Responses to Reviewer Comments

October 22, 2020

We thank the reviewers for their positive comments, acceptance, and agreement with the revised manuscript in the second round of review. Meanwhile, we appreciate that **Reviewer 4** has given some additional suggestions and comments, which are valuable for revising the manuscript and guiding our research work in the future.

With careful studies on all the comments point by point, we have revised the manuscript accordingly. The amendments for the suggestions and comments from the reviewers are highlighted in a different color in the revised manuscript.

In summary, the modifications in the revised manuscript are as follows.

1. We have carefully addressed the ambiguity of the manuscript (e.g., obscure presentation, improper usage of words/phases/symbols, etc.), according to the reviewers' comments and suggestions in the second round of review.

2. We have also polished the original manuscript, making it more readable.

Below are the replies to the comments from the reviewers one by one. The comments are reproduced and our responses (marked in orange) are given directly afterward.

I Reviewer 1

Reviewer Comment: This version has addressed all of my comments, and can be accepted with no change. No further comments.
Author Response: We thank the reviewer for accepting and agreeing to our changes and extensions of the manuscript. We are pleased to hear that our revised version addresses all raised concerns and would like to thank again the reviewer for the constructive feedback.

II Reviewer 2

Reviewer Comment: The revision addressed my concerns.
Author Response: We thank the reviewer for accepting and agreeing to our changes and extensions of the manuscript. We are pleased to hear that our revised version addresses all raised concerns and would like to thank again the reviewer for the constructive feedback.

III Reviewer 3

Reviewer Comment: Thanks for the authors to address all my comments on the first round.

Author Response: We thank the reviewer for accepting and agreeing to our changes and extensions of the manuscript. We are pleased to hear that our revised version addresses all raised concerns and would like to thank again the reviewer for the constructive feedback.

IV Reviewer 4

Thank you for reviewing the manuscript with detailed and valuable feedback. We are happy to know that we have addressed most of the issues raised in the previous reviews. Meanwhile, we appreciate the new comments and suggestions from the reviewer, which are constructive and enlightening. The modifications are marked in red in the revised manuscript. Below are our replies to specific points.

1: Reviewer Comment: The current paper structure is very tight. Some redundant content could be removed and referenced to the conference version.

Author Response: Thank you for pointing out the paper structure problem and the suggestion. We explain the problem as follows:

- According to the policy of IEEE TPDS and the special section, submissions (including the revised manuscripts) should be no longer than 12 pages.
- To make the journal paper self-contained, we have kept the background studies and technical design details in the previous submissions, which we think are helpful to impress the readers.
- We have a long list of authors in the manuscript, and the author biography has occupied much space, which is also counted into the page length.

To present the paper in detail while keeping the paper length no more than 12 pages, we have compressed the paper structure in the previous submissions.

We have made the revisions and hope to address the problem as follows:

1. We have reviewed the previous version and delete some content, which we think is less important. Now, the technical content part of the paper has a more readable structure, whereas the author biography still looks a little tight.
2. We notice that, when a paper is accepted, the authors have the chance to apply for extra pages (with overlength page charges[1]) to prepare the final version of the paper. We are still trying to keep the journal paper self-contained, and we would like to apply for another page (with overlength page charges) to fully relax the paper structure in the final version once we get the paper accepted.

2: Reviewer Comment: In the introduction, the authors mentioned some new challenges when "burning a volume of CPU/GPU cores at runtime" (marked in red). This reviewer is wondering how to solve these challenges? It would be nice if the authors can give a brief solution description.

Author Response: Thank you for pointing out the obscure presentation in the introduction (Para. 3, Page 1), where we would like to share with readers our studies on the shortcomings of online data preprocessing backends in DL systems (i.e., burning too many CPU/GPU cores at runtime). We have corrected the improper sentence "it is also facing the following challenges" as "it also has the following shortcomings" in the revised manuscript (marked in red, refer to Para. 3, Section 1, Pages 1-2) and would like to draw the reviewer's attention to the revised narrative logic as follows.

[1] https://www.computer.org/digital-library/journals/td/call-for-papers-special-section-on-parallel-and-distributed-computing-techniques-for-ai-ml-and-dl

1. We study the limitations of existing data preprocessing backends with practical experiments (Para. 2, Section 1, Page 1) and show the performance bottleneck of data preprocessing in DL workflows with comprehensive analysis (Para. 3, Section 1, Pages 1-2).
2. Meanwhile, we have noticed and evaluated the potential of FPGAs to accelerate cutting-edge applications (Para. 4, Section 1, Page 2).
3. Based on the above observations, we propose DLBooster in this work to redesign a high-performance and efficient data preprocessing backend with FPGAs to speed up end-to-end DL workflows.
4. Finally, we have summarized the closely related works on topic of data preprocessing for DL and shown the differences by comparing them with DLBooster to impress readers (marked in red, refer to Para. 1, Section 8, Pages 10-11).

As for the challenges of "burning a volume of CPU/GPU cores at runtime", we would like to share our investigations/studies with the reviewer as follows.

In addition to data preprocessing, many other workloads in DL also put heavy demands on computing resources. Particularly, the evolution of computer systems, including but not limited to computing hardware, cache/storage subsystem, communication technologies, has become even more complex and brought about revolutionary transformations in many aspects. According to our previous studies, there are a couple of ideas (listed below) to work on, to build economic and efficient deep learning systems.

