A Communication Efficient Gradient Compression Algorithm for Distributed Learning

Tian Ye * Peijun Xiao † Ruoyu Sun ‡

May 13, 2020

Abstract

High communication complexity is a significant bottleneck in many distributed optimization problems. Recently, gradient quantization methods have received much attention in machine learning area. Nevertheless, the theoretical understanding of this class of methods is still limited. For instance, existing works often only characterize the number of bits per iteration, but not the total number of bits needed to achieve a certain accuracy.

In this paper, we propose Accelerated Increasing Accuracy and Double Encoding (AIADE), an algorithm which combines Nesterov acceleration and double encoding on the gradient differences. Our algorithm uses a new quantization scheme to ensure decreasing error which makes our algorithm different from other works that have constant quantization error in every update. AIADE is proved to have accelerated linear convergence rate for strongly convex problems, which is not achieved by existing works to our knowledge. We also compute the total number of bits of AIADE to achieve a certain accuracy in terms of the quantization level, which is missing in most existing works. We further extend our algorithm to limited memory systems and show that a combination of SGD and the proposed algorithm converges at a linear rate under Weak Growth Condition, thus leading to a small communication complexity.

1 Introduction

There is a surge of interest in distributed learning for large-scale computation in recent decade [1, 2, 3, 4, 5, 6, 7, 8]. In the past few years, new application scenarios such as multi-GPU computation [9, 10, 11, 12, 13], mobile edge computing [14, 15] and federated learning [16, 17] have received much attention. These systems are often bandwidth limited, and one of the most important questions in distributed learning is how to reduce the communication complexity.

A natural method to reduce the communication complexity is to compress the gradients transmitted between the machines. A host of recent works proposed to quantize the gradients and transfer the quantized gradients to save the communication cost [9, 10, 11, 12, 13]. These gradient quantization methods are successfully applied on training large-scale problems, and are shown to achieve similar performance to the original methods using less training time [12, 13].

*Institute for Interdisciplinary Information Sciences, Tsinghua University, Beijing, Beijing (yet17@mails.tsinghua.edu.cn). The work is performed when Tian Ye is visiting UIUC.

†Coordinated Science Laboratory, Department of ISE, University of Illinois at Urbana-Champaign, Urbana, IL (peijux2@illinois.edu).

‡Coordinated Science Laboratory, Department of ISE, University of Illinois at Urbana-Champaign, Urbana, IL (ruoyus@illinois.edu).

Despite the great popularity of gradient quantization methods, the theoretical understanding of them is still limited. Several popular algorithms, such as QSGD and TernGrad, are only shown to converge sub-linearly [12, 13]. QSVRG [12] which combines QSGD and variance reduction is shown to converge linearly, but each node in QSVRG has to broadcast its full gradient at the beginning of every epoch, which costs $32d$ bits (d is the dimension) and is quite time-consuming in practice. Another linearly convergent algorithm is DQGD [18] which quantizes the difference of gradients between two consecutive updates instead of quantizing the gradient in one single update. The convergent rate of DQGD is $\tilde{O}(\kappa)$ where κ is the condition number (definition 1.3) of the problem. However, the the total number of bits is not explicitly described.

1.1 Summary of Contributions

In this paper, we consider a star-network where there are N computing nodes and 1 central node. Suppose we want to minimize a function $f : \mathbb{R}^n \to \mathbb{R}$ decomposed as

$$f(x) = \frac{1}{N} \sum_{i=1}^{N} f_i(x) \tag{1}$$

where each function f_i is held on the i-th machine (or computing node), $i = 1, \ldots, N$.

We first consider the large-memory setting, where each local server f_i has enough memory to hold its data and use them to compute the full-batch gradient of f_i. Our contribution is two-fold. First, we improve the linear convergence rate from $\tilde{O}(\kappa)$ to $\tilde{O}(\sqrt{\kappa})$ by combining Nesterov acceleration and a novel quantization scheme with increasing accuracy. Second, we provide an explicit bound of the total number of bits (communication complexity) and show that the communication complexity of our algorithm is better than the the works mentioned above. We name our algorithm as Accelerated Increasing Accuracy and Double Encoding (AIADE). AIADE performs double encoding on gradient differences, i.e. we first quantize the gradient difference on the computing nodes and then quantize the gradient difference of the averaged gradients between two consecutive updates at the central node.

We also consider the limited memory setting (e.g. only one GPU) that each server is only able to compute the stochastic gradients since the data cannot be fit into one server. We extend our IADE algorithm to Increasing Accuracy and Double Encoding for Stochastic Gradient Descent (IADE-SGD) algorithm 3. A well-known classical result stated that SGD with constant stepsize can fail to converge even for minimizing a simple convex function [19], and SGD with diminishing stepsizes can converge. An interesting recent discovery is that in the interpolation regime (i.e., the learner can fit the data) it is possible to prove convergence of SGD with constant stepsize [20]. More formally, with the weak growth condition (WGC), [21] showed that SGD with constant stepsize can achieve a linear convergence rate. We adopt this WGC assumption and show that IADE-SGD also achieves a linear convergence rate in distributed setting. Note that it is not trivial to combine a quantized algorithm with SGD; for instance, it is not straightforward to see whether the existing results of DQGD can be applied to the limited-memory setting. Our framework is rather flexible and can be adapted to the SGD setting to achieve a similar convergence rate to the non-quantized version of SGD.

We summarize our main contributions as follows.

- In the first half of the paper, we consider the large-memory star-network setting where all data can be loaded into the memory of the system. To our knowledge, our algorithm **AIADE**(Accelerated Increasing Accuracy and Double Encoding) is the first gradient-quantization algorithm with rate $\tilde{O}(\sqrt{L/\mu})$ for minimizing a μ-strongly convex and L-smooth function in a star-network.

Algorithm	Convergence rate	Number of bits per communication
IADE	$\tilde{O}(\kappa)$	$O(d \log L)$
AIADE	$\tilde{O}(\sqrt{\kappa})$	$O(d \log L)$

Table 1: Summary of our theoretical results in solving problem 1 in large-memory setting. We assume f is L-smooth and μ-strongly-convex with condition number $\kappa := \frac{L}{\mu}$, and d is the dimension of the problem.

