FEDERATED LEARNING BASED ON DYNAMIC REGULARIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

We propose a novel federated learning method for distributively training neural network models, where the server orchestrates cooperation between a subset of randomly chosen devices in each round. We view Federated Learning problem primarily from a communication perspective and allow more device level computations to save transmission costs. We point out a fundamental dilemma, in that the minima of the local-device level empirical loss are inconsistent with those of the global empirical loss. Different from recent prior works, that either attempt inexact minimization or utilize devices for parallelizing gradient computation, we propose a dynamic regularizer for each device at each round, so that in the limit the global and device solutions are aligned. We demonstrate both through empirical results on real and synthetic data as well as analytical results that our scheme leads to efficient training, in both convex and non-convex settings, while being fully agnostic to device heterogeneity and robust to large number of devices, partial participation and unbalanced data.

1 INTRODUCTION

In (McMahan et al., 2017), the authors proposed federated learning (FL), a concept that leverages data spread across many devices, to learn classification tasks distributively without recourse to data sharing. The authors identified four principle characteristics of FL based on several use cases. First, the *communication links* between the server and devices are unreliable, and at any time, there may only be a small subset of devices that are active. Second, data is *massively distributed*, namely the number of devices are large, while amount of data per device is small. Third, device data is *heterogeneous*, in that data in different devices are sampled from different parts of the sample space. Finally, data is *unbalanced*, in that the amount of data per device is highly variable.

The basic FL problem can be cast as one of empirical minimization of a global loss objective, which is decomposable as a sum of device-level empirical loss objectives. The number of communication rounds, along with the amount of bits communicated per round, has emerged as a fundamental *gold standard* for FL problems. Many mobile and IoT devices are bandwidth constrained, and wireless transmission and reception is significantly more power hungry than computation (Halgamuge et al., 2009). As such schemes that reduce communication are warranted. While distributed SGD is a viable method in this context, it is nevertheless communication inefficient.

A Fundamental Dilemma. Motivated by these ideas, recent work has proposed to push optimization burden onto the devices, in order to minimize amount of communications. Much of the work in this context, propose to optimize the local risk objective based on running SGD over mini-batched device data, analogous to what one would do in a centralized scenario. On the one hand, training models on local data that minimize local empirical loss appears to be meaningful, but yet, doing so, is fundamentally *inconsistent* with minimizing the global empirical loss[1]. Prior works (McMahan et al., 2017; Karimireddy et al., 2019; Reddi et al., 2020) attempt to overcome this issue by running fewer epochs or rounds of SGD on the devices, or attempt to stabilize server-side updates so that

[1]To see this consider the situation where losses are differentiable. As such stationary points for global empirical loss demand that only the sum of the gradients of device empirical losses are zero, and not necessarily that the individual device gradients are zero. Indeed, in statistically heterogeneous situations, such as where we have heterogeneous dominance of classes, stationary points of local empirical functions do not coincide.

the resulting fused models correspond to inexact minimizations and can result in globally desirable properties.

Dynamic Regularization. To overcome these issues, we revisit the FL problem, and view it primarily from a communication perspective, with the goal of minimizing communication, and as such allowing for significantly more processing and optimization at the device level. This approach, while increasing computation for devices, leads to substantial improvement in communication efficiency over existing state-of-the-art methods, uniformly across the four FL scenarios (unreliable links, massive distribution, substantial heterogeneity, and unbalanced data). Specifically, in each round, we dynamically modify the device objective with a penalty term so that, in the limit, when model parameters converge, they do so to stationary points of the global empirical loss. Concretely, we add linear and quadratic penalty terms, whose minima is consistent with the global stationary point. We then provide an analysis of our proposed FL algorithm and demonstrate convergence of the local device models to models that satisfy conditions for local minima of global empirical loss with a rate of $O\left(\frac{1}{T}\right)$ where T is number of rounds communicated. For convex smooth functions, with m devices, and P devices active per round, our convergence rate for average loss with balanced data scales as $O\left(\frac{1}{T}\sqrt{\frac{m}{P}}\right)$, substantially improving over the state-of-art (SCAFFOLD $O\left(\frac{1}{T}\frac{m}{P}\right)$). For non-convex smooth functions, we establish a rate of $O\left(\frac{1}{T}\frac{m}{P}\right)$.

We perform experiments on both visual and language real-world datasets including MNIST, EMNIST, CIFAR-10, CIFAR-100 and Shakespeare. We tabulate performance studying cases that are reflective of FL scenarios, namely, for (i) varying device participation levels, (ii) massively distributed data, (iii) various levels of heterogeneity, as well as (iv) unbalanced local data settings. Our proposed algorithm, FedDyn, has similar overhead to competing approaches, but converges at a significantly faster rate. This results in a substantial reduction in communication compared to baseline approaches such as conventional FedAvg (McMahan et al., 2017), FedProx (Li et al., 2020) and SCAFFOLD (Karimireddy et al., 2019), for achieving target accuracy. Furthermore, our approach is simple to implement, requiring far less hyperparameter tuning compared to competing methods.

Contributions. We summarize our main results here.

