Rethinking gradient sparsification as total error minimization

Atal Narayan Sahu KAUST
Aritra Dutta KAUST
Ahmed M. Abdelmoniem KAUST
Trambak Banerjee University of Kansas
Marco Canini KAUST
Panos Kalnis KAUST

Abstract

Gradient compression is a widely-established remedy to tackle the communication bottleneck in distributed training of large deep neural networks (DNNs). Under the error-feedback framework, Top-k sparsification, sometimes with k as little as 0.1% of the gradient size, enables training to the same model quality as the uncompressed case for a similar iteration count. From the optimization perspective, we find that Top-k is the communication-optimal sparsifier given a per-iteration k element budget. We argue that to further the benefits of gradient sparsification, especially for DNNs, a different perspective is necessary — one that moves from per-iteration optimality to consider optimality for the entire training.

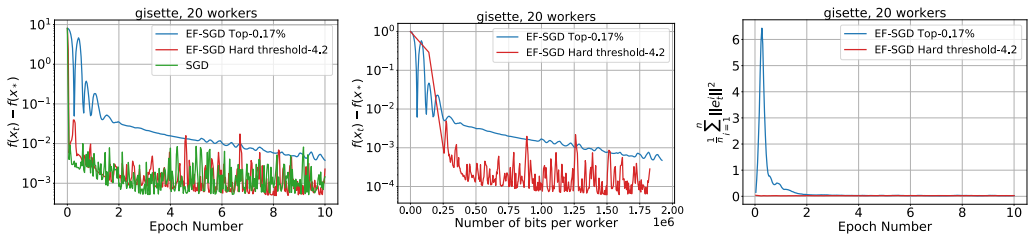
We identify that the *total error* — the sum of the compression errors for all iterations — encapsulates sparsification throughout training. Then, we propose a communication complexity model that minimizes the total error under a communication budget for the entire training. We find that the *hard-threshold sparsifier*, a variant of the Top-k sparsifier with k determined by a constant hard-threshold, is the optimal sparsifier for this model. Motivated by this, we provide convex and non-convex convergence analyses for the hard-threshold sparsifier with error-feedback. We show that hard-threshold has the same asymptotic convergence and linear speedup property as SGD in both the case, and unlike with Top-k sparsifier, has no impact due to data-heterogeneity. Our diverse experiments on various DNNs and a logistic regression model demonstrate that the hard-threshold sparsifier is more communication-efficient than Top-k. Code is available at `https://github.com/sands-lab/rethinking-sparsification`.

1 Introduction

With the emergence of huge DNNs consisting of hundreds of millions to billions of parameters [12, 50], distributed data-parallel training [66] is an increasingly important workload. As the training process typically spans several compute nodes (or workers) that periodically exchange the local gradient vectors at each iteration of the optimizer (e.g., SGD), communication among nodes remains in many cases the main performance bottleneck [32, 40, 46].

Lossy gradient compression techniques are becoming a common approach to rein in communication efficiency [62]. In particular, sparsification, which sends only a subset of gradient coordinates (e.g., Top-k [4, 8] sends the k largest gradient coordinates by magnitude in each iteration), may significantly reduce data volumes and thus speed up training. However, due to its lossy nature, compression raises a complex trade-off between training performance and accuracy. For instance, Agarwal et al. [3] note that training ResNet-18 on CIFAR-100 using sparsification speeds up training significantly ($3.6\times$), but it also degrades final accuracy by 1.5%. On the other hand, Lin et al. [37] reports a $500\times$ data

35th Conference on Neural Information Processing Systems (NeurIPS 2021).



(a) (b) (c)

Figure 1: Convergence of Top-k and Hard-threshold for a logistic regression model on `gisette` LIBSVM dataset with 20 workers: (a) Functional suboptimality vs. epochs; (b) functional suboptimality vs. bits communicated; (c) error norm vs. epochs. Hard-threshold converges as fast as the baseline, no compression SGD and much faster than Top-k because of a smaller total-error than Top-k.

reduction via sparsification under deep gradient compression (DGC) for ResNet-50 on ImageNet while preserving the same final accuracy when adopting a carefully-tuned warmup phase.

The vast literature on gradient compression largely considers a fixed communication budget per iteration while leaving it up to practitioners to grapple with specifying an additional hyper-parameter that determines the degree of compression before training begins. Meanwhile, recent adaptive Top-k sparsifiers [3, 65] empirically demonstrate that tuning the degree of compression in different phases of DNN training yields a more communication-efficient scheme than a fixed compression scheme (e.g., a static k for Top-k). However, these works lack a theoretical framework proving that adaptive compression enjoys better convergence guarantees than the fixed compression scheme.

This raises a fundamental question: *Given a fixed communication budget, is there a provably better communication scheme than fixed per-iteration compressed communication?* We first observe that Top-k is the communication-optimal sparsifier for a fixed per-iteration communication budget (§4.3). Then, our insight is that by adopting a different perspective that accounts for *the effect of sparsification throughout training*, a more efficient communication scheme is possible under a revised notion of optimality that considers an overall communication budget (instead of a per-iteration budget).

We consider sparsification by using the *error-feedback* (EF) mechanism [8, 53], a delayed gradient component update strategy that is instrumental for the convergence of the state-of-the-art sparsifiers [10]. Let e_t denote the error arising due to sparsification at iteration t. In EF, this error is added to the gradient update at iteration $t+1$. We identify that the term affecting the non-convex convergence in EF-SGD is the *total-error:* $\sum_t \|e_t\|^2$ [33, 54].

Directly minimizing the total-error is not possible; thus, Top-k minimizes $\|e_t\|^2$ at each iteration. We argue that it is possible to focus on the *sum of* $\|e_t\|^2$ and devise a communication scheme that achieves a smaller total-error than any fixed communication sparsifier. We demonstrate that to achieve this change of perspective; it is sufficient to consider a practical yet straightforward mechanism that is a natural counterpart of Top-k: the *hard-threshold sparsifier*, which communicates the gradient coordinates with magnitude greater than or equal to a fixed given threshold, $\lambda \geq 0$, in each iteration. Although the two sparsifiers are in an equivalence relation (a given λ corresponds to a k), under the total-error minimization perspective, we adopt a *fixed* threshold, λ, which implies a *variable* k at every iteration.

To illustrate intuitively why this change of perspective yields benefits, consider the following example. Figure 1 shows an experiment in the distributed setting where 20 workers train a 6,000-parameter logistic regression model on the `gisette` LIBSVM dataset [14] by using the Top-k and hard-threshold sparsifiers, configured to send the *same data volume*.[1] The loss function is strongly convex and has a unique minimizer, $x^\star$, therefore, a unique optimum, $f(x^\star)$. We see that hard-threshold converges at the same speed as SGD while communicating $\sim 600\times$ less data, whereas Top-k has a significantly slower convergence speed. We attribute this to the fact that Top-k has a large error accumulation in the initial 500 iterations, while the error magnitude for hard-threshold is less than 0.04 throughout training (cf. Figure 1c). Our results with DNN training also reflect this insight (§6).

[1] We train for 10 epochs and set $k = 0.17\%$ for Top-k, and $\lambda = 4.2$ for hard-threshold.

Moreover, the hard-threshold sparsifier has computational benefits over Top-k sparsifier, as hard-threshold's underlying filtering operation requires d comparisons in each iteration, where d is the number of parameters. In contrast, Top-k is a compute-intensive sparsifier (e.g., on CPU, the computational complexity is $\mathcal{O}(d \log_2 k)$ [48]). For GPUs, several optimized implementations are proposed but they rely on the data distribution and are efficient only for a small k [48]. For instance, PyTorch uses Radix select algorithm [5] which has a computational complexity of $\mathcal{O}\left(\lceil b/r \rceil\ d\right)$ where b is the number of bits to represent gradient values and r is the radix size [42].

Finally, while the hard-threshold sparsifier already exists in the literature [20, 55], we are the first to formally study it and theoretically demonstrate its benefits as an adaptive counterpart of Top-k. Moreover, our argument in favor of hard-threshold precisely falsifies the claim by Dryden et al. [18] that stopped its widespread adoption — *a hard-threshold may lead to a degenerate situation when the EF in gradient compression builds up.*

This paper makes the following contributions:

Communication complexity model (§4). We propose a communication complexity model that captures the effects of compression in the entire optimization process. We allow for variable communication in each iteration by only imposing a total communication budget. We show that the hard-threshold sparsifier is the communication-optimal sparsifier in this model.

Absolute compressors (§5). We identify that the hard-threshold sparsifier, along with other existing compressors [16, 46], belongs to the class of *absolute compressors*, which have an absolute bound on the error. Absolute compressors have not been formally studied before with EF. We show that absolute compressors with EF converge for both strongly convex and non-convex loss functions. In both cases, similar to the δ-contraction operators [33], absolute compressors enjoy the same asymptotic convergence with linear speedup (with respect to the number of workers) as no-compression SGD. However, δ-contraction operators have a worse dependence on δ in the distributed setting with heterogeneous data, while absolute compressors do not have such an anomaly.

Experiments (§6). We conduct diverse experiments on both strongly convex and non-convex (for DNNs) loss functions to substantiate our claims. Our DNN experiments include computer vision, language modeling, and recommendation tasks, and our strongly convex experiment is on logistic regression. We find that the hard-threshold sparsifier is consistently more communication-efficient than the Top-k sparsifier given the same communication budget.

2 Related work

Gradient compression techniques can be broadly classified into quantization [7, 16, 33, 47, 60], sparsification [4, 37, 59], hybrid compressors [9, 18, 55], and low-rank methods [57, 58]. The state-of-the-art compressors are biased δ-contraction operators [37, 57], see §4.5. We refer to [62] for a recent survey and quantitative evaluation of these techniques.

Error-feedback (EF) or memory was first empirically used in [47, 55]. However, [33, 53, 54] were the first to give a convergence analysis of the EF framework, which was extended to the distributed setup in [10, 64]. Recently, [61] proposed *error-reset*, a different form of EF, while [29] introduced another alternative by communicating a compressed version of the error. EF has also been combined with variance-reduction [22, 43] and acceleration [44].

Communication-optimal compression. [6, 15, 21, 45] devised a communication-optimal compressor by minimizing the worst-case compression factor[2] under a per-vector communication budget.

Adaptive compression. [59] designed an adaptive sparsifier that minimizes expected sparsity of the compressed vector under a given variance budget. While AdaQS [23] periodically doubles the quantization states in QSGD [7] to reduce the compression factor, DQSGD [63] sets the number of quantization states proportional to the gradient norm. ACCORDION [3] chooses a low compression ratio if training is in a critical regime [2], and a high compression ratio otherwise.

[2]For a vector x and a possibly randomized compression operator $\mathcal{C}$, we denote the compression error as $\mathbb{E}_{\mathcal{C}}[\|x - \mathcal{C}(x)\|^2]$, and compression factor as $\mathbb{E}_{\mathcal{C}}[\|x - \mathcal{C}(x)\|^2]/\|x\|^2$.

Efficient Top-k estimation is a focus of many recent works. While [37] estimates the Top-k threshold on a randomly sampled subset, [1, 49] estimate it by fitting parametric statistical distributions. [25] estimates the threshold every $1,000$ iterations. These works determine a threshold to approximately determine a *fixed* Top-k set, while we use a hard-threshold to determine the k in each iteration.

3 Background on Error-Feedback (EF) SGD

Consider the following distributed optimization problem with n workers:

$$\min_{x\in\mathbb{R}^d} f(x) \coloneqq \frac{1}{n}\sum_{i\in[n]} f_i(x), \tag{1}$$

where $f_i(x) = \mathbb{E}_{z_i\sim\mathcal{D}_i} l(x; z_i)$ denotes the loss function evaluated on input z_i sampled from $\mathcal{D}_i$, the data-distribution at the i^{th} worker. Let $g_{i,t}$ denote the stochastic gradient computed at i^{th} worker at iteration t such that $g_{i,t} = \nabla f_i(x_t) + \xi_{i,t}$ with $\mathbb{E}[\xi_{i,t}|x_t] = \mathbf{0}$.

Algorithm 1 gives the pseudo-code of compressed EF SGD [8, 33, 53] to solve (1). Let $e_{i,t}$ denote the locally accumulated error at i^{th} worker due to compression from previous steps. Adding this error to the current gradient, $\gamma_t g_{i,t}$ provides the corrected update, $p_{i,t}$, where $\gamma_t > 0$ is the step-size. This corrected update is further compressed and exchanged with other machines, and the local error is updated for the next step.

Algorithm 1: Distributed EF SGD

Input: $\mathcal{C}$-compressor
Initialize $x_0 \in \mathbb{R}^d$;
for *worker* $i \in [n]$ **in parallel do**
  Initialize error $e_{i,0} = 0$;
  for *iteration* $t = 0, \dots, T-1$ **do**
    $p_{i,t} \leftarrow e_{i,t} + \gamma_t g_{i,t}$ `/* Incorporate EF into update */;`
    $\Delta_{i,t} \leftarrow \gamma_t \mathcal{C}(\frac{p_{i,t}}{\gamma_t})$;
    $e_{i,t+1} \leftarrow p_{i,t} - \Delta_{i,t}$ `/* Update error */;`
    $\bar{\Delta}_t = \frac{1}{n}\sum_{i\in[n]} \Delta_{i,t}$ `/* Exchange and average` $\Delta_{i,t}$ `among workers */;`
    $x_{t+1} \leftarrow x_t - \bar{\Delta}_t$;

Remark 1. In Algorithm 1, we have $\Delta_{i,t} \leftarrow \gamma_t \mathcal{C}(\frac{p_{i,t}}{\gamma_t})$, while [8, 33, 53] consider $\Delta_{i,t} \leftarrow \mathcal{C}(p_{i,t})$. We do this to extend the EF framework for absolute compressors. We note that Algorithm 1 is more general as $\gamma_t \mathcal{C}(\frac{p_{i,t}}{\gamma_t})$ is equivalent to $\mathcal{C}(p_{i,t})$ for all known δ-contraction operators.

3.1 Assumptions

We consider the following general assumptions on the loss function.

Assumption 1. ***(Smoothness)*** *The function, $f_i : \mathbb{R}^d \to \mathbb{R}$ at each worker, $i \in [n]$ is L-smooth, i.e., for every $x, y \in \mathbb{R}^d$ we have, $f_i(y) \le f_i(x) + \langle \nabla f_i(x), y - x\rangle + \frac{L}{2}\|y - x\|^2$.*

Assumption 2. ***(Global minimum)*** *There exists $x^\star$ such that, $f(x^\star) = f^\star \le f(x)$, for all $x \in \mathbb{R}^d$.*

Assumption 3. ***(M, σ^2) bounded noise)*** *[54] For every stochastic noise $\xi_{i,t}$, there exist $M, \sigma^2 \ge 0$, such that $\mathbb{E}[\|\xi_{i,t}\|^2 \mid x_t] \le M\|\nabla f_i(x_t)\|^2 + \sigma^2$, for all $x_t \in \mathbb{R}^d$.*

Remark 2. The above assumption implies $\mathbb{E}[\|g_{i,t}\|^2 \mid x_t] \le (M+1)\|\nabla f_i(x_t)\|^2 + \sigma^2$. This general noise model does not uniformly bound the second moment of stochastic gradients as in [9, 33, 64].

Assumption 4. ***(C, ζ^2) bounded similarity)*** *The variance of gradients among workers is bounded, i.e., there exist constants, $C, \zeta \ge 0$ such that, $\frac{1}{n}\sum_{i\in[n]}\|\nabla f_i(x) - \nabla f(x)\|^2 \le C\|\nabla f(x)\|^2 + \zeta^2$, for all $x \in \mathbb{R}^d$.*

Remark 3. If all the workers have the same training data, all f_i are the same, resulting in $C, \zeta = 0$. This assumption is an extension from [31, 36], which consider $C = 0$.

For the *convergence of strongly convex functions*, we require an additional assumption as follows.

Assumption 5. ***(μ-strong convexity)*** *The functions, $f_i : \mathbb{R}^d \to \mathbb{R}$ are μ-strongly convex, i.e., there exists $\mu \ge 0$, such that $f_i(y) \ge f_i(x) + \langle \nabla f_i(x), y - x\rangle + \frac{\mu}{2}\|x - y\|^2$, for all $x \in \mathbb{R}^d$.*

Convergence of EF-SGD. The following result shows the convergence of EF-SGD [33] in minimizing general smooth functions for a single worker ($n = 1$) case.

Theorem 1. *[33, 54] Let Assumption 1, 2 and 3 hold. Then Algorithm 1 with a constant step-size, γ where $\gamma \leq \frac{1}{2L(M+1)}$ and $n = 1$ follows*

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\|\nabla f(x_t)\|^2 \leq \frac{4(f(x_0)-f^\star)}{\gamma T} + 2\gamma L\sigma^2 + 2L^2\sum_{t=0}^{T-1}\frac{\gamma^2\mathbb{E}\|g_t + \frac{e_t}{\gamma} - \mathcal{C}(g_t + \frac{e_t}{\gamma})\|^2}{T}.$$

Remark 4. Theorem 1 is a simplified version of the distributed case for $n > 1$ and quoted to emphasize the effect of compression between the error-corrected gradient, $g_t + \frac{e_t}{\gamma}$, and its compressed form, $\mathcal{C}(g_t + \frac{e_t}{\gamma})$. The term that *solely accounts for the effect of compression in the entire training process* is the *total-error*: $\sum_{t=0}^{T-1}\mathbb{E}\|e_{t+1}\|^2 = \sum_{t=0}^{T-1}\gamma^2\mathbb{E}\|g_t + \frac{e_t}{\gamma} - \mathcal{C}(g_t + \frac{e_t}{\gamma})\|^2$.

4 A communication complexity perspective to sparsification

We now propose a communication complexity model and contrast it with the existing communication-optimal strategies. We note that our use of the word *communication-optimal* is in the sense of optimization upper-bounds. Convergence analyses of compressed SGD capture the effect of compression via the compression error, and this effect is always inverse — the lower the compression error, the better the optimization upper bound [22, 29]. Therefore, ours and the existing works [6, 15, 21, 45] design *communication-optimal* strategies by optimizing the compression error related term. We start with a sparse approximation problem that we encounter in our subsequent discussions.

