Escaping Saddle Points in Heterogeneous Federated Learning via Distributed SGD with Communication Compression

Sijin Chen
Princeton University

Zhize Li
Carnegie Mellon University

Yuejie Chi
Carnegie Mellon University

Abstract

We consider the problem of finding second-order stationary points of heterogeneous federated learning (FL). Previous works in FL mostly focus on first-order convergence guarantees, which do not rule out the scenario of unstable saddle points. Meanwhile, it is a key bottleneck of FL to achieve communication efficiency without compensating the learning accuracy, especially when local data are highly heterogeneous across different clients. Given this, we propose a novel algorithm PowerEF-SGD that only communicates compressed information via a novel error-feedback scheme. To our knowledge, PowerEF-SGD is the first distributed and compressed SGD algorithm that provably escapes saddle points in heterogeneous FL without any data homogeneity assumptions. In particular, PowerEF-SGD improves to second-order stationary points after visiting first-order (possibly saddle) points, using additional gradient queries and communication rounds only of almost the same order required by first-order convergence, and the convergence rate exhibits a linear speedup in terms of the number of workers. Our theory improves/recovers previous results, while extending to much more tolerant settings on the local data. Numerical experiments are provided to complement the theory.

1 INTRODUCTION

The prevalence of large-scale data and enormous model size in modern machine learning problems give rise to an increasing interest in distributed machine learning, where a number of clients cooperate to handle the extremely heavy computation in the learning task without the need to move data around.

We consider a distributed server-client setting. Suppose that each client $i \in [n]$ has access to a local dataset $\mathcal{W}^{(i)}$ distributed over an unknown space Ω, and a central server maintains a model parameterized by $\boldsymbol{x} \in \mathbb{R}^d$. Given a cost function $F : \mathbb{R}^d \times \Omega \to \mathbb{R}$ that evaluates the performance of a model $\boldsymbol{x}$ on an input data sample $\omega \in \Omega$, the i-th local objective function f_i is defined by $f_i(\boldsymbol{x}) := \mathbb{E}_{\omega^{(i)} \sim \mathcal{W}^{(i)}}[F(\boldsymbol{x}, \omega^{(i)})]$. We would like to find a model parameter $\boldsymbol{x}$ that minimizes the local objectives in an averaged manner, which leads to a finite-sum minimization problem:

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

where the local objective functions $\{f_i\}_{i=1}^n$ and the global objective function $f = \frac{1}{n}\sum_{i=1}^n f_i$ are in general nonconvex, especially in machine learning applications.

Heterogeneous federated learning. Assumptions on data homogeneity across the clients can be deployed to underplay this problem to a certain extent, since intuitively, there are less disagreements across the local objectives to reconcile. For example, each local dataset $\mathcal{W}^{(i)}$ may take similar distributions, or may be uploaded to a data center that maintains global knowledge (Konečný et al., 2016). However, in many real applications such as Internet of Things (IoT) (Nguyen et al., 2021; Savazzi et al., 2020), smart healthcare (Xu et al., 2021), and networked model devices (Kang et al., 2020), such assumptions become impractical in that local datasets display a strongly heterogeneous pattern, while they should not be exchanged or exposed to a third party due to privacy sensitivity or communication infeasibility (Konečný et al., 2016). These thorny scenarios of data heterogeneity correspond to a framework for distributed learning, namely federated learning (FL) (Kairouz et al., 2019),

Proceedings of the 27th International Conference on Artificial Intelligence and Statistics (AISTATS) 2024, Valencia, Spain. PMLR: Volume 238.

which is now accumulating special attention from both academia and industry. The heterogeneous data constitute a major challenge in the distributed optimization problem under federated settings, which we refer to as heterogeneous FL.

Distributed SGD with communication compression. A prevalent approach to solve (1) is by distributed stochastic gradient descent (SGD) (Koloskova et al., 2020), a family of algorithms following the essential idea that each client computes its local stochastic gradient and then sends the gradient (or a carefully designed surrogate for the gradient) to the central server for parameter update. Distributed SGD has to take good care of communication efficiency: due to the large client number n (Savazzi et al., 2020) and model scale d (Brown et al., 2020) in modern machine learning tasks, the communication cost from the clients to the server becomes the main bottleneck of optimization. Moreover, many resource constraints in real communication systems, such as limited bandwidth and stringent delay requirements, also highlight the importance of establishing efficient communication for the distributed training procedure.

A natural method to attain communication efficiency is (lossy) compression: one can deploy a *compressor* $\mathcal{C} : \mathbb{R}^d \to \mathbb{R}^d$ in distributed SGD, which compresses any message $\boldsymbol{x} \in \mathbb{R}^d$ the client would like to send to the server, so that the traffic $\mathcal{C}(\boldsymbol{x})$ takes up a smaller bandwidth. In literature, a randomized operator $\mathcal{C} : \mathbb{R}^d \to \mathbb{R}^d$ is said to be a μ-compressor if the (expected) relative distortion of the compressed output is bounded by μ (Fatkhullin et al., 2021; Huang et al., 2022; Richtárik et al., 2021; Stich et al., 2018), which helps quantify the information loss due to compression.

Motivation. It has been a recent interest to establish convergence results for distributed SGD with communication compression. Many among these works (Huang et al., 2022; Koloskova et al., 2019a; Stich et al., 2018; Xie et al., 2020) assume bounded local gradients $\|\nabla f_i(\boldsymbol{x})\|^2 \leq G^2$, or bounded dissimilarity of local gradients $\frac{1}{n}\|\sum_{i=1}^n \nabla f_i(\boldsymbol{x}) - \nabla f(\boldsymbol{x})\|^2 \leq G^2$, reflecting a reliance on data homogeneity that fails to hold in heterogeneous FL. Another body of the works (Fatkhullin et al., 2021; Richtárik et al., 2021, 2022; Zhao et al., 2022), although allowing heterogeneous data, only ensures first-order optimality, i.e. convergence to an ϵ-optimal first-order stationary point $\boldsymbol{x}$ with $\|\nabla f(\boldsymbol{x})\| \leq \epsilon$, which does not suffice to justify the goodness of the solution in the nonconvex setting where saddle points are abundant and do not necessarily lead to generalizable performance (Dauphin et al., 2014). It is then important to obtain second-order convergence guarantees that ensure the algorithm escapes the saddle points and converges to an ϵ-optimal second-order stationary point, with an additional control on the Hessian positive-definiteness that says $-\lambda_{\min}(\nabla^2 f(\boldsymbol{x})) \leq O(\sqrt{\epsilon})$. Despite the growing literature of saddle-point escaping algorithms in the centralized setting (Daneshmand et al., 2018; Ge et al., 2015; Jin et al., 2021; Li, 2019), to the best of our knowledge, no existing distributed SGD algorithms succeed with second-order guarantees in the presence of both communication compression and data heterogeneity. In summary, the current research sparked a natural question as the primary concern of this paper:

On heterogeneous data, is there a distributed SGD algorithm with communication compression that attains second-order convergence guarantees for nonconvex problems?

