, e.g. deep learning, due to lack of strong duality. In the non-convex setting, Federated Averaging (`FedAvg`), a heuristic method based on averaging local Stochastic Gradient Descent (SGD) updates, has instead been shown to work well empirically (McMahan et al., 2017).

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 heterogeneous. Recent works have made steps towards analyzing `FedAvg` in simpler settings. For instance, parallel SGD and related variants (Zhang et al., 2015; Zhou & Cong, 2017; Stich, 2018; Wang & Joshi, 2018; Woodworth et al., 2018), which make local updates similar to `FedAvg`, have been studied in the *IID* setting. Although some works (Yu et al., 2018; Wang et al., 2018; Hao et al., 2019) have recently explored convergence guarantees in heterogeneous settings, they make the limiting assumptions such as full participation of all devices, convexity (Wang et al., 2018), or uniformly bounded gradients (Yu et al., 2018). There are also several heuristic approaches that aim to tackle statistical heterogeneity, either by sharing the local device data or some server-side proxy data (Jeong et al., 2018; Zhao et al., 2018; Huang et al., 2018), which may be unrealistic in practical federated settings.

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., McMahan et al., 2017; Smith et al., 2017; Lin et al., 2018) are designed to handle multiple devices collecting data and a central server coordinating the global learning objective across the network. The aim is to minimize:

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

where N is the number of devices, p_k $0, 8k$, and $\sum_k p_k$=1. In general, the local objectives measure the local empirical risk over possibly differing data distributions D_k, i.e., $F_k(w) := \mathsf{E}_{x_k \; D_k}[f_k(w; x_k)]$, 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.

To reduce communication and handle systems constraints, federated optimization methods commonly allow for low participation and local updating. At each round, a subset of the devices are selected and use *local solvers* to optimize the local objectives. Then the local updates are aggregated via a central server. Each of the local objectives can be solved *inexactly*, as formally defined below.

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

We use γ-inexactness in our analysis (Section 4) to measure the amount of local computation from each local solver. In experiments (Section 5), we simply run an iterative local solver for some number of local epochs, which can be seen as a proxy for γ-inexactness.

3.1. Federated Averaging (FedAvg)

In Federated Averaging (`FedAvg`) (McMahan et al., 2017), at each round, a subset K N of devices are selected and run SGD locally for E number of epochs to optimize the local objective F_k on device k, and then the resulting model updates are averaged. McMahan et al. (2017) shows empirically that it is crucial to tune the number of local epochs for `FedAvg` to converge, as additional local epochs allow local models to move further away from the initial global model, potentially causing divergence. Thus, 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 serves as our inspiration for `FedProx`, introduced below.

3.2. Proposed Framework: FedProx

Instead of just minimizing the local function F_k, in FedProx, device k uses its local solver to approximately minimize the following surrogate objective h_k:

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

The proximal term in the above expression effectively limits the impact of local updates by restricting them to be close to the current model w^t. We note that proximal terms such as the one above are a popular tool utilized throughout the optimization literature (see Appendix C). 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 settings.

Algorithm 1 FedProx (Proposed Framework)

INPUT: K, T, μ, γ, w^0, N, p_k, $k = 1, \ldots, N$
forall $t = 0, \ldots, T-1$ **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 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 Section 4, we see that the usage of the proximal term makes FedProx more amenable to theoretical analysis. In Section 5, we also see the modified local subproblem in FedProx results in more robust and stable convergence compared to FedAvg for heterogeneous datasets. Note that FedAvg is a special case of FedProx with $\mu = 0$.

4. FedProx: Convergence Analysis

In this section we first introduce a metric that specifically measures the dissimilarity among local functions. We call this metric local dissimilarity. We then analyze FedProx under an assumption on bounded local dissimilarity.

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

Here $\mathbb{E}_k[\,]$ denotes the expectation over devices with masses $p_k = n_k/n$ and $\sum_{k=1}^N p_k = 1$. Note that $B(w) \ge 1$ and the larger the value of $B(w)$, the larger is the dissimilarity among the local functions. Moreover, if $F_k(\,)$'s are associated with empirical risk objectives and the samples on all the devices are homogeneous, then $B(w) \to 1$ for every w as all the local functions converge to the same expected risk function. Interestingly, similar assumptions (e.g., Vaswani et al., 2019; Yin et al., 2018) have been explored elsewhere for differing purposes; see more in Appendix C. Using Definition 2, we now state our formal dissimilarity assumption, which we use in our convergence analysis.

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

Using Assumption 1, we analyze the amount of expected objective decrease if 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. $B(w^t) \le B$. If μ, K, and γ in Algorithm 1 are chosen such that*

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

then at iteration t of Algorithm 1, we have the following expected decrease in the global objective:

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

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

We direct the reader to Appendix A.1 for a detailed proof. Theorem 3 uses the dissimilarity in Definition 2 to identify sufficient decrease at each iteration for FedProx. In Appendix A.2, we provide a corollary characterizing the performance with a more common (though slightly more restrictive) bounded variance assumption.

Remark 4. *In order for ρ in Theorem 3 to be positive, we need $\gamma B < 1$. Moreover, we also need $\sqrt{2}B/\sqrt{K} < 1$. These conditions help to quantify the trade-off between dissimilarity bound (B) and the algorithm parameters (γ, K).*

Finally, we can use the above sufficient decrease to characterize the rate of convergence under Assumption 1. Note that these results hold for general non-convex $F_k(\,)$.

Theorem 5 (Convergence rate: FedProx)**.** *Given some $\epsilon > 0$, assume that for $B \le B_\epsilon$, 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 $\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\left[\|\nabla f(w^t)\|^2\right] \le \epsilon$.*

While the results thus far hold for non-convex $F_k(\ )$, we prove the convergence for convex loss in Appendix A.3. To help provide context for the rate in Theorem 5, we compare it with SGD in the convex case in Appendix A.4, Remark 9.

