Decentralized Distributed Deep Learning

Wei Zhang, Xiangru Lian, Ce Zhang, Ji Liu
IBM T.J.Watson Research Center, University of Rochester, ETH Zurich, University of Rochester
weiz@us.ibm.com,lianxiangru@gmail.com,ce.zhang@inf.ethz.ch,ji.liu.uwisc@gmail.com

Abstract

Distributed deep learning is the de facto approach to training neural networks at scale. Traditional training approaches include Allreduce-based Synchronous Stochastic Gradient Descent(SSGD) and Parameter-server based Asynchronous Stochastic Gradient Descent (ASGD). The former approach suffers from the straggler problem and the latter approach is known to have unstable convergence behavior. In this paper, we take advantage of the recent breakthrough in theoretical non-convex optimization (i.e. Decentralized Parallel Stochastic Gradient Descent (DPSGD)) and demonstrate how its synchronous/asynchronous implementation is a simple yet powerful mechanism, for distributed deep learning training, that is friendly to heterogeneous system , low-bandwidth/high-latency hardware and provide excellent convergence guarantee.

Keywords Decentralized, Distributed, Deep Learning

1 Introduction

Traditional approaches to DL training include Allreduce-based Synchronous Stochastic Gradient Descent (SSGD) approach and parameter-server based Asynchronous Stochastic Gradient Descent (ASGD) approach. Allreduce is a classical HPC collectives operation[6] that can fast compute/reduce over a list of elements to get the final reduction results. When a message is large and network latency is low, ring-based allreduce algorithm[5, 7] is known to optimize the network bandwidth utilization. The total cost of allreduce over N-byte over arbitrary number of learners is $2\times N/B+O(n)$, where B is the network bandwidth between a pair of learners and n is the number of messages exchanged between a pair of learners. To enable pipe-lining of message transferring, N-byte message needs be chunked into n block of messages, thus n tends to be large. Parameter-server based approach[2, 3] allows different learners to work at different speed and not synchronize with one another. However, this approach suffers from two problems: (i) a centralized parameter server (group) that could become a performance bottleneck and (ii) unstable convergence behavior because of the staleness issue in ASGD[1, 8].

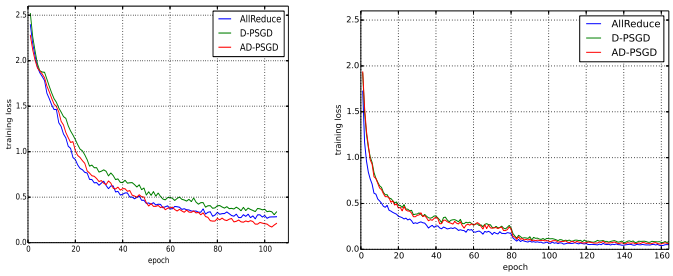
Decentralized Parallel Stochastic Gradient Descent (D-PSGD) Recent breakthrough in theoretical non-convex optimization [4], for the first time, proves a decentralized SGD approach has the same convergence rate as the centralized SGD approach (e.g., Allreduce) where all learners observe exactly the same weights during the gradients calculation. Specifically, (D-PSGD) only requires adjacent learners to exchange weights after each iteration of gradients calculation. D-PSGD outperforms Allreduce because it transfers the same amount of bulk message whereas the hand-shake message is O(1) and still maintains the same convergence rate (w.r.t #epochs). Its advantage over Parameter-server based asynchronous approach is the removal of a central bottleneck imposed by parameter server and has better empirical convergence speed (w.r.t #epochs). In this paper, we developed the AD-PSGD, the asynchronous version of the D-PSGD. In AD-PSGD, learners do not pose a global barrier at the end of gradients calculation as in D-PSGD, but only exchange weights with their neighbors and rely on a gossip-like protocol to reach the consensus of the weights. We demonstrate its superior convergence and runtime performance overall D-PSGD and Allreduce in this paper.

AISys'17, Octoboer 28, 2017, Shanghai, China
2017.

2 Preliminary Results

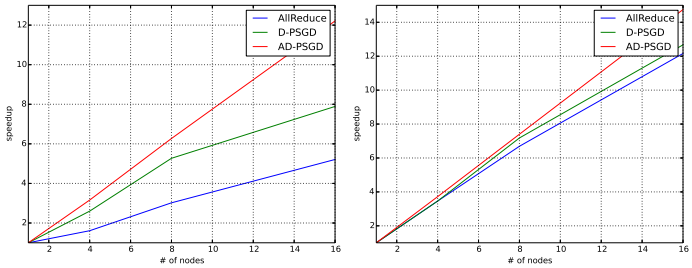
In [4], we have established that D-PSGD converges much faster than the ASGD and [1] demonstrates that in many cases SSGD(e.g., Allreduce) converges faster than ASGD. In this paper, we are particularly interested in comparing Allreduce, D-PSGD, AD-PSGD. We use IBM HPC cluster. Each computing node is an IBM Power8 S822LC system, the communication between each node is 100Gb/s mellonox EDR infiniband. We use CIFAR10 as the dataset and VGG (batch-size 128 per learner), RESNET-20 (batchsize 32 per learner) as the deep learning model. CIFAR10 contains 50K 32x32 color images that fall into 10 different classes. VGG model is of 60MB size and RESNET-20 is of 1MB size. We use Torch-7 as the underlying solver that calculates gradients, we implement the communication library using multi-threaded OpenMPI processes. Figure 1 shows that Allreduce, D-PSGD and AD-PSGD converge at the same rate (w.r.t #epochs).

