High Performance Distributed Deep Learning: A Beginner's Guide

Dhabaleswar K. (DK) Panda
Dept. of Computer Science and Engg.
The Ohio State University
panda@cse.ohio-state.edu

Ammar Ahmad Awan
Dept. of Computer Science and Engg.
The Ohio State University
awan.10@osu.edu

Hari Subramoni
Dept. of Computer Science and Engg.
The Ohio State University
subramon@cse.ohio-state.edu

Abstract

The current wave of advances in Deep Learning (DL) has led to many exciting challenges and opportunities for Computer Science and Artificial Intelligence researchers alike. Modern DL frameworks like Caffe2, TensorFlow, Cognitive Toolkit (CNTK), PyTorch, and several others have emerged that offer ease of use and flexibility to describe, train, and deploy various types of Deep Neural Networks (DNNs). In this tutorial, we will provide an overview of interesting trends in DNN design and how cutting-edge hardware architectures are playing a key role in moving the field forward. We will also present an overview of different DNN architectures and DL frameworks. Most DL frameworks started with a single-node/single-GPU design. However, approaches to parallelize the process of DNN training are also being actively explored. The DL community has moved along different distributed training designs that exploit communication runtimes like gRPC, MPI, and NCCL. In this context, we will highlight new challenges and opportunities for communication runtimes to efficiently support distributed DNN training. We also highlight some of our co-design efforts to utilize CUDA-Aware MPI for large-scale DNN training on modern GPU clusters. Finally, we include hands-on exercises in this tutorial to enable the attendees to gain first-hand experience of running distributed DNN training experiments on a modern GPU cluster.

Keywords High-Performance Deep Learning, Machine Learning, DNN Training, HPC, MPI

1 Introduction and Scope

Recent advancements in Artificial Intelligence (AI) have been fueled by the resurgence of Deep Neural Networks (DNNs) and various Deep Learning (DL) frameworks like Caffe [5], Facebook Caffe2 [4], Facebook Torch/PyTorch [2], Chanter/ChainerMN [8], Google TensorFlow [3], and Microsoft Cognitive Toolkit (CNTK) [7]. DNNs have found widespread applications in classical areas like Image Recognition, Speech Processing, Textual Analysis, as well as areas like Cancer Detection, Medical Imaging, and even Autonomous Vehicle systems. Two driving elements can be attributed to the momentum that DL has gained recently; first is the public availability of various data sets like ImageNet [1], CIFAR-10 [6], etc., and second is the widespread adoption of data-parallel hardware like GPUs and accelerators to perform DNN training. The raw number crunching capabilities of GPUs have significantly improved DNN training. Today, the community is designing better, bigger, and deeper networks for improving the accuracy through models like AlexNet, GoogLeNet, Inception v3, and VGG. The models differ in the architecture (number and type of layers) but share the common requirement of faster computation and communication capabilities of the underlying systems. Based on these trends, this tutorial is proposed with the following objectives:

- Help newcomers to the field of distributed Deep Learning (DL) on modern high-performance computing clusters to understand various design choices and implementations of several popular DL frameworks.
- Guide Message Passing Interface (MPI) application researchers, designers and developers to achieve optimal training performance with distributed DL frameworks like Google TensorFlow, OSU-Caffe, CNTK, and ChainerMN on modern HPC clusters with high-performance interconnects (e.g., InfiniBand), NVIDIA GPUs, and multi/many core processors.
- Demonstrate the impact of advanced optimizations and tuning of CUDA-Aware MPI libraries like MVAPICH2 on DNN training performance through case studies with representative benchmarks and applications.

2 Tutorial Audience

This tutorial is targeted for various categories of people working in the areas of Deep Learning and MPI-based distributed DNN training on modern HPC clusters with high-performance interconnects. The specific audience this tutorial is aimed at include:

PPoPP '19, February 16–20, 2019, Washington, DC, USA

ACM ISBN 978-1-4503-6225-2/19/02.
https://doi.org/10.1145/3293883.3302260

- Scientists, engineers, researchers, and students engaged in designing next-generation Deep Learning frameworks and applications over high-performance interconnects and GPUs
- Designers and developers of Caffe, TensorFlow, and other DL frameworks who are interested in scaling-out DNN training to multiple nodes of a cluster
- Newcomers to the field of Deep Learning on modern high-performance computing clusters who are interested in familiarizing themselves with Caffe, CNTK, OSU-Caffe, and other MPI-based DL frameworks
- Managers and administrators responsible for setting-up next generation Deep Learning executions environments and modern high-performance clusters/facilities in their organizations/laboratories

