Scaling Deep Learning across Heterogeneous GPU/CPU Clusters with Accelerated Training Frameworks: A Brief Survey

Anh Vuong Tuan, Young Han Kim*

Soongsil University

anhvt@dcn.ssu.ac.kr, *younghak@ssu.ac.kr

Abstract

Distributed deep learning (DDL) on heterogeneous GPU/CPU clusters often faces significant performance bottlenecks due to issues like load imbalance, communication overhead, and inefficient hardware utilization. Recent systems have proposed effective strategies to address these challenges and improve training efficiency in such environments. This survey reviews four representative frameworks—BytePS, BytePS-Compress, Espresso, and StellaTrain—that have shown strong performance and scalability, even on heterogeneous clusters, by leveraging techniques such as decoupled communication, gradient compression, pipelining, and adaptive hyperparameter tuning. We summarize the architectural designs, optimization techniques, and discuss their applicability in real-world, resource-diverse infrastructures.

Ⅰ. Introduction

The rapid growth of deep neural network (DNN) complexity and massive datasets has necessitated the development of large-scale distributed systems to enable efficient training. However, optimizing model in heterogeneous clusters, where diverse hardware resources such as GPUs and CPUs are combined, presents several challenges. Common issues may include workload imbalance, communication overhead, and hardware underutilization, which can severely hinder training performance.

Over the past few years, several frameworks have emerged to tackle such obstacles and accelerate distributed training on multiple nodes. They leverage various optimization strategies, such as decoupled communication, gradient compression, pipelining, and adaptive hyperparameter tuning, to maximize hardware effectiveness and minimize communication bottlenecks.

In this survey, we provide an analysis of four prominent frameworks—BytePS, BytePS-Compress, Espresso, and StellaTrain—that have demonstrated strong performance and scalability in heterogeneous resources. We will discuss their architectural designs, highlight their optimization approaches, and explore their suitability and ease of integration in production-level systems.

Ⅱ. Frameworks

A. BytePS

BytePS [1] is a unified distributed framework designed to speed up DNN training across clusters with heterogeneous hardware components. The BytePS architecture (Figure 1) has two main modules – Summarization Service (SS) and Communication Service (CS). SS performs the aggregation operations on CPUs, while CS handles synchronization and communication between servers. Thus, the system can flexibly utilize CPU resources and network bandwidth, ensuring optimal communication performance across various server cluster configurations.

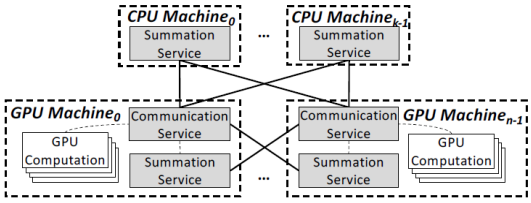

Figure 1. BytePS architecture [1]

In addition, BytePS [2] is open-source, and although its core is implemented in C++ for general-purpose use, it provides user-friendly plugins for major frameworks like TensorFlow, PyTorch, and MXNet, enabling seamless integration with minimal code changes.

B. BytePS-Compress

BytePS-Compress [3] is developed based on BytePS [1] and utilizes a two-stage communication process: the first stage is intra-node communication, which performs All-Reduce [4] gradient operations across GPUs using NVIDIA Collective Communication Library; the second stage is inter-node compression, where gradients are exchanged with Parameter Servers [5] over TCP or Remote Direct Memory Access networks.

A key innovation in BytePS-Compress is the integration of a novel adaptive gradient algorithm called compressed LANS (CLAN), which enables effective gradient compression while preserving the benefits of adaptive optimization methods. By applying

different compression schemes for intra-node and inter-node communication, CLAN aligns well with the underlying hardware characteristics. This approach significantly reduces communication overhead while preserving model accuracy and convergence speed, making the system highly scalable for large-scale distributed training.

C. *Espresso*

Espresso [6] aims to optimize training throughput in compression-enabled DDL by selecting a near-optimal compression strategy from a very large search space using two main techniques. First, the framework introduces a decision tree abstraction to express all possible compression strategies for DDL training jobs, along with empirical models that estimate the time required for tensor computation, communication, and compression. Second, an algorithm is proposed by the authors to select a near-optimal compression strategy based on the previous analysis. This strategy is then applied during training, where the appropriate compression is performed on each tensor as soon as its gradients are ready to be communicated.

The entire process is illustrated in Figure 2, which summarizes how Espresso integrates these techniques into a complete compression-enabled distributed training workflow.