1.1 Our contribution

To the best of our knowledge, this work is the first to answer the above question affirmatively. Our specific contributions are as follows.

- **A novel error-feedback mechanism:** we propose PowerEF-SGD, a new distributed SGD algorithm that contains a novel error-feedback mechanism for communication compression.
- **First-order convergence:** we prove that, with high probability, PowerEF-SGD converges to ϵ-optimal first-order stationary points within $\widetilde{O}\left(\frac{1}{n\epsilon^4} + \frac{1}{\mu^{1.5}\epsilon^3} + \frac{1}{\mu^2\epsilon^2}\right)$ stochastic gradient queries and communication rounds. The algorithm shows a linear speedup pattern in that the convergence rate benefits from the number of workers n.
- **Second-order convergence:** we prove that, with high probability, PowerEF-SGD escapes the saddle points and converges to ϵ-optimal second-order stationary points within $\widetilde{O}\left(\frac{1}{n\epsilon^4} + \frac{1}{\mu^{1.5}\epsilon^3} + \frac{\mu n+1}{\mu^3\epsilon^{2.5}}\right)$ stochastic gradient queries and communication rounds. This suggests that PowerEF-SGD finds second-order stationary points with almost the same order of gradient and communication complexities as it takes to for first-order convergence.
- **Convergence under arbitrary data heterogeneity:** importantly, the theory of PowerEF-SGD does not require assumptions on data similarity between different clients, thus allowing arbitrary heterogeneity in federated learning tasks.

See also Table 1 and 2 for a detailed comparison be-

tween our proposed method and existing algorithms.

1.2 Related works

Communication compression. A communication operator, or a compressor, is deployed to reduce the communication cost in distributed SGD. Various instances of compressors include Quantized SGD (Alistarh et al., 2017) that rounds real-valued gradient vectors to discrete buckets, Sign SGD (Bernstein et al., 2018) that represents the gradient with the sign of each coordinate, Top-k (Stich et al., 2018) that selects k coordinates out of the total dimension d with the largest magnitudes, and Random-k (Stich et al., 2018) that performs the above selection uniformly at random, among others. Regardless of the specific design, a general biased compressor is characterized by a parameter $\mu \in (0, 1]$ that controls the aforementioned distortion of the operator.

With a compressor at hand, one also needs a mechanism that specifies what message should be compressed and transmitted between clients. A naive, prototypical mechanism is to directly replace the gradient with its compressed version in the regular routine of SGD or its momentum variants. This mechanism underpins Alistarh et al. (2017); Bernstein et al. (2018), among others. However, error may accumulate in this simple replacement due to the lossy compression and menace its convergence. Various works propose new mechanisms to properly handle the error to boost the convergence performance, including Error-Feedback (Avdiukhin and Yaroslavtsev, 2021; Karimireddy et al., 2019; Li and Chi, 2023; Li et al., 2022; Seide et al., 2014; Stich et al., 2018) and its variants (Fatkhullin et al., 2021; Huang et al., 2022; Richtárik et al., 2021), with adaptations to decentralized optimization (Koloskova et al., 2019a,b; Zhao et al., 2022). Most of the works guarantee first-order convergence subject to different levels of assumptions on data homogeneity, cf. Tables 1 and 2.

Second-order convergence of gradient methods. It is well-known that gradient methods converge to first-order stationary points (Nesterov, 2004). In nonconvex problems, however, first-order convergence can be easily attacked by saddle points that may trap the GD trajectory. It is therefore important to investigate whether the algorithm is capable of escaping saddle points and converging to second-order stationary points. Asymptotically, Lee et al. (2016) proved that GD with random initialization converges to a local minimum almost surely. However, the algorithm may still have to take an exponential time to escape the saddle points (Du et al., 2017).

As to the polynomial-time guarantees, it is known that perturbing the gradient with isotropic noise helps GD converge to local minimizers (Ge et al., 2015; Jin et al., 2017). The perturbation technique gives rise to similar guarantees for other gradient methods, from SGD (Jin et al., 2021) to SVRG (Ge et al., 2019) and stochastic recursive gradient descent (Li, 2019). On the other hand, instead of gradient perturbation, Daneshmand et al. (2018) establishes the saddle-escaping property of SGD under an additional Correlated Negative Curvature (CNC) assumption regarding the statistical property of the stochastic gradient oracle.

Recently, Avdiukhin and Yaroslavtsev (2021) leverages the perturbation technique to analyze the second-order stationarity of SGD with communication compression. The theoretical derivation is based on *single-node* implementation, which does not directly extend to the distributed settings. Further, it requires a conditional reset procedure in each iteration to achieve second-order convergence, at the expense of high communication cost as the server has to collect and maintain the local error terms using *uncompressed* channel. Therefore, it remains obscure if the results therein still apply to the distributed setting with communication efficiency demands.

1.3 Notation

Throughout, we use lowercase boldface letters to denote vectors, and uppercase boldface letters to denote matrices. Let $\boldsymbol{I}$ be the identity matrix. Let $\langle \boldsymbol{u}, \boldsymbol{v} \rangle := \boldsymbol{u}^\top \boldsymbol{v}$ denote the standard Euclidean inner product of two vectors $\boldsymbol{u}$ and $\boldsymbol{v}$. The operator $\|\cdot\|$ denotes the Euclidean norm when exerted on a vector, i.e. $\|\boldsymbol{x}\| := \sqrt{\langle \boldsymbol{x}, \boldsymbol{x} \rangle} = \sqrt{\boldsymbol{x}^\top \boldsymbol{x}}$, and denotes the spectral (operator) norm when exerted on a matrix, i.e. $\|\boldsymbol{A}\| := \sup_{\boldsymbol{x}} \|\boldsymbol{A}\boldsymbol{x}\| / \|\boldsymbol{x}\|$. In addition, we use the standard order notation $O(\cdot)$ to hide absolute constants, and $\widetilde{O}(\cdot)$ to hide polylog factors.

2 PROBLEM FORMULATION

This paper is primarily concerned with solving the nonconvex finite-sum minimization problem in a federated setting, while each client should only query a local stochastic gradient oracle, and communicate their information with the server in an efficient manner using compression. We detail this formulation in the following.