5. Experiments

We now present empirical results for FedProx. We study the effect of statistical heterogeneity on the convergence of FedAvg and FedProx, explore properties of the FedProx framework, and show how empirical convergence relates to the bounded dissimilarity assumption. We show a subset of our experiments here due to space constraints; for full results we defer the reader to Appendix B. All code, data, and experiments are publicly available at github.com/litian96/FedProx.

Experimental Details. We evaluate FedProx on diverse tasks, models, and both synthetic and real-world datasets. The real datasets are curated from prior work in federated learning (McMahan et al., 2017; Caldas et al., 2018). In particular, We study convex models on partitioned MNIST (LeCun et al., 1998), Federated Extended MNIST (Cohen et al., 2017; Caldas et al., 2018) (FEMNIST), and FMNIST*, and non-convex models on Sentiment140 (Go et al., 2009) (Sent140) and *The Complete Works of William Shakespeare* (McMahan et al., 2017) (Shakespeare). More details are provided in Appendix B.1.

Effect of Statistical Heterogeneity. In Figure 1, we study how statistical heterogeneity affects convergence using four synthetic datasets. From left to right, as data become more heterogeneous, convergence becomes worse for FedProx with μ=0 (FedAvg). Setting $\mu > 0$ is particularly useful in heterogeneous settings although that may slow convergence for IID data.

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. We study FedProx under different values of E and μ using the federated datasets described in Table 1 in Appendix B.1. We report the results on Shakespeare dataset here and provide similar results on all datasets in Appendix B.3.

(1) Dependence on E. We explore the effect of E in Figure 2 (left) and show the convergence in terms of the training loss. We see that large E leads to divergence on Shakespeare. In Appendix B.3, we further show that large E leads to similar instability on other heterogeneous datasets. We note here that a large E may be particularly useful in practice when communication is expensive (which is common in federated networks) where small E is prohibitive. In Figure 3, e.g., 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).

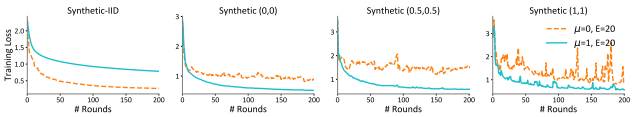

Figure 1. Effect of data heterogeneity on convergence. We show training loss (see testing accuracy and dissimilarity metric in Appendix B.3, Figure 7) on four synthetic datasets whose heterogeneity increases from left to right. The method with $\mu = 0$ corresponds to FedAvg. Increasing heterogeneity leads to worse convergence, but setting $\mu > 0$ can help to combat this.

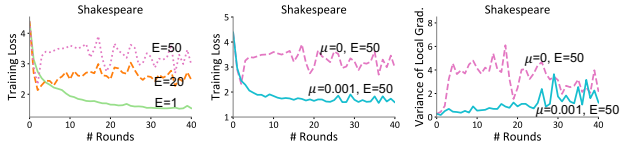

Figure 2. Properties of the FedProx framework. *Left:* Effect of increasing E on the Shakespeare dataset where μ=0. Too many local updates can cause divergence for heterogeneous datasets. *Middle:* Effect of μ. FedProx with μ>0 forces divergent methods to converge. *Right:* The dissimilarity measurement (variance of gradients) on Shakespeare. This metric captures statistical heterogeneity and is consistent with training loss (middle subfigure).

(2) Dependence on μ. We consider the effect of μ on convergence in Figure 2 (middle). We observe that the appropriate μ can force divergent methods to converge or increase the stability for unstable methods (Figure 5, Appendix B.3), thus making the performance of FedProx less dependent on E. In practice, μ can be adaptively chosen based on the current performance of the models. 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 B.5.

Dissimilarity Measurement and Divergence. Finally, in Figure 2 (right), 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 6). We observe that the dissimilarity metric in Definition 2 is consistent with the training loss. Therefore, smaller dissimilarity indicates better convergence, which can be enforced by setting μ appropriately.

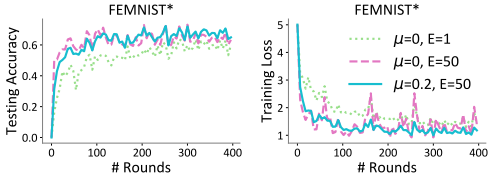

Figure 3. FedProx can provide faster and more stable convergence in communication-constraint environments (those requiring large E) with appropriate μ.

Acknowledgements

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

Abadi, M. et al. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL https://www.tensorflow.org/. Software available from tensorflow.org.

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

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

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

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

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

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

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

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

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

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

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

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

Li, M., Zhang, T., Chen, Y., and Smola, A. J. Efficient mini-batch training for stochastic optimization. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 661–670. ACM, 2014b.

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

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

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

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

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

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

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

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

Stich, S. U. Local sgd converges fast and communicates little. *arXiv preprint arXiv:1805.09767*, 2018.

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

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

Wang, S., Tuor, T., Salonidis, T., Leung, K. K., Makaya, C., He, T., and Chan, K. Adaptive federated learning in resource constrained edge computing systems. *arXiv preprint arXiv:1804.05271*, 2018.

Woodworth, B., Wang, J., McMahan, B., and Srebro, N. Graph oracle models, lower bounds, and gaps for parallel stochastic optimization. *arXiv preprint arXiv:1805.10222*, 2018.

Yin, D., Pananjady, A., Lam, M., Papailiopoulos, D., Ramchandran, K., and Bartlett, P. Gradient diversity: a key ingredient for scalable distributed learning. In *Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics*, pp. 1998–2007, 2018.

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

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

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

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

Zhou, F. and Cong, G. On the convergence properties of a k-step averaging stochastic gradient descent algorithm for nonconvex optimization. *arXiv preprint arXiv:1708.01012*, 2017.