- We present, FedDyn, a novel dynamic regularization method for FL. Key to FedDyn is a new concept, where in each round the risk objective for each device is dynamically updated so as to ensure that the device optima is asymptotically consistent with stationary points of the global empirical loss,
- We prove convergence results for FedDyn in both convex and non-convex settings, and obtain sharp results for communication rounds required for achieving target accuracy. Our results for convex case improves significantly over state-of-art prior works. FedDyn in theory is unaffected by heterogeneity, massively distributed data, and quality of communication links,
- On benchmark examples FedDyn achieves significant communication savings over competing methods uniformly across various choices of device heterogeneity and device participation on massively distributed large-scale text and visual datasets.

Related Work. FL is a fast evolving topic, and we only describe closely related approaches here. Comprehensive field studies have appeared in (Kairouz et al., 2019; Li et al., 2020). The general FL setup involves two types of updates, the server and device, and each of these updates are associated with minimizing some local loss function, which by itself could be updated dynamically over different rounds. At any round, there are methods that attempt to fully optimize or others that propose inexact optimization. We specifically focus on relevant works that address the four FL scenarios (massive distribution, heterogeneity, unreliable links, and unbalanced data) here.

One line of work proposes local SGD (Stich, 2019) based updates, wherein each participating device performs a single local SGD step. The server then averages received models. In contrast to local SGD, our method proposes to minimize a local penalized empirical loss.

FedAvg (McMahan et al., 2017) is a generalization of local SGD, which proposes a larger number of local SGD steps per round. Still, FedAvg inexactly solves device side optimization. Identifying when to stop minimizing so that one gets a good accuracy-communication trade-off is based on tuning the number of epochs and the learning rate (McMahan et al., 2017; Li et al., 2020b). Despite the strong empirical performance of FedAvg in IID settings, performance degrades in non-IID scenarios (Zhao et al., 2018).

Several modifications of FedAvg have been proposed to handle non-IID settings. These variants include using a decreasing learning rate (Li et al., 2020b); modifying device empirical loss dynamically (Li et al., 2020a); or modifying server side updates (Hsu et al., 2019; Reddi et al., 2020). Methods that use a decreasing learning rate or customized server side updates still rely on local SGD updates within devices. While these works do recognize the incompatibility of local and global stationary points, their proposed fix is based on inexact minimization. Additionally, in order to establish convergence for non-IID situations, these works impose additional "bounded-non-IID" conditions.

FedProx (Li et al., 2020a) is related to our method. Like us they propose a dynamic regularizer, which is modified based on server supplied models. This regularizer penalizes updates that are far away from the server model. Nevertheless, the resulting regularizer does not result in aligning the global and local stationary points, and as such inexact minimization is warranted, and they do so by carefully choosing learning rates and epochs. Furthermore, tuning requires some knowledge of statistical heterogeneity.

In a similar vein, there are works that augment updates with extra device variables that are also transmitted along with the models (Karimireddy et al., 2019; Shamir et al., 2014). These works prove convergence guarantees through adding device-dependent regularizers. Nevertheless, they suffer additional communication costs and they are not extensively experimented with deep neural networks. Among them, SCAFFOLD (Karimireddy et al., 2019) is a closely related work even though it transmits extra variables and a more detailed comparison is given in Section 2.

Another line of distributed optimization methods (Konečnỳ et al., 2016; Makhdoumi & Ozdaglar, 2017; Shamir et al., 2014; Yuan & Ma, 2020) could be considered in this setting. However, these algorithms are proposed for full device participation case which fails to satisfy one important aspect of FL. FedSVRG (Konečnỳ et al., 2016) and DANE (Shamir et al., 2014) need gradient information from all devices at each round and they are not directly applicable to partial FL settings. For example, FedDANE (Li et al., 2019) is a version of DANE that works in partial participation. However, FedDANE performs worse than FedAvg empirically with partial participation (Li et al., 2019).

2 METHOD

We assume there is a cloud server which can transmit and receive messages from m client devices. Each device, $k \in [m]$ consists of N_k training instances in the form of features, $\boldsymbol{x} \in \mathcal{X}$ and corresponding labels $y \in \mathcal{Y}$ that are drawn IID from a device-indexed joint distribution, $(\boldsymbol{x}, y) \sim P_k$. Our objective is to solve

$$\arg\min_{\boldsymbol{\theta} \in \mathbb{R}^d} \left[\ell(\boldsymbol{\theta}) \triangleq \frac{1}{m} \sum_{k \in [m]} L_k(\boldsymbol{\theta}) \right]$$

where, $L_k(\boldsymbol{\theta}) = \mathbb{E}_{(\boldsymbol{x},y)\sim\mathcal{D}_k}[\ell_k(\boldsymbol{\theta}; (\boldsymbol{x}, y))]$ is the empirical loss of the kth device, and $\boldsymbol{\theta}$ are the parameters of our neural network, whose structure is assumed to be identical across the devices and the server. We denote by $\boldsymbol{\theta}^*$ a local minima of the global empirical loss function.