2.1 Nonconvex finite-sum minimization

Recall that we consider a federated optimization problem of finding an optimal parameter $\boldsymbol{x}$ to minimize the local objectives $\{f_i\}_{i=1}^n$ in an averaged manner, which is stated as an unconstrained finite-sum minimization

Table 1: Comparison of algorithms using *stochastic gradients* for nonconvex problems. Stochastic gradient complexity refers to the number of stochastic gradient queries required to converge to ϵ-optimal first-order or ϵ-optimal second-order stationary points, and μ refers to the parameter of the compressor.

Algorithm	Stochastic gradient complexity	Result guarantee	Data homogeneity assumption	Distributed?	Compression?
SGD (Ghadimi et al., 2016)	$O\left(\frac{1}{\epsilon^4}\right)$	1st-order	not applicable	NO	NO
Compressed SGD (Avdiukhin and Yaroslavtsev, 2021)	$O\left(\frac{1}{\epsilon^4}+\frac{1}{\mu\epsilon^3}\right)$	1st-order	not applicable	NO	**YES**
CHOCO-SGD (Koloskova et al., 2019a)	$O\left(\frac{1}{n\epsilon^4}+\frac{1}{\mu\epsilon^3}\right)$	1st-order	bounded gradient	**YES**	**YES**
CSER (Xie et al., 2020)	$O\left(\frac{1}{n\epsilon^4}+\frac{1}{\mu\epsilon^3}\right)$	1st-order	bounded gradient	**YES**	**YES**
NEOLITHIC (Huang et al., 2022)	$\widetilde{O}\left(\frac{1}{n\epsilon^4}+\frac{1}{\mu\epsilon^2}\right)$	1st-order	gradient similarity	**YES**	**YES**
EF21-SGD (Fatkhullin et al., 2021)	$O\left(\frac{1}{\mu^3\epsilon^4}+\frac{1}{\mu\epsilon^2}\right)$	1st-order	**NONE**	**YES**	**YES**
PowerEF-SGD (Algorithm 1)	$\widetilde{O}\left(\frac{1}{n\epsilon^4}+\frac{1}{\mu^{1.5}\epsilon^3}+\frac{1}{\mu^2\epsilon^2}\right)$	1st-order	**NONE**	**YES**	**YES**
Noisy SGD (Ge et al., 2015)	$\text{poly}(\frac{1}{\epsilon})$	2nd-order	not applicable	NO	NO
CNC-SGD (Daneshmand et al., 2018)	$\widetilde{O}\left(\frac{1}{\epsilon^5}\right)$	2nd-order	not applicable	NO	NO
Perturbed SGD (Jin et al., 2021)	$\widetilde{O}\left(\frac{1}{\epsilon^4}\right)$	2nd-order	not applicable	NO	NO
Compressed SGD (Avdiukhin and Yaroslavtsev, 2021)	$\widetilde{O}\left(\frac{1}{\epsilon^4}+\frac{1}{\mu\epsilon^3}+\frac{1}{\mu^2\epsilon^{2.5}}\right)$	2nd-order	not applicable	NO	**YES**
PowerEF-SGD (Algorithm 1)	$\widetilde{O}\left(\frac{1}{n\epsilon^4}+\frac{1}{\mu^{1.5}\epsilon^3}+\frac{\mu n+1}{\mu^3\epsilon^{2.5}}\right)$	2nd-order	**NONE**	**YES**	**YES**

Table 2: Comparison of *distributed and compressed* algorithms using stochastic gradients for nonconvex problems. Communication rounds refers to the number of compressed messages transmitted between clients and the server.

Algorithm	Communication rounds	Result guarantee	Data homogeneity assumption
CHOCO-SGD (Koloskova et al., 2019a)	$O\left(\frac{1}{n\epsilon^4}+\frac{1}{\mu\epsilon^3}\right)$	1st-order	bounded gradient
CSER (Xie et al., 2020)	$O\left(\frac{1}{n\epsilon^4}+\frac{1}{\mu\epsilon^3}\right)$	1st-order	bounded gradient
NEOLITHIC (Huang et al., 2022)	$\widetilde{O}\left(\frac{1}{n\epsilon^4}+\frac{1}{\mu\epsilon^2}\right)$	1st-order	gradient similarity
EF21-SGD (Fatkhullin et al., 2021)	$O\left(\frac{1}{\mu^3\epsilon^4}+\frac{1}{\mu\epsilon^2}\right)$	1st-order	**NONE**
PowerEF-SGD (Algorithm 1)	$\widetilde{O}\left(\frac{1}{n\epsilon^4}+\frac{1}{\mu^{1.5}\epsilon^3}+\frac{1}{\mu^2\epsilon^2}\right)$	1st-order	**NONE**
PowerEF-SGD (Algorithm 1)	$\widetilde{O}\left(\frac{1}{n\epsilon^4}+\frac{1}{\mu^{1.5}\epsilon^3}+\frac{\mu n+1}{\mu^3\epsilon^{2.5}}\right)$	2nd-order	**NONE**

problem:

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

where $\{f_i\}_{i=1}^n$'s are the local objective functions, and n is the number of clients.

We focus on the case where the objective functions are nonconvex, subject to the following assumptions.

Assumption 2.1. *There exists some $f_{\min} > -\infty$ such that $f(\boldsymbol{x}) \geq f_{\min}$ for all $\boldsymbol{x} \in \mathbb{R}^d$.*

We will leverage the boundedness in Assumption 2.1 to establish first-order convergence results. For second-order results, similar to Avdiukhin and Yaroslavtsev (2021), the following alternative is required.

Assumption 2.1*. *There exists some $f_{\max} < \infty$ such that $|f(\boldsymbol{x}_1) - f(\boldsymbol{x}_2)| \leq f_{\max}$ for all $\boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathbb{R}^d$.*

Besides boundedness, we also assume the smoothness of f.

Assumption 2.2. *f is differentiable and L-smooth, i.e.*

$$\|\nabla f(\boldsymbol{x}_1) - \nabla f(\boldsymbol{x}_2)\| \leq L\|\boldsymbol{x}_1 - \boldsymbol{x}_2\|, \quad \forall \boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathbb{R}^d.$$

In the same spirit as what we do for the boundedness assumption, we need to further assume a Lipschitz property of the Hessian to prove second-order results.

Assumption 2.3. *f is twice differentiable and ρ-Hessian Lipschitz, i.e.,*

$$\left\|\nabla^2 f(\boldsymbol{x}_1) - \nabla^2 f(\boldsymbol{x}_2)\right\| \leq \rho \left\|\boldsymbol{x}_1 - \boldsymbol{x}_2\right\|, \quad \forall \boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathbb{R}^d.$$

We emphasize that no assumption is made on the boundedness of, or similarity between, the local gradients.

