A Flexible Framework for Communication-Efficient Machine Learning

Sarit Khirirat [1], Sindri Magnússon [2], Arda Aytekin [3], Mikael Johansson [1]
[1] Division of Decision and Control Systems, KTH Royal Institute of Technology, Sweden;
[2] Department of Computer and System Science, Stockholm University, Sweden; [3] Ericsson, Sweden
sarit@kth.se, sindri.magnusson@dsv.su.se, arda@aytekin.biz, mikaelj@kth.se

Abstract

With the increasing scale of machine learning tasks, it has become essential to reduce the communication between computing nodes. Early work on gradient compression focused on the bottleneck between CPUs and GPUs, but communication-efficiency is now needed in a variety of different system architectures, from high-performance clusters to energy-constrained IoT devices. In the current practice, compression levels are typically chosen before training and settings that work well for one task may be vastly sub-optimal for another dataset on another architecture. In this paper, we propose a flexible framework which adapts the compression level to the true gradient at each iteration, maximizing the improvement in the objective function that is achieved per communicated bit. Our framework is easy to adapt from one technology to the next by modeling how the communication cost depends on the compression level for the specific technology. Theoretical results and practical experiments indicate that the automatic tuning strategies significantly increase communication efficiency on several state-of-the-art compression schemes.

Introduction

The vast size of modern machine learning is shifting the focus on optimization and learning algorithms from centralized to distributed architectures. State-of-the-art models are now typically trained using multiple CPUs or GPUs, and data is increasingly being collected and processed in networks of resource-constrained devices, e.g., IoT devices, smart phones, or wireless sensors. This trend is shifting the bottleneck from the computation to the communication. The shift is particularly striking when learning is performed on energy-constrained devices that communicate over shared wireless channels. Indeed, distributed training is often communication bound since the associated optimization algorithms hinge on frequent transmissions of gradients between nodes. These gradients are typically huge: it is not uncommon for state-of-the-art models to have millions of parameters. To get a sense of the corresponding communication cost, transmitting a single gradient or stochastic gradient using single precision (32 bits per entry) requires 40 MB for a model with 10 million parameters. If we use 4G communications, this means that we can expect to transmit roughly one gradient per second. The huge communication load easily overburdens training even on loosely interconnected clusters and may render federated learning on some IoT or edge devices infeasible.

To alleviate the communication bottleneck, much recent research has focused on compressed gradient methods. These methods achieve communication efficiency by using only the most informative parts of the gradients at each iteration. We may, for example, sparsify the gradient, *i.e.* use only the most significant entries at each iteration and set the rest to be zero (Alistarh et al. 2017, 2018; Stich, Cordonnier, and Jaggi 2018; Wen et al. 2017; Wang et al. 2018; Khirirat, Johansson, and Alistarh 2018; Wangni et al. 2018). We may also quantize the gradient elements or do some mix of quantization and sparsification (Alistarh et al. 2017; Khirirat, Feyzmahdavian, and Johansson 2018; Magnússon et al. 2017; Wangni et al. 2018; Zhu et al. 2016; Rabbat and Nowak 2005; Vogels, Karimireddy, and Jaggi 2020; Magnússon, Shokri-Ghadikolaei, and Li 2020).

Several of the cited papers have demonstrated huge communication improvements for specific training problems. However, these communication benefits are often realized after a careful tuning of the compression level before training, *e.g.* the number of elements to keep when sparsifying the gradient. We cannot expect there to be a universally good compressor that works well on all problems, as shown by the worst-case communication complexity of any optimization methods in (Tsitsiklis and Luo 1987). There is generally a delicate problem-specific balance between compressing too much or too little. Trying to strike the right balance by hyper-parameter tuning is expensive and the resulting tuning parameters will be problem-specific. Moreover, most existing compression schemes are agnostic of the disparate communication costs for different technologies. In contrast, our proposed on-line mechanism adapts the compression level to each gradient information and each platform-specific communication cost.

Contributions: We consider deterministic and stochastic gradient methods in distributed settings where compressed gradients are communicated at every iteration. We propose a flexible framework for an on-line adaption of the gradient compression level to the problem data and communication technology used. This Communication-aware Adaptive Tuning (CAT) optimally adjusts the compression of each communicated gradient by maximizing the ratio between the guaranteed objective function improvement and the communication cost. The communication cost can be easily adjusted to the technology used, either by analytical models of the communication protocols or through empirical measurements. We illustrate these ideas on three state-of-the-art compression schemes: a) sparsification, b) sparsification with quantization, and c) stochastic sparsification. In all cases, we first derive descent lemmas specific to the compression, relating the function improvement to the tuning parameter. Using these results we can find the tuning that optimizes the communication efficiency measured in the descent direction relative to the given communication costs. Even though most of our theoretical results are for a single node case, we illustrate the efficiency of CAT to all three compression schemes in large-scale experiments in multi-node settings. For the stochastic sparsification we also prove convergence for stochastic gradient methods in the multi-node settings.

Related work on federated learning: Another approach to improve communication efficiency is to increase local computations in hope of reducing the number of iterations, and thereby the number of communication rounds. This is typically done by letting the nodes perform multiple gradient or proximal gradient updates locally before communicating their updated parameters, see, e.g., (Li et al. 2020; Khaled, Mishchenko, and Richtárik 2019; Stich 2019; Yu, Yang, and Zhu 2019; Wang and Joshi 2018),(Li et al. 2018). By adapting the number of local updates within a communication time interval (Wang and Joshi 2019), it is possible to find a good balance between the communication savings and sub-optimality guarantees of the solution. In contrast, we focus on adaptive compression, where our CAT framework strikes this balance by adjusting the compression level online, e.g. by optimizing the transmitted bits per iteration. In general, these two adaptive strategies can be combined to obtain further savings without compromising the convergence guarantees. In other words, CAT can be seen as a complement to existing adaptive schemes, addressing another dimension of communication efficiency.

Notation: We let $\mathbb{N}$, $\mathbb{N}_0$, and $\mathbb{R}$ be the set of natural numbers, the set of natural numbers including zero, and the set of real numbers, respectively. The set $\{a, a+1, \dots, b\}$ is denoted by $[a, b]$ where $a, b \in \mathbb{N}_0$ and $a \leq b$. For $x \in \mathbb{R}^d$, $\|x\|$ is the Euclidean norm and $\|x\|_q$ is the ℓ_q norm with $q \in (0, \infty]$. A continuously differentiable function $F : \mathbb{R}^d \to \mathbb{R}$ is L-smooth if $\|\nabla F(x) - \nabla F(y)\| \leq L\|x - y\|$ for all $x, y \in \mathbb{R}^d$. We say that F is μ-strongly convex if $F(y) \geq F(x) + \langle \nabla F(x), y - x\rangle + (\mu/2)\|y - x\|^2$.

