LocalNewton: Reducing Communication Rounds for Distributed Learning

Vipul Gupta[†] **Avishek Ghosh**[†] **Michal Derezinski**[‡] **Rajiv Khanna**[‡]
Kannan Ramchandran[†] **Michael W. Mahoney**[‡]
[†]Department of EECS, UC Berkeley
[‡]Department of Statistics, UC Berkeley

Abstract

We propose LocalNewton, a distributed second-order algorithm for solving empirical risk minimization problems in a communication-efficient fashion. In LocalNewton, each worker updates its model in every iteration by finding a gradient, a second-order descent direction, and a step-size (through line-search) using only the data and model stored in its local memory. Workers run multiple such iterations locally and communicate the models to the master node only once every few (say L) iterations. LocalNewton is highly practical since it requires only one hyperparameter—the number of local iterations L. We develop novel techniques to prove theoretical convergence guarantees for LocalNewton, and we provide a detailed empirical evaluation. Inspired by theoretical results and empirical observations, we devise an adaptive scheme to choose L. This adaptive scheme reduces the amount of local computation at each worker between two model synchronizations as the training proceeds, successively refining the model quality at the master. Through extensive experiments using several real-world datasets on AWS Lambda (a high-latency serverless computing platform), we show that LocalNewton requires fewer than 60% of the communication rounds (between master and workers) and less than 40% of the end-to-end running time, compared to the state-of-the-art algorithms, to reach the same training loss.

1 INTRODUCTION

An explosion in data generation and data collection capabilities in recent years has resulted in the segregation of computing and storage resources. Distributed machine learning is one example where each worker machine processes only a subset of the data, while the master machine coordinates with workers to learn a good model. Such coordination can be time-consuming since it requires frequent communication between the master and worker nodes, especially for systems that have large compute resources but are bottlenecked by communication costs.

Communication costs in distributed setups can be broadly classified into two types—latency and bandwidth (Demmel, 2013; Gupta et al., 2018). Latency is the fixed cost associated with sending a message and is generally independent of the size of the message. Bandwidth cost, on the other hand, is directly proportional to the size of the message. Many recent works have focused on reducing the bandwidth cost by reducing the size of the gradient or the model to be communicated using techniques such as sparsification (Acharya et al., 2019; Stich et al., 2018), sketching (Ivkin et al., 2019; Konečnỳ et al., 2016) and quantization (Gandikota et al., 2019; Ghosh et al., 2020; Lin et al., 2017; Bernstein et al., 2018). Such schemes that perform inexact updates in each iteration, however, can *increase* the number of iterations required to converge to the same quality model both theoretically and empirically (Acharya et al., 2019; Mayekar and Tyagi, 2020). This can, in turn, increase the total training time in systems where latency costs dominate bandwidth costs.

One example where latency costs outweigh bandwidth costs is federated learning, where the computation is performed locally at the mobile device (which is generally the source of data) due to a high-cost barrier in transferring the data to traditional computing platforms (Kairouz et al., 2019; Konečnỳ et al., 2016). Such mobile resources (e.g., mobile phones, wearable devices, etc.) have reasonable compute power but can be severely limited by communication latency (e.g., inadequate network connection to synchronize frequent model updates). For this reason, schemes like Local Stochastic Gradient Descent (Local SGD) have become popular, since they try to mitigate the communication costs by performing more *local* computation at the worker machines, thus substantially reducing the number of communication rounds required (McMahan et al., 2017).

Serverless systems–such as Amazon Web Services (AWS) Lambda and Microsoft Azure Functions–are yet another example where high communication latency between worker machines dominates the running time of the algorithm. Serverless computing has recently gained a lot of attention from the research community (Baldini et al., 2017; Jonas et al., 2019, 2017; Shankar et al., 2018), with a significant focus on optimization in these settings (Feng et al., 2018; Gupta et al., 2019; Carreira et al., 2019; Wang et al., 2019). Such systems are gaining popularity due to the ease-of-management, greater elasticity and high scalability. These systems use cloud storage (like AWS S3) to store enormous amounts of data, while using a large number of low-quality workers for large-scale computation. Naturally, the communication between the high-latency storage and the commodity workers is extremely slow (e.g., see Jonas et al., 2017; Shankar et al., 2018), resulting in impractical end-to-end times for many popular optimization algorithms such as SGD (Hellerstein et al., 2018; Gupta et al., 2019). Furthermore, communication failures between the cloud storage and serverless workers consistently give rise to stragglers, and this introduces synchronization delays (Gupta et al., 2018, 2020a).

These trends suggest that optimization schemes that reduce communication rounds between workers are highly desirable. In this paper, we focus our attention on devising such schemes for systems with high latency costs but sufficient computing power. To this end, we propose a distributed algorithm, called LocalNewton, that significantly reduces the rounds of communication, and hence, end-to-end training time, required to solve large-scale empirical risk minimization problems on such systems.

Our contributions. Inspired by recent progress in local optimization methods (that reduce communication cost by limiting the frequency of synchronization) and distributed and stochastic second order methods (that use the local curvature information), we propose a local second-order algorithm called LocalNewton. The proposed LocalNewton method saves on communication costs in two ways. First, it updates the models at the master only sporadically, thus requiring only one communication round per multiple iterations. Second, it uses the second-order information to reduce the number of iterations, and hence reduces the overall rounds of communication.

Fig. 1 illustrates the savings due to LocalNewton, where we plot training loss and test accuracy w.r.t. communication rounds, for several popular communication-efficient schemes for logistic regression on the w8a dataset (Chang and Lin, 2011) (see Sec. 4 for detailed experiments). By virtue of using both local iterations and second-order information to improve communication efficiency, LocalNewton reaches close to the optimal training loss very quickly, when compared to schemes like Local SGD (Kairouz et al., 2019), GIANT (Wang et al., 2018) and BFGS (Fletcher, 2013). Important features of LocalNewton include:

1. *Simplicity*: In LocalNewton, each worker takes a few Newton steps (Boyd and Vandenberghe, 2004) on local data, agnostic of other workers. These local models are averaged once every $L(\geqslant 1)$ iterations at the master node.
2. *Practicality*: Unlike many first-order and distributed second-order schemes, LocalNewton does not require hyperparameter tuning for step-size, mini-batch size, etc., and the only hyperparameter required is the number of local iterations L. Further, we also propose an adaptive version of LocalNewton which automatically reduces L as the training proceeds by monitoring the training loss at the master.
3. *Provable convergence*: In general, proving convergence guarantees for local algorithms is not straightforward. Only recently, it has been proved (Stich, 2018; Haddadpour et al., 2019; Dieuleveut and Patel, 2019) that local SGD converges as fast as SGD, thereby explaining the well-studied empirical successes (Konečný et al., 2016). In this paper, we develop novel techniques to highlight the convergence behaviour of LocalNewton to the optimal solution.
4. *Reduced training times*: We implement LocalNewton on the serverless environment AWS Lambda using the Pywren framework (Jonas et al., 2017). Through extensive empirical evaluation, we show that the significant savings in terms of communication rounds translate to savings in running time on this high-latency distributed computing environment.

Related Work. In recent years, schemes such as local SGD have gained popularity, as they are communication efficient due to only sporadic model updates at the master (McMahan et al., 2017; Kairouz et al., 2019). Such schemes that show great promise have eluded a thorough theoretical analysis until recently, when it was shown that local SGD converges at the same rate as mini-batch SGD (Stich, 2018; Haddadpour et al., 2019; Dieuleveut and Patel, 2019). Similar

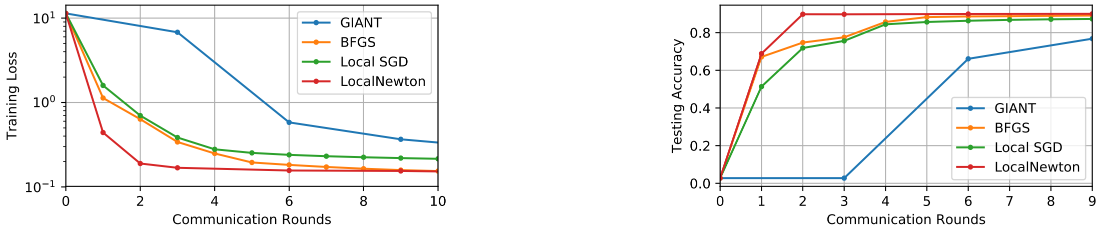


Figure 1: Training loss and testing accuracy versus communication rounds for LocalNewton and existing schemes for communication-efficient optimization (more experiments, along with wall-clock time results, are given in Section 4).

ideas that reduce communication by averaging the local models sporadically have also been applied in training neural networks to improve the training times and/or model performance. (Lin et al., 2020; Gupta et al., 2020b). Apart from local first-order methods, many communication-efficient distributed second-order (also know as Newton-type) algorithms have been recently proposed (Wang et al., 2018; Shamir et al., 2014; Zhang and Lin, 2015; Reddi et al., 2015; Smith et al., 2016; Dereziński and Mahoney, 2019; Dereziński et al., 2020). Such methods use both the gradient and the curvature information to provide an order of improvement in convergence, compared to vanilla first-order methods. This is done at the cost of more local computation per iteration, which is ideal for systems with high communication latency. However, such algorithms require at least two communication rounds (for averaging gradients and the second-order descent direction), and a thorough knowledge of a fundamental trade-off between communication and local computation is still lacking for these methods.

2 PROBLEM SETUP

Notation: We begin by defining our notation. Throughout the paper, vectors (e.g., $\mathbf{g}$) and matrices (e.g., $\mathbf{H}$) are represented as bold lowercase and uppercase letters, respectively. For a vector $\mathbf{g}$, $\|\mathbf{g}\|$ denotes its ℓ_2 norm and $\|\mathbf{H}\|_2$ denotes the spectral norm of matrix $\mathbf{H}$. The identity matrix is denoted as $\mathbf{I}$, and the set $\{1, 2, \cdots, n\}$ is denoted as $[n]$ for all positive integers n. Further, we use superscript (e.g., $\mathbf{g}^k$) to denote the worker index and subscript (e.g., $\mathbf{g}_t$) to denote the iteration counter, unless stated otherwise.

We are interested in solving empirical risk minimization problems of the following form in a distributed fashion

$$\min_{\mathbf{w}\in\mathbb{R}^d} \left\{ f(\mathbf{w}) \triangleq \frac{1}{n}\sum_{j=1}^{n} f_j\left(\mathbf{w}\right) \right\}, \tag{1}$$

where $f_j(\cdot) : \mathbb{R}^d \to \mathbb{R}$, for all $j \in [n] = \{1, 2, \cdots, n\}$, models the loss of the j-th observation given an underlying parameter estimate $\mathbf{w} \in \mathbb{R}^d$. In machine learning, such problems arise frequently, e.g. logistic and linear regression, support vector machines, neural networks and graphical models. Specifically, in the case of logistic regression,

$$f_j(\mathbf{w}) = \ell_j(\mathbf{w}^T\mathbf{x}_j) = \log(1 + e^{-y_j\mathbf{w}^T\mathbf{x}_j}) + \frac{\gamma}{2}\|\mathbf{w}\|^2,$$

where $\ell_j(\cdot)$ is the loss function for sample $j \in [n]$ and γ is an appropriately chosen regularization parameter. Also, $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n] \in \mathbb{R}^{d\times n}$ is the sample matrix containing the input feature vectors $\mathbf{x}_j \in \mathbb{R}^d, j \in [n]$, $\mathbf{y} = [y_1, y_2, \cdots, y_n]$ is the corresponding label vector. Hence, $(\mathbf{x}_j, y_j)$ together define the j-th observation and $(\mathbf{X}, \mathbf{y})$ define the training dataset.

For such problems, the gradient and the Hessian at the t-th iteration are given by

$$\mathbf{g}_t = \nabla f\left(\mathbf{w}_t\right) = \frac{1}{n}\sum_{j=1}^{n} \nabla f_j(\mathbf{w}_t) \text{ and}$$

$$\mathbf{H}_t = \nabla^2 f\left(\mathbf{w}_t\right) = \frac{1}{n}\sum_{j=1}^{n} \nabla^2 f_j\left(\mathbf{w}_t\right),$$

respectively, where $\mathbf{w}_t$ is the model estimate at the t-th iteration. Next, we delineate some assumptions on $f(\cdot)$ required to prove theoretical convergence of the proposed method.

Assumptions: We make the following standard assumptions on the objective function $f(\cdot)$ for all $\mathbf{w} \in \mathbb{R}^d$:

1. $f_i(\cdot)$, for all $i \in [n]$, is twice differentiable.
2. $f(\cdot)$ is κ-strongly convex, that is, $\nabla^2 f(\mathbf{w}) \succcurlyeq \kappa\mathbf{I}$.
3. $f(\cdot)$ is M-smooth, that is, $\nabla^2 f(\mathbf{w}) \preccurlyeq M\mathbf{I}$.
4. $\|\nabla^2 f_i(\cdot)\|_2, i \in [n]$, is upper bounded. That is, $\nabla^2 f_i(\mathbf{w}) \preccurlyeq B\mathbf{I}$, for all $i \in [n]$.
5. $f_i(.)$ has bounded gradient, i.e., $\|\nabla f_i(.)\| \leqslant \Gamma \ \forall \ i$.