A. Complete Proofs and Convergence Analysis

A.1. 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:

$$\begin{aligned} & r\, F_k(w_k^{t+1}) + \mu(w_k^{t+1} \quad w^t) \quad e_k^{t+1} = 0, \\ & k e_k^{t+1} k \quad \gamma k\, r\, F_k(w^t) k\,. \end{aligned} \tag{3}$$

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

$$w^{t+1} \quad w^t = \frac{1}{\mu}\mathsf{E}_k\left[\, r\, F_k(w_k^{t+1})\right] + \frac{1}{\mu}\mathsf{E}_k\left[e_k^{t+1}\right]. \tag{4}$$

Let us define $\bar{\mu} = \mu \quad L \quad > 0$ 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

$$k\hat{w}_k^{t+1} \quad w_k^{t+1} k \quad \frac{\gamma}{\mu} k\, r\, F_k(w^t) k. \tag{5}$$

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

$$k w_k^{t+1} \quad w^t k \quad \frac{1+\gamma}{\mu} k\, r\, F_k(w^t) k. \tag{6}$$

Therefore,

$$\begin{aligned} & k w^{t+1} \quad w^t k \quad \mathsf{E}_k\left[k w_k^{t+1} \quad w^t k\right] \quad \frac{1+\gamma}{\mu}\mathsf{E}_k\left[k\, r\, F_k(w^t) k\right] \\ & \frac{1+\gamma}{\mu}\sqrt{\mathsf{E}_k[k\, r\, F_k(w^t) k^2]} \quad \frac{B(1+\gamma)}{\mu} k\, r\, f(w^t) k, \end{aligned} \tag{7}$$

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

Now let us define M_{t+1} such that $w^{t+1} \quad w^t = \frac{1}{\mu}\left(\, r\, f(w^t) + M_{t+1}\right)$, i.e. $M_{t+1} = \mathsf{E}_k\left[\, r\, F_k(w_k^{t+1}) \quad r\, F_k(w^t) \quad e_k^{t+1}\right]$. We can bound $k M_{t+1} k$:

$$k M_{t+1} k \quad \mathsf{E}_k\left[L k w_k^{t+1} \quad w_k^t k + k e_k^{t+1} k\right] \quad \left(\frac{L(1+\gamma)}{\mu} + \gamma\right) \quad \mathsf{E}_k\left[k\, r\, F_k(w^t) k\right] \quad \left(\frac{L(1+\gamma)}{\mu} + \gamma\right) B k\, r\, f(w^t) k, \tag{8}$$

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

$$\begin{aligned} f(w^{t+1}) \quad & f(w^t) + h\, r\, f(w^t), w^{t+1} \quad w^t i + \frac{L}{2} k w^{t+1} \quad w^t k^2 \\ f(w^t) \quad & \frac{1}{\mu} k\, r\, f(w^t) k^2 \quad \frac{1}{\mu} h\, r\, f(w^t), M_{t+1} i + \frac{L(1+\gamma)^2 B^2}{2\mu^2} k\, r\, f(w^t) k^2 \\ f(w^t) \quad & \left(\frac{1 \quad \gamma B}{\mu} \quad \frac{LB(1+\gamma)}{\mu\mu} \quad \frac{L(1+\gamma)^2 B^2}{2\mu^2}\right) \quad k\, r\, f(w^t) k^2. \end{aligned} \tag{9}$$

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 $f(\bar{w}^{t+1}) \quad f(w^t)$ which is proportional to $k\, r\, f(w^t) k^2$. 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 $\mathsf{E}\left[f(w^{t+1})\right]$, we use local Lipschitz continuity of the function f.

$$f(w^{t+1}) \quad f(w^{t+1}) + L_0 k w^{t+1} \quad w^{t+1} k, \tag{10}$$

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

$$\begin{aligned} L_0 \quad & k\, r\, f(w^t) k + L \max(k w^{t+1} \quad w^t k, k w^{t+1} \quad w^t k) \\ & k\, r\, f(w^t) k + L(k w^{t+1} \quad w^t k + k w^{t+1} \quad w^t k). \end{aligned} \tag{11}$$

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

$$\mathbb{E}_{S_t}\left[f(w^{t+1})\right] \quad f(w^{t+1}) + Q_t, \tag{12}$$

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

$$\begin{aligned} Q_t \quad & \mathbb{E}_{S_t}\left[\left(k \, r \, f(w^t) k + L(k w^{t+1} \quad w^t k + k w^{t+1} \quad w^t k)\right) \quad k w^{t+1} \quad w^{t+1} k\right] \\ & \left(k \, r \, f(w^t) k + L k w^{t+1} \quad w^t k\right) \mathbb{E}_{S_t}\left[k w^{t+1} \quad w^{t+1} k\right] + L \mathbb{E}_{S_t}\left[k w^{t+1} \quad w^t k \;\; k w^{t+1} \quad w^{t+1} k\right] \\ & \left(k \, r \, f(w^t) k + 2L k w^{t+1} \quad w^t k\right) \mathbb{E}_{S_t}\left[k w^{t+1} \quad w^{t+1} k\right] + L \mathbb{E}_{S_t}\left[k w^{t+1} \quad w^{t+1} k^2\right] \end{aligned} \tag{13}$$

From (7), we have that $k\bar{w}^{t+1} \quad w^t k \quad \frac{B(1+\gamma)}{\mu} k \, r \, f(w^t) k$. Moreover,

$$\mathbb{E}_{S_t}\left[k w^{t+1} \quad w^{t+1} k\right] \quad \sqrt{\mathbb{E}_{S_t}[k w^{t+1} \quad w^{t+1} k^2]} \tag{14}$$

and

$$\begin{aligned} & \mathbb{E}_{S_t}\left[k w^{t+1} \quad w^{t+1} k^2\right] \quad \frac{1}{K} \mathbb{E}_k\left[k w_k^{t+1} \quad w^{t+1} k^2\right] \\ & \quad \frac{2}{K} \mathbb{E}_k\left[k w_k^{t+1} \quad w^t k^2\right], \quad (\text{as } w^{t+1} = \mathbb{E}_k\left[w_k^{t+1}\right]) \\ & \quad \frac{2}{K} \frac{(1+\gamma)^2}{\mu^2} \mathbb{E}_k\left[k \, r \, F_k(w^t) k^2\right] \quad (\text{from (6)}) \\ & \quad \frac{2B^2}{K} \frac{(1+\gamma)^2}{\mu^2} k \, r \, f(w^t) k^2, \end{aligned} \tag{15}$$