Background

The main focus of this paper is empirical risk minimization

$$\underset{x \in \mathbb{R}^d}{\text{minimize}}\ F(x) = \frac{1}{|\mathcal{D}|} \sum_{z \in \mathcal{D}} L(x; z)$$

where $\mathcal{D}$ is a set of data points, x is the model parameters and $L(\cdot)$ is a loss function.

Gradient Compression

Consider the standard compressed gradient iteration

$$x^{i+1} = x^i - \gamma^i Q_T(\nabla F(x^i)). \tag{1}$$

Here, $Q_T(\cdot)$ is a compression operator and T is a parameter that controls the compression level. The goal is to achieve communication efficiency by using only the most significant information. One of the simplest compression schemes is to sparsify the gradient, *i.e.* we let

$$[Q_T(g)]_j = \begin{cases} g_j & \text{if } j \in I_T(g) \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

where $I_T(g)$ is the index set for the T components of g with largest magnitude. The following combination of sparsification and quantization has been shown to give good practical performance (Alistarh et al. 2017):

$$[Q_T(g)]_j = \begin{cases} ||g||\texttt{sign}(g_j) & \text{if } i \in I_T(g) \\ 0 & \text{otherwise.} \end{cases} \tag{3}$$

In this case, we communicate only the gradient magnitude and the sparsity pattern of the gradient. It is sometimes advantageous to use stochastic sparsification. Rather than sending the top T entries of each gradient, we then send T components on average. We can achieve this by setting

$$[Q_{T,p}(g)]_j = (g_j/p_j)\xi_j, \tag{4}$$

where $\xi_j \sim \texttt{Bernouli}(p_j)$ and $T = \sum_{j=1}^d p_j$. Ideally, we would like p_j to represent the magnitude of g_j, so that p_j should be large if $|g_j|$ is large relative to the other entries. There are many heuristic methods to choose p_j. For example, if we set $p_j = |g_j|/\|g\|_q$ with $q = 2$, $q = \infty$, and $q \in (0, \infty]$, then we get, respectively, the stochastic sparsifications in (Alistarh et al. 2017) with $s = 1$, the *TernGrad* in (Wen et al. 2017), and ℓ_q-quantization in (Wang et al. 2018). We can also optimize p, see (Wang et al. 2018) and the section titled "Dynamic Sparsification" for details.

Experimental results have shown huge communication savings by compressed gradient methods in large-scale machine learning (Shi et al. 2019a,b). Nevertheless, we can easily create pedagogical examples where they are no more communication efficient than full gradient descent. For sparsification, consider the function $F(x) = ||x||^2/2$. Then, gradient descent with the step-size $\gamma = 1$ converges in one iteration, and thus communicates only d floating points (one for each element of $\nabla F(x) \in \mathbb{R}^d$) to reach any ϵ-accuracy. On the other hand, T-sparsified gradient descent (where T divides d) needs d/T iterations, which implies d communicated floating points in total. In fact, the sparsified method is even worse since it requires additional $d\log(d)$ communicated bits to indicate the sparsity pattern.

This example shows that the benefits of sparsification cannot be seen on worst-case problems, and that traditional worst-case analysis (*e.g.* Khirirat, Johansson, and Alistarh (2018)) is unable to guarantee the improved communication complexity. Rather, sparsification is useful for exploiting the structure that appears in real-world problems. The key in exploiting this structure is to choose T properly at each iteration. In this paper we describe how to choose T dynamically to optimize the communication efficiency of sparsification.

Communication Cost: Bits, Packets, Energy and Beyond

The compressors discussed above have a tuning parameter T, which controls the sparsity budget of compressed gradient descent. Our goal is to tune T adaptively to optimize the communication efficiency. To explain how this is done, we first need to discuss how to model the communication cost. Let $C(T)$ denote the communication cost per iteration as a function of T, e.g., the total number of transmitted bits. Then, $C(T)$ consists of payload (actual data) and communication overhead. The payload is the number of bits required to communicate the compressed gradient. For the sparsification in Eq. (2) and the quantization in Eq. (3), the payload consumes, respectively,

$$\begin{aligned} P^{\mathrm{S}}(T) &= T \times (\lceil \log_2(d) \rceil + \mathtt{FPP}) \ \text{ bits and} \\ P^{\mathrm{SQ}}(T) &= \mathtt{FPP} + T \times \lceil \log_2(d) \rceil \ \text{ bits,} \end{aligned} \tag{5}$$

where the $\log_2(d)$ factor comes from indicating T indices in the d-dimensional gradient vector. Here FPP is our floating point precision, e.g., $\mathtt{FPP} = 32$ or $\mathtt{FPP} = 64$ for, respectively, single or double precision floating-points. Our simplest communication model accounts only for the payload, i.e., $C(T) = P(T)$. We call this the *payload model*. In real-world networks, however, each communication also includes the overhead and set-up costs. A more realistic model is therefore affine $C(T) = c_1 P(T) + c_0$, where $P(T)$ is the payload. Here c_0 is the communication overhead, while c_1 is the cost of transmitting a single payload byte. For example, if we just count transmitted bits ($c_1 = 1$), then a single UDP packet transmitted over the Ethernet requires an overhead of $c_0 = 54 \times 8$ bits and can have a payload of up to 1472 bytes. In the wireless standard IEEE 802.15.4, the overhead ranges from 23-82 bytes, leaving $51 - 110$ bytes of payload before the maximum packet size of 133 bytes is reached (Kozłowski and Sosnowski 2017). Another possibility is to use a *packet model*, *i.e.* to have a fixed cost per packet

$$C(T) = c_1 \times \lceil P(T)/P_{\max} \rceil + c_0, \tag{6}$$

where $P_{\max}$ is the number of payload bits per packet. The term $\lceil P(T)/P_{\max} \rceil$ counts the number of packets required to send the $P(T)$ payload bits, c_1 is the cost per packet, and c_0 is the cost of initiating the communication. These are just two examples; ideally, $C(T)$ should be tailored to the specific communication standard used, and possibly even estimated from system measurements.