2.2 Local stochastic gradient oracle

Each client i is allowed to query a local stochastic gradient oracle $\widetilde{\nabla} f_i$.

Assumption 2.4. *Each $\widetilde{\nabla} f_i$ is $\widetilde{L}_i$-Lipschitz, i.e.*

$$\left\|\widetilde{\nabla} f_i(\boldsymbol{x}_1) - \widetilde{\nabla} f_i(\boldsymbol{x}_2)\right\| \leq \widetilde{L}_i \left\|\boldsymbol{x}_1 - \boldsymbol{x}_2\right\|, \quad \forall \boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathbb{R}^d.$$

Based on Assumption 2.4, it is straightforward to verify that the global stochastic gradient $\widetilde{\nabla} f$ is $\widetilde{L}$-smooth with $\widetilde{L} := \sqrt{\frac{1}{n}\sum_{i=1}^n \widetilde{L}_i^2}$.

Besides smoothness, the stochastic gradients should also approximate the true gradients.

Assumption 2.5. *For any $\boldsymbol{x} \in \mathbb{R}^d$, the mutually independent stochastic gradient oracles $\widetilde{\nabla} f_i$ satisfy*

$$\mathbb{E}\left[\widetilde{\nabla} f_i(\boldsymbol{x})\right] = \nabla f_i(\boldsymbol{x}),$$
$$\Pr\left(\left\|\widetilde{\nabla} f_i(\boldsymbol{x}) - \nabla f_i(\boldsymbol{x})\right\| \geq t\right) \leq 2\exp\left(-\frac{t^2}{2\sigma^2}\right)$$

for all $t \geq 0$ and some $\sigma > 0$.

Assumption 2.5 is a high-probability variant of the commonly-used bounded variance assumption, stated in expectation. Switching to such a high-probability variant is again necessary for second-order analysis (Jin et al., 2021; Li, 2019) because we aim at a convergence guarantee with probability bounds.

Gradient accumulation. For an integer k, let $\widetilde{\nabla} f_i^{(1)}(\boldsymbol{x}), ..., \widetilde{\nabla} f_i^{(k)}(\boldsymbol{x})$ be the k independent queries to the stochastic oracle at $\boldsymbol{x}$. The accumulated gradient is defined as their average, i.e.

$$\widetilde{\nabla}_k f_i(\boldsymbol{x}) = \frac{1}{k}\sum_{j=1}^k \widetilde{\nabla} f_i^{(j)}(\boldsymbol{x}).$$

We shall stick to the accumulated gradient in our algorithm design.

2.3 Communication compression

To enable efficient communication over bandwidth-limited scenarios, our setting demands that the communication between the clients and the server should be compressed according to a possibly randomized scheme $\mathcal{C}$. Specifically, for any input $\boldsymbol{x} \in \mathbb{R}^d$, the scheme should compute a surrogate $\mathcal{C}(\boldsymbol{x}) \in \mathbb{R}^d$ so that the transmission of $\mathcal{C}(\boldsymbol{x})$ between machines would take up a smaller bandwidth than the direct transmission of $\boldsymbol{x}$.

Definition 2.6. *A possibly random mapping $\mathcal{C} : \mathbb{R}^d \to \mathbb{R}^d$ is said to be a μ-compressor for some $\mu \in (0, 1]$ if*

$$\left\|\boldsymbol{x} - \mathcal{C}(\boldsymbol{x})\right\|^2 \leq (1-\mu)\left\|\boldsymbol{x}\right\|^2, \quad \forall \boldsymbol{x} \in \mathbb{R}^d.$$

In literature, the definition of a compresssor is usually stated in the expectation sense, i.e., $\mathbb{E}[\|\boldsymbol{x} - \mathcal{C}(\boldsymbol{x})\|^2] \leq (1-\mu)\|\boldsymbol{x}\|^2$. Here, we use the deterministic version only to comply with our high-probability analysis framework. Technically, it is not hard to adapt our entire theory to the language of expectations, where the expected version of Definition 2.6 comes into use. Examples of compressors that satisfy Definition 2.6 include Top-k (Stich et al., 2018) and a family of compressors named general biased rounding (Beznosikov et al., 2020).

3 PROPOSED ALGORITHM

This section introduces our proposed algorithm PowerEF-SGD that is suitable to heterogeneous FL with communication compression.

3.1 Fast Compressed Communication

We first introduce the Fast Compressed Communication (FCC) module proposed by Huang et al. (2022), which is deployed at each client in their compressed SGD algorithm NEOLITHIC. For input $\boldsymbol{x} \in \mathbb{R}^d$, the FCC module with parameter $p \in \mathbb{Z}_+$ recursively computes the residual $\{\boldsymbol{v}_i\}_{i=1}^p$ for p rounds, where

$$\boldsymbol{v}_1 = \boldsymbol{x}; \quad \boldsymbol{v}_i = \boldsymbol{x} - \sum_{j=1}^{i-1} \mathcal{C}(\boldsymbol{v}_j), \quad i = 2, ..., p.$$

It then outputs $\mathrm{FCC}_p(\boldsymbol{x}) = \sum_{i=1}^p \mathcal{C}(\boldsymbol{v}_i)$. To transmit the output to the server efficiently, the client transmits the set of compressed vectors $\{\mathcal{C}(\boldsymbol{v}_i)\}_{i=1}^p$ through the channel, and the exact output is assembled by summation on the server side.

Defining $\mathcal{D} : \boldsymbol{x} \mapsto \boldsymbol{x} - \mathcal{C}(\boldsymbol{x})$, one can observe that $\mathrm{FCC}_p(\boldsymbol{x}) = \boldsymbol{x} - \mathcal{D}^p(\boldsymbol{x})$. In fact, the FCC module is able to refine the compression loss by harnessing

the contraction property of $\mathcal{D}$. Specifically, $\mathcal{D}$ is a contraction because $\|\mathcal{D}(\boldsymbol{x})\|^2 \leq (1-\mu)\|\boldsymbol{x}\|^2$ due to Definition 2.6. Hence, the error of the FCC module $\|\boldsymbol{x} - \mathrm{FCC}_p(\boldsymbol{x})\|^2 \leq (1-\mu)^p \|\boldsymbol{x}\|^2$ enjoys a geometric decay with p.

3.2 PowerEF-SGD

We integrate the FCC module into our algorithm PowerEF-SGD, as summarized in Algorithm 1. The algorithm takes as input an initial model $\boldsymbol{x}_0$, step size η, FCC parameter p, perturbation radius r, and the number of iterations T. After a simple initialization procedure, PowerEF-SGD iteratively produces a sequence $\{\boldsymbol{x}_t\}_{t=0}^T$ to gradually update the initial model by SGD-type descent. In each iteration, we use the accumulated gradient $\widetilde{\nabla}_p f_i$ to balance the number of communication rounds and stochastic gradient complexity. Each iteration of PowerEF-SGD contains four conceptual stages interpreted as follows.