- Different from other recently proposed quantization algorithms, we quantize the vector with increasing accuracy according to the number of iterations. We also prove that this quantization scheme only costs a constant number of bits per iteration.
- To our knowledge, our paper is the first one to provide an explicit bound on the number of bits to achieve a certain accuracy in a star-network. In addition, we characterize how the quantization level affects the trade-off between the communication complexity and the convergence speed.
- In the second half of the paper, we consider the limited-memory distributed system. We adopt the Weak Growth Condition from [21] and prove that our algorithm IADE-SGD converges to the optimal solution at a linear rate.

1.2 Related Work

Two-node Case. A classical work [22] studied convex minimization of the problem (1) with two computing nodes. The authors of [22] provided a lower bound on the communication cost, and proposed a near optimal algorithm using quantized gradient differences. However, this work is restricted to the two-node case, and does not consider the limited-memory system.

Gradient Quantization Method. One of the most popular gradient quantization algorithms is called QSGD [12]. QSGD contains two parts. In the first part, QSGD performs unbiased quantization of the stochastic gradient v, and in the second part, QSGD compresses the quantized vector v to integers. More specifically, for quantization level $s \in \mathbb{Z}_+$ and vector v, QSGD uses 32 bits to encode the norm of v denoted as $\|v\|$ and use $\left\{-1, -\frac{s-1}{s}, \cdots, 1\right\}$ to encode each coordinate of unit vector $\frac{v}{\|v\|}$. In the second part, QSGD uses Elias method to encode each nonzero coordinate and its position respectively. This could reduce a large number of bits due to the sparsity of unit vector in high dimension. TernGrad [13] is another gradient quantization method which uses three fixed quantization levels $\{-1, 0, 1\}$. These algorithms are different from our proposed algorithms, since we use diminishing error quantization scheme rather than schemes that have constant errors.

Gradient Sparsification. Another popular choice to save communication complexity is gradient sparsification. [23] proposes a method to heuristically truncate the smallest gradients and transmit only the remaining large ones in communication. A few following work such as [24, 25] focus on providing theoretical understanding on gradient sparficiation methods and provide the convergence guarantee of the algorithms. The technique used in our proposed algorithm is orthogonal to gradient sparsification.

1.3 Definition and Outline

In this subsection, we first provide a few definitions which will be used throughout this paper.

Definition 1.1. *A differentiable function $h : \mathbb{R}^d \to \mathbb{R}$ is L-smooth if $\forall x, y \in \mathbb{R}^d$,*

$$|h(y) - h(x) - \langle y - x, \nabla h(x)\rangle| \leq \frac{L}{2}\|x - y\|^2 \tag{2}$$

Definition 1.2. *A differentiable function $f : \mathbb{R}^d \to \mathbb{R}$ is μ-strongly-convex if $\forall x, y \in \mathbb{R}^d$,*

$$f(y) - f(x) - \langle y - x, \nabla f(x)\rangle \geq \frac{\mu}{2}\|x - y\|^2 \tag{3}$$

Definition 1.3. *Suppose $f : \mathbb{R}^d \to \mathbb{R}, f(x) := \frac{1}{N}\sum_{i=1}^{N} f_i(x)$, where f_i is L-smooth $\forall i \in [N]$ and f is μ-strongly-convex. Then the condition number κ of this collection of functions is defined as $\kappa = \frac{L}{\mu}$.*

Assumption 1.4. *Assume that in (1) f is μ-strongly-convex and f_i is L-smooth $\forall i \in [N]$.*

The outline of the paper is as follows. In section 2, we introduce the most elementary gadgets of quantization, i.e. the quantization algorithm of a single vector. In section 3, we provide the quantization version of GD. In section 4, we introduce the quantization of Nesterov acceleration. In section 5, we introduce the quantization version of SGD. All the proposed algorithms converge linearly while using only small number of bits. The proofs of the lemmas and theorems in are shown in the appendix.

2 Quantization of a vector

A deterministic quantization algorithm of vectors contains two steps: encoding and decoding. We view the encoding step as a function $\mathbf{E} : S \to \mathbb{Z}^{\geq 0}$ and the decoding step as a function $\mathbf{D} : \mathbb{Z}^{\geq 0} \to \mathbb{R}^d$ where $S \subseteq \mathbb{R}^d$ is the set of vectors we need to do quantization. To bound the absolute error, it requires $\|\mathbf{D}(\mathbf{E}(x)) - x\|_2 \leq \sigma$. In this case, the number of bits required is $\lceil \log_2 |\mathbf{E}(S)| \rceil$. We call this kind of quantization **uniform compression**.

2.1 Uniform compression

For uniform compression, we propose the lower bound of number of bits given the precision σ and an algorithm matches this lower bound.

Lemma 2.1. *For any uniform compression algorithm and $S = \left\{ x \in \mathbb{R}^d \middle| \|x\|_2 \leq M \right\}$, define $\varepsilon = \frac{\sigma}{M}$. Then the minimal number of bits required is $\left\lceil d \log_2 \frac{1}{\varepsilon} \right\rceil$. In addition, there exists an algorithm that takes only $\left\lceil 1.05d + d \cdot \log_2 \frac{1+2\varepsilon}{\varepsilon} \right\rceil$ bits.*

3 Quantization in gradient descent

Consider the following function $f(x) := \frac{1}{N}\sum_i^N f_i(x)$ where each $f_i : \mathbb{R}^d \to \mathbb{R}$ is L-smooth and μ-strongly convex. We assume there are N machines, and the i-th machine can hold data needed to compute a function $f_i(x)$.

Without considering the communication, one could aggregate $\nabla f_i(x)$ in the central node and obtain $\nabla f(x)$, performing a gradient descent step at the central node to update the iterate, and then broadcasts the new iterate to each node. This is just the batch gradient descent. However, due to the bandwidth restriction

between the nodes, exact aggregation of all gradients is not possible. We would like to use less communication to find ε-optimal point.