Key Idea: Communication-Aware Adaptive Tuning (CAT)

When communicating the compressed gradients, we would like to use each bit as efficiently as possible. In optimization terms, we would like the objective function improvement for each communicated bit to be as large as possible. In other words, we want to maximize the ratio

$$\mathtt{Efficiency}(T) = \frac{\mathtt{Improvement}(T)}{C(T)}, \tag{7}$$

where $\mathtt{Improvement}(T)$ is the improvement in the objective function when we use T-sparsification as the given compressor. We will demonstrate how the value of $\mathtt{Improvement}(T)$ can be obtained from novel descent lemmas and derive dynamic sparsification policies which, at each iteration, find the T that optimizes $\mathtt{Efficiency}(T)$. We derive optimal T-values for the three compressors and two communication models introduced above. However, the idea is general and can be used to improve the communication efficiency for many other compression algorithms.

We begin by describing how our CAT framework can be applied to sparsified gradient methods. To this end, the following lemma introduces a useful measure $\alpha(T)$ of function value improvement:

Lemma 1. *Suppose that $F : \mathbb{R}^d \to \mathbb{R}$ is (possibly non-convex) L-smooth and $\gamma = 1/L$. Then for any $x, x^+ \in \mathbb{R}^d$ with $x^+ = x - \gamma Q_T(\nabla F(x))$ we have*

$$F(x^+) \le F(x) - \frac{\alpha(T)}{2L} ||\nabla F(x)||^2,$$

where $\alpha(T) = ||Q_T(\nabla F(x))||^2 / ||\nabla F(x)||^2$. Moreover, there are L-smooth functions $F(\cdot)$ for which the inequality is tight for every $T = 1, \ldots, d$.

This lemma is in the category of descent lemmas, which are standard tools to study the convergence for convex and non-convex functions. In fact, Lemma 1 generalizes the standard descent lemma for L-smooth functions (see, e.g., in Proposition A.24 in (Bertsekas 1999)). In particular, if the gradient $\nabla F(x)$ is T-sparse (or $T = d$) then Lemma 1 gives the standard descent

$$F(x^+) \le F(x) - \frac{1}{2L} ||\nabla F(x)||^2.$$

In the next subsection, we will use Lemma 1 to derive novel convergence rate bounds for sparsified gradient methods, extending many standard results for gradient descent. First, however, we will use $\mathtt{Improvement}(T) = \alpha(T)$ to define the following CAT mechanism for dynamic sparsification:

$$\textbf{Step 1: } T^i = \underset{T \in [1,d]}{\text{argmax}} \ \frac{\alpha^i(T)}{C(T)}, \tag{8}$$

$$\textbf{Step 2: } x^{i+1} = x^i - \frac{1}{L} Q_{T^i}(\nabla F(x^i)). \tag{9}$$

The algorithm first finds the sparsity budget T^i that optimizes the communication efficiency defined in (7), and then performs a standard sparsification using this value of T^i. Since $\|\nabla F(x)\|^2/2L$ is independent of T, we can maximize efficiency by maximizing $\alpha(T)/C(T)$.

To find T^i at each iteration we need to solve the maximization problem in Eq. (9). This problem has one dimension, and even a brute force search would be feasible in many cases. As the next two results show, however, the problem has a favourable structure that allows the maximization to be solved very efficiently. The first result demonstrates that the descent always increases with T and is bounded.

Lemma 2. *For $g \in \mathbb{R}^d$ the function $\alpha(T) = ||Q_T(g)||^2/||g||^2$ is increasing and concave when extended to the continuous interval $[0, d]$. Moreover, $\alpha(T) \ge T/d$*

for all $T \in \{0, \ldots d\}$ and there exists an L-smooth function such that $||Q_T(\nabla f(x))||^2/||\nabla f(x)||^2 = T/d$ for all $x \in \mathbb{R}^d$.

Lemma 2 results in many consequences in the next section, but first we make another observation:

Proposition 1. *Let $\alpha(T)$ be increasing and concave. If $C(T) = \tilde{c}_1 T + c_0$, then $\alpha(T)/C(T)$ is quasi-concave and has a unique maximum on $[0, d]$. When $C(T) = \tilde{c}_1 \lceil T/\tau_{\max} \rceil + c_0$, on the other hand, $\alpha(T)/C(T)$ attains its maximum for a T which is an integer multiple of $\tau_{\max}$.*

This proposition shows that the optimization in Eq. (9) is easy to solve. For the affine communication model, one can simply sort the elements in the decreasing order, initialize $T = 1$ and increase T until $\alpha(T)/C(T)$ decreases. In the packet model, the search for the optimal T is even more efficient, as one can increase T in steps of $\tau_{\max}$.

Dynamic Sparsification Benefits in Theory and Practice

Although the CAT framework applies to general communication costs, it is instructive to see what our results say about the communication complexity, i.e., the number of bits that needs to be communicated to guarantee that a solution is found within an ϵ-accuracy. Table 1 compares the iteration complexity of Gradient Descent (GD) in row 1 and T-Sparsified Gradient Descent (T-SGD) in rows 2 and 3 with constant T for strongly-convex, convex, and non-convex problems. The results for gradient descent are well-known and found in, e.g., (Nesterov 2018), while the worst-case analysis is from (Khirirat, Johansson, and Alistarh 2018). The results for T-sparsified gradient descent are derived using Lemma 1 instead of the standard descent lemma; see proofs in the supplementary materials.

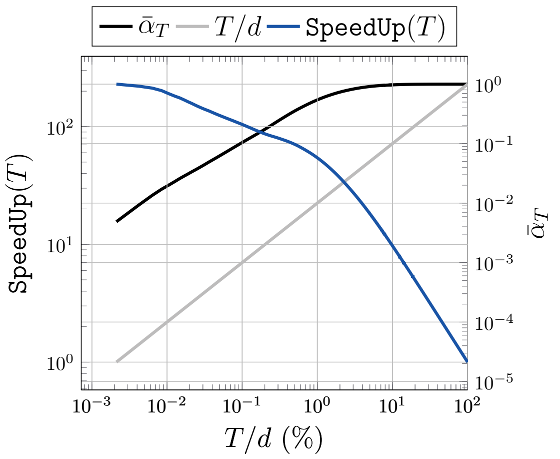
Comparing rows 1 and 3 in the table, we see that the worst-case analysis does not guarantee any improvements in the number of communicated floating points. Although T-SGD only communicates T out of d gradient entries in each round, we need to perform d/T times more iterations with T-SGD than with SGD, so both of these approaches will need to communicate the same number of floating points. In fact, T-SGD will be worse in terms of communicated bits since it requires $T\lceil \log_2(d) \rceil$ additional bits per iteration to indicate the sparsity pattern.