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 (13) we get

$$Q_t \quad \left(\frac{B(1+\gamma)}{\mu} \frac{p}{K} \overline{2} + \frac{LB^2(1+\gamma)^2}{\mu^2 K}\left(2 \, p \overline{2K} + 2\right)\right) k \, r \, f(w^t) k^2 \tag{16}$$

Combining (9), (12), (10) and (16) and using the notation $\alpha = \frac{1}{\mu}$ we get

$$\begin{aligned} \mathbb{E}_{S_t}\left[f(w^{t+1})\right] \quad & f(w^t) \quad \left(\frac{1}{\mu} \quad \frac{\gamma B}{\mu} \quad \frac{B(1+\gamma)}{\mu} \frac{p}{K} \overline{2} \quad \frac{LB(1+\gamma)}{\mu\mu}\right. \\ & \left. \frac{L(1+\gamma)^2 B^2}{2\mu^2} \quad \frac{LB^2(1+\gamma)^2}{\mu^2 K}\left(2 \, p \overline{2K} + 2\right)\right) k \, r \, f(w^t) k^2. \end{aligned}$$

□

A.2. 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 6 (Bounded Variance Equivalence)**.** *Let Assumption 1 hold. Then, in the case of bounded variance, i.e.,* $\mathbb{E}_k\left[k \, r \, F_k(w) \quad r \, f(w) k^2\right] \quad \sigma^2$*, for any* $\epsilon > 0$ *it follows that* $B_\epsilon \quad \sqrt{1 + \frac{\sigma^2}{\epsilon}}$.

Proof. We have,

$$\begin{aligned} & E_k[k \, r \, F_k(w) \quad r \, f(w) k^2] = E_k[k \, r \, F_k(w) k^2] \quad k \, r \, f(w) k^2 \quad \sigma^2 \\ & \;) \; E_k[k \, r \, F_k(w) k^2] \quad \sigma^2 + k \, r \, f(w) k^2 \\ & \;) \; B_\epsilon = \sqrt{\frac{E_k[k \, r \, F_k(w) k^2]}{k \, r \, f(w) k^2}} \quad \sqrt{1 + \frac{\sigma^2}{\epsilon}}. \end{aligned}$$

With Corollary 6 in place, we can restate the main result in Theorem 3 in terms of the bounded variance assumption.

Theorem 7 (Non-Convex FedProx Convergence: Bounded Variance)**.** *Let the assertions of Theorem 3 hold. In addition, let the iterate w^t be such that $\|\nabla f(w^t)\|^2 \leq \epsilon$, 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 1 are chosen such that*

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

then at iteration t of Algorithm 1, 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 7 follows from the proof of Theorem 3 by noting the relationship between the bounded variance assumption and the dissimilarity assumption as portrayed by Corollary 6.

A.3. Convergence: Convex Case

Corollary 8 (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 $1 \leq B \leq 0.5\sqrt{K}$, then we can choose $\mu \geq 6LB^2$ from which it follows that $\rho \geq \frac{1}{24LB^2}$.*

Proof. In the convex case, where $L \geq 0$ 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

$$\mathbb{E}_{S_t}\left[f(w^{t+1})\right] \lessapprox f(w^t) - \frac{1}{2\mu}\|\nabla f(w^t)\|^2 + \frac{3LB^2}{2\mu^2}\|\nabla f(w^t)\|^2 . \tag{17}$$

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

$$\mathbb{E}_{S_t}\left[f(w^{t+1})\right] \lessapprox f(w^t) - \frac{1}{24LB^2}\|\nabla f(w^t)\|^2 . \tag{18}$$

Note that the expectation in (18) 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 $O(\frac{LB^2}{\epsilon})$.

A.4. Comparison with SGD

Remark 9 (Comparison with SGD)**.** *Note that FedProx achieves the same asymptotic convergence guarantee as SGD. In other words, under the bounded variance assumption, for small ϵ, if we replace B_ϵ with its upper-bound in Corollary 6 and choose μ large enough, then the iteration complexity of FedProx when the subproblems are solved exactly and $F_k(\cdot)$'s are convex would be $O(\frac{L}{\epsilon} + \frac{L\sigma^2}{\epsilon^2})$, which is the same as SGD (Ghadimi & Lan, 2013).*

B. Experimental Details

Synthetic data. To generate synthetic data, we follow a similar setup to that described in (Shamir et al., 2014), additionally imposing heterogeneity among devices. Full details are given in Appendix B.1. In particular, for each device k, we generate synthetic samples (X_k, Y_k) according to the model $y = \text{argmax}(\text{softmax}(Wx + b))$, $x \in \mathbb{R}^{60}, W \in \mathbb{R}^{10 \times 60}, b \in \mathbb{R}^{10}$. 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, 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.

Real data. We also explore five real datasets, their statistics summarized in Table 1 in Appendix B.1. These datasets are curated from prior work in federated learning as well as recent federated learning-related benchmarks (McMahan et al., 2017; Caldas et al., 2018). We study two convex models on partitioned MNIST (LeCun et al., 1998), Federated Extended MNIST (Cohen et al., 2017; Caldas et al., 2018) (FEMNIST), and FMNIST*. We study two non-convex models on Sentiment140 (Go et al., 2009) (Sent140) and *The Complete Works of William Shakespeare* (McMahan et al., 2017) (Shakespeare). Details of datasets, models, and workloads are provided in Appendix B.1.

Implementation. We implement `FedAvg` and `FedProx` in Tensorflow (Abadi et al., 2015). See details in Appendix B.2.