- **Domain-specific accelerators**: As general computing units, CPUs/GPUs have been optimized for a variety of tasks. Although they can achieve the satisfying processing performance in most computing tasks and applications, they are usually not the best choices. To this end, many domain-specific accelerators are being designed and applied to some emerging tasks. For example, the Data Processing Unit (DPU), which is competitive to as many as 128 CPU cores in terms of data processing performance, is recently announced by NVIDIA (GTC'20) [1] to be applied to emerging workloads in data centers. In Azure, researchers are considering offloading some key networking logic to the FPGA-based SmartNIC (NSDI'18) [2] to provide high-throughput, ultralow-latency networking services while avoiding burning the CPU in the cloud.
- **Improving the computing efficiency**: Some researchers are exploring more advanced algorithms and scheduling schemes to improve the computing efficiency, thus boosting the performance of applications. Usually, the runtime execution of computing tasks can hardly approach the peak performance of hardware. Particularly in a complex system, the unreasonable pipeline, overlapping, and sub-task scheduling would finally degrade the overall performance, leaving opportunities to further improve the utilization of hardware computing resources. For example, TVM (OSDI'18) [3] tries to explore more reasonable scheduling strategies to better utilize the computing resources when running DL tasks, according to the specifications of hardware devices. When designing the FPGA decoder of DLBooster, we have also tried to seek the optimizations (e.g., submitting decoding tasks asynchronously, scaling the sub-decoding units to different number of instances, etc.) to pipeline each sub-components, improving the overall decoding performance.
- **Optimizing the deployment and architecture of systems**: The architecture and deployment also have significant impacts on the overall performance of a system. When deploying a system in practice, we may need to evaluate how many resources should be used, whether it is possible/cost-effective to apply for more resources to get higher performance, how to deploy the task on given resources, etc. For example, BytePS (SOSP'19, OSDI'20) [4–6] chooses to add more CPU-only servers in distributed ML training. They argue that the investment for the extra CPU-only servers is cost-effective: The limited extra fees for the CPU-only servers can double the available bandwidth and leverage more CPU cores to aggregate gradients in DML training.

Including but not limited to the above, we may need to consider more when designing the system, and there are many challenging yet important research topics to be explored in the area of DML systems, along with the evolution of computer systems.

3: Reviewer Comment: The authors used many tilde symbols (∼). Please check if it is really necessary. Sentences like this "There are only ∼1.5 cores consumed by DLBooster when training all three DNNs, whereas LMDB consumes ∼2.5 cores in the training. (2) The CPU-based backend consumes much more CPU cores: each GPU requires ∼ 12 cores and ∼ 7 cores " looks a little wired.

Author Response: Thank you for pointing it out and the suggestion, and we have checked the manuscript and corrected the improper usage of symbols.

To avoid repeated patterns, we have changed the sentence to "(1) There are around 1.5 cores consumed by DLBooster when training all three DNNs, whereas LMDB consumes around 2.5 cores in the training. (2) The CPU-based backend consumes much more CPU cores: each GPU requires about 12 cores and 7 cores in the training of AlexNet and ResNet-18 respectively." (Section 6.2.2, Pages 8-9)

We have carefully studied the usage of symbols and revised the improper ones in the manuscript accordingly. However, they are not listed one by one here in avoidance of overwhelming the letter.

References

[1] NVIDIA GPU Technology Conference Keynote Oct 2020 — Part 5. Data center infrastructure-on-a-chip. `https://www.nvidia.com/en-us/gtc/keynote/?video=5`. Last accessed: October, 2020.

[2] Daniel Firestone, Andrew Putnam, Sambhrama Mundkur, Derek Chiou, Alireza Dabagh, Mike Andrewartha, Hari Angepat, Vivek Bhanu, Adrian Caulfield, Eric Chung, et al. Azure accelerated networking: Smartnics in the public cloud. In *15th USENIX Symposium on Networked Systems Design and Implementation (NSDI 18)*, pages 51–66, 2018.

[3] Tianqi Chen, Thierry Moreau, Ziheng Jiang, Lianmin Zheng, Eddie Yan, Haichen Shen, Meghan Cowan, Leyuan Wang, Yuwei Hu, Luis Ceze, et al. TVM: An automated end-to-end optimizing compiler for deep learning. In *13th USENIX Symposium on Operating Systems Design and Implementation (OSDI 18)*, pages 578–594, 2018.

[4] Github Open-Source Project. BytePS: a high performance and generic framework for distributed dnn training. `https://github.com/bytedance/byteps`. Last accessed: October, 2020.

[5] Yanghua Peng, Yibo Zhu, Yangrui Chen, Yixin Bao, Bairen Yi, Chang Lan, Chuan Wu, and Chuanxiong Guo. A generic communication scheduler for distributed dnn training acceleration. In *Proceedings of the 27th ACM Symposium on Operating Systems Principles*, pages 16–29, 2019.

[6] Yimin Jiang, Yibo Zhu, Chang Lan, Bairen Yi, Yong Cui, and Chuanxiong Guo. A unified architecture for accelerating distributed DNN training in heterogeneous gpu/cpu clusters. In *14th USENIX Symposium on Operating Systems Design and Implementation (OSDI 20)*. USENIX Association, November 2020. To appear soon.