
On Distributed Adaptive Optimization with Gradient Compression

Anonymous Author(s)

Affiliation

Address

email

Abstract

1 This paper presents a new algorithm – COMP-AMS – for tackling single-machine
2 and distributed optimization. Unlike prior works which rely on full gradient com-
3 munication between the workers and the parameter-server, we design a distributed
4 adaptive optimization method with gradient compression coupled with an error-
5 feedback technique to alleviate the bias introduced by the compression. While the
6 former permits to transmit fewer bits of gradient vectors to the server, we show
7 that using the latter, which correct for the bias, our methods reach a stationary
8 point in $\mathcal{O}(1/\sqrt{T})$ iterations, matching that of state-of-the-art single-machine and
9 distributed methods, without any error-feedback. We illustrate our theoretical re-
10 sults by showing the effectiveness of our method both under the single-machine
11 and distributed settings on various benchmark datasets.

12 1 Introduction

13 Deep neural network has achieved the state-of-the-art learning performance on numerous AI appli-
14 cations, e.g., computer vision [25, 28, 51], Natural Language Processing [27, 58, 63], Reinforcement
15 Learning [40, 48] and recommendation systems [17, 53]. With the increasing size of both data and
16 deep networks, standard single-machine training confronts with at least two major challenges:

- 17 • Due to the limited computing power of a single-machine, it would take a long time to
18 process the massive number of data samples—training would be slow.
- 19 • In many practical scenarios, data are typically stored in multiple servers, possibly at differ-
20 ent locations, due to the storage constraints (massive user behavior data, Internet images,
21 etc.) or privacy reasons [11]. Transmitting data might be costly.

22 *Distributed learning* framework [19] has been a common training strategy to tackle the above two
23 issues. For example, in centralized distributed stochastic gradient descent (SGD) protocol, data are
24 located at n local nodes, at which the gradients of the model are computed in parallel. In each
25 iteration, a central server aggregates the local gradients, updates the global model, and transmits
26 back the updated model to the local nodes for subsequent gradient computation. As we can see, this
27 setting naturally solves aforementioned issues: 1) We use n computing nodes to train the model, so
28 the time per training epoch can be largely reduced; 2) There is no need to transmit the local data to
29 central server. Besides, distributed training also provides stronger error tolerance since the training
30 process could continue even one local machine breaks down. As a result of these advantages, there
31 has been a surge of study and applications on distributed systems [10, 42, 22, 26, 29, 37, 35].

32 Among many optimization strategies, SGD is still the most popular prototype in distributed training
33 for its simplicity and effectiveness [15, 1, 39]. Yet, when the deep learning model is very large,
34 the communication between local nodes and central server could be expensive. Burdensome gra-
35 dient transmission would slow down the whole training system, or even be impossible because of

the limited bandwidth in some applications. Thus, reducing the communication cost in distributed SGD has become an active topic, and an important ingredient of large-scale distributed systems (e.g. [45]). Solutions based on quantization, sparsification and other compression techniques of the local gradients are proposed, e.g., [4, 54, 52, 49, 3, 7, 18, 56, 30]. As one would expect, in most approaches, there exists a trade-off between compression and learning performance. In general, larger bias and variance of the compressed gradients usually bring more significant performance downgrade in terms of convergence [49, 2]. Interestingly, studies (e.g., [33]) show that the technique of *error feedback* can to a large extent remedy the issue of such biased compressors, achieving same convergence rate as full-gradient SGD.

On the other hand, in recent years, adaptive optimization algorithms (e.g. AdaGrad [23], Adam [34] and AMSGrad [44]) have become popular because of their superior empirical performance. These methods use different implicit learning rates for different coordinates that keep changing adaptively throughout the training process, based on the learning trajectory. In many learning problems, adaptive methods have been shown to converge faster than SGD, sometimes with better generalization as well. Despite of the great popularity of adaptive methods, the body of literature that extends them to distributed training is still very limited. In particular, even the simple gradient averaging approach, though appearing standard, has not been considered for adaptive optimization algorithms. Meanwhile, adopting gradient compression in adaptive methods has also been rarely studied in literature. We try to fill this gap in this paper, by studying COMP-AMS, a distributed adaptive optimization framework using the gradient averaging protocol. Gradient compression is incorporated to reduce the communication cost. We provide theoretical analysis and show that our method can achieve satisfactory performance with significantly reduced communication overhead.

1.1 Our Contributions

Specifically, we develop a simple optimization framework leveraging the adaptivity of AMSGrad, and focus on the computational virtue of local gradient compression technique.

Our technique is shown to be both theoretically and empirically effective under *the classical centralized setting* and *the distributed setting*.

In this contribution,

- We derive COMP-AMS, a distributed optimization method with gradient compression occurring at the worker level. Our scheme is coupled with a error-feedback technique to reduce the bias implied by the compression step.
- Throughout this paper, we provide single-machine and decentralized views of our method both on the empirical and theoretical levels. We exhibits the advantage of the compression and error-feedback steps within an adaptive optimization trajectory under those two settings.
- Under mild assumptions, such as nonconvexity and smoothness, we provide a non-asymptotic convergence rate of COMP-AMS in the general case, *i.e.*, when the number of workers is equal to n and with unspecified values for the hyperparameters. Our theoretical analysis includes the special cases of single-machine setting ($n = 1$) and exhibits a linear speedup (linear in n) of our method in the particular case of $\beta_1 = 0$.
- We highlight the effectiveness of our compressed adaptive method through several numerical experiments for single-machine and distributed optimization tasks.

We review Section 2 the contributions to date, related to compression techniques in optimization, such as quantization and sparsification, and to error feedback technique. Then, we develop in Section 3, our communication-efficient method, namely COMP-AMS, using AMSGrad as a prototype optimization algorithm. Theoretical understanding of our method's behaviour with respect to convergence towards a stationary point is developed in Section 4. Numerical results are illustrated in Section 5 to show the effectiveness of the proposed approach.

84 2 Related Work

85 2.1 Distributed SGD with Compressed Gradients

86 **Quantization.** As we mentioned before, SGD is the most commonly adopted optimization method
 87 in distributed training of deep neural nets. To reduce the expensive communication in large-scale
 88 distributed systems, extensive works have considered various compression techniques applied to the
 89 gradient transaction procedure. The first strategy is quantization. [20] condenses 32-bit floating
 90 numbers into 8-bits when representing the gradients. [45, 7, 33, 8] use the extreme 1-bit information
 91 (sign) of the gradients, combined with tricks like momentum, majority vote and memory. Other
 92 quantization-based methods include QSGD [4, 55, 62] and LPC-SVRG [60], leveraging unbiased
 93 stochastic quantization. The saving in communication of quantization methods is moderate: for
 94 example, 8-bit quantization reduces the cost to 25% (compared with 32-bit full-precision). Even in
 95 the extreme 1-bit case, the largest compression ratio is around $1/32 \approx 3.1\%$.

96 **Sparsification.** Gradient sparsification is another popular solution which may provide higher com-
 97 pression rate. Instead of commuting the full gradient, each local worker only passes a few coordi-
 98 nates to the central server and zeros out the others. Thus, we can more freely choose higher com-
 99 pression ratio (e.g., 1%, 0.1%), still achieving impressive performance in many applications [36].
 100 Stochastic sparsification methods, including uniform sampling and magnitude based sampling [52],
 101 select coordinates based on some sampling probability yielding unbiased gradient compressors.
 102 Deterministic methods are simpler, e.g., Random- k , Top- k [49, 47] (selecting k elements with
 103 largest magnitude), Deep Gradient Compression [36], but usually lead to biased gradient estima-
 104 tion. In [30], the central server identifies heavy-hitters from the count-sketch [12] of the local gradi-
 105 ents, which can be regarded as a noisy variant of Top- k strategy. More applications and analysis of
 106 compressed distributed SGD can be found in [32, 46, 5, 6, 31], among others.

107 **Error Feedback (EF).** Biased gradient estimation, which is a consequence of many foremen-
 108 tioned methods (e.g., signSGD, Top- k), undermines the model training, both theoretically and em-
 109 pirically, with slower convergence and worse generalization [2, 9]. The technique of *error feedback*
 110 is able to “correct for the bias” and fix the convergence issues. In this procedure, the difference
 111 between the true stochastic gradient and the compressed one is accumulated locally, which is then
 112 added back to the local gradients in later iterations. [49, 33] prove the $\mathcal{O}(\frac{1}{T})$ and $\mathcal{O}(\frac{1}{\sqrt{T}})$ conver-
 113 gence rate of EF-SGD in strongly convex and non-convex setting respectively, matching the rates
 114 of vanilla SGD [43, 24]. More works on the convergence rate of SGD with error feedback in-
 115 clude [66, 50], among other related papers.

116 2.2 Adaptive Optimization

117 In each SGD update, all the gradient coordinates share the same learning rate. This
 118 latter is either constant or decreasing through the iterations. Adaptive optimization
 119 methods cast different learning rate on each dimension. For instance, AdaGrad, de-
 120 veloped in [23], divides the gradient element-wisely by $\sqrt{\sum_{t=1}^T g_t^2} \in \mathbb{R}^d$, where
 121 $g_t \in \mathbb{R}^d$ is the gradient vector at time t and d is the model dimensionality.

122 Thus, it intrinsically assigns different learning
 123 rates to different coordinates throughout the
 124 training – elements with smaller previous gra-
 125 dient magnitude tend to move a larger step via
 126 larger learning rate. AdaGrad has been shown
 127 to perform well especially under some sparsity
 128 structure in the model and data. Other adaptive
 129 methods include AdaDelta [61] and Adam [34],
 130 which introduce momentum and moving aver-
 131 age of second moment estimation into AdaGrad
 132 hence leading to better performances. AMS-
 133 Grad [44] fixes the potential convergence issue of Adam, which will serve as the prototype in this
 134 paper. We present the pseudocode in Algorithm 1.

Algorithm 1 AMSGRAD optimization method

```

1: Input: parameters  $\beta_1, \beta_2$ , and  $\eta_t$ .
2: Initialize:  $\theta_1 \in \Theta$  and  $v_0 = \epsilon \mathbf{1} \in \mathbb{R}^d$ .
3: for  $t = 1$  to  $T$  do
4:   Compute stochastic gradient  $g_t$  at  $\theta_t$ .
5:    $m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t$ .
6:    $v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2$ .
7:    $\hat{v}_t = \max(\hat{v}_{t-1}, v_t)$ .
8:    $\theta_{t+1} = \theta_t - \eta_t \frac{\theta_t}{\sqrt{\hat{v}_t}}$ .
9: end for

```

In general, adaptive optimization methods are easier to tune in practice, and usually exhibit faster convergence than SGD. Thus, they have been widely used in training deep learning models in language and computer vision applications, e.g., [16, 57, 64]. In distributed setting, the work [41] proposes a decentralized system in online optimization. However, communication efficiency is not considered. The recent work [13] is the most relevant to our paper. Yet, their method is based on Adam, and requires every local node to store a local estimation of the moments of the gradient. Thus, one has to keep extra two more tensors of the model size on each local worker, which may be less feasible in terms of memory especially with large models. We will present more detailed comparison in Section 3.