4.1 A sparse approximation problem

Let $p \in \mathbb{R}^m$ be a given vector. We want to approximate p with a sparse vector, q, that has at most $0 < \tau \leq m$ non-zero elements. Formally, we write the *constrained sparse approximation* problem as:

$$q^\star = \arg\min_{q\in\mathbb{R}^m}\|p-q\|^2 \quad \text{subject to } \|q\|_0 \leq \tau, \tag{2}$$

where $\|\cdot\|_0$ denotes the number of non-zero elements in a vector. Problem (2) and its variants are well studied and arise in signal processing [13, 17, 19] and matrix approximation [11, 51].

Lemma 1. *The solution $q^\star$ to* (2) *is obtained by keeping Top-τ magnitude entries from p and setting the rest to zeros.*

4.2 Minimizing the total-error is not possible

Let $\mathfrak{C}$ denote the class of all compressors. We constrain to the class of deterministic sparsifiers, denoted by $\mathbf{S} \subset \mathfrak{C}$, but one can similarly consider other subclasses in $\mathfrak{C}$. For each $x \in \mathbb{R}^d$, a deterministic sparsifier, $\mathcal{C}_p$ with sparsification parameter, p determines a sparse support set, $S_p(x) \subseteq [d]$ and sparsifies as

$$\mathcal{C}_p(x) = \textstyle\sum_{i\in S_p(x)} x[i]e_i,$$

where e_i denotes the i^{th} standard basis in $\mathbb{R}^d$, and $x[i]$ denotes the corresponding element in x. For example, for hard-threshold sparsifier, $\mathcal{C}_\lambda$, we have $S_\lambda(x) = \{i \mid |x[i]| \geq \lambda\}$. Motivated by Theorem 1 and Remark 4, we now propose the following communication complexity model:

$$\min_{\mathcal{C}\in\mathbf{S}}\sum_{t=0}^{T-1}\ \mathbb{E}\|g_t + \frac{e_t}{\gamma} - \mathcal{C}(g_t + \frac{e_t}{\gamma})\|^2 \quad \text{subject to } \sum_{t=0}^{T-1}\|\mathcal{C}(g_t + \frac{e_t}{\gamma})\|_0 \leq K, \tag{3}$$

where K is the budget on the number of elements communicated in T iterations. However, solving (3) is intractable owing to complex DNN loss functions and multiple sources of randomness.

4.3 Top-k is communication-optimal for a per-iteration k element budget

To simplify (3), one can focus individually at the error at each iteration. Based on this, we show in Lemma 2 that Top-k has the best compression error among all sparsifiers under a per-iteration k-element communication budget.

Lemma 2. *Given the gradient g_t and error e_t at iteration t, Top-k sparsifier achieves the optimal objective for the optimization problem:*

$$\min_{\mathcal{C}\in\mathbf{S}}\|g_t+\frac{e_t}{\gamma}-\mathcal{C}(g_t+\frac{e_t}{\gamma})\|^2 \quad \textit{subject to } \|\mathcal{C}(g_t+\frac{e_t}{\gamma})\|_0\leq k. \tag{4}$$

Similar to Lemma 1, (4) is solved when $\mathcal{C}(g_t+\frac{e_t}{\gamma})$ contains the k highest magnitude elements of $g_t+\frac{e_t}{\gamma}$. That is, when $\mathcal{C}$ is the Top-k sparsifier. Additionally, based on the above model, a per-iteration k-element communication budget (resulting in a total budget of kT elements throughout training), implies that Top-k is performed at each iteration. However, to have a more communication-efficient compression, we require a communication complexity model that (*i*) better captures total-error in Theorem 1; and (*ii*) allows for adaptive communication, i.e., sends variable data in each iteration.

4.4 A communication complexity model for adaptive sparsification

Although the total-error cannot be minimized (§4.2), Lemma 2 motivates us to consider a *simplified model* that can capture the total-error. Instead of $(g_t+\frac{e_t}{\gamma})_{t=0}^{T-1}$, we consider a fixed sequence $(a_t)_{t=0}^{T-1}$ and examine the following communication complexity model:

$$\min_{\mathcal{C}\in\mathbf{S}}\sum_{t=0}^{T-1}\|a_t-\mathcal{C}(a_t)\|^2 \quad \text{subject to } \sum_{t=0}^{T-1}\|\mathcal{C}(a_t)\|_0\leq K, \tag{5}$$

where $K\in\mathbb{N}$ denotes the total communication budget. For the sake of simplicity, we assume that no two elements in $(a_t)_{t=0}^{T-1}$ have the same magnitude.

Let $\mathcal{A}\in\mathbb{R}^{dT}$ be formed by stacking $(a_t)_{t=0}^{T-1}$ vertically and consider the following sparse approximation problem:

$$\min_{B\in\mathbb{R}^{dT}}\|\mathcal{A}-B\|^2 \quad \text{subject to } \|B\|_0\leq K, \tag{6}$$

Note that (6) allows for all B that are formed by stacking $(\mathcal{C}(a_t))_{t=0}^{T-1}$ vertically, for some sparsifier $\mathcal{C}$ satisfying $\sum_{t=0}^{T-1}\|\mathcal{C}(a_t)\|_0\leq K$. Therefore, the optimal objective for (6) is a lower bound to the optimal objective for (5). Let $\mathcal{A}_{(i)}$ denote the element with i^{th} largest magnitude in $\mathcal{A}$, and since no two elements have the same magnitude, we have, $\mathcal{A}_{(i+1)}\neq\mathcal{A}_{(i)}$, for all $i\in[dT]$. Then, $B=\mathcal{C}_\lambda(A)$ with $\lambda\in\left(A_{(K+1)},A_{(K)}\right]$ contains the Top-K magnitude entries from A, and therefore, by Lemma 1 is optimal for (6). Moreover, since hard-threshold is an element-wise sparsifier, $\mathcal{C}_\lambda(A)$ is equivalent to stacking $(\mathcal{C}_\lambda(a_t))_{t=0}^{T-1}$ vertically. Therefore, $\mathcal{C}_\lambda$ with $\lambda=\left(\mathcal{A}_{(K+1)},\mathcal{A}_{(K)}\right]$ achieves optimal objective in (5). The following lemma formalizes this.

Lemma 3. *$\mathcal{C}_\lambda$ is optimal for the communication complexity model* (5)*. That is, for every budget K, there exists a $\lambda\geq 0$ such that $\mathcal{C}_\lambda$ minimizes* (5)*.*

4.5 Discussion

To capture the effect of compression, existing works [7, 33, 53] use a bound on the *compression factor*, $\max_{x\in\mathbb{R}^d}\frac{\mathbb{E}_\mathcal{C}\|\mathcal{C}(x)-x\|^2}{\|x\|^2}$. We formally define them as relative compressors.

Definition 2. Relative Compressor [7, 53]. An operator, $\mathcal{C}:\mathbb{R}^d\to\mathbb{R}^d$ is a *relative compressor* if for all vector, $x\in\mathbb{R}^d$ it satisfies

$$\mathbb{E}_\mathcal{C}\|x-\mathcal{C}(x)\|^2\leq\Omega\|x\|^2, \tag{7}$$

where $\Omega>0$ is the compression factor and the expectation, $\mathbb{E}_\mathcal{C}$, is taken with respect to the randomness of $\mathcal{C}$. *δ-contraction operators* [33, 53] with $\Omega=1-\delta$ and $\delta\in(0,1]$, are special cases of relative compressors.

Top-k is a δ-contraction operator with $\delta=\frac{k}{d}$ [53]. Therefore, by (7), Top-k allows for larger compression error with larger inputs. Our communication complexity model demonstrates that this might not necessarily be a good idea. Moreover, with EF, a large error at any iteration has a cascading effect — a large e_t results in a large γg_t+e_t, out of which only k/d fraction of the total components are kept by the Top-k strategy. This results in a large e_{t+1} (see §C.2). Figure 1 shows that this *error-buildup* has severe implications on the total-error. On the other hand, the hard-threshold performs a variable Top-k in each iteration and sends an element as soon as its magnitude is bigger than the threshold. This prohibits the error build-up.

Comparison with existing communication-optimal compression strategies. Since the compression factor, Ω solely determines the effect of compression in convergence [29, 54], many recent works [6, 15, 21, 45] propose communication-optimal compression strategies by optimizing for Ω under a *communication budget* for each vector, i.e., they propose to solve

$$\min_{\mathcal{C}\in\mathfrak{C}} \max_{x\in\mathbb{R}^d} \frac{\mathbb{E}_{\mathcal{C}}\|x-\mathcal{C}(x)\|^2}{\|x\|^2} \quad \text{subject to } \texttt{Bits}(\mathcal{C}(x)) \leq B, \tag{8}$$

where $\texttt{Bits}(\mathcal{C}(x))$ denotes the number of bits needed to encode $\mathcal{C}(x)$. We stress that while the compression affected term in Theorem 1 has the sum of compression errors over the iterations, the above communication complexity model only captures the compression factor.

5 Absolute compressors and their convergence

Motivated by the previous section, we formally define *absolute compressors* — compressors that have an absolute bound on the error.

Definition 3. Absolute Compressor. An operator, $\mathcal{C}:\mathbb{R}^d\to\mathbb{R}^d$ is an *absolute compressor* if there exists a $\upsilon>0$ such that for all vectors, $x\in\mathbb{R}^d$ it satisfies

$$\mathbb{E}_{\mathcal{C}}\|x-\mathcal{C}(x)\|^2 \leq \upsilon^2. \tag{9}$$

In contrast to the relative compressors in (7), the compression error (or variance) of absolute compressors is bounded by a constant, independent of x. Based on the above definition, hard-threshold is an absolute sparsifier with $\upsilon^2=d\lambda^2$. The stochastic rounding schemes, used for model quantization, with bounded rounding error in [24] are absolute compressors. Precisely, any rounding scheme, with rounding error bounded by ϵ, is an absolute compressor with $\upsilon^2=d\epsilon^2$. Similarly, the scaled integer rounding scheme in [46] is an absolute compressor. While this class of compressors existed in the literature, we are the first to provide their convergence result with an EF.

5.1 Convergence results

Inspired by [54], we establish convergence of EF-SGD (Algorithm 1) with absolute compressors. Convergence analysis for the momentum case [64] can be extended similarly. However, we do not include it for brevity, and the existing analyses do not show any benefit over vanilla SGD. Similarly, analysis for error-reset [61] and local updates [9, 61] can also be extended. We provide convergence results for the convex and non-convex cases and compare them to δ-contraction operators. We start with a bound on the error for absolute compressors.

Remark 5. **(Error bound)** For all $i\in[n], t\in\{0,\dots T-1\}$, we have

$$\mathbb{E}_{\mathcal{C}}[\|e_{i,t+1}\|^2 \mid p_{i,t}] = \mathbb{E}_{\mathcal{C}}\|p_{i,t}-\gamma_t\mathcal{C}(\tfrac{p_{i,t}}{\gamma_t})\|^2 = \gamma_t^2\mathbb{E}_{\mathcal{C}}\|\tfrac{p_{i,t}}{\gamma_t}-\mathcal{C}(\tfrac{p_{i,t}}{\gamma_t})\|^2 \leq \gamma_t^2\upsilon^2.$$

A similar absolute bound for δ-contraction operators requires the bounded gradient assumption [33, 9], but absolute compressors achieve this by design.

5.1.1 Convex convergence

Let $\bar{x}_T = \frac{1}{W_T}\sum_{t=0}^T w_t x_t$ be the weighted average of the iterates with weights, $w_t\geq 0$ and $W_T=\sum_{t=0}^T w_t$. Additionally, let $P_t := \mathbb{E}[f(\bar{x}_t)]-f^\star$ be the expected suboptimality gap at the average iterate. Further denote $R_t := \|x_t-x^\star\|^2$ and $D := \frac{1}{n}\sum_{i=1}^n\|\nabla f_i(x^\star)\|^2$. With these notations, we quote the strongly convex ($\mu>0$) and convex ($\mu=0$) convergence results for absolute compressors, and compare them with the δ-contraction operators from [10] for distributed case ($n\geq 1$). The results below are for specific choices of step-sizes and weights; we refer to §B.4.1 for these choices.

Theorem 4. *Let $\mu>0$ and Assumptions 1, 2, 3, and 5 hold. Then the iterates, $\{x_t\}_{t\geq 0}$ of Algorithm 1 with an absolute compressor, $\mathcal{C}_\upsilon$, a constant step-size, $\gamma(T)$ with $\gamma(T)\leq\frac{1}{4L(1+2M/n)}$ follow* [3]

$$P_T = \tilde{\mathcal{O}}\left(LR_0(1+M/n)\exp\left[-\tfrac{\mu T}{8L(1+2M/n)}\right] + \tfrac{\sigma^2+MD}{\mu nT} + \tfrac{L\upsilon^2}{\mu^2T^2}\right).$$

[3] The $\tilde{\mathcal{O}}$ notation hides constants and factors polylogarithmic in the problem parameters.

Remark 6. Under the same setting as in Theorem 4, iterates of Algorithm 1 with δ-contraction operators follow:

$$P_T = \tilde{\mathcal{O}}\left(LR_0\left(\frac{\sqrt{1+M\delta}}{\delta}\right)\exp\left[-\frac{\mu\delta T}{16\sqrt{3}L\sqrt{2+M\delta}}\right] + \frac{\sigma^2+MD}{\mu nT} + \frac{L(D(1+M\delta)+\delta\sigma^2)}{\mu^2\delta^2T^2}\right).$$

Remark 6 implies, in distributed settings with heterogeneous data ($D \neq 0$), δ-contraction operators have an $1/\delta^2$ dependence on δ, as compared to an $1/\delta$ dependence in the homogeneous case ($D = 0$). In contrast, absolute compressors have the same υ^2 dependence on υ in both cases. Therefore, we conjecture it is beneficial to use absolute compressors in settings such as federated learning [38], where data heterogeneity is widely encountered.

Theorem 5. *Let $\mu = 0$ and Assumptions 1, 2, 3, and 5 hold with $D \neq 0$. Then the iterates, $\{x_t\}_{t\geq 0}$ of Algo. 1 with an absolute compressor, $\mathcal{C}_\upsilon$, a constant step-size, $\gamma(T)$ with $\gamma(T) \leq \frac{1}{4L(1+2M/n)}$ follow*

$$P_T = \mathcal{O}\left(\frac{\sqrt{(\sigma^2+MD)R_0}}{\sqrt{nT}} + \frac{\left(\frac{nL\upsilon^2}{\sigma^2+MD}+L(1+M/n)\right)R_0}{T}\right).$$

Remark 7. Theorem 5 holds when both σ^2 and D are not simultaneously zero. Typically, we encounter heterogeneous data settings where $D \neq 0$, and Theorem 5 holds. In case both σ^2 and D are zero, we get $\mathcal{O}(\frac{(L\upsilon R_0)^{\frac{2}{3}}}{T^{\frac{2}{3}}} + \frac{L(1+M/n))R_0}{T})$ convergence.

Remark 8. Under the same setting as in Theorem 5, iterates of Algorithm 1 with δ-contraction operators follow:

$$P_T = \mathcal{O}\left(\frac{\sqrt{(\sigma^2+MD)R_0}}{\sqrt{nT}} + \frac{\left(\frac{L\sqrt{1+M\delta}}{\delta}+\frac{nL(D(1+M\delta)+\delta\sigma^2)}{\delta^2(\sigma^2+MD)}\right)R_0}{T}\right).$$

Similar to Remark 6, we observe that δ-contraction operators have $1/\delta^2$ dependence on δ in the heterogeneous case, and a $1/\delta$ dependence in the homogeneous case, while absolute compressors have no such anomaly.

Designing a variance-reduced algorithm [22, 43, 44] by using absolute compressors with EF is a fruitful direction of future research.

5.1.2 Non-convex convergence

Theorem 6. ***(Non-convex convergence of absolute compressors)*** *Let Assumptions 1, 2, 3, and 4 hold. Then the iterates, $\{x_t\}_{t\geq 0}$ of Algorithm 1 with an absolute compressor and a constant step-size $\gamma \leq \frac{n}{2L(M(C+1)+n)}$ follow*

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\|\nabla f(x_t)\|^2 \leq \frac{4(f(x_0)-f^\star)}{\gamma T} + \frac{2\gamma L(M\zeta^2+\sigma^2)}{n} + 2\gamma^2L^2\upsilon^2.$$

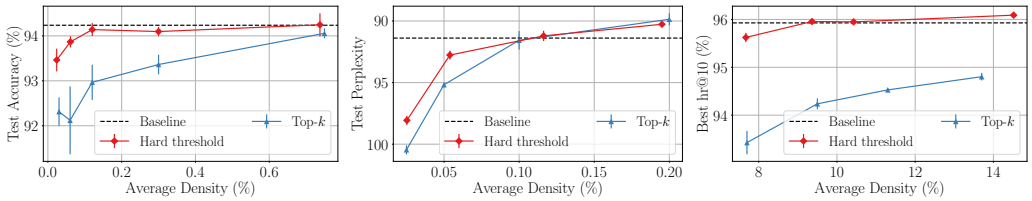
Alongside, we compare with the non-convex convergence for δ-contraction operators in a distributed setting. The existing analyses tackle this by using a stronger *uniform bounded gradient* assumption [64, 33, 20]. We use weaker Assumption 3 and 4, to establish the convergence analysis.