In Figure 2, we plot the speedup for 16 learners (1 learner per computing node) for the VGG and ResNet-20. We use the same hyper-parameter setup (e.g., learning rate scheme, batch size) for Allreduce, D-PSGD and AD-PSGD.The model size of VGG is 60MB and the model size of ResNet-20 is 1MB. It takes 13.60 seconds to calculate through one epoch for VGG and 18.04 seconds to calculate through one epoch for ResNet-20. The reason that VGG is 60X bigger than the ResNet-20 but only takes similar time to compute is that VGG contains



(a) CIFAR10-VGG-loss **(b)** CIFAR10-RESNET20-loss

Figure 1. Loss for CIFAR10 VGG/Resnet.



(a) CIFAR10-VGG-speedup **(b)** CIFAR10-resnet20-speedup

Figure 2. Speedup for CIFAR10 VGG/Resnet

Table 1. Final accuracy of AllReduce, D-PSGD and AD-PSGD. We train VGG for 100 epochs, and ResNet for 164 epochs.

	AllReduce	D-PSGD	**AD-PSGD**
VGG	84.04%	84.48%	**88.58%**
ResNet	90.72%	90.81%	**91.02%**

less convolution layers but more fully-connected layers. Evidently VGG is a less communication friendly model and harder to parallelize. For the VGG workload, AD-PSGD runs twice faster than AllReduce and 20% faster than D-PSGD. For the ResNet-20 workload, AD-PSGD achieves near-linear speedup.

In addition, the final accuracy of the three algorithms listed in Table 1 shows decentralized algorithms especially AD-PSGD will achieve a better accuracy in the end. This suggests decentralized algorithms will help generalization.

A major advantage of asynchronous algorithms over their synchronous counterparts is that it can dynamically load-balance the system and transparently handle the situation where the speed of each participant is heterogeneous. We adjust the node's speed so that one learner (out of 16 learners) runs 2X, 10X, 100X, 1000X and 10000X slower in the system and compare the convergence and speedup. Table 2

Table 2. Speedup when there is one slower node in a 16-learner system on CIFAR-10 dataset and ResNet-20 model. TPE[1] stands for Time per Epoch

Slowdown of one node	AD-PSGD		AllReduce/D-PSGD	
	TPE[1] (sec)	Speedup	TPE (sec)	Speedup
no slowdown	1.22	14.78	1.47/1.45	12.27/12.44
2X	1.28	14.09	2.6/2.36	6.93/7.64
10X	1.33	13.56	11.51/11.24	1.56/1.60
100X	1.33	13.56	100.4	0.18
1000X	1.33	13.56	>1000	<0.018
10000X	1.33	13.56	>10000	<0.0018

records the speedup when there is a slower node. In addition, AD-PSGD trained model accuracy, in the presence of a slow learner, stays above 91%, which is higher than that of Allreduce trained model, which is 90.72%.

3 Conclusion and Future work

In this paper, we have demonstrated how to leverage the recent theoretical breakthrough in solving non-convex optimization (AD/D-PSGD) and apply it to deep learning training at scale. Future work includes: (i) Complete the theoretical study of AD-PSGD convergence rate. (ii) Examine the larger dataset (e.g., ImageNet-1K) and other models(e.g., LSTM) at a larger scale (e.g., several hundreds of GPUs). (iii) Experiment on other system types (e.g., cloud computing platform) and IoT systems where network latency is much higher and A/D-PSGD would perform much better than Allreduce because of smaller number of handshake messages.

Acknowledgments

Xiangru Lian and Ji Liu are supported by an IBM Faculty Award.

References

[1] Jianmin Chen, Rajat Monga, Samy Bengio, and Rafal Józefowicz. 2016. Revisiting Distributed Synchronous SGD. *CoRR* abs/1604.00981 (2016). http://arxiv.org/abs/1604.00981

[2] Trishul Chilimbi, Yutaka Suzue, Johnson Apacible, and Karthik Kalyanaraman. 2014. Project Adam: Building an Efficient and Scalable Deep Learning Training System *(OSDI'14)*. 571–582.

[3] Mu Li, David G. Andersen, Jun Woo Park, Alexander J. Smola, Amr Ahmed, Vanja Josifovski, James Long, Eugene J. Shekita, and Bor-Yiing Su. 2014. Scaling Distributed Machine Learning with the Parameter Server *(OSDI'14)*. 583–598.

[4] Xiangru Lian, Ce Zhang, Huan Zhang, Cho-Jui Hsieh, Wei Zhang, and Ji Liu. 2017. Can Decentralized Algorithms Outperform Centralized Algorithms? A Case Study for Decentralized Parallel Stochastic Gradient Descent. In *Advances in neural information processing systems NIPS'2017(Oral)*.

[5] Nathan Luehr. 2016. Fast Multi-GPU collectives with NCCL. (2016). https://devblogs.nvidia.com/parallelforall/fast-multi-gpu-collectives-nccl/

[6] MPI contributors. 2015. MPI AllReduce. (2015). http://mpi-forum.org/docs/

[7] Pitch Patarasuk and Xin Yuan. 2009. Bandwidth Optimal All-reduce Algorithms for Clusters of Workstations. *J. Parallel Distrib. Comput.* 69, 2 (Feb. 2009), 117–124. https://doi.org/10.1016/j.jpdc.2008.09.002

[8] Wei Zhang, Suyog Gupta, Xiangru Lian, and Ji Liu. 2016. Staleness-Aware Async-SGD for Distributed Deep Learning. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016*. 2350–2356.