An Empirical Study on Compressed Decentralized Stochastic Gradient Algorithms with Overparameterized Models

Arjun Ashok Rao* and Hoi-To Wai*

* Dept. of SEEM; The Chinese University of Hong Kong; Shatin, Hong Kong
E-mail: arjunrao@link.cuhk.edu.hk, htwai@cuhk.edu.hk

Abstract—This paper considers decentralized optimization with application to machine learning on graphs. The growing size of neural network (NN) models has motivated prior works on decentralized stochastic gradient algorithms to incorporate communication compression. On the other hand, recent works have demonstrated the favorable convergence and generalization properties of overparameterized NNs. Through simulations on an MPI network environment, we observe that the convergence rates of popular compressed decentralized stochastic gradient (DSG) algorithms with overparameterized NNs. Our findings suggest a gap between theories and practice of the compressed DSG algorithms in the existing literature.

Index Terms—decentralized optimization, communication efficiency, overparameterized models

I. INTRODUCTION

A popular approach for enabling scalable machine learning is applying decentralized algorithms to tackle the large-scale optimization problem through collaboration between a group of workers connected on a network/graph. Let $N \in \mathbb{N}$ be the number of workers and $d \in \mathbb{N}$ be the problem’s dimension, we consider the following unconstrained optimization problem:

$$\min_{\theta \in \mathbb{R}^d} J(\theta) \quad \text{where} \quad J(\theta) := \frac{1}{N} \sum_{i=1}^{N} J_i(\theta),$$

where $J_i : \mathbb{R}^d \to \mathbb{R}$ is a continuously differentiable function representing the private data held by the $i$th worker. Furthermore, the $N$ workers are connected on an undirected graph denoted by $G = (V, E)$, where $V = [N] = \{1, \ldots, N\}$ is the set of workers and $E \subseteq V \times V$ is the set of edges of $G$ with self loops such that $(i, i) \in E$ for all $i \in V$. Let $G$ be a connected graph, we note that (1) is equivalent to the consensus optimization problem:

$$\min_{\theta_i \in \mathbb{R}^d, i \in V} \sum_{i=1}^{N} J_i(\theta_i) \quad \text{s.t.} \quad \theta_i = \theta_j, \ \forall (i, j) \in E,$$

where $\theta_i \in \mathbb{R}^d$ is a private/local variable held by the $i$th worker. The applications of (1) include decentralized regression, sensor fusion for wireless sensor network [1], etc.

In this paper, we are concerned with the application of (1) to machine learning (ML) tasks via training a neural network (NN) model. Following the design of (1), our goal is to train a common model $\theta$ at all workers. For example, if we consider a supervised learning problem for classification, the $i$th private function takes the form of the empirical risk:

$$J_i(\theta) = \frac{1}{|D_i|} \sum_{j=1}^{|D_i|} \text{loss}(f(x_j; \theta); y_j),$$

where $x_j \in \mathbb{R}^d$ and $y_j \in \mathbb{R}$ are the $j$th feature and label known by worker $i$, respectively, and $|D_i|$ is the number of samples held by worker $i$. The loss function loss($\cdot$) can be taken as the cross-entropy, or the quadratic loss. The nonlinear function $f(x; \theta)$ is the output of a neural network, e.g., a two-layer neural network with ReLU activation is given by

$$f(x; \theta) = \frac{1}{\sqrt{m}} \sum_{j=1}^{m} b_j \max\{0, \langle x, \theta^{(j)} \rangle\},$$

where $b_j$ is the $j$th output weight and we have defined the parameters as $\theta = (\theta^{(1)}, \ldots, \theta^{(m)}) \in \mathbb{R}^{mf}$ such that $d = mf$. Notice that despite its simplicity, the NN architecture (3) exhibits good representation power provided that $m \to \infty$ [2].

To tackle (1) when only local communications are allowed, decentralized first-order optimization algorithms have been developed for over a decade [1], [3], [4]. The main idea behind these algorithms is a simultaneous “consensus + optimize” strategy where workers communicate with each other to reach a common model $\theta$ (i.e., achieving consensus) while optimizing their local models $\theta_i$ via gradient steps on the private functions. For a non-convex optimization model, the convergence of decentralized optimization algorithms has been analyzed in [5]–[7] under the deterministic setting. Recent works have also analyzed the extension to stochastic gradient (SG) based methods [8]–[12]; also see [13] for a recent overview.