Algorithm 1: Federated Dynamic Regularizer - (FedDyn)

Input: $T, \boldsymbol{\theta}^0, \alpha > 0, \nabla L_k(\boldsymbol{\theta}_k^0) = \mathbf{0}$.
for $t = 1, 2, \ldots T$ **do**
  Sample devices $\mathcal{P}_t \subseteq [m]$ and transmit $\boldsymbol{\theta}^{t-1}$ to each selected device,
  for *each device $k \in \mathcal{P}_t$, and in parallel* **do**
    Set $\boldsymbol{\theta}_k^t = \underset{\boldsymbol{\theta}}{\operatorname{argmin}}\, L_k(\boldsymbol{\theta}) - \langle \nabla L_k(\boldsymbol{\theta}_k^{t-1}), \boldsymbol{\theta} \rangle + \frac{\alpha}{2}\|\boldsymbol{\theta} - \boldsymbol{\theta}^{t-1}\|^2$,
    Set $\nabla L_k(\boldsymbol{\theta}_k^t) = \nabla L_k(\boldsymbol{\theta}_k^{t-1}) - \alpha\left(\boldsymbol{\theta}_k^t - \boldsymbol{\theta}^{t-1}\right)$,
    Transmit device model $\boldsymbol{\theta}_k^t$ to server,
  end for
  for *each device $k \notin \mathcal{P}_t$, and in parallel* **do**
    Set $\boldsymbol{\theta}_k^t = \boldsymbol{\theta}_k^{t-1}$, $\nabla L_k(\boldsymbol{\theta}_k^t) = \nabla L_k(\boldsymbol{\theta}_k^{t-1})$,
  end for
  Set $\boldsymbol{h}^t = \boldsymbol{h}^{t-1} - \alpha \frac{1}{m}\left(\sum_{k \in \mathcal{P}_t} \boldsymbol{\theta}_k^t - \boldsymbol{\theta}^{t-1}\right)$,
  Set $\boldsymbol{\theta}^t = \left(\frac{1}{|\mathcal{P}_t|}\sum_{k \in \mathcal{P}_t} \boldsymbol{\theta}_k^t\right) - \frac{1}{\alpha}\boldsymbol{h}^t$
end for

FedDyn Method. Our proposed method, FedDyn, is displayed in Algorithm 1. In each round, $t \in [T]$, a subset of devices $\mathcal{P}_t \subset [m]$ are active, and the server transmits its current model, θ^{t-1}, to these devices. Each active device then optimizes a local empirical risk objective, which is the sum of its local empirical loss and a penalized risk function. The penalized risk, which is dynamically updated, is based on current local device model, and the received server model:

$$\boldsymbol{\theta}_k^t = \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \left[\mathfrak{R}_k(\boldsymbol{\theta}; \boldsymbol{\theta}_k^{t-1}, \boldsymbol{\theta}^{t-1}) \triangleq L_k(\boldsymbol{\theta}) - \langle \nabla L_k(\boldsymbol{\theta}_k^{t-1}), \boldsymbol{\theta} \rangle + \frac{\alpha}{2}\|\boldsymbol{\theta} - \boldsymbol{\theta}^{t-1}\|^2 \right]. \quad (1)$$

Devices compute their local gradient, $\nabla L_k\left(\boldsymbol{\theta}_k^{t-1}\right)$, recursively, by noting that the first order condition for local optima must satisfy,

$$\nabla L_k(\boldsymbol{\theta}_k^t) - \nabla L_k(\boldsymbol{\theta}_k^{t-1}) + \alpha(\boldsymbol{\theta}_k^t - \boldsymbol{\theta}^{t-1}) = \mathbf{0} \tag{2}$$

Stale devices do not update their models. Updated device models, $\boldsymbol{\theta}_k^t,\ k \in \mathcal{P}_t$ are then transmitted to server, which then updates its model to $\boldsymbol{\theta}^t$ as displayed in Algorithm 1.

Intuitive Justification. To build intuition into our method, we first highlight a fundamental issue about the Federated Dynamic Regularizer setup. It is that stationary points for device losses, in general, do not conform to global losses. Indeed, a global stationary point, $\boldsymbol{\theta}^*$ must necessarily satisfy,

$$\nabla \ell(\boldsymbol{\theta}^t) \triangleq \frac{1}{m}\sum_{k\in[m]} \nabla L_k(\boldsymbol{\theta}_*) = \sum_{k\in[m]} \mathbb{E}_{(\boldsymbol{x},y)\sim\mathcal{D}_k} \nabla \ell_k(\boldsymbol{\theta}_*; (\boldsymbol{x}, y)) = \mathbf{0}. \tag{3}$$

In contrast a device's stationary point, $\boldsymbol{\theta}_k^*$ satisfies, $\nabla L_k(\boldsymbol{\theta}_k^*) = \mathbf{0}$, and in general due to heterogeneity of data ($P_k \neq P_j$ for $k \neq j$), the individual device-wise gradients are non-zero $\nabla L_k(\boldsymbol{\theta}_*) \neq \mathbf{0}$. This means that the dual goals of (i) seeking model convergence to a consensus, namely, $\boldsymbol{\theta}_k^t \to \boldsymbol{\theta}^t \to \boldsymbol{\theta}_*$, and (ii) the fact that model updates are based on optimizing local empirical losses is inconsistent[2].

Dynamic Regularization. Our proposed risk objective in Eq. 1 dynamically modifies local loss functions, so that, if in fact local models converge to a consensus, the consensus point is consistent with stationary point of the global loss. To see this, first note that if we initialize at a consensus point, namely, $\boldsymbol{\theta}_k^{t-1} = \boldsymbol{\theta}^{t-1}$, we have, $\nabla\mathfrak{R}(\boldsymbol{\theta}, \boldsymbol{\theta}_k^{t-1}, \boldsymbol{\theta}^{t-1}) = \mathbf{0}$ for $\boldsymbol{\theta} = \boldsymbol{\theta}^{t-1}$. Thus our choice can be seen as modifying the device loss so that the stationary points of device risk is consistent with server model.