3 Communication-Efficient Adaptive Optimization

Consider the distributed optimization task where n workers jointly solve a large finite-sum optimization problem written as:

$$\min_{\theta \in \Theta} \frac{1}{n} \sum_{i=1}^n f_i(\theta) := \frac{1}{n} \sum_{i=1}^n \mathbb{E}_{x \sim \mathcal{X}_i} [F_i(\theta; x)], \quad (1)$$

where the non-convex function f_i represents the average loss (over the local data samples) for worker $i \in [n]$ and θ the global model parameter taking value in Θ , a subset of \mathbb{R}^d . \mathcal{X}_i is the data distribution on each local node.

3.1 Gradient Compressors

In this paper, we mainly consider deterministic q -deviate compressors defined as below.

Assumption 1. The gradient compressor $\mathcal{C} : \mathbb{R}^d \mapsto \mathbb{R}^d$ is q -deviate: for $\forall x \in \mathbb{R}^d$, $\exists 0 \leq q < 1$ such that $\|\mathcal{C}(x) - x\| \leq q \|x\|$.

Note that, larger q indicates important an compression while smaller q implies better approximation of the true gradient. Hence, $q = 0$ implies no compression, i.e., $\mathcal{C}(x) = x$. We give two popular and highly efficient q -deviate compressors that will be compared in this paper.

Definition 1 (Top- k). For $x \in \mathbb{R}^d$, denote \mathcal{S} as the size- k set of $i \in [d]$ with largest k magnitude $|x_i|$. The **Top- k** compressor is defined as $\mathcal{C}(x)_i = x_i$, if $i \in \mathcal{S}$; $\mathcal{C}(x)_i = 0$ otherwise.

Definition 2 (Block-Sign). For $x \in \mathbb{R}^d$, define M blocks indexed by \mathcal{B}_i , $i = 1, \dots, M$, with $d_i := |\mathcal{B}_i|$. The **Block-Sign** compressor is defined as $\mathcal{C}(x) = [\text{sign}(x_{\mathcal{B}_1}) \frac{\|x_{\mathcal{B}_1}\|_1}{d_1}, \dots, \text{sign}(x_{\mathcal{B}_M}) \frac{\|x_{\mathcal{B}_M}\|_1}{d_M}]$.

Remark 1. It is well-known [49, 66] that for **Top- k** , $q^2 = 1 - \frac{k}{d}$; for **Block-Sign**, by Cauchy-Schwartz inequality we have $q^2 = 1 - \min_{i \in [M]} \frac{1}{d_i}$ where M and d_i are defined in Definition 2.

The intuition of **Top- k** is that, it has been observed in many deep neural networks that during training, most gradients are typically very small and can be regarded as redundant—gradients with large magnitude contain most information. The **Block-Sign** compressor is a simple extension of the 1-bit **SIGN** compressor [45, 7], adapted to different gradient magnitude in different blocks, which, for neural nets, are usually set as the distinct network layers. The scaling factor in Definition 2 is to preserve the (possibly very different) gradient magnitude in each layer. In principle, **Top- k** would perform the best when the gradient is sparse, or only has a few very large absolute values, while **Block-Sign** compressor is beneficial when most gradients have similar magnitude within each layer.

3.2 COMP-AMS for Distributed Optimization

We present in Algorithm 2 our proposed communication-efficient distributed adaptive method in this paper, COMP-AMS. This framework can be regarded as an analogue to the standard synchronous distributed SGD protocol: in each iteration, each local worker transmits to the central server the compressed stochastic gradient computed using local data. Then the central server takes the average of local gradients, and performs an AMSGrad update. Despite that this method seems a straightforward extension of distributed SGD with gradient compression, no formal analysis of COMP-AMS has been conducted in prior literature.

179 In Algorithm 2, line 7-8 depict the error feedback operation at local nodes. $e_{t,i}$ is the accumulated
 180 error from gradient compression on the i -th worker up to time $t - 1$. This residual is added back
 181 to $g_{t,i}$ to get the “correct” gradient. In Section 4 and Section 5, we will show that error feedback,
 182 similar to the case of SGD, also brings good convergence behavior under gradient compression in
 183 adaptive optimization methods.

184 **Comparison with Quantized Adam [13].** The first difference of COMP-AMS compared with [13]
 185 which develops a quantized variant of Adam [34] is that they use Adam as the underlying algorithm.
 186 It does not use the variable \hat{v} (line 15 of Algorithm 2) that ensures non-decreasing second moment
 187 estimation, which has been shown to be an important factor for the convergence guarantee [44, 14].
 188 Indeed, the convergence rate given by [13] does not match the rate of vanilla AMSGrad (in fact
 189 it does not converge to 0) when decreasing learning rate (e.g., $\mathcal{O}(\frac{1}{\sqrt{T}})$) is taken, which could be
 190 problematic both intuitively and given the fact that decreasing learning rate is standard in theoretical
 191 analysis and practical implementation. In this paper, we will prove the good convergence behavior
 192 of COMP-AMS that matches the rate of full-gradient AMSGrad (Section 4).

193 Another key difference is that, in COMP-AMS, only compressed gradients are transmitted from the
 194 workers to the central server. In [13], each worker keeps a local copy of the moment estimates
 195 commonly noted m and v , and compresses and transmits the ratio $\frac{m}{v}$ as a whole to the server. Thus,
 196 that method is very much like the compressed distributed SGD, with the exception that the ratio $\frac{m}{v}$
 197 plays the role of the gradient vector g communication-wise. Thus, two local moment estimators are
 198 additionally required, which have same size as the deep learning model. In our optimization method
 199 in Algorithm 2, the moment estimates m and v are kept and updated only at the central server, thus
 200 not introducing any extra variables (tensors) on local nodes during training (except for the error
 201 accumulator). In other words, COMP-AMS is not only effective in communication reduction, but
 202 also efficient in terms of memory (space), which is favorable for the distributed adaptive training
 203 of large-scale learners like BERT and CTR prediction models, e.g. [21, 65], to lower the hardware
 204 consumption happening in practice.

Algorithm 2 Distributed COMP-AMS with error-feedback

```

1: Input: parameters  $\beta_1, \beta_2$ , learning rate  $\eta_t$ .
2: Initialize: central server parameter  $\theta_1 \in \Theta \subseteq \mathbb{R}^d$ ;  $e_{1,i} = 0$  the error accumulator for each
   worker; sparsity parameter  $k$ ;  $n$  local workers;  $m_0 = 0, v_0 = 0, \hat{v}_0 = 0$ 
3: for  $t = 1$  to  $T$  do
4:   parallel for worker  $i \in [n]$  do:
5:     Receive model parameter  $\theta_t$  from central server
6:     Compute stochastic gradient  $g_{t,i}$  at  $\theta_t$ 
7:     Compute  $\tilde{g}_{t,i} = \mathcal{C}(g_{t,i} + e_{t,i}, k)$ 
8:     Update the error  $e_{t+1,i} = e_{t,i} + g_{t,i} - \tilde{g}_{t,i}$ 
9:     Send  $\tilde{g}_{t,i}$  back to central server
10:  end parallel
11:  Central server do:
12:     $\bar{g}_t = \frac{1}{n} \sum_{i=1}^n \tilde{g}_{t,i}$ 
13:     $m_t = \beta_1 m_{t-1} + (1 - \beta_1) \bar{g}_t$ 
14:     $v_t = \beta_2 v_{t-1} + (1 - \beta_2) \bar{g}_t^2$ 
15:     $\hat{v}_t = \max(v_t, \hat{v}_{t-1})$ 
16:    Update the global model  $\theta_{t+1} = \theta_t - \eta_t \frac{m_t}{\sqrt{\hat{v}_t + \epsilon}}$ 
17: end for

```

205 4 Convergence Analysis

206 In this section, we provide a finite time convergence result of our method, true for any termination
 207 iteration index T . We make the following additional assumptions.

208 **Assumption 2. (Smoothness)** For $\forall i \in [n]$, f_i is L -smooth: $\|\nabla f_i(\theta) - \nabla f_i(\vartheta)\| \leq L \|\theta - \vartheta\|$.

209 **Assumption 3. (Unbiased and bounded stochastic gradient)** For $\forall t > 0, \forall i \in [n]$, the stochastic
 210 gradient is unbiased and uniformly bounded: $\mathbb{E}[g_{t,i}] = \nabla f_i(\theta_t)$ and $\|g_{t,i}\| \leq G$.

211 **Assumption 4.** (Bounded variance) For $\forall t > 0, \forall i \in [n]$: (i) the **local variance** of the stochastic
 212 gradient is bounded: $\mathbb{E}[\|g_{t,i} - \nabla f_i(\theta_t)\|^2] < \sigma_g^2$; (ii) the **global variance** is bounded by
 213 $\frac{1}{n} \sum_{i=1}^n \|\nabla f_i(\theta_t) - \nabla f(\theta_t)\|^2 \leq \sigma_g^2$.

214 In Assumption 3, the uniform bound on the stochastic gradient is common in convergence analysis
 215 of adaptive methods, e.g., [44, 67, 14]. The global variance bound σ_g^2 characterizes the difference
 216 among local objective functions, which, is mainly caused by different local data distribution \mathcal{X}_i in
 217 (1). In classical distributed setting where all the workers can access the same dataset and local data
 218 are assigned randomly, $\sigma_g^2 \equiv 0$. The scenario where \mathcal{X}_i 's are different gives rise to the recently
 219 proposed Federated Learning (FL) [38] framework where local data can be non-i.i.d. While typical
 220 FL method with periodical model averaging is beyond the scope of this present paper, we consider
 221 the global variance in our analysis to shed some light on the impact of non-i.i.d. data distribution in
 222 the federated setting for broader interest and future investigation.

223 Under the mild assumptions stated above, we derive the following general convergence rate of
 224 COMP-AMS in the distributed setting.

225 **Theorem 1.** Denote $C_0 = \sqrt{\frac{4(1+q^2)^3}{(1-q^2)^2} G^2 + \epsilon}$, $C_1 = \frac{\beta_1}{1-\beta_1} + \frac{2q}{1-q^2}$. Under Assumption 1 to Assump-
 226 tion 4, with $\eta_t = \eta \leq \frac{\epsilon}{3C_0 \sqrt{2L \max\{2L, C_2\}}}$, for any $T > 0$, COMP-AMS satisfies

$$\begin{aligned} \frac{1}{T} \sum_{t=1}^T \mathbb{E}[\|\nabla f(\theta_t)\|^2] &\leq 2C_0 \left(\frac{\mathbb{E}[f(\theta_1) - f(\theta^*)]}{T\eta} + \frac{\eta L \sigma^2}{n\epsilon} + \frac{3\eta^2 L C_0 C_1 \sigma^2}{\epsilon^2} \right. \\ &\quad \left. + \frac{12\eta^2 q^2 L C_0 \sigma_g^2}{(1-q^2)^2 \epsilon^2} + \frac{(1+C_1)G^2 d}{T\sqrt{\epsilon}} + \frac{\eta(1+2C_1)C_1 L G^2 d}{T\epsilon} \right). \end{aligned}$$

227 The LHS of Theorem 1 is the expected squared norm of the gradient from a uniformly chosen iterate
 228 $t \in [T]$, which is a common convergence measure. From Theorem 1, we see that the more com-
 229 pression we apply to the gradient vectors (i.e., larger q), the larger the upper bound of the stationary
 230 condition is, i.e., the slower the algorithm converges. This is intuitive as heavier compression loses
 231 more gradient information which would slower down the learner to find a good solution.