Setup. For each experiment, we tune the learning rate and ratio of active devices per round on `FedAvg`. We randomly split the data on each local device into 80% training set and 20% testing set. For each comparison, the devices selected and data read at each round are the same across all runs. We report all metrics based on the global objective $f(w)$. Note that `FedAvg` ($\mu = 0$) and `FedProx` (μ 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.

B.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 (McMahan et al., 2017), and some proposed in LEAF, a benchmark for federated settings (Caldas et al., 2018). We also create synthetic data to directly test the effect of heterogeneity on convergence, as in Section 5.

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 $N(0, 1)$ on all devices and X_k to follow the same distribution $N(v, \Sigma)$ where each element in the mean vector v is drawn from $N(0, 1)$ 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 (LeCun et al., 1998) 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 (Cohen et al., 2017) 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* (McMahan et al., 2017). Each speaking role in a play represents a different device. We use a two layer LSTM classifier containing 100 hidden units with a 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 character 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 (Go et al., 2009) (Sent140) with a two layer LSTM binary classifier containing 256 hidden units with pretrained 300D GloVe embedding (Pennington et al., 2014). 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.

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

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.

B.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.

B.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 `FedAvg` and 1: sampling devices uniformly and averaging updates with weights proportional to the number of local data points (as originally proposed in (McMahan et al., 2017)). 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. Interestingly, we also observe that the sampling scheme proposed herein results in more stable performance for both methods (see Appendix B.4, Figure 10). 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.

(**Hyperparameters**) For each dataset, we tune the ratio of active clients per round from $f0.01, 0.05, 0.1g$ on `FedAvg`. For synthetic datasets, roughly 10% of the devices are active at each round. For MNIST, FEMNIST, Shakespeare, Sent140 and FEMNIST*, the number 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 (Abadi et al., 2015) Version 1.10.1. Please see github.com/litian96/FedProx for full details.

B.3. Full Experiments

We explore the effect of E in Figure 4. 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 data distributed across devices after partitioning FEMNIST and Sent140 lack significant heterogeneity. We validate this hypothesis by observing instability on FEMNIST*, which is a skewed variant of the FEMNIST dataset.

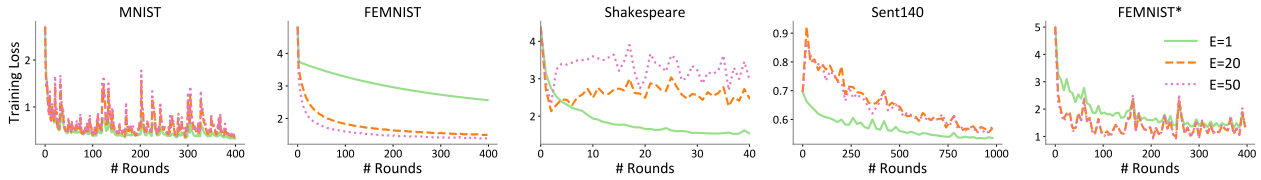

Figure 4. Effect of increasing E on real federated datasets where $\mu = 0$ (corresponds to `FedAvg`). Too many local updates can cause divergence or instability for heterogeneous datasets. Note that FEMNIST* is a more skewed version of FEMNIST.

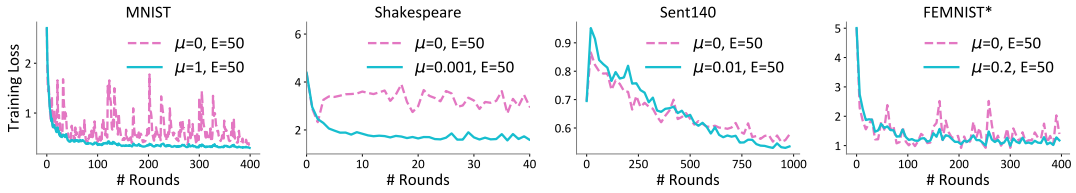

Figure 5. Effect of μ on real datasets. The setting $\mu = 0$ corresponds to `FedAvg`. `FedProx` with $\mu > 0$ leads to more stable convergence and enables otherwise divergent methods to converge.

We consider the effect of μ on convergence in Figure 5. For each experiment, in the case of $E = 50$, we compare the results between $\mu = 0$ and the best μ. For three out of the four datasets (all but Sent140) we observe that the appropriate μ can increase the stability for unstable methods and can force divergent methods to converge.

Finally, in Figure 6, 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 6). We 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 B.3.

We present testing accuracy, training loss and dissimilarity measurements of all the experiments in Figure 7, Figure 8 and Figure 9.

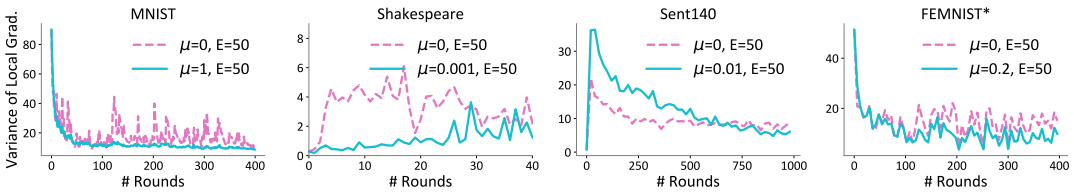

Figure 6. The dissimilarity measurement (variance of gradients) on four federated datasets. This metric captures statistical heterogeneity and is consistent with training loss (Figure 5). Smaller dissimilarity indicates better convergence.