One trivial algorithm is the following. Initially each party holds the same weight w_0. In each round, each party i computes gradient g_i individually, computes a quantized version of g_i and sends to the central node. The central node aggregates the quantized gradients to obtain an approximation of the gradient, updates the iterate by this approximate gradient, and then broadcasts the new iterate to each node.

However, there are two drawbacks of this type of algorithms. First of all, because the optimal point of f is not necessary close to the optimal point of f_i, the vector they need to do quantization could be always large. Hence they either costs large number of bits or cause large error. Besides, as the error is approximately constant level, it could only converge sub-linearly.

In our algorithm algorithm 1, we use diminishing error in quantization. Here $Q(\cdot, \varepsilon)$ means encoding a vector with maximal absolute error ε.

Algorithm 1: Increasing Accuracy and Double Encoding (IADE)

Initialization: Each server $i \in [N]$ holds $x_0 = s_{-1}^i = v_{-1} = 0$, server 0 holds $v_{-1} = 0$, $k = 0$;
$\eta \in \left(0, \frac{2}{L+\mu}\right)$, $c = 1 - \eta\mu$, $c < c' < 1$ and s is the quantization level;
while *the precision is not enough* **do**
  for $i \in [N]$ **do**
    server i computes $g_k^i = \nabla f_i(x_k)$;
    server i does quantization $d_k^i = Q(g_k^i - s_{k-1}^i, \frac{sc'^{k+1}}{2})$;
    server i updates $s_k^i = d_k^i + s_{k-1}^i$;
    server i send d_k^i to server 0;
  end
  server 0 computes $s_k = \frac{1}{N}\sum_{i=1}^{N} d_k^i + s_{k-1}$;
  server 0 does quantization $u_k = Q(s_k - v_{k-1}, \frac{sc'^{k+1}}{2})$;
  server 0 sends u_k to server $i, \forall i \in [N]$;
  server 0 updates $v_k = u_k + v_{k-1}$;
  for $i \in [N]$ **do**
    server i updates $v_k = u_k + v_{k-1}$;
    server i updates $x_{k+1} = x_k - \eta v_k$;
  end
  $k = k + 1$;
end

We have the following theorem on the convergence rate of the SGD variant of our algorithm IADE-SGD.

Theorem 3.1. *Algorithm 1 only uses $O(Nd)$ bits at iteration $k \geq 1$, and*

$$\|x_k - x^*\| \leq (c')^k \left(\max\left\{0, \|x_0 - x^*\| - \frac{cs}{c' - c}\right\} + \frac{c's}{c' - c}\right).$$

Proof sketch. First we can prove $\|v_k - \nabla f(x_k)\| \leq sc'^{k+1}$ by triangle inequality. By diminishing error, we can prove linear convergence of series $\{\|x_k - x^*\|\}_{k\geq 0}$. Then the vector we need to quantize is close to $\nabla f_i(x_{k+1}) - f_i(x_k)$ which can be bounded by $L\|x_{k+1} - x_k\|$. Because $\{\|x_k - x^*\|\}$ converges linearly, $\|\nabla f_i(x_{k+1}) - f_i(x_k)\|$ converges at the same rate. By lemma 2.1 we know that it costs only constant bits in each quantization. □

4 Quantization of Nesterov acceleration

The accelerating algorithm is very similar to algorithm 1.

Algorithm 2: Accelerated Increasing Accuracy and Double Encoding (AIADE)

Initialization: Each server $i \in [N]$ holds $x_0 = s_{-1}^i = v_{-1} = 0$, server 0 holds $v_{-1} = 0$ $k = 0$;

Parameter setting: $\eta = \frac{1}{L}$, $\tau = \frac{\sqrt{L}-\sqrt{\mu}}{\sqrt{L}+\sqrt{\mu}}$, $c = \sqrt{1-\sqrt{\frac{\mu}{L}}}$;

while *the precision is not enough* **do**

  for $i \in [N]$ **do**

    server i computes $g_k^i = \nabla f_i(y_k)$;

    server i does quantization $d_k^i = Q(g_k^i - s_{k-1}^i, \frac{\varepsilon c^{k+1}}{2k+2})$;

    server i updates $s_k^i = d_k^i + s_{k-1}^i$;

    server i send d_k^i to server 0;

  end

  server 0 computes $s_k = \frac{1}{N}\sum_{i=1}^{N} d_k^i + s_{k-1}$;

  server 0 does quantization $u_k = Q(s_k - v_{k-1}, \frac{\varepsilon c^{k+1}}{2k+2})$;

  server 0 sends u_k to server $i, \forall i \in [N]$;

  server 0 updates $v_k = u_k + v_{k-1}$;

  for $i \in [N]$ **do**

    server i updates $v_k = u_k + v_{k-1}$;

    server i updates $x_{k+1} = y_k - \eta v_k$;

    server i updates $y_{k+1} = x_{k+1} + \tau(x_{k+1} - x_k)$;

  end

  $k = k + 1$;

end

We have the following theorem.

Theorem 4.1. *Algorithm 2 has the following two properties:*

(1) $\|x_k - x^*\| \le c^k \cdot G(k)$ *where* $G(k) = O(\ln k)$.

(2) The number of bits at iteration $k \ge 1$ *is* $O\left(Nd\ln(k\ln k)\right)$

Proof sketch. The proof sketch is similar to theorem 1. First of all, we can prove that $\|v_k - \nabla f(y_k)\|$ is diminishing, which ensures linear convergence of $\{\|x_k - x^*\|\}_{k\ge 0}$. Hence the vector we need to quantize is also diminishing, which implies constant number of bits. The only difference is that we need more efforts to prove linear convergence under such error, i.e. we need to use mathematical induction carefully to bound the error in an appropriate interval. □

5 Quantization Methods in Limited Memory Systems

In the previous sections, we consider the large-memory setting where there are enough machines to hold all the data samples in the memory. For instance, if the total size of the data set is 1000 G and we have 100 machines each with memory 10 G, then the above method can be applied.