Data distribution at each worker: Let there be a total of K workers. We assume that the k-th worker is assigned a subset $\mathcal{S}_k \subset [n]$, for all $k \in [K] = \{1, 2, \cdots, K\}$, of the n data points, chosen uniformly at random without replacement.[1] Let the number of samples at each worker be $s = |\mathcal{S}_k| \;\forall\; k \in [K]$, where $s \ll n$ in practice. Also, by the virtue of sampling without replacement, we have $\mathcal{S}_1 \cup \mathcal{S}_2 \cup \cdots \cup \mathcal{S}_K = [n]$ and $S_i \cap S_j = \Phi$ for all $i, j \in [K]$. Hence, the number of workers is given by $K = n/s$.

Thus, at the k-th worker in the t-th iteration, the local function value (at the local iterate $\mathbf{w}_t^k$) is

$$f^k(\mathbf{w}_t^k) = \frac{1}{s}\sum_{j \in \mathcal{S}_k} f_j\left(\mathbf{w}_t^k\right). \tag{2}$$

The k-th worker tries to minimize the local function value in Eq 2 in each iteration. The corresponding local gradient $\mathbf{g}_t^k$ and local Hessian $\mathbf{H}_t^k$, respectively, at k-th worker in t-th iteration can be written as

$$\mathbf{g}_t^k = \nabla f^k(\mathbf{w}_t^k) = \frac{1}{s}\sum_{j \in \mathcal{S}_k} \nabla f_j(\mathbf{w}_t^k) \text{ and}$$

$$\mathbf{H}_t^k = \nabla^2 f^k(\mathbf{w}_t^k) = \frac{1}{s}\sum_{j \in \mathcal{S}_k} \nabla^2 f_j\left(\mathbf{w}_t^k\right).$$

In the following lemma, we make use of matrix concentration inequalities to show that, for sufficiently large sample size s, the local Hessian at each worker is also strongly convex and smooth with high probability.

Lemma 2.1. *Let $f(\cdot)$ satisfy assumptions 1-4 and $0 < \epsilon \leqslant 1/2$ and $0 < \delta < 1$ be fixed constants. Then, if $s \geqslant \frac{4B}{\kappa\epsilon^2}\log\frac{2d}{\delta}$, the local Hessian at the k-th worker satisfies*

$$(1-\epsilon)\kappa \leqslant \nabla^2 f^k(\mathbf{w}) = \mathbf{H}^k(\mathbf{w}) \leqslant (1+\epsilon)M, \tag{3}$$

for all $\mathbf{w} \in \mathbb{R}^d$ and $k \in [K]$ with probability (w.p.) at least $1-\delta$.

Proof. See Appendix C. □

2.1 LocalNewton: The Algorithm

We consider synchronous second-order methods for distributed learning where local models are synced after every L iterations. Let $\mathcal{I}_t \subseteq [t]$ be the set of indices where the model is synced, that is, $\mathcal{I}_t = [0, L, 2L, \cdots, t_0]$, where t_0 is the last iteration just before t where the models were synced. Let us consider the following LocalNewton update at the k-th worker and $(t+1)$-th iteration:

$$\mathbf{w}_{t+1}^k = \begin{cases} \mathbf{w}_t^k - \alpha_t^k \mathbf{H}^k(\mathbf{w}_t^k)^{-1}\mathbf{g}^k(\mathbf{w}_t^k), & \text{if } t \notin \mathcal{I}_t \\ \bar{\mathbf{w}}_t - \alpha_t^k \mathbf{H}^k(\bar{\mathbf{w}}_t)^{-1}\mathbf{g}^k(\bar{\mathbf{w}}_t), & \text{if } t \in \mathcal{I}_t, \end{cases} \tag{4}$$

where $\bar{\mathbf{w}}_t = \frac{1}{K}\sum_{k=1}^K \mathbf{w}_t^k \;\forall\; t$, and α_t^k is the step-size at the k-th worker at iteration t.[2]

Algorithm 1: LocalNewton in a nutshell

```
Input: Local function $f_k(\cdot)$ at the $k$-th worker;
       Initial iterate $\bar{\mathbf{w}}_0 \in \mathbb{R}^d$; Line search
       parameter $0 < \beta \leqslant 1/2$; Number of
       iterations $T$, Set $\mathcal{I}_T \subseteq \{1, 2, \cdots, T\}$ where
       models are synced
for $k = 1$ to $K$ in parallel do
    Initialization: $\mathbf{w}_0^k = \bar{\mathbf{w}}_0$
    for $t = 0$ to $T - 1$ do
        if $t \in \mathcal{I}_T$ then
            $\bar{\mathbf{w}}_t = \frac{1}{K}\sum_{k=1}^K \mathbf{w}_t^k$ (Master averages the
             local models)
            $\mathbf{w}_t^k = \bar{\mathbf{w}}_t$
        end
        Compute local gradient
         $\mathbf{g}_k(\mathbf{w}_t^k) = \nabla f_k(\mathbf{w}_t^k)$.
        Compute $\mathbf{p}_t^k = \mathbf{H}_k(\mathbf{w}_t^k)^{-1}\mathbf{g}_k(\mathbf{w}_t^k)$ using
         conjugate gradient methods.
        Update $\mathbf{w}_{t+1}^k = \mathbf{w}_t^k - \alpha_t^k \mathbf{p}_t^k$   where $\alpha_t^k$ is
         the step-size obtained from line search
         condition (5).
    end
end
```

Also, define the local descent direction at the k-th worker at iteration t as $\mathbf{p}_t^k = \alpha_t^k \mathbf{H}_k(\mathbf{w}_t^k)^{-1}\mathbf{g}_k(\mathbf{w}_t^k)$ and similarly define $\bar{\mathbf{p}}_t = \frac{1}{K}\sum_{k=1}^K \mathbf{p}_t^k$. We can see that $\bar{\mathbf{w}}_{t+1} = \bar{\mathbf{w}}_t - \bar{\mathbf{p}}_t$. Detailed steps for LocalNewton are provided in Algorithm 1.

Note that $\bar{\mathbf{w}}_t$ is not explicitly calculated for all t, but only for $t \in \mathcal{I}_t$. However, we will use the technique of perturbed iterate analysis and show the convergence of the sequence $f(\bar{\mathbf{w}}_1), f(\bar{\mathbf{w}}_2), \cdots, f(\bar{\mathbf{w}}_t)$ to $f(\mathbf{w}^*)$.

Step-size selection: Let each worker locally choose a step-size according to the following rule

$$\alpha_t^k = \max_{\alpha \leqslant \alpha^\star} \alpha \quad \text{such that}$$

$$f_k\left(\mathbf{w}_t^k - \alpha\mathbf{p}_t^k\right) \leqslant f_k\left(\mathbf{w}_t^k\right) - \alpha\beta(\mathbf{p}_t^k)^T \nabla f_k\left(\mathbf{w}_t^k\right), \tag{5}$$

for some constant $\beta \in (0, 1/2]$, where the parameter $\alpha^\star (\leqslant 1)$ depends on the properties of the objective

[1] Note that this corresponds to simply partitioning the dataset and assigning equal number observations to each worker if the observations are independent and identically distributed. If not, randomly shuffling the observations and then a data-independent partitioning is equivalent to uniform sampling without replacement.

[2] Note that in practice, one need not calculate the exact $\mathbf{H}^k(\mathbf{w}_t^k)^{-1}\mathbf{g}^k(\mathbf{w}_t^k)$ and efficient algorithms like conjugate gradient descent can be used (Shewchuk et al., 1994).

function: [3]

$$\alpha^\star \leqslant \min\left\{\frac{(1-\beta)\kappa}{M}, \frac{2\beta\kappa^2}{3M[M-\kappa/4]}\right\}. \qquad (6)$$

In the next section, we prove convergence guarantees for LocalNewton.

3 CONVERGENCE GUARANTEES

In this section, we state and prove the main theoretical results of the paper. Theorem 3.2 discusses the case when $L = 1$, and Theorem 3.3 discusses the case when $L > 1$.

First, we state the following auxiliary lemma which is required to prove the results in this paper.

Lemma 3.1. *Let the function $f(\cdot)$ satisfy assumptions 1-3, and suppose that step-size α_t^k satisfies the line-search condition in* (5)*. Also, let $0 < \epsilon < 1/2$ and $0 < \delta < 1$ be fixed constants. Moreover, let the sample size $s \geqslant \frac{4B}{\kappa\epsilon^2}\log\frac{2d}{\delta}$. Then, the LocalNewton update, defined in Eq.* (4)*, at the k-th worker satisfies*

$$f^k(\mathbf{w}_{t+1}^k) - f^k(\mathbf{w}_t^k) \leqslant -\psi||\mathbf{g}_t^k||^2 \ \forall\ k \in [K],$$

w.p. at least $1-\delta$, where $\psi = \frac{\alpha^\star\beta}{M(1+\epsilon)}$.

Proof. See Appendix C. □

Next, we use the result in Lemma 3.1 to prove linear convergence for the global function $f(\cdot)$. We first prove guarantees for the $L = 1$ case, where the models are communicated every iteration but the gradient is computed locally instead of globally contrary to previous results (Wang et al., 2018) (thus reducing two communication rounds per iteration). Later, we extend it to the general case of $L > 1$ and show that the updates converge at a sublinear rate in that case.

Theorem 3.2 ($L = 1$ case)**.** *Suppose Assumptions 1-5 hold and the step-size α_t^k satisfies the line-search condition* (5)*. Also, let $0 < \delta < 1$ and $0 < \epsilon, \epsilon_1 < 1/2$ be fixed constants. Moreover, assume that the sample size for each worker satisfies $s \geqslant \frac{4B}{\kappa\epsilon^2}\log\frac{2dK}{\delta}$, where the samples are chosen without replacement. Then, with the LocalNewton updates, $\{\bar{\mathbf{w}}_t\}_{t\geqslant 0}$, from Algorithm 1 and $L = 1$, we obtain*

1. *If $s \gtrsim \frac{\Gamma^2}{\epsilon_1^2 G^2}\log(d/\delta)$ for $G = \min_k \|g^k(\bar{\mathbf{w}}_t)\|$, we get with probability at least $1-6K\delta$,*

$$f(\bar{\mathbf{w}}_{t+1}) - f(\mathbf{w}^*) \leqslant \rho_1(f(\bar{\mathbf{w}}_t) - f(\mathbf{w}^*)).$$

2. *We obtain, (with probability at least $1-6K\delta$)*

$$f(\bar{\mathbf{w}}_{t+1}) - f(\mathbf{w}^*) \leqslant \rho_2(f(\bar{\mathbf{w}}_t) - f(\mathbf{w}^*)) + \eta\cdot\frac{\Gamma}{\kappa(1-\epsilon)},$$

where $\eta = \frac{1}{\sqrt{s}}\Gamma(1+\sqrt{2\log(\frac{1}{\delta})})$.

Here $\rho_i = (1-2\kappa C_i)$, for $i = \{1,2\}$, $C_1 = \frac{(1-\epsilon)\psi}{2} - \frac{\epsilon_1}{\kappa(1-\epsilon)}$, $C_2 = \frac{\psi(1-\epsilon)}{2}$, and $\psi = \frac{\alpha^\star\beta}{M(1+\epsilon)}$.

Proof. The proof is presented in Appendix A. Here, we provide a sketch of the the proof, which takes the following steps.

1. Due to uniform sampling guarantee from Lemma 2.1, strong-convexity and smoothness of the global function $f(\cdot)$ implies that the local function at the k-th worker, $f^k(\cdot)$, also satisfy similar properties. Using this, lower bound $f(\bar{\mathbf{w}}_t) - f(\bar{\mathbf{w}}_{t+1})$ in terms of $\frac{1}{K}\sum_{k=1}^K f_k(\mathbf{w}_t^k) - f_k(\mathbf{w}_{t+1}^k)$.
2. Apply lemma 3.1 (that is, the result for standard Newton step) which says $f_k(\mathbf{w}_t^k) - f_k(\mathbf{w}_{t+1}^k) \geqslant \psi||\mathbf{g}_t^k||^2 \ \forall\ k \in [K]$.
3. Finally, using uniform sketching argument, we show local gradients $g^k(\bar{\mathbf{w}}_t)$ are close to global gradient $g(\bar{\mathbf{w}}_t)$. □

Some remarks regarding the convergence guarantee in Theorem 3.2 are in order.