Key Property of Algorithm 1. If local device models converge, they converge to the server model, and the convergence point is a stationary point of the global loss. To see this, observe from Eq 2 that if $\boldsymbol{\theta}_k^t \to \boldsymbol{\theta}_k^\infty$, it generally follows that, $\nabla L_k(\boldsymbol{\theta}_k^t) \to \nabla L_k(\boldsymbol{\theta}_k^\infty)$, and as a consequence, we have $\boldsymbol{\theta}^t \to \boldsymbol{\theta}_k^\infty$. In turn this implies that $\boldsymbol{\theta}_k^\infty \triangle \boldsymbol{\theta}^\infty$, i.e., is independent of k. Putting all of this together with our server update equations we have that $\boldsymbol{\theta}^t$ convergence implies $\boldsymbol{h}^t \to 0$. Now the server state $\boldsymbol{h}^t \triangleq \sum_k \nabla L_k(\boldsymbol{\theta}_k^t)$, and as such in the limit we are left with $\sum_k \nabla L_k(\boldsymbol{\theta}_k^t) \to \sum_k \nabla L_k(\boldsymbol{\theta}^\infty) = \mathbf{0}$. This implies that we converge to a point that turns out to be a stationary point of the global risk.

2.1 Convergence Analysis of FedDyn.

Properties outlined in the previous section, motivates our FedDyn convergence analysis of device and server models. We will present theoretical results for convex and non-convex functions.

Theorem 1. *Assuming a constant number of devices are selected uniformly at random in each round, $|\mathcal{P}_t| = P$, for a suitably chosen of $\alpha > 0$, Algorithm 1 satisfies,*

- *Convex and L smooth $\{L_k\}_{k=1}^m$ functions,*

$$E\left[\ell\left(\frac{1}{T}\sum_{t=0}^{T-1}\boldsymbol{\gamma}^t\right) - \ell_*\right] = O\left(\frac{1}{T}\sqrt{\frac{m}{P}}\left(L\left\|\boldsymbol{\theta}^0 - \boldsymbol{\theta}_*\right\| + \frac{1}{L}\frac{1}{m}\sum_{k\in[m]}\|\nabla L_k(\boldsymbol{\theta}_*)\|^2\right)\right)$$

- *Non convex and L smooth $\{L_k\}_{k=1}^m$ functions,*

$$E\left[\left\|\frac{1}{T}\sum_{t=1}^{T}\nabla\ell(\boldsymbol{\gamma}^{t-1})\right\|^2\right] = O\left(\frac{1}{T}\left(L\frac{m}{P}\left(\ell(\boldsymbol{\theta}^0) - \ell_*\right) + L^2\frac{1}{m}\sum_{k\in[m]}E\|\boldsymbol{\theta}_k^0 - \boldsymbol{\theta}^0\|^2\right)\right)$$

where $\boldsymbol{\gamma}^t = \frac{1}{P}\sum_{k\in\mathcal{P}_t}\boldsymbol{\theta}_k^t$, $\boldsymbol{\theta}_* = \arg\min_{\boldsymbol{\theta}} \ell(\boldsymbol{\theta})$ and $\ell_* = \ell(\boldsymbol{\theta}_*)$.

[2] As pointed in related work prior works based on SGD implicitly account for the inconsistency by performing inexact minimization, and additional hyperparameter tuning.

Theorem 1 gives rates for convex and non convex local losses. For convex and smooth functions, in expectation, the global loss of active device averages, convergences at a rate $O\left(\frac{1}{T}\sqrt{\frac{m}{P}}\right)$. Following convention, this rate is for the empirical loss averaged across devices. As such this rate would hold with moderate data imbalance. In situations with significant imbalance, which scales with data size, these results would have to account for the variance in the amount of data/device. Furthermore, the $\sqrt{\frac{m}{P}}$ factor might appear surprising, but note that our bounds hold under expectation, namely, the error reflects the average over all random choices of devices. Similarly, for non convex and smooth functions, in expectation, average of active device models converges to a stationary point at $O\left(\frac{1}{T}\frac{m}{P}\right)$ rate. The expectation is taken over randomness in active device set at each round. Similar to known convergence theorems, the problem dependent constants are related to how good the algorithm is initialized. We refer to Appendix B for a detailed proof.