In this section, we consider the limited-memory setting. In many other application scenarios, we may not have enough machines to hold all data samples. For instance, if the total size of the data set is 1000 G and we only have 10 machines each with memory 10 G, then we can only load a mini-batch of the dataset into the 10 machines in one iteration. This necessitates the usage of SGD.

We distinguish the two setting specifically for the following reason. It is impossible to prove linear convergence of quantized SGD in the limited-memory setting without further assumptions, since even with infinite bandwidth SGD cannot achieve linear convergence rate [19]. Therefore, to achieve linear convergence of quantized SGD-type methods, it is necessary to study a setting that a SGD-type method with constant stepsize can achieve linear convergence rate.

There are two lines of research that can prove linear convergence of SGD-type methods. Along the first line, a few variance-reduction based methods such as SVRG [26] and SAGA [27] can achieve linear convergence. Along the second line, with extra assumption such as **WGC(Weak Growth Condition)** assumption, the vanilla SGD with constant stepsize can already achieve linear convergence [21]. This line of research is strongly motivated by the interpolation assumption in machine learning that the learner can fit the data, which is considered a reasonable assumption in recent literature. Therefore, we focus on designing quantization algorithms along the second line.

The WGC assumption is formally defined as follows.

Assumption 5.1. *[21] Suppose* $f : \mathbb{R}^d \to \mathbb{R}, f(x) = \frac{1}{N}\sum_{i=1}^{N} f_i(x)$ *is the objective function. Stochastic "functions"(algorithms)* $\{\overline{\nabla}_i\}_{i\in[N]}$ *satisfy* ***WGC*** *if*

- $\mathbb{E}[\overline{\nabla}_i(f_i, x)] = \nabla f_i(x)$, $\forall i \in [N], x \in \mathbb{R}^d$.
- $\frac{1}{N}\sum_{i=1}^{N} \mathbb{E}_{\overline{\nabla}_i}\left[\|\overline{\nabla}_i f_i(x)\|^2\right] \le 2\rho L(f(x) - f(x^*))$.

With the assumption above, we can prove the convergence of the algorithm 3 below. Here $Q(\cdot, \varepsilon)$ is an unbiased random quantization algorithm whose error's square is bounded by ε.

5.1 Theoretical analysis

Theorem 5.2. *Algorithm 3 uses* $O(Nd)$ *bits at iteration* $k \ge 1$ *in expectation, and*

$$\mathbb{E}\left[\|x_k - x^*\|^2\right] \le {c'}^k \left(\max\left\{0, \|x_0 - x^*\|^2 - \frac{cs}{c' - c}\right\} + \frac{c's}{c' - c}\right),$$

where $c = 1 - \frac{\mu}{\rho L}$ *and* $c < c' < 1$.

Proof sketch. Similarly, we can prove that $\|v_k - \frac{1}{N}\sum_{i=1}^{N} \overline{\nabla}_i f_i(x)\| \le \rho^2 L^2 s {c'}^{k+1}$. Then, fixing x_k and we have

$$\mathbb{E}[\|x_{k+1} - x^*\|^2] \le \|x_k - x^*\|^2 - 2\eta\mathbb{E}[\nabla f(x_k), x_k - x^*] + eta^2 \cdot 2\rho L(f(x_k) - f^*) + \eta^2\rho^2 L^2 s {c'}^{k+1}.$$

Taking $\eta = \frac{1}{\rho L}$ and we have

$$\mathbb{E}[\|x_{k+1} - x^*\|^2] \le c\|x_k - x^*\|^2 + s{c'}^{k+1}.$$

Finally, we can use mathematical induction to prove the convergence result above. Because $b(x) := \ln x^2 = 2\ln x$ is concave, we know the number of bits is $O(Nd)$ in expectation. □

Algorithm 3: Increasing Accuracy and Double Encoding for Stochastic Gradient Descent (IADE-SGD)

Initialization: Each server $i \in [N]$ holds $x_0 = s_{-1}^i = v_{-1} = 0$, server 0 holds $v_{-1} = 0$, $k = 0$;
$\eta = \frac{1}{\rho L}$, $c = 1 - \frac{\mu}{\rho L}$, $c < c' < 1$ and s is the quantization level;
while *the precision is not enough* **do**
  for $i \in [N]$ **do**
    server i computes $g_k^i = \overline{\nabla}_i f_i(x_k)$;
    server i does quantization $d_k^i = Q(g_k^i - s_{k-1}^i, \frac{\rho^2 L^2 s c'^{k+1}}{2})$;
    server i updates $s_k^i = d_k^i + s_{k-1}^i$;
    server i send d_k^i to server 0;
  end
  server 0 computes $s_k = \frac{1}{N}\sum_{i=1}^{N} d_k^i + s_{k-1}$;
  server 0 does quantization $u_k = Q(s_k - v_{k-1}, \frac{\rho^2 L^2 s c'^{k+1}}{2})$;
  server 0 sends u_k to server $i, \forall i \in [N]$;
  server 0 updates $v_k = u_k + v_{k-1}$;
  for $i \in [N]$ **do**
    server i updates $v_k = u_k + v_{k-1}$;
    server i updates $x_{k+1} = x_k - \eta v_k$;
  end
  $k = k + 1$;
end

6 Conclusion

We have presented AIADE, an accelerated algorithm to compress the gradient difference on both the slave nodes and the master node to achieve efficient communication for distributed learning. We proved that AIADE achieves an accelerated linear convergence rate, and provided the best bound so far on the total number of bits to achieve a certain accuracy. We also presented a SGD version of the algorithm in the limited-memory setting, and showed its linear convergence rate with constant stepsize under Weak Growth Condition.

References

[1] B. Recht, C. Re, S. Wright, and F. Niu, "Hogwild: A lock-free approach to parallelizing stochastic gradient descent," in *Advances in neural information processing systems*, pp. 693–701, 2011.

[2] J. Dean, G. Corrado, R. Monga, K. Chen, M. Devin, M. Mao, M. Ranzato, A. Senior, P. Tucker, K. Yang, *et al.*, "Large scale distributed deep networks," in *Advances in neural information processing systems*, pp. 1223–1231, 2012.