Let us now turn our attention to our novel analysis shown in row 2 of Table 1. Here, the parameter $\bar{\alpha}_T$ is a lower bound on $\alpha^i(T)$ over every iteration, that is

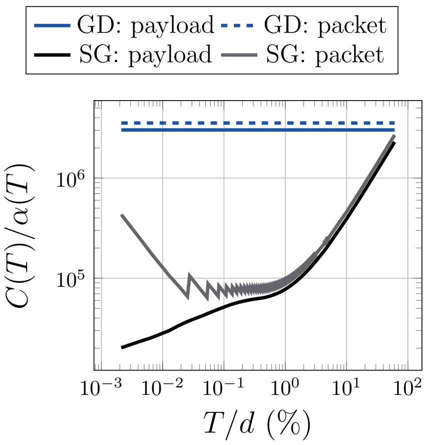
$$\alpha^i(T) \geq \bar{\alpha}_T \quad \text{for all } i.$$

Unfortunately, $\bar{\alpha}_T$ is not useful for algorithm development: we know from Lemma 2 that it can be as low as T/d, and it is not easy to compute a tight data-dependent bound off-line, since $\bar{\alpha}_T$ depends on the iterates produced by the algorithm. However, $\bar{\alpha}_T$ explains why gradient sparsification is communication efficient. In practice, only few top entries cover the majority of the gradient energy, so $\alpha^i(T)$ grows rapidly for small values of T and is much larger than T/d.

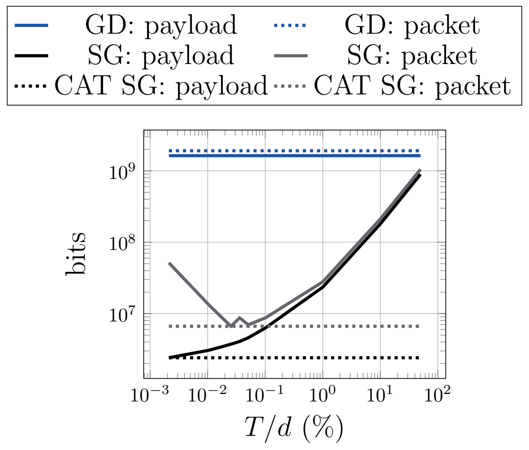
To illustrate the benefits of sparsification, let us look at the concrete example of logistic regression on the standard



(a) $\bar{\alpha}_T$ and speedup



(b) Hypothetical: Communications to reach ϵ-accuracy



(c) Experiments: Communications (in bits) to reach ϵ-accuracy

Figure 1: CAT sparsified gradient descent on the RCV1 data set.

	Deterministic Sparsification			Stochastic Sparsification		
Upper-Bound	μ-convex	convex	nonconvex	μ-convex	convex	nonconvex
No-Compression	A_ϵ^{SC}	A_ϵ^{C}	A_ϵ^{NC}	B_ϵ^{SC}	B_ϵ^{C}	B_ϵ^{NC}
Data-Dependent	$\frac{1}{\bar{\alpha}_T} \cdot A_\epsilon^{\text{SC}}$	$\frac{1}{\bar{\alpha}_T} \cdot A_\epsilon^{\text{C}}$	$\frac{1}{\bar{\alpha}_T} \cdot A_\epsilon^{\text{NC}}$	$\frac{1}{\bar{\omega}_T} \cdot B_\epsilon^{\text{SC}}$	$\frac{1}{\bar{\omega}_T} \cdot B_\epsilon^{\text{C}}$	$\frac{1}{\bar{\omega}_T} \cdot B_\epsilon^{\text{NC}}$
Worst-Case	$\frac{d}{T} \cdot A_\epsilon^{\text{SC}}$	$\frac{d}{T} \cdot A_\epsilon^{\text{C}}$	$\frac{d}{T} \cdot A_\epsilon^{\text{NC}}$	$\frac{d}{T} \cdot B_\epsilon^{\text{SC}}$	$\frac{d}{T} \cdot B_\epsilon^{\text{C}}$	$\frac{d}{T} \cdot B_\epsilon^{\text{NC}}$

Table 1: Iteration complexity for T-sparsified (stochastic) gradient descent. We prove these results and analogous results for $S+Q$ compression in the Appendix. For Deterministic Sparsification, A_ϵ^{SC}, A_ϵ^{C}, and A_ϵ^{NC} are standard upper bounds for gradient descent on iteration counts needed to achieve ϵ-accuracy for, respectively, strongly-convex, convex, and non-convex problems. In particular, $A_\epsilon^{\text{SC}} = \kappa \log(\epsilon_0/\epsilon)$, $A_\epsilon^{\text{C}} = 2LR^2/\epsilon$, $A_\epsilon^{\text{NC}} = 2L\epsilon_0/\epsilon$, where $\kappa = L/\mu$, $\epsilon_0 = F(x^0) - F^\star$, and R is a constant such that $||x^i - x^\star|| \leq R$ (for some optimizer $x^\star$). For Stochastic Sparsification, B_ϵ^{SC}, B_ϵ^{C}, and B_ϵ^{NC} are standard upper bounds for multi-node stochastic gradient descent on iteration counts needed to achieve ϵ-accuracy (in expected value) for, respectively, strongly-convex, convex, and non-convex problems. In particular, $B_\epsilon^{\text{SC}} = 2(1 + 2\sigma^2/(\mu\epsilon L))(A_\epsilon^{\text{SC}} + \delta)$, $B_\epsilon^{\text{C}} = 2(1+2\sigma^2/(\epsilon L))A_\epsilon^{\text{C}}$, and $B_\epsilon^{\text{NC}} = 2(1+2\sigma^2/\epsilon)A_\epsilon^{\text{NC}}$, where $\delta = \kappa\log(2)$ and σ^2 is a variance bound of stochastic gradients. The ϵ-accuracy is measured in $\mathbf{E}[F(x) - F(x^\star)]$ for convex problems, and in $\mathbf{E}\|\nabla F(x)\|^2$ otherwise.

benchmark data set `RCV1` (with $d = 47,236$ and $697,641$ data points). Figure 1a depicts $\bar{\alpha}_T$ computed after running 1000 iterations of gradient descent and compares it to the worst case bound T/d. The results show a dramatic difference between these two measures. We quantify this difference by the ratio

$$\texttt{SpeedUp}(T) = \frac{d}{T} \bigg/ \frac{1}{\bar{\alpha}_T} = \frac{\bar{\alpha}_T}{T/d}.$$

Note that this measure is the ratio between rows 2 and 3 in Table 1, and hence tells us the hypothetical speedup by sparsification, i.e., the ratio between the number of communicated floating points needed by GD and T-SGD to reach ϵ-accuracy. The figure shows a dramatic speedup; for small values of T, the speed-up is of three orders of magnitude (we confirm this in experiments below).