FedDyn vs. SCAFFOLD (Karimireddy et al., 2019). While SCAFFOLD appears to be similar to our method, there are fundamental differences. Practically, SCAFFOLD communicates twice as many bits as FedDyn or Federated Dynamic Regularizer, transmitting back and forth, both a model and its gradient. The $2\times$ increase in bits can be substantial for many low-power IoT applications, since energy consumption for communication dominates computation. Conceptually, we attribute the increased bit-rate to algorithmic differences. At the device-level, our modified risk incorporates a linear term, $\nabla L_k(\boldsymbol{\theta}_k^t)$ (which we can compute readily (Eq. 2)). Applying our perspective to SCAFFOLD, we see SCAFFOLD as replacing our linear term $\left\langle \nabla L_k(\boldsymbol{\theta}_k^t), \boldsymbol{\theta} \right\rangle$ with $\left\langle \nabla L_k(\boldsymbol{\theta}^t) - \frac{1}{m}\sum_{k\in[m]} \nabla L_k(\boldsymbol{\theta}_k^t), \boldsymbol{\theta} \right\rangle$. While $\nabla L_k(\boldsymbol{\theta}^t)$ can be locally computed, after $\boldsymbol{\theta}^t$ is received, the term $\frac{1}{m}\sum_{k\in[m]} \nabla L_k(\boldsymbol{\theta}_k^t)$ is unknown and must be transmitted by the server, leading to increased bit-rate. Note that this is unavoidable, since ignoring this term, leads to freezing device updates (optimizing $L_k(\boldsymbol{\theta}) - \left\langle \nabla L_k(\boldsymbol{\theta}^t), \boldsymbol{\theta} - \boldsymbol{\theta}^t \right\rangle + \frac{\alpha}{2}\|\boldsymbol{\theta} - \boldsymbol{\theta}^t\|^2$ results in $\boldsymbol{\theta} = \boldsymbol{\theta}^t$). This extra term is a surrogate for $\nabla \ell(\boldsymbol{\theta}^t)$, which is unavailable. As such we believe that these differences are responsible for FedDyn's improved rate (in rounds) in theory as well as practice.

Finally, apart from conceptual differences, there are also implementation differences. SCAFFOLD runs SGD, and adapts hyperparameter tuning for a given number of rounds to maximize accuracy. In contrast, our approach, based on exact minimization, is agnostic to specific implementation, and as such we utilize significantly less tuning.

3 EXPERIMENTS

Our goal in this section is to evaluate FedDyn against competing methods on benchmark datasets for various FL scenarios. Our results will highlight tradeoffs and benefits of our exact minimization relative to prior inexact minimization methods. To ensure a fair comparison, the usual SGD procedure is adapted for the FedDyn algorithm in the device update as in FedAvg rather than leveraging an off the shelf optimization solver. We provide a brief description of the datasets and the models used in the experiments. A detailed description of our setup can be found in Appendix A.1. Partial participation was handled by sampling devices at random in each round independent of previous rounds.

Datasets. We used benchmark datasets with the same train/test splits as in previous works (McMahan et al., 2017; Li et al., 2020a) which are MNIST (LeCun et al., 1998), CIFAR-10, CIFAR-100 (Krizhevsky et al., 2009), a subset of EMNIST (Cohen et al., 2017) (EMNIST-L), Shakespeare (Shakespeare, 1994) as well as a synthetic dataset. The IID split is generated by randomly assigning datapoints to the devices. The Dirichlet distribution is used on the label ratios to ensure uneven label distributions among devices for non-IID splits as in (Yurochkin et al., 2019). For example, in MNIST, 100 device experiments, each device has about 5 and 3 classes that consume 80% of local data at Dirichlet parameter settings of 0.6 and 0.3 respectively. To generate unbalanced data, we sample the number of datapoints from a lognormal distribution. Controlling the variance of lognormal distribution gives unbalanced data. For instance, in CIFAR-10, 100 device experiments, balanced and unbalanced data settings have standard deviation of device sample size of 0 and 0.3 respectively.

Models. We use fully-connected neural network architectures for MNIST and EMNIST-L with 2 hidden layers. The number of neurons in the layers are 200 and 100; and the models achieve 98.4% and 95.0% test accuracy in MNIST and EMNIST-L respectively. The model used for MNIST is the same as used in (McMahan et al., 2017). For CIFAR-10 and CIFAR-100, we use a CNN model, similar to (McMahan et al., 2017), consisting of 2 convolutional layers with 64 5×5 filters followed

Table 1: Number of parameters transmitted relative to one round of FedAvg to reach target test accuracy for moderate and large number of devices in IID and Dirichlet .3 settings. SCAFFOLD communicates the current model and its associated gradient per round, while others communicate only the current model. As such number of rounds for SCAFFOLD is one half of those reported.

Device Number	Dataset	Acc.	FedDyn	SCAFFOLD	FedAvg	FedProx
Moderate	**IID**					
	CIFAR-10	84.5	637	1852(2.9×)	1000+(>1.6)	1000+(>1.6)
		82.3	240	512(2.1×)	994(4.1×)	901(3.8×)
	CIFAR-100	51.0	522	1854(3.6×)	1000+(>1.9×)	1000+(>1.9×)
		40.9	159	286(1.8×)	822(5.2×)	989(6.2×)
	Dirichlet (.3)					
	CIFAR-10	82.5	444	1880(4.2×)	1000+(>2.3×)	1000+(>2.3×)
		80.7	232	594(2.6×)	863(3.7×)	858(3.7×)
	CIFAR-100	51.0	561	1884(3.4×)	1000+(>1.8×)	1000+(>1.8×)
		42.3	170	330(1.9×)	959(5.6×)	859(5.1×)
Massive	**IID**					
	CIFAR-10	80.0	840	4000+(>4.8×)	2000+(>2.4×)	2000+(>2.4×)
		62.3	305	928(3.0×)	1277(4.2×)	1549(5.1×)
	CIFAR-100	50.1	1445	3982(2.8×)	2000+(>1.4×)	2000+(>1.4×)
		38.3	477	1408(3.0×)	1997(4.2×)	1956(4.1×)
	Dirichlet (.3)					
	CIFAR-10	80.0	831	4000+(>4.8×)	2000+(>2.4×)	2000+(>2.4×)
		70.6	350	2138(6.1×)	1962(5.6×)	1954(5.6×)
	CIFAR-100	47.0	969	4000(4.1×)	2000+(>2.1×)	2000+(>2.1×)
		39.9	467	2266(4.9×)	1913(4.1×)	1984(4.2×)

by 2 fully connected layers with 394 and 192 neurons, and a softmax layer. The model achieves 85.2% and 55.3% test accuracy for CIFAR-10 and CIFAR-100 respectively. For the next character prediction task (Shakespeare), we use a stacked LSTM, similar to (Li et al., 2020a). This architecture achieves a test accuracy of 50.8% and 51.2% in IID and non-IID settings respectively. Both IID and non-IID performances are reported since splits are randomly regenerated from the entire Shakespeare writing. Hence centralized data and the centralized model performance is different.