A major obstacle in applying the above algorithms to ML tasks [3] lies in the communication overhead. Taking the decentralized gradient (DGD) method from [3] as an example, each training iteration requires workers to transmit the entire $d$-dimensional local model. Notice that the state-of-the-art NN models are typically large with $d \gg 1$, e.g., the VGG16 NN comprises of $d \approx 1.38 \times 10^8$ parameters [14]. When applying the DGD method to train such a model, workers would be required to send 526 MB of data over the network.
per-iteration: This may cause a significant slowdown to the practical convergence of the algorithms.

In light of the above, previous works have been motivated to develop communication efficient variants of decentralized SG algorithms. To list a few examples, [15], [16] proposed the CHOCO-SGD algorithm which compresses incremental vectors at the workers prior to transmission; also see its generalization in [17]; [18] proposed the ECD-PSGD algorithm based on an extrapolation technique. It is worth noting that the above algorithms share similarities with the compression techniques proposed for distributed SG such as [19].

On the other hand, recent works have studied the global convergence of centralized SG algorithms for training overparameterized NN models [20] via the non-convex optimization [1]. 3. Particularly, these works consider simple architectures such as (4) and show that the optimal weights \( \theta^* \) to (1) can be found using SG algorithms when \( m \to \infty \), e.g., [21], [22]. Interestingly, it is demonstrated that the convergence rate for the \( L^2 \) function distance will be independent of \( m \), implying that there is no loss in iteration complexity (for centralized learning) when deploying overparameterized NNs.

The above observation illustrates a dilemma when deciding the NN architecture to be used for decentralized training: on one hand, it is desired to adopt a compact NN model to reduce communication cost, on the other hand, using an overparameterized NN model provides enticing theoretical convergence properties and higher representation power. The aim of this paper is to empirically study the effects of overparameterization on the performance of (communication efficient) decentralized SG algorithms. Particularly, we verify the claim that decentralized SG algorithms are practical (in terms of overall communication efficiency), even for training overparameterized NNs. Our contributions are as follows:

• In Section II we highlight the pitfalls of existing theories for training NNs with communication-efficient decentralized SG algorithms by showing that the curse of dimensionality persists. We argue that the latter can be an artifact of the analysis as it does not exploit the structure of the NN training problems.

• To support our claim, in Section III, we concentrate on the CHOCO-SGD algorithm [16] and experiment with this algorithm in various setting, providing the first set of experiments that highlights on the effects of an increasing NN width \( m \) in the model [4]. We show that, when the amount of information transmitted per iteration is fixed, the algorithm achieves a lower training loss / testing error at the same speed (or even faster) as the NN’s width is increased.

In addition, we discuss the intuitive reason behind the current gap in the theories and suggest potential solutions to fix it.

II. DECENTRALIZED LEARNING: ALGORITHMS AND COMMUNICATION EFFICIENCY

In this section, we discuss the algorithms for decentralized learning/training and their communication efficient variants. Furthermore, we will review the convergence guarantees of these algorithms and discuss how they may fail to yield meaningful insights for training overparameterized NNs.

To fix ideas, we define the doubly stochastic mixing matrix \( W \in \mathbb{R}^{N \times N} \) satisfying the row/column sum condition \( \mathbf{W} \mathbf{1} = \mathbf{W}^T \mathbf{1} = \mathbf{1} \); it respects the graph topology such that \( W_{ij} = W_{ji} = 0 \) whenever \((i,j) \notin E\); moreover, it satisfies the fast mixing condition of a Markov chain such that

\[
\|W - \mathbf{1}\mathbf{1}^T/N\| \leq 1 - \rho,
\]

where \( \rho \in (0, 1] \) is the spectral gap. Notice that such matrix exists for any connected graph \( G \).