Theorem 7. ***(Non-convex convergence of δ-contraction operators)*** *Let Assumptions 1, 2, 3, and 4 hold. Then the iterates, $\{x_t\}_{t\geq 0}$ of Algorithm 1 with a δ-compressor and a constant step-size $\gamma \leq \min\{\frac{n}{2L(M(C+1)+n)}, \frac{1}{2L(2/\delta+M)\sqrt{C+1}}\}$ follow*

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\|\nabla f(x_t)\|^2 \leq \frac{8(f(x_0)-f^\star)}{\gamma T} + \frac{4\gamma L(M\zeta^2+\sigma^2)}{n} + \frac{8\gamma^2L^2}{\delta}\left(\left(\tfrac{2}{\delta}+M\right)\zeta^2+\sigma^2\right).$$

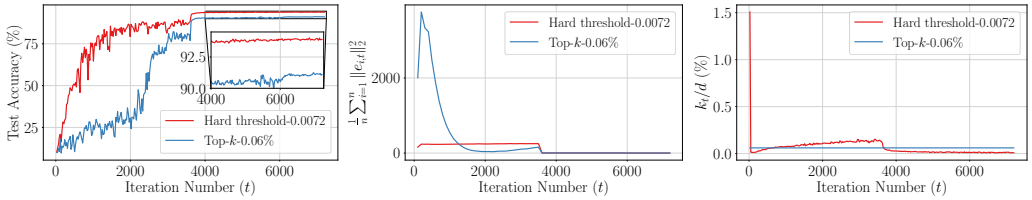
Again, similar to Remarks 6 and 8, for δ-contraction operators, we find the $1/\delta^2$ (heterogeneous case, $\zeta \neq 0$) vs. $1/\delta$ (homogeneous case, $\zeta = 0$) anomaly, while absolute compressors have υ^2 dependence on υ in both homogeneous and heterogeneous cases.

Remark 9. With appropriate choices of step-size, both absolute compressors and δ-contraction operators with EF-SGD achieve the same $\mathcal{O}(1/\sqrt{nT})$ asymptotic rate of SGD. See Corollary 1 in §B.3 for the full result.



(a) ResNet-18 on CIFAR-10 (b) LSTM on Wikitext (c) NCF on Movielens-20M

Figure 2: **Test metric vs. Data volume.** For 3 benchmarks, average test quality with std. dev. over 3 runs. The dashed black line denotes the no compression baseline.



(a) (b) (c)

Figure 3: **Convergence of Top-k and Hard-threshold for ResNet-18 on CIFAR-10 at 0.06% average density:** (a) Test-accuracy vs. Iterations, (b) Error-norm vs. Iterations, (c) Density (k_t/d) vs. Iterations. $k = 0.06\%$ of d, and $\lambda = 0.0072$. Hard-threshold has better convergence than Top-k because of a smaller total-error.

6 Experiments

Experimental setup. We compare Top-k and hard-threshold sparsifiers on image classification, language modelling, and recommendation tasks. We use different optimizers: vanilla SGD, SGD with Nesterov momentum, and ADAM [34]. All experiments were run on an 8-GPU cluster, using `Allgather` as the communication primitive. We perform compression in the standard layer-wise fashion [20, 37, 47] and follow the EF strategy used in [57]. For hyper-parameter configuration, comparison with entire-model compression, discussion on different EF approaches, experiments without EF, and experiments with logistic regression, we refer to Appendix C.

Test metric vs. Data volume. We tune the sparsification parameters for both sparsifiers such that they send similar total data volumes during training. We use *average density*: $\frac{1}{T}\sum_{t=0}^{T-1}\frac{k_t}{d}$ as a measure of total data volume, where k_t denotes the number of elements transmitted in iteration t. Figure 2 shows the average test quality across three repetitions with different initial random seeds. We observe that fixing the average density, *hard-threshold consistently has better test performance* than Top-k. For ResNet-18 on CIFAR-10, we observe that hard-threshold at an average density of 0.12% almost achieves the baseline accuracy and is better than Top-k at 0.75% density ($\sim 6\times$ more total data volume). For LSTM on Wikitext, at an average density of 0.025%, hard-threshold has > 2 better perplexity than Top-k. For NCF on Movielens-20M, hard-threshold has $> 1\%$ better Hit-Rate@10 at all considered average densities.

We now demonstrate that hard-threshold has faster convergence because of a smaller total-error in comparison to Top-k. In Figure 3, we introspect a run with average density of 0.06% from Figure 2a. In Figure 3a, while hard-threshold converges to an accuracy of 93.9%, Top-k achieves 91.1% accuracy. At the same time, in Figure 3b, we observe large error-accumulation in the initial $1,200$ iterations for Top-k. Consequently, hard-threshold has a significantly lower total-error than Top-k, and therefore has better convergence. This observation about large error accumulation for Top-k is consistent across all our benchmarks (see §C.2).

Comparison against ACCORDION. We compare against the state-of-the-art adaptive sparsifier: ACCORDION [3] on CIFAR-10 and CIFAR-100 datasets. ACCORDION shifts between two user-defined k values: $k_{\max}$ and $k_{\min}$, by using Top-$k_{\max}$ when the training is in a *critical regime*, else using Top-$k_{\min}$. We compare against ACCORDION with hard-threshold $\lambda = \frac{1}{2\sqrt{k_{\min}}}$. For complete experiment details; see §C.4.

We report the CIFAR-10 result in Table 1, while the CIFAR-100 result is reported in §C.4. Each setting is repeated with 6 different seeds and we report the average. For the CIFAR-10 dataset,

Table 1: Comparison against ACCORDION [3] on CIFAR-10.

Network	Method	Accuracy (%)	Average Density (%)
ResNet-18	Top-1% ($k_{\max}/d$)	94.1	1.00 (1×)
	Top-0.1% ($k_{\min}/d$)	93.2	0.10 (10×)
	ACCORDION	93.5	0.53 (1.9×)
	Hard-threshold ($\frac{1}{2\sqrt{k_{\min}}}$)	**94.0**	**0.13** (**7.7**×)
GoogleNet	Top-1% ($k_{\max}/d$)	94.1	1.00 (1×)
	Top-0.1% ($k_{\min}/d$)	92.9	0.10 (10×)
	ACCORDION	93.4	0.47 (2.1×)
	Hard-threshold ($\frac{1}{2\sqrt{k_{\min}}}$)	**94.2**	**0.13** (**7.7**×)
SENet18	Top-1% ($k_{\max}/d$)	94.0	1.00 (1×)
	Top-0.1% ($k_{\min}/d$)	92.5	0.10 (10×)
	ACCORDION	93.5	0.47 (2.1×)
	Hard-threshold ($\frac{1}{2\sqrt{k_{\min}}}$)	**94.2**	**0.14** (**7.1**×)

we observe that hard-threshold has $0.5\% - 0.8\%$ higher test accuracy than ACCORDION and is approximately $3.5\times$ more communication efficient than ACCORDION. For the CIFAR-100 dataset, except the ResNet-18 model, we observe that hard-threshold obtains more than 0.8% higher accuracy than ACCORDION with more than $1.26\times$ communication savings over ACCORDION.

How to tune the hard-threshold? We use the non-convex convergence results from §5.1.2 to suggest a hard-threshold value which has better convergence than Top-k with parameter k for non-convex loss functions (including DNNs). Let $\hat{M}$, $\hat{\zeta}$, and $\hat{\sigma}$ be the estimates of M, ζ, and σ, respectively, in Assumptions 3 and 4. We set the threshold as $\lambda \sim \frac{2}{\sqrt{k}}\sqrt{\left(\frac{2d}{k} + \hat{M}\right)\hat{\zeta}^2 + \hat{\sigma}^2}$; see discussion in §D. The λ in Table 1 is derived from simplifying this formula. But how to tune the hard-threshold such that it achieves no-compression baseline performance with the least total-data transmission remains an open question. We remark, as of now, this question remains unanswered for Top-k as well.

When and when not to use hard-threshold? In a standard cluster setting with a dedicated network, the speedup in terms of per-iteration training time due to gradient compression depends on the characteristics of the DNN being trained [46]. One of the determining characteristics is the extent to which the communication phase overlaps with computation. If the fraction of non-overlapped communication is significant, then communication is a bottleneck, even if Top-k compression is applied. However, in the case of hard-threshold sparsification (configured for the same total communication volume), during iterations with high data transmission, the non-overlapped communication remains; but during iterations with low data transmission, non-overlapped communication reduces, thereby reducing the overall training time. On the other hand, if there is complete overlap between computation and communication for Top-k, then a hard-threshold with the same total communication volume may introduce non-overlapped communication in some iterations with high data transmission, thereby increasing overall training time. Here, we ignored two important aspects of hard-threshold: (i) Hard-threshold may require fewer iterations to a target accuracy owing to its better statistical efficiency, and that (ii) hard-threshold has negligible computation overhead in comparison to Top-k.

7 Conclusion

We proposed a total-error perspective to compressed communication that captures the effect of compression during the entire training process. Under this, we showed that the hard-threshold sparsifier is more communication-efficient than the state-of-the-art Top-k sparsifier, and is a principled way to perform adaptive sparsification. Absolute compressors – the class of compressors in which hard-threshold belongs – have promising convergence in the heterogeneous data settings, which is a prominent issue in Federated Learning [38]. As the EF framework is also applicable to Local SGD [52], we hope that this inspires more communication-efficient versions of Local SGD that adaptively determine when to communicate, rather than naively communicating in fixed intervals. Furthermore, similar to hard-threshold, we believe adaptive absolute compressor counterparts of quantization schemes and low-rank methods can also be developed.

Acknowledgements

We thank Chen-Yu Ho and Hang Xu for many helpful discussions and the reviewers for their feedback. For computer time, this research used the resources of the Supercomputing Laboratory at KAUST.

References

[1] A. M. Abdelmoniem, A. Elzanaty, M-S. Alouini, and M. Canini. An Efficient Statistical-based Gradient Compression Technique for Distributed Training Systems. In *MLSys*, 2021.

[2] A. Achille, M. Rovere, and S. Soatto. Critical Learning Periods in Deep Networks. In *ICLR*, 2019.

[3] S. Agarwal, H. Wang, K. Lee, S. Venkataraman, and D. Papailiopoulos. Accordion: Adaptive Gradient Communication via Critical Learning Regime Identification. In *MLSys*, 2021.

[4] A. F. Aji and K. Heafield. Sparse communication for distributed gradient descent. In *EMNLP*, 2017.

[5] T. Alabi, J. D. Blanchard, B. Gordon, and R. Steinbach. Fast K-Selection Algorithms for Graphics Processing Units. *ACM J. Exp. Algorithmics*, 17, October 2012.

[6] A. Albasyoni, M. Safaryan, L. Condat, and P. Richtárik. Optimal Gradient Compression for Distributed and Federated Learning. *arXiv 2010.03246*, 2020.

[7] D. Alistarh, D. Grubic, J. Li, R. Tomioka, and M. Vojnovic. QSGD: Communication-Efficient SGD via Gradient Quantization and Encoding. In *NeurIPS*, 2017.

[8] D. Alistarh, T. Hoefler, M. Johansson, N. Konstantinov, S. Khirirat, and C. Renggli. The Convergence of Sparsified Gradient Methods. In *NeurIPS*, 2018.

[9] D. Basu, D. Data, C. Karakus, and S. Diggavi. Qsparse-local-SGD: Distributed SGD with Quantization, Sparsification, and Local Computations. In *NeurIPS*, 2019.

[10] A. Beznosikov, S. Horváth, P. Richtárik, and M. Safaryan. On Biased Compression for Distributed Learning. *arXiv 2002.12410*, 2020.

[11] T. Boas, A. Dutta, X. Li, K. Mercier, and E. Niderman. Shrinkage Function And Its Applications In Matrix Approximation. *The Electronic Journal of Linear Algebra*, 32, 2017.

[12] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei. Language Models are Few-Shot Learners. In *NeurIPS*, 2020.

[13] K. Bryan and T. Leise. Making Do with Less: An Introduction to Compressed Sensing. *SIAM Review*, 55(3), 2013.

[14] C-C Chang and C-J Lin. LIBSVM: A Library for Support Vector Machines. *ACM Trans. Intell. Syst. Technol.*, 2(3), 2011.

[15] W.-N. Chen, P. Kairouz, and A. Ozgur. Breaking the Communication-Privacy-Accuracy Trilemma. In *NeurIPS*, 2020.

[16] T. Dettmers. 8-Bit Approximations for Parallelism in Deep Learning. In *ICLR*, 2016.

[17] D. L. Donoho. Compressed sensing. *IEEE Transactions on information theory*, 52(4), 2006.

[18] N. Dryden, T. Moon, S. A. Jacobs, and B. V. Essen. Communication Quantization for Data-Parallel Training of Deep Neural Networks. In *MLHPC*, 2016.

[19] A. Dutta. *Weighted Low-Rank Approximation of Matrices: Some Analytical and Numerical Aspects*. PhD thesis, University of Central Florida, 2016.

[20] A. Dutta, E. H. Bergou, A. M. Abdelmoniem, C.-Y. Ho, A. N. Sahu, M. Canini, and P. Kalnis. On the Discrepancy between the Theoretical Analysis and Practical Implementations of Compressed Communication for Distributed Deep Learning. In *AAAI*, 2020.

[21] V. Gandikota, D. Kane, R. K. Maity, and A. Mazumdar. vqSGD: Vector Quantized Stochastic Gradient Descent. *arXiv 1911.07971*, 2019.

[22] E. Gorbunov, D. Kovalev, D. Makarenko, and P. Richtarik. Linearly Converging Error Compensated SGD. In *NeurIPS*, 2020.

[23] J. Guo, W. Liu, W. Wang, J. Han, R. Li, Y. Lu, and S. Hu. Accelerating Distributed Deep Learning By Adaptive Gradient Quantization. In *ICASSP*, 2020.

[24] S. Gupta, A. Agrawal, K. Gopalakrishnan, and P. Narayanan. Deep Learning with Limited Numerical Precision. *ICML*, 2015.

[25] V. Gupta, D. Choudhary, P. T. P. Tang, X. Wei, X. Wang, Y. Huang, A. Kejariwal, K. Ramchandran, and M. W. Mahoney. Fast Distributed Training of Deep Neural Networks: Dynamic Communication Thresholding for Model and Data Parallelism. *arXiv 2010.08899*, 2020.

[26] K. He, X. Zhang, S. Ren, and J. Sun. Deep Residual Learning for Image Recognition. In *CVPR*, 2015.

[27] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua. Neural Collaborative Filtering. In *WWW*, 2017.

[28] S. Hochreiter and J. Schmidhuber. Long Short-Term Memory. *Neural Computing*, 9(8), 1997.

[29] S. Horváth and P. Richtarik. A Better Alternative to Error Feedback for Communication-Efficient Distributed Learning. In *ICLR*, 2021.

[30] J. Hu, L. Shen, and G. Sun. Squeeze-and-excitation networks. In *CVPR*, 2018.

[31] P. Jiang and G. Agrawal. A linear speedup analysis of distributed deep learning with sparse and quantized communication. In *NeurIPS*, 2018.

[32] Y. Jiang, Y. Zhu, C. Lan, B. Yi, Y. Cui, and C. Guo. A Unified Architecture for Accelerating Distributed DNN Training in Heterogeneous GPU/CPU Clusters. In *OSDI*, 2020.

[33] S. P. Karimireddy, Q. Rebjock, S. U. Stich, and M. Jaggi. Error Feedback Fixes SignSGD and other Gradient Compression Schemes. In *ICML*, 2019.

[34] D. P. Kingma and J. Ba. ADAM: A Method for Stochastic Optimization. In *ICLR*, 2015.

[35] A. Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

[36] X. Lian, C. Zhang, H. Zhang, C.-J. Hsieh, W. Zhang, and J. Liu. Can Decentralized Algorithms Outperform Centralized Algorithms? A Case Study for Decentralized Parallel Stochastic Gradient Descent. In *NeurIPS*, 2017.

[37] Y. Lin, S. Han, H. Mao, Y. Wang, and W. Dally. Deep Gradient Compression: Reducing the Communication Bandwidth for Distributed Training. In *ICLR*, 2018.

[38] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. Agüera y Arcas. Communication-Efficient Learning of Deep Networks from Decentralized Data. In *AISTATS*, 2017.

[39] S. Merity, C. Xiong, J. Bradbury, and R. Socher. Pointer Sentinel Mixture Models. In *ICLR*, 2017.

[40] D. Narayanan, A. Harlap, A. Phanishayee, V. Seshadri, N. R. Devanur, G. R. Ganger, P. B. Gibbons, and M. Zaharia. PipeDream: Generalized Pipeline Parallelism for DNN Training. In *SOSP*, 2019.

[41] Y. Nesterov. *Lectures on Convex Optimization*, volume 137. Springer, 2018.

[42] PyTorch. `https://pytorch.org/`.

[43] X. Qian, H. Dong, P. Richtárik, and T. Zhang. Error Compensated Loopless SVRG for Distributed Optimization. In *Workshop on Optimization for Machine Learning*, 2020.

[44] X. Qian, P. Richtárik, and T. Zhang. Error Compensated Distributed SGD Can Be Accelerated. *arXiv 2010.00091*, 2020.

[45] M. Safaryan, E. Shulgin, and P. Richtárik. Uncertainty principle for communication compression in distributed and federated learning and the search for an optimal compressor. *Information and Inference: A Journal of the IMA*, 2021.

[46] A. Sapio, M. Canini, C-Y Ho, J. Nelson, P. Kalnis, C. Kim, A. Krishnamurthy, M. Moshref, D. Ports, and P. Richtarik. Scaling Distributed Machine Learning with In-Network Aggregation. In *NSDI*, 2021.

[47] F. Seide, H. Fu, J. Droppo, G. Li, and D. Yu. 1-Bit Stochastic Gradient Descent and Application to Data-Parallel Distributed Training of Speech DNNs. In *INTERSPEECH*, 2014.

[48] A. Shanbhag, H. Pirk, and S. Madden. Efficient Top-K Query Processing on Massively Parallel Hardware. In *SIGMOD*, 2018.

[49] S. Shi, X. Chu, K. C. Cheung, and S. See. Understanding Top-k Sparsification in Distributed Deep Learning. *arXiv 1911.08772*, 2019.

[50] M. Shoeybi, M. Patwary, R. Puri, P. LeGresley, J. Casper, and B. Catanzaro. Megatron-LM: Training Multi-Billion Parameter Language Models Using Model Parallelism. *arXiv 1909.08053*, 2019.

[51] G. W. Stewart. On the Early History of the Singular Value Decomposition. *SIAM review*, 35(4), 1993.

[52] S. U. Stich. Local SGD Converges Fast and Communicates Little. In *ICLR*, 2019.