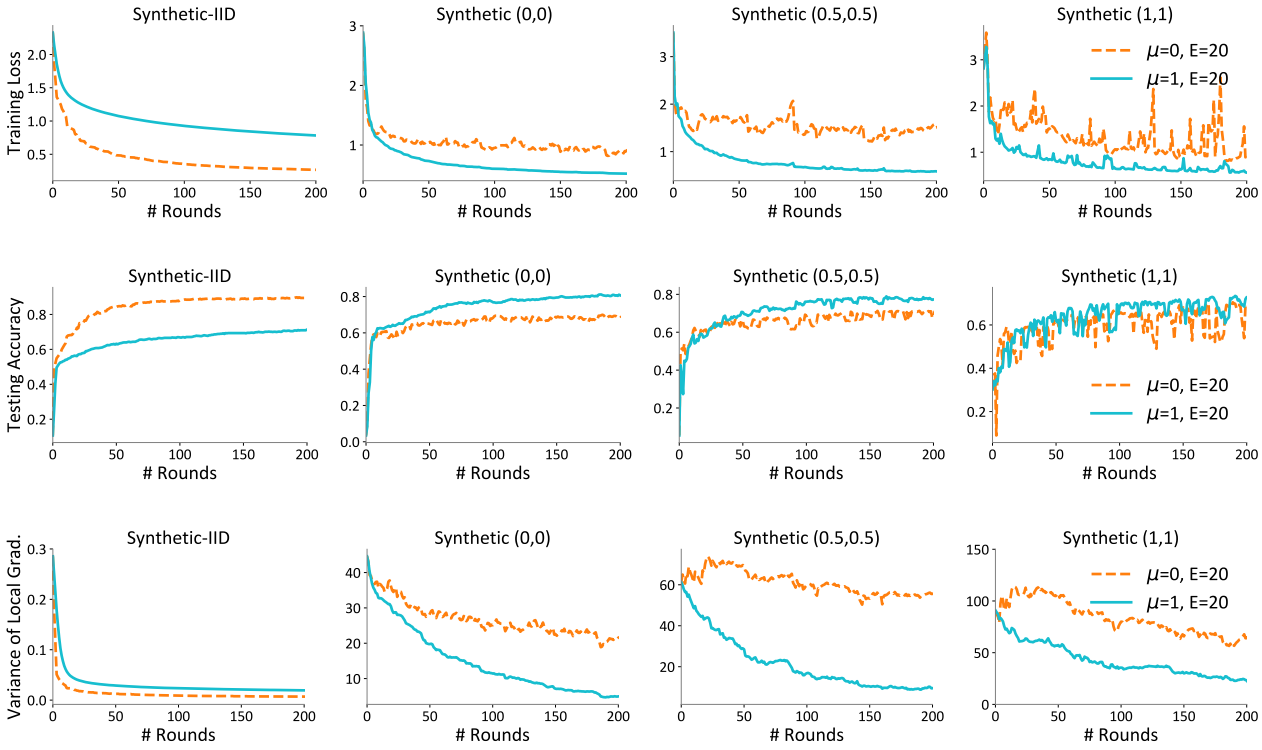

Figure 7. Training loss, testing accuracy and dissimilarity measurement for experiments in Figure 1

B.4. `FedProx` with two device sampling schemes

We show the training loss, testing accuracy and dissimilarity measurement of `FedProx` using two different device sampling schemes in Figure 10.

B.5. Adaptively setting μ

We show a simple adaptive heuristic of setting μ on four synthetic datasets in Figure 11.

C. Connections to other federated and distributed methods

Two aspects of the proposed work: our framework, `FedProx`, and analysis tool, the bounded dissimilarity assumption, have been utilized throughout the optimization literature—though often with very different motivations. For completeness, we provide a discussion below on our relation to these prior works.

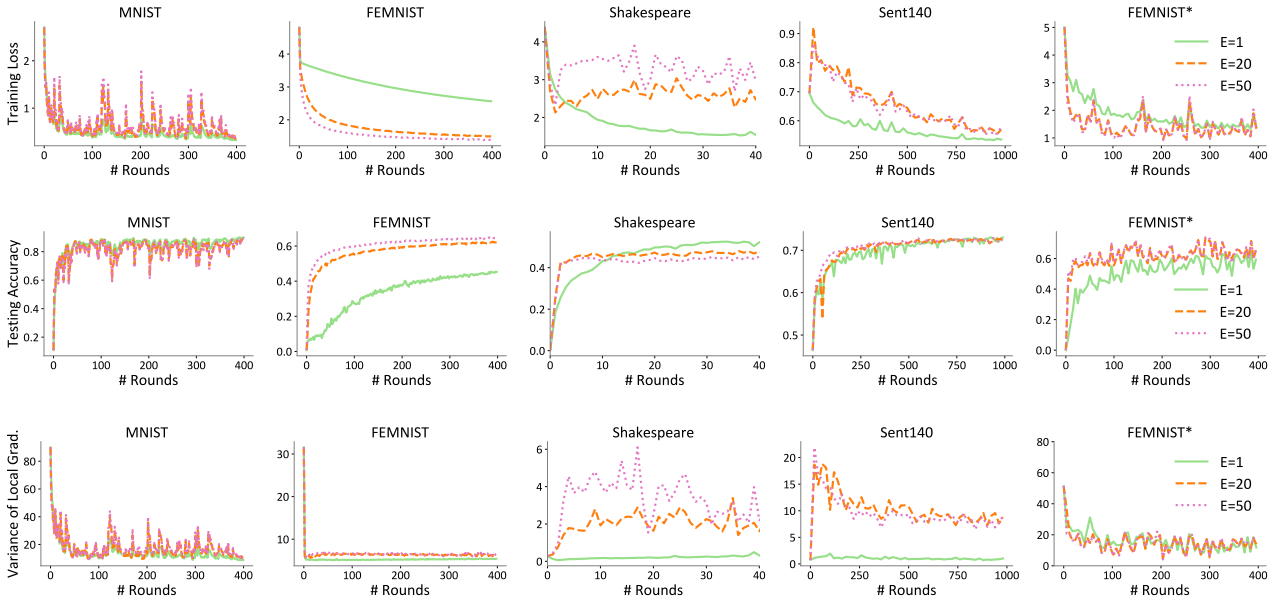

Figure 8. Training loss, testing accuracy and dissimilarity measurement for experiments in Figure 4