[3] A. Coates, B. Huval, T. Wang, D. Wu, B. Catanzaro, and N. Andrew, "Deep learning with cots hpc systems," in *International conference on machine learning*, pp. 1337–1345, 2013.

[4] M. Li, D. G. Andersen, J. W. Park, A. J. Smola, A. Ahmed, V. Josifovski, J. Long, E. J. Shekita, and B.-Y. Su, "Scaling distributed machine learning with the parameter server," in *11th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 14)*, pp. 583–598, 2014.

[5] T. Chilimbi, Y. Suzue, J. Apacible, and K. Kalyanaraman, "Project adam: Building an efficient and scalable deep learning training system," in *11th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 14)*, pp. 571–582, 2014.

[6] E. P. Xing, Q. Ho, W. Dai, J. K. Kim, J. Wei, S. Lee, X. Zheng, P. Xie, A. Kumar, and Y. Yu, "Petuum: A new platform for distributed machine learning on big data," *IEEE Transactions on Big Data*, vol. 1, no. 2, pp. 49–67, 2015.

[7] T. Chen, M. Li, Y. Li, M. Lin, N. Wang, M. Wang, T. Xiao, B. Xu, C. Zhang, and Z. Zhang, "Mxnet: A flexible and efficient machine learning library for heterogeneous distributed systems," *arXiv preprint arXiv:1512.01274*, 2015.

[8] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, *et al.*, "Tensorflow: Large-scale machine learning on heterogeneous distributed systems," *arXiv preprint arXiv:1603.04467*, 2016.

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

[10] F. Seide, H. Fu, J. Droppo, G. Li, and D. Yu, "1-bit stochastic gradient descent and its application to data-parallel distributed training of speech dnns," in *Fifteenth Annual Conference of the International Speech Communication Association*, 2014.

[11] S. Gupta, A. Agrawal, K. Gopalakrishnan, and P. Narayanan, "Deep learning with limited numerical precision," in *International Conference on Machine Learning*, pp. 1737–1746, 2015.

[12] D. Alistarh, D. Grubic, J. Li, R. Tomioka, and M. Vojnovic, "Qsgd: Communication-efficient sgd via gradient quantization and encoding," in *Advances in Neural Information Processing Systems*, pp. 1709–1720, 2017.

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

[14] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, *et al.*, "Communication-efficient learning of deep networks from decentralized data," *arXiv preprint arXiv:1602.05629*, 2016.

[15] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: The communication perspective," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 4, pp. 2322–2358, 2017.

[16] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," *arXiv preprint arXiv:1908.07873*, 2019.

[17] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated machine learning: Concept and applications," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 10, no. 2, pp. 1–19, 2019.

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

[19] Z.-Q. Luo, "On the convergence of the lms algorithm with adaptive learning rate for linear feedforward networks," *Neural Computation*, vol. 3, no. 2, pp. 226–245, 1991.

[20] M. Schmidt and N. L. Roux, "Fast convergence of stochastic gradient descent under a strong growth condition," *arXiv preprint arXiv:1308.6370*, 2013.

[21] S. Vaswani, F. Bach, and M. Schmidt, "Fast and faster convergence of sgd for over-parameterized models and an accelerated perceptron," *arXiv preprint arXiv:1810.07288*, 2018.

[22] J. N. Tsitsiklis and Z.-Q. Luo, "Communication complexity of convex optimization," *Journal of Complexity*, vol. 3, no. 3, pp. 231–243, 1987.

[23] A. F. Aji and K. Heafield, "Sparse communication for distributed gradient descent," *arXiv preprint arXiv:1704.05021*, 2017.

[24] J. Wangni, J. Wang, J. Liu, and T. Zhang, "Gradient sparsification for communication-efficient distributed optimization," in *Advances in Neural Information Processing Systems*, pp. 1299–1309, 2018.

[25] D. Alistarh, T. Hoefler, M. Johansson, N. Konstantinov, S. Khirirat, and C. Renggli, "The convergence of sparsified gradient methods," in *Advances in Neural Information Processing Systems*, pp. 5973–5983, 2018.

[26] R. Johnson and T. Zhang, "Accelerating stochastic gradient descent using predictive variance reduction," in *Advances in neural information processing systems*, pp. 315–323, 2013.

[27] A. Defazio, F. Bach, and S. Lacoste-Julien, "Saga: A fast incremental gradient method with support for non-strongly convex composite objectives," in *Advances in neural information processing systems*, pp. 1646–1654, 2014.

7 Appendix A

Lemma 7.1. *Suppose $c < c' < 1$ and $x_i > 0, \forall i \geq 0$, $s > 0$. Then series $\{x_i\}_{i\geq 0}$ with iteration formula*

$$x_{k+1} \leq c \cdot x_k + s \cdot {c'}^{k+1} \tag{4}$$

satisfy $x_k \leq {c'}^k \xi_s$ where $\xi_s := \max\left\{0, x_0 - \frac{cs}{c'-c}\right\} + \frac{c's}{c'-c}$.

Proof. According to definition, we have

$$\frac{x_{k+1}}{c^{k+1}} \leq \frac{x_k}{c^k} + s \cdot \gamma^{k+1}$$

where $\gamma := \frac{c'}{c} > 1$. Taking summation from 0 to k we have

$$\begin{aligned} \frac{x_{k+1}}{c^{k+1}} &\leq x_0 + \sum_{i=1}^{k+1} s \cdot \gamma^i \\ &= x_0 + s\frac{\gamma^{k+2} - \gamma}{\gamma - 1}. \end{aligned}$$

Finally, we have

$$x_{k+1} \leq c^{k+1}\left(x_0 - \frac{s\gamma}{\gamma - 1}\right) + {c'}^{k+1}\frac{s\gamma}{\gamma - 1}.$$

The proof follows by the fact that $c < c' < 1$. □

Lemma 7.2. *Suppose $f : \mathbb{R}^d \to \mathbb{R}$ is L-smooth and μ-strongly-convex, then $\forall \eta \in \left(0, \frac{2}{L+\mu}\right]$, and $x_{k+1} = x_k - \eta\nabla f(x_k)$, we have*

$$\|x_{k+1} - x^*\| \leq c_\eta \|x_k - x^*\|$$

where x^* is the unique minimal point of f and $c_\eta := 1 - \eta\mu$.

