Characterizing I/O in Machine Learning with MLPerf Storage

Oana Balmau
McGill University

Data is the driving force behind machine learning (ML) algorithms. The way we ingest, store, and serve data can impact the performance of end-to-end training and inference significantly [11]. However, efficient storage and pre-processing of training data has received far less focus in ML compared to efforts in building specialized software frameworks and hardware accelerators. The amount of data that we produce is growing exponentially, making it expensive and difficult to keep entire training datasets in main memory. Increasingly, ML algorithms will need to access data from persistent storage in an efficient way.

To address this challenge, this work sets out to characterize I/O patterns in ML, *with a focus on data preprocessing and training*. Our goal is to create the first open-access storage-focused benchmark for ML.

Why a new benchmark? An extensive body of work proposes benchmarks for ML algorithms (e.g., MLPerf [8], OpenAI Gym [6], Deepmind Lab [5], DawnBench [7]). While these benchmarks provide a valuable end-to-end test of an ML environment, they make it difficult to isolate the value of each component. Moreover, existing benchmarks tend to focus on the *compute* required for training and inference. As a consequence, the storage setup is simplified and the cost of data pre-processing is largely ignored. Finally, prior benchmarks have a high barrier to entry for non-ML practitioners, requiring expensive accelerators (e.g., GPUs, TPUs, AWS Inferentia) and extensive ML-specific knowledge to run. All these reasons justify the need for a storage-focused benchmark for ML that is easy to deploy and is accelerator-agnostic.

Approach. We are exploring trace collection to understand storage impact in ML, similar to the SPEC-storage benchmark [2]. Key factors we are investigating include the workload type, software framework used (e.g., PyTorch [10], Tensorflow [3, 9]), accelerator type, dataset size to memory ratio, and degree of parallelism. The trace collection is done through eBPF [1] and system monitoring tools such as `mpstat`, and NVIDIA Nsight. Our traces include VFS-layer calls such