With the above mixing matrix, the DGD (or its stochastic variant, DSGD) algorithm [3], [8] is given by: starting with any \( \theta^{(0)}_i \in \mathbb{R}^d \), \( i \in V \), we have

\[
\theta_i^{(t+1)} = \sum_{j=1}^{N} W_{ij} \hat{\theta}_j^{(t)} - \eta_i g_i^{(t)}, \quad \forall \ i \in V, \ \forall \ t \geq 0,
\]

where \( \eta_i > 0 \) is the step size and \( g_i^{(t)} \) is the local stochastic gradient which is an unbiased estimate of \( \nabla f_i(\theta^{(t)}) \). The algorithm (6) operates through a “consensus-then-optimize” strategy. At each iteration, the worker \( i \) first calculates an average of the local models held by the neighboring workers (\( \sum_{j=1}^{N} W_{ij} \theta_j^{(t)} \)), then a stochastic gradient step w.r.t. the private function \( -\eta_i g_i^{(t)} \) is performed. We now consider implementing algorithm (6) for decentralized training of an NN model such as (4). As the latter is parameterized by the weights \( \theta \in \mathbb{R}^{mf} \), the workers are required to transmit \( mf \) real numbers to the neighbors on the network/graph at each iteration. For \( m \gg 1 \), this poses a significant challenge due to the limited communication bandwidth. A natural remedy is to compress information before transmitting.

However, directly compressing the local parameters \( \theta_i^{(t)} \) in (6) can result in a non-convergent algorithm since the compression operation can lead to unrecoverable information loss. A better idea is to exploit smoothness and focus on compressing the differences in the iterates. In particular, [13] proposed the CHOCO-SGD algorithm for communication efficient decentralized training, as summarized in Algorithm 1.

We observe that step \( 3 \) in the algorithm is the only step of CHOCO-SGD involving peer-to-peer communication between the workers. To understand the algorithm better, we observe that when the compression operator is an identity operator, i.e., \( Q(\theta) = \theta \) for any \( \theta \in \mathbb{R}^d \), together with the consensus parameter \( \gamma = 1 \), the CHOCO-SGD algorithm is reduced into:

\[
\theta_i^{(t+1)} = \sum_{j=1}^{N} W_{ij} (\hat{\theta}_j^{(t)} - \eta_i g_j^{(t)}),
\]

which resembles the classical DSGD algorithm with a swapped order for the ‘optimize’ and ‘consensus’ steps.

In general, the communication step depends on a compression operator \( Q : \mathbb{R}^d \to \mathbb{R}^k \) which reduces the amount of information transmitted. Furthermore, we are compressing the difference between the successive iterates with the local
**Algorithm 1** CHOCO-SGD Algorithm [15]

1. **INPUT:** initial weights \( \theta_i^{(0)} \mid i = 1, \ldots, N \), max. no. of iterations \( T \), consensus parameter \( \gamma \in (0,1) \), step sizes \( \eta_i \geq 0 \).
2. Set the auxiliary variables \( \theta_i^{(0)} = 0 \), \( j \in N_i \), \( i \in [N] \).
3. Draw the stopping iteration number \( T \sim \mathcal{U}(0, \ldots, T) \).
4. for \( t = 0, 1, \ldots, T \) do
5. for \( i = 1, \ldots, N \) do //Local SGD step//
6. Compute the local SGD:
   \[
   \theta_i^{(t+1)} = \theta_i^{(t)} - \eta_i g_i^{(t)},
   \]
   where \( g_i^{(t)} \) is the stochastic estimate of \( \nabla J_i(\theta_i^{(t)}) \).
7. end for
8. For each worker \( i = 1, \ldots, N \), broadcast the compressed difference vector \( Q_i^{(t+1)} = \hat{\theta}_i^{(t+1)} - \hat{\theta}_{i,i}^{(t+1)} \) to the neighbors, where \( Q(\cdot) \) is a compression operator satisfying [8].
9. for \( i = 1, \ldots, N \) do //Combination step//
10. Update the auxiliary variable:
   \[
   \hat{\theta}_{i,j}^{(t+1)} = \hat{\theta}_{i,j}^{(t)} + Q_i^{(t+1)} - \hat{\theta}_{i,j}^{(t)}, \quad \forall j \in N_i.
   \]
11. Update the local NN weights:
   \[
   \theta_i^{(t+1)} = \theta_i^{(t+1)} + \gamma \sum_{j \in N_i} W_{ij} (\hat{\theta}_{i,j}^{(t+1)} - \hat{\theta}_{i,j}^{(t+1)}).
   \]
12. end for
13. end for
14. **OUTPUT:** trained weights \( \{\theta_i^{(T)}\}_{i=1}^N \).