232 **Single machine rate** ($n = 1$). Note that, COMP-AMS with $n = 1$ naturally reduces to the single
 233 machine (sequential) AMSGrad (Algorithm 1) with compressed gradients instead of full-precision
 234 ones. The paper [33] shows that for SGD, error feedback fixes the convergence issue of biased
 235 compressors in the sense that SGD-EF (with compressed gradients) has the same convergence rate
 236 as vanilla SGD using full gradients. For AMSGrad, we have a similar result.

237 **Corollary 1.** Assume $n = 1$. Under Assumption 1 to Assumption 4, setting the stepsize as $\eta =$
 238 $\min\{\frac{\epsilon}{3C_0 \sqrt{2L \max\{2L, C_2\}}}, \frac{1}{\sqrt{T}}\}$, the sequence of iterates $\{\theta_t\}_{t>0}$ output from Algorithm 2 satisfies:

$$\frac{1}{T} \sum_{t=1}^T \mathbb{E}[\|\nabla f(\theta_t)\|^2] \leq \mathcal{O}\left(\frac{1}{\sqrt{T}} + \frac{\sigma^2}{\sqrt{T}} + \frac{d}{T}\right).$$

239 Corollary 1 states that with error feedback, single machine AMSGrad with biased compressed gra-
 240 dients can also match the convergence rate $\mathcal{O}(\frac{1}{\sqrt{T}} + \frac{d}{T})$ of standard AMSGrad [67] in non-convex
 241 optimization. It also achieves the same rate $\mathcal{O}(\frac{1}{\sqrt{T}})$ of vanilla SGD [33] when T is sufficiently large.
 242 In other words, EF also fixes the convergence issue of using compressed gradients in AMSGrad. To
 243 the best of our knowledge, this is the first result in literature showing that compressed adaptive meth-
 244 ods with EF converges as fast as the standard counterpart. We will validate this benefit of EF in our
 245 numerical experiments.

246 **Linear Speedup** ($n > 1$). In Theorem 1, the convergence rate is derived assuming constant learning
 247 rate. By carefully choosing a decreasing learning rate dependent on the number of workers, we have
 248 the following result.

249 **Corollary 2.** *Under the same setting as Theorem 1, set $\eta = \min\{\frac{\epsilon}{3C_0\sqrt{2L\max\{2L,C_2\}}}, \frac{\sqrt{n}}{\sqrt{T}}\}$. Ignor-*
 250 *ing irrelevant quantities, we have*

$$\frac{1}{T} \sum_{t=1}^T \mathbb{E}[\|\nabla f(\theta_t)\|^2] \leq \mathcal{O}\left(\frac{1}{\sqrt{nT}} + \frac{\sigma^2}{\sqrt{nT}} + \frac{n(\sigma^2 + \sigma_g^2)}{T}\right). \quad (2)$$

251 In Corollary 2, we see that the global variance σ_g^2 appears in the $\mathcal{O}(\frac{1}{T})$ term, which says that it
 252 asymptotically has no impact on the convergence. This matches the result of momentum SGD [59].
 253 When $T \geq \mathcal{O}(n^3)$ is sufficiently large, the third term in (2) vanishes, and the convergence rate
 254 becomes $\mathcal{O}(\frac{1}{\sqrt{nT}})$. Therefore, to reach a $\mathcal{O}(\delta)$ stationary point, one worker ($n = 1$) needs $T =$
 255 $\mathcal{O}(\frac{1}{\delta^2})$ iterations, while distributed training with n workers requires only $T = \mathcal{O}(\frac{1}{N\delta^2})$ iterations,
 256 which is n times faster than single machine training. That is, COMP-AMS has a linear speedup
 257 in terms of the number of the local workers. Such acceleration effect has also been reported for
 258 compressed SGD with error feedback [66, 32] and momentum SGD [59]. To our knowledge, this is
 259 also the first result showing the linear speedup for distributed adaptive methods under compression.

260 5 Experiments

261 In this section, we provide numerical results on several real-world datasets. Our main objective is to
 262 validate the theoretical results, and demonstrate that the proposed COMP-AMS can give satisfactory
 263 learning performance with significantly reduced communication costs.

264 5.1 Error Feedback Fixes the Convergence of Compressed AMSGrad

265 In Corollary 1, we established that for standard single machine AMSGrad under compression, EF
 266 helps achieve same convergence rate as the full-gradient AMSGrad. We implement COMP-AMS
 267 with a single worker and different gradient compressors, with or without EF, to justify this claim.
 268 We train MNIST classification using a simple Convolutional Neural Network (CNN). The model has
 269 two convolutional layers followed by two fully connected layers with ReLU activation. Dropout
 270 is applied after the max-pooled convolutional layer with rate 0.5. The training batch size is 200.
 271 The parameters in COMP-AMS are set as default $\beta_1 = 0.9$ and $\beta_2 = 0.999$, which is true for all
 272 the experiments in this paper. The learning rate is searched over a fine grid and the best result is
 273 reported. In Figure 1, the methods are (all performed on single worker):

- 274 • TopK-EF-0.01: COMP-AMS (Algorithm 2) and **Top- k** compressor, with sparsity 0.01 (i.e.,
 275 keeping 1% gradient coordinates with largest magnitude).
- 276 • TopK-EF-0.001: COMP-AMS with **Top- k** compressor, with sparsity 0.001.
- 277 • BkSign-EF: COMP-AMS with **Block-Sign** compression.
- 278 • Methods without “-EF”: AMSGrad using corresponding compressors without error feed-
 279 back (directly trained with compressed gradients).
- 280 • MVSS-0.01: AMSGrad with Minimal Variance Stochastic Sparsification (MVSS) [52] at
 281 0.01 sparsity. This method probabilistically chooses gradient coordinates according to their
 282 magnitude, and divides each selected coordinate by its sampling probability to generate
 283 unbiased output (of the full gradient). Therefore, error feedback is not used for MVSS¹.

284 From the train loss and test accuracy in Figure 1, we observe:

- 285 • AMSGrad without EF (dash curves) performs very poorly in terms of both convergence
 286 speed (more fluctuations in later epochs) and generalization (much worse test accuracy).
 287 With error feedback, the training loss and test accuracy both approach those of full-gradient
 288 AMSGrad with faster convergence. The issue of biased compression is fixed.
- 289 • **Top- k** -EF-0.01 performs better than **Top- k** -EF-0.001, which justifies the influence of q in
 290 Theorem 1 that higher compression ratio would undermine the learning performance.

¹We also tested MVSS with error feedback, but the performance is uncompetitive with other methods.

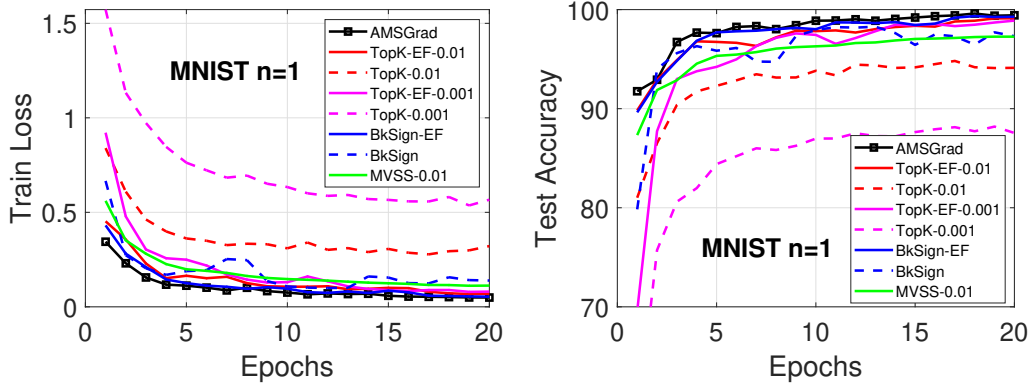


Figure 1: Training loss and test accuracy of COMP-AMS using a single machine.

- MVSS-0.01 is beaten by the proposed EF-corrected **Block-Sign** and **Top- k** even with 0.001 sparsity. This suggests that using biased compressors with EF in COMP-AMS is more effective than using unbiased stochastic compressors.

5.2 Linear Speedup of COMP-AMS

Corollary 2 reveals the linear speedup of COMP-AMS in distributed training. We validate this claim in Figure 2, where the training loss on MNIST against the number of iterations is provided. Here we use COMP-AMS with **Top- k** and 0.01 sparsity. We test $n = 1, 2, 4, 8$, where the local mini-batch size is 100. As suggested by the theory, we use $10^{-4}\sqrt{n}$ as the learning rate. From Figure 2, we observe that the number of iterations for multiple workers to achieve a certain loss decreases as n increases. For example, to achieve 0.5 loss, we need around 700, 400, 240, 120 iterations for $n = 1, 2, 4, 8$ respectively, which decreases approximately linearly. This numerically justifies the linear speedup effect ($\mathcal{O}(\frac{1}{\sqrt{nT}})$ convergence rate) of COMP-AMS.

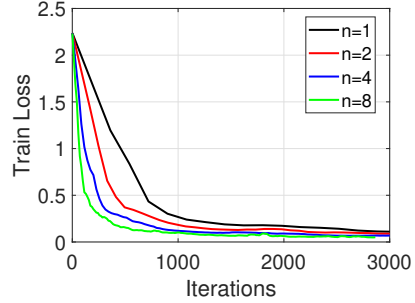


Figure 2: Training loss of COMP-AMS with **Top- k** -0.01 on MNIST.

5.3 General Evaluation and Communication Efficiency

In this section, we present general evaluation on more datasets and compare the communication efficiency. For CIFAR-10 image classification, we train a larger CNN model with 3 convolutional layers (more detailed architecture can be found in the supplement). For IMDB movie review sentiment analysis task, we train a LSTM network with 64 cells, equipped with an embedding layer which embeds top 1000 words in the movie reviews into 32-dimensional vectors.