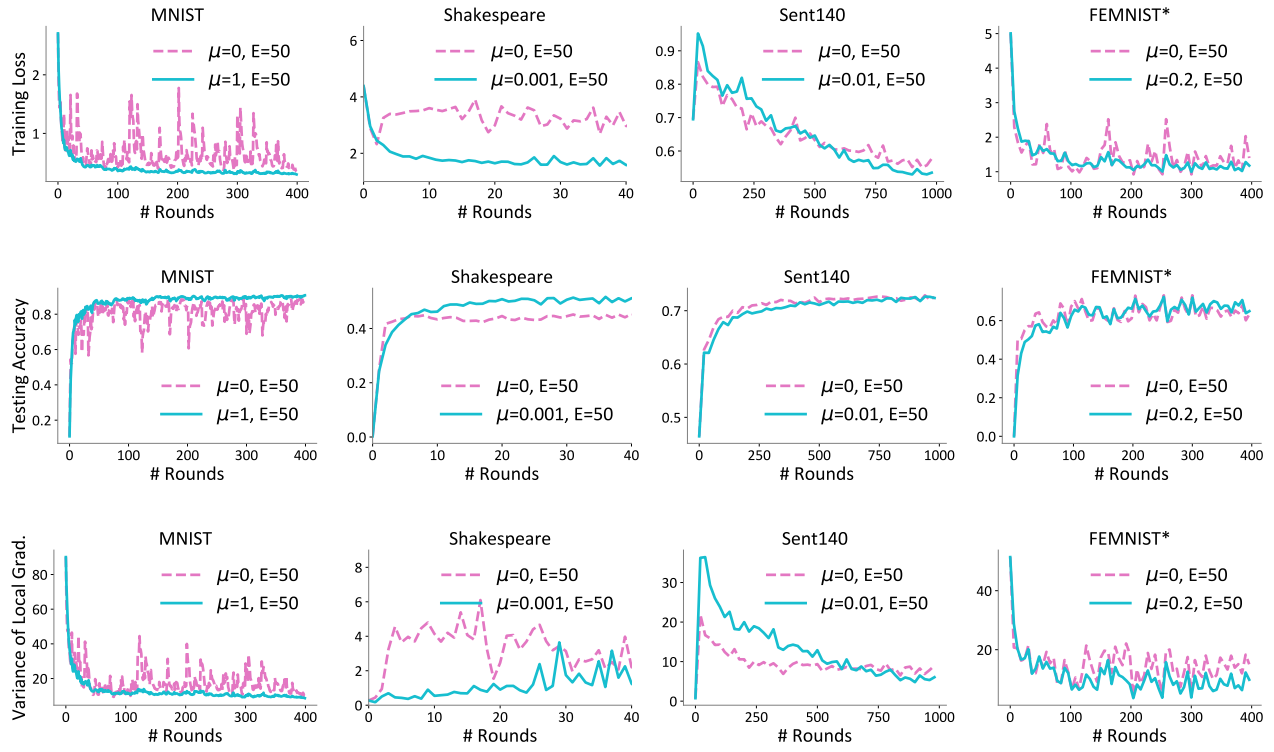

Figure 9. Training loss, testing accuracy and dissimilarity measurement for experiments in Figure 5

Proximal term. We note here a connection to elastic averaging SGD (EASGD) (Zhang et al., 2015), 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