[53] S. U. Stich, J. B. Cordonnier, and M. Jaggi. Sparsified SGD with Memory. In *NeurIPS*, 2018.

[54] S. U. Stich and S. P. Karimireddy. The Error-Feedback Framework: Better Rates for SGD with Delayed Gradients and Compressed Updates. *Journal of Machine Learning Research*, 21, 2020.

[55] N. Strom. Scalable Distributed DNN Training using Commodity GPU Cloud Computing. In *INTERSPEECH*, 2015.

[56] C. Szegedy, Wei L., Yangqing J., P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going Deeper with Convolutions. In *CVPR*, 2015.

[57] T. Vogels, S. P. Karimireddy, and M. Jaggi. PowerSGD: Practical Low-Rank Gradient Compression for Distributed Optimization. In *NeurIPS*, 2019.

[58] H. Wang, S. Sievert, S. Liu, Z. Charles, D. Papailiopoulos, and S. Wright. ATOMO: Communication-efficient Learning via Atomic Sparsification. In *NeurIPS*, 2018.

[59] J. Wangni, J. Wang, J. Liu, and T. Zhang. Gradient Sparsification for Communication-Efficient Distributed Optimization. In *NeurIPS*, 2018.

[60] W. Wen, C. Xu, F. Yan, C. Wu, Y. Wang, Y. Chen, and H. Li. TernGrad: Ternary Gradients to Reduce Communication in Distributed Deep Learning. In *NeurIPS*, 2017.

[61] C. Xie, S. Zheng, O. Koyejo, I. Gupta, M. Li, and H. Lin. CSER: Communication-efficient SGD with Error Reset. In *NeurIPS*, 2020.

[62] H. Xu, C.-Y. Ho, A. M. Abdelmoniem, A. Dutta, E. H. Bergou, K. Karatsenidis, M. Canini, and P. Kalnis. GRACE: A Compressed Communication Framework for Distributed Machine Learning. In *ICDCS*, 2021.

[63] G. Yan, S-L Huang, T. Lan, and L. Song. DQ-SGD: Dynamic Quantization in SGD for Communication-Efficient Distributed Learning. *arXiv 2107.14575*, 2021.

[64] S. Zheng, Z. Huang, and J. T Kwok. Communication-Efficient Distributed Blockwise Momentum SGD with Error-Feedback. In *NeurIPS*, 2019.

[65] Y. Zhong, C. Xie, S. Zheng, and H. Lin. Compressed Communication for Distributed Training: Adaptive Methods and System. *arXiv 2105.07829*, 2021.

[66] M. Zinkevich, M. Weimer, L. Li, and A. J. Smola. Parallelized Stochastic Gradient Descent. In *NeurIPS*, 2010.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [N/A]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] . See §3.1.
   (b) Did you include complete proofs of all theoretical results? [Yes] . All proofs are in §B.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] We provide the URL to our code repository.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See §C.1.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See Experimental setup in section §6.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] See §C.1.
   (b) Did you mention the license of the assets? [Yes] We use open-source code.
   (c) Did you include any new assets either in the supplemental material or as a URL? [No]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A] We use open-source datasets.
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

Contents

Appendices are supporting material that has not been peer-reviewed.

A Notations

In this paper, by $[d]$ we denote the set of d natural numbers $\{1, 2, \cdots, d\}$. We denote the ℓ_2 norm of a vector $x \in \mathbb{R}^d$ by $\|x\|$, and the ℓ_1 and ℓ_∞-norms are denoted by $\|x\|_1$ and $\|x\|_\infty$, respectively. By $\mathbf{0}$ we denote a vector of all 0s in $\mathbb{R}^d$. In the proofs, we use the notation $\mathbb{E}_t[\cdot]$ to denote expectation conditioned on the iterate, x_t, that is, $\mathbb{E}[\cdot|x_t]$.

B Convergence analysis

In this section, we provide the proofs of convex and non-convex convergence results of the absolute compressors with EF, and compare them with that of the δ-contraction operators, and vanilla SGD.

B.1 Overview of results

In §B.2, we provide the technical lemmas and inequalities necessary for the analyses. In §B.3 we provide the non-convex convergence results, and §B.4 contains the convex convergence results.

B.2 Technical results

Lemma 4. *If $a, b \in \mathbb{R}^d$ then the Young's inequality is: For all $\rho > 0$, we have*

$$\|a + b\|^2 \leq (1 + \rho)\|a\|^2 + (1 + \rho^{-1})\|b\|^2. \tag{10}$$

Alternatively,

$$2\langle a, b\rangle \leq \rho\|a\|^2 + \rho^{-1}\|b\|^2. \tag{11}$$

Lemma 5. *For $a_i \in \mathbb{R}^d$ we have:*

$$\|\frac{1}{n}\sum_{i=1}^{n} a_i\|^2 \leq \frac{1}{n}\sum_{i=1}^{n}\|a_i\|^2. \tag{12}$$

Lemma 6. *[54] Let $r_0, c \geq 0$, $d, T > 0$, and $0 < \gamma \leq \frac{1}{d}$. Then choosing $\gamma = \min(\frac{1}{d}, \sqrt{\frac{r_0}{cT}})$, the following holds:*

$$\frac{r_0}{\gamma T} + c\gamma \leq \frac{dr_0}{T} + \frac{2\sqrt{cr_0}}{\sqrt{T}}$$

Proof. We consider two cases. If $\frac{r_0}{cT} \leq \frac{1}{d^2}$, then choosing the step-size $\gamma = \left(\frac{r_0}{cT}\right)^{1/2}$, we get

$$\frac{r_0}{\gamma T} + c\gamma \leq \frac{2\sqrt{cr_0}}{\sqrt{T}}.$$

Else, if $\frac{r_0}{cT} > \frac{1}{d^2}$, then choosing $\gamma = \frac{1}{d}$, we get

$$\frac{r_0}{\gamma T} + c\gamma \leq \frac{dr_0}{T} + \frac{c}{d} \leq \frac{dr_0}{T} + \frac{\sqrt{cr_0}}{\sqrt{T}}.$$

Combining both bounds, we get the result. □

Lemma 7. *Let $r_0, b \geq 0$, $c, d, T > 0$, and $0 < \gamma \leq \frac{1}{d}$. Then choosing $\gamma = \min(\frac{1}{d}, \sqrt{\frac{r_0}{cT}})$, the following holds:*

$$\frac{r_0}{\gamma T} + c\gamma + b\gamma^2 \leq \frac{dr_0}{T} + \frac{2\sqrt{cr_0}}{\sqrt{T}} + \frac{br_0}{cT}.$$

Proof. The proof follows similar to Lemma 6. We consider two cases. If $\frac{r_0}{cT} \leq \frac{1}{d^2}$, then choosing the step-size $\gamma = \left(\frac{r_0}{cT}\right)^{1/2}$, we get

$$\frac{r_0}{\gamma T} + c\gamma + b\gamma^2 \leq \frac{2\sqrt{cr_0}}{\sqrt{T}} + \frac{br_0}{cT}.$$

Else, if $\frac{r_0}{cT} > \frac{1}{d^2}$, then choosing $\gamma = \frac{1}{d}$, we get

$$\frac{r_0}{\gamma T} + c\gamma + b\gamma^2 \leq \frac{dr_0}{T} + \frac{c}{d} + \frac{b}{d^2} \leq \frac{dr_0}{T} + \frac{\sqrt{cr_0}}{\sqrt{T}} + \frac{br_0}{cT}.$$

Combining both bounds, we get the result. □

Lemma 8. *[54] Let $r_0 \geq 0$, $d, T > 0$, and $0 < \gamma \leq \frac{1}{d}$. Then choosing $\gamma = \min(\frac{1}{d}, \left(\frac{r_0}{bT}\right)^{1/3})$, the following holds:*

$$\frac{r_0}{\gamma T} + b\gamma^2 \leq \frac{dr_0}{T} + \frac{2(br_0)^{2/3}}{T^{2/3}}.$$

Proof. We consider two cases. If $\frac{r_0}{bT} \leq \frac{1}{d^3}$, then choosing the step-size $\gamma = \left(\frac{r_0}{bT}\right)^{1/3}$, we get

$$\frac{r_0}{\gamma T} + b\gamma^2 \leq \frac{2(br_0)^{2/3}}{T^{2/3}}.$$

Else, if $\frac{r_0}{bT} > \frac{1}{d^3}$, then choosing $\gamma = \frac{1}{d}$, we get

$$\frac{r_0}{\gamma T} + b\gamma^2 \leq \frac{dr_0}{T} + \frac{b}{d^2} \leq \frac{dr_0}{T} + \frac{(br_0)^{2/3}}{T^{2/3}}.$$

Combining both bounds, we get the result. □

Lemma 9. *For every non-negative sequence $\{r_t\}_{t\geq 0}$ and parameters, $a > 0, b, c \geq 0$, $T \geq 2$,$\phi \geq 1$, decreasing step-sizes $\{\gamma_t := \frac{2}{a(\phi+t)}\}_{t\geq 0}$, and weights $\{w_t := (\phi + t)\}_{t\geq 0}$, satisfy*

$$\Psi_T := \frac{1}{W_T}\sum_{t=0}^{T}\left(\frac{w_t}{\gamma_t}(1 - a\gamma_t)r_t - \frac{w_t}{\gamma_t}r_{t+1} + c\gamma_t w_t + b\gamma_t^2 w_t\right) \leq \frac{4c}{aT} + \frac{a\phi^2 r_0}{T^2} + \frac{16b\ln(T)}{a^2T^2},$$

where $W_T := \sum_{t=0}^{T} w_t$.

Proof. This proof is motivated from Lemma 11 in [54]. We observe

$$\frac{w_t}{\gamma_t}(1 - a\gamma_t)r_t = \frac{a}{2}(\phi + t)(\phi + t - 2)r_t = \frac{a}{2}((\phi + t - 1)^2 - 1)r_t \leq \frac{a}{2}(\phi + t - 1)^2 r_t. \quad (13)$$

By plugging in the definition of γ_t and w_t in Ψ_t, we find

$$\begin{aligned}\Psi_T &\overset{(13)}{\leq} \frac{1}{W_T}\sum_{t=0}^{T}\left(\frac{a}{2}(\phi + t - 1)^2 r_t - \frac{a}{2}(\phi + t)^2 r_{t+1}\right) + \sum_{t=0}^{T}\frac{2c}{aW_T} + \sum_{t=0}^{T}\frac{4b}{a^2(\phi + t)W_T} \\ &\leq \frac{a(\phi - 1)^2 r_0}{2W_T} + \frac{2c(T + 1)}{aW_T} + \frac{4b}{a^2 W_T}\sum_{t=0}^{T}\frac{1}{\phi + t}.\end{aligned}$$

By using $(\phi - 1)^2 \leq \phi^2$, $W_T = \sum_{t=0}^{T}(\phi + t) \geq \frac{(2\phi+T)(T+1)}{2} \geq \frac{(T+1)(T+2)}{2}$, and $\sum_{t=0}^{T}\frac{1}{\phi+t} \leq \sum_{t=0}^{T}\frac{1}{1+t} \leq \ln(T + 1) + 1$, we have

$$\Psi_T \leq \frac{a\phi^2 r_0}{(T + 1)(T + 2)} + \frac{4c}{a(T + 2)} + \frac{8b(\ln(T + 1) + 1)}{a^2(T + 1)(T + 2)}.$$

For $T \geq 2$, we have $\frac{(\ln(T+1)+1)}{(T+1)(T+2)} \leq \frac{2\ln(T)}{T^2}$. By using this, we get

$$\Psi_T \leq \frac{a\phi^2 r_0}{T^2} + \frac{4c}{aT} + \frac{16b\ln(T)}{a^2T^2}.$$

Hence the result. □

Lemma 10. *(Lemma D.2 in [22]) For every non-negative sequence $\{r_t\}_{t\geq 0}$ and parameters, $d \geq a > 0$, $b, c, T \geq 0$, with a bound on the step-size $\gamma_t \leq \frac{1}{d}$, there exists a constant step-size,*

$$\gamma_t = \gamma = \min\{\frac{1}{d}, \frac{\ln(\max\{2, \min\{a^2 r_0 T^2/c, a^3 r_0 T^3/b\}\})}{aT}\}$$

and weights, $w_t := (1-a\gamma)^{-(t+1)}$, such that for all T satisfying $\frac{\ln(\max\{2,\min\{a^2 r_0 T^2/c, a^3 r_0 T^3/b\}\})}{T} \leq 1$, we have

$$\Psi_T := \frac{1}{W_T}\sum_{t=0}^{T}\left(\frac{w_t}{\gamma_t}(1-a\gamma_t)r_t - \frac{w_t}{\gamma_t}r_{t+1} + c\gamma_t w_t + b\gamma_t^2 w_t\right) = \tilde{\mathcal{O}}\left(dr_0 \exp\left[-\frac{a}{d}T\right] + \frac{c}{aT} + \frac{b}{a^2T^2}\right).$$

Proof. Substituting the values for γ_t and w_t, we get

$$\begin{aligned}\Psi_T &= \frac{1}{\gamma W_T}\sum_{t=0}^{T}(w_{t-1}r_t - w_t r_{t+1}) + \frac{c\gamma}{W_T}\sum_{t=0}^{T} w_t + \frac{b\gamma^2}{W_T}\sum_{t=0}^{T} w_t \\ &\leq \frac{r_0}{\gamma W_T} + c\gamma + b\gamma^2 \\ &\leq \frac{r_0}{\gamma}\exp[-a\gamma T] + c\gamma + b\gamma^2, \end{aligned} \tag{14}$$

where we use $W_T \geq w_T \geq (1-a\gamma)^{-T} \geq \exp[a\gamma T]$ in the last inequality. To tune γ, we consider following two cases:
• If $\frac{1}{d} \geq \frac{\ln(\max\{2,\min\{a^2 r_0 T^2/c, a^3 r_0 T^3/b\}\})}{aT}$, then we choose $\gamma = \frac{\ln(\max\{2,\min\{a^2 r_0 T^2/c, a^3 r_0 T^3/b\}\})}{aT}$ and (14) becomes $\tilde{\mathcal{O}}(\frac{c}{aT} + \frac{b}{a^2T^2})$, as
• If $\frac{1}{d} < \frac{\ln(\max\{2,\min\{a^2 r_0 T^2/c, a^3 r_0 T^3/b\}\})}{aT}$, then we choose $\gamma = \frac{1}{d}$ and (14) becomes $\tilde{\mathcal{O}}(dr_0 \exp\left[-\frac{a}{d}T\right] + \frac{c}{aT} + \frac{b}{a^2T^2})$.
Combining both bounds, we get the result. □

The recurrence relation in the next lemma is instrumental for perturbed iterate analysis of Algorithm 1 used in both convex and non-convex cases.

Lemma 11. *Let $\bar{e}_t = \frac{1}{n}\sum_{i=1}^{n} e_{i,t}$, $\bar{g}_t = \frac{1}{n}\sum_{i=1}^{n} g_{i,t}$, and $\bar{p}_t = \frac{1}{n}\sum_{i=1}^{n} p_{i,t}$. Define the sequence of iterates $\{\tilde{x}_t\}_{t\geq 0}$ as $\tilde{x}_t = x_t - \bar{e}_t$, with $\tilde{x}_0 = x_0$. Then $\{\tilde{x}_t\}_{t\geq 0}$ satisfy the recurrence: $\tilde{x}_{t+1} = \tilde{x}_t - \gamma_t \bar{g}_t$.*

Proof. We have

$$\tilde{x}_{t+1} = x_{t+1} - \bar{e}_{t+1} = x_t - (\bar{e}_t + \gamma_t \bar{g}_t) = \tilde{x}_t - \gamma_t \bar{g}_t.$$

Hence the result. □

B.3 Non-convex convergence analysis

In this section, we provide the non-convex convergence analyses. Lemma 13 provides a one-step descent recurrence which leads to Theorem 1 and a key result for proving convergence. Based on this, in §B.3.1, §B.3.2, §B.3.3 we discuss the convergence of absolute compressors, δ-contraction operators, and uncompressed SGD, respectively. In §B.3.4 we provide the convergence result for absolute compressors and δ-contraction operators for an appropriate choice of step-size. The following lemma bounds the quantity $\mathbb{E}_t\|\frac{1}{n}\sum_{i=1}^{n} g_{i,t}\|^2$.

Lemma 12. *Let f follow Assumption 4 and the stochastic noise, $\xi_{i,t}$ follow Assumption 3. Then we have*

$$\mathbb{E}_t\|\frac{1}{n}\sum_{i=1}^{n} g_{i,t}\|^2 \leq (1 + \frac{M(C+1)}{n})\|\nabla f(x_t)\|^2 + \frac{M\zeta^2 + \sigma^2}{n}. \tag{15}$$

Proof. Let the stochastic gradient, $g_{i,t}$ computed at i^{th} worker at iteration t follows $g_{i,t} = \nabla f_i(x_t) + \xi_{i,t}$ with $\mathbb{E}[\xi_{i,t}|x_t] = \mathbf{0}$. Hence, we have

$$\begin{aligned}
\mathbb{E}_t\|\frac{1}{n}\sum_{i=1}^n g_{i,t}\|^2 &= \mathbb{E}_t\|\frac{1}{n}\sum_{i=1}^n(\nabla f_i(x_t) + \xi_{i,t})\|^2 \\
&\overset{\mathbb{E}[\xi_{i,t}|x_t]=0}{=} \|\nabla f(x_t)\|^2 + \mathbb{E}_t\|\frac{1}{n}\sum_{i=1}^n \xi_{i,t}\|^2 \\
&\overset{\mathbb{E}[\xi_{i,t}|x_t]=0}{=} \|\nabla f(x_t)\|^2 + \frac{1}{n^2}\sum_{i=1}^n \mathbb{E}_t\|\xi_{i,t}\|^2 \\
&\overset{\text{By Assumption 3}}{\leq} \|\nabla f(x_t)\|^2 + \frac{1}{n^2}\sum_{i=1}^n (M\|\nabla f_i(x_t)\|^2 + \sigma^2) \\
&= \|\nabla f(x_t)\|^2 + \frac{M}{n^2}\sum_{i=1}^n \|\nabla f_i(x_t) - \nabla f(x_t)\|^2 + \frac{M\|\nabla f(x_t)\|^2}{n} + \frac{\sigma^2}{n} \\
&\overset{\text{By Assumption 4}}{\leq} (1 + \frac{M}{n})\|\nabla f(x_t)\|^2 + \frac{M}{n}(C\|\nabla f(x_t)\|^2 + \zeta^2) + \frac{\sigma^2}{n}.
\end{aligned}$$

By rearranging the terms we get the result. □

The following non-convex descent lemma is the key result used to establish convergence of both absolute compressors and δ-contraction operators.