Remark 1. *The above theorem implies that for $L = 1$, the convergence rate of LocalNewton is linear with high probability. Choosing $\delta = 1/\mathsf{poly}(K)$, we obtain the high probability as $1 - 1/\mathsf{poly}(K)$.*

Remark 2. *Note that we have 2 different settings in the above theorem. Setting 1 implies that provided the local gradients $\{g^k(\bar{\mathbf{w}}_t)\}_{k=1}^K$ are large enough, and the amount of local data s is reasonably large, the convergence is* purely *linear and does not suffer an error floor. This will typically happen in the earlier iterations of LocalNewton. This condition could be restrictive. If this is violated, we move to the next setting.*

Remark 3. *If the gradient condition and the restriction of s are violated, we show that, although the convergence rate of LocalNewton is still linear, the algorithm incurs an error floor. However, the error floor is $\mathcal{O}(1/\sqrt{s})$, and hence quite small for sufficiently large sample-size, s, at each worker.*

Assume all the workers initialize at $\bar{\mathbf{w}}_0$ and run LocalNewton with $L = 1$ for T iterations. Then, from Theorem 3.2, to reach within ξ of the optimal function value (that is, $f(\bar{\mathbf{w}}_T) - f(\bar{\mathbf{w}}^*) \leqslant \xi$), the number of iterations T is upper bounded by

$$T \leqslant \left(\log\frac{1}{\rho}\right)\log\frac{\xi}{f(\bar{\mathbf{w}}_0) - f(\bar{\mathbf{w}}^*)}$$

[3] Note that we introduce $\alpha^\star$ here purely for proving theoretical guarantees. In our experiments, we use the Armijo backtracking line-search rule with $\alpha^\star = 1$ (e.g. see Boyd and Vandenberghe (2004)) to find the right step-size.

w.p. $1-\delta$ for a sample size $s \geqslant \frac{4B}{\kappa\epsilon^2}\log\frac{2dKT}{\delta}$. (Note the increase in sample size s by a factor of T in the $\log(\cdot)$ due to a union bound). The fully synchronized second order method GIANT (Wang et al., 2018) also has similar linear quadratic convergence but it assumes that the gradients are synchronized in every iteration. We remove this assumption by tracking how far the iterate deviates when the gradients are computed locally, thereby cutting the communication costs in half while still showing linear convergence (within some error floor in the most general case). Next, we prove convergence guarantees for the case when $L > 1$.

Theorem 3.3 ($L \geqslant 1$ case)**.** *Suppose Assumptions 1-5 hold and step-size α_t^k solves the line-search condition* (5)*. Also, let $0 < \delta < 1$ and $0 < \epsilon < 1/2$ be fixed constants. Moreover, assume that the sample size for each worker satisfies $s \geqslant \frac{4B}{\kappa\epsilon^2}\log\frac{2dK}{\delta}$, where the samples are chosen without replacement. Then, the LocalNewton updates, $\{\bar{\mathbf{w}}_t\}_{t\geqslant 0}$, from Algorithm 1 and $L \geqslant 1$ with probability at least $1-6LK\delta$ satisfy*

$$f(\bar{\mathbf{w}}_{t+1}) - f(\bar{\mathbf{w}}_{t_0}) \leqslant -C\sum_{\tau=t_0}^{t}\left(\frac{1}{K}\sum_{k=1}^{K}\|\mathbf{g}_\tau^k\|^2\right) + \eta\cdot\frac{L\Gamma}{\kappa(1-\epsilon)},$$

where $\eta = \frac{1}{\sqrt{s}}\Gamma(1+\sqrt{2\log(\frac{1}{\delta})})$, $C = \psi - \frac{(M-\kappa(1-\epsilon)^2)}{2K\kappa^2(1-\epsilon)^2}$. where t_0 is the last iteration where the models were synced, $\psi = \frac{\alpha^\star\beta}{M(1+\epsilon)}$, and $C = \frac{\psi(1-\epsilon)^3}{2}$.

Proof. The general idea of the proof follows the proof for the easier case in Theorem 3.2. See Appendix B for a detailed proof. □

Remark 4. *The theorem shows that LocalNewton with high probability produces a descent direction, provided that the error floor is sufficiently small. Observe that the convergence rate here is no longer linear. In other words, we are trading-off the rate of convergence for local iterations ($L > 1$).*

Remark 5. *Choosing $\delta = 1/\mathsf{poly}(K, L)$, we get that the theorem holds with probability at least $1 - 1/\mathsf{poly}(K, L)$. Note that this is not restrictive since the dependence on δ is logarithmic.*

While the theoretical guarantees for $L > 1$ in Theorem 3.3 are not as strong as those for $L = 1$ in Theorem 3.2 (linear vs sublinear convergence), empirically we observe a fast rate of convergence even when $L > 1$; see Sections 3.1 and 4 for the experiments. Nevertheless, to the best of our knowledge, Theorem 3.3 is the first to show a descent guarantee for a distributed second-order method without synchronizing at every iteration. Obtaining a better rate of convergence for general L, with or without error floor, is an interesting and relevant future research direction. Theoretical results in this section (that is, convergence of LocalNewton to the optimal plus an error term) inspires adaptive LocalNewton, described in detail in the next section.

3.1 GIANT, LocalNewton and Adaptive Synchronization

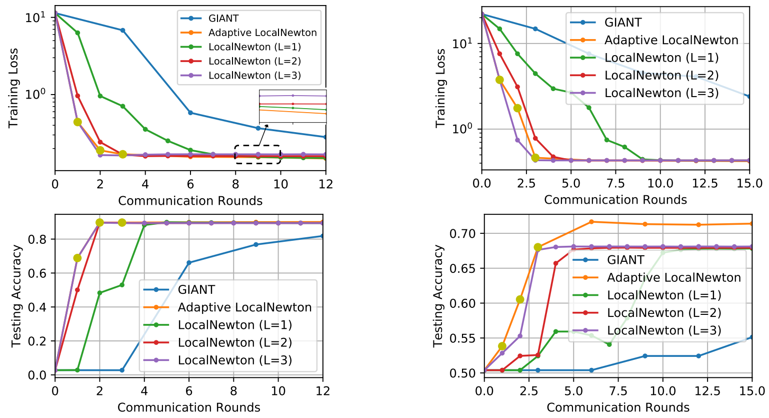
Arguably the most relevant prior work to LocalNewton is the GIANT algorithm (Wang et al., 2018). This approach has compared favorably to other popular distributed second-order methods (e.g. DANE Shamir et al. (2014), AGD Nesterov (2014), LBFGS Liu and Nocedal (1989), CoCoA Smith et al. (2016), DiSCO Zhang and Lin (2015)). GIANT synchronizes the local gradients and the local descent direction in every iteration. Furthermore, it finds the step-size by doing a distributed backtracking line-search requiring an additional round of communication (see Sec. 5.2, Wang et al. (2018)). Finally, the master updates the model by using the average descent direction and the obtained step-size and ships the model to all the workers. Thus, each iteration in GIANT requires three rounds of communication.

In Fig. 2, we compare GIANT to LocalNewton, where LocalNewton is run with 100 workers for $L = 1, 2$ and 3 for three LIBSVM (Chang and Lin, 2011) datasets—w8a, Covtype and EPSILON (see Sec. 4 for further details on datasets).[4] Note that LocalNewton converges much faster w.r.t. communication rounds for all the three datasets since it communicates intermittently, i.e. once every few local second-order iterations (e.g. after 3 local iterations for $L = 3$). Testing accuracy follows the same trends. Further, the quality of the final solution improves as we reduce L. However, we also note that for some datasets (e.g. w8a and EPSILON), it reaches extremely close to the optimal training loss but converges very slowly (or flattens out) after that.

This empirical observation motivates Adaptive LocalNewton: a second-order distributed algorithm that adapts the number of local iterations as the training progresses and ultimately finishes with GIANT. This can be done by monitoring the validation loss at the master, i.e. reduce L if loss stops improving (or switch to GIANT if $L = 1$).[5] For all our experiments, and regardless of datasets, Adaptive LocalNewton proceeds

[4] Additional experiments on two more datasets—a9a and ijcnn1—are provided in Appendix E.1.

[5] To further reduce the communication rounds and dependency on L, each worker can update the model for multiple values of L and send the concatenated model updated to the master. The master can decide the right value of L by evaluating the loss/accuracy at the validation dataset.



(a) w8a dataset (b) Covtype dataset (c) EPSILON dataset

Figure 2: Comparing LocalNewton (for different values of L) and GIANT. In general, LocalNewton reaches very close to the optimal solution. In that region, however, we note that the convergence rate of naive LocalNewton is slow. This is mitigated by using adaptive LocalNewton which appends the LocalNewton iterations with better quality (but more expensive) updates from GIANT.

as follows: It starts with $L = 3$, then decreases L by one after each communication round, and finally (after reaching $L = 1$), it switches to GIANT.

Adaptive LocalNewton is further motivated by theoretical guarantees established in Theorems 3.2 and 3.3 (a convergence of LocalNewton to the optimal solution within an error floor starting with any initial point in $\mathbb{R}^d$) and Theorem 2 in GIANT (Wang et al., 2018) (a linear convergence to the exact optimal solution when the current model is sufficiently close to the optimal model). We see that Adaptive LocalNewton significantly outperforms GIANT in terms of rounds of communication. Yellow dots in Adaptive LocalNewton denote the reduction in the value of L or a switch to GIANT if $L = 1$.

4 EMPIRICAL EVALUATION

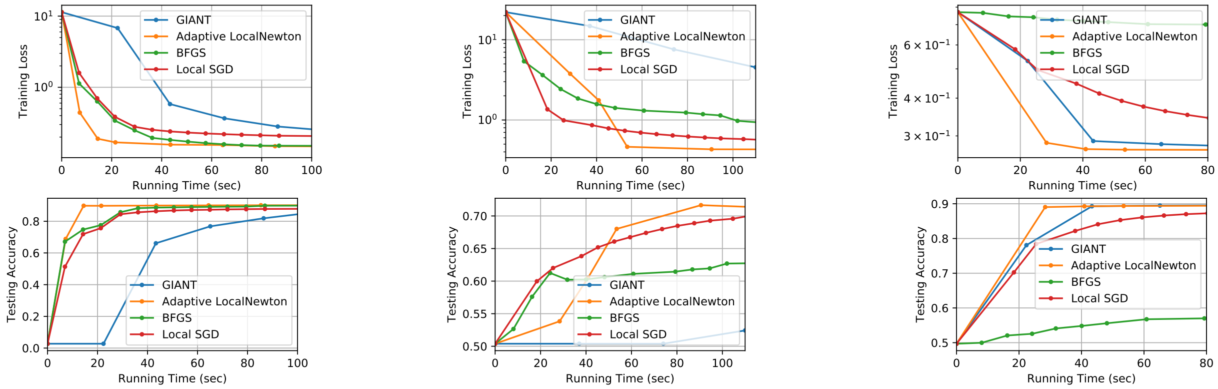
In this section, we present an empirical evaluation of our approach when solving a large-scale logistic regression problem. We ran our experiments on AWS Lambda, a serverless computing platform, using the PyWren (Jonas et al., 2017) framework which uses a high-latency cloud storage (AWS S3) to exchange data with the workers. We ran experiments on the real-world datasets described in Table 1 (obtained from LIBSVM (Chang and Lin, 2011)).

We compare the following distributed optimization schemes for the above datasets:
1. Local SGD (Stich, 2018): The best step-size is chosen for all datasets through hyperparameter search (see Appendix E.2 for details). The workers communicate their models once every epoch, where training on one epoch implies applying SGD (with mini-batch size one) over one pass of the dataset stored locally at the worker.
2. BFGS (Fletcher, 2013): BFGS is a popular quasi-Newton method that estimates an approximate Hessian from the gradient information from previous iterations. The best step-size was obtained through hyperparameter tuning (see Appendix E.2 for details).
3. GIANT (Wang et al., 2018): A state-of-the-art distributed second order algorithm proposed in Wang et al. (2018). The authors show that GIANT outperforms many popular schemes such as DANE, AGD, etc.
4. Adaptive LocalNewton: For all the considered datasets, Adaptive LocalNewton gradually reduces $L = 3$ to $L = 1$ in the first three rounds of communication and then switches to GIANT owing to its better convergence rate when $\bar{\mathbf{w}}_t$ is sufficiently close to $\mathbf{w}^*$.

Dataset	Training samples (n)	Features (d)	Testing samples
w8a	$48,000$	300	$15,000$
Covtype	$500,000$	2916	$81,000$
EPSILON	$400,000$	2000	$100,000$
a9a	$32,000$	123	$16,000$
ijcnn1	$49,000$	22	$91,000$

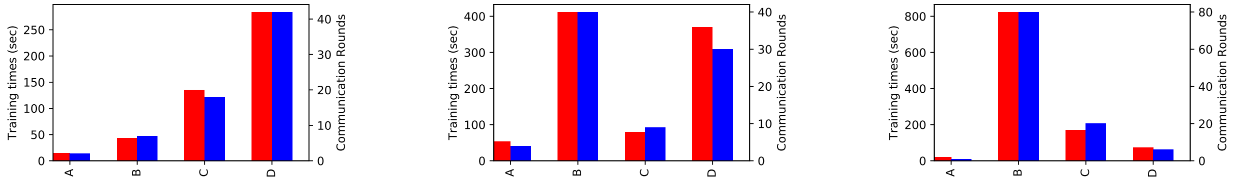
Table 1: Datasets considered for experiments in this paper

For all the experiments presented in this paper, we fixed the number of workers, K, to be 100. Hence, the number of samples per worker, $s = n/100$, for all datasets. The regularization parameter was chosen to be $\gamma = 1/n$. Note that there are several other schemes, such as AGD (Nesterov, 2014), DANE (Shamir et al., 2014), SVRG, etc., that have been proposed in the literature for communication-efficient optimization. How-



(a) w8a dataset (b) Covtype dataset (c) EPSILON dataset

Figure 3: Experiments on the w8a, Covtype and EPSILON datasets on AWS Lambda. Both in terms of training loss and testing accuracy, adaptive LocalNewton converges to the optimal value at least 50% faster than existing schemes.