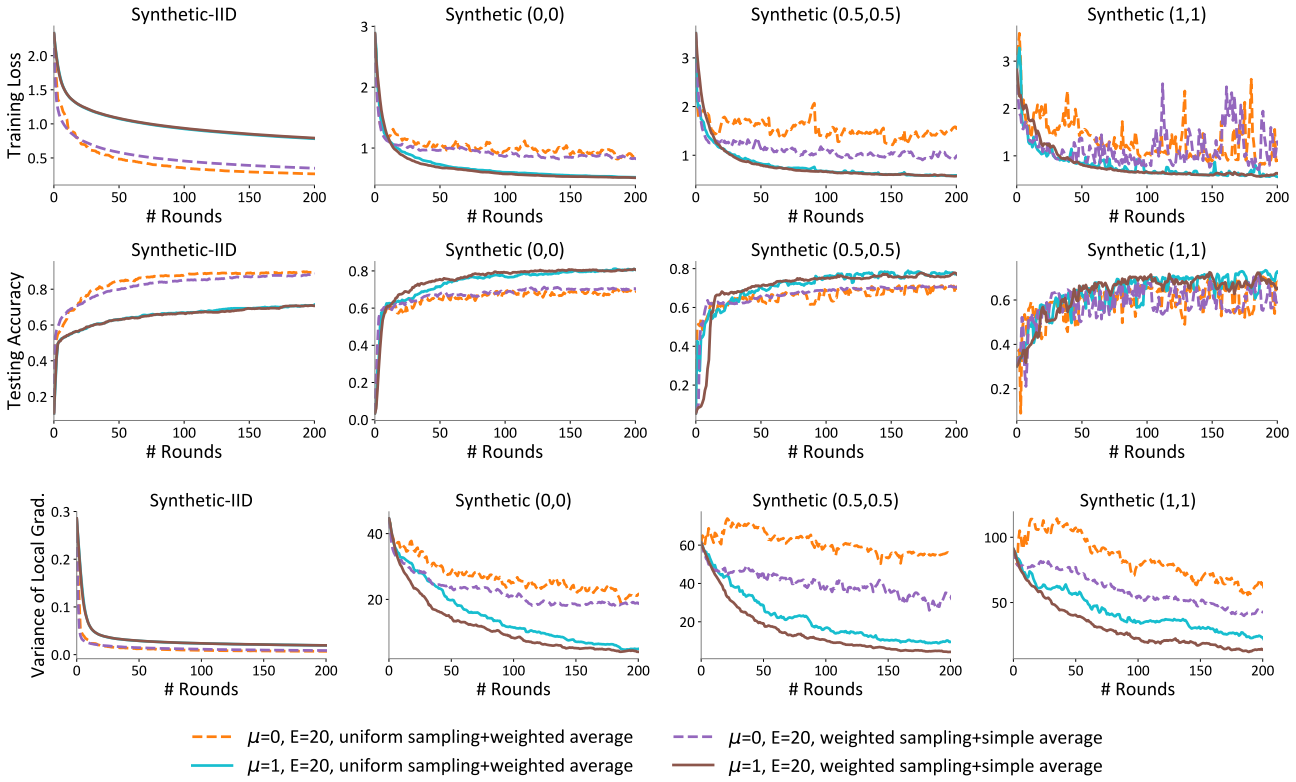

Figure 10. Differences between two sampling schemes in terms of training loss, testing accuracy and dissimilarity measurement. Sampling devices with a probability proportional to the number of local data points and then simply averaging local models performs slightly better than uniformly sampling devices 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 $\mu = 0$.

for simple quadratic problems. The proximal term we introduce has also been explored in previous optimization literature with very different purposes, such as (Allen-Zhu, 2018), to speed up (mini-batch) SGD training on a single machine. Li et al. (2014b) also employs a similar proximal term for efficient SGD training both in a single machine and distributed settings, but their analysis is limited to a single machine setting with different assumptions (e.g., IID data and solving the subproblem exactly at each round). DANE (Shamir et al., 2014) also includes a proximal term in the local objective function. However, due to the inexact estimation of full gradients (i.e., $r\,\phi(w^{(t\ \ 1)})$ in (Shamir et al., 2014, Eq (13))) with device subsampling schemes and the staleness of the gradient correction term (Shamir et al., 2014, Eq (13)) in local updating methods, it is not directly applicable to our setting and performs worse on heterogeneous datasets (see Figure 12).

Bounded dissimilarity assumption. The bounded dissimilarity assumption has appeared in different forms, for example in (Yin et al., 2018; Vaswani et al., 2019). In (Yin et al., 2018), the bounded similarity assumption is used in context of asserting gradient diversity and quantifying the benefit in terms of scaling of the mean square error for mini-batch SGD for data which is i.i.d. In (Vaswani et al., 2019), 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 better convergence rates for SGD with constant step-size. Note that this is different with 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.

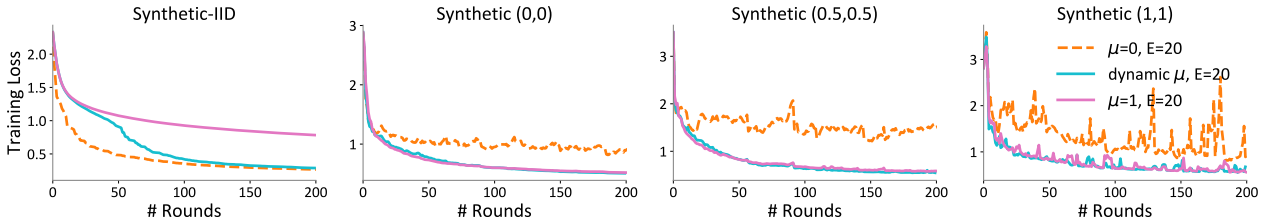

Figure 11. Results of choosing μ adaptively. We increase μ by 0.1 whenever the loss increases and decreases it by 0.1 whenever the loss decreases for 5 consecutive rounds. We initialize μ to 1 for the IID data (Synthetic-IID) (in order to be adversarial to our methods), and initialize it to 0 for the other three non-IID datasets. We observe that this simple heuristic works well in practice.

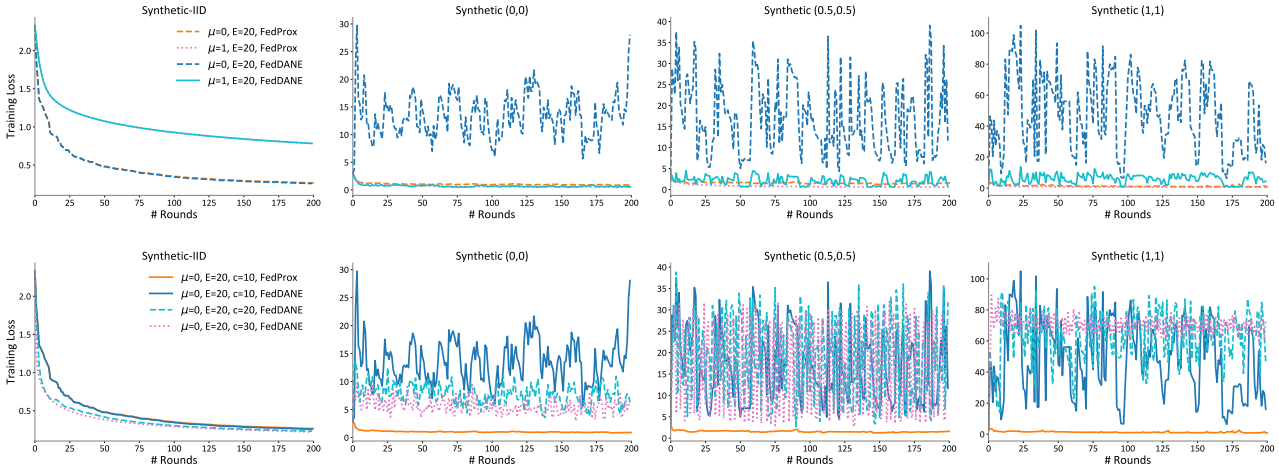

Figure 12. We twist DANE (Shamir et al., 2014) to federated settings by allowing for local updating and low participation of devices, which we call `FedDane`. We show the convergence of `FedDane` on synthetic datasets. In the top figures, we subsample 10 devices out of 30 on all datasets for both `FedProx` and `FedDane`. While `FedDane` performs similarly as `FedProx` on the IID data, it suffers from poor convergence on other 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 divergent.