In passing, we note that while the accuracies reported are state-of-art for our chosen models, higher capacity models can achieve higher performance on these datasets. As such, our aim is to compare the relative performance of these models in FL using FedDyn and other strong baselines.

Comparison of Methods. We report the performance of FedDyn, SCAFFOLD, FedAvg and FedProx on synthetic and real datasets. We also experimented with distributed SGD, where devices in each round compute gradients on the server supplied model on local data, and communicate these gradients. Its performance was not competitive relative to other methods. Therefore, we do not tabulate it here. We cover synthetic data generation and its results in Appendix A.1.

The standard goal in FL is to minimize amount of bits transferred. For this reason, we adopt the number of models transmitted to achieve a target accuracy as our metric. This metric is different than comparing communication rounds since not all methods communicate the same amount of information per round. FedDyn, FedAvg and FedProx transmit/receive the same amount of models for a fixed number of rounds whereas SCAFFOLD costs twice due to transmission of states. We compare algorithms for two different accuracy levels which we pick them to be close to performance obtained by centralizing data. Along with transmission costs of each method, we report the communication savings of FedDyn compared to each baseline in parenthesis. For methods that could not achieve aimed accuracy within the communication constraint, we append transmission cost with $+$ sign. We observe FedDyn results in communication savings compared to all baselines to reach a target accuracy. We test FedDyn under the four characteristic properties of FL which are partial participation, large number of devices, heterogeneous data, and unbalanced data.

Moderate vs. Large Number of Devices. *FedDyn significantly outperforms competing methods in the practically relevant massively distributed scenario.* We report the performance of FedDyn on CIFAR-10 and CIFAR-100 with moderate and large number of devices in Table 1, while keeping the participation level constant (10%) and the data amounts balanced. Specifically, the moderately distributed setting has 100 devices with 500 images per device. The massively distributed setting has 1000 devices with 50 images per device for CIFAR-10, as well as 500 devices with 100 images per

device for CIFAR-100. In each distributed setting, the data is partitioned in both IID and non-IID (Dirichlet 0.3) fashion. FedDyn leads to substantial transmission reduction in each of the regimes.

First, the communication saving in the massive setting is significantly larger relative to the moderate setting. Compared to SCAFFOLD, FedDyn leads to $4.8\times$ and $2.9\times$ gains respectively on CIFAR-10 IID setting. SCAFFOLD is not able to achieve 80% within 2000 rounds in the massive setting (shown in Figure 2a), thus actual saving is more than $4.8\times$. Similar trend is observed in the non-IID setting of CIFAR-10 and CIFAR-100. Second, all the methods require more communications to achieve a reasonable accuracy in the massive setting as the dataset is more decentralized. For instance, it takes FedDyn 637 rounds to achieve 84.5% with 100 devices, while it takes 840 rounds to achieve 80.0% with 1000 devices. Similar trend is observed for CIFAR-100 and other methods. FedDyn always achieves the target accuracy with fewer rounds and thus leads to significant saving. Third, a higher target accuracy may result in a greater saving. For instance, the saving relative to SCAFFOLD increases from $3\times$ to $4.8\times$ in the CIFAR-10 IID massive setting. We may attribute this to the fact that FedDyn aligns device functions to global loss and efficiently optimizes the problem.

Full vs. Partial Participation Levels. *FedDyn outperforms baseline methods across different device participation levels.* We consider different device participation levels with 100 devices and balanced data in Table 2 where part of CIFAR-10 and CIFAR-100 results are omitted since they are reported in moderate number of devices section of Table 1. The Shakespeare non-IID results are separately shown, since it has a natural non-IID split which does not conform with the Dirichlet distribution. The communication gain, with respect to best baseline, increases with greater participation levels from $2.9\times$ to $9.4\times$; $4.0\times$ to $12.8\times$ and $4.2\times$ to $7.9\times$ for CIFAR-10 in different device distribution settings. We observe a similar performance increase in full participation for most of the datasets. This validates our hypothesis that FedDyn more efficiently incorporates information from all devices compared to other methods, and results in more savings in full participation. Similar to previous results, a greater target accuracy gives a greater savings in most of the settings. We also report results for 1% participation regime with different device distribution settings (See Table 5 in Appendix A.1).

Balanced vs. Unbalanced Data. *FedDyn is more robust to unbalanced data than competing methods.* We fix number of devices (100) and participation level (10%) and consider effect of unbalanced data (Table 4 (Appendix A.1)). FedDyn achieves $4.3\times$ gains over the best competitor, SCAFFOLD to achieve the target accuracy. As before, gains increase with the target accuracy.