(a) w8a dataset (b) Covtype dataset (c) EPSILON dataset

Figure 4: Training times (red bars) and communication rounds (blue bars) required to reach the same training loss of $0.19, 0.65$ and 0.3 for the w8a, Covtype and EPSILON datasets, respectively, on AWS Lambda. Here, A: Adaptive LocalNewton, B: BFGS, C: Local SGD, D: GIANT.

ever, most of these schemes have been shown to be outperformed by one of Local SGD, BFGS or GIANT, and hence, we do not perform the comparison again. In Fig. 3, we plot the training loss and testing accuracy for w8a, covtype[6] and EPSILON datasets (see Fig. 2 in Appendix E.3 for experiments on a9a and ijcnn1 datasets). For all the datasets considered, Adaptive LocalNewton significantly outperforms its competitors in terms of time required to reach the same training loss (or testing accuracy).

In Fig. 4, we highlight the fact that runtime savings on AWS Lambda are a direct consequence of significantly fewer rounds of communication. Specifically, to reach the same training loss, we plot the training times and communication rounds as bar plots for three datasets, and note that savings in communication rounds result in commensurate savings on end-to-end runtimes on AWS Lambda. In Fig. 3 in Appendix E.3, we provide detailed plots for training loss and testing accuracies w.r.t. communication rounds for all the five datasets.

5 CONCLUSION

The practicality of second-order methods has been limited due to large compute and storage power required to work with the Hessian. However, in the last few decades, trends such as Moore's law have made computation faster and memory cheaper, whereas the improvements communication costs were at best marginal. These trends, combined with a flurry of efficient but approximate algorithms (e.g., see Pilanci and Wainwright (2017); Roosta-Khorasani and Mahoney (2016a,b); Gupta et al. (2019)) have made second-order methods popular, further bolstered by their agnosticism to hyperparameter tuning. Even in deep neural networks with extremely large model sizes, second-order information has found a role to play (e.g., see Dong et al. (2019); Yao et al. (2019, 2018)). In this paper, we identify and concretize the role that second-order methods – combined with local optimization algorithms – can play in reducing the communication costs during distributed training. We believe that our method will play a significant role in motivating and designing next-generation communication-efficient algorithms for fast distributed training of machine learning models.

[6] The covtype dataset has $d = 54$ features and it does not perform well with logistic regression. Hence, we apply polynomial feature extension (using pairwise products) to increase the number of features to $d^2 = 2916$.

References

Acharya, J., De Sa, C., Foster, D., and Sridharan, K. (2019). Distributed learning with sublinear communication. In *International Conference on Machine Learning*, pages 40–50.

Baldini, I., Castro, P. C., Chang, K. S.-P., Cheng, P., Fink, S. J., Ishakian, V., Mitchell, N., Muthusamy, V., Rabbah, R. M., Slominski, A., and Suter, P. (2017). Serverless computing: Current trends and open problems. *CoRR*, abs/1706.03178.

Bernstein, J., Wang, Y.-X., Azizzadenesheli, K., and Anandkumar, A. (2018). signsgd: Compressed optimisation for non-convex problems. In *International Conference on Machine Learning*, pages 560–569.

Boyd, S. and Vandenberghe, L. (2004). *Convex Optimization.* Cambridge University Press, New York, NY, USA.

Carreira, J., Fonseca, P., Tumanov, A., Zhang, A., and Katz, R. (2019). Cirrus: a serverless framework for end-to-end ml workflows. In *Proceedings of the ACM Symposium on Cloud Computing*, pages 13–24.

Chang, C.-C. and Lin, C.-J. (2011). Libsvm: a library for support vector machines. *ACM transactions on intelligent systems and technology (TIST)*, 2(3):27.

Demmel, J. (2013). Communication-avoiding algorithms for linear algebra and beyond. In *2013 IEEE 27th Int. Sym. on Parallel and Distributed Processing*, pages 585–585.

Dereziński, M., Bartan, B., Pilanci, M., and Mahoney, M. W. (2020). Debiasing distributed second order optimization with surrogate sketching and scaled regularization. *arXiv preprint arXiv:2007.01327.*

Dereziński, M. and Mahoney, M. W. (2019). Distributed estimation of the inverse hessian by determinantal averaging. In *Advances in Neural Information Processing Systems 32*, pages 11405–11415. Curran Associates, Inc.

Dieuleveut, A. and Patel, K. K. (2019). Communication trade-offs for local-sgd with large step size. In *Advances in Neural Information Processing Systems*, pages 13579–13590.

Dong, Z., Yao, Z., Gholami, A., Mahoney, M. W., and Keutzer, K. (2019). Hawq: Hessian aware quantization of neural networks with mixed-precision. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 293–302.

Feng, L., Kudva, P., Da Silva, D., and Hu, J. (2018). Exploring serverless computing for neural network training. In *2018 IEEE 11th International Conference on Cloud Computing (CLOUD)*, pages 334–341. IEEE.

Fletcher, R. (2013). *Practical methods of optimization.* John Wiley & Sons.

Gandikota, V., Maity, R. K., and Mazumdar, A. (2019). vqsgd: Vector quantized stochastic gradient descent. *arXiv preprint arXiv:1911.07971.*

Ghosh, A., Maity, R. K., Kadhe, S., Mazumdar, A., and Ramachandran, K. (2020). Communication efficient and byzantine tolerant distributed learning. In *2020 IEEE International Symposium on Information Theory (ISIT)*, pages 2545–2550.

Gupta, V., Carrano, D., Yang, Y., Shankar, V., Courtade, T., and Ramchandran, K. (2020a). Serverless straggler mitigation using local error-correcting codes. *arXiv preprint arXiv:2001.07490.*

Gupta, V., Kadhe, S., Courtade, T., Mahoney, M. W., and Ramchandran, K. (2019). Oversketched newton: Fast convex optimization for serverless systems. *arXiv preprint arXiv:1903.08857.*

Gupta, V., Serrano, S. A., and DeCoste, D. (2020b). Stochastic weight averaging in parallel: Large-batch training that generalizes well. In *International Conference on Learning Representations.*

Gupta, V., Wang, S., Courtade, T., and Ramchandran, K. (2018). Oversketch: Approximate matrix multiplication for the cloud. *IEEE International Conference on Big Data, Seattle, WA, USA.*

Haddadpour, F., Kamani, M. M., Mahdavi, M., and Cadambe, V. (2019). Local sgd with periodic averaging: Tighter analysis and adaptive synchronization. In *Advances in Neural Information Processing Systems*, pages 11080–11092.

Hellerstein, J. M., Faleiro, J., Gonzalez, J. E., Schleier-Smith, J., Sreekanti, V., Tumanov, A., and Wu, C. (2018). Serverless computing: One step forward, two steps back. *arXiv preprint arXiv:1812.03651.*

Ivkin, N., Rothchild, D., Ullah, E., Stoica, I., Arora, R., et al. (2019). Communication-efficient distributed sgd with sketching. In *Advances in Neural Information Processing Systems*, pages 13144–13154.

Jonas, E., Pu, Q., Venkataraman, S., Stoica, I., and Recht, B. (2017). Occupy the cloud: distributed computing for the 99%. In *Proceedings of the 2017 Symposium on Cloud Computing*, pages 445–451. ACM.

Jonas, E., Schleier-Smith, J., Sreekanti, V., Tsai, C.-C., Khandelwal, A., Pu, Q., Shankar, V., Carreira, J., Krauth, K., Yadwadkar, N., et al. (2019). Cloud programming simplified: A berkeley view on serverless computing. *arXiv preprint arXiv:1902.03383.*

Kairouz, P., McMahan, H. B., Avent, B., Bellet, A., Bennis, M., Bhagoji, A. N., Bonawitz, K., Charles, Z., Cormode, G., Cummings, R., et al. (2019). Advances and open problems in federated learning. *arXiv preprint arXiv:1912.04977.*

Konečnỳ, J., McMahan, H. B., Yu, F. X., Richtárik, P., Suresh, A. T., and Bacon, D. (2016). Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492.*

Lin, T., Stich, S. U., Patel, K. K., and Jaggi, M. (2020). Don't use large mini-batches, use local sgd. In *International Conference on Learning Representations.*

Lin, Y., Han, S., Mao, H., Wang, Y., and Dally, W. J. (2017). Deep gradient compression: Reducing the communication bandwidth for distributed training. *arXiv preprint arXiv:1712.01887.*

Liu, D. C. and Nocedal, J. (1989). On the limited memory bfgs method for large scale optimization. *Mathematical programming*, 45(1-3):503–528.

Mayekar, P. and Tyagi, H. (2020). Limits on Gradient Compression for Stochastic Optimization. *arXiv e-prints*, page arXiv:2001.09032.

McMahan, B., Moore, E., Ramage, D., Hampson, S., and y Arcas, B. A. (2017). Communication-Efficient Learning of Deep Networks from Decentralized Data. In Singh, A. and Zhu, J., editors, *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, pages 1273–1282, Fort Lauderdale, FL, USA. PMLR.

Nesterov, Y. (2014). *Introductory Lectures on Convex Optimization: A Basic Course*. Springer Publishing Company, Incorporated, 1 edition.

Pilanci, M. and Wainwright, M. J. (2017). Newton sketch: A near linear-time optimization algorithm with linear-quadratic convergence. *SIAM Jour. on Opt.*, 27:205–245.

Reddi, S. J., Hefny, A., Sra, S., Pöczos, B., and Smola, A. (2015). On variance reduction in stochastic gradient descent and its asynchronous variants. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'15, pages 2647–2655, Cambridge, MA, USA. MIT Press.

Roosta-Khorasani, F. and Mahoney, M. W. (2016a). Sub-Sampled Newton Methods I: Globally Convergent Algorithms. *arXiv e-prints*, page arXiv:1601.04737.

Roosta-Khorasani, F. and Mahoney, M. W. (2016b). Sub-Sampled Newton Methods II: Local Convergence Rates. *arXiv e-prints*, page arXiv:1601.04738.

Shamir, O., Srebro, N., and Zhang, T. (2014). Communication-efficient distributed optimization using an approximate Newton-type method. In *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*, ICML'14, pages II–1000–II–1008. JMLR.org.

Shankar, V., Krauth, K., Pu, Q., Jonas, E., Venkataraman, S., Stoica, I., Recht, B., and Ragan-Kelley, J. (2018). numpywren: serverless linear algebra. *ArXiv e-prints.*

Shewchuk, J. R. et al. (1994). An introduction to the conjugate gradient method without the agonizing pain.

Smith, V., Forte, S., Ma, C., Takác, M., Jordan, M. I., and Jaggi, M. (2016). Cocoa: A general framework for communication-efficient distributed optimization. *arXiv preprint arXiv:1611.02189.*

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

Stich, S. U., Cordonnier, J.-B., and Jaggi, M. (2018). Sparsified sgd with memory. In *Advances in Neural Information Processing Systems*, pages 4447–4458.

Wang, H., Niu, D., and Li, B. (2019). Distributed machine learning with a serverless architecture. In *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*, pages 1288–1296. IEEE.

Wang, S., Roosta-Khorasani, F., Xu, P., and Mahoney, M. W. (2018). Giant: Globally improved approximate newton method for distributed optimization. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R., editors, *Advances in Neural Information Processing Systems 31*, pages 2332–2342. Curran Associates, Inc.

Yao, Z., Gholami, A., Keutzer, K., and Mahoney, M. (2018). Large batch size training of neural networks with adversarial training and second-order information. *arXiv preprint arXiv:1810.01021.*

Yao, Z., Gholami, A., Keutzer, K., and Mahoney, M. (2019). Pyhessian: Neural networks through the lens of the hessian. *arXiv preprint arXiv:1912.07145.*

Zhang, Y. and Lin, X. (2015). Disco: Distributed optimization for self-concordant empirical loss. In Bach, F. and Blei, D., editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 362–370, Lille, France. PMLR.

A Proof of Theorem 3.2

The proofs for theorems in this paper utilize the auxiliary lemmas in Appendix C.

Proof. The proof of the theorem is based on the following two high probability lower bounds:

Case 1:

$$f(\bar{\mathbf{w}}_t) - f(\bar{\mathbf{w}}_{t+1}) \geqslant C\|\mathbf{g}(\bar{\mathbf{w}}_t)\|^2, \tag{7}$$

where $C = \frac{\alpha^\star \beta(1-\epsilon)}{2M(1+\epsilon)}$ is a constant, and

Case 2

$$f(\bar{\mathbf{w}}_t) - f(\bar{\mathbf{w}}_{t+1}) \geqslant C_1\|\mathbf{g}(\bar{\mathbf{w}}_t)\|^2 - \frac{\eta\Gamma}{\kappa(1-\epsilon)}, \tag{8}$$

where C_1 is a constant (> 0) and $\eta = (1 + \sqrt{2\log(\frac{1}{\delta})})\sqrt{\frac{1}{s}}\Gamma$.

We will prove the above result shortly, but let us complete the proof of the theorem assuming that Eq. (7) and Eq. (8) are true.