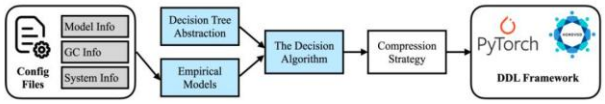

Figure 2. Espresso Overview [6]

D. *StellaTrain*

StellaTrain [7] is the pioneering distributed training framework that holistically speeds up model training in heterogeneous consumer-grade GPU clusters over a WAN. StellaTrain achieves high training speeds through two key techniques. First, it leverages cache-aware gradient compression to optimize network usage in low-bandwidth conditions, alongside a CPU-based sparse optimizer to develop computationally efficient methods for compression and optimization. Second, StellaTrain implements layer-wise partial staleness, where some layers receive immediate gradient updates while others are delayed by one iteration, ensuring synchronized updates with minimal staleness and interleaving gradient transfer with computation.

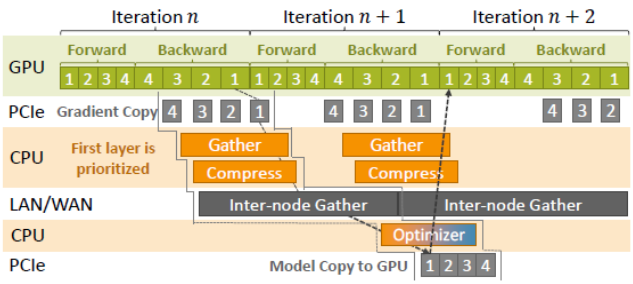

Figure 3. StellaTrain training pipeline [7]

Figure 3 illustrates the StellaTrain training pipeline, in which compression and optimization are strategically offloaded to the CPU, while partial staleness in gradient updates is carefully applied to maximize GPU utilization. However, combining partial staleness with gradient compression introduces new challenges for StellaTrain, as convergence becomes increasingly sensitive to batch size and compression rate under fluctuating WAN bandwidth. To overcome this limitation, StellaTrain dynamically adjusts these hyperparameters using Bayesian Optimization and the Nelder-Mead method. Several experimental results further demonstrate that StellaTrain effectively adapts its training strategy to varying network conditions and surpasses BytePS [1], BytePS-Compress [3], and Espresso [6] in minimizing Time-To-Accuracy.

Ⅲ. Conclusion

This survey examines four frameworks that accelerate DDL across heterogeneous GPU/CPU clusters by using some methods like decoupled communication, gradient compression, pipelining, and adaptive hyperparameter tuning to improve efficiency and training performance. Future research could focus on combining these techniques into hybrid approaches to boost scalability and reduce Time-To-Accuracy in large-scale, dynamic systems.

ACKNOWLEDGMENT

This work was partly supported by Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (RS-2020-II200946, Development of Fast and Automatic Service recovery and Transition software in Hybrid Cloud Environment and RS-2022-II221015, Development of Candidate Element Technology for Intelligent 6G Mobile Core Network)

REFERENCES

[1] Y. Jiang, Y. Zhu, C. Lan, B. Yi, Y. Cui, and C. Guo. A Unified Architecture for Accelerating Distributed DNN Training in Heterogeneous GPU/CPU Clusters. In *Proceedings of USENIX Symposium on Operating Systems Design and Implementation 2020.*

[2] BytePS. https://github.com/bytedance/byteps.

[3] Yuchen Zhong, Cong Xie, Shuai Zheng, and Haibin Lin. Compressed communication for distributed training: Adaptive methods and system. *arXiv preprint arXiv:2105.07829 (2021)*

[4] Pitch Patarasuk and Xin Yuan. Bandwidth Optimal All-reduce Algorithms for Clusters of Workstations. *Journal of Parallel and Distributed Computing, 2009.*

[5] Mu Li, David G Andersen, Jun Woo Park, Alexander J Smola, Amr Ahmed, Vanja Josifovski, James Long, Eugene J Shekita, and Bor-Yiing Su. Scaling Distributed Machine Learning with the Parameter Server. In *OSDI 2014.*

[6] Zhuang Wang, Haibin Lin, Yibo Zhu, and TS Eugene Ng. Hi-Speed DNN Training with Espresso: Unleashing the Full Potential of Gradient Compression with Near-Optimal Usage Strategies. In *Proceedings of the Eighteenth European Conference on Computer Systems 2023.*

[7] H. Lim, J. Ye, S. Abdu Jyothi, and D. Han. Accelerating model training in multi-cluster environments with consumer-grade gpus. In *ACM SIGCOMM 2024.*