Interestingly, the speedup decreases with T and is maximized at $T = 1$. This happens because doubling T doubles the number of communicated bits, while the additional descent is often less significant. Thus, an increase in T worsens communication efficiency. This suggests that we should always take $T = 1$ if the communication efficiency in terms of bits is optimized without considering overhead. In the context of the dynamic algorithm in Eq. (9), this leads to the following result:

Proposition 2. *Consider the dynamic sparsified gradient algorithm in Eq.* (9) *with $C(T) = P^{\text{S}}(T)$ given by Eq. (5). Then, the maximization problem* (9) *has the solution $T^i = 1$ for all i.*

Figures 1b and 1c depict, respectively, the hypothetical and true values of the total number of bits needed to reach an ϵ-accuracy for different communication models. In particular, Figure 1b depicts the ratio $C(T)/\bar{\alpha}_T$ (compare with Table 1) and Figure 1c depicts the experimental results of running T-SGD for different values of T. We consider: a) the payload model with $C(T) = P^{\text{S}}(T)$ (dashed lines) and b) the packet model in Eq. (6) with $c_1 = 128$ bytes, $c_0 = 64$ bytes and $P_{\max} = 128$ bytes (solid lines). In both cases, the floating point precision is $\texttt{FPP} = 64$. We compare the results with GD (blue lines) with payload $d \times \texttt{FPP}$ bits per iteration. As expected, if we ignore overheads, then $T = 1$ is optimal and the improvement compared to GD are of three orders of magnitude. For the packet model, there is a delicate balance between choosing T too small and too big. For general communication models it is difficult to find the right value of T *a priori*, and the costs of choosing a bad T can be of many orders of magnitude. To find a good T we could do a hyper-parameter search, e.g. by first estimating $\bar{\alpha}_T$ from data and then by using it to find the optimal T. However, this will be expensive and moreover $\bar{\alpha}_T$ might not be a good estimate of $\alpha^i(T)$ we get at each iteration. In contrast, our CAT framework finds the optimal T at each iteration without any hyper-parameter optimization. In Figure 1c we show the number of communicated bits needed to reach ϵ-accuracy with our algorithm. The results show that for both communication models, our algorithm achieves the same communication efficiency as if we would choose the optimal T.

Dynamic Sparsification + Quantization

We now describe how our CAT framework can improve the communication efficiency of compressed gradient methods that use sparsification combined with quantization, i.e., using $Q_T(\cdot)$ in Equation (3). As before, our goal is to choose T^i dynamically by maximizing the communication efficiency per iteration defined in (7). This selection can be performed based on the following descent lemma.

Lemma 3. *Suppose that $F : \mathbb{R}^d \to \mathbb{R}$ is (possibly nonconvex) L-smooth. Then for any $x, x^+ \in \mathbb{R}^d$ with $x^+ = x - \gamma Q_T(\nabla F(x))$ where $Q_T(\cdot)$ is as defined in Eq.* (3) *and $\gamma = \sqrt{\beta(T)}/(\sqrt{T}L)$ then*

$$F(x^+) \leq F(x) - \frac{\beta(T)}{2L}||\nabla F(x)||^2,$$

where $\beta(T) = \langle \nabla F(x), Q_T(\nabla F(x))\rangle^2 / \big(T \cdot \|\nabla F(x)\|^4\big)$.

Since this compression operator affects the descent differently from sparsification, this lemma differs from

Lemma 1, e.g, in terms of the step-size and descent measure ($\beta(T)$ vs. $\alpha(T)$). Unlike $\alpha(T)$ in Lemma 1, $\beta(T)$ does not converge to 1 as T goes to d. In fact, $\beta(T)$ is not even an increasing function, and $Q_T(g)$ does not converge to g when T increases. Nevertheless, $\langle \nabla F(x), Q_T(\nabla F(x))\rangle^2$ is non-negative, increasing and concave. Under the affine communication model, $T \times C(T) = \tilde{c}_0 T^2 + c_1 T$ is non-negative and convex, which implies that $\beta(T)/C(T)$ is quasi-concave. The optimal T can then be efficiently found similarly to what was done for the CAT-sparsification. Therefore, Lemma 3 allows us to apply the CAT framework for this compression. In particular, with $\beta^i(T) = \langle \nabla F(x^i), Q_T(\nabla F(x^i))\rangle^2 / \left(T \cdot \|\nabla F(x^i)\|_2^4\right)$ we get the algorithm

$$\textbf{Step 1: } T^i = \underset{T\in[1,d]}{\text{argmax}} \frac{\beta^i(T)}{C(T)} \tag{10}$$

$$\textbf{Step 2: } \gamma^i = \frac{\sqrt{\beta^i(T^i)}}{\sqrt{T^i}L} \tag{11}$$

$$\textbf{Step 3: } x^{i+1} = x^i - \gamma^i Q_{T^i}(\nabla F(x^i)). \tag{12}$$

The algorithm optimizes T^i, based on each gradient and the actual communication cost. Note that Alistarh et al. (2017) proposes a dynamic mechanism that chooses T^i so that $I_{T^i}(g^i)$ is the smallest subset such that $\sum_{j\in I_{T^i}(g^i)} |g_j^i| \geq \|g^i\|$. However, this heuristic has no clear connection to the descent or consideration for communication costs. Our experiments show that our framework outperforms this heuristic in both running time and communication efficiency.