Case 1 (using Eq. (7)**)** Invoking the κ strong convexity of the the function f we have

$$f(\bar{\mathbf{w}}_t) - f(\mathbf{w}^*) \leqslant \frac{1}{2\kappa}\|\mathbf{g}(\bar{\mathbf{w}}_t)\|^2, \tag{9}$$

where $\bar{\mathbf{w}}^*$ is the unique global minimizer of the function f. Combining the last lower bound with equation (7) we obtain

$$f(\bar{\mathbf{w}}_{t+1}) - f(\bar{\mathbf{w}}_t) \leqslant (1 - 2\kappa C)(f(\bar{\mathbf{w}}_t) - f(\mathbf{w}^*)), \tag{10}$$

with probability $1 - \delta$. Also note that

$$1 > 1 - 2\kappa C = 1 - \frac{\kappa\alpha^\star\beta(1-\epsilon)}{M(1+\epsilon)} > 0,$$

where the last inequality uses the definition of $\alpha^\star$ from Eq. (6). The completes the proof of Theorem 3.2.

Case 2 (using Eq. (8)**)** Using the same steps as before, and using the condition of Eq. (8), we obtain Theorem 3.2.

It remains to prove the claim (7) and (8).

Proof of the claim (7)**:** Recall that for $L = 1$, we have

$$\mathbf{w}_{t+1}^k = \bar{\mathbf{w}}_t - \alpha_t^k \mathbf{p}_t^k, \quad \text{and} \quad \bar{\mathbf{w}}_{t+1} := \frac{1}{K}\sum_{k=1}^K \mathbf{w}_{t+1}^k = \bar{\mathbf{w}}_t - \frac{1}{K}\sum_{k=1}^K \alpha_t^k \mathbf{p}_t^k,$$

where the $\mathbf{p}_t^k = (\mathbf{H}_t^k)^{-1}\mathbf{g}_t^k, \mathbf{H}_t^k = (\mathbf{H}^k)^{-1}(\bar{\mathbf{w}}_t)$ and $\mathbf{g}_t^k = \mathbf{g}^k(\bar{\mathbf{w}}_t)$. Invoking the M-smoothness of the function $f(\cdot)$ we have

$$\begin{aligned}
f(\bar{\mathbf{w}}_t) - f(\bar{\mathbf{w}}_{t+1}) &\geqslant \frac{-M}{2K^2}\|\bar{\mathbf{w}}_t - \bar{\mathbf{w}}_{t+1}\|^2 + \langle \mathbf{g}(\bar{\mathbf{w}}_t), \bar{\mathbf{w}}_t - \bar{\mathbf{w}}_{t+1}\rangle \\
&\geqslant \frac{-M}{2K^2}\left\|\sum_{k=1}^K (\alpha_t^k)\mathbf{p}_t^k\right\|^2 + \langle \mathbf{g}(\bar{\mathbf{w}}_t), \frac{1}{K}\sum_{k=1}^K \alpha_t^k \mathbf{p}_t^k\rangle \\
&\overset{(i)}{\geqslant} \frac{-M}{2K}\sum_{k=1}^K (\alpha_t^k)^2\|\mathbf{p}_t^k\|^2 + \langle \mathbf{g}(\bar{\mathbf{w}}_t), \frac{1}{K}\sum_{k=1}^K \alpha_t^k \mathbf{p}_t^k\rangle \\
&= \frac{1}{K}\sum_{k=1}^K \left(\alpha_t^k(\mathbf{p}_t^k)^T\mathbf{g}(\bar{\mathbf{w}}_t) - \frac{M}{2}(\alpha_t^k)^2\|\mathbf{p}_t^k\|^2\right)
\end{aligned} \tag{11}$$

where the inequality (i) uses the following fact

$$\left\| \frac{1}{K} \sum_{k=1}^{K} \mathbf{a}^k \right\|^2 \leqslant \frac{1}{K} \sum_{k=1}^{K} \|\mathbf{a}^k\|^2, \tag{12}$$

for all vectors $\mathbf{a}^1, \mathbf{a}^2, \cdots, \mathbf{a}^K \in \mathbb{R}^d$.

We now complete the proof by using the following bound on the first term in Eq. (11). In particular, In the first case, we show that, for all $k \in [K]$ provided

$$s \gtrsim \left(\frac{\Gamma^2}{\epsilon_1^2 G^2} \log(d/\delta) \right),$$

and $\|g^k(\bar{\mathbf{w}}_t)\| \geqslant G$, where $\epsilon_1 > 0$ (small number), we have

$$\alpha_t^k (\mathbf{p}_t^k)^T \mathbf{g}(\bar{\mathbf{w}}_t) \geqslant \left(\psi - \frac{\epsilon_1}{\kappa(1-\epsilon)} \right) \|\mathbf{g}_t^k\|^2 + \frac{\kappa(1-\epsilon)(\alpha_t^k)^2}{2} \|\mathbf{p}_t^k\|^2 \tag{13}$$

with probability at least $1 - 4\delta$.

Let us substitute Eq. (13) in equation (11), we get

$$\begin{aligned} f(\bar{\mathbf{w}}_t) - f(\bar{\mathbf{w}}_{t+1}) &\geqslant \frac{1}{K} \sum_{k=1}^{K} \left[\left(\psi - \frac{\epsilon_1}{\kappa(1-\epsilon)} \right) \|\mathbf{g}_t^k\|^2 - \frac{(M - \kappa(1-\epsilon))(\alpha_t^k)^2}{2} \|\mathbf{p}_t^k\|^2 \right] \\ &\geqslant \frac{1}{K} \sum_{k=1}^{K} \left[\left(\psi - \frac{\epsilon_1}{\kappa(1-\epsilon)} \right) \|\mathbf{g}_t^k\|^2 - \frac{(M - \kappa(1-\epsilon))(\alpha_t^k)^2}{2\kappa^2(1-\epsilon)^2} \|\mathbf{g}_t^k\|^2 \right] \end{aligned} \tag{14}$$

where the last inequality follows from the fact that the function f_k is $\kappa(1-\epsilon)$ strongly convex with probability $1-\delta$, and thus

$$\|\mathbf{p}_t^k\|^2 := \|(\mathbf{H}_t^k)^{-1} \mathbf{g}_t^k\|_2^2 \leqslant \|(\mathbf{H}_t^k)^{-1}\|_2^2 \|\mathbf{g}_t^k\|^2 \leqslant \frac{1}{\kappa^2(1-\epsilon)^2} \|\mathbf{g}_t^k\|^2. \tag{15}$$

with probability $1-\delta$. Now, using the upper bound on α_t^k, we have

$$\begin{aligned} f(\bar{\mathbf{w}}_t) - f(\bar{\mathbf{w}}_{t+1}) &\geqslant \frac{1}{K} \sum_{k=1}^{K} \left[\left(\psi - \frac{\epsilon_1}{\kappa(1-\epsilon)} \right) \|\mathbf{g}_t^k\|^2 - \frac{(M - \kappa(1-\epsilon)^2)}{2} \frac{\alpha^{\star 2}}{\kappa^2(1-\epsilon)^2} \|\mathbf{g}_t^k\|^2 \right] \\ &= \left(\psi - \frac{\epsilon_1}{\kappa(1-\epsilon)} - \frac{(M - \kappa(1-\epsilon)^2)}{2} \frac{\alpha^{\star 2}}{\kappa^2(1-\epsilon)^2} \right) \frac{1}{K} \sum_{k=1}^{K} \|\mathbf{g}_t^k\|^2 \\ &\geqslant C \frac{1}{K} \sum_{k=1}^{K} \|\mathbf{g}_t^k\|^2, \end{aligned} \tag{16}$$

with probability exceeding $1 - 6\delta$, where $C = \frac{(1-\epsilon)\psi}{2} - \frac{\epsilon_1}{\kappa(1-\epsilon)}$, and the last bound follows by substituting the value of α^* from equation (6) and using the fact that $0 < \epsilon < 1/2$. Moreover, using Eq. (12), we get

$$\|\mathbf{g}(\cdot)\|^2 \leqslant \frac{1}{K} \sum_{k=1}^{K} \|\mathbf{g}^k(\cdot)\|^2,$$

which prove Eq. (7).

It now remains to prove bound (13)

Proof of bound (13)**:** From the uniform subsampling property (similar to Lemma C.1, see Appendix D.2), we get

$$|(\mathbf{p}_t^k)^T\mathbf{g}(\bar{\mathbf{w}}_t) - (\mathbf{p}_t^k)^T\mathbf{g}^k(\bar{\mathbf{w}}_t)| \leqslant \epsilon_1\|(\mathbf{p}_t^k)\|\|\mathbf{g}_k(\bar{\mathbf{w}}_t)\| \text{ w.p. } 1-\delta. \tag{17}$$

Thus,

$$(\mathbf{p}_t^k)^T\mathbf{g}(\bar{\mathbf{w}}_t) \geqslant (\mathbf{p}_t^k)^T\mathbf{g}_k(\bar{\mathbf{w}}_t) - \epsilon_1\|(\mathbf{p}_t^k)\|\|\mathbf{g}_k(\bar{\mathbf{w}}_t)\| \tag{18}$$

w.p. $1-\delta$. Now, since the function f_k is $\kappa(1-\epsilon)$ strongly-convexity with probability $1-\delta$, we have the following bound w.p. at least $1-\delta$:

$$\alpha_t^k(\mathbf{p}_t^k)^T\mathbf{g}_t^k \geqslant (f_k(\bar{\mathbf{w}}_t) - f_k(\mathbf{w}_{t+1}^k)) + \frac{\kappa(1-\epsilon)}{2}(\alpha_t^k)^2\|\mathbf{p}_t^k\|^2 \tag{19}$$

Combing the equations (18)-(19) and using Lemma C.2 we have

$$\begin{aligned}
\alpha_t^k(\mathbf{p}_t^k)^T\mathbf{g}(\bar{\mathbf{w}}_t) &\geqslant (f_k(\bar{\mathbf{w}}_t) - f_k(\mathbf{w}_{t+1}^k)) + \frac{\kappa(1-\epsilon)}{2}(\alpha_t^k)^2\|\mathbf{p}_t^k\|^2 - \epsilon_1\|(\mathbf{p}_t^k)\|\|\mathbf{g}_k(\bar{\mathbf{w}}_t)\| \\
&\overset{(i)}{\geqslant} \psi\|\mathbf{g}_t^k\|^2 + \frac{\kappa(1-\epsilon)}{2}(\alpha_t^k)^2\|\mathbf{p}_t^k\|^2 - \epsilon_1\|(\mathbf{p}_t^k)\|\|\mathbf{g}_k(\bar{\mathbf{w}}_t)\| \\
&\overset{(ii)}{\geqslant} \psi\|\mathbf{g}_t^k\|^2 + \frac{\kappa(1-\epsilon)(\alpha_t^k)^2}{2}\|\mathbf{p}_t^k\|^2 - \frac{\epsilon_1}{\kappa(1-\epsilon)}\|\mathbf{g}_k(\bar{\mathbf{w}}_t)\|^2 \\
&= \left(\psi - \frac{\epsilon_1}{\kappa(1-\epsilon)}\right)\|\mathbf{g}_t^k\|^2 + \frac{\kappa(1-\epsilon)(\alpha_t^k)^2}{2}\|\mathbf{p}_t^k\|^2
\end{aligned}$$

with probability exceeding $1-4\delta$, where the inequality (i) follows from Lemma C.2 and inequality (ii) follows from (15).

Note that the bound in (13) hold for all $k \in [K]$ with probability $1-\delta_1$ (thus, the sample size increases by a factor of K in the $\log(\cdot)$ term). This concludes the Case 1 of our proof. We now move to Case 2.

Proof of the claim (8)**:** We now continue with the same analysis and show the following

$$f(\bar{\mathbf{w}}_t) - f(\bar{\mathbf{w}}_{t+1}) \geqslant C_1\frac{1}{K}\sum_{k=1}^{K}\|\mathbf{g}_t^k\|^2 - \frac{\eta\Gamma}{\kappa(1-\epsilon)}, \tag{20}$$

with probability at least $1-4\delta$.

In this case, we show that the requirement of a lower bound on $\|g^k(\bar{\mathbf{w}}_t)\|$ and s can be relaxed at the expense of getting hit by an error floor. In particular, we show that

$$\alpha_t^k(\mathbf{p}_t^k)^T\mathbf{g}(\bar{\mathbf{w}}_t) \geqslant \psi\|\mathbf{g}_t^k\|^2 + \frac{\kappa(1-\epsilon)(\alpha_t^k)^2}{2}\|\mathbf{p}_t^k\|^2 - \frac{\eta\Gamma}{\kappa(1-\epsilon)} \tag{21}$$

with probability at least $1-4\delta$, where $\eta = (1+\sqrt{2\log(\frac{1}{\delta})})\sqrt{\frac{1}{s}}\Gamma$. Substituting this yields the bound of Eq. (20).

