Simple and Automatic Distributed Machine Learning on Ray

Hao Zhang
hao@cs.berkeley.edu
UC Berkeley

Zhuohan Li
zhuohan@cs.berkeley.edu
UC Berkeley

Lianmin Zheng
lianminzheng@gmail.com
UC Berkeley

Ion Stoica
istoica@cs.berkeley.edu
UC Berkeley

ABSTRACT

In recent years, the pace of innovations in the fields of machine learning (ML) has accelerated, researchers in SysML have created algorithms and systems that parallelize ML training over multiple devices or computational nodes. As ML models become more structurally complex, many systems have struggled to provide all-round performance on a variety of models. Particularly, ML scale-up is usually underestimated in terms of the amount of knowledge and time required to map from an appropriate distribution strategy to the model. Applying parallel training systems to complex models adds nontrivial development overheads in addition to model prototyping, and often results in lower-than-expected performance. This tutorial identifies research and practical pain points in parallel ML training, and discusses latest development of algorithms and systems on addressing these challenges in both usability and performance. In particular, this tutorial presents a new perspective of unifying seemingly different distributed ML training strategies. Based on it, introduces new techniques and system architectures to simplify and automate ML parallelization. This tutorial is built upon the authors’ years’ of research and industry experience, comprehensive literature survey, and several latest tutorials and papers published by the authors and peer researchers.

The tutorial consists of four parts. The first part will present a landscape of distributed ML training techniques and systems, and highlight the major difficulties faced by real users when writing distributed ML code with big model or big data. The second part dives deep to explain the mainstream training strategies, guided with real use case. By developing a new and unified formulation to represent the seemingly different data- and model- parallel strategies, we describe a set of techniques and algorithms to achieve ML auto-parallelization, and compiler system architectures for auto-generating and exercising parallelization strategies based on models and clusters. The third part of this tutorial exposes a hidden layer of practical pain points in distributed ML training: hyper-parameter tuning and resource allocation, and introduces techniques to improve these aspects. The fourth part is designed as a hands-on coding session, in which we will walk through the audiences on writing distributed training programs in Python, using the various distributed ML tools and interfaces provided by the Ray ecosystem.

KEYWORDS

distributed deep learning; distributed system; parallelism

ACM Reference Format:
Hao Zhang, Zhuohan Li, Lianmin Zheng, and Ion Stoica. 2021. Simple and Automatic Distributed Machine Learning on Ray. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD ’21), August 14–18, 2021, Virtual Event, Singapore.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3447548.3470816

TARGET AUDIENCES AND PREREQUISITES

This tutorial is for ML practitioners and researchers who want to scale up their ML model training speeds, especially for structurally complex models (such as deep learning, tn) that cannot fit in a single node, or are very difficult to tune in practice. The tutorial addresses real-world, practical factors that are often overlooked or dismissed in typical academic presentations: such as heterogeneous clusters with different types of hardware, clusters with multiple users, spending budgets and cost limits, amount of code that needs to be written by developers and scientists, and meeting strict service level agreements. The audience will learn why these factors are important and affect ML training speeds, as well as the state-of-the-art ideas and solutions for addressing them. The audience will be able to apply this knowledge to improve their own ML research and application productivity.

The audience should be familiar with ML and DL basics. Knowledge of TensorFlow [1], PyTorch [3], Ray [5] is helpful but not required. Knowledge of distributed ML systems is also helpful but not required. To get full value out of this tutorial, the audience should be trying to scale up their own ML model training, but is running into difficulties, and wants to understand why these difficulties occur and how to solve them.

OUTLINE

The tutorial covers the following topics.

- Big models for big data: an overview of distributed machine learning training
  - Distributed ML training strategies
  - System techniques to support distributed ML training: memory, scheduling, communication, resource management
  - Programming interfaces of distributed ML

KDD ’21, August 14–18, 2021, Virtual Event, Singapore
© 2021 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-8332-5/21/08.
https://doi.org/10.1145/3447548.3470816

- New challenges of large-scale distributed ML
  - Heterogeneity of ML models, algorithms and cluster environment
  - Complexity of programming with big models and big data on distributed clusters
  - Added overheads on tuning and resource allocation
- Introduction of Ray: simple and universal API for building distributed applications in Python

End-to-end Automated ML Parallelization (1 hour).

- Industry case study: distributed pretraining of large language models
- Deep Dive: Data- and model- parallel training strategies
  - Data-parallel training: parameter server, AllReduce, etc. [2, 9]
  - Model-parallel training: operation partitioning, pipeline parallelism, etc. [4]
  - Combinations of parallelisms [7]
- Representing distributed ML training strategies using a unified representation
- Techniques and algorithms for optimizing training strategies w.r.t. model and clusters [8]
- Compiler systems for auto-generating and exercising distributed training strategies

Tuning, Resource Scheduling, and Auto-scaling (40 mins).