Meanwhile, biased compression operators \( Q(\cdot) \) are also widely adopted. For example, the sign compression [24] and top sparsification which retains the top-\( k \) coordinates in the \( d \)-dimensional vector with the highest magnitude [23], [25]. Notably, the top sparsification satisfies (8) with \( \delta = \frac{\rho^2}{2} \) since the latter compressor always yield an error lower than that of the rand\( k \) sparsifier.

### A. Convergence Guarantee and Its Pitfalls

We discuss the convergence guarantees of the CHOCO-SGD algorithm and highlights on its pitfalls in decentralized training of overparameterized NNs.

We describe the assumptions used. First, we assume that each private function is smooth, i.e., the gradient map is Lipschitz continuous:

**Assumption 1** For any \( i \in [N] \), there exists \( L \geq 0 \) such that
\[
\|\nabla J_i(\theta) - \nabla J_i(\theta')\| \leq L\|\theta - \theta'\|, \forall \theta, \theta' \in \mathbb{R}^d.
\] (10)

Next, we specify conditions on the stochastic gradient estimates. Denote \( F_t \) as the filtration generated by the random variables \( \{\theta_i^{(t)} : i \in V, 0 \leq t \leq T\} \). We assume that:

**Assumption 2** There exists \( \sigma, G \geq 0 \) such that for any \( i \in [N] \), \( t \geq 0 \), the stochastic gradient \( g_i^{(t)} \) satisfies
\[
\mathbb{E}[g_i^{(t)}|F_t] = \nabla J_i(\theta_i^{(t)}), \quad \mathbb{E}[\|g_i^{(t)}\|^2|F_t] \leq G^2, \\
\mathbb{E}[\|g_i^{(t)} - \nabla J_i(\theta_i^{(t)})\|^2|F_t] \leq \sigma^2.
\] (11)

Notice that the first condition is satisfied by a uniform sampler for the (mini-batch) stochastic gradient, e.g., when \( g_i^{(t)} = \nabla \text{loss}(f(x_j; \theta_i^{(t)}); y_{j,i}) \) such that \( j_i \) is selected uniformly at random from \( \{1, \ldots, |D_i|\} \). We have also assumed that the stochastic gradients have bounded second order moments.

We observe the following result that is borrowed from [16] Theorem 4.1 on the CHOCO-SGD algorithm:

**Theorem 1** Under Assumptions 1, 2 and suppose that the compressor satisfies (8) there exists \( \eta_t > 0 \) such that if we consider a constant step size with \( \eta_t \equiv \eta \) then for any \( T \geq 1 \), the output generated by Algorithm 1 satisfy:
\[
\mathbb{E}[\|\nabla J(\bar{\theta}^{(T)})\|^2] \leq \mathcal{O}\left(\frac{L\sigma^2 J_0}{NT} + \left(\frac{LGJ_0}{\rho^2 T}\right)^{\frac{2}{3}}\right),
\]
where the expectation is taken over \( T \) and the stochastic quantities in the algorithm, \( \delta \) was defined in [8], \( \rho \in (0,1) \) is the spectral gap of \( W \) defined in [5], \( \bar{\theta}^{(0)} = \sum_{i=1}^{N} \theta_i^{(0)}/N \) is the network average iterate, and \( J_0 = J(\bar{\theta}^{(0)}) - \min_\theta J(\theta) \).

The theorem suggests that the CHOCO-SGD algorithm finds an \( O(1/\sqrt{T}) \)-stationary solution to (1) in at most \( T \) iterations as we note that \( T \leq T \).