Proof of bound Eq. (21) **:** From the uniform subsampling property (see Appendix D.1), we get

$$|(\mathbf{p}_t^k)^T\mathbf{g}(\bar{\mathbf{w}}_t) - (\mathbf{p}_t^k)^T\mathbf{g}^k(\bar{\mathbf{w}}_t)| \leqslant \eta\|(\mathbf{p}_t^k)\| \text{ w.p. } 1-\delta. \tag{22}$$

where $\eta = (1+\sqrt{2\log(\frac{1}{\delta})})\sqrt{\frac{1}{s}}\Gamma$. Thus,

$$(\mathbf{p}_t^k)^T\mathbf{g}(\bar{\mathbf{w}}_t) \geqslant (\mathbf{p}_t^k)^T\mathbf{g}_k(\bar{\mathbf{w}}_t) - \eta\|(\mathbf{p}_t^k)\| \tag{23}$$

w.p. $1-\delta$. Now, since the function f_k is $\kappa(1-\epsilon)$ strongly-convexity with probability $1-\delta$, we have the following bound w.p. at least $1-\delta$:

$$\alpha_t^k(\mathbf{p}_t^k)^T\mathbf{g}_t^k \geqslant (f_k(\bar{\mathbf{w}}_t) - f_k(\mathbf{w}_{t+1}^k)) + \frac{\kappa(1-\epsilon)}{2}(\alpha_t^k)^2\|\mathbf{p}_t^k\|^2 \tag{24}$$

Combing the equations (23)-(24) and using Lemma C.2 we have

$$\begin{aligned}\alpha_t^k(\mathbf{p}_t^k)^T\mathbf{g}(\bar{\mathbf{w}}_t) &\geqslant (f_k(\bar{\mathbf{w}}_t)-f_k(\mathbf{w}_{t+1}^k))+\frac{\kappa(1-\epsilon)}{2}(\alpha_t^k)^2\|\mathbf{p}_t^k\|^2-\eta\|(\mathbf{p}_t^k)\| \\ &\overset{(i)}{\geqslant} \psi\|\mathbf{g}_t^k\|^2+\frac{\kappa(1-\epsilon)}{2}(\alpha_t^k)^2\|\mathbf{p}_t^k\|^2-\eta\|(\mathbf{p}_t^k)\| \\ &\overset{(ii)}{\geqslant} \psi\|\mathbf{g}_t^k\|^2+\frac{\kappa(1-\epsilon)(\alpha_t^k)^2}{2}\|\mathbf{p}_t^k\|^2-\frac{\eta}{\kappa(1-\epsilon)}\Gamma\end{aligned}$$

with probability exceeding $1-4\delta$, where the inequality (i) follows from Lemma C.2 and inequality (ii) follows from (15) and the fact that $\|g^k(\bar{\mathbf{w}}_t)\| \leqslant \Gamma$.

□

B Proof of Theorem 3.3

Proof. Recall from perturbed iterate analysis

$$\bar{\mathbf{w}}_{t+1}=\bar{\mathbf{w}}_{t_0}-\sum_{\tau=t_0}^{t}\bar{\mathbf{p}}_\tau, \tag{25}$$

where $\bar{\mathbf{p}}_\tau=\frac{1}{K}\sum_{k=1}^K\alpha_\tau^k\mathbf{p}_\tau^k$ is the average descent direction and $\mathbf{p}_\tau^k=(\mathbf{H}_\tau^k)^{-1}\mathbf{g}_\tau^k$ is the local descent direction at the k-th worker at time τ.

Similar to the proof of theorem 3.2, we next invoke the M-smoothness property of $f(\cdot)$ to get

$$\begin{aligned}f(\bar{\mathbf{w}}_{t_0})-f(\bar{\mathbf{w}}_{t+1}) &\geqslant \frac{-M}{2}\|\sum_{\tau=t_0}^{t}\bar{\mathbf{p}}_\tau\|^2+\langle\mathbf{g}(\bar{\mathbf{w}}_{t_0}),\sum_{\tau=t_0}^{t}\bar{\mathbf{p}}_\tau\rangle \\ &= \frac{-M}{2}\|\frac{1}{K}\sum_{k=1}^{K}\sum_{\tau=t_0}^{t}\alpha_\tau^k\mathbf{p}_\tau^k\|^2+\frac{1}{K}\sum_{k=1}^{K}\sum_{\tau=t_0}^{t}\langle\mathbf{g}(\bar{\mathbf{w}}_{t_0}),\alpha_\tau^k\mathbf{p}_\tau^k\rangle \\ &\geqslant \frac{-M}{2K}\sum_{k=1}^{K}\|\sum_{\tau=t_0}^{t}\alpha_\tau^k\mathbf{p}_\tau^k\|^2+\frac{1}{K}\sum_{k=1}^{K}\sum_{\tau=t_0}^{t}\langle\mathbf{g}(\bar{\mathbf{w}}_{t_0}),\alpha_\tau^k\mathbf{p}_\tau^k\rangle,\end{aligned} \tag{26}$$

where the last inequality uses the fact

$$\left(\sum_{k=1}^{K}\|\mathbf{a}_k\|\right)^2 \leqslant K\sum_{k=1}^{K}\|\mathbf{a}_k\|^2,\ \forall\ \mathbf{a}_k\in\mathbb{R}^d, k\in[K]. \tag{27}$$

Similarly, by $\kappa(1-\epsilon)$ strong-convexity of $f_k(\cdot)$, we get

$$f_k(\mathbf{w}_{t_0}^k)-f_k(\mathbf{w}_{t+1}^k) \leqslant \frac{-\kappa(1-\epsilon)}{2}\|\sum_{\tau=t_0}^{t}\alpha_\tau^k\mathbf{p}_t^k\|^2+\langle\mathbf{g}_t^k,\sum_{\tau=t_0}^{t}\alpha_\tau^k\mathbf{p}_\tau^k\rangle, \tag{28}$$

with probability $1-\delta$. The above inequality, when averaged across k, becomes

$$\frac{1}{K}\sum_{k=1}^{K}\big(f_k(\mathbf{w}_{t_0}^k)-f_k(\mathbf{w}_{t+1}^k)\big) \leqslant \frac{-\kappa(1-\epsilon)}{2K}\sum_{k=1}^{K}\|\sum_{\tau=t_0}^{t}\alpha_\tau^k\mathbf{p}_t^k\|^2+\frac{1}{K}\sum_{k=1}^{K}\sum_{\tau=t_0}^{t}\langle\mathbf{g}_t^k,\alpha_\tau^k\mathbf{p}_\tau^k\rangle \tag{29}$$

Moreover, similar to Eq. (22), we get

$$|\mathbf{r}^T\mathbf{g}(\bar{\mathbf{w}}_t)-\mathbf{r}^T\mathbf{g}^k(\bar{\mathbf{w}}_t)| \leqslant \eta\|\mathbf{r}\| \text{ w.p. } 1-\delta. \tag{30}$$

where $\eta=(1+\sqrt{2\log(\frac{m}{\delta})})\sqrt{\frac{1}{s}}\Gamma$. Keeping $\mathbf{r}=\alpha_\tau^k\mathbf{p}_\tau^k$ and $\mathbf{w}=\bar{\mathbf{w}}_{t_0}$, we get

$$(\alpha_\tau^k\mathbf{p}_\tau^k)^T\mathbf{g}(\bar{\mathbf{w}}_{t_0}) \geqslant (\alpha_\tau^k\mathbf{p}_\tau^k)^T\mathbf{g}^k(\bar{\mathbf{w}}_{t_0})-\eta\alpha_\tau^k\|\mathbf{p}_\tau^k\|, \tag{31}$$

w. p. $1-\delta$, where $\eta = (1+\sqrt{2\log(\frac{m}{\delta})})\sqrt{\frac{1}{s}}\Gamma$.

Now, after combining inequalities (26) and (29) using (31) to eliminate the terms $\frac{1}{K}\sum_{k=1}^{K}\sum_{\tau=t_0}^{t}\langle \mathbf{g}(\bar{\mathbf{w}}_{t_0}), \alpha_\tau^k \mathbf{p}_\tau^k\rangle$ and $\frac{1}{K}\sum_{k=1}^{K}\sum_{\tau=t_0}^{t}\langle \mathbf{g}^k(\bar{\mathbf{w}}_{t_0}), \alpha_\tau^k \mathbf{p}_\tau^k\rangle$, we get

$$f(\bar{\mathbf{w}}_{t_0}) - f(\bar{\mathbf{w}}_{t+1}) \geqslant \frac{1}{K}\sum_{k=1}^{K}(f_k(\bar{\mathbf{w}}_{t_0}^k) - f_k(\bar{\mathbf{w}}_{t+1}^k) - \frac{(M-\kappa(1-\epsilon))}{2K}\sum_{k=1}^{K}(\|\sum_{\tau=t_0}^{t}\alpha_\tau^k \mathbf{p}_t^k\|^2) - \frac{1}{K}\sum_{k=1}^{K}\sum_{\tau=t_0}^{t}\eta\alpha_\tau^k\|\mathbf{p}_\tau^k\|. \tag{32}$$

Also, from Lemma C.2, we have

$$f_k(\bar{\mathbf{w}}_{t_0}^k) - f_k(\bar{\mathbf{w}}_{t+1}^k) \geqslant \psi\sum_{\tau=t_0}^{t}\|\mathbf{g}_\tau^k\|^2. \tag{33}$$

Using above, we get

$$f(\bar{\mathbf{w}}_{t_0} - f(\bar{\mathbf{w}}_{t+1}) \geqslant \frac{1}{K}\psi\sum_{k=1}^{K}\sum_{\tau=t_0}^{t}\|\mathbf{g}_\tau^k\|^2 - \frac{(M-\kappa(1-\epsilon))}{2K}\sum_{k=1}^{K}(\|\sum_{\tau=t_0}^{t}\alpha_\tau^k \mathbf{p}_t^k\|^2) - \frac{1}{K}\sum_{k=1}^{K}\sum_{\tau=t_0}^{t}\eta\alpha_\tau^k\|\mathbf{p}_\tau^k\|. \tag{34}$$

Using triangle inequality above, we get

$$f(\bar{\mathbf{w}}_{t_0}) - f(\bar{\mathbf{w}}_{t+1}) \geqslant \frac{1}{K}\psi\sum_{k=1}^{K}\sum_{\tau=t_0}^{t}\|\mathbf{g}_\tau^k\|^2 - \frac{(M-\kappa(1-\epsilon))}{2K}\sum_{k=1}^{K}\sum_{\tau=t_0}^{t}(\alpha_\tau^k)^2\|\mathbf{p}_\tau^k\|^2 - \frac{1}{K}\sum_{k=1}^{K}\sum_{\tau=t_0}^{t}\eta\alpha_\tau^k\|\mathbf{p}_\tau^k\|. \tag{35}$$

Also, since $\alpha_t^k \leqslant 1$ and $\|\mathbf{p}_\tau^k\| \leqslant \frac{1}{\kappa(1-\epsilon)}\|\mathbf{g}_\tau^k\|$, we get

$$f(\bar{\mathbf{w}}_{t_0} - f(\bar{\mathbf{w}}_{t+1}) \geqslant \frac{1}{K}\psi\sum_{k=1}^{K}\sum_{\tau=t_0}^{t}\|\mathbf{g}_\tau^k\|^2 - \frac{(M-\kappa(1-\epsilon))}{2K\kappa^2(1-\epsilon)^2}\sum_{k=1}^{K}\sum_{\tau=t_0}^{t}\|\mathbf{g}_\tau^k\|^2 - \frac{1}{K}\sum_{k=1}^{K}\sum_{\tau=t_0}^{t}\frac{\eta}{\kappa(1-\epsilon)}\|\mathbf{g}_\tau^k\| \tag{36}$$

$$= \frac{C}{K}\sum_{k=1}^{K}\sum_{\tau=t_0}^{t}\|\mathbf{g}_\tau^k\|^2 - \frac{\eta L\Gamma}{\kappa(1-\epsilon)} \tag{37}$$

where $C = \psi - \frac{(M-\kappa(1-\epsilon))}{2K\kappa^2(1-\epsilon)^2}$

□

C Auxiliary Lemmas and their Proofs

Here, we prove the auxiliary lemmas that are used in the main proofs of the paper. (For completeness, we restate the lemma statements).

Lemma C.1. *Let $f(\cdot)$ satisfy assumptions 1-4 and $0 < \epsilon \leqslant 1/2$ and $\delta < 1$ be fixed constants. Then, if $s \geqslant \frac{4B}{\kappa\epsilon^2}\log\frac{2d}{\delta}$, the local Hessian at the k-th worker satisfies*

$$(1-\epsilon)\kappa \preccurlyeq \nabla^2 f^k(\mathbf{w}) = \mathbf{H}^k(\mathbf{w}) \preccurlyeq (1+\epsilon)M, \tag{38}$$

for all $\mathbf{w} \in \mathbb{R}^d$ and $k \in [K]$ with probability (w.p.) at least $1-\delta$.