IID vs. non-IID Device Distribution. *FedDyn outperforms baseline methods across different device distribution levels.* We consider heterogeneous device distributions in the context of varying device numbers, participation levels and balanced-unbalanced settings in Table 1, 2 and 4 (Appendix A.1) respectively. Device distributions become more non-IID as we go from IID, Dirichlet .6 to Dirichlet .3 splits which makes global optimization problem harder. We see a clear effect of this change in Table 2 for 10% participation level and in Table 4 for unbalanced setting. For instance, increasing non-IID level results in a greater communication saving such as from $2.9\times$, $4.0\times$ to $4.2\times$ in CIFAR-10 10% participation. Similar statement holds for MNIST, EMNIST-L and Shakespeare in Table 2 and for CIFAR-10 unbalanced setting in Table 4. We do not observe a significant difference in savings for full participation setting in Table 2.

Summary. Overall, FedDyn consistently leads to substantial communication savings compared to baseline methods uniformly across various FL regimes of interest. We realize large gains in the practically relevant massively distributed data setting.

4 Conclusion

We proposed FedDyn, a novel FL method for distributively training neural network models. FedDyn is based on exact minimization, wherein at each round, each participating device, dynamically updates its regularizer so that the optimal model for the regularized loss is in conformity with the global empirical loss. Our approach is different from prior works that attempt to parallelize gradient computation, and in doing so they tradeoff target accuracy with communications, and necessitate inexact minimization. We investigate different characteristic FL settings to validate our method. We demonstrate both through empirical results on real and synthetic data as well as analytical results that our scheme leads to efficient training with convergence rate as $O\left(\frac{1}{T}\right)$ where T is number of rounds, in both convex and non-convex settings, while being fully agnostic to device heterogeneity and robust to large number of devices, partial participation and unbalanced data.

Table 2: Number of parameters transmitted relative to one round of FedAvg to reach target test accuracy for 100% and 10% participation regimes in the IID, non-IID settings. SCAFFOLD communicates the current model and its associated gradient per round, while others communicate only the current model. As such number of rounds for SCAFFOLD is one half of those reported.

Participation	Dataset	Acc.	FedDyn	SCAFFOLD	FedAvg	FedProx
100%	**IID**					
	CIFAR-10	85.0	198	1860(9.4×)	1000+(>5.1×)	1000+(>5.1×)
		81.4	67	320(4.8×)	754(11.3×)	653(9.7×)
	CIFAR-100	51.0	259	1744(6.7×)	1000+(>3.9×)	1000+(>3.9×)
		39.4	55	172(3.1×)	998(18.1×)	635(11.5×)
	MNIST	98.2	38	72(1.9×)	292(7.7×)	302(7.9×)
		97.2	9	18(2.0×)	32(3.6×)	35(3.9×)
	EMNIST-L	94.6	65	414(6.4×)	307(4.7×)	500(7.7×)
		93.6	16	36(2.2×)	66(4.1×)	66(4.1×)
	Shakespeare	46.4	33	74(2.2×)	96(2.9×)	148(4.5×)
		45.4	28	64(2.3×)	59(2.1×)	62(2.2×)
	Dirichlet (.6)					
	CIFAR-10	84.0	148	1890(12.8×)	1000+(>6.8×)	1000+(>6.8×)
		80.3	64	392(6.1×)	869(13.6×)	714(11.2×)
	CIFAR-100	51.0	468	1838(3.9×)	1000+(>2.1×)	1000+(>2.1×)
		40.6	73	206(2.8×)	998(13.7×)	546(7.5×)
	MNIST	98.2	88	200(2.3×)	492(5.6×)	466(5.3×)
		97.2	12	28(2.3×)	51(4.2×)	55(4.6×)
	EMNIST-L	94.9	207	552(2.7×)	410(2.0×)	295(1.4×)
		93.9	20	42(2.1×)	73(3.6×)	70(3.5×)
	Dirichlet (.3)					
	CIFAR-10	83.5	223	1762(7.9×)	1000+(>4.5×)	1000+(>4.5×)
		80.2	70	504(7.2×)	705(10.1×)	949(13.6×)
	CIFAR-100	50.5	405	1940(4.8×)	1000+(>2.5×)	1000+(>2.5×)
		41.0	80	224(2.8×)	911(11.4×)	678(8.5×)
	MNIST	98.2	47	148(3.1×)	495(10.5×)	422(9.0×)
		97.2	11	26(2.4×)	50(4.5×)	57(5.2×)
	EMNIST-L	94.5	65	210(3.2×)	492(7.6×)	323(5.0×)
		93.5	23	46(2.0×)	78(3.4×)	87(3.8×)
	Non-IID					
	Shakespeare	47.3	33	70(2.1×)	134(4.1×)	131(4.0×)
		46.3	28	62(2.2×)	53(1.9×)	64(2.3×)
10%	**IID**					
	MNIST	98.2	100	142(1.4×)	588(5.9×)	526(5.3×)
		97.2	31	52(1.7×)	49(1.6×)	52(1.7×)
	EMNIST-L	94.6	104	160(1.5×)	330(3.2×)	303(2.9×)
		93.6	58	84(1.4×)	69(1.2×)	75(1.3×)
	Shakespeare	46.9	63	94(1.5×)	138(2.2×)	250(4.0×)
		45.9	56	76(1.4×)	96(1.7×)	75(1.3×)
	Dirichlet (.6)					
	CIFAR-10	83.5	403	1618(4.0×)	1000+(>2.5×)	1000+(>2.5×)
		81.3	189	486(2.6×)	977(5.2×)	992(5.2×)
	CIFAR-100	51.0	521	1910(3.7×)	1000+(>1.9×)	1000+(>1.9×)
		41.6	170	302(1.8×)	931(5.5×)	852(5.0×)
	MNIST	98.1	129	194(1.5×)	581(4.5×)	295(2.3×)
		97.1	37	60(1.6×)	57(1.5×)	61(1.6×)
	EMNIST-L	94.9	192	296(1.5×)	306(1.6×)	407(2.1×)
		93.9	55	102(1.9×)	95(1.7×)	88(1.6×)
	Dirichlet (.3)					
	MNIST	98.2	109	214(2.0×)	621(5.7×)	506(4.6×)
		97.2	39	70(1.8×)	76(1.9×)	79(2.0×)
	EMNIST-L	94.4	107	178(1.7×)	804(7.5×)	330(3.1×)
		93.4	58	100(1.7×)	81(1.4×)	88(1.5×)
	Non-IID					
	Shakespeare	47.6	63	102(1.6×)	169(2.7×)	242(3.8×)
		46.6	56	82(1.5×)	80(1.4×)	70(1.2×)