- **Feedback the local gradient estimate.** We intend to use $\boldsymbol{e}_t^{(i)}$, the error up to the *last* iteration, to feedback our estimate of the local gradient $\boldsymbol{g}_t^{(i)}$ for the *current* round. Firstly, based on the error, the client invokes FCC module to compute the feedback term $\boldsymbol{w}_t^{(i)} + \boldsymbol{c}_t^{(i)}$ (Lines 9–10). Then each client i gets its current gradient estimate $\boldsymbol{g}_t^{(i)}$ by complementing the existing estimate $\boldsymbol{g}_{t-1}^{(i)}$ with the feedback term (Line 11).
- **Update the error.** Upon completion of the feedback, we increase the error term by the discrepancy between the real stochastic gradient (after artificial perturbation) and our local estimate $\boldsymbol{g}_t^{(i)}$ (Line 12). In this way, the error term essentially stores the *cumulative* estimation discrepancy of $\boldsymbol{g}_t^{(i)}$, which is ready for feedback again on the next run.
- **Prepare the global gradient estimate.** The update of global gradient estimate is conducted on a par with the local update method in an averaged manner (Line 16), so that we always have $\boldsymbol{g}_t = \frac{1}{n}\sum_{i=1}^n \boldsymbol{g}_t^{(i)}$.
- **Update the model.** Finally, the server updates the current model $\boldsymbol{x}_t$ by a descending step along our global gradient estimate $\boldsymbol{g}_t$ (Line 17).

3.3 Discussion

General design. At its core, PowerEF-SGD benefits from the power contraction underlying the FCC module to upgrade the classical error-feedback mechanism (Avdiukhin and Yaroslavtsev, 2021; Stich et al., 2018), hence the name. Specifically, our algorithm inherits the classical design of error term to track the cumulative discrepancy of gradient estimation (Line 12), but refines the way errors are used to feedback the current gradient estimation by the FCC module. Technically, the design of $\boldsymbol{w}_t^{(i)}$ and $\boldsymbol{c}_t^{(i)}$ helps us connect PowerEF-SGD steps towards Definition 2.6, giving rise to a recurrence on $\boldsymbol{e}_t^{(i)}$ which can be manipulated by theory. Moreover, while still guaranteeing second-order results, PowerEF-SGD manages to remove from the prior work (Avdiukhin and Yaroslavtsev, 2021) an expensive procedure of conditional reset that inevitably occupies the uncompressed bandwidth.

Data heterogeneity. Mathematically, our mechanism is able to induce an error term recurrence irrelevant to local gradients, thus circumventing from data similarity assumptions. This favorable property originates from our design of PowerEF-SGD, which is nontrivially different from the existing NEOLITHIC (Huang et al., 2022) algorithm where FCC module also plays a part. For example, NEOLITHIC inputs the gradient estimate to FCC while we input the estimation discrepancy, and error terms are also computed distinctly. As a notable result, contrary to our algorithm, the theory of NEOLITHIC still has to assume local gradient similarity.

Gradient perturbation. We add an isotropic Gaussian noise to each stochastic gradient to help the model escape from saddle points. Intuitively, around saddle points, the isotropic perturbation ensures that the SGD trajectory can traverse a sufficient distance along the descending direction, i.e. the eigenvector of Hessian $\nabla^2 f(\boldsymbol{x}_t)$ with a negative eigenvalue, thus escaping the saddle region and gaining an objective decrease. The perturbation is not required for first-order convergence, in which case one can safely set $r = 0$.

4 PERFORMANCE GUARANTEES

In this section, we state the theoretical guarantees for PowerEF-SGD, where the proofs are deferred to the appendix. To begin, we first define the first-order and second-order approximate stationarity conditions.

Definition 4.1. *$\boldsymbol{x} \in \mathbb{R}^d$ is said to be an ϵ-optimal first-order stationary point (ϵ-FOSP) if $\|\nabla f(\boldsymbol{x})\| \leq \epsilon$.*

Definition 4.2. *Suppose that $\boldsymbol{x} \in \mathbb{R}^d$ is an ϵ-FOSP. Then, $\boldsymbol{x}$ is said to be an ϵ-optimal second-order stationary point (ϵ-SOSP) if*

$$\nabla^2 f(\boldsymbol{x}) \succeq -\sqrt{\rho\epsilon}\cdot \boldsymbol{I}.$$

Otherwise, $\boldsymbol{x}$ is said to be an ϵ-strict saddle point.

Moreover, we denote $\chi^2 := \sigma^2 \log d + r^2$ the effective variance of stochastic gradient and perturbation, and

Algorithm 1 PowerEF-SGD

Input: $\boldsymbol{x}_0$, step size η, contraction exponent p, perturbation radius r, number of iterations T
Initialization: $\boldsymbol{e}_0^{(i)} \leftarrow \mathbf{0}$, $\boldsymbol{e}_{-1}^{(i)} \leftarrow \mathbf{0}$, $\boldsymbol{g}_{-1}^{(i)} \leftarrow \mathbf{0}$, $\boldsymbol{g}_{-1} \leftarrow \mathbf{0}$
for $t = 0, 1, 2, ..., T-1$ **do**
  for parameter server **do**
    sample $\boldsymbol{\xi}_t \sim \mathcal{N}(\mathbf{0}, \frac{r^2}{npd}\boldsymbol{I})$
    broadcast $\boldsymbol{\xi}_t$ to every client
  end for
  for client $i = 1, 2, ..., n$ in parallel **do**
    $\boldsymbol{w}_t^{(i)} \leftarrow \mathrm{FCC}_p(\boldsymbol{e}_t^{(i)} - \boldsymbol{e}_{t-1}^{(i)})$
    $\boldsymbol{c}_t^{(i)} \leftarrow \mathcal{C}(\boldsymbol{e}_t^{(i)} + \widetilde{\nabla}_p f_i(\boldsymbol{x}_t) + \boldsymbol{\xi}_t - \boldsymbol{g}_{t-1}^{(i)} - \boldsymbol{w}_t^{(i)})$
    $\boldsymbol{g}_t^{(i)} \leftarrow \boldsymbol{g}_{t-1}^{(i)} + \boldsymbol{w}_t^{(i)} + \boldsymbol{c}_t^{(i)}$ {Feedback the local gradient estimate}
    $\boldsymbol{e}_{t+1}^{(i)} \leftarrow \boldsymbol{e}_t^{(i)} + \widetilde{\nabla}_p f_i(\boldsymbol{x}_t) + \boldsymbol{\xi}_t - \boldsymbol{g}_t^{(i)}$ {Update the error}
    upload $\boldsymbol{c}_t^{(i)}$ and $\boldsymbol{w}_t^{(i)}$ (as a sum of p compressed vectors) to server
  end for
  for parameter server **do**
    $\boldsymbol{g}_t \leftarrow \boldsymbol{g}_{t-1} + \frac{1}{n}\sum_{i=1}^n \boldsymbol{w}_t^{(i)} + \frac{1}{n}\sum_{i=1}^n \boldsymbol{c}_t^{(i)}$ {Prepare the global gradient}
    $\boldsymbol{x}_{t+1} \leftarrow \boldsymbol{x}_t - \eta \boldsymbol{g}_t$ {Update the model}
    broadcast $\boldsymbol{x}_t$ to every client
  end for
end for