Lemma 13. ***(Non-convex descent lemma)*** *Let Assumptions 1, 3, and 4 hold. If $\{x_t\}_{t\geq 0}$ denote the iterates of Algorithm 1 for a constant step-size, $\gamma \leq \dfrac{n}{2L(M(C+1)+n)}$, then*

$$\mathbb{E}[f(\tilde{x}_{t+1})]] \leq \mathbb{E}[f(\tilde{x}_t)] - \frac{\gamma}{4}\mathbb{E}\|\nabla f(x_t)\|^2 + \frac{\gamma^2 L(M\zeta^2 + \sigma^2)}{2n} + \frac{\gamma L^2}{2n}\sum_{i=1}^n \mathbb{E}\|e_{i,t}\|^2. \tag{16}$$

Proof. By using the L-smoothness of f and taking expectation we have

$$\begin{aligned}
\mathbb{E}_t[f(\tilde{x}_{t+1})] &\leq f(\tilde{x}_t) - \langle \nabla f(\tilde{x}_t), \mathbb{E}_t[\tilde{x}_{t+1} - \tilde{x}_t]\rangle + \frac{L}{2}\mathbb{E}_t\|\tilde{x}_{t+1} - \tilde{x}_t\|^2 \\
&= f(\tilde{x}_t) - \gamma\langle \nabla f(\tilde{x}_t), \nabla f(x_t)\rangle + \frac{\gamma^2 L}{2}\mathbb{E}_t\|\frac{1}{n}\sum_{i=1}^n g_{i,t}\|^2 \\
&\overset{(15)}{\leq} f(\tilde{x}_t) - \gamma\langle \nabla f(\tilde{x}_t), \nabla f(x_t)\rangle \\
&\quad + \frac{\gamma^2 L}{2}\left((1 + \frac{M(C+1)}{n})\|\nabla f(x_t)\|^2 + \frac{M\zeta^2}{n} + \frac{\sigma^2}{n}\right) \\
&\leq f(\tilde{x}_t) - \gamma\|\nabla f(x_t)\|^2 + \gamma\langle \nabla f(x_t) - \nabla f(\tilde{x}_t), \nabla f(x_t)\rangle \\
&\quad + \frac{\gamma^2 L(M(C+1)+n)}{2n}\|\nabla f(x_t)\|^2 + \frac{\gamma^2 L(M\zeta^2 + \sigma^2)}{2n} \\
&\overset{(11)}{\leq} f(\tilde{x}_t) - (\gamma - \frac{\gamma}{2} - \frac{\gamma^2 L(M(C+1)+n)}{2n})\|\nabla f(x_t)\|^2 + \\
&\quad \frac{\gamma\|\nabla f(x_t) - \nabla f(\tilde{x}_t)\|^2}{2} + \frac{\gamma^2 L(M\zeta^2 + \sigma^2)}{2n} \\
&\overset{\substack{\text{By } L-\text{smoothness} \\ \text{and } \gamma \leq \frac{n}{2L(M(C+1)+n)}}}{\leq} f(\tilde{x}_t) - \frac{\gamma\|\nabla f(x_t)\|^2}{4} + \frac{\gamma L^2\|x_t - \tilde{x}_t\|^2}{2} + \frac{\gamma^2 L(M\zeta^2 + \sigma^2)}{2n} \\
&= f(\tilde{x}_t) - \frac{\gamma\|\nabla f(x_t)\|^2}{4} + \frac{\gamma L^2\|\bar{e}_t\|^2}{2} + \frac{\gamma^2 L(M\zeta^2 + \sigma^2)}{2n} \\
&\overset{(12)}{\leq} f(\tilde{x}_t) - \frac{\gamma\|\nabla f(x_t)\|^2}{4} + \frac{\gamma L^2 \frac{1}{n}\sum_{i=1}^n\|e_{i,t}\|^2}{2} + \frac{\gamma^2 L(M\zeta^2 + \sigma^2)}{2n}.
\end{aligned}$$

Taking total expectation yields the lemma. □

Remark 10. Rearranging the terms in Lemma 13, performing telescopic sum, and noting that $\zeta = 0$ for $n = 1$, we get the result in Theorem 1.

B.3.1 Absolute compressors

Theorem. *6 (Non-convex convergence of absolute compressors) Let Assumptions 1, 2, 3, and 4 hold. Then the iterates, $\{x_t\}_{t\geq 0}$ of Algorithm 1 with an absolute compressor, $\mathcal{C}$ and a constant step-size, $\gamma \leq \frac{n}{2L(M(C+1)+n)}$, follow*

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\|\nabla f(x_t)\|^2 \leq \frac{4(f(x_0)-f^\star)}{\gamma T} + \frac{2\gamma L(M\zeta^2+\sigma^2)}{n} + 2\gamma^2 L^2 \upsilon^2.$$

Proof. By using Lemma 13, we have

$$\begin{aligned}\mathbb{E}[f(\tilde{x}_{t+1})] &\leq \mathbb{E}[f(\tilde{x}_t)] - \frac{\gamma\mathbb{E}\|\nabla f(x_t)\|^2}{4} + \frac{\gamma L^2 \frac{1}{n}\sum_{i=1}^n \mathbb{E}\|e_{i,t}\|^2}{2} + \frac{\gamma^2 L(M\zeta^2+\sigma^2)}{2n} \\ &\overset{\text{Remark 5}}{\leq} \mathbb{E}[f(\tilde{x}_t)] - \frac{\gamma\mathbb{E}\|\nabla f(x_t)\|^2}{4} + \frac{\gamma^3 L^2\upsilon^2}{2} + \frac{\gamma^2 L(M\zeta^2+\sigma^2)}{2n}.\end{aligned}$$

By taking summation over the iterates, we get

$$\begin{aligned}\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\|\nabla f(x_t)\|^2 &\leq \frac{4\sum_{t=0}^{T-1}(\mathbb{E}[f(\tilde{x}_t)] - \mathbb{E}[f(\tilde{x}_{t+1})])}{\gamma T} + \frac{2\gamma L(M\zeta^2+\sigma^2)}{n} + 2\gamma^2 L^2\upsilon^2 \\ &\leq \frac{4(f(x_0)-f^\star)}{\gamma T} + \frac{2\gamma L(M\zeta^2+\sigma^2)}{n} + 2\gamma^2 L^2\upsilon^2.\end{aligned}$$

Hence the result. □

B.3.2 δ-contraction operators

We now provide an error-bound for δ-contraction operators, which is an extension of the single node case in [54].

Lemma 14. *Let f follow Assumption 4 and the stochastic noise follow Assumptions 3. Define $e_{i,t}$ as in Algorithm 1. Then by using a δ-compressor, $\mathcal{C}$, with a constant step-size, $\gamma \leq \frac{1}{2L(2/\delta+M)\sqrt{C+1}}$, we have*

$$\sum_{t=0}^{T}\left[\frac{1}{n}\sum_{i=1}^n \mathbb{E}\|e_{i,t}\|^2\right] \leq \frac{1}{4L^2}\sum_{t=0}^{T}\mathbb{E}\|\nabla f(x_t)\|^2 + \frac{2\gamma^2(T+1)}{\delta}\left(\left(\frac{2}{\delta}+M\right)\zeta^2+\sigma^2\right). \quad (17)$$

Proof. We note that the compression operator, $\mathcal{C}$ and the stochastic noise, $\xi_{i,t}$ are independent of each other. Therefore, by taking expectation on the randomness of the compression operator, $\mathcal{C}$ in the following expression we have

$$\begin{aligned}\frac{1}{n}\sum_{i=1}^n \mathbb{E}_{\mathcal{C}}\|e_{i,t+1}\|^2 &= \frac{1}{n}\sum_{i=1}^n \mathbb{E}_{\mathcal{C}}\|e_{i,t}+\gamma g_{i,t} - \gamma\mathcal{C}(\frac{e_{i,t}}{\gamma}+g_{i,t})\|^2 \\ &\overset{\text{By (7)}}{\leq} \frac{1}{n}\sum_{i=1}^n \gamma^2(1-\delta)\|\frac{e_{i,t}}{\gamma}+g_{i,t}\|^2,\end{aligned}$$

which further by taking expectation conditioned on x_t becomes

$$\begin{aligned}
\frac{1}{n}\sum_{i=1}^{n}\mathbb{E}\left(\mathbb{E}_{\mathcal{C}}\|e_{i,t+1}\|^2|x_t\right) &\overset{\mathbb{E}[\xi_{i,t}|x_t]=0}{\leq} \frac{(1-\delta)}{n}\sum_{i=1}^{n}\|e_{i,t}+\gamma\nabla f_i(x_t)\|^2+\frac{(1-\delta)}{n}\sum_{i=1}^{n}\gamma^2\mathbb{E}\left[\|\xi_{i,t}\|^2|x_t\right] \\
&\overset{\text{Assumption 3}}{\leq} \frac{(1-\delta)}{n}\sum_{i=1}^{n}\|e_{i,t}+\gamma\nabla f_i(x_t)\|^2+\frac{(1-\delta)\gamma^2}{n}\sum_{i=1}^{n}\left(M\|\nabla f_i(x_t)\|^2+\sigma^2\right) \\
&\overset{(10)}{\leq} \frac{(1-\delta)(1+\rho)}{n}\sum_{i=1}^{n}\|e_{i,t}\|^2+\frac{(1-\delta)(1+\rho^{-1}+M)\gamma^2}{n}\sum_{i=1}^{n}\|\nabla f_i(x_t)\|^2 \\
&\quad +(1-\delta)\gamma^2\sigma^2 \\
&\overset{\text{Assumption 4}}{\leq} \frac{(1-\delta)(1+\rho)}{n}\sum_{i=1}^{n}\|e_{i,t}\|^2+\left((1-\delta)(1+\rho^{-1}+M)\gamma^2(C+1)\right)\|\nabla f(x_t)\|^2 \\
&\quad +\left((1-\delta)(1+\rho^{-1}+M)\gamma^2\zeta^2\right)+(1-\delta)\gamma^2\sigma^2 \\
&\leq \frac{(1-\delta)(1+\rho)}{n}\sum_{i=1}^{n}\|e_{i,t}\|^2 \\
&\quad +\gamma^2\left((1+\rho^{-1}+M)(C+1)\|\nabla f(x_t)\|^2+(1+\rho^{-1}+M)\zeta^2+\sigma^2\right).
\end{aligned}$$

By unrolling the recurrence, taking total expectation, setting $\rho=\frac{\delta}{2(1-\delta)}$, such that $(1+\rho^{-1})=\frac{2-\delta}{\delta}\leq\frac{2}{\delta}$ and $(1-\delta)(1+\rho)\leq(1-\frac{\delta}{2})$, and using the fact that $e_{i,0}=0$, for all i, we find

$$\begin{aligned}
\frac{1}{n}\sum_{i=1}^{n}\mathbb{E}\|e_{i,t+1}\|^2 &\leq \gamma^2\sum_{i=0}^{t}[(1-\delta)(1+\rho)]^{t-i}\left(\left(\frac{2}{\delta}+M\right)(C+1)\mathbb{E}\|\nabla f(x_i)\|^2+\left(\frac{2}{\delta}+M\right)\zeta^2+\sigma^2\right) \\
&\leq \gamma^2\sum_{i=0}^{t}(1-\frac{\delta}{2})^{t-i}\left(\left(\frac{2}{\delta}+M\right)(C+1)\mathbb{E}\|\nabla f(x_i)\|^2+\left(\frac{2}{\delta}+M\right)\zeta^2+\sigma^2\right).
\end{aligned}$$

Finally,

$$\begin{aligned}
\sum_{t=0}^{T}\left[\frac{1}{n}\sum_{i=1}^{n}\mathbb{E}\|e_{i,t}\|^2\right] &= \gamma^2\sum_{t=0}^{T}\sum_{i=0}^{t-1}(1-\frac{\delta}{2})^{t-1-i}\left(\left(\frac{2}{\delta}+M\right)(C+1)\mathbb{E}\|\nabla f(x_i)\|^2+\left(\frac{2}{\delta}+M\right)\zeta^2+\sigma^2\right) \\
&\leq \gamma^2\sum_{t=0}^{T-1}\sum_{j=0}^{T-t-1}(1-\frac{\delta}{2})^{j}\left(\left(\frac{2}{\delta}+M\right)(C+1)\mathbb{E}\|\nabla f(x_t)\|^2+\left(\frac{2}{\delta}+M\right)\zeta^2+\sigma^2\right) \\
&\leq \gamma^2\sum_{t=0}^{T-1}\left(\left(\frac{2}{\delta}+M\right)(C+1)\mathbb{E}\|\nabla f(x_t)\|^2+\left(\frac{2}{\delta}+M\right)\zeta^2+\sigma^2\right)\sum_{j=0}^{\infty}(1-\frac{\delta}{2})^{j} \\
&= \gamma^2\sum_{t=0}^{T-1}\left(\frac{2}{\delta}\right)\left(\left(\frac{2}{\delta}+M\right)(C+1)\mathbb{E}\|\nabla f(x_t)\|^2+\left(\frac{2}{\delta}+M\right)\zeta^2+\sigma^2\right) \\
&\leq \gamma^2\sum_{t=0}^{T}\left(\frac{2}{\delta}\right)\left(\left(\frac{2}{\delta}+M\right)(C+1)\mathbb{E}\|\nabla f(x_t)\|^2+\left(\frac{2}{\delta}+M\right)\zeta^2+\sigma^2\right) \\
&= \sum_{t=0}^{T}\left(\gamma^2\left(\frac{2}{\delta}\right)\left(\frac{2}{\delta}+M\right)(C+1)\mathbb{E}\|\nabla f(x_t)\|^2\right)+\sum_{t=0}^{T}\frac{2\gamma^2}{\delta}\left(\left(\frac{2}{\delta}+M\right)\zeta^2+\sigma^2\right).
\end{aligned}$$

Choosing $\gamma\leq\frac{1}{2L(2/\delta+M)\sqrt{C+1}}$, we get $\gamma^2\left(\frac{2}{\delta}\right)\left(\frac{2}{\delta}+M\right)\leq\frac{1}{4L^2(C+1)}$. Combining all together we have

$$\sum_{t=0}^{T}\left[\frac{1}{n}\sum_{i=1}^{n}\mathbb{E}\|e_{i,t}\|^2\right]\leq\frac{1}{4L^2}\sum_{t=0}^{T}\mathbb{E}\|\nabla f(x_t)\|^2+\frac{2\gamma^2(T+1)}{\delta}\left(\left(\frac{2}{\delta}+M\right)\zeta^2+\sigma^2\right).$$

Hence the result. □

By using the previous bound, we now provide the non-convex convergence result for δ-contraction operators.

Theorem. *7 (**Non-convex convergence of δ-contraction operators**) Let Assumptions 1, 2, 3, and 4 hold. Then the iterates, $\{x_t\}_{t\geq 0}$ of Algorithm 1 with a δ-contraction operator and a constant step-size $\gamma \leq \min\{\frac{n}{2L(M(C+1)+n)}, \frac{1}{2L(2/\delta+M)\sqrt{C+1}}\}$ follow*

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\|\nabla f(x_t)\|^2 \leq \frac{8(f(x_0)-f^\star)}{\gamma T} + \frac{4\gamma L(M\zeta^2+\sigma^2)}{n} + \frac{8\gamma^2L^2}{\delta}\left(\left(\frac{2}{\delta}+M\right)\zeta^2+\sigma^2\right).$$

Proof. Summing over the iterates $t=0$ to $t=T-1$ in (16) of Lemma 13, we have

$$\begin{aligned}\mathbb{E}[f(\tilde{x}_T)] \;&\leq\; f(x_0) - \frac{\sum_{t=0}^{T-1}\gamma\mathbb{E}\|\nabla f(x_t)\|^2}{4} + \frac{\gamma L^2\sum_{t=0}^{T-1}\frac{1}{n}\sum_{i=1}^n\mathbb{E}\|e_{i,t}\|^2}{2} + \sum_{t=0}^{T-1}\frac{\gamma^2L(M\zeta^2+\sigma^2)}{2n}\\ &\overset{(17)}{\leq} f(x_0) - (\frac{\gamma}{4}-\frac{\gamma}{8})\sum_{t=0}^{T-1}\mathbb{E}\|\nabla f(x_t)\|^2 + \frac{\gamma^3L^2T}{\delta}\left(\left(\frac{2}{\delta}+M\right)\zeta^2+\sigma^2\right) + \frac{\gamma^2TL(M\zeta^2+\sigma^2)}{2n}.\end{aligned}$$

Rearranging, we get

$$\begin{aligned}\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\|\nabla f(x_t)\|^2 &\leq \frac{8(f(x_0)-\mathbb{E}[f(\tilde{x}_t)])}{\gamma T} + \frac{4\gamma L(M\zeta^2+\sigma^2)}{n} + \frac{8\gamma^2L^2}{\delta}\left(\left(\frac{2}{\delta}+M\right)\zeta^2+\sigma^2\right)\\ &\leq \frac{8(f(x_0)-f^\star)}{\gamma T} + \frac{4\gamma L(M\zeta^2+\sigma^2)}{n} + \frac{8\gamma^2L^2}{\delta}\left(\left(\frac{2}{\delta}+M\right)\zeta^2+\sigma^2\right).\end{aligned}$$

Hence the result. □

B.3.3 Uncompressed SGD

We provide the convergence result of no-compression SGD (Algorithm 1 with an identity compressor, i.e., $\mathcal{C}(x)=x$ for all $x\in\mathbb{R}^d$).