Proof. Consider the function $F : \mathbb{R}^d \to \mathbb{R}^d$, $F(x) = x - \eta\nabla f(x)$. Because $\nabla F(x) = I - \eta\nabla^2 f(x)$, we know F is c_η-Lip. Proof follows by the fact that x^* is the unique stable point of F. □

7.1 Proof of theorem 3.1

Proof. We first prove the second statement. By lemma 7.1 and lemma 7.2, we only need to prove $\|v_k - g_k\| \leq s{c'}^{k+1}$ where $g_k = \nabla f(x_k)$. By induction, we have $s_k - \frac{1}{N}\sum_{i=1}^N s_k^i = \frac{1}{N}\sum_{i=1}^N d_k^i + s_{k-1} - \frac{1}{N}\sum_{i=1}^N (d_k^i + s_{k-1}^i) = s_{k-1} - \frac{1}{N}\sum_{i=1}^N s_{k-1}^i = s_{-1} - \frac{1}{N}\sum_{i=1}^N s_{-1}^i = 0$. Then,

$$\begin{aligned} \|v_k - g_k\| &= \|u_k + v_{k-1} - g_k\| \\ &\leq \|u_k - (s_k - v_{k-1})\| + \|s_k - g_k\| \\ &\leq \|u_k - (s_k - v_{k-1})\| + \left\|\frac{1}{N}\sum_{i=1}^N (d_k^i + s_{k-1}^i - g_k^i)\right\| \end{aligned}$$

$$\begin{aligned} &\leq \|u_k - (s_k - v_{k-1})\| + \frac{1}{N}\sum_{i=1}^{N}\left\|d_k^i + s_{k-1}^i - g_k^i\right\| \\ &\leq sc'^{k+1}. \end{aligned}$$

For convenience, we define constant $\xi_s = \max\left\{0, \|x_0 - x^*\| - \frac{cs}{c'-c}\right\} + \frac{c's}{c'-c}$. The convergence result comes directly from lemma 7.1.

To bound the number of bits, we only need to calculate the maximal norm of the vector we need to encode. Actually, $\forall i \in [N]$, $\forall k \geq 1$ we have

$$\begin{aligned} \|g_k^i - s_{k-1}^i\| &\leq \|g_k^i - g_{k-1}^i\| + \left\|g_{k-1}^i - s_{k-1}^i\right\| \\ &\leq L\|x_k - x_{k-1}\| + \frac{sc'^k}{2} \\ &\leq L\eta\|v_{k-1}\| + \frac{sc'^k}{2} \\ &\leq L\eta\|g_{k-1}\| + L\eta sc'^k + \frac{sc'^k}{2} \\ &\leq L^2\eta\|x_{k-1} - x^*\| + L\eta sc'^k + \frac{sc'^k}{2} \\ &\leq L^2\eta c'^{k-1}\xi_s + L\eta sc'^k + \frac{sc'^k}{2} \\ &= \left(\frac{L^2\eta\xi_s + L\eta sc' + \frac{sc'}{2}}{c'^2}\right)c'^{k+1}. \end{aligned}$$

Similarly, we have

$$\begin{aligned} \|s_k - v_{k-1}\| &\leq \|s_k - g_k\| + \|g_k - g_{k-1}\| + \|g_{k-1} - v_{k-1}\| \\ &\leq L\|x_k - x_{k-1}\| + \frac{3sc'^k}{2} \\ &\leq \left(\frac{L^2\eta\xi_s + L\eta sc' + \frac{3sc'}{2}}{c'^2}\right)c'^{k+1}. \end{aligned}$$

In this case, the error fraction (length divided by error, i.e. inverse of relative error) is bounded by $\zeta_s := \frac{2L^2\eta\frac{\xi_s}{s} + 2L\eta c' + 3c'}{c'^2}$, and the number of bits is at most $(1.05 + \log_2(\zeta_s + 2))\,d$ by lemma 2.1. □

8 Appendix B

In this section, we prove theorem 4.1.

Proof. Choose an arbitrary point $x_0 = y_0 = v_0 \in \mathbb{R}^d$, we can define $\phi_0(x) := \phi_0^* + \frac{\mu}{2}\|x - v_0\|^2$ where $\phi_0^* := f(v_0)$. Then by definition we know $\phi(x) \leq f(x)$ and $\phi_0^* \geq f(x_0)$.

Next, we inductively define the following quantity. Suppose $\ell : \mathbb{Z}^{\geq 0} \to \mathbb{R}^+$ is an arbitrary function.

$$x_{k+1} = y_k - \frac{1}{L}m_k$$

$$\begin{aligned}
\phi_{k+1}(x) &= (1-\alpha)\phi_k(x) + \alpha\left[f(y_k) + \langle m_k, x - y_k\rangle + \frac{\mu}{2}\|x - y_k\|^2\right] \\
v_{k+1} &= \arg\min_{v\in\mathbb{R}^n} \phi_{k+1}(v) \\
y_{k+1} &= \frac{x_{k+1} + \alpha v_{k+1}}{1+\alpha} \\
\phi^*_{k+1} &= \min \phi_{k+1}
\end{aligned}$$

where $\alpha = \sqrt{\frac{\mu}{L}}$ and $m_k \in \mathbb{R}^d$ such that $\|m_k - \nabla f(y_k)\| \le c^{k+1}\ell(k)$.

Besides, we will construct monotonically increasing functions $h, g : \mathbb{Z}^{\ge 0} \to \mathbb{R}$ inductively such that

$$\begin{aligned}
f(x_k) &\le \phi^*_k + h(k)\cdot c^{2k} \\
\phi_k(x^*) &\le (1-c^{2k})f^* + c^{2k}\phi_0(x^*) + g(k)\cdot c^{2k}.
\end{aligned}$$

Obviously, it is appropriate to set $g(0) = h(0) = 0$.