6 Conclusion

In this paper, we develop a strategy for deriving communication-efficient and fast optimizations algorithms for distributed and single-machine learning tasks. Specifically, we integrate a compression step of the gradient vectors at the worker level only, via a **Top- k** operation, coupled with an error-feedback technique within a AMSGrad-type of algorithm to alleviate the communication bottleneck of distributed learning among a large number of workers. We derive bounds for the performance, in terms of stationarity, of the proposed algorithm and show that our algorithm convergence rate matches state-of-the-art rates for distributed learning while compressing most of the transmitted information. We also show that a linear speedup in the number of workers is possible in some special cases. We verify our theoretical results via numerical experiments involving benchmark datasets for

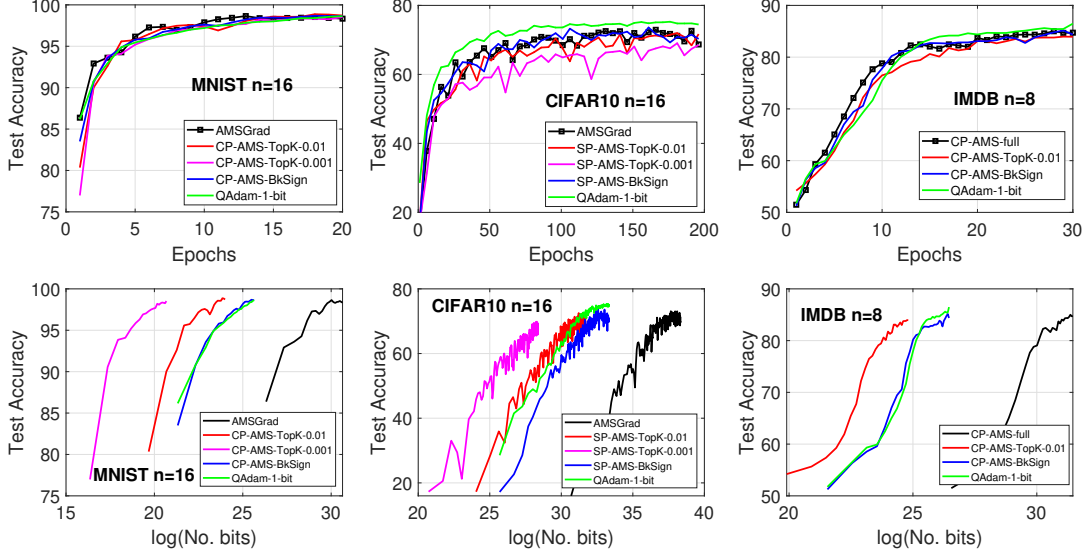


Figure 3: Test accuracy of distributed COMP-AMS. Top row: accuracy vs. epochs. Bottom row: accuracy vs. number of bits transmitted per worker.

supervision learning tasks. We show through those runs that our method exhibits similar empirical convergence speed using drastically less computational resources.

References

- [1] Naman Agarwal, Ananda Theertha Suresh, Felix X. Yu, Sanjiv Kumar, and Brendan McMahan. cpsgd: Communication-efficient and differentially-private distributed SGD. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 7575–7586, 2018.
- [2] Ahmad Ajalloeian and Sebastian U Stich. Analysis of sgd with biased gradient estimators. *arXiv preprint arXiv:2008.00051*, 2020.
- [3] Alham Fikri Aji and Kenneth Heafield. Sparse communication for distributed gradient descent. *arXiv preprint arXiv:1704.05021*, 2017.
- [4] Dan Alistarh, Demjan Grubic, Jerry Li, Ryota Tomioka, and Milan Vojnovic. Qsgd: Communication-efficient sgd via gradient quantization and encoding. In *Advances in Neural Information Processing Systems*, pages 1709–1720, 2017.
- [5] Dan Alistarh, Torsten Hoefler, Mikael Johansson, Sarit Khirirat, Nikola Konstantinov, and Cédric Renggli. The convergence of sparsified gradient methods. *arXiv preprint arXiv:1809.10505*, 2018.
- [6] Debraj Basu, Deepesh Data, Can Karakus, and Suhas N. Diggavi. Qsparse-local-sgd: Distributed SGD with quantization, sparsification and local computations. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 14668–14679, 2019.
- [7] Jeremy Bernstein, Yu-Xiang Wang, Kamyar Azizzadenesheli, and Animashree Anandkumar. signsgd: Compressed optimisation for non-convex problems. In *International Conference on Machine Learning*, pages 560–569. PMLR, 2018.
- [8] Jeremy Bernstein, Jiawei Zhao, Kamyar Azizzadenesheli, and Anima Anandkumar. signsgd with majority vote is communication efficient and fault tolerant. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.
- [9] Aleksandr Beznosikov, Samuel Horváth, Peter Richtárik, and Mher Safaryan. On biased compression for distributed learning. *CoRR*, abs/2002.12410, 2020.
- [10] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, Jonathan Eckstein, et al. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine learning*, 3(1):1–122, 2011.
- [11] Ken Chang, Niranjan Balachandar, Carson K. Lam, Darvin Yi, James M. Brown, Andrew Beers, Bruce R. Rosen, Daniel L. Rubin, and Jayashree Kalpathy-Cramer. Distributed deep learning networks among institutions for medical imaging. *J. Am. Medical Informatics Assoc.*, 25(8):945–954, 2018.
- [12] Moses Charikar, Kevin C. Chen, and Martin Farach-Colton. Finding frequent items in data streams. In *Automata, Languages and Programming, 29th International Colloquium, ICALP 2002, Malaga, Spain, July 8-13, 2002, Proceedings*, volume 2380 of *Lecture Notes in Computer Science*, pages 693–703. Springer, 2002.
- [13] Congliang Chen, Li Shen, Haozhi Huang, Qi Wu, and Wei Liu. Quantized adam with error feedback. *arXiv preprint arXiv:2004.14180*, 2020.
- [14] Xiangyi Chen, Sijia Liu, Ruoyu Sun, and Mingyi Hong. On the convergence of A class of adam-type algorithms for non-convex optimization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

- [15] Trishul Chilimbi, Yutaka Suzue, Johnson Apacible, and Karthik Kalyanaraman. Project adam: Building an efficient and scalable deep learning training system. In *Symposium on Operating Systems Design and Implementation*, pages 571–582, 2014.
- [16] Dami Choi, Christopher J. Shallue, Zachary Nado, Jaehoon Lee, Chris J. Maddison, and George E. Dahl. On empirical comparisons of optimizers for deep learning. *CoRR*, abs/1910.05446, 2019.
- [17] Paul Covington, Jay Adams, and Emre Sargin. Deep neural networks for youtube recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems, Boston, MA, USA, September 15-19, 2016*, pages 191–198. ACM, 2016.
- [18] Christopher De Sa, Matthew Feldman, Christopher Ré, and Kunle Olukotun. Understanding and optimizing asynchronous low-precision stochastic gradient descent. In *Proceedings of the 44th Annual International Symposium on Computer Architecture*, pages 561–574, 2017.
- [19] Jeffrey Dean, Greg Corrado, Rajat Monga, Kai Chen, Matthieu Devin, Quoc V. Le, Mark Z. Mao, Marc’Aurelio Ranzato, Andrew W. Senior, Paul A. Tucker, Ke Yang, and Andrew Y. Ng. Large scale distributed deep networks. In *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States*, pages 1232–1240, 2012.
- [20] Tim Dettmers. 8-bit approximations for parallelism in deep learning. In Yoshua Bengio and Yann LeCun, editors, *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016.
- [21] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics, 2019.
- [22] John C Duchi, Alekh Agarwal, and Martin J Wainwright. Dual averaging for distributed optimization: Convergence analysis and network scaling. *IEEE Transactions on Automatic control*, 57(3):592–606, 2011.
- [23] John C. Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. In *COLT 2010 - The 23rd Conference on Learning Theory, Haifa, Israel, June 27-29, 2010*, pages 257–269, 2010.
- [24] Saeed Ghadimi and Guanghui Lan. Stochastic first-and zeroth-order methods for nonconvex stochastic programming. *SIAM Journal on Optimization*, 23(4):2341–2368, 2013.
- [25] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 2672–2680, 2014.
- [26] Priya Goyal, Piotr Dollár, Ross B. Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. Accurate, large minibatch SGD: training imagenet in 1 hour. *CoRR*, abs/1706.02677, 2017.
- [27] Alex Graves, Abdel-rahman Mohamed, and Geoffrey E. Hinton. Speech recognition with deep recurrent neural networks. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2013, Vancouver, BC, Canada, May 26-31, 2013*, pages 6645–6649. IEEE, 2013.
- [28] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 770–778. IEEE Computer Society, 2016.

- [29] Mingyi Hong, Davood Hajinezhad, and Ming-Min Zhao. Prox-pda: The proximal primal-dual algorithm for fast distributed nonconvex optimization and learning over networks. In *International Conference on Machine Learning*, pages 1529–1538, 2017.
- [30] Nikita Ivkin, Daniel Rothchild, Enayat Ullah, Vladimir Braverman, Ion Stoica, and Raman Arora. Communication-efficient distributed SGD with sketching. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 13144–13154, 2019.
- [31] Jiawei Jiang, Fangcheng Fu, Tong Yang, and Bin Cui. Sketchml: Accelerating distributed machine learning with data sketches. In *Proceedings of the 2018 International Conference on Management of Data, SIGMOD Conference 2018, Houston, TX, USA, June 10-15, 2018*, pages 1269–1284. ACM, 2018.
- [32] Peng Jiang and Gagan Agrawal. A linear speedup analysis of distributed deep learning with sparse and quantized communication. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 2530–2541, 2018.
- [33] Sai Praneeth Karimireddy, Quentin Rebjock, Sebastian U Stich, and Martin Jaggi. Error feedback fixes signsgd and other gradient compression schemes. *arXiv preprint arXiv:1901.09847*, 2019.
- [34] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- [35] Anastasia Koloskova, Sebastian U Stich, and Martin Jaggi. Decentralized stochastic optimization and gossip algorithms with compressed communication. In *International Conference on Machine Learning*, pages 3478–3487, 2019.
- [36] Yujun Lin, Song Han, Huizi Mao, Yu Wang, and Bill Dally. Deep gradient compression: Reducing the communication bandwidth for distributed training. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018.
- [37] Songtao Lu, Xinwei Zhang, Haoran Sun, and Mingyi Hong. Gnsd: A gradient-tracking based nonconvex stochastic algorithm for decentralized optimization. In *2019 IEEE Data Science Workshop (DSW)*, pages 315–321, 2019.
- [38] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial Intelligence and Statistics*, pages 1273–1282. PMLR, 2017.
- [39] Hiroaki Mikami, Hisahiro Sukanuma, Yoshiki Tanaka, Yuichi Kageyama, et al. Massively distributed sgd: Imagenet/resnet-50 training in a flash. *arXiv preprint arXiv:1811.05233*, 2018.
- [40] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin A. Riedmiller. Playing atari with deep reinforcement learning. *CoRR*, abs/1312.5602, 2013.
- [41] Parvin Nazari, Davoud Ataee Tarzanagh, and George Michailidis. Dadam: A consensus-based distributed adaptive gradient method for online optimization. *arXiv preprint arXiv:1901.09109*, 2019.
- [42] Angelia Nedic and Asuman Ozdaglar. Distributed subgradient methods for multi-agent optimization. *IEEE Transactions on Automatic Control*, 54(1):48, 2009.
- [43] Arkadi Nemirovski, Anatoli Juditsky, Guanghui Lan, and Alexander Shapiro. Robust stochastic approximation approach to stochastic programming. *SIAM Journal on optimization*, 19(4):1574–1609, 2009.
- [44] Sashank J Reddi, Satyen Kale, and Sanjiv Kumar. On the convergence of adam and beyond. In *International Conference on Learning Representations*, 2018.