Proof. At the k-th worker which samples $\mathcal{S}_k$ observations from $[n]$, the following is true by Matrix Chernoff (see Theorem 2.2 in Tropp (2011))

$$\mathbb{P}(\lambda_{\min}\left(\nabla^2 f^k(\mathbf{w})\right) \leqslant (1-\epsilon)\kappa\Big) \leqslant \delta_1 = d\left[\frac{e^{-\epsilon}}{(1-\epsilon)^{1-\epsilon}}\right]^{s\kappa/B}, \tag{39}$$

$$\mathbb{P}(\lambda_{\max}\left(\nabla^2 f^k(\mathbf{w})\right) \geqslant (1+\epsilon)M\Big) \leqslant \delta_2 = d\left[\frac{e^{\epsilon}}{(1+\epsilon)^{1+\epsilon}}\right]^{sM/B}. \tag{40}$$

Now, using the inequality $\log(1-\epsilon) \leqslant \frac{-\epsilon}{\sqrt{1-\epsilon}}$ for $0 \leqslant \epsilon < 1$, we get

$$\frac{e^{-\epsilon}}{(1-\epsilon)^{1-\epsilon}} \leqslant e^{-\epsilon+\epsilon\sqrt{1-\epsilon}}.$$

Further, utilizing the fact that $\sqrt{1-\epsilon} \leqslant \frac{1}{1+\epsilon/2}$, we get

$$e^{-\epsilon+\epsilon\sqrt{1-\epsilon}} \leqslant e^{\frac{-\epsilon^2}{1+\epsilon/2}} \leqslant e^{-\epsilon^2/4}.$$

Hence, we have $\delta_1 \leqslant de^{-s\kappa\epsilon^2/4B}$. Further, using the fact that $\log(1+\epsilon) \geqslant \epsilon - \epsilon^2/2$, we get

$$\frac{e^{\epsilon}}{(1+\epsilon)^{1+\epsilon}} \leqslant e^{-\epsilon^2/2+\epsilon^3/2} \leqslant e^{-\epsilon^2/4},$$

where the last inequality follows from the fact that $\epsilon \leqslant 1/2$. Hence, $\delta_2 \leqslant de^{-sM\epsilon^2/4B}$. Thus, by union bound and subsequently using upper bounds on δ_1 and δ_2, we get

$$\begin{aligned}\mathbb{P}\left[(1-\epsilon)\kappa\mathbf{I} \preccurlyeq \nabla^2 f^k(\mathbf{w}) \preccurlyeq (1+\epsilon)M\mathbf{I}\right] &\geqslant 1-(\delta_1+\delta_2)\\ &\geqslant 1-(de^{-s\kappa\epsilon^2/4B}+de^{-sM\epsilon^2/4B})\\ &\geqslant 1-(2de^{-s\kappa\epsilon^2/4B}),\end{aligned}$$

where the last inequality follows from the fact that $\kappa \leqslant M$. Hence, the result follows by noting that

$$(1-\epsilon)\kappa\mathbf{I} \preccurlyeq \nabla^2 f^k(\mathbf{w}) \preccurlyeq (1+\epsilon)M\mathbf{I} \;\text{ w. p. at least } 1-\delta,$$

and requiring that $\delta \geqslant 2de^{-s\kappa\epsilon^2/4B}$ (or $s \geqslant \frac{4B}{\kappa\epsilon^2}\log\frac{2d}{\delta}$).

□

Lemma C.2. *Let the function $f(\cdot)$ satisfy assumptions 1-3, and step-size α_t^k that solves the line-search condition in Eq. (5). Also, let $0 < \epsilon \leqslant 1/2$ and $0 < \delta < 1$ be fixed constants. Moreover, let the sample size $s \geqslant \frac{4B}{\kappa\epsilon^2}\log\frac{2d}{\delta}$. Then, the LocalNewton update at the k-th worker satisfy*

$$f^k(\mathbf{w}_{t+1}^k) - f^k(\mathbf{w}_t^k) \leqslant -\psi||\mathbf{g}_t^k||^2 \;\forall\; k \in [K],$$

w.p. at least $1-\delta$, where $\psi = \frac{\alpha^\star\beta}{M(1+\epsilon)}$.

Proof. From Lemma C.1, we know that $f^k(\cdot)$ is $M(1-\epsilon)$ smooth with probability $1-\delta$. M-smoothness of a function $g(\cdot)$ implies

$$g(\mathbf{y}) - g(\mathbf{x}) \leqslant (\mathbf{y}-\mathbf{x})^T\nabla g(\mathbf{x}) + \frac{M}{2}||\mathbf{y}-\mathbf{x}||^2 \;\forall\; \mathbf{x},\mathbf{y} \in \mathbb{R}^d. \tag{41}$$

Hence,

$$f^k(\mathbf{w}_t^k - \alpha\mathbf{p}_t^k) - f^k(\mathbf{w}_t^k) \leqslant (-\alpha\mathbf{p}_t^k)^T\mathbf{g}^k(\mathbf{w}_t^k) + \frac{M(1-\epsilon)}{2}\alpha^2||\mathbf{p}_t^k||^2. \tag{42}$$

The above inequality is satisfied for all $\alpha \in \mathbb{R}$. We know that α_t^k, the local step-size at worker k, satisfies the line-search constraint in Eq. (5). Thus, for $\alpha_t^k \in (0,1]$ to exist that satisfies the line-search condition, it is enough to find $\alpha > 0$ that satisfies

$$-\alpha(\mathbf{p}_t^k)^T\mathbf{H}_t^k\mathbf{p}_t^k + \frac{M(1-\epsilon)}{2}\alpha^2\|\mathbf{p}_t^k\|^2 \leqslant -\alpha\beta(\mathbf{p}_t^k)^T\mathbf{H}_t^k\mathbf{p}_t^k, \tag{43}$$

where we have used the fact that $\mathbf{g}_t^k = \mathbf{H}_t^k\mathbf{p}_t^k$. Thus, α must satisfy

$$\frac{M(1-\epsilon)}{2}\alpha\|\mathbf{p}_t^k\|^2 \leqslant (1-\beta)(\mathbf{p}_t^k)^T\mathbf{H}_t^k\mathbf{p}_t^k. \tag{44}$$

Using lemma C.1, we know that for sufficiently large sample-size at the k-th worker, we get

$$(1-\epsilon)\nabla^2 f(\mathbf{w}) \preceq \nabla^2 f^k(\mathbf{w}) \preceq (1+\epsilon)\nabla^2 f(\mathbf{w}) \tag{45}$$

with probability $1-\delta$. Also, by κ-strong convexity of $f(\cdot)$, we know that $\nabla^2 f(\mathbf{w}) \succeq \kappa\mathbf{I}$. Thus, the local line-search constraint is always satisfied for

$$\alpha \leqslant \frac{2(1-\beta)\kappa(1-\epsilon)}{M(1+\epsilon)}.$$

Hence, if we choose $\alpha^\star \leqslant \frac{2(1-\beta)\kappa(1-\epsilon)}{M(1+\epsilon)}$, or $\alpha^\star \leqslant \frac{\kappa(1-\beta)}{M}$ for $\epsilon < 1/2$, we are guaranteed to have the line-search condition from Eq. (5) satisfied with $\alpha_t^k = \alpha^\star$. This is satisfied by the line search equation in Eq. (5). Hence, from the line-search guarantee, we get

$$\begin{aligned} f^k(\mathbf{w}_{t+1}^k) - f^k(\mathbf{w}_t^k) &\leqslant -\alpha^\star\beta(\mathbf{p}_t^k)^T\mathbf{g}_t^k \quad (46)\\ &= \alpha^\star\beta(\mathbf{g}_t^k)^T(\mathbf{H}_t^k)^{-1}\mathbf{g}_t^k, \quad (47)\\ &\leqslant -\alpha^\star\beta\frac{1}{M(1+\epsilon)}\|\mathbf{g}_t^k\|^2, \quad (48)\end{aligned}$$

w.p. $1-\delta$. Here, the last inequality uses the fact that $f^k(\cdot)$ is $M(1+\epsilon)$–smooth, that is, $\mathbf{H}_t^k \preceq M(1+\epsilon)\mathbf{I}$. This proves the desired result. □

D Concentration Inequalities: With and without Error Floor

Consider a vector $v \in \mathbb{R}^d$. We have defined the following: $g(\bar{\mathbf{w}}_t) = \frac{1}{n}\sum_i g_i(\bar{\mathbf{w}}_t)$ and $g^k(\bar{\mathbf{w}}_t) = \frac{1}{s}\sum_{i\in\mathcal{S}} g_i(\bar{\mathbf{w}}_t)$, where g_i denotes the local gradient in worker machine i, and $\mathcal{S}$ is the random set consisting data points for machine k. Let us do the calculation in 2 settings:

D.1 With error floor

Here we have the error floor. Note that having an error floor is not restrictive, if we go for the adaptive variation of the algorithm, where we run GIANT for the final iterations. Since GIANT has no error floor, the final accuracy won't be affected by the error floor obtained in the first few steps of the algorithm (check if this is true).

Lemma D.1 (McDiarmid's Inequality). *Let $X = X_1, \ldots, X_m$ be m independent random variables taking values from some set A, and assume that $f : A^m \to \mathbb{R}$ satisfies the following condition (bounded differences):*

$$\sup_{x_1,\ldots,x_m,\hat{x}_i} |f(x_i,\ldots,x_i,\ldots,x_m) - f(x_i,\ldots,\hat{x}_i,\ldots,x_m)| \leqslant c_i,$$

for all $i \in \{1,\ldots,m\}$. Then for any $\epsilon > 0$ we have

$$P\left[f(X_1,\ldots,X_m) - \mathbb{E}[f(X_1,\ldots,X_m)] \geqslant \epsilon\right] \leqslant \exp\left(-\frac{2\epsilon^2}{\sum_{i=1}^m c_i^2}\right).$$

The property described in the following is useful for uniform row sampling matrix.

Let $\mathbf{S} \in \mathbb{R}^{n\times s}$ be any uniform sampling matrix, then for any matrix $\mathbf{B} = [\mathbf{b}_1,\ldots,\mathbf{b}_n] \in \mathbb{R}^{d\times n}$ with probability $1-\delta$ for any $\delta > 0$ we have,

$$\|\frac{1}{n}\mathbf{B}\mathbf{S}\mathbf{S}^\top\mathbf{1} - \frac{1}{n}\mathbf{B}\mathbf{1}\| \leqslant (1+\sqrt{2\log(\frac{1}{\delta})})\sqrt{\frac{1}{s}}\max_i \|\mathbf{b}_i\|, \tag{49}$$

where $\mathbf{1}$ is all ones vector.

Let us first see the justification of the above statement.The vector $\mathbf{B1}$ is the sum of column of the matrix $\mathbf{B}$ and $\mathbf{B}\mathbf{S}\mathbf{S}^\top\mathbf{1}$ is the sum of uniformly sampled and scaled column of the matrix $\mathbf{B}$ where the scaling factor is $\frac{1}{\sqrt{sp}}$ with $p = \frac{1}{n}$. If $(i_1,\ldots,i_s)$ is the set of sampled indices then $\mathbf{B}\mathbf{S}\mathbf{S}^\top\mathbf{1} = \sum_{k\in(i_1,\ldots,i_s)}\frac{1}{sp}\mathbf{b}_k$.

Define the function $f(i_1, \ldots, i_s) = \|\frac{1}{n}\mathbf{B}\mathbf{S}\mathbf{S}^\top\mathbf{1} - \frac{1}{n}\mathbf{B}\mathbf{1}\|$. Now consider a sampled set $(i_1, \ldots, i_{j'}, \ldots, i_s)$ with only one item (column) replaced then the bounded difference is

$$\begin{aligned}\Delta &= |f(i_1, \ldots, i_j, \ldots, i_s) - f(i_1, \ldots, i_{j'}, \ldots, i_s)| \\ &= |\frac{1}{n}\|\frac{1}{sp}\mathbf{b}_{i'_j} - \frac{1}{sp}\mathbf{b}_{i_j}\|| \leqslant \frac{2}{s}\max_i \|\mathbf{b}_i\|.\end{aligned}$$

Now we have the expectation

$$\begin{aligned}&\mathbb{E}[\|\frac{1}{n}\mathbf{B}\mathbf{S}\mathbf{S}^\top\mathbf{1} - \frac{1}{n}\mathbf{B}\mathbf{1}\|^2] \leqslant \frac{n}{sn^2}\sum_{i=1}^n \|\mathbf{b}_i\|^2 = \frac{1}{s}\max_i \|\mathbf{b}_i\|^2 \\ \Rightarrow &\mathbb{E}[\|\frac{1}{n}\mathbf{B}\mathbf{S}\mathbf{S}^\top\mathbf{1} - \frac{1}{n}\mathbf{B}\mathbf{1}\|] \leqslant \sqrt{\frac{1}{s}}\max_i \|\mathbf{b}_i\|.\end{aligned}$$

Using McDiarmid inequality (Lemma D.1) we have

$$P\left[\|\frac{1}{n}\mathbf{B}\mathbf{S}\mathbf{S}^\top\mathbf{1} - \frac{1}{n}\mathbf{B}\mathbf{1}\| \geqslant \sqrt{\frac{1}{s}}\max_i \|\mathbf{b}_i\| + t\right] \leqslant \exp\left(-\frac{2t^2}{s\Delta^2}\right).$$

Equating the probability with δ we have

$$\begin{aligned}&\exp(-\frac{2t^2}{s\Delta^2}) = \delta \\ \Rightarrow &t = \Delta\sqrt{\frac{s}{2}\log(\frac{1}{\delta})} = \max_i \|\mathbf{b}_i\|\sqrt{\frac{2}{s}\log(\frac{1}{\delta})}.\end{aligned}$$

Finally we have with probability $1 - \delta$

$$\|\frac{1}{n}\mathbf{B}\mathbf{S}\mathbf{S}^\top\mathbf{1} - \frac{1}{n}\mathbf{B}\mathbf{1}\| \leqslant (1 + \sqrt{2\log(\frac{1}{\delta})})\sqrt{\frac{1}{s}}\max_i \|\mathbf{b}_i\|,$$

and hence equation (49) is justified.