We compared CAT to the dynamic tuning introduced in (Alistarh et al. 2017). In Figure 2, algorithms with both tuning rules are comparable if we only account for the payload in Equation (5). However, the heuristic rule in (Alistarh et al. 2017) is agnostic to the actual communication model $C(T)$ in Equation (6) with $c_1 = 128$ bytes, $c_0 = 64$ bytes and $P_{\max} = 128$ bytes. The blue lines show that the CAT is roughly two times more communication efficient than the dynamic tuning rule in (Alistarh et al. 2017) for the packet communication model.

Dynamic Stochastic Sparsification: Stochastic Gradient & Multiple Nodes

We finally illustrate how the CAT framework can improve the communication efficiency of stochastic sparsification. Our goal is to choose T^i and p^i dynamically for the stochastic sparsification in Eq. (4) to maximize the communication efficiency per iteration. To this end, we need the following descent lemma, similarly to the ones we proved for deterministic sparsifications in the last two sections.

Lemma 4. *Suppose that $F : \mathbb{R}^d \to \mathbb{R}$ is (possibly nonconvex) L-smooth. Then for any $x, x^+ \in \mathbb{R}^d$ with $x^+ = x - \gamma Q_{T,p}(\nabla F(x))$ where $Q_{T,p}(\cdot)$ is defined in* (4) *and $\gamma = \omega_p(T)/L$ we have*

$$\mathbf{E}F(x^+) \leq \mathbf{E}F(x) - \frac{\omega_p(T)}{2L}\mathbf{E}||\nabla F(x)||^2,$$

where $\omega_p(T) = ||\nabla F(x)||^2 / \mathbf{E}||Q_{T,p}(\nabla F(x))||^2$.

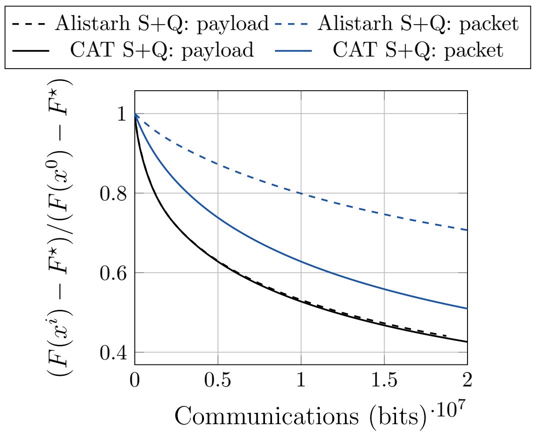


Figure 2: CAT sparsification + quantization on the RCV1 data set.

Similarly as before, we optimize the descent and the communication efficiency by maximizing, respectively, $\omega_p(T)$ and $\omega_p(T)/C(T)$. For a given T, the $p^\star$ minimizing $\omega_p(T)$ can be found efficiently, see (Wang et al. 2018) and our discussion in the supplementary materials. In this paper we always use $p^\star$ and omit p in $Q_T(\cdot)$ and $\omega(T)$. We can now use our CAT framework to optimize the communication efficiency. If we set $\omega^i(T) = ||\nabla F(x^i)||^2 / \mathbf{E}||Q_{T,p}(\nabla F(x^i))||^2$ we get the dynamic algorithm:

$$\textbf{Step 1: } T^i = \underset{B\in[1,d]}{\text{argmax}} \frac{\omega^i(T)}{C(T)} \tag{13}$$

$$\textbf{Step 2: } \gamma^i = \frac{\omega^i(T^i)}{L} \tag{14}$$

$$\textbf{Step 3: } x^{i+1} = x^i - \gamma^i Q_{T^i}(\nabla F(x^i)). \tag{15}$$

This algorithm can maximize communication efficiency by finding the optimal sparsity budget T to the one-dimensional problem. This can be solved efficiently since the sparsification parameter $\omega(T)$ has properties that are similar to $\alpha(T)$ for deterministic sparsification. Like Lemma 2 for deterministic sparsification, the following result shows that $\omega(T)$ is increasing with the budget $T \in [1, d]$ and is lower-bounded by T/d.

Lemma 5. *For any vector $g \in \mathbb{R}^d$ the function $\omega(T) = \|g\|^2 / \|Q_{T,p}(g)\|^2$ is increasing over $T \in [1, d]$. Moreover, $\omega(T) \geq T/d$ for all $T \in [1, d]$, where we obtain the equality when $p_j = T/d$ for all j.*

This lemma leads to many consequences for $\omega(T)$, analogous to $\alpha(T)$. For instance, by following proof arguments in Proposition 1, $\omega(T)/C(T)$ attains its maximum for a T which is an integer multiple of $\tau_{\max}$ when $C(T) = \tilde{c}_1 \lceil T/\tau_{\max} \rceil + c_0$.

Furthermore, stochastic sparsification has some favorable properties that allow us to generalize our theoretical results to stochastic gradient methods and to multi-node settings. Suppose that we have n nodes that wish to solve the minimization problem with $F(x) = (1/n)\sum_{j=1}^n f_j(x)$ where

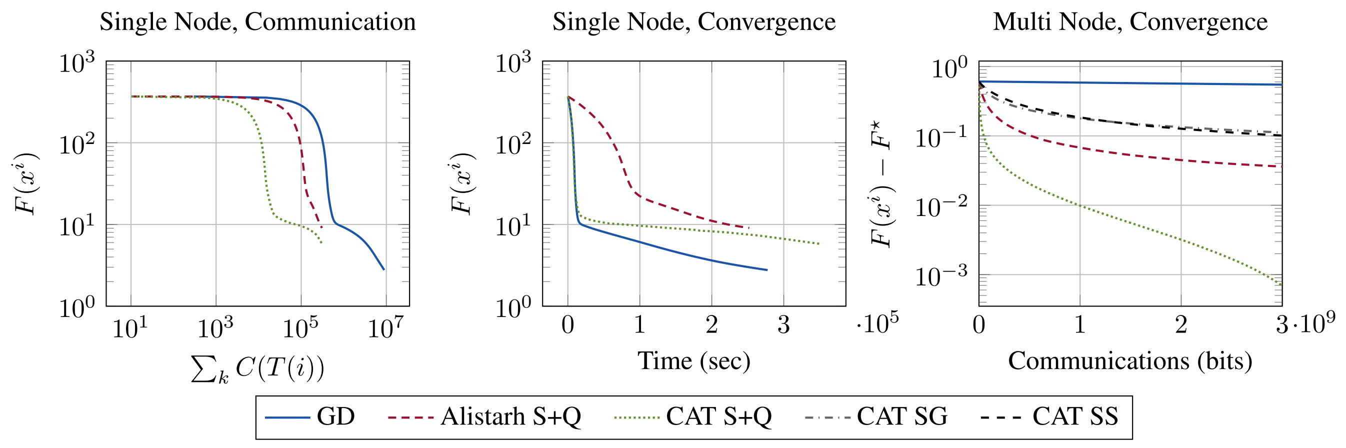


Figure 3: Performance of CAT frameworks on three compressors for solving logistic regression problems. We used the URL data set in the single-node (one master/one worker) architecture and RCV1 in the multi-node (one master/four worker) setting.