- [45] Frank Seide, Hao Fu, Jasha Droppo, Gang Li, and Dong Yu. 1-bit stochastic gradient descent and its application to data-parallel distributed training of speech dnns. In *INTERSPEECH 2014, 15th Annual Conference of the International Speech Communication Association, Singapore, September 14-18, 2014*, pages 1058–1062. ISCA, 2014.
- [46] Zebang Shen, Aryan Mokhtari, Tengfei Zhou, Peilin Zhao, and Hui Qian. Towards more efficient stochastic decentralized learning: Faster convergence and sparse communication. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 4631–4640. PMLR, 2018.
- [47] Shaohuai Shi, Kaiyong Zhao, Qiang Wang, Zhenheng Tang, and Xiaowen Chu. A convergence analysis of distributed sgd with communication-efficient gradient sparsification. In *IJCAI*, pages 3411–3417, 2019.
- [48] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, Yutian Chen, Timothy P. Lillicrap, Fan Hui, Laurent Sifre, George van den Driessche, Thore Graepel, and Demis Hassabis. Mastering the game of go without human knowledge. *Nat.*, 550(7676):354–359, 2017.
- [49] Sebastian U Stich, Jean-Baptiste Cordonnier, and Martin Jaggi. Sparsified sgd with memory. In *Advances in Neural Information Processing Systems*, pages 4447–4458, 2018.
- [50] Sebastian U. Stich and Sai Praneeth Karimireddy. The error-feedback framework: Better rates for SGD with delayed gradients and compressed communication. *CoRR*, abs/1909.05350, 2019.
- [51] Athanasios Voulodimos, Nikolaos Doulamis, Anastasios D. Doulamis, and Eftychios Protopapadakis. Deep learning for computer vision: A brief review. *Comput. Intell. Neurosci.*, 2018:7068349:1–7068349:13, 2018.
- [52] Jianqiao Wangni, Jiale Wang, Ji Liu, and Tong Zhang. Gradient sparsification for communication-efficient distributed optimization. In *Advances in Neural Information Processing Systems*, pages 1299–1309, 2018.
- [53] Jian Wei, Jianhua He, Kai Chen, Yi Zhou, and Zuoyin Tang. Collaborative filtering and deep learning based recommendation system for cold start items. *Expert Systems with Applications*, 69:29–39, 2017.
- [54] Wei Wen, Cong Xu, Feng Yan, Chunpeng Wu, Yandan Wang, Yiran Chen, and Hai Li. Terngrad: Ternary gradients to reduce communication in distributed deep learning. *arXiv preprint arXiv:1705.07878*, 2017.
- [55] Jiayang Wu, Weidong Huang, Junzhou Huang, and Tong Zhang. Error compensated quantized SGD and its applications to large-scale distributed optimization. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 5321–5329. PMLR, 2018.
- [56] Guandao Yang, Tianyi Zhang, Polina Kirichenko, Junwen Bai, Andrew Gordon Wilson, and Chris De Sa. Swalp: Stochastic weight averaging in low precision training. In *International Conference on Machine Learning*, pages 7015–7024. PMLR, 2019.
- [57] Yang You, Jing Li, Sashank J. Reddi, Jonathan Hseu, Sanjiv Kumar, Srinadh Bhojanapalli, Xiaodan Song, James Demmel, Kurt Keutzer, and Cho-Jui Hsieh. Large batch optimization for deep learning: Training BERT in 76 minutes. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.
- [58] Tom Young, Devamanyu Hazarika, Soujanya Poria, and Erik Cambria. Recent trends in deep learning based natural language processing [review article]. *IEEE Comput. Intell. Mag.*, 13(3):55–75, 2018.

- 521 [59] Hao Yu, Rong Jin, and Sen Yang. On the linear speedup analysis of communication efficient
522 momentum SGD for distributed non-convex optimization. In *Proceedings of the 36th Inter-
523 national Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, Cal-
524 ifornia, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 7184–7193.
525 PMLR, 2019.
- 526 [60] Yue Yu, Jiayang Wu, and Junzhou Huang. Exploring fast and communication-efficient algo-
527 rithms in large-scale distributed networks. In *The 22nd International Conference on Artificial
528 Intelligence and Statistics, AISTATS 2019, 16-18 April 2019, Naha, Okinawa, Japan*, vol-
529 ume 89 of *Proceedings of Machine Learning Research*, pages 674–683. PMLR, 2019.
- 530 [61] Matthew D. Zeiler. ADADELTA: an adaptive learning rate method. *CoRR*, abs/1212.5701,
531 2012.
- 532 [62] Hantian Zhang, Jerry Li, Kaan Kara, Dan Alistarh, Ji Liu, and Ce Zhang. Zipml: Training
533 linear models with end-to-end low precision, and a little bit of deep learning. In *Proceedings of
534 the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia,
535 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 4035–
536 4043. PMLR, 2017.
- 537 [63] Lei Zhang, Shuai Wang, and Bing Liu. Deep learning for sentiment analysis: A survey. *Wiley
538 Interdiscip. Rev. Data Min. Knowl. Discov.*, 8(4), 2018.
- 539 [64] Tianyi Zhang, Felix Wu, Arzoo Katiyar, Kilian Q. Weinberger, and Yoav Artzi. Revisiting
540 few-sample BERT fine-tuning. *CoRR*, abs/2006.05987, 2020.
- 541 [65] Weijie Zhao, Deping Xie, Ronglai Jia, Yulei Qian, Ruiquan Ding, Mingming Sun, and Ping Li.
542 Distributed hierarchical GPU parameter server for massive scale deep learning ads systems. In
543 *Proceedings of Machine Learning and Systems 2020, MLSys 2020, Austin, TX, USA, March
544 2-4, 2020*. mlsys.org, 2020.
- 545 [66] Shuai Zheng, Ziyue Huang, and James T. Kwok. Communication-efficient distributed block-
546 wise momentum SGD with error-feedback. In *Advances in Neural Information Processing
547 Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS
548 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 11446–11456, 2019.
- 549 [67] Dongruo Zhou, Yiqi Tang, Ziyang Yang, Yuan Cao, and Quanquan Gu. On the convergence of
550 adaptive gradient methods for nonconvex optimization. *CoRR*, abs/1808.05671, 2018.

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] Section 5 on the limitations of using compression techniques
- (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]
- (b) Did you include complete proofs of all theoretical results? [Yes]

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)? [No] We can provide upon request
- (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
- (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [No] Yet our results Section 5 are average over several runs.
- (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)? [No]

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? [N/A]
- (b) Did you mention the license of the assets? [Yes]
- (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]
- (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]

589 A Intermediary Lemmas

590 **Lemma 1.** *Under Assumption 1 to Assumption 4 we have:*

$$\sum_{t=1}^T \mathbb{E} \|\bar{m}'_t\|^2 \leq T\sigma^2 + \sum_{\tau=1}^t \mathbb{E} [\|\nabla f(\theta_t)\|^2].$$

591 *Proof.* Firstly, the expected squared norm of average stochastic gradient can be bounded by

$$\begin{aligned} \mathbb{E}[\|\bar{g}_t^2\|] &= \mathbb{E}[\|\frac{1}{n} \sum_{i=1}^n g_{t,i} - \nabla f(\theta_t) + \nabla f(\theta_t)\|^2] \\ &= \mathbb{E}[\|\frac{1}{n} \sum_{i=1}^n (g_{t,i} - \nabla f_i(\theta_t))\|^2] + \mathbb{E}[\|\nabla f(\theta_t)\|^2] \\ &\leq \sigma^2 + \mathbb{E}[\|\nabla f(\theta_t)\|^2], \end{aligned}$$

592 where we use Assumption 4 that $g_{t,i}$ is unbiased and has bounded variance. Let $\bar{g}_{t,i}$ denote the i -th
593 coordinate of \bar{g}_t . By the updating rule of COMP-AMS

$$\begin{aligned} \mathbb{E}[\|\bar{m}'_t\|^2] &= \mathbb{E}[\|(1 - \beta_1) \sum_{\tau=1}^t \beta_1^{t-\tau} \bar{g}_\tau\|^2] \\ &\leq (1 - \beta_1)^2 \sum_{i=1}^d \mathbb{E}[(\sum_{\tau=1}^t \beta_1^{t-\tau} \bar{g}_{\tau,i})^2] \\ &\stackrel{(a)}{\leq} (1 - \beta_1)^2 \sum_{i=1}^d \mathbb{E}[(\sum_{\tau=1}^t \beta_1^{t-\tau})(\sum_{\tau=1}^t \beta_1^{t-\tau} \bar{g}_{\tau,i}^2)] \\ &\leq (1 - \beta_1) \sum_{\tau=1}^t \beta_1^{t-\tau} \mathbb{E}[\|\bar{g}_\tau\|^2] \\ &\leq \sigma^2 + (1 - \beta_1) \sum_{\tau=1}^t \beta_1^{t-\tau} \mathbb{E}[\|\nabla f(\theta_t)\|^2], \end{aligned}$$

594 where (a) is due to Cauchy-Schwartz inequality. Summing over $t = 1, \dots, T$, we obtain

$$\sum_{t=1}^T \mathbb{E} \|\bar{m}'_t\|^2 \leq T\sigma^2 + \sum_{t=1}^T \mathbb{E} [\|\nabla f(\theta_t)\|^2].$$

595 This completes the proof.

596

□

597 **Lemma 2.** *Under Assumption 4, we have for $\forall t$ and each local worker $\forall i \in [n]$,*

$$\begin{aligned} \|e_{t,i}\|^2 &\leq \frac{4q^2}{(1 - q^2)^2} G^2, \\ \mathbb{E}[\|e_{t+1,i}\|^2] &\leq \frac{4q^2}{(1 - q^2)^2} \sigma^2 + \frac{2q^2}{1 - q^2} \sum_{\tau=1}^t \left(\frac{1 + q^2}{2}\right)^{t-\tau} \mathbb{E}[\|\nabla f_i(\theta_\tau)\|^2]. \end{aligned}$$