**Convergence for Overparameterized Models.** We concentrate on the performance of CHOCO-SGD when \( d \gg 1 \), for example, when training an overparameterized NN such as [4].
CHOCO-SGD algorithm may require a higher communication cost during training, which is in contrast to the said observation. We conjecture that such phenomena is a consequence of an artifact in the existing analysis and discuss the possible fix for the observation.

III. EMPIRICAL STUDIES

In this section, we perform numerical experiments to examine the performance of compressed DSGD method when applied to training overparameterized NNs. We concentrate on the effects of the number of parameters on the communication complexity using the CHOCO-SGD method [16] [cf. Algorithm 1]. For simplicity, we consider a two-layer NN with ReLU activation described in [4] and adjust the width, \( m \), of the NN.

We consider the task of training a classifier with the CIFAR-10 dataset [27] that contains 50K (resp. 10K) training (resp. test) samples. Each sample consists of a 32 \times 32 RGB image which can be represented as a 3072-dimensional vector, and is associated with a label selected from 10 image classes. To simulate the decentralized training environment, samples from the 10 image classes are uniformly split among \( N \) workers and shuffled at every epoch – as in [16, 28]. To establish a challenging generalization task, we test the trained models on CIFAR-10.1 [29].

For the CHOCO-SGD method, we use a minibatch size of \( \xi = 128 \) for every iteration. Our chosen mode of communication compression is to\( \rho_k \), and random\( \rho_k \) with a fixed number of co-ordinates \( k \) allowed to be communicated between workers. We choose a constant consensus parameter \( \gamma = 0.0375 \) in Algorithm [1] and an SGD stepsize \( \eta = 0.1 \) which is decreased by a factor of 10 on epochs 100, 150, 200. Our to\( \rho_k \) and random\( \rho_k \) simulations are run on \( N = 8 \) nodes of a ring topology. The decentralized training environment is simulated on an MPI-based [30] network where we assign an independent CPU process to each worker.

Our first empirical example aims at comparing the quality of solution found by CHOCO-SGD after performing 300 epochs of iterations against the number of parameters of the trained NN (\( d \)). Notice that as \( k \) is fixed in the compressor design, the
communication complexity (i.e., number of bits transmitted over the network) is fixed. Our results are presented in Fig. 1(a) and 1(b). From the figures, we observe that the training loss decreases with \( d \), especially when the top node sparsifier is used. Notice that the above is actually in contrast to the observations made in [12], based on Theorem 1, which predicts that the solution quality decreases as the problem dimension increases. In Table I we compare the converged test accuracy achieved for different NN layer widths. Overparameterized models enjoy significantly greater consensus among workers with only marginal dependence on sparsification co-ordinate bandwidth for larger models.

The above metric is compared against the width of the hidden layer in Fig. 2. Notice that the spectral gap decreases with the number of coordinates kept in the sparsifier \( k \). Moreover, an intriguing observation is that the consensus error increases when the width of the NN increases from \( m = 1024 \) to \( m = 2048 \). We recall from Fig. 1 that the training loss/testing accuracy actually decreases with this change in the number of parameters. This suggests that a consensus solution may not be necessary for achieving a lower training loss in the overparameterized setting.

Our second empirical example examines the consensus error in the converged solution after 300 epochs of CHOCO-SGD with \( N = 8 \) workers. We aim at studying the consensus error of the converged solutions and the effects of problem dimension. Here, the normalized consensus error is defined as:

\[
\gamma = \frac{1}{N} \sum_{i=1}^{N} \frac{\| \theta_i^T - \theta_{top}^T \|^2}{\| \theta_{top}^T \|^2},
\]

(14)

where \( \theta_i^T \) denotes the CHOCO-SGD solution after 300 epochs. The above metric is compared against the width of the hidden layer, \( m \), in the NN in Fig. 3. Naturally, we observe that the consensus error increases when the number of coordinates kept in the sparsifier \( k \) decreases. Moreover, an intriguing observation is that the consensus error increases when the width of the NN increases from \( m = 1024 \) to \( m = 2048 \). We recall from Fig. 1 that the training loss/testing accuracy actually decreases with this change in the number of parameters. This suggests that a consensus solution may not be necessary for achieving a lower training loss in the overparameterized setting.