introduce the initialization quality

$$\Phi = \frac{1}{n}\sum_{i=1}^n \left\| \widetilde{\nabla}_p f_i(\boldsymbol{x}_0) + \boldsymbol{\xi}_0 \right\|^2 + \widetilde{L}[f(\boldsymbol{x}_0) - f_{\min}].$$

We are now ready to state the main theorems. Theorem 4.3 establishes that PowerEF-SGD converges with high probability to ϵ-FOSP.

Theorem 4.3 (Convergence to ϵ-FOSP)**.** *Suppose that Assumptions 2.1, 2.2, 2.5 hold, and the parameters T, η, p, r satisfy*

$$T = \kappa_T \cdot \max\left\{ \frac{f(\boldsymbol{x}_0) - f_{\min}}{\eta\epsilon^2}, \frac{\chi^2\iota}{np\epsilon^2} \right\},$$
$$\eta = \kappa_\eta \cdot \min\left\{ \frac{\mu}{\widetilde{L}}, \frac{\mu\epsilon}{L\sqrt{\mu\Phi + \frac{\chi^2\iota}{np}}}, \frac{np\epsilon^2}{\chi^2 L} \right\},$$
$$p = \kappa_p \cdot \frac{1}{\mu}\log\left(\frac{1}{\mu}\right)$$

for some constants $\kappa_T, \kappa_\eta, \kappa_p > 0$, and the parameter ι controlling the tightness of the probability bound. Then, with probability at least $1 - 7e^{-\iota}$, at least 3/4 of the iterates $\{\boldsymbol{x}_t\}_{t=0}^T$ generated by Algorithm 1 are ϵ-FOSPs.

In words, first-order convergence is guaranteed with high probability (controlled by ι), under an appropriate choice of the algorithm parameters. Note that the theorem does not specify a choice for the perturbation radius r, resonating with Section 3.3 in that perturbation is not required for first-order convergence.

Regarding the second-order convergence, we have the following theorem.

Theorem 4.4 (Convergence to ϵ-SOSP)**.** *Suppose that Assumptions 2.1*, 2.2, 2.3, 2.5 hold, and the parameters T, η, p, r satisfy*

$$T = \kappa_T \cdot \max\left\{ \frac{\iota^5 f_{\max}}{\eta\epsilon^2}, \frac{\chi^2\iota}{np\epsilon^2} \right\},$$
$$\eta = \kappa_\eta \cdot \min\left\{ \begin{matrix} \frac{\mu}{\widetilde{L}}, & \frac{\mu\epsilon}{\iota^5 L\sqrt{\mu\Phi + \frac{\chi^2\iota}{np}}}, \\ \frac{np\epsilon^2}{\iota^5\chi^2 L}, & \frac{\iota\sigma^2\sqrt{\rho\epsilon}\log d}{L^2\left(np\Phi + \frac{\chi^2\iota}{\mu^2}\right)} \end{matrix} \right\},$$
$$p = \kappa_p \cdot \frac{1}{\mu}\log\left(\frac{1}{\mu}\right),$$
$$r = \kappa_r \cdot \sigma\sqrt{\iota d \log d}$$

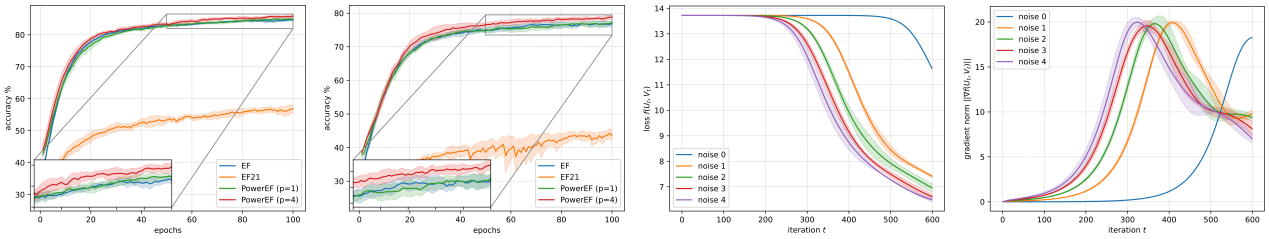
for some constants $\kappa_T, \kappa_\eta, \kappa_p, \kappa_r > 0$, and the parameter ι controlling the tightness of the probability bound. Set $\mathcal{I} = \frac{\iota}{\eta\sqrt{\rho\epsilon}}$. Then, with probability at least $1 - 8T^2(\mathcal{I}^2 + d\mathcal{I} + \mathcal{I} + T)e^{-\iota}$, at least half of the iterates $\{\boldsymbol{x}_t\}_{t=0}^T$ generated by Algorithm 1 are ϵ-SOSPs.

Unlike Theorem 4.3, perturbing the local stochastic gradient with an appropriate radius plays a vital part in the second-order guarantee by assisting the iteration to escape the saddle points.

Based on Theorem 4.3 and 4.4, it is now immediate to compute the gradient complexity and communication rounds of PowerEF-SGD to attain first-order and second-order optimality, respectively.

Table 3: Test accuracy achieved by different algorithms at epoch 100 under two levels of data heterogeneity.

algorithm	EF	EF21	PowerEF$_{p=1}$	PowerEF$_{p=4}$
heterogeneity level ℓ_1	84.58 ± 0.43	56.80 ± 1.26	84.87 ± 0.32	$\mathbf{85.85 \pm 0.40}$
heterogeneity level ℓ_2	76.91 ± 0.85	43.56 ± 1.62	77.18 ± 0.59	$\mathbf{78.86 \pm 0.65}$



(a) ℓ_1 test accuracy (b) ℓ_2 test accuracy (c) loss (d) gradient norm

Figure 1: (a–b) Test accuracy curves of different algorithms in CIFAR-10 training tasks under two heterogeneity levels. (c–d) Loss and gradient norm curves in the synthetic experiments.