Theorem 8. ***(Non-convex convergence of SGD)** Let Assumptions 1, 2, 3, and 4 hold. Then the iterates, $\{x_t\}_{t\geq 0}$ of Algorithm 1 by using an identity compressor ($\mathcal{C}(x)=x$, for all $x\in\mathbb{R}^d$) with a constant step-size, $\gamma \leq \frac{n}{L(M(C+1)+n)}$ follow*

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\|\nabla f(x_t)\|^2 \leq \frac{2(f(x_0)-f^\star)}{\gamma T} + \frac{\gamma L(M\zeta^2+\sigma^2)}{n}.$$

Proof. We use the L-smoothness of f to find

$$\begin{aligned}\mathbb{E}_t[f(x_{t+1})] \quad &\leq \quad f(x_t) - \langle\nabla f(x_t), \mathbb{E}_t[x_{t+1}-x_t]\rangle + \frac{L}{2}\mathbb{E}_t\|x_{t+1}-x_t\|^2\\ &= \quad f(x_t) - \gamma\left\langle\nabla f(x_t), \mathbb{E}_t[\bar{g}_t]\right\rangle + \frac{\gamma^2L}{2}\mathbb{E}_t\|\bar{g}_t\|^2\\ &= \quad f(x_t) - \gamma\|\nabla f(x_t)\|^2 + \frac{\gamma^2L}{2}\mathbb{E}_t\|\frac{1}{n}\sum_{i=1}^n g_{i,t}\|^2\\ &\overset{(15)}{\leq} \quad f(x_t) - \gamma\|\nabla f(x_t)\|^2 + \frac{\gamma^2L}{2}\left((1+\frac{M(C+1)}{n})\|\nabla f(x_t)\|^2 + \frac{M\zeta^2}{n} + \frac{\sigma^2}{n}\right)\\ &= \quad f(x_t) - \gamma\left(1-\frac{\gamma L(M(C+1)+n)}{2n}\right)\|\nabla f(x_t)\|^2 + \frac{\gamma^2L(M\zeta^2+\sigma^2)}{2n}\\ &\overset{\gamma\leq\frac{n}{L(M(C+1)+n)}}{\leq} f(x_t) - \frac{\gamma}{2}\|\nabla f(x_t)\|^2 + \frac{\gamma^2L(M\zeta^2+\sigma^2)}{2n}.\end{aligned}$$

By summing over the iterates and taking total expectation, we get

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\|\nabla f(x_t)\|^2 \leq \frac{2(f(x_0)-f^\star)}{\gamma T} + \frac{\gamma L(M\zeta^2+\sigma^2)}{n}.$$

Hence the result. □

B.3.4 Final convergence result

From Remark 9, the following corollary describes the $\mathcal{O}(1/\sqrt{nT})$ convergence with an appropriate step-size for absolute compressors and δ-contraction operators.

Corollary 1. *Let Assumptions 1, 2, 3, and 4 hold with $M\zeta^2+\sigma^2>0$ and let $\{x_t\}_{t\geq 0}$ denote the iterates of algorithm 1. Then, if*

• $\mathcal{C}$ is an absolute compressor, we have

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\|\nabla f(x_t)\|^2=\mathcal{O}\left(\frac{\sqrt{L(M\zeta^2+\sigma^2)(f(x_0)-f^\star)}}{\sqrt{nT}}+\frac{L((\frac{M}{n}(C+1)+1)+\frac{nv^2}{M\zeta^2+\sigma^2})(f(x_0)-f^\star)}{T}\right).$$

• $\mathcal{C}$ is a δ-contraction operator, we have

$$\tfrac{1}{T}\textstyle\sum_{t=0}^{T-1}\mathbb{E}\|\nabla f(x_t)\|^2=\mathcal{O}\left(\frac{\sqrt{(L(M\zeta^2+\sigma^2))(f(x_0)-f^\star)}}{\sqrt{nT}}+\frac{L\left(\max\{\frac{M}{n}(C+1)+1),(\frac{1}{\delta}+M)\sqrt{C+1}\}+\frac{n\left((1+M\delta)\zeta^2+\delta\sigma^2\right)}{\delta^2(M\zeta^2+\sigma^2)}\right)(f(x_0)-f^\star)}{T}\right).$$

• $\mathcal{C}$ is the identity compressor, we have

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\|\nabla f(x_t)\|^2=\mathcal{O}\left(\frac{\sqrt{L(M\zeta^2+\sigma^2)(f(x_0)-f^\star)}}{\sqrt{nT}}+\frac{L(\frac{M}{n}(C+1)+1)(f(x_0)-f^\star)}{T}\right).$$

Proof. Invoking Lemma 7 in Theorem 6 and Theorem 7, and Lemma 6 in Theorem 8 we get the results. □

We note that the above results are for the cases with $M\zeta^2+\sigma^2>0$. If $M\zeta^2+\sigma^2=0$, i.e. a non-stochastic setting, then one can derive the convergence result using Lemma 8.

While compression does not affect the slower decaying $\mathcal{O}(1/\sqrt{nT})$ term for both absolute compressors and δ-contraction operators, we observe δ-contraction operators have $1/\delta^2$ dependence in the $\mathcal{O}(1/T)$ term when $\zeta\neq 0$ (heterogeneous data). Therefore, in this setting, the Top-k sparsifier has d^2/k^2 in the numerator of $\mathcal{O}(1/T)$ term. On the other hard, hard-threshold has $d\lambda^2$ in the numerator of $\mathcal{O}(1/T)$ term even when $\zeta\neq 0$, and thus has a significantly better dependence on d.

B.4 Convex convergence analysis

In this Section, we provide convergence results for *distributed compressed SGD* with *absolute compressors* and an *EF* where the loss function on each worker f_i is μ-strongly convex with $\mu\geq 0$ (see Assumption 5). Our analysis is inspired by the proof techniques in [54] which analyzes an EF SGD with δ-contraction operators in the single node ($n=1$) case. [10] extended this analysis to the distributed ($n>1$) case for δ-contraction operators.

We start with the following key result by Nesterov [41] for convex and smooth functions.

Lemma 15. *Let f_i follow Assumptions 1 and Assumption 5 with $\mu\geq 0$, then*

$$\|\nabla f_i(y)-\nabla f_i(x)\|^2\leq 2L(f_i(y)-f_i(x)-\langle\nabla f_i(x),y-x\rangle),\qquad\forall x,y\in\mathbb{R}^d. \tag{18}$$

We start with the convex decent lemma from [10]. For completeness, we also provide the proof.

Lemma 16. ***(Convex descent lemma)*** *(Lemma 21 in [10]) Let Assumptions 1, 2, 3, and 5 hold. Denote $D:=\frac{1}{n}\sum_{i=1}^n\|\nabla f_i(x^\star)\|^2$. If $\gamma_t\leq\frac{1}{4L(1+2M/n)}$, for all $t\geq 0$, then the iterates, $\{\tilde{x}_t\}_{t\geq 0}$ of Algorithm 1 follow*

$$\mathbb{E}_t\|\tilde{x}_{t+1}-x^\star\|^2\leq(1-\frac{\mu\gamma_t}{2})\|\tilde{x}_t-x^\star\|^2-\frac{\gamma_t}{2}[f(x_t)-f^\star]+3L\gamma_t\|x_t-\tilde{x}_t\|^2+(\gamma_t^2)\frac{\sigma^2+2MD}{n}.$$

Proof. We have

$$\begin{aligned}\|\tilde{x}_{t+1}-x^\star\|^2&\overset{\text{Lemma 11}}{=}\|\tilde{x}_t-x^\star\|^2-2\gamma_t\left\langle\bar{g}_t,\tilde{x}_t-x^\star\right\rangle+\gamma_t^2\|\bar{g}_t\|^2\\&=\|\tilde{x}_t-x^\star\|^2-2\gamma_t\left\langle\bar{g}_t,x_t-x^\star\right\rangle+\gamma_t^2\|\bar{g}_t\|^2+2\gamma_t\left\langle\bar{g}_t,x_t-\tilde{x}_t\right\rangle.\end{aligned}$$

Therefore,

$$\begin{aligned}\mathbb{E}_t\|\tilde{x}_{t+1}-x^\star\|^2 &= \|\tilde{x}_t-x^\star\|^2 - 2\gamma_t\left\langle \mathbb{E}_t[\bar{g}_t], x_t-x^\star\right\rangle + \gamma_t^2\mathbb{E}_t\|\bar{g}_t\|^2 + 2\gamma_t\left\langle \mathbb{E}_t[\bar{g}_t], x_t-\tilde{x}_t\right\rangle \\ &= \|\tilde{x}_t-x^\star\|^2 - 2\gamma_t\left\langle \nabla f(x_t), x_t-x^\star\right\rangle + \gamma_t^2\mathbb{E}_t\|\bar{g}_t\|^2 + 2\gamma_t\left\langle \nabla f(x_t), x_t-\tilde{x}_t\right\rangle. \end{aligned} \tag{19}$$

First, we bound $2\left\langle \nabla f(x_t), x_t-\tilde{x}_t\right\rangle$. We use Young's inequality (11) with $\rho=\frac{1}{2L}$ and get

$$\begin{aligned}2\left\langle \nabla f(x_t), x_t-\tilde{x}_t\right\rangle &\leq \frac{1}{2L}\|\nabla f(x_t)\|^2 + 2L\|x_t-\tilde{x}_t\|^2 \\ &\overset{(18),\nabla f(x^\star)=0}{\leq} f(x_t)-f(x^\star)+2L\|x_t-\tilde{x}_t\|^2. \end{aligned} \tag{20}$$

Next, we bound $-2\left\langle \nabla f(x_t), x_t-x^\star\right\rangle$. We use the μ-strong convexity of f to find

$$-2\left\langle \nabla f(x_t), x_t-x^\star\right\rangle \leq 2(f(x^\star)-f(x_t)) - \mu\|x_t-x^\star\|^2. \tag{21}$$

However, since we want to work with $\|\tilde{x}_t-x^\star\|^2$ instead of $\|x_t-x^\star\|^2$, we get rid of $\|x_t-x^\star\|^2$ using (10) with $\rho=1$ as

$$\|x_t-x^\star\|^2 \geq \frac{1}{2}\|\tilde{x}_t-x^\star\|^2 - \|x_t-\tilde{x}_t\|^2.$$

Substituting this in Equation (21), we get

$$-2\left\langle \nabla f(x_t), x_t-x^\star\right\rangle \leq 2(f(x^\star)-f(x_t)) - \frac{\mu}{2}\|\tilde{x}_t-x^\star\|^2 + \mu\|x_t-\tilde{x}_t\|^2. \tag{22}$$

Finally, we bound $\mathbb{E}_t\|\bar{g}_t\|^2$ as

$$\begin{aligned}\mathbb{E}_t\|\frac{1}{n}\sum_{i=1}^n g_{i,t}\|^2 &= \mathbb{E}\left[\|\frac{1}{n}\sum_{i=1}^n(\nabla f_i(x_t)+\xi_{i,t})\|^2|x_t\right] \\ &= \mathbb{E}\left[\|\nabla f(x_t)+\frac{1}{n}\sum_{i=1}^n\xi_{i,t}\|^2|x_t\right] \\ &\overset{\mathbb{E}[\xi_{i,t}|x_t]=0}{=} \|\nabla f(x_t)\|^2 + \mathbb{E}\left[\|\frac{1}{n}\sum_{i=1}^n\xi_{i,t}\|^2|x_t\right] \\ &\overset{\mathbb{E}[\xi_{i,t}|x_t]=0}{=} \|\nabla f(x_t)\|^2 + \frac{1}{n^2}\sum_{i=1}^n\mathbb{E}\left[\|\xi_{i,t}\|^2|x_t\right] \\ &\overset{\text{Assumption 3}}{\leq} \|\nabla f(x_t)\|^2 + \frac{1}{n^2}\sum_{i=1}^n(M\|\nabla f_i(x_t)\|^2+\sigma^2) \\ &= \|\nabla f(x_t)\|^2 + \frac{M}{n^2}\sum_{i=1}^n\|\nabla f_i(x_t)-\nabla f_i(x^\star)+\nabla f_i(x^\star)\|^2 + \frac{\sigma^2}{n} \\ &\leq \|\nabla f(x_t)\|^2 + \frac{2M}{n^2}\sum_{i=1}^n\left(\|\nabla f_i(x_t)-\nabla f_i(x^\star)\|^2+\|\nabla f_i(x^\star)\|^2\right) \\ &\quad + \frac{\sigma^2}{n} \\ &\overset{(18),D=\frac{1}{n}\sum_{i=1}^n\|\nabla f_i(x^\star)\|^2}{\leq} \|\nabla f(x_t)\|^2 + \frac{2M}{n^2}\sum_{i=1}^n 2L[f_i(x_t)-f_i(x^\star)-\langle\nabla f_i(x^\star), x_t-x^\star\rangle] \\ &\quad + \frac{2MD}{n} + \frac{\sigma^2}{n} \end{aligned} \tag{23}$$

$$\begin{aligned} &\overset{\nabla f(x^\star)=0}{=} \|\nabla f(x_t)-\nabla f(x^\star)\|^2 + \frac{4LM}{n}(f(x_t)-f(x^\star)) + \frac{2MD+\sigma^2}{n} \\ &\overset{(18),\nabla f(x^\star)=0}{\leq} 2L\left(1+\frac{2M}{n}\right)(f(x_t)-f(x^\star)) + \frac{2MD+\sigma^2}{n}. \end{aligned} \tag{24}$$

We now substitute (20), (22), and (24) in (19) to get

$$\begin{aligned}\mathbb{E}_t\|\tilde{x}_{t+1}-x^\star\|^2 &= \|\tilde{x}_t-x^\star\|^2 - 2\gamma_t\left\langle \nabla f(x_t), x_t-x^\star\right\rangle + \gamma_t^2\mathbb{E}_t\|\bar{g}_t\|^2 + 2\gamma_t\left\langle \nabla f(x_t), x_t-\tilde{x}_t\right\rangle \\ &\leq \left(1-\frac{\mu\gamma_t}{2}\right)\|\tilde{x}_t-x^\star\|^2 - \gamma_t\left(1-\gamma_t\cdot 2L\left(1+\frac{2M}{n}\right)\right)(f(x_t)-f(x^\star)) \\ &\quad +\gamma_t(2L+\mu)\|x_t-\tilde{x}_t\|^2 + \gamma_t^2\frac{2MD+\sigma^2}{n}.\end{aligned}$$

Choosing $\gamma_t \leq \frac{1}{4L(1+2M/n)}$ gives the desired result. □

Next, we give the convex convergence result of *distributed EF SGD* with *absolute compressors*.

B.4.1 Absolute compressors

The next theorem combines the results of Theorems 4 and 5 from the main paper. We present them as a single theorem (Theorem 9) to keep the structure of the proofs simple.

Theorem 9. *Let Assumptions 1, 2, 3, and 5 hold. Denote $D := \frac{1}{n}\sum_{i=1}^n\|\nabla f_i(x^\star)\|^2$, and $R_0 = \|x_0-x^\star\|^2$. Then the iterates, $\{x_t\}_{t\geq 0}$ of Algorithm 1 with an absolute compressor, $\mathcal{C}_\upsilon$ have the following convergence rates if Assumption 5 is satisfied with the following choices of the parameters:*

*i) **(Theorem 4)** If $\mu > 0$, a constant step-size $\{\gamma_t = \gamma\}_{t\geq 0}$, with $\gamma \leq \frac{1}{4L(1+2M/n)}$ is chosen as in Lemma 10 and weights $\{w_t = (1-\mu\gamma/2)^{-(t+1)}\}_{t\geq 0}$ then*

$$\mathbb{E}[f(\bar{x}_T)] - f^\star = \tilde{\mathcal{O}}\left(L(1+M/n)R_0\exp\left[-\frac{\mu T}{8L(1+2M/n)}\right] + \frac{\sigma^2+MD}{\mu nT} + \frac{L\upsilon^2}{\mu^2T^2}\right).$$

*ii) **(Theorem 5)** If $\mu = 0$, a constant step-size $\{\gamma_t = \gamma\}_{t\geq 0}$, with $\gamma \leq \frac{1}{4L(1+2M/n)}$ is chosen as in Lemma 7 and weights $\{w_t = 1\}_{t\geq 0}$ then*

$$\mathbb{E}[f(\bar{x}_T)] - f^\star = \mathcal{O}\left(\frac{\sqrt{(\sigma^2+MD)R_0}}{\sqrt{nT}} + \frac{\left(\frac{nL\upsilon^2}{\sigma^2+MD} + L(1+M/n)\right)R_0}{T}\right).$$

iii) If $\mu > 0$, step-sizes $\{\gamma_t = \frac{4}{\mu(\phi+t)}\}_{t\geq 0}$, and weights $\{w_t = \phi+t\}_{t\geq 0}$, respectively with $\phi = \frac{16L}{\mu}(1+\frac{2M}{n})$ then

$$\mathbb{E}[f(\bar{x}_T)] - f^\star = \mathcal{O}\left(\frac{\sigma^2+MD}{\mu nT} + \frac{\mu L^2(1+M/n)^2R_0 + L\upsilon^2\ln(T)}{\mu^2T^2}\right).$$

In the above, $\bar{x}_T = \frac{1}{W_T}\sum_{t=0}^T w_t x_t$, and $W_T = \sum_{t=0}^T w_t$.