598 *Proof.* We start by using Assumption 1 and Young's inequality to get

$$\begin{aligned} \|e_{t+1,i}\|^2 &= \|g_{t,i} + e_{t,i} - \mathcal{C}(g_{t,i} + e_{t,i})\|^2 \\ &\leq q^2 \|g_{t,i} + e_{t,i}\|^2 \\ &\leq q^2 (1 + \rho) \|e_{t,i}\|^2 + q^2 (1 + \frac{1}{\rho}) \|g_{t,i}\|^2 \\ &\leq \frac{1 + q^2}{2} \|e_{t,i}\|^2 + \frac{2q^2}{1 - q^2} \|g_{t,i}\|^2, \end{aligned} \tag{3}$$

599 by choosing $\rho = \frac{1-q^2}{2q^2}$. Now by recursion and the initialization $e_{1,i} = 0$, we have

$$\begin{aligned}\mathbb{E}[\|e_{t+1,i}\|^2] &\leq \frac{2q^2}{1-q^2} \sum_{\tau=1}^t \left(\frac{1+q^2}{2}\right)^{t-\tau} \mathbb{E}[\|g_{\tau,i}\|^2] \\ &\leq \frac{4q^2}{(1-q^2)^2} \sigma^2 + \frac{2q^2}{1-q^2} \sum_{\tau=1}^t \left(\frac{1+q^2}{2}\right)^{t-\tau} \mathbb{E}[\|\nabla f_i(\theta_\tau)\|^2],\end{aligned}$$

600 which proves the second argument. Meanwhile, the absolute bound $\|e_{t,i}\|^2 \leq \frac{4q^2}{(1-q^2)^2} G^2$ follows
601 directly from (3). \square

602 **Lemma 3.** For the moving average error sequence \mathcal{E}_t , it holds that

$$\sum_{t=1}^T \mathbb{E}[\|\mathcal{E}_t\|^2] \leq \frac{4Tq^2}{(1-q^2)^2} (\sigma^2 + \sigma_g^2) + \frac{4q^2}{(1-q^2)^2} \sum_{t=1}^T \mathbb{E}[\|\nabla f(\theta_t)\|^2].$$

603 *Proof.* Let $\bar{e}_{t,i}$ be the j -th coordinate of \bar{e}_t . Denote $K_{t,i} := \sum_{\tau=1}^t \left(\frac{1+q^2}{2}\right)^{t-\tau} \mathbb{E}[\|\nabla f_i(\theta_\tau)\|^2]$ and
604 $K_{t,i} = 0, \forall i \in [n]$. Using the same technique as in the proof of Lemma 1, we have

$$\begin{aligned}\mathbb{E}[\|\mathcal{E}_t\|^2] &= \mathbb{E}[\|(1-\beta_1) \sum_{\tau=1}^t \beta_1^{t-\tau} \bar{e}_\tau\|^2] \\ &\leq (1-\beta_1)^2 \sum_{j=1}^d \mathbb{E}[(\sum_{\tau=1}^t \beta_1^{t-\tau} \bar{e}_{\tau,j})^2] \\ &\stackrel{(a)}{\leq} (1-\beta_1)^2 \sum_{j=1}^d \mathbb{E}[(\sum_{\tau=1}^t \beta_1^{t-\tau}) (\sum_{\tau=1}^t \beta_1^{t-\tau} \bar{e}_{\tau,j}^2)] \\ &\leq (1-\beta_1) \sum_{\tau=1}^t \beta_1^{t-\tau} \mathbb{E}[\|\bar{e}_\tau\|^2] \\ &\leq (1-\beta_1) \sum_{\tau=1}^t \beta_1^{t-\tau} \mathbb{E}[\frac{1}{n} \sum_{i=1}^n \|e_{\tau,i}\|^2] \\ &\stackrel{(b)}{\leq} \frac{4q^2}{(1-q^2)^2} \sigma^2 + \frac{2q^2(1-\beta_1)}{(1-q^2)} \sum_{\tau=1}^t \beta_1^{t-\tau} (\frac{1}{n} \sum_{i=1}^n K_{\tau,i}),\end{aligned}$$

605 where (a) is due to Cauchy-Schwartz and (b) is a result of Lemma 2. Summing over $t = 1, \dots, T$
606 and using the technique of geometric series summation leads to

$$\begin{aligned}\sum_{t=1}^T \mathbb{E}[\|\mathcal{E}_t\|^2] &= \frac{4Tq^2}{(1-q^2)^2} \sigma^2 + \frac{2q^2(1-\beta_1)}{(1-q^2)} \sum_{t=1}^T \sum_{\tau=1}^t \beta_1^{t-\tau} (\frac{1}{n} \sum_{i=1}^n K_{\tau,i}) \\ &\leq \frac{4Tq^2}{(1-q^2)^2} \sigma^2 + \frac{2q^2}{(1-q^2)} \sum_{t=1}^T \sum_{\tau=1}^t \left(\frac{1+q^2}{2}\right)^{t-\tau} \mathbb{E}[\frac{1}{n} \sum_{i=1}^n \|\nabla f_i(\theta_\tau)\|^2] \\ &\leq \frac{4Tq^2}{(1-q^2)^2} \sigma^2 + \frac{4q^2}{(1-q^2)^2} \sum_{t=1}^T \mathbb{E}[\frac{1}{n} \sum_{i=1}^n \|\nabla f_i(\theta_t)\|^2] \\ &\stackrel{(a)}{\leq} \frac{4Tq^2}{(1-q^2)^2} \sigma^2 + \frac{4q^2}{(1-q^2)^2} \sum_{t=1}^T \mathbb{E}[\|\frac{1}{n} \sum_{i=1}^n \nabla f_i(\theta_t)\|^2] + \frac{1}{n} \sum_{i=1}^n \|\nabla f_i(\theta_t) - \nabla f(\theta_t)\|^2] \\ &\leq \frac{4Tq^2}{(1-q^2)^2} (\sigma^2 + \sigma_g^2) + \frac{4q^2}{(1-q^2)^2} \sum_{t=1}^T \mathbb{E}[\|\nabla f(\theta_t)\|^2],\end{aligned}$$

607 where (a) is derived by the variance decomposition and the last inequality holds due to Assumption 4.
608 The desired result is obtained.

609

□

610 **Lemma 4.** *It holds that $\forall t \in [T], \forall i \in [d], \hat{v}_{t,i} \leq \frac{4(1+q^2)^3}{(1-q^2)^2} G^2$.*

611 *Proof.* For any t , by Lemma 2 and Assumption 3 we have

$$\begin{aligned} \|\tilde{g}_t\|^2 &= \|\mathcal{C}(g_t + e_t)\|^2 \\ &\leq \|\mathcal{C}(g_t + e_t) - (g_t + e_t) + (g_t + e_t)\|^2 \\ &\leq 2(q^2 + 1)\|g_t + e_t\|^2 \\ &\leq 4(q^2 + 1)(G^2 + \frac{4q^2}{(1-q^2)^2} G^2) \\ &= \frac{4(1+q^2)^3}{(1-q^2)^2} G^2. \end{aligned}$$

612 It's then easy to show by the updating rule of \hat{v}_t ,

$$\hat{v}_{t,i} = (1 - \beta_2) \sum_{\tau=1}^t \beta_2^{t-\tau} \tilde{g}_{t,i}^2 \leq \frac{4(1+q^2)^3}{(1-q^2)^2} G^2.$$

613

□

614 B Proof of Theorem 1

615 **Theorem.** *Denote $C_0 = \sqrt{\frac{4(1+q^2)^3}{(1-q^2)^2} G^2 + \epsilon}$, $C_1 = \frac{\beta_1}{1-\beta_1} + \frac{2q}{1-q^2}$. Under Assumption 1 to Assump-*
 616 *tion 4, with $\eta_t = \eta \leq \frac{\epsilon}{3C_0\sqrt{2L\max\{2L, C_2\}}}$, for any $T > 0$, COMP-AMS satisfies*

$$\begin{aligned} \frac{1}{T} \sum_{t=1}^T \mathbb{E}[\|\nabla f(\theta_t)\|^2] &\leq 2C_0 \left(\frac{\mathbb{E}[f(\theta_1) - f(\theta^*)]}{T\eta} + \frac{\eta L \sigma^2}{n\epsilon} + \frac{3\eta^2 L C_0 C_1 \sigma^2}{\epsilon^2} \right. \\ &\quad \left. + \frac{12\eta^2 q^2 L C_0 \sigma_g^2}{(1-q^2)^2 \epsilon^2} + \frac{(1+C_1)G^2 d}{T\sqrt{\epsilon}} + \frac{\eta(1+2C_1)C_1 L G^2 d}{T\epsilon} \right). \end{aligned}$$

617 *Proof.* We first clarify some notations. At time t , let the full-precision gradient of the j -th worker
 618 be $g_{t,j}$, the error accumulator be $e_{t,j}$, and the compressed gradient be $\tilde{g}_{t,j} = \mathcal{C}(g_{t,j} + e_{t,j})$. Denote
 619 $\bar{g}_t = \frac{1}{n} \sum_{j=1}^N g_{t,j}$, $\bar{\tilde{g}}_t = \frac{1}{n} \sum_{j=1}^N \tilde{g}_{t,j}$ and $\bar{e}_t = \frac{1}{n} \sum_{j=1}^N e_{t,j}$. The second moment computed by the
 620 compressed gradients is denoted as $v_t = \beta_2 v_{t-1} + (1 - \beta_2) \bar{\tilde{g}}_t^2$, and $\hat{v}_t = \max\{\hat{v}_{t-1}, v_t\}$. Also, the
 621 first order moving average sequence

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) \bar{\tilde{g}}_t \quad \text{and} \quad m'_t = \beta_1 m'_{t-1} + (1 - \beta_1) \bar{g}_t.$$

622 By construction we have $m'_t = (1 - \beta_1) \sum_{i=1}^k \beta_1^{t-i} \bar{g}_t$.

623 Denote the following auxiliary sequences,

$$\begin{aligned} \mathcal{E}_{t+1} &:= (1 - \beta_1) \sum_{\tau=1}^{t+1} \beta_1^{t+1-\tau} \bar{e}_\tau \\ \theta'_{t+1} &:= \theta_{t+1} - \eta \frac{\mathcal{E}_{t+1}}{\sqrt{\hat{v}_t + \epsilon}}. \end{aligned}$$

624 Then,

$$\begin{aligned}
\theta'_{t+1} &= \theta_{t+1} - \eta \frac{\mathcal{E}_{t+1}}{\sqrt{\hat{v}_t + \epsilon}} \\
&= \theta_t - \eta \frac{(1 - \beta_1) \sum_{\tau=1}^t \beta_1^{t-\tau} \tilde{g}_\tau + (1 - \beta_1) \sum_{\tau=1}^{t+1} \beta_1^{t+1-\tau} \bar{e}_\tau}{\sqrt{\hat{v}_t + \epsilon}} \\
&= \theta_t - \eta \frac{(1 - \beta_1) \sum_{\tau=1}^t \beta_1^{t-\tau} (\tilde{g}_\tau + \bar{e}_{\tau+1}) + (1 - \beta) \beta_1^t \bar{e}_1}{\sqrt{\hat{v}_t + \epsilon}} \\
&= \theta_t - \eta \frac{(1 - \beta_1) \sum_{\tau=1}^t \beta_1^{t-\tau} \bar{e}_\tau}{\sqrt{\hat{v}_t + \epsilon}} - \eta \frac{m'_t}{\sqrt{\hat{v}_t + \epsilon}} \\
&= \theta_t - \eta \frac{\mathcal{E}_t}{\sqrt{\hat{v}_{t-1} + \epsilon}} - \eta \frac{m'_t}{\sqrt{\hat{v}_t + \epsilon}} + \eta \left(\frac{1}{\sqrt{\hat{v}_{t-1} + \epsilon}} - \frac{1}{\sqrt{\hat{v}_t + \epsilon}} \right) \mathcal{E}_t \\
&\stackrel{(a)}{=} \theta'_t - \eta \frac{m'_t}{\sqrt{\hat{v}_t + \epsilon}} + \eta \left(\frac{1}{\sqrt{\hat{v}_{t-1} + \epsilon}} - \frac{1}{\sqrt{\hat{v}_t + \epsilon}} \right) \mathcal{E}_t \\
&:= \theta'_t - \eta a'_t + \eta D_t \mathcal{E}_t,
\end{aligned}$$

625 where (a) uses the fact that for every $j \in [n]$, $\tilde{g}_{t,j} + e_{t+1,j} = g_{t,j} + e_{t,j}$, and $e_{t,1} = 0$ at initialization.