3 Tutorial Outline

The tutorial is organized along the following topics

1. Introduction
   - The Past, Present, and Future of Deep Learning (DL)
     - Brief History and Current/Future Trends
     - DL Resurgence in the Many-core Era
   - What are Deep Neural Networks?
     - Brief Introduction
     - Training and Inference
   - Diverse Applications of Deep Learning?
     - Vision
     - Speech
     - Text
     - Autonomous Driving
   - Deep Learning Frameworks?
     - Why we need DL frameworks?
     - Define-by-run frameworks vs. Define-and-run
     - Caffe/Caffe2
     - Microsoft Cognitive Toolkit (CNTK)
     - Chainer/ChainerMN
     - Torch/PyTorch
     - Google TensorFlow
2. Overview of Execution Environments
   - Where do we run our DL Framework?
   - Conventional vs. Upcoming Execution Environments
   - DL Frameworks and Underlying (BLAS/DNN) Libraries
   - Holistic Performance Characterization
3. Parallel and Distributed DNN Training
   - The Need for Parallel and Distributed Training
   - Parallelization Strategies
   - Communication Runtimes
   - Scale-up and Scale-out
4. Latest Trends in HPC Technologies
   - HPC Hardware
     - Interconnects (InfiniBand, RoCE, and Omni-Path)
     - GPUs, Multi-/Many-cores, FPGAs, TPUs, Intel Neural Network Processor, Intelligence Processing Unit
     - Storage - NVMe, SSDs, Burst Buffers, etc.
   - Communication Middleware
     - Message Passing Interface (MPI)
     - NVIDIA NCCL/NCCL2 and Facebook Gloo
     - Intel Machine Learning Scaling Library
5. Challenges in Exploiting HPC Technologies
   - Large Batch and Model Size, Accuracy, and Scalability
   - Exploiting GPUs and CUDA-Aware MPI
   - Co-design of Communication Runtimes and DL Frameworks
   - Efficient Collective Communication for DL Workloads
6. Solutions and Case Studies
   - NVIDIA NCCL/NCCL2
   - LLNL Aluminum
   - Baidu-allreduce
   - Facebook Gloo and Caffe2
   - Co-design MPI Runtimes and DL Frameworks
     - MPI+NCCL for CUDA-Aware CNTK
     - Ring-based Optimized MPI for CUDA-Aware CNTK
     - OSU-Caffe
   - Distributed Training for TensorFlow
     - TensorFlow with gRPC
     - TensorFlow with No-gRPC
     - TensorFlow with gRPC+X (X=MPI,NCCL/NCCL2)
   - Scaling DNN Training on Multi-/Many-core CPUs
     - Intel Optimized Caffe + Intel MLSL
     - Intel Optimized TensorFlow
   - PowerAI Distributed Deep Learning
7. Hands-on Exercises
   - Distributed Training for TensorFlow + MPI (Horovod)
   - Horovod MPI with MVAPICH2 and MVAPICH2-GDR
   - Horovod MPI with MVAPICH2-GDR and NCCL2
8. Open Issues and Challenges
   - Which Framework should I use?
     - Use-cases, Eco-systems, and Application Domains
   - What is the Rationale behind NCCL/NCCL2, Gloo, and MPI?
   - Convergence of DL and HPC research
   - Thoughts on DL Benchmarks and Standardization
   - Scalability and Large batch-size training?
9. Conclusion

Acknowledgments

This research is supported in part by National Science Foundation grants #CNS-1513120, #ACI-1450440 and #CCF-1565414.

References

[1] 2018. ImageNet. http://image-net.org/. [Online; accessed January 3, 2019].
[2] 2018. PyTorch. https://pytorch.org/. [Online; accessed January 3, 2019].

[3] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, et al. 2016. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems, 2015. *arXiv preprint arXiv:1603.04467* (2016).

[4] Facebook. 2018. Caffe2 Framework. https://caffe2.ai.

[5] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. 2014. Caffe: Convolutional Architecture for Fast Feature Embedding. *arXiv preprint arXiv:1408.5093* (2014).

[6] A. Krizhevsky and G. Hinton. 2009. Learning Multiple Layers of Features from Tiny Images. http://www.cs.toronto.edu/˜kriz/learning-features-2009-TR.pdf.

[7] Microsoft. 2018. CNTK. http://www.cntk.ai/. [Online; accessed January 3, 2019].

[8] Preferred Networks. 2018. Chainer: A flexible framework for neural networks. https://chainer.org/. [Online; accessed January 3, 2019].