$f_j(\cdot)$ is kept by node j. Then, we may solve the problem by distributed compressed gradient descent

$$x^{i+1} = x^i - \gamma \frac{1}{n} \sum_{j=1}^{n} Q_{T_j^i}\left(g_j(x^i; \xi_j^i)\right), \tag{16}$$

where $Q(\cdot)$ is the stochastic sparsifier and $g_j(x;\xi_j)$ is a stochastic gradient at x. We assume that $g_j(x;\xi_j)$ is unbiased and satisfies a bounded variance assumption, i.e. $\mathbf{E}_\xi g_j(x;\xi_j) = \nabla f_j(x)$ and $\mathbf{E}_\xi \|g_j(x;\xi_j) - \nabla F(x)\|^2 \leq \sigma^2$. The expectation is with respect to a local data distribution at node j. These conditions are standard to analyze randomized first-order algorithms in machine learning (Feyzmahdavian, Aytekin, and Johansson 2016; Lian et al. 2015).

We can derive a descent lemma for Algorithm (16) similarly as Lemma 4 for the single-node sparsification method (see Theorem 3 in the Appendix). This means that we easily prove similar data-dependent convergence results as we did for deterministic sparsification in Table 1. To illustrate this, suppose that for a given T there is $\bar{\omega}_T$ satisfying $\omega_j^i(T) \geq \bar{\omega}_T$ where $\omega_j^i(T) = ||\nabla F_j(x^i)||^2/\mathbf{E}||Q_T(\nabla F_j(x^i))||^2$. Then, the iteration complexity of Algorithm (16) is as given in the right part of Table 1. The parameter $\bar{\omega}_T$ captures the sparsification gain, similarly as $\bar{\alpha}_T$ did for deterministic sparsification. In the worst case there is no communication improvement of sparsification compared to sending full gradients, but when $\bar{\omega}_T$ is large the communication improvement can be significant.

Experimental Results

Experiment 1 (single node). We evaluate the performance of our CAT framework for dynamic sparsification and quantization (S+Q) in the single-master, single-worker setup on the URL data set with 2.4 million data points and 3.2 million features. The master node, located 500 km away from the worker node, is responsible for maintaining the decision variables based on the gradient information received from the worker node. The nodes communicate with each other over a 1000 Mbit Internet connection using the ZMQ library. We implemented vanilla gradient descent (GD), Alistarh's S+Q (Alistarh et al. 2017) and CAT- S+Q using the C++ library POLO (Aytekin, Biel, and Johansson 2018). We first set FPP $= 32$ and measure the communication cost in a wall-clock time. After obtaining a linear fit to the measured communication cost (see the supplementary materials for details), we ran $30,000$ iterations and with step-size according to Lemma 3. Figure 3 shows the loss improvement with respect to the total communication cost (leftmost) and wall-clock time (middle). We observe that CAT S+Q outperforms GD and Alistarh's S+Q up to two orders and one order of magnitude, respectively, in communication efficiency. In terms of wall-clock time, CAT S+Q takes 26% (respectively, 39%) more time to finish the full $30,000$ iterations than that of GD (respectively, Alistarh's S+Q). Note, however, that CAT S+Q achieves an order of magnitude loss improvement in an order of magnitude shorter time, and the loss value is always lower in CAT S+Q than that in Alistarh's S+Q. Such a performance is desirable in most of the applications (e.g., hyper-parameter optimizations and day-ahead market-price predictions) that do not impose a strict upper bound on the iteration counts but rather on the wall-clock time of the algorithm.

Experiment 2 (MPI - multiple nodes): We evaluate the performance of our CAT tuning rules on deterministic sparsification (SG), stochastic sparsification (SS), and sparsification with quantization (S+Q) in a multi-node setting on RCV1. We compare the results to gradient descent and Alistarh's S+Q (Alistarh et al. 2017). We implement all algorithms in Julia, and run them on 4 nodes using MPI, splitting the data evenly between the nodes. In all cases we use the packet communication model (6) with $c_1 = 576$ bytes, $c_0 = 64$ bytes and $P_{\max} = 512$ bytes. The right-most plot in Figure 3 shows that our CAT S+Q outperforms all other compression schemes. In particular, CAT is roughly 6 times more communication efficient than the dynamic rule in (Alistarh et al. 2017) for the same compression scheme (compare number of bits needed to reach $\epsilon = 0.4$).

Conclusions

We have proposed communication-aware adaptive tuning to optimize the communication-efficiency of gradient sparsification. The adaptive tuning relies on the data-dependent measure of an objective function improvement, and adapts the compression level to maximize the descent per communicated bit. Unlike existing heuristics, our tuning rules are guaranteed to save communications in realistic communication models. In particular, our rules are more communication-efficient when communication overhead or packet transmissions are accounted for. We experimentally showed how CAT can be used to optimize a transmission time (Experiment 1) and communicated bits (Experiment 2). Moreover, we illustrated the promise of CAT in multi-node settings with all compressions considered (Experiment 2).

From the theoretical point of view we showed that worst-case analysis of gradient compression does not guarantee *any* advantages or provide insight into why compression improves communication efficiency. However, we provide problem/data dependent convergence results that demonstrate these benefits of compression on problems with favorable structures. Our theoretical results cover multi-node settings if the compression is stochastic and unbiased, and single-node settings if the compression is deterministic. To the best of our knowledge, all analytical results on multi-node compression use such assumptions or assumptions on the similarity of the nodes' loss functions. The single-node setting helps us quantify the communication-efficiency ratio and develop the CAT framework. Moreover, this setting is a representative abstraction of many industrial applications. Consider the case in which either the problem data (available in workers) or the optimization problem (available in the master) is not shared due to either privacy or intellectual property concerns. In this case, the master node can communicate with a single "worker" node (e.g., on a Kubernetes cluster), which in turn can aggregate (e.g., using Spark or Ray) the gradient information from multiple worker nodes in the cluster in the premises. We are interested in compressing this aggregated gradient from the premises and sending it over the network to a different site.