626 Further define the virtual iterates:

$$x_{t+1} := \theta'_{t+1} - \eta \frac{\beta_1}{1 - \beta_1} a'_t = \theta'_{t+1} - \eta \frac{\beta_1}{1 - \beta_1} \frac{m'_t}{\sqrt{\hat{v}_t + \epsilon}},$$

627 which follows the recurrence:

$$\begin{aligned}
x_{t+1} &= \theta'_{t+1} - \eta \frac{\beta_1}{1 - \beta_1} \frac{m'_t}{\sqrt{\hat{v}_t + \epsilon}} \\
&= \theta'_t - \eta \frac{m'_t}{\sqrt{\hat{v}_t + \epsilon}} - \eta \frac{\beta_1}{1 - \beta_1} \frac{m'_t}{\sqrt{\hat{v}_t + \epsilon}} + \eta D_t \mathcal{E}_t \\
&= \theta'_t - \eta \frac{\beta_1 m'_{t-1} + (1 - \beta_1) \bar{g}_t + \frac{\beta_1^2}{1 - \beta_1} m'_{t-1} + \beta_1 \bar{g}_t}{\sqrt{\hat{v}_t + \epsilon}} + \eta D_t \mathcal{E}_t \\
&= \theta'_t - \eta \frac{\beta_1}{1 - \beta_1} \frac{m'_{t-1}}{\sqrt{\hat{v}_t + \epsilon}} - \eta \frac{\bar{g}_t}{\sqrt{\hat{v}_t + \epsilon}} + \eta D_t \mathcal{E}_t \\
&= x_t - \eta \frac{\bar{g}_t}{\sqrt{\hat{v}_t + \epsilon}} + \eta \frac{\beta_1}{1 - \beta_1} D_t m'_{t-1} + \eta D_t \mathcal{E}_t.
\end{aligned}$$

628 When summing over $t = 1, \dots, T$, the difference sequence D_t satisfies the following bounds.

629 **Lemma 5.** Let $D_t := \frac{1}{\sqrt{\hat{v}_{t-1} + \epsilon}} - \frac{1}{\sqrt{\hat{v}_t + \epsilon}}$ be defined as above. Then,

$$\sum_{t=1}^T \|D_t\|_1 \leq \frac{d}{\sqrt{\epsilon}}, \quad \sum_{t=1}^T \|D_t\|^2 \leq \frac{d}{\epsilon}.$$

630 *Proof.* By the updating rule of COMP-AMS, $\hat{v}_{t-1} \leq \hat{v}_t$ for $\forall t$. Therefore, by the initialization
631 $\hat{v}_0 = 0$, we have

$$\begin{aligned}
\sum_{t=1}^T \|D_t\|_1 &= \sum_{t=1}^T \sum_{i=1}^d \left(\frac{1}{\sqrt{\hat{v}_{t-1,i} + \epsilon}} - \frac{1}{\sqrt{\hat{v}_{t,i} + \epsilon}} \right) \\
&= \sum_{i=1}^d \left(\frac{1}{\sqrt{\hat{v}_{0,i} + \epsilon}} - \frac{1}{\sqrt{\hat{v}_{T,i} + \epsilon}} \right) \\
&\leq \frac{d}{\sqrt{\epsilon}}.
\end{aligned}$$

632 For the sum of squared l_2 norm, note the fact that for $a \geq b > 0$, it holds that

$$(a - b)^2 \leq (a - b)(a + b) = a^2 - b^2.$$

633 Thus,

$$\begin{aligned} \sum_{t=1}^T \|D_t\|^2 &= \sum_{t=1}^T \sum_{i=1}^d \left(\frac{1}{\sqrt{\hat{v}_{t-1,i}} + \epsilon} - \frac{1}{\sqrt{\hat{v}_{t,i}} + \epsilon} \right)^2 \\ &\leq \sum_{t=1}^T \sum_{i=1}^d \left(\frac{1}{\hat{v}_{t-1,i} + \epsilon} - \frac{1}{\hat{v}_{t,i} + \epsilon} \right) \\ &\leq \frac{d}{\epsilon}, \end{aligned}$$

634 which gives the desired result. \square

635 By Assumption 2 we have

$$f(x_{t+1}) \leq f(x_t) - \eta \langle \nabla f(x_t), x_{t+1} - x_t \rangle + \frac{L}{2} \|x_{t+1} - x_t\|^2.$$

636 Taking expectation w.r.t. the randomness at time t , we obtain

$$\begin{aligned} &\mathbb{E}[f(x_{t+1})] - f(x_t) \\ &\leq -\eta \mathbb{E}[\langle \nabla f(x_t), \frac{\bar{g}_t}{\sqrt{\hat{v}_t} + \epsilon} \rangle] + \eta \mathbb{E}[\langle \nabla f(x_t), \frac{\beta_1}{1 - \beta_1} D_t m'_{t-1} + D_t \mathcal{E}_t \rangle] \\ &\quad + \frac{\eta^2 L}{2} \mathbb{E}[\| \frac{\bar{g}_t}{\sqrt{\hat{v}_t} + \epsilon} - \frac{\beta_1}{1 - \beta_1} D_t m'_{t-1} - D_t \mathcal{E}_t \|^2] \\ &= \underbrace{-\eta \mathbb{E}[\langle \nabla f(\theta_t), \frac{\bar{g}_t}{\sqrt{\hat{v}_t} + \epsilon} \rangle]}_I + \underbrace{\eta \mathbb{E}[\langle \nabla f(x_t), \frac{\beta_1}{1 - \beta_1} D_t m'_{t-1} + D_t \mathcal{E}_t \rangle]}_{II} \\ &\quad + \underbrace{\frac{\eta^2 L}{2} \mathbb{E}[\| \frac{\bar{g}_t}{\sqrt{\hat{v}_t} + \epsilon} - \frac{\beta_1}{1 - \beta_1} D_t m'_{t-1} - D_t \mathcal{E}_t \|^2]}_{III} + \underbrace{\eta \mathbb{E}[\langle \nabla f(\theta_t) - \nabla f(x_t), \frac{\bar{g}_t}{\sqrt{\hat{v}_t} + \epsilon} \rangle]}_{IV}, \end{aligned} \tag{4}$$

637 **Bounding term I.** We have

$$\begin{aligned} I &= -\eta \mathbb{E}[\langle \nabla f(\theta_t), \frac{\bar{g}_t}{\sqrt{\hat{v}_{t-1}} + \epsilon} \rangle] - \eta \mathbb{E}[\langle \nabla f(\theta_t), (\frac{1}{\sqrt{\hat{v}_t} + \epsilon} - \frac{1}{\sqrt{\hat{v}_{t-1}} + \epsilon}) \bar{g}_t \rangle] \\ &\leq -\eta \mathbb{E}[\langle \nabla f(\theta_t), \frac{\nabla f(\theta_t)}{\sqrt{\hat{v}_{t-1}} + \epsilon} \rangle] + \eta G^2 \mathbb{E}[\|D_t\|]. \\ &\leq -\frac{\eta}{\sqrt{\frac{4(1+q^2)^3}{(1-q^2)^2} G^2 + \epsilon}} \mathbb{E}[\|\nabla f(\theta_t)\|^2] + \eta G^2 \mathbb{E}[\|D_t\|_1], \end{aligned} \tag{5}$$

638 where we use Assumption 3, Lemma 4 and the fact that l_2 norm is no larger than l_1 norm.

639 **Bounding term II.** It holds that

$$\begin{aligned} II &\leq \eta (\mathbb{E}[\langle \nabla f(\theta_t), \frac{\beta_1}{1 - \beta_1} D_t m'_{t-1} + D_t \mathcal{E}_t \rangle] + \mathbb{E}[\langle \nabla f(x_t) - \nabla f(\theta_t), \frac{\beta_1}{1 - \beta_1} D_t m'_{t-1} + D_t \mathcal{E}_t \rangle]) \\ &\leq \eta \mathbb{E}[\|\nabla f(\theta_t)\| \| \frac{\beta_1}{1 - \beta_1} D_t m'_{t-1} + D_t \mathcal{E}_t \|] + \eta^2 L \mathbb{E}[\| \frac{\beta_1}{1 - \beta_1} m'_{t-1} + \mathcal{E}_t \| \| \frac{\beta_1}{1 - \beta_1} D_t m'_{t-1} + D_t \mathcal{E}_t \|] \\ &\leq \eta C_1 G^2 \mathbb{E}[\|D_t\|_1] + \frac{\eta^2 C_1^2 L G^2}{\sqrt{\epsilon}} \mathbb{E}[\|D_t\|_1], \end{aligned} \tag{6}$$

640 where $C_1 := \frac{\beta_1}{1-\beta_1} + \frac{2q}{1-q^2}$. The second inequality is because of smoothness of $f(\theta)$, and the last
 641 inequality is due to Lemma 2, Assumption 3 and the property of norms.