Before we go deeper, here is an important lemma.

Lemma 8.1. *With the definition above, we have*

$$y_{k+1} = x_{k+1} + \tau(x_{k+1} - x_k)$$

where $\tau = \frac{\sqrt{L}-\sqrt{\mu}}{\sqrt{L}+\sqrt{\mu}}$.

Proof. Because $\phi_k(x) = \phi^*_k + \frac{\mu}{2}\|x - v_k\|^2$, taking derivative $\nabla\phi_{k+1}(x) = \mu(1-\alpha)(x - v_k) + \alpha m_k + \alpha\mu(x - y_k)$. And then we get $v_{k+1} = (1-\alpha)v_k + \alpha y_k - \frac{\alpha}{\mu}m_k$.

Because $y_k = \frac{x_k + \alpha v_k}{1+\alpha}$, we can substitute $v_k = \frac{1+\alpha}{\alpha}y_k - \frac{1}{\alpha}x_k$ and get

$$\begin{aligned}
v_{k+1} &= \frac{1-\alpha^2}{\alpha}y_k - \frac{1-\alpha}{\alpha}x_k + \alpha y_k - \frac{\alpha}{\mu}m_k \\
&= \frac{1}{\alpha}\left(y_k - \frac{1}{L}m_k\right) - \frac{1-\alpha}{\alpha}x_k \\
&= \frac{x_{k+1} - x_k}{\alpha} + x_k.
\end{aligned}$$

Hence

$$\begin{aligned}
y_{k+1} &= \frac{x_{k+1} + \alpha v_{k+1}}{1+\alpha} \\
&= x_{k+1} + \tau(x_{k+1} - x_k).
\end{aligned}$$

□

Then we have

$$\begin{aligned}
f(x_k) - f^* &\le \phi^*_k + h(k)\cdot c^{2k} - f^* \\
&\le c^{2k}(\phi_0(x^*) - f^* + g(k) + h(k)) \\
&= c^{2k}(\Delta + g(k) + h(k))
\end{aligned}$$

where $\Delta := \phi_0(x^*) - f^* = f(v_0) - f^* + \frac{\mu}{2}\|x_0 - x^*\|^2$. Hence,

$$\|x_k - x^*\| \le \sqrt{\frac{2}{\mu}(f(x_k - x^*))}$$

$$= \sqrt{\frac{2}{\mu}} \cdot c^k \cdot \sqrt{\Delta + g(k) + h(k)}.$$

Furthermore, according to lemma 3 and the monotony of g and h, we have

$$\begin{aligned}
\|y_k - x^*\| &= \left\| \frac{2\sqrt{L}}{\sqrt{L}+\sqrt{\mu}}(x_k - x^*) - \frac{\sqrt{L}-\sqrt{\mu}}{\sqrt{L}+\sqrt{\mu}}(x_{k-1} - x^*) \right\| \\
&\leq \left(\frac{2\sqrt{L}}{\sqrt{L}+\sqrt{\mu}} + \frac{1}{c}\frac{\sqrt{L}-\sqrt{\mu}}{\sqrt{L}+\sqrt{\mu}} \right) \sqrt{\frac{2}{\mu}} \cdot c^k \cdot \sqrt{\Delta + g(k) + h(k)} \\
&\leq 3\sqrt{\frac{2}{\mu}} \cdot c^k \cdot \sqrt{\Delta + g(k) + h(k)}.
\end{aligned}$$

And then we can give the first upper bound

$$\begin{aligned}
\phi_{k+1}(x^*) &= (1-\alpha)\phi_k(x^*) + \alpha \left[f(y_k) + \langle m_k, x^* - y_k \rangle + \frac{\mu}{2}\|x^* - y_k\|^2 \right] \\
&\leq (1-\alpha)\phi_k(x^*) + \alpha \left[f(y_k) + \langle \nabla f(y_k), x^* - y_k \rangle + \frac{\mu}{2}\|x^* - y_k\|^2 \right] \\
&\quad + \alpha \cdot \ell(k) c^{k+1} \|x^* - y_k\| \\
&\leq (1-\alpha)\phi_k(x^*) + \alpha f^* + \alpha \cdot \ell(k) c^{k+1} \|x^* - y_k\| \\
&\leq c^2 \left((1-c^{2k}) f^* + c^{2k}\phi_0(x^*) + g(k) \cdot c^{2k} \right) + (1-c^2) f^* \\
&\quad + \alpha \cdot \ell(k) c^{k+1} \|x^* - y_k\| \\
&= (1-c^{2k+2}) f^* + c^{2k+2}\phi_0(x^*) + g(k) \cdot c^{2k+2} \\
&\quad + 3\ell(k)\sqrt{\frac{2}{L}} \cdot c^{2k+1} \cdot \sqrt{\Delta + g(k) + h(k)},
\end{aligned}$$

which means we only need to make

$$g(k+1) \geq g(k) + \frac{3\ell(k)\sqrt{\frac{2}{L}} \cdot \sqrt{\Delta + g(k) + h(k)}}{c}$$

To make an upper bound of $h(k+1)$, we notice that

$$\begin{aligned}
\phi^*_{k+1} &= \phi_{k+1}(v_{k+1}) \\
&= (1-\alpha)\left(\phi^*_k + \frac{\mu}{2}\|v_{k+1} - v_k\|^2 \right) + \alpha f(y_k) + \alpha \langle m_k, v_{k+1} - y_k \rangle \\
&\quad + \frac{\alpha\mu}{2}\|v_{k+1} - y_k\|^2.
\end{aligned}$$

Substitute $v_{k+1} - y_k = (1-\alpha)(v_k - y_k) - \frac{\alpha}{\mu} m_k$, we have

$$\begin{aligned}
\phi^*_{k+1} &= (1-\alpha)\phi^*_k + \alpha f(y_k) - \frac{1}{2L}\|m_k\|^2 \\
&\quad + \alpha(1-\alpha)\left(\frac{\mu}{2}\|y_k - v_k\|^2 + \langle m_k, v_k - y_k \rangle \right) \\
&\geq (1-\alpha)\left(f(x_k) - h(k) \cdot c^{2k} \right) + \alpha f(y_k) - \frac{1}{2L}\|m_k\|^2 \\
&\quad + \alpha(1-\alpha)\left(\frac{\mu}{2}\|y_k - v_k\|^2 + \langle m_k, v_k - y_k \rangle \right).
\end{aligned}$$