Proof. By using Lemma 11 in Lemma 16, and taking total-expectation over all the previous iterates, we have

$$\begin{aligned}\mathbb{E}\|\tilde{x}_{t+1}-x^\star\|^2 &\leq (1-\frac{\mu\gamma_t}{2})\mathbb{E}\|\tilde{x}_t-x^\star\|^2 - \frac{\gamma_t}{2}\mathbb{E}[f(x_t)-f^\star] + 3L\gamma_t\mathbb{E}\|\bar{e}_t\|^2 + \gamma_t^2(\frac{\sigma^2+2MD}{n}) \\ &\overset{(12)}{\leq} (1-\frac{\mu\gamma_t}{2})\mathbb{E}\|\tilde{x}_t-x^\star\|^2 - \frac{\gamma_t}{2}\mathbb{E}[f(x_t)-f^\star] + 3L\gamma_t\sum_{i=1}^n\frac{1}{n}\mathbb{E}\|e_{i,t}\|^2 \qquad (25) \\ &\quad +\gamma_t^2(\frac{\sigma^2+2MD}{n}) \qquad (26) \\ &\overset{\text{Remark 5}}{\leq} (1-\frac{\mu\gamma_t}{2})\mathbb{E}\|\tilde{x}_t-x^\star\|^2 - \frac{\gamma_t}{2}\mathbb{E}[f(x_t)-f^\star] + 3L\gamma_t^3\upsilon^2 + \gamma_t^2(\frac{\sigma^2+2MD}{n}).\end{aligned}$$

Rearranging, we get

$$\mathbb{E}[f(x_t)]-f^\star \leq \frac{2}{\gamma_t}(1-\frac{\mu\gamma_t}{2})\mathbb{E}\|\tilde{x}_t-x^\star\|^2 - \frac{2}{\gamma_t}\mathbb{E}\|\tilde{x}_{t+1}-x^\star\|^2 + \gamma_t\frac{2\sigma^2+4MD}{n} + 6L\gamma_t^2\upsilon^2. \quad (27)$$

With $r_t = 2\mathbb{E}\|\tilde{x}_t - x^\star\|^2$, $a = \frac{\mu}{2}$, $c = \frac{2\sigma^2+4MD}{n}$, $b = 6L\upsilon^2$, we can see the RHS as $\frac{1}{\gamma_t}(1-a\gamma_t)r_t - \frac{1}{\gamma_t}r_{t+1} + c\gamma_t + b\gamma_t^2$. Thus, we use Lemma 10 and Lemma 9 to get the first and the third result respectively. Note that to get the LHS, we use the convexity of f as $\frac{1}{W_T}\sum_{t=0}^{T} w_t f(x_t) \geq f(\bar{x}_T)$. Finally, to get the second result, we substitute $\mu = 0$ in Equation (27) and perform telescopic sum to get

$$\frac{\sum_{t=0}^{T}\mathbb{E}[f(x_t)]}{T+1} - f^\star \leq \frac{2\|x_0 - x^\star\|^2}{\gamma(T+1)} + \frac{2\sigma^2+4MD}{n}\gamma + 6L\upsilon^2\gamma^2.$$

We now use Lemma 7 and convexity of f to arrive at the desired result. Similarly, for the result of Remark 7, we use Lemma 8.

□

B.4.2 δ-contraction operators

The rates for δ-contraction operators is based on [22], except we consider a slightly different set of assumptions. Below we provide the sketch of the proof.

First, using equation (18), we can have

$$\begin{aligned}
\frac{1}{n}\sum_{i=1}^{n}\|\nabla f_i(x_t)\|^2 &= \frac{1}{n}\sum_{i=1}^{n}\|\nabla f_i(x_t)\|^2 \\
&= \frac{1}{n}\sum_{i=1}^{n}\|\nabla f_i(x_t) - \nabla f_i(x^\star) + \nabla f_i(x^\star)\|^2 \\
&\leq \frac{2}{n}\sum_{i=1}^{n}\|\nabla f_i(x_t) - \nabla f_i(x^\star)\|^2 + \frac{2}{n}\sum_{i=1}^{n}\|\nabla f_i(x^\star)\|^2 \\
&\overset{(18)}{\leq} 4L(f(x_t) - f^\star) + 2D. \qquad (28)
\end{aligned}$$

Second, from Assumption 3, we have

$$\begin{aligned}
\frac{1}{n}\sum_{i=1}^{n}\mathbb{E}[\|\xi_{i,t}\|^2 \mid x_t] &\leq \frac{M}{n}\sum_{i=1}^{n}\|\nabla f_i(x_t)\|^2 + \sigma^2 \\
&\overset{(28)}{\leq} 4LM(f(x_t) - f^\star) + 2MD + \sigma^2. \qquad (29)
\end{aligned}$$

Third, from (24), we have

$$\mathbb{E}\left[\|\frac{1}{n}\sum_{i=1}^{n} g_{i,t}\|^2 | x_t\right] \leq 2L\left(1 + \frac{2M}{n}\right)(f(x_t) - f(x^\star)) + \frac{2MD+\sigma^2}{n}. \qquad (30)$$

Using (28), (29), and (30), we can show that Assumption 3.3 in [22] is satisfied with $A = 2L$, $D_1 = 2D$, $\tilde{A} = 2LM$, $\tilde{D_1} = 2MD + \sigma^2$, $A' = L(1 + \frac{2M}{n})$, $D_1' = \frac{2MD+\sigma^2}{n}$, $\rho_1 = \rho_2 = 1$, and all the other quantities as zero. Then, using Lemma G.1 in [22] with $\gamma \leq \frac{\delta}{8L\sqrt{3(2+M\delta)}}$, we can show that Assumption 3.4 in [22] is satisfied with $F_1 = 0$, $F_2 = 0$, and $D_3 = \frac{6L\gamma}{\delta^2}\left(D(4+2M\delta) + \delta\sigma^2\right)$. We subsequently use (25), followed by Lemma 10 for the strongly-convex case (Remark 6), and Lemma 7 for the convex case (Remark 8).

B.5 Comparison against unbiased compressors

Till now, we have discussed the convergence of compressed SGD using EF. However, unbiased relative compressors which satisfy (*i*) $\mathbb{E}_{\mathcal{C}}[\mathcal{C}(x)] = x$; and (*ii*) $\mathbb{E}_{\mathcal{C}}\|\mathcal{C}(x) - x\|^2 \leq \Omega\|x\|^2$ do not require EF. We compare the convergence of such unbiased compressors and absolute compressors with EF. With the notations above, [29] provide the following convergence result for unbiased compressors in the strongly convex case:

$$\mathbb{E}[f(\bar{x}_T)] - f^\star + \mu\mathbb{E}[\|x_T - x^*\|^2] \leq 64\Omega_n L(1+M/n)R_0 \exp\left[-\frac{\mu T}{4\Omega_n L(1+M/n)}\right] + 36\frac{(\Omega_n - 1)D + \Omega\sigma^2/n}{\mu T},$$

Table 2: Summary of the benchmarks used

Model	Task	Dataset	No. of Parameters	Optimizer
ResNet-18 [26]	Image classification	CIFAR-10 [35]	11,173,962	SGD+Nesterov momentum
LSTM [28]	Language modelling	Wikitext-2 [39]	28,949,319	Vanilla SGD
NCF [27]	Recommendation	Movielens-20M	31,832,577	ADAM [34]

where $\Omega_n = \frac{\Omega-1}{n} + 1$. Comparing with Theorem 4, we find unbiased compressors have compression affecting the slower-decaying $\frac{1}{T}$ term. Although, we note that their convergence is in both the iterates and functional values, whereas ours is only in functional values.

C Addendum to numerical experiments

Overview. In this section, we provide:

i) The experimental settings and implementation details of our DNN experiments (§C.1).

ii) Further discussion on the large error-accumulation of Top-k and its effect on total-error (§C.2).

iii) Logistic regression experiments (§C.3).

iv) Comparison against the state-of-the-art adaptive sparsifier ACCORDION [3]. (§C.4)

v) Experiment with Entire-model Top-k (§C.5).

vi) Experiments without EF, and discussion on different forms of EF (§C.6).

C.1 Experimental settings and implementation details

We implement the sparsifiers in `PyTorch`. For each method, a gradient reducer class is defined, which invokes the appropriate compression function and then perform the aggregation among the workers. Tables 2, 3, 4, and 5 provide the experimental details for each of the tasks. We used the default hyper-parameters provided in the mentioned repositories for each task.

Table 3: Image classification task

Dataset	CIFAR-10
Architecture	ResNet-18
Repository	PowerSGD [57] See `https://github.com/epfml/powersgd`
License	MIT
Number of workers	8
Global Batch-size	256×8
Optimizer	SGD with Nesterov Momentum
Momentum	0.9
Post warmup LR	0.1×16
LR-decay	$/10$ at epoch 150 and 250
LR-warmup	Linearly within 5 epochs, starting from 0.1
Number of Epochs	300
Weight decay	10^{-4}
Repetitions	3, with different seeds
Hard-threshold: λ values	$\{1.2 \times 10^{-2}, 7.2 \times 10^{-3}, 5 \times 10^{-3}, 3 \times 10^{-3}, 1.8 \times 10^{-3}\}$
Top-k: k values	$\{0.03\%, 0.06\%, 0.12\%, 0.3\%, 0.75\%\}$

C.2 Top-k suffers from large error accumulation

In Figure 4, we show the cascading effect (mentioned in §4.5) for the experiment in Figure 1. We observe that the error norm profile in Figure4 c closely follows the error compensated gradient norm profile in Figure4 b.

Table 4: Language modelling task

Dataset	WikiText2
Architecture	LSTM
Repository	PowerSGD [57]
	See `https://github.com/epfml/powersgd`
License	MIT
Number of workers	8
Global Batch-size	128×8
Optimizer	vanilla SGD
Post warmup LR	1.25×16
LR-decay	$/10$ at epoch 60 and 80
LR-warmup	Linearly within 5 epochs, starting from 1.25
Number of Epochs	90
Weight decay	0
Repetitions	3, with different seeds
Hard-threshold: λ values	$\{4.5 \times 10^{-3}, 2.75 \times 10^{-3}, 1.6 \times 10^{-3}, 1.12 \times 10^{-3}\}$
Top-k: k values	$\{0.025\%, 0.05\%, 0.1\%, 0.2\%\}$

Table 5: Recommendation task

Dataset	Movielens-20M
Architecture	NCF
Repository	NVIDIA Deep Learning Examples
	See `https://github.com/NVIDIA/DeepLearningExamples`
Number of workers	8
Global Batch-size	2^{20}
Optimizer	ADAM
ADAM β_1	0.25
ADAM β_2	0.5
ADAM LR	4.5×10^{-3}
Number of Epochs	30
Weight decay	0
Dropout	0.5
Repetitions	3, with different seeds
Hard-threshold: λ values	$\{2 \times 10^{-6}, 1.3 \times 10^{-6}, 1 \times 10^{-6}, 4 \times 10^{-7}\}$
Top-k: k values	$\{7.7\%, 9.5\%, 11.3\%, 13.7\%\}$
License	Open Source

In Figure 5 and Figure 6, we show that hard-threshold has a better convergence because of a smaller total-error in LSTM-WikiText2 and NCF-Ml-20m benchmarks. We note that we use the ADAM optimizer on the NCF-Ml-20m benchmark, and therefore our total-error insight is not theoretically justified in this case. Nevertheless, our experiment empirically confirms that the total-error perspective is useful for optimizers beyond vanilla SGD and momentum SGD.

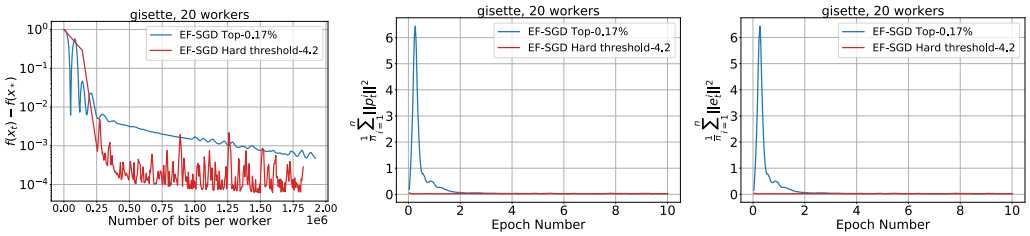
C.3 Logistic regression experiments

For the convex experiments, we consider the following ℓ_2 regularized logistic regression experiment considered in [22][4]:

$$\min_{x \in \mathbb{R}^d} f(x) = \frac{1}{N} \sum_{i=1}^{N} \log(1 + \exp(-y_i \mathbf{A}[i,:]x)) + \frac{\mu}{2}\|x\|^2, \quad \text{where } \mathbf{A} \in \mathbb{R}^{N \times d}, y \in \mathbb{R}^N. \tag{31}$$

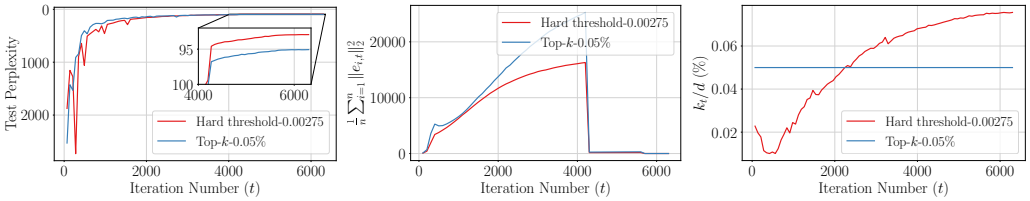
The function, $f(x)$ in (31) is μ-strongly convex and L-smooth with $L = \mu + \frac{\lambda_{\max}(\mathbf{A}^T\mathbf{A})}{4N}$. As in [22], we use the step-size $\gamma = 1/L$, and $\mu = 10^{-4}\frac{\lambda_{\max}(\mathbf{A}^T\mathbf{A})}{4N}$. We use standard LIBSVM datasets [14],

[4]Open source code: `https://github.com/eduardgorbunov/ef_sigma_k`



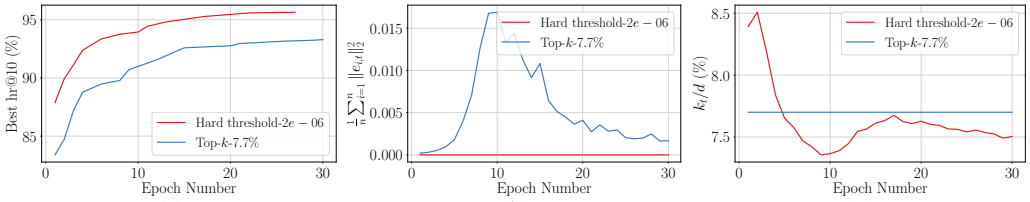
(a) (b) (c)

Figure 4: Convergence of Top-k and Hard-threshold for a logistic regression model on `gisette` LIBSVM dataset with 20 workers: (a) Functional suboptimality vs. bits communicated; (b) Error-compensated gradient norm vs. Epoch; (c) Error-norm vs. iterations. Top-k has large error-accumulation due to the cascading-effect.



(a) (b) (c)

Figure 5: **Convergence of Top-k and Hard-threshold for an LSTM on WikiText2 at 0.05% average density:** (a) Test-perplexity vs. Iterations, (b) Error-norm vs. Iterations, (c) Density (k_t/d) vs. Iterations. $k = 0.05\%$ of d, and λ = 0.0072. Hard-threshold has better convergence than Top-k because of a smaller total-error.



(a) (b) (c)

Figure 6: **Convergence of Top-k and Hard-threshold for NCF on ML-20m at 7.7% average density:** (a) Best Hit-rate@10 vs. Epochs, (b) Error-norm vs. Epochs, (c) Density (k_t/d) vs. Epochs. $k = 0.06\%$ of d, and λ = 0.0072. Hard-threshold has better convergence than Top-k because of a smaller total-error.

and split the dataset into number of worker partitions. For distributed EF-SGD, we use a local batch size of 1 at each node, where the new batch is chosen uniformly at random at each step.

Tuning the hard-threshold: Our goal is to make $f(x_T) - f(x^\star) \leq \epsilon$, for a given precision, $\epsilon > 0$. We set λ such that $d\gamma^2\lambda^2 = \epsilon$, i.e., $\lambda = \frac{\sqrt{\epsilon}}{d\sqrt{\gamma}}$.

Justification: Remark 5 states that by using a hard-threshold $\lambda > 0$, the noise due to compression is $d\gamma^2\lambda^2$. Due to this compression noise, we expect (although we did not prove) that x_T will oscillate in a $d\gamma^2\lambda^2$ neighborhood of the optimum, $x^\star$, i.e. $\|x_T - x^\star\|^2 \leq d\gamma^2\lambda^2$. Furthermore, by L-smoothness, we have

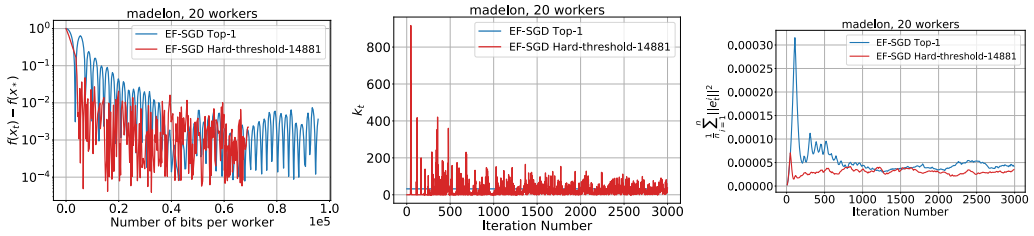
$$f(x_T) - f(x^\star) \leq \frac{L}{2}\|x_T - x^\star\|^2.$$

Therefore, if we want to converge to a ϵ-close functional-suboptimality value, $f(x_T) - f(x^\star)$, then ensuring $d\gamma^2\lambda^2 \leq \epsilon$ guarantees $\|x_T - x^\star\|^2 \leq \epsilon$, and implies, $f(x_T) - f(x^\star) \leq \frac{L}{2}\epsilon$. The above is an upper bound, and we observe in our experiments by using $\lambda = \frac{\sqrt{\epsilon}}{d\sqrt{\gamma}}$, gives $f(x_t) - f(x^\star) \leq \epsilon$.

C.3.1 Extreme sparsification

In Figure 7, we perform extreme sparsification to train a logistic regression model on the `madelon` LIBSVM dataset. We compare Top-k with $k = 1$, and hard-threshold with $\lambda = 14881$ set via $d\gamma^2\lambda^2 = 1.25 \times 10^{-4}$, so that they both communicate *same data volume*. In Figure 7b, we see

that Hard-threshold sparsifier does not communicate any elements in many iterations. Despite this, hard-threshold has faster convergence than Top-k in Figure 7 a. Figure 7 c demonstrates that this is because hard-threshold has a smaller total-error than Top-k.