Our third empirical example examines the effects of graph topology on the converged training loss after 300 epochs of CHOCO-SGD iterations. In this experiment, we fix the number of hidden neurons at \( m = 256 \) and use a top node sparsifier with \( k = 51 \). We experiment with common graph topologies such as ring (\( deg_{ring} = 2 \)), torus (\( deg_{torus} = 4 \)), and Watts–Strogatz (small-world) graphs [31], and for different numbers of connected neighbors. Notice that the spectral gap parameter \( \rho \) decreases with the number of workers \( N \), especially for the ring topology. The result is presented against the number of workers in Fig. 5. As observed in the figures, the training loss decreases with the number of workers \( N \).

The deterioration in performance is the most severe with the ring topology. Unlike the previous example, we notice that the observed behavior is consistent with the theory in Theorem 1.
which predicts a slower convergence when the graph topology has a smaller $\rho$.

Lastly, we study a case of decentralized training with heterogeneous data. Particularly, we allocate samples from a single class to each of the $N = 8$ workers. We evaluate the training loss with global data using the local model $\theta_i^t$ obtained by CHOCO-SGD with $\text{top}_k$ sparsifier after 300 epochs; compared to the averaged model $(1/N)\sum_{i=1}^{N} \theta_i^t$. The simulation result is shown in Fig. 5. We observe that a similar trend as Fig. 1 holds as the training loss decreases with the hidden layer width $m$ given that the algorithms are run with a similar communication budget. Furthermore, we note that the training loss is higher with the local model.

**Discussions.** The above numerical examples demonstrate a consistent discrepancy between existing theories on compressed DSGD methods and their practical performances when training overparameterized NN models. In particular, we show that with the same communication cost allowed, the performance of the CHOCO-SGD trained NN improves with the number of neurons $m$ employed in the NN, contrary to \cite{12}.

The careful readers may notice that the poor dependence on dimension $d$ in the convergence bounds \cite{12} can be seen as a direct consequence of the worst-case analysis of the compression error in \cite{8}. This is because one is forced to take $\delta = d/k$ to account for the condition that holds for all $x \in \mathbb{R}^d$ in case of the $\text{top}_k/\text{rand}_k$ sparsifier. Furthermore, we remark that the existing theories are developed for general distributed optimization problems, assuming only high level properties such as smoothness of the objective function.

On the other hand, recent works on overparameterized models \cite{20,22} have shown that the training of such models with SGD shall be compared to the learning of a kernel model in the RKHS. Among others, a key innovation therein is that the optimality gap is measured in terms of the distance in the (infinite-dimensional) function space, where the rate of convergence is shown to be independent of $m$ even as $m \to \infty$. We believe that extending these results to a decentralized setting such as the CHOCO-SGD method can help break the curse of dimensionality in the analysis. We remark that an interesting observation was made in \cite{52} on the stability of overparameterized NN models with dropout, an operation that is akin to applying the sparsifier.

**IV. Conclusions**

This paper provides the first empirical study on the performance of compressed DSGD methods on overparameterized NN models. On the positive side, our result shows that utilizing overparameterized NN models in a decentralized learning is both practical and beneficial, contradicting existing theories that suggest otherwise. Furthermore, we identify a gap in the existing theories that attempt to analyze compressed DSGD methods. We believe that our article will serve as a motivating study to develop improved communication efficient algorithms.