Because $f(x_k) \geq f(y_k) + \langle \nabla f(y_k), x_k - y_k \rangle \geq f(y_k) + \langle m_k, x_k - y_k \rangle - \ell(k) \cdot c^{k+1} \|x_k - y_k\|$ and $\|x_k - y_k\| = \tau \|x_k - x_{k-1}\| \leq c^2 \left(\|x_k - x^*\| + \|x^* - x_{k-1}\| \right) \leq c^{k+1} \cdot (1+c) \sqrt{\frac{2}{\mu}} \sqrt{\Delta + g(k) + h(k)}$.

Hence, we can bound ϕ^*_{k+1} by

$$\begin{aligned} & f(y_k) - \frac{1}{2L} \|m_k\|^2 - h(k) \cdot c^{2k+2} \\ & - c^{2k+4} \cdot \ell(k) \cdot (1+c) \sqrt{\frac{2}{\mu}} \sqrt{\Delta + g(k) + h(k)}. \end{aligned}$$

Recall that

$$\begin{aligned} f(x_{k+1}) - f(y_k) + \frac{1}{2L} \|m_k\|^2 &\leq \frac{1}{L} \langle m_k - \nabla f(y_k), m_k \rangle \\ &\leq \frac{1}{L} \ell(k) \cdot c^{k+1} \|m_k\| \\ &\leq c^{2k+1} \cdot \left(\frac{c \ell^2(k)}{L} + 3 \sqrt{\frac{2}{\mu}} \ell(k) \sqrt{\Delta + g(k) + h(k)} \right) \end{aligned}$$

we have

$$\begin{aligned} \phi^*_{k+1} \geq\ & f(x_{k+1}) - c^{2k+1} \cdot \left(\frac{c \ell^2(k)}{L} + 3 \sqrt{\frac{2}{\mu}} \ell(k) \sqrt{\Delta + g(k) + h(k)} \right) \\ & - h(k) \cdot c^{2k+2} - c^{2k+4} \cdot \ell(k) \cdot (1+c) \sqrt{\frac{2}{\mu}} \sqrt{\Delta + g(k) + h(k)}, \end{aligned}$$

which means $h(\cdot)$ only need to satisfy

$$g(k+1) \geq g(k) + \frac{5}{c} \ell(k) \cdot \sqrt{\frac{2}{\mu}} \sqrt{\Delta + g(k) + h(k)} + \frac{\ell^2(k)}{L}.$$

Finally, it is appropriate to choose $\ell k = \frac{\varepsilon}{k+1}$ for any $\varepsilon > 0$, then solving ODE shows there exists constants $E^1_\varepsilon, E^2_\varepsilon, E^3_\varepsilon$ and E^4_ε such that

$$\begin{aligned} g(k) &\leq E^1_\varepsilon + E^2_\varepsilon \ln^2(k+1) \\ h(k) &\leq E^3_\varepsilon + E^4_\varepsilon \ln^2(k+1), \end{aligned}$$

and $\|x_k - x^*\| = O(c^k \cdot \ln k)$ follows. And the bound of number of bits follows by lemma 2.1 since we already have bound on $\|y_k - x^*\|$. □

9 Appendix C

In this section we deal with theorem 5.2.

Lemma 9.1. *Suppose $f : \mathbb{R}^d \to \mathbb{R}$ is μ-strongly-convex. Besides, $f(x) = \frac{1}{N} \sum_{i=1}^{N} f_i(x)$, where each f_i is L-smooth. We assume WGC is satisfied for approximate gradient $\overline{\nabla}_i$ with parameter ρ. Then series $\{x_i\}_{i \geq 0}$ generated by iteration formula*

$$x_{k+1} := x_k - \frac{1}{N} \sum_{i=1}^{N} \overline{\nabla}_i f_i(x) \tag{5}$$

satisfy

$$\mathbb{E}\left[\|x_{k+1}-x^*\|^2|x_k\right] \leq \left(1-\frac{\mu}{\rho L}\right)\|x_k-x^*\|^2.$$

This is exactly the theorem 5 in [21]. Using this lemma, we have the following corollary.

Corollary 9.2. *Suppose $f:\mathbb{R}^d\to\mathbb{R}$ is μ-strongly-convex. Besides, $f(x)=\frac{1}{N}\sum_{i=1}^{N} f_i(x)$, where each f_i is L-smooth. We assume WGC is satisfied for approximate gradient $\overline{\nabla}_i$ with parameter ρ. Then series $\{x_i\}_{i\geq 0}$ generated by iteration formula*

$$x_{k+1} := x_k - \frac{1}{N}\sum_{i=1}^{N}\overline{\nabla}_i f_i(x) + e_k \tag{6}$$

where e_k is the random error such that 1) $\|e_k\|^2 \leq \rho^2 L^2 s c'^{k+1}$, and 2) $\mathbb{E}[e_k]=0$, then we have $\|x_k-x_0\|^2 \leq c'^k \xi_s$ where $\xi_s := \max\left\{0, x_0-\frac{cs}{c'-c}\right\}+\frac{c's}{c'-c}$.

Proof. By lemma 9.1 we know

$$\begin{aligned}\|x_{k+1}-x^*\|^2 &\leq c\|x_k-x^*\|^2+\eta^2\cdot\rho^2L^2sc'^{k+1}\\ &= c\|x_k-x^*\|^2+sc'^{k+1}\end{aligned}$$

by taking $\eta=\frac{1}{\rho L}$.

Then by lemma 7.1 we know $\mathbb{E}\left[\|x_k-x_0\|^2\right]$ converge linearly. Finally, by the concavity of $\ln x^2 = 2\ln x$ and lemma 2.1 we know it cost at most $O(d)$ bits per communication on average. □