Acknowledgements

This work was partially supported by the Wallenberg Artificial Intelligence, Autonomous Systems and Software Program (WASP) funded by Knut and Alice Wallenberg Foundation. This work was also supported in parts by the Swedish Research Council (Vetenskapsrådet) under grant 2020-03607.

References

Alistarh, D.; Grubic, D.; Li, J.; Tomioka, R.; and Vojnovic, M. 2017. QSGD: Communication-efficient SGD via gradient quantization and encoding. In *Advances in Neural Information Processing Systems*, volume 30, 1709–1720.

Alistarh, D.; Hoefler, T.; Johansson, M.; Konstantinov, N.; Khirirat, S.; and Renggli, C. 2018. The convergence of sparsified gradient methods. In *Advances in Neural Information Processing Systems*, volume 31, 5973–5983.

Aytekin, A.; Biel, M.; and Johansson, M. 2018. POLO: a POLicy-based optimization library. *arXiv preprint arXiv:1810.03417* .

Bertsekas, D. P. 1999. *Nonlinear programming: 2nd Edition*. Athena Scientific. ISBN 1-886529-00-0.

Feyzmahdavian, H. R.; Aytekin, A.; and Johansson, M. 2016. An asynchronous mini-batch algorithm for regularized stochastic optimization. *IEEE Transactions on Automatic Control* 61(12): 3740–3754.

Khaled, A.; Mishchenko, K.; and Richtárik, P. 2019. First analysis of local gd on heterogeneous data. *arXiv preprint arXiv:1909.04715* .

Khirirat, S.; Feyzmahdavian, H. R.; and Johansson, M. 2018. Distributed learning with compressed gradients. *arXiv preprint arXiv:1806.06573* .

Khirirat, S.; Johansson, M.; and Alistarh, D. 2018. Gradient compression for communication-limited convex optimization. In *2018 IEEE Conference on Decision and Control (CDC)*, 166–171.

Kozłowski, A.; and Sosnowski, J. 2017. Analysing efficiency of IPv6 packet transmission over 6LoWPAN network. In Romaniuk, R. S.; and Linczuk, M., eds., *Photonics Applications in Astronomy, Communications, Industry, and High Energy Physics Experiments 2017*, 456 – 467. SPIE.

Li, T.; Sahu, A. K.; Zaheer, M.; Sanjabi, M.; Talwalkar, A.; and Smith, V. 2018. Federated optimization in heterogeneous networks. *arXiv preprint arXiv:1812.06127* .

Li, X.; Huang, K.; Yang, W.; Wang, S.; and Zhang, Z. 2020. On the convergence of FedAvg on Non-IID data. In *International Conference on Learning Representations*.

Lian, X.; Huang, Y.; Li, Y.; and Liu, J. 2015. Asynchronous parallel stochastic gradient for nonconvex optimization. In *Advances in Neural Information Processing Systems*, volume 28, 2737–2745.

Magnússon, S.; Enyioha, C.; Li, N.; Fischione, C.; and Tarokh, V. 2017. Convergence of limited communication gradient methods. *IEEE Transactions on Automatic Control* 63(5): 1356–1371.

Magnússon, S.; Shokri-Ghadikolaei, H.; and Li, N. 2020. On maintaining linear convergence of distributed learning and optimization under limited communication. *IEEE Transactions on Signal Processing* 68: 6101–6116.

Nesterov, Y. 2018. *Lectures on convex optimization*, volume 137. Springer.

Rabbat, M. G.; and Nowak, R. D. 2005. Quantized incremental algorithms for distributed optimization. *IEEE Journal on Selected Areas in Communications* 23(4): 798–808.

Shi, S.; Wang, Q.; Zhao, K.; Tang, Z.; Wang, Y.; Huang, X.; and Chu, X. 2019a. A distributed synchronous SGD algorithm with global Top-k sparsification for low bandwidth networks. In *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*, 2238–2247.

Shi, S.; Zhao, K.; Wang, Q.; Tang, Z.; and Chu, X. 2019b. A convergence analysis of distributed SGD with communication-efficient gradient sparsification. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, 3411–3417.

Stich, S. U. 2019. Local SGD converges fast and communicates little. In *International Conference on Learning Representations*.

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

Tsitsiklis, J. N.; and Luo, Z.-Q. 1987. Communication complexity of convex optimization. *Journal of Complexity* 3(3): 231–243.

Vogels, T.; Karimireddy, S. P.; and Jaggi, M. 2020. Practical low-rank communication compression in decentralized deep learning. *Advances in Neural Information Processing Systems* 33: 14171–14181.

Wang, H.; Sievert, S.; Liu, S.; Charles, Z.; Papailiopoulos, D.; and Wright, S. 2018. ATOMO: Communication-efficient learning via atomic sparsification. In *Advances in Neural Information Processing Systems*, volume 31, 9850–9861.

Wang, J.; and Joshi, G. 2018. Cooperative SGD: A unified framework for the design and analysis of communication-efficient SGD algorithms. *CoRR* abs/1808.07576.

Wang, J.; and Joshi, G. 2019. Adaptive communication strategies to achieve the best error-runtime trade-off in local-update SGD. In Talwalkar, A.; Smith, V.; and Zaharia, M., eds., *Proceedings of Machine Learning and Systems*, volume 1, 212–229.

Wangni, J.; Wang, J.; Liu, J.; and Zhang, T. 2018. Gradient sparsification for communication-efficient distributed optimization. In *Advances in Neural Information Processing Systems*, volume 31, 1299–1309.

Wen, W.; Xu, C.; Yan, F.; Wu, C.; Wang, Y.; Chen, Y.; and Li, H. 2017. Terngrad: Ternary gradients to reduce communication in distributed deep learning. In *Advances in neural information processing systems*, volume 30, 1509–1519.

Yu, H.; Yang, S.; and Zhu, S. 2019. Parallel restarted SGD with faster convergence and less communication: Demystifying why model averaging works for deep learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, 5693–5700.

Zhu, C.; Han, S.; Mao, H.; and Dally, W. J. 2016. Trained ternary quantization. *arXiv preprint arXiv:1612.01064* .