- Hyper-parameter optimization: algorithms and systems
- Resource allocation of multiple training jobs on shared clusters [6]
  - Trade-offs between resource allocation, training strategies, and hyper-parameters
  - Elastically schedule distributed DL training jobs in shared clusters
  - Cost-aware resource auto-scaling in cloud computing environments (e.g. AWS)
  - Automatic batch size and learning rate scaling for distributed training

Hands-on session: Writing End-to-end Distributed ML Programs using Ray and Parax (30 mins).

- Understanding a Ray cluster: writing distributed code as if on a single machine
- Using Ray collective APIs: high-performance communication APIs for distributed ML
- Generating training strategies using Ray distributed training compiler
- Efficient resource management and adaptive batch size scaling with AdaptDL on Ray
- Exploiting spot instances and auto-scaling up and down with Ray auto-scaler APIs

RELATED TUTORIALS

- **AAAI 2021** Tutorial on "Simplifying and Automating Parallel Machine Learning via a Programmable and Composable Parallel ML System" by Hao Zhang (same author), Aurick Qiao, Qirong Ho, and Eric P. Xing. Febuary 3, 2021. Tutorial website: https://sites.google.com/view/aaai-2021-tutorial-ah9/home. The prior tutorial is a variant of the proposed tutorial with its main scope on categorizing and introducing the various aspects in *data-parallel* distributed machine learning [2, 6–9]. Differently, this tutorial turns the focus on the latest development on model-parallel distributed ML, distributed ML compiler, automatic parallelization of ML programs, and state-of-the-art tools and systems to support the training of very large models, such as Ray.

SOCIETAL IMPACT

Given the explosion of interest in machine learning and its increasing impact on different application domains, this tutorial provides data scientists and practitioners with a refreshing, systematic picture over the landscape of ML algorithms for both a holistic understanding and practical guidance of designing ML solutions to problems in fields such as healthcare, manufacturing, finance, social science, and many others.

REFERENCES

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, and Michael Isard. Tensorflow: A system for large-scale machine learning. *arXiv preprint arXiv:1605.08695*, 2016.

[2] Henggang Cui, Hao Zhang, Gregory R Ganger, Phillip B Gibbons, and Eric P Xing. Geeps: Scalable deep learning on distributed gpus with a gpu-specialized parameter server. In *Proceedings of the Eleventh European Conference on Computer Systems*, pages 1–16, 2016.

[3] Facebook. Pytorch. http://pytorch.org/, 2018.

[4] Zhuohan Li, Siyuan Zhuang, Shiyuan Guo, Danyang Zhuo, Hao Zhang, Dawn Song, and Ion Stoica. Terapipe: Token-level pipeline parallelism for training large-scale language models. *arXiv preprint arXiv:2102.07988*, 2021.

[5] Philipp Moritz, Robert Nishihara, Stephanie Wang, Alexey Tumanov, Richard Liaw, Eric Liang, Melih Elibol, Zongheng Yang, William Paul, Michael I Jordan, et al. Ray: A distributed framework for emerging {AI} applications. In *13th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 18)*, pages 561–577, 2018.

[6] Aurick Qiao, Sang Keun Choe, Suhas Jayaram Subramanya, Willie Neiswanger, Qirong Ho, Hao Zhang, Gregory R. Ganger, and Eric P. Xing. Pollux: Co-adaptive cluster scheduling for goodput-optimized deep learning. In *15th USENIX Symposium on Operating Systems Design and Implementation (OSDI 21)*, pages 1–18. USENIX Association, July 2021.

[7] Hao Zhang. *Machine Learning Parallelism Could Be Adaptive, Composable and Automated.* PhD thesis, Carnegie Mellon University, Pittsburgh, PA, October 2020.

[8] Hao Zhang, Yuan Li, Zhijie Deng, Xiaodan Liang, Lawrence Carin, and Eric Xing. Autosync: Learning to synchronize for data-parallel distributed deep learning. *Advances in Neural Information Processing Systems*, 33, 2020.

[9] Hao Zhang, Zeyu Zheng, Shizhen Xu, Wei Dai, Qirong Ho, Xiaodan Liang, Zhiting Hu, Jinliang Wei, Pengtao Xie, and Eric P Xing. Poseidon: An efficient communication architecture for distributed deep learning on {GPU} clusters. In *2017 {USENIX} Annual Technical Conference ({USENIX}{ATC} 17)*, pages 181–193, 2017.