We now apply the above in distributed gradient estimation. For the k-th worker machine, we have

$$\|\frac{1}{n}\mathbf{B}\mathbf{S}_k\mathbf{S}_k^\top\mathbf{1} - \frac{1}{n}\mathbf{B}\mathbf{1}\| \leqslant (1 + \sqrt{2\log(\frac{1}{\delta})})\sqrt{\frac{1}{s}}\max_i \|\mathbf{b}_i\|,$$

with probability $1 - \delta$, which implies

$$\|g^k(\bar{\mathbf{w}}_t) - g(\bar{\mathbf{w}}_t)\| \leqslant (1 + \sqrt{2\log(\frac{1}{\delta})})\sqrt{\frac{1}{s}}\Gamma,$$

with probability at least $1 - \delta$ provided $\|g_i(\bar{\mathbf{w}}_t)\| \leqslant \Gamma$ for all $i \in [m]$

Writing, $\eta = (1 + \sqrt{2\log(\frac{1}{\delta})})\sqrt{\frac{1}{s}}L$, we succinctly write

$$|\langle v, g^k(\bar{\mathbf{w}}_t) - g(\bar{\mathbf{w}}_t)\rangle| \leqslant \|v\|\|g^k(\bar{\mathbf{w}}_t) - g(\bar{\mathbf{w}}_t)\| \leqslant \eta\|v\|$$

with probability at least $1 - \delta$, where $\eta = \mathcal{O}(1/\sqrt{s})$ is small.

D.2 Without error floor

In this section, we analyze the same quantity using vector Bernstein inequality. Intuitively, we show that unless $g(\bar{\mathbf{w}}_t)$ is too small, we can overcome the error floor shown in the previous calculation. In particular, we assume that

$$\|g^k(\bar{\mathbf{w}}_t)\| \geqslant G.$$

The idea here is to use the vector Bernstein inequality. Using the notation of Appendix D.1, $g^k(\bar{\mathbf{w}}_t) = \frac{1}{n}\mathbf{B}\mathbf{S}\mathbf{S}^\top\mathbf{1}$, where $\mathbf{S}$ is appropriately defined sampling matrix. Also $g(\bar{\mathbf{w}}_t) = \frac{1}{n}\mathbf{B}\mathbf{1}$. For the k-th machine,

$$g^k(\bar{\mathbf{w}}_t) = \frac{1}{s}\sum_{i\in\mathcal{S}} g_i(\bar{\mathbf{w}}_t),$$

and so,

$$g^k(\bar{\mathbf{w}}_t) - g(\bar{\mathbf{w}}_t) = \frac{1}{s}\sum_{i\in\mathcal{S}}(g_i(\bar{\mathbf{w}}_t) - g(\bar{\mathbf{w}}_t)),$$

with $|\mathcal{S}| = s$. We also have $\|g_i(\bar{\mathbf{w}}_t) - g(\bar{\mathbf{w}}_t)\| \leqslant \Gamma + \Gamma = 2\Gamma$, and $\mathbb{E}\|g_i(\bar{\mathbf{w}}_t) - g(\bar{\mathbf{w}}_t)\|^2 \leqslant 4\Gamma^2$. Using vector Bernstein inequality with $t = \epsilon_1\|g^k\|$, we obtain

$$\mathbb{P}\left(\|g^k(\bar{\mathbf{w}}_t) - g(\bar{\mathbf{w}}_t)\| \geqslant \epsilon_1\|g^k(\bar{\mathbf{w}}_t)\|\right) \leqslant d\exp(-s\frac{\epsilon_1^2\|g^k\|^2}{32\Gamma^2} + 1/4) \leqslant d\exp(-s\frac{\epsilon_1^2 G^2}{32L^2} + 1/4).$$

So, as long as

$$G^2 = \Omega\left(\frac{\Gamma^2}{\epsilon_1^2 s}\log(d/\delta)\right),$$

or,

$$s \gtrsim \left(\frac{\Gamma^2}{\epsilon_1^2 G^2}\log(d/\delta)\right),$$

we have,

$$|\langle v, g^k(\bar{\mathbf{w}}_t) - g(\bar{\mathbf{w}}_t)\rangle| \leqslant \|v\|\|g^k(\bar{\mathbf{w}}_t) - g(\bar{\mathbf{w}}_t)\| \leqslant \epsilon_1\|v\|\|g^k\|$$

with probability at least $1-\delta$.

E Experiments: Additional Details and Plots

E.1 Additional Figures from Section 3.1

In this section, we include comparisons on additional datasets that couldn't be added in the main paper due to space constraints. In Figure 5, we compare GIANT and LocalNewton (for L=1, 2 and 3) and adaptive LocalNewton.

E.2 Hyperparameters for Local SGD and BFGS

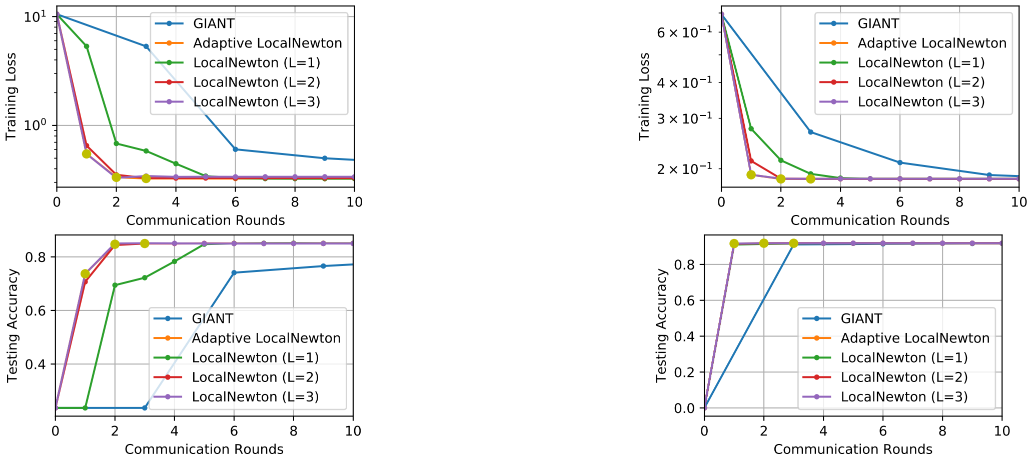
In Table 2, we provide the step-sizes for local SGD and BFGS obtained through hyperparameter tuning.

Dataset	Samples per worker (s)	Local SGD	BFGS
w8a	480	$10/s$	100
Covtype	5000	$10/s$	1
EPSILON	4000	$500/s$	10
a9a	320	$10/s$	1
ijcnn1	490	$100/s$	10

Table 2: Step-sizes obtained using tuning for Local SGD and BFGS for several datasets

E.3 Additional Figures from Section 4

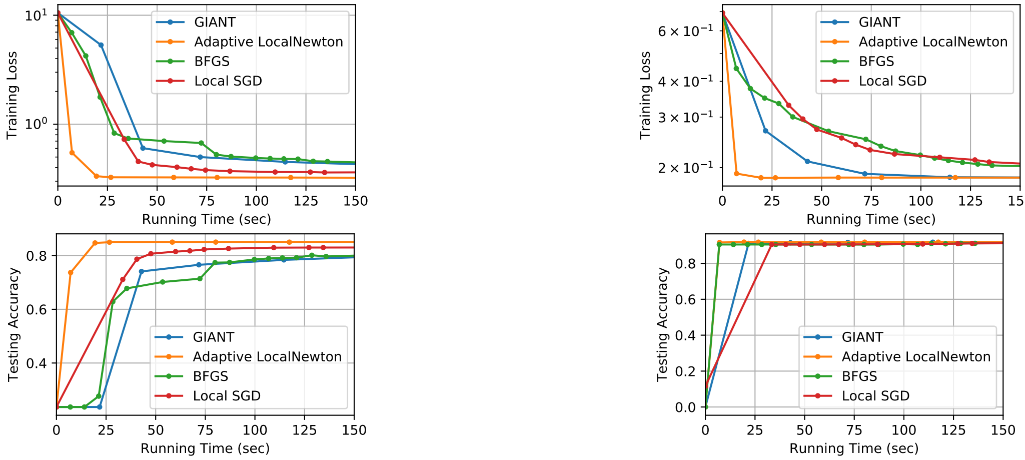
In Figure 6, we plot the results on AWS Lambda for a9a and ijcnn1 datasets. Again, adaptive LocalNewton considerable outperforms Local SGD, GIANT and BFGS in terms of end-to-end runtimes.



(a) a9a dataset

(b) ijcnn1 dataset

Figure 5: Comparing LocalNewton (for different values of L) and GIANT w.r.t. communication rounds

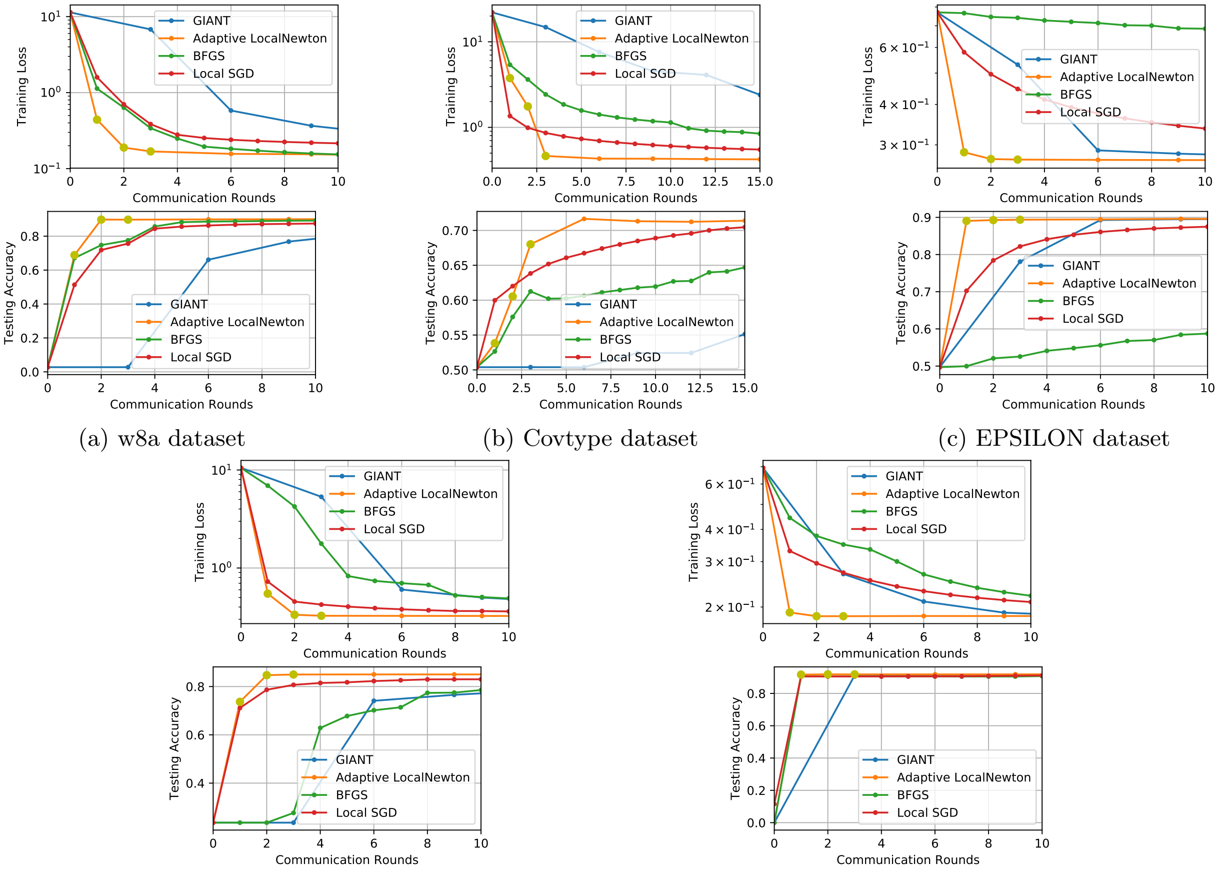


(a) a9a dataset

(b) ijcnn1 dataset

Figure 6: Experiments on the a9a and ijcnn1 datasets on AWS Lambda

Convergence w.r.t. communication rounds: In our main paper, we skipped the plots for convergence behavior w.r.t. communication rounds due to space constraints. In Figure 7, we show the convergence of adaptive LocalNewton, GIANT, Local SGD and BFGS with communication rounds for all the five datasets considered in this paper. Again, LocalNewton significantly outperforms existing schemes by reducing the communication rounds by at least 60% to reach the same training loss.



(a) w8a dataset (b) Covtype dataset (c) EPSILON dataset

(d) a9a dataset (e) ijcnn1 dataset

Figure 7: Comparing LocalNewton with competing schemes w.r.t. communication rounds. Yellow dots on adaptive LocalNewton denote transition from larger to smaller values of L (or to GIANT if $L = 1$).