642 **Bounding term III.** This term can be bounded as follows:

$$\begin{aligned} III &\leq \eta^2 L \mathbb{E}[\|\frac{\bar{g}_t}{\sqrt{\hat{v}_t} + \epsilon}\|^2] + \eta^2 L \mathbb{E}[\|\frac{\beta_1}{1-\beta_1} D_t m'_{t-1} - D_t \mathcal{E}_t\|^2] \\ &\leq \frac{\eta^2 L}{\epsilon} \mathbb{E}[\|\frac{1}{n} \sum_{j=1}^n g_{t,j} - \nabla f(\theta_t) + \nabla f(\theta_t)\|^2] + \eta^2 L \mathbb{E}[\|D_t(\frac{\beta_1}{1-\beta_1} m'_{t-1} - \mathcal{E}_t)\|^2] \\ &\stackrel{(a)}{\leq} \frac{\eta^2 L}{\epsilon} \mathbb{E}[\|\nabla f(\theta_t)\|^2] + \frac{\eta^2 L \sigma^2}{n\epsilon} + \eta^2 C_1^2 L G^2 \mathbb{E}[\|D_t\|^2], \end{aligned} \quad (7)$$

643 where (a) follows from $\nabla f(\theta_t) = \frac{1}{n} \sum_{j=1}^n \nabla f_j(\theta_t)$ and Assumption 4 that $g_{t,j}$ is unbiased of
 644 $\nabla f_j(\theta_t)$ and has bounded variance σ^2 .

645 **Bounding term IV.** We have

$$\begin{aligned} IV &= \eta \mathbb{E}[\langle \nabla f(\theta_t) - \nabla f(x_t), \frac{\bar{g}_t}{\sqrt{\hat{v}_{t-1}} + \epsilon} \rangle] + \eta \mathbb{E}[\langle \nabla f(\theta_t) - \nabla f(x_t), (\frac{1}{\sqrt{\hat{v}_t} + \epsilon} - \frac{1}{\sqrt{\hat{v}_{t-1}} + \epsilon}) \bar{g}_t \rangle] \\ &\leq \eta \mathbb{E}[\langle \nabla f(\theta_t) - \nabla f(x_t), \frac{\nabla f(\theta_t)}{\sqrt{\hat{v}_{t-1}} + \epsilon} \rangle] + \eta^2 L \mathbb{E}[\|\frac{\frac{\beta_1}{1-\beta_1} m'_{t-1} + \mathcal{E}_t}{\sqrt{\hat{v}_{t-1}} + \epsilon}\| \|D_t g_t\|] \\ &\stackrel{(a)}{\leq} \frac{\eta \rho}{2\epsilon} \mathbb{E}[\|\nabla f(\theta_t)\|^2] + \frac{\eta}{2\rho} \mathbb{E}[\|\nabla f(\theta_t) - \nabla f(x_t)\|^2] + \frac{\eta^2 C_1 L G^2}{\sqrt{\epsilon}} \mathbb{E}[\|D_t\|] \\ &\stackrel{(b)}{\leq} \frac{\eta \rho}{2\epsilon} \mathbb{E}[\|\nabla f(\theta_t)\|^2] + \frac{\eta^3 L}{2\rho} \mathbb{E}[\|\frac{\frac{\beta_1}{1-\beta_1} m'_{t-1} + \mathcal{E}_t}{\sqrt{\hat{v}_{t-1}} + \epsilon}\|^2] + \frac{\eta^2 C_1 L G^2}{\sqrt{\epsilon}} \mathbb{E}[\|D_t\|_1], \end{aligned} \quad (8)$$

646 where (a) is due to Young's inequality and (b) is based on Assumption 2.

647 Regarding the second term in (8), by Lemma 3 and Lemma 1, summing over $t = 1, \dots, T$ we have

$$\begin{aligned} &\sum_{t=1}^T \frac{\eta^3 L}{2\rho} \mathbb{E}[\|\frac{\frac{\beta_1}{1-\beta_1} m'_{t-1} + \mathcal{E}_t}{\sqrt{\hat{v}_{t-1}} + \epsilon}\|^2] \\ &\leq \sum_{t=1}^T \frac{\eta^3 L}{2\rho\epsilon} \mathbb{E}[\|\frac{\beta_1}{1-\beta_1} m'_{t-1} + \mathcal{E}_t\|^2] \\ &\leq \sum_{t=1}^T \frac{\eta^3 L}{\rho\epsilon} \left[\frac{\beta_1^2}{(1-\beta_1)^2} \mathbb{E}[\|m'_t\|^2] + \mathbb{E}[\|\mathcal{E}_t\|^2] \right] \\ &\leq \frac{T\eta^3 \beta_1^2 L \sigma^2}{\rho(1-\beta_1)^2 \epsilon} + \frac{\eta^3 \beta_1^2 L}{\rho(1-\beta_1)^2 \epsilon} \sum_{t=1}^T \mathbb{E}[\|\nabla f(\theta_t)\|^2] \\ &\quad + \frac{4T\eta^3 q^2 L}{\rho(1-q^2)^2 \epsilon} (\sigma^2 + \sigma_g^2) + \frac{4\eta^3 q^2 L}{\rho(1-q^2)^2 \epsilon} \sum_{t=1}^T \mathbb{E}[\|\nabla f(\theta_t)\|^2] \\ &= \frac{T\eta^3 L C_2 \sigma^2}{\rho\epsilon} + \frac{4T\eta^3 q^2 L \sigma_g^2}{\rho(1-q^2)^2 \epsilon} + \frac{\eta^3 L C_2}{\rho\epsilon} \sum_{t=1}^T \mathbb{E}[\|\nabla f(\theta_t)\|^2], \end{aligned} \quad (9)$$

648 with $C_2 := \frac{\beta_1^2}{(1-\beta_1)^2} + \frac{4q^2}{(1-q^2)^2}$. Now integrating (5), (6), (7), (8) and (9) into (4), taking the tele-
 649 scoping summation over $t = 1, \dots, T$, we obtain

$$\begin{aligned} &\mathbb{E}[f(x_{T+1}) - f(x_1)] \\ &\leq (-\frac{\eta}{C_0} + \frac{\eta^2 L}{\epsilon} + \frac{\eta \rho}{2\epsilon} + \frac{\eta^3 L C_2}{\rho\epsilon}) \sum_{t=1}^T \mathbb{E}[\|\nabla f(\theta_t)\|^2] + \frac{T\eta^2 L \sigma^2}{n\epsilon} + \frac{T\eta^3 L C_2 \sigma^2}{\rho\epsilon} + \frac{4T\eta^3 q^2 L \sigma_g^2}{\rho(1-q^2)^2 \epsilon} \\ &\quad + (\eta(1+C_1)G^2 + \frac{\eta^2(1+C_1)C_1 L G^2}{\sqrt{\epsilon}}) \sum_{t=1}^T \mathbb{E}[\|D_t\|_1] + \eta^2 C_1^2 L G^2 \sum_{t=1}^T \mathbb{E}[\|D_t\|^2]. \end{aligned}$$

with $C_0 := \sqrt{\frac{4(1+q^2)^3}{(1-q^2)^2} G^2 + \epsilon}$. Setting $\eta \leq \frac{\epsilon}{3C_0 \sqrt{2L \max\{2L, C_2\}}}$ and choosing $\rho = \frac{\epsilon}{3C_0}$, we obtain

$$\begin{aligned} & \mathbb{E}[f(x_{T+1}) - f(x_1)] \\ & \leq -\frac{\eta}{2C_0} \sum_{t=1}^T \mathbb{E}[\|\nabla f(\theta_t)\|^2] + \frac{T\eta^2 L \sigma^2}{n\epsilon} + \frac{3T\eta^3 L C_0 C_2 \sigma^2}{\epsilon^2} + \frac{12T\eta^3 q^2 L C_0 \sigma_g^2}{(1-q^2)^2 \epsilon^2} \\ & \quad + \frac{\eta(1+C_1)G^2 d}{\sqrt{\epsilon}} + \frac{\eta^2(1+2C_1)C_1 L G^2 d}{\epsilon}. \end{aligned}$$

where the last inequality follows from Lemma 5. Re-arranging terms, we get that

$$\begin{aligned} \frac{1}{T} \sum_{t=1}^T \mathbb{E}[\|\nabla f(\theta_t)\|^2] & \leq 2C_0 \left(\frac{\mathbb{E}[f(x_1) - f(x_{T+1})]}{T\eta} + \frac{\eta L \sigma^2}{n\epsilon} + \frac{3\eta^2 L C_0 C_2 \sigma^2}{\epsilon^2} \right. \\ & \quad \left. + \frac{12\eta^2 q^2 L C_0 \sigma_g^2}{(1-q^2)^2 \epsilon^2} + \frac{(1+C_1)G^2 d}{T\sqrt{\epsilon}} + \frac{\eta(1+2C_1)C_1 L G^2 d}{T\epsilon} \right) \\ & \leq 2C_0 \left(\frac{\mathbb{E}[f(\theta_1) - f(\theta^*)]}{T\eta} + \frac{\eta L \sigma^2}{n\epsilon} + \frac{3\eta^2 L C_0 C_1 \sigma^2}{\epsilon^2} \right. \\ & \quad \left. + \frac{12\eta^2 q^2 L C_0 \sigma_g^2}{(1-q^2)^2 \epsilon^2} + \frac{(1+C_1)G^2 d}{T\sqrt{\epsilon}} + \frac{\eta(1+2C_1)C_1 L G^2 d}{T\epsilon} \right), \end{aligned}$$

where $C_0 = \sqrt{\frac{4(1+q^2)^3}{(1-q^2)^2} G^2 + \epsilon}$, $C_1 = \frac{\beta_1}{1-\beta_1} + \frac{2q}{1-q^2}$. The last inequality is because $\theta'_1 = \theta_1$, $\theta^* := \arg \min_{\theta} f(\theta)$ and the fact that $C_2 \leq C_1$. This completes the proof.

C Additional content

C.1 Extension to the single-machine setting

We first provide in this subsection the formulation of our method in the single-worker setting, see Algorithm 3. Here, the computations, of the stochastic gradient and the various moment estimates, are all performed on a single-machine and the data is stored in this same worker. Then, we establish its convergence rate with similar compression and error-feedback techniques, as seen prior.

Algorithm 3 COMP-AMS with error-feedback for a single-machine

- 1: **Input:** parameter β_1, β_2 , learning rate η_t .
 - 2: Initialize: central server parameter $\theta_1 \in \Theta \subseteq \mathbb{R}^d$; $e_1 = 0$ the error accumulator; sparsity parameter k ; $m_0 = 0, v_0 = 0, \hat{v}_0 = 0$
 - 3: **for** $t = 1$ to T **do**
 - 4: Compute stochastic gradient $g_t := g_{t,i_t}$ at θ_t for randomly sampled index i_t among the available observations indices.
 - 5: Compute $\tilde{g}_t = \text{Top-}k(g_t + e_t, k)$
 - 6: Update the error $e_{t+1} = e_t + g_t - \tilde{g}_t$
 - 7: $m_t = \beta_1 m_{t-1} + (1 - \beta_1) \tilde{g}_t$
 - 8: $v_t = \beta_2 v_{t-1} + (1 - \beta_2) \tilde{g}_t^2$
 - 9: $\hat{v}_t = \max(v_t, \hat{v}_{t-1})$
 - 10: Update the global model $\theta_{t+1} = \theta_t - \eta_t \frac{m_t}{\sqrt{\hat{v}_t + \epsilon}}$
 - 11: **end for**
-

660

□