| # Neurons | # Coordinates $k$ | $\text{top}_k$ sparsifier | $\text{rand}_k$ sparsifier |
|-----------|-----------------|--------------------------|--------------------------|
| $m = 2048$ | 20, 100, 128, 200 | 46.481 ± 0.767, 46.837 ± 0.354, 46.663 ± 1.544, 46.711 ± 0.755 | 46.919 ± 1.401, 43.288 ± 0.769, 46.365 ± 1.1 |
| $m = 1021$ | 20, 100, 128, 200 | 46.776 ± 0.423, 46.184 ± 0.937, 46.688 ± 1.05, 47.156 ± 0.999 | 48.245 ± 0.855, 47.135 ± 1.103, 47.309 ± 0.984 |
| $n = 256$  | 20, 100, 128, 200 | 45.651 ± 2.319, 46.266 ± 0.300, 47.787 ± 1.111, 46.292 ± 0.584 | 46.497 ± 0.653, 46.47 ± 0.401, 45.505 ± 0.81 |
| $n = 128$  | 20, 100, 128, 200 | 45.286 ± 0.413, 45.148 ± 1.103, 45.051 ± 1.100, 45.123 ± 0.855 | 45.707 ± 0.751, 44.928 ± 0.846, 44.733 ± 0.616 |
| $n = 128$  | 20, 100, 200, 46.497 | 44.59 ± 0.984, 43.967 ± 0.851, 42.778 ± 0.45 | N/A |

Table I. Generalization performance of CHOCO-SGD to CIFAR-10.1 with overparameterized models. Contrary to convergence performance, large-width NNs trained with $\text{top}_k$ and $\text{rand}_k$ sparsification exhibit marginal dependence to model dimension at test-time (average accuracy disparity of 2.39% between 2048 and 128 unit models). Test results averaged over 5 independent trials with altered seeds on a ring-topology communication graph with 8 nodes. CHOCO-SGD diverges when parameters are compressed with $\text{rand}_k$ sparsification of $k = 20$ co-ordinates.
for decentralized training of NNs.

ACKNOWLEDGEMENTS

This work is supported by CUHK Direct Grant #4055113.

REFERENCES

[1] A. Sayed, “Adaptation, learning, and optimization over networks,” Foundations and Trends® in Machine Learning, vol. 7, no. 4-5, pp. 311–801, 2014.

[2] F. Bach, “Breaking the curse of dimensionality with convex neural networks,” The Journal of Machine Learning Research, vol. 18, no. 1, pp. 629–681, 2017.

[3] A. Nedic and A. Ozdaglar, “Distributed subgradient methods for multi-agent optimization,” IEEE Transactions on Automatic Control, vol. 54, no. 1, pp. 48–61, 2009.

[4] W. Shi, Q. Ling, G. Wu, and W. Yin, “Extra: An exact first-order algorithm for decentralized consensus optimization,” SIAM Journal on Optimization, vol. 25, no. 2, pp. 944–966, 2015.

[5] P. Bianchi and J. Jakubowicz, “Convergence of a multi-agent projected stochastic gradient algorithm for non-convex optimization,” IEEE transactions on automatic control, vol. 58, no. 2, pp. 391–405, 2012.

[6] T. H. Lorenzo and G. Scutari, “Next: In-network nonconvex optimization,” IEEE Transactions on Signal and Information Processing over Networks, vol. 2, no. 2, pp. 120–136, 2016.

[7] M. Hong, D. Hajinezhad, and M.-M. Zhao, “Prox-pda: The proximal primal-dual algorithm for fast distributed nonconvex optimization and learning over networks,” in International Conference on Machine Learning. PMLR, 2017, pp. 1529–1538.

[8] X. Lian, C. Zhang, H. Zhang, C.-J. Hsieh, W. Zhang, and J. Liu, “Can decentralized algorithms outperform centralized algorithms? a case study for decentralized parallel stochastic gradient descent,” arXiv preprint arXiv:1705.09056, 2017.

[9] J. Zeng and W. Yin, “On nonconvex decentralized gradient descent,” IEEE Transactions on signal processing, vol. 66, no. 11, pp. 2834–2848, 2018.

[10] H. Tang, X. Lian, M. Yan, C. Zhang, and J. Liu, “D²: decentralized training over decentralized data,” in International Conference on Machine Learning. PMLR, 2018, pp. 4848–4856.

[11] H. Sun, S. Lu, and M. Hong, “Improving the same and communication complexity for decentralized non-convex optimization: Joint gradient estimation and tracking,” in International Conference on Machine Learning. PMLR, 2020, pp. 9217–9228.

[12] R. Xin, U. A. Khan, and S. Kar, “An improved convergence analysis for decentralized online stochastic non-convex optimization,” IEEE Transactions on Signal Processing, vol. 69, pp. 1842–1856, 2021.