Corollary 4.5 (ϵ-FOSP complexity)**.** *Under the same setting of Theorem 4.3, Algorithm 1 requires* $\widetilde{O}\Big(\frac{1}{n\epsilon^4} + \frac{1}{\mu^{1.5}\epsilon^3} + \frac{1}{\mu^2\epsilon^2}\Big)$ *queries to the stochastic gradient oracle and communication rounds.*

Corollary 4.6 (ϵ-SOSP complexity)**.** *Under the same setting of Theorem 4.4, Algorithm 1 requires* $\widetilde{O}\Big(\frac{1}{n\epsilon^4} + \frac{1}{\mu^{1.5}\epsilon^3} + \frac{\mu n+1}{\mu^3\epsilon^{2.5}}\Big)$ *queries to the stochastic gradient oracle and communication rounds.*

According to the corollaries, PowerEF-SGD improves to second-order stationary points after visiting first-order (possibly saddle) points, using additional gradient queries and communication rounds only of almost the same order required by first-order convergence when ϵ is typically small to be the dominant parameter. Contrary to another work allowing heterogeneous data (Fatkhullin et al., 2021), our convergence rate exhibits a linear speedup in terms of n, implying that our algorithm significantly benefits from the distributed framework.

5 EXPERIMENTS

In this section, we conduct two types of experiments to evaluate the performance of PowerEF-SGD in heterogeneous FL tasks, and its capability of escaping saddle points using the gradient perturbation technique.

5.1 Heterogeneous federated learning

We experiment on distributed deep learning tasks with heterogeneous data to demonstrate the advantage of PowerEF-SGD over the baseline algorithms EF (Avdiukhin and Yaroslavtsev, 2021) and EF21 (Richtárik et al., 2021).

We train a ResNet18 model on CIFAR10 dataset (Krizhevsky and Hinton, 2009) using 4 clients and 1 server, optimized by different methods. To simulate heterogeneity, we sample local data from CIFAR10 so that the local distributions of y-class are imbalanced and heterogeneous across workers. We experiment on 2 heterogeneity levels: (ℓ_1) min class size / max class size ≈ 0.08; (ℓ_2) min class size / max class size ≈ 0.01. In both settings, we train EF, EF21, PowerEF$_{p=1}$ and PowerEF$_{p=4}$ 3 times each, deploying Top-k compressor that keeps top 1% coordinates of the largest magnitudes. All the training procedures take 100 epochs with a step size of 10^{-2} and weight decay of 10^{-4}. The algorithms are implemented on PyTorch (Paszke et al., 2019) 2.0.0 and the experiments are conducted on NVIDIA Tesla P100 GPU.

In Figure 1(a–b) we plot the test accuracy curves of each algorithm in both settings, and the accuracy at the final epoch is reported in Table 3 for comparison. Although the increase of heterogeneity level hinders the performance of each algorithm, it is clear that PowerEF$_{p=4}$ consistently outperforms the baselines in both tasks, and an increase of FCC contraction p effectively facilitates learning when heterogeneity is present.

5.2 Escaping saddle points

We conduct synthetic experiments to show the saddle-escaping property of PowerEF-SGD. Consider the fol-

lowing finite-sum matrix factorization problem

$$f(\boldsymbol{U},\boldsymbol{V})=\frac{1}{n}\sum_{i=1}^{n}f_i(\boldsymbol{U},\boldsymbol{V})=\frac{1}{n}\sum_{i=1}^{n}\|\boldsymbol{M}_i-\boldsymbol{U}\boldsymbol{V}^\top\|_F^2,$$

where $\boldsymbol{M}_i$ are given data matrices generated from Gaussian distribution. Initialized around a saddle point, PowerEF-SGD is tested under 5 different levels of gradient perturbation variance, starting from level 0 (no noise). We experiment on each noise level 3 times, and plot the curve of loss and gradient norm with respect to the number of iterations t in Figure 1(c–d). Increasing the perturbation level, it takes shorter time to observe the drop of loss and increase of gradient norm, suggesting that the algorithm can escape from saddle points more efficiently with the help of gradient perturbation.

6 CONCLUSION

In this paper, we propose and analyze PowerEF-SGD, which is the first distributed SGD algorithm with communication compression that provably attains second-order optimality under heterogeneous data, to the best of our knowledge. Specifically, subject to mild and standard assumptions, we show that PowerEF-SGD converges to ϵ-SOSPs with high probability, which is almost on par with the gradient and communication complexity it takes to find ϵ-FOSPs, and the convergence rate shows a linear speedup with respect to n. Our theory is complemented by the performance of PowerEF-SGD in the distributed learning experiments. For future work, it will be of great interest to develop privacy-preserving distributed SGD algorithms that can escape saddle points with communication compression.

Acknowledgements

This work is supported in part by NSF under the grants CCF-2007911, ECCS-2318441, CNS-2148212, by funds from federal agency and industry partners as specified in the NSF Resilient & Intelligent NextG Systems (RINGS) program, and by AFRL under the grant FA8750-20-2-0504.

References

Alistarh, D., Grubic, D., Li, J., Tomioka, R., and Vojnovic, M. (2017). QSGD: Communication-efficient SGD via gradient quantization and encoding. In *Advances in Neural Information Processing Systems*, pages 1709–1720.

Avdiukhin, D. and Yaroslavtsev, G. (2021). Escaping saddle points with compressed sgd. In *Advances in Neural Information Processing Systems*, volume 34, pages 10273–10284. Curran Associates, Inc.

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. PMLR.

Beznosikov, A., Horváth, S., Richtárik, P., and Safaryan, M. (2020). On biased compression for distributed learning. *arXiv preprint arXiv:2002.12410*.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. (2020). Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Daneshmand, H., Kohler, J., Lucchi, A., and Hofmann, T. (2018). Escaping saddles with stochastic gradients. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 1155–1164. PMLR.

Dauphin, Y. N., Pascanu, R., Gulcehre, C., Cho, K., Ganguli, S., and Bengio, Y. (2014). Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. *Advances in neural information processing systems*, 27.

Du, S. S., Jin, C., Lee, J. D., Jordan, M. I., Singh, A., and Poczos, B. (2017). Gradient descent can take exponential time to escape saddle points. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Fatkhullin, I., Sokolov, I., Gorbunov, E., Li, Z., and Richtárik, P. (2021). EF21 with bells & whistles: Practical algorithmic extensions of modern error feedback. *arXiv preprint arXiv:2110.03294*.