REFERENCES

Sebastian Caldas, Peter Wu, Tian Li, Jakub Konecný, H. Brendan McMahan, Virginia Smith, and Ameet Talwalkar. LEAF: A benchmark for federated settings. *CoRR*, abs/1812.01097, 2018. URL http://arxiv.org/abs/1812.01097.

Gregory Cohen, Saeed Afshar, Jonathan Tapson, and Andre Van Schaik. Emnist: Extending mnist to handwritten letters. In *2017 International Joint Conference on Neural Networks (IJCNN)*, pp. 2921–2926. IEEE, 2017.

Malka N Halgamuge, Moshe Zukerman, Kotagiri Ramamohanarao, and Hai L Vu. An estimation of sensor energy consumption. *Progress in Electromagnetics Research*, 12:259–295, 2009.

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

Tzu-Ming Harry Hsu, Hang Qi, and Matthew Brown. Measuring the effects of non-identical data distribution for federated visual classification. *CoRR*, abs/1909.06335, 2019. URL http://arxiv.org/abs/1909.06335.

Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Keith Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. Advances and open problems in federated learning. *arXiv preprint arXiv:1912.04977*, 2019.

Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank J. Reddi, Sebastian U. Stich, and Ananda Theertha Suresh. SCAFFOLD: stochastic controlled averaging for on-device federated learning. *CoRR*, abs/1910.06378, 2019. URL http://arxiv.org/abs/1910.06378.

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

Alex Krizhevsky et al. Learning multiple layers of features from tiny images. *Technical report*, 2009.

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

T. Li, A. K. Sahu, A. Talwalkar, and V. Smith. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, 37(3):50–60, 2020.

Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smithy. Feddane: A federated newton-type method. In *2019 53rd Asilomar Conference on Signals, Systems, and Computers*, pp. 1227–1231. IEEE, 2019.

Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. In *Proceedings of Machine Learning and Systems 2020*, pp. 429–450, 2020a.

Xiang Li, Kaixuan Huang, Wenhao Yang, Shusen Wang, and Zhihua Zhang. On the convergence of fedavg on non-iid data. In *International Conference on Learning Representations*, 2020b. URL https://openreview.net/forum?id=HJxNAnVtDS.

Ali Makhdoumi and Asuman Ozdaglar. Convergence rate of distributed admm over networks. *IEEE Transactions on Automatic Control*, 62(10):5082–5095, 2017.

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial Intelligence and Statistics*, pp. 1273–1282, 2017.

Sashank Reddi, Zachary Charles, Manzil Zaheer, Zachary Garrett, Keith Rush, Jakub Konečnỳ, Sanjiv Kumar, and H Brendan McMahan. Adaptive federated optimization. *arXiv preprint arXiv:2003.00295*, 2020.

William Shakespeare. The complete works of william shakespeare, 1994. URL http://www.gutenberg.org/files/100/old/1994-01-100.zip.

Ohad Shamir, Nati Srebro, and Tong Zhang. Communication-efficient distributed optimization using an approximate newton-type method. In *International conference on machine learning*, pp. 1000–1008, 2014.

Sebastian Urban Stich. Local SGD converges fast and communicates little. *International Conference on Learning Representations (ICLR)*, pp. arXiv:1805.09767, 2019. URL https://arxiv.org/abs/1805.09767.

Yuxin Wu and Kaiming He. Group normalization. In *Proceedings of the European conference on computer vision (ECCV)*, pp. 3–19, 2018.

Honglin Yuan and Tengyu Ma. Federated accelerated stochastic gradient descent. *arXiv preprint arXiv:2006.08950*, 2020.

Mikhail Yurochkin, Mayank Agarwal, Soumya Ghosh, Kristjan Greenewald, Nghia Hoang, and Yasaman Khazaeni. Bayesian nonparametric federated learning of neural networks. In *International Conference on Machine Learning*, pp. 7252–7261, 2019.

Yue Zhao, Meng Li, Liangzhen Lai, Naveen Suda, Damon Civin, and Vikas Chandra. Federated learning with non-iid data. *arXiv preprint arXiv:1806.00582*, 2018.