[13] T.-H. Chang, M. Hong, H.-T. Wai, X. Zhang, and S. Lu, “Distributed learning in the nonconvex world: From batch data to streaming and beyond,” IEEE Signal Processing Magazine, vol. 37, no. 3, pp. 26–38, 2020.

[14] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” arXiv preprint arXiv:1409.1556, 2014.

[15] A. Koloskova, S. Stich, and M. Jaggi, “Decentralized stochastic optimization and gossip algorithms with compressed communication,” in International Conference on Machine Learning. PMLR, 2019, pp. 3478–3487.

[16] A. Koloskova, T. Lin*, S. U. Stich, and M. Jaggi, “Decentralized deep learning with arbitrary communication compression,” in International Conference on Learning Representations, 2020.

[17] D. Kovalev, A. Koloskova, M. Jaggi, P. Richtarik, and S. Stich, “A linearly convergent algorithm for decentralized optimization: Sending less bits for free!” in International Conference on Artificial Intelligence and Statistics. PMLR, 2021, pp. 4087–4095.

[18] H. Tang, S. Gan, C. Zhang, T. Zhang, and J. Liu, “Communication compression for decentralized training,” Advances in Neural Information Processing Systems, vol. 31, pp. 7652–7662, 2018.

[19] D. Alistarh, D. Grubic, J. Li, R. Tomioka, and M. Vojnovic, “Qsgd: Communication-efficient sgd via gradient quantization and encoding,” Advances in Neural Information Processing Systems, vol. 30, pp. 1709–1720, 2017.

[20] A. Jacot, F. Gabriel, and C. Hongler, “Neural tangent kernel: Convergence and generalization in neural networks,” arXiv preprint arXiv:1806.07572, 2018.

[21] S. Arora, S. Du, W. Hu, Z. Li, and R. Wang, “Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks,” in International Conference on Machine Learning. PMLR, 2019, pp. 322–332.

[22] A. Bakshi, R. Jayaram, and D. P. Woodruff, “Learning two layer rectified neural networks in polynomial time,” in Conference on Learning Theory. PMLR, 2019, pp. 195–268.

[23] 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, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds., vol. 31. Curran Associates, Inc., 2018.

[24] J. Bernstein, Y.-X. Wang, K. Azizzadenesheli, and A. Anandkumar, “signsgd: Compressed optimisation for non-convex problems,” in International Conference on Machine Learning. PMLR, 2018, pp. 560–569.

[25] S. Shi, X. Chu, K. C. Cheung, and S. See, “Understanding top-k sparsification in distributed deep learning,” arXiv preprint arXiv:1911.08772, 2019.

[26] Y. Lin, S. Han, H. Mao, Y. Wang, and B. Dally, “Deep gradient compression: Reducing the communication bandwidth for distributed training,” in International Conference on Learning Representations, 2018.

[27] A. Krizhevsky, G. Hinton et al., “Learning multiple layers of features from tiny images,” 2009.

[28] P. Goyal, P. Dollár, R. Girshick, P. Noordhuis, L. Wesolowski, A. Kyrola, A. Tulloch, Y. Jia, and K. He, “Accurate, large minibatch sgd: Training imagenet in 1 hour,” arXiv preprint arXiv:1706.02677, 2017.

[29] B. Recht, R. Roelofs, L. Schmidt, and V. Shankar, “Do cifar-10 classifiers generalize to cifar-10?” arXiv preprint arXiv:1806.00451, 2018.

[30] R. L. Graham, T. S. Woodall, and J. M. Squyres, “Open mpi: A flexible high performance mpi,” in International Conference on Parallel Processing and Applied Mathematics. Springer, 2005, pp. 228–239.

[31] D. J. Watts and S. H. Strogatz, “Collective dynamics of ‘small-world’ networks,” nature, vol. 393, no. 6684, pp. 440–442, 1998.

[32] A. Shevchenko and M. Mondelli, “Landscape connectivity and dropout stability of sgd solutions for over-parameterized neural networks,” in International Conference on Machine Learning. PMLR, 2020, pp. 8773–8784.