Ge, R., Huang, F., Jin, C., and Yuan, Y. (2015). Escaping from saddle points — online stochastic gradient for tensor decomposition. In *Conference on Learning Theory*, pages 797–842.

Ge, R., Li, Z., Wang, W., and Wang, X. (2019). Stabilized SVRG: Simple variance reduction for nonconvex optimization. In *Conference on Learning Theory*, pages 1394–1448.

Ghadimi, S., Lan, G., and Zhang, H. (2016). Mini-batch stochastic approximation methods for nonconvex stochastic composite optimization. *Mathematical Programming*, 155(1-2):267–305.

Huang, X., Chen, Y., Yin, W., and Yuan, K. (2022). Lower bounds and nearly optimal algorithms in distributed learning with communication compression. In Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K., and Oh, A., editors, *Advances in Neural Information Processing Systems*, volume 35, pages 18955–18969. Curran Associates, Inc.

Jin, C., Ge, R., Netrapalli, P., Kakade, S. M., and Jordan, M. I. (2017). How to escape saddle points efficiently. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1724–1732. JMLR. org.

Jin, C., Netrapalli, P., Ge, R., Kakade, S. M., and Jordan, M. I. (2021). On nonconvex optimization for machine learning: Gradients, stochasticity, and saddle points. *Journal of the ACM (JACM)*, 68(2):1–29.

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*.

Kang, J., Xiong, Z., Niyato, D., Zou, Y., Zhang, Y., and Guizani, M. (2020). Reliable federated learning for mobile networks. *IEEE Wireless Communications*, 27(2):72–80.

Karimireddy, S. P., Rebjock, Q., Stich, S., and Jaggi, M. (2019). Error feedback fixes signsgd and other gradient compression schemes. In *International Conference on Machine Learning*, pages 3252–3261. PMLR.

Koloskova, A., Lin, T., Stich, S. U., and Jaggi, M. (2019a). Decentralized deep learning with arbitrary communication compression. In *International Conference on Learning Representations*.

Koloskova, A., Loizou, N., Boreiri, S., Jaggi, M., and Stich, S. (2020). A unified theory of decentralized SGD with changing topology and local updates. In *International Conference on Machine Learning*, pages 5381–5393. PMLR.

Koloskova, A., Stich, S., and Jaggi, M. (2019b). Decentralized stochastic optimization and gossip algorithms with compressed communication. In *International Conference on Machine Learning*, pages 3478–3487. PMLR.

Konečný, 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*.

Krizhevsky, A. and Hinton, G. (2009). Learning multiple layers of features from tiny images.

Lee, J. D., Simchowitz, M., Jordan, M. I., and Recht, B. (2016). Gradient descent only converges to minimizers. In *29th Annual Conference on Learning Theory*, volume 49 of *Proceedings of Machine Learning Research*, pages 1246–1257, Columbia University, New York, New York, USA. PMLR.

Li, B. and Chi, Y. (2023). Convergence and privacy of decentralized nonconvex optimization with gradient clipping and communication compression. *arXiv preprint arXiv:2305.09896*.

Li, Z. (2019). SSRGD: Simple stochastic recursive gradient descent for escaping saddle points. In *Advances in Neural Information Processing Systems*, pages 1523–1533.

Li, Z., Bao, H., Zhang, X., and Richtárik, P. (2021). PAGE: A simple and optimal probabilistic gradient estimator for nonconvex optimization. In *International Conference on Machine Learning*, pages 6286–6295. PMLR.

Li, Z., Zhao, H., Li, B., and Chi, Y. (2022). SoteriaFL: A unified framework for private federated learning with communication compression. *Advances in Neural Information Processing Systems*, 35:4285–4300.

Nesterov, Y. (2004). *Introductory Lectures on Convex Optimization: A Basic Course*. Kluwer.

Nguyen, D. C., Ding, M., Pathirana, P. N., Seneviratne, A., Li, J., and Poor, H. V. (2021). Federated learning for internet of things: A comprehensive survey. *IEEE Communications Surveys & Tutorials*, 23(3):1622–1658.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., and Antiga, L. (2019). PyTorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, pages 8024–8035.

Richtárik, P., Sokolov, I., and Fatkhullin, I. (2021). EF21: A new, simpler, theoretically better, and practically faster error feedback. *arXiv preprint arXiv:2106.05203*.

Richtárik, P., Sokolov, I., Gasanov, E., Fatkhullin, I., Li, Z., and Gorbunov, E. (2022). 3PC: Three point compressors for communication-efficient distributed training and a better theory for lazy aggregation. In *International Conference on Machine Learning*, pages 18596–18648. PMLR.

Savazzi, S., Nicoli, M., and Rampa, V. (2020). Federated learning with cooperating devices: A consensus approach for massive IoT networks. *IEEE Internet of Things Journal*, 7(5):4641–4654.

Seide, F., Fu, H., Droppo, J., Li, G., and Yu, D. (2014). 1-bit stochastic gradient descent and its application

to data-parallel distributed training of speech dnns. In *Fifteenth annual conference of the international speech communication association*.

Stich, S. U., Cordonnier, J.-B., and Jaggi, M. (2018). Sparsified SGD with memory. *Advances in Neural Information Processing Systems*, 31:4447–4458.

Xie, C., Zheng, S., Koyejo, S., Gupta, I., Li, M., and Lin, H. (2020). Cser: Communication-efficient sgd with error reset. In *Advances in Neural Information Processing Systems*, volume 33, pages 12593–12603. Curran Associates, Inc.

Xu, J., Glicksberg, B. S., Su, C., Walker, P., Bian, J., and Wang, F. (2021). Federated learning for healthcare informatics. *Journal of Healthcare Informatics Research*, 5(1):1–19.

Zhao, H., Li, B., Li, Z., Richtárik, P., and Chi, Y. (2022). BEER: Fast $O(1/T)$ rate for decentralized nonconvex optimization with communication compression. In *Advances in Neural Information Processing Systems*.

Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. [Yes]
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. [Yes]
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. [Not Applicable]

2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results. [Yes]
   (b) Complete proofs of all theoretical results. [Yes]
   (c) Clear explanations of any assumptions. [Yes]

3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). [Yes]
   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). [Yes]
   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). [Yes]
   (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). [Yes]

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:
   (a) Citations of the creator If your work uses existing assets. [Yes]
   (b) The license information of the assets, if applicable. [Not Applicable]
   (c) New assets either in the supplemental material or as a URL, if applicable. [Not Applicable]
   (d) Information about consent from data providers/curators. [Not Applicable]
   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. [Not Applicable]

5. If you used crowdsourcing or conducted research with human subjects, check if you include:
   (a) The full text of instructions given to participants and screenshots. [Not Applicable]
   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. [Not Applicable]
   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. [Not Applicable]