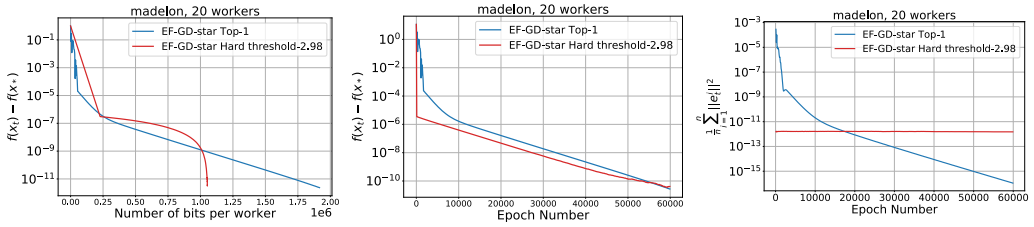
(a) (b) (c)

Figure 7: Convergence of Top-k and Hard-threshold for a logistic regression model on `madelon` LIBSVM dataset with 20 workers: (a) Functional suboptimality vs. bits communicated; (b) parameters communicated vs. iterations; (c) error norm vs. iterations. Hard-threshold has a faster convergence than Top-k even when it does not communicate any parameter in some iterations.

C.3.2 Convergence to an arbitrary neighborhood of the optimum

For the experiments in this section, the uncompressed baseline is distributed gradient descent (GD). Unlike SGD, GD has linear convergence to the exact optimum. However, Distributed EF-GD does not converge to the exact optimum due to compression noise. To remedy this, Gorbunov et al. [22] introduced a family of variance-reduced compression algorithms that have linear convergence to the exact optimum. We consider algorithm `EF-GDstar` from [22] (known as `EC-GDstar` in [22]).

We empirically show that `EF-GDstar` with hard-threshold compressor, can converge to an arbitrarily small neighborhood around the optimum, for an appropriate choice of hard-threshold. Figure 8 and Figure 9 demonstrate the convergence of `EF-GDstar` using Hard-threshold and Top-k sparsifiers with 20 workers and 100 workers, respectively. We choose (i) $k = 1$ for 20 workers and $k = 5$ for 100 workers, respectively; (ii) $\lambda = 2.98$, such that $d\gamma^2\lambda^2 = 5 \times 10^{-12}$. By using this λ, the compression error for hard-threshold is less than 5×10^{-12} in Figures 8 c and 9 c. Moreover, hard-threshold converges to $f(x_T) - f(x^\star) \leq 5 \times 10^{-12}$ in both Figures 8 b and 9 b. Additionally, hard-threshold sends $1.7\times$ and $8\times$ less data than Top-k in Figure 8 a and Figure 9 a, respectively. Furthermore, Figure 8 is an extreme sparsification scenario where hard-threshold communicates < 1 parameter per iteration per worker.



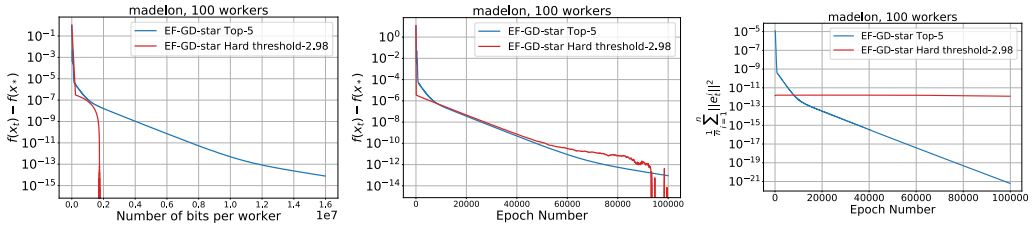
(a) (b) (c)

Figure 8: Convergence of `EF-GDstar` using Top-k and Hard-threshold sparsifiers on a logistic regression model on `madelon` LIBSVM dataset with 20 workers: (a) Functional suboptimality vs. bits communicated; (b) functional suboptimality vs. epochs; (c) error-norm vs. epochs.

Our results demonstrate that it is possible to use the hard-threshold compressor to converge to an arbitrarily small neighborhood around the optimum. We leave the convergence analyses, and devising practical variants for future research.

C.4 Comparison against ACCORDION

The experiment details are provided in 6, and the CIFAR-100 results are provided in Table 7.



(a) (b) (c)

Figure 9: Convergence of `EF-GDstar` using Top-k and Hard-threshold sparsifiers on a logistic regression model on `madelon` LIBSVM dataset with 100 workers: (a) Functional suboptimality vs. bits communicated; (b) functional suboptimality vs. epochs; (c) error norm vs. epochs.

Table 6: ACCORDION experiments

Dataset	CIFAR-10 and CIFAR-100
Architectures	ResNet-18 [26], SENet18 [30], GoogleNet [56]
Repository	PowerSGD [57] See `https://github.com/epfml/powersgd`
License	MIT
Number of workers	8
Global Batch-size	256×8
Optimizer	SGD with Nesterov Momentum
Momentum	0.9
Post warmup LR	0.1×16
LR-decay	$/10$ at epoch 150 and 250
LR-warmup	Linearly within 5 epochs, starting from 0.1
Number of Epochs	300
Weight decay	10^{-4}
Repetitions	6, with different seeds
Accordion: $k_{\min}$ value	0.1% for both CIFAR-10 and CIFAR-100
Accordion: $k_{\max}$ value	1% for CIFAR-10 and 2% for CIFAR-100
Hard-threshold: λ values (Calculated using $\lambda = \frac{1}{2\sqrt{k_{\min}}}$)	ResNet-18-CIFAR-10: 4.73×10^{-3} ResNet-18-CIFAR-100: 4.72×10^{-3} GoogleNet-CIFAR-10: 6.37×10^{-3} GoogleNet-CIFAR-100: 6.32×10^{-3} SENet18-CIFAR-10: 4.68×10^{-3} SENet18-CIFAR-100: 4.68×10^{-3}

Table 7: Comparison against ACCORDION [3] on CIFAR-100

Network	Method	Accuracy (%)	Average Density (%)
ResNet-18	Top-2% ($k_{\max}/d$)	71.8	2.00 (1×)
	Top-0.1% ($k_{\min}/d$)	70.6	0.10 (20×)
	ACCORDION	**71.6**	0.57 (3.5×)
	Hard-threshold ($\frac{1}{2\sqrt{k_{\min}}}$)	71.4	**0.35 (5.7×)**
GoogleNet	Top-2% ($k_{\max}/d$)	75.5	2.00 (1×)
	Top-0.1% ($k_{\min}/d$)	73.1	0.10 (20×)
	ACCORDION	74.2	0.48 (4.2×)
	Hard-threshold ($\frac{1}{2\sqrt{k_{\min}}}$)	**75.0**	**0.38 (5.3×)**
SENet18	Top-2% ($k_{\max}/d$)	71.9	2.00 (1×)
	Top-0.1% ($k_{\min}/d$)	70.1	0.10 (20×)
	ACCORDION	71.0	0.55 (3.6×)
	Hard-threshold ($\frac{1}{2\sqrt{k_{\min}}}$)	**72.1**	**0.36 (5.6×)**

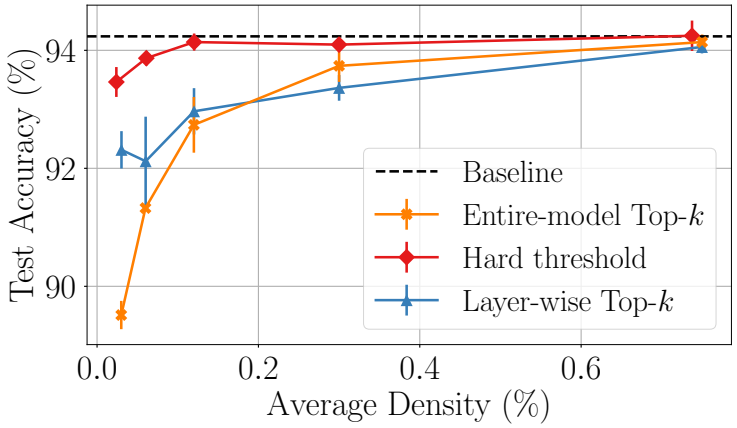


Figure 10: ResNet-18 on CIFAR-10

Figure 11: **Test metric vs. Data volume for entire-model compression**. The dashed black line in each plot denotes the no compression baseline. Each setting is repeated with three seeds, and we plot the average with standard deviation. For description on parameters, see Tables 3, 4, and 5.

C.5 Entire-model sparsification

Sparsification can be performed in two ways: layer-wise or entire-model. In layer-wise sparsification, the sparsifier is invoked individually on each tensor resulting from each layer. In contrast, in entire-model sparsification, the sparsifier is applied to a single concatenated tensor resulting from all layers. Since hard-threshold is an element-wise sparsifier, layer-wise and entire-model sparsification result in the same sparsified vector. However, it is expected that layer-wise and entire model vary substantially for Top-k. Layer-wise Top-k is used in all practical implementations [47, 37, 62] because performing entire-model Top-k is both compute and memory intensive.

While we employ layer-wise Top-k in our experiments, we present in Figure 11 the *test metric vs. data volume* experiment for ResNet-18-CIFAR-10 benchmark (Figure 2a) using entire-model Top-k. We find that hard-threshold is more communication-efficient than entire-model Top-k as well. Notably, at an average density ratio of 0.003%, hard-threshold has more than 4% higher accuracy than entire-model Top-k.

C.6 Error-Feedback (EF)

In this section, we discuss various aspects of EF (or memory). Particularly, in §C.6.1 we investigate if hard-threshold is more communication-efficient than Top-k without EF. Then, in Section C.6.2, we discuss and compare the different ways to perform EF in the literature.

C.6.1 Convergence without EF

To understand how the sparsifiers perform without the EF, we conduct experiments without EF for ResNet-18 benchmark. We report this in Figure 12. Similar to the with EF case, we find that hard-threshold has better convergence than Top-k. We note that with EF, hard-threshold achieved baseline performance at an extreme average density of 0.12%. However, without EF, hard-threshold fails to achieve baseline performance (94.2%) even at a significantly higher average density of 5%. Hence, EF is a necessary tool to ensure faster convergence.

C.6.2 Different types of EF

For optimizers other than vanilla SGD, one can compress and aggregate quantities other than stochastic gradients (such as momentum). Consider an example for SGD with Nesterov momentum, where the compression and aggregation can be performed in the following two ways:

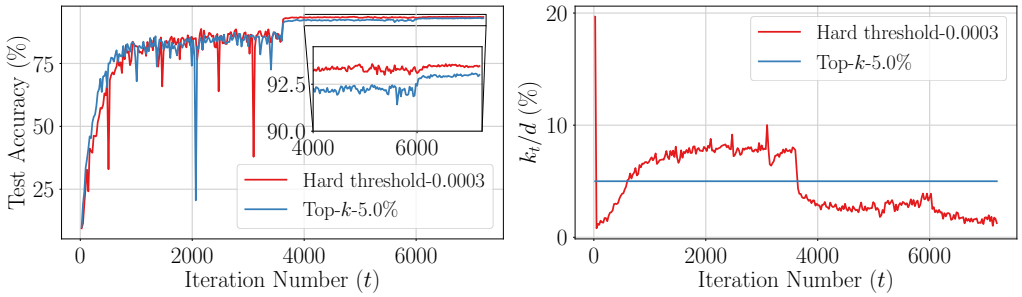


Figure 12: Top-k and hard-threshold without error compensation for ResNet-18 on CIFAR-10: (a) Accuracy vs. Iterations, (b) density, (k_t/d) vs. iterations. Average density is 5% for Top-k and 4.7% for hard-threshold.

Algorithm 2: Distributed EF SGD with momentum by using gradient compression

for *worker* $w = 1, .., W$ **in parallel do**
  for *iteration* $t = 1, 2, \cdots,$ **do**
    Compute local stochastic gradient g_w
    $\Delta_w \leftarrow g_w + e_w$
    $\mathcal{C}(\Delta_w) \leftarrow \texttt{COMPRESS}(\Delta_w)$
    $e_w \leftarrow \Delta_w - \texttt{DECOMPRESS}(\Delta_w)$
    $\mathcal{C}(\Delta) \leftarrow$
     $\texttt{AGGREGATE}(\mathcal{C}(\Delta_1), \ldots, \mathcal{C}(\Delta_W))$
    $\Delta^{'} \leftarrow \texttt{DECOMPRESS}(\mathcal{C}(\Delta))$
    $m \leftarrow \lambda m + \Delta^{'}$
    $x \leftarrow x - \gamma(\Delta^{'} + m)$

Algorithm 3: Distributed EF SGD with momentum by using update compression

for *worker* $w = 1, .., W$ **in parallel do**
  for *iteration* $t = 1, 2, \cdots,$ **do**
    Compute local stochastic gradient g_w
    $m_w \leftarrow \lambda m_w + g_w$
    $u_w \leftarrow m_w + g_w$
    $\Delta_w \leftarrow u_w + e_w$
    $\mathcal{C}(\Delta_w) \leftarrow \texttt{COMPRESS}(\Delta_w)$
    $e_w \leftarrow \Delta_w - \texttt{DECOMPRESS}(\Delta_w)$
    $\mathcal{C}(\Delta) \leftarrow$
     $\texttt{AGGREGATE}(\mathcal{C}(\Delta_1), \ldots, \mathcal{C}(\Delta_W))$
    $\Delta^{'} \leftarrow \texttt{DECOMPRESS}(\mathcal{C}(\Delta))$
    $x \leftarrow x - \gamma(\Delta^{'})$

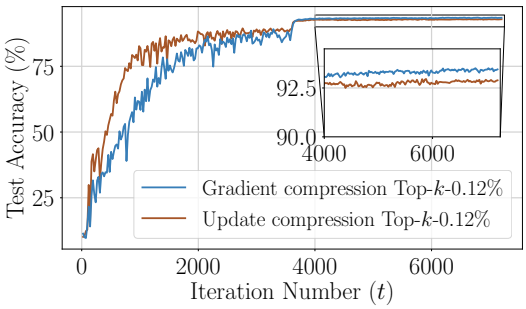


Figure 13: Test Accuracy for gradient compression vs. update compression for Top-k on ResNet-18 on CIFAR-10. We experiment with three different seeds, and the plot represents the run with highest final accuracy for each setting. The test accuracy statistics ($\mu \pm \sigma$) are: Gradient compression ($92.96 \pm 0.39\%$) and update compression ($90.78 \pm 2.03\%$).

- **Gradient compression.** This was proposed in [57] and is depicted in Algorithm 2. In the case of SGD with Nesterov momentum, this update rule ensures that every worker maintains the same momentum state. However, the updates to momentum is sparse, as the momentum is calculated using sparsified gradients.
- **Update compression.** This was proposed in [37], and is depicted in Algorithm 3. In the case of SGD with Nesterov momentum, every worker maintains a different momentum state calculated from their local stochastic gradients. Although updates to the momentum state is dense in this case, the momentum state is completely unaware of the compression and does not reflect the actual history of the updates. In order to circumvent this issue, Lin et. al. [37] had proposed momentum factor-masking to clear old local momentum states of a parameter once the parameter is updated. However, it is not easy to devise such modifications for optimizers which maintain multiple states derived from complicated calculations, such as RMSProp and ADAM.

Nomenclature for Algorithm 2 and 3. In Algorithm 2 and 3 we show the distributed training loop. We denote the learning rate by γ, momentum factor by λ, the model parameters by $x \in \mathbb{R}^d$, the momentum at worker w by m_w, and the error at worker w by e_w. At the beginning of the training, m_w and e_w are initialized to zero for all workers. By `COMPRESS`, `DECOMPRESS`, and `AGGREGATE` we denote the compression, decompression, and aggregate function, respectively.

We also conduct experiments for Top-k on ResNet-18 benchmark by using aforementioned update rules and find that gradient compression (Algorithm 2) results in better performance (see Figure 13). In light of the above discussion and experimental evidence, we stick to gradient compression (Algorithm 2) for our main experiments.

D How to tune the hard-threshold?

Substituting $\upsilon^2 = d\lambda^2$ for hard-threshold in Theorem 6 we get

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\|\nabla f(x_t)\|^2 \leq \frac{4(f(x_0)-f^\star)}{\gamma T} + \frac{2\gamma L(M\zeta^2+\sigma^2)}{n} + 2\gamma^2 L^2 d\lambda^2. \tag{32}$$

Similarly, substituting $\delta = \frac{k}{d}$ for Top-k in Theorem 7 we get

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\|\nabla f(x_t)\|^2 \leq \frac{8(f(x_0)-f^\star)}{\gamma T} + \frac{4\gamma L(M\zeta^2+\sigma^2)}{n} + \frac{8\gamma^2 L^2 d}{k}\left(\left(\frac{2d}{k}+M\right)\zeta^2+\sigma^2\right). \tag{33}$$

We ignore the first two terms unaffected by compression in (32) and (33), and focus on the last term. To ensure that hard-threshold has better convergence than Top-k we have

$$2L^2 d\lambda^2 \leq \frac{8L^2 d}{k}\left(\left(\frac{2d}{k}+M\right)\zeta^2+\sigma^2\right),$$

that is,

$$\lambda \leq \frac{2}{\sqrt{k}}\sqrt{\left(\frac{2d}{k}+M\right)\zeta^2+\sigma^2}.$$

Therefore, if $\hat{M}$, $\hat{\zeta}$, and $\hat{\sigma}$ are estimates of M, ζ, and σ, respectively, then we suggest setting the threshold as

$$\lambda \sim \frac{2}{\sqrt{k}}\sqrt{\left(\frac{2d}{k}+\hat{M}\right)\hat{\zeta}^2+\hat{\sigma}^2}.$$

In our comparison against ACCORDION, we assume $\hat{\zeta} = 0$ (homogeneous distributed data), and $\hat{\sigma} \sim \frac{1}{4}$. This leads us to the hard-threshold value

$$\lambda \sim \frac{1}{2\sqrt{k}}.$$

We find that $\lambda = \frac{1}{2\sqrt{k_{\min}}}$ has better performance (with similar total-data volume) than Top-$k_{\min}$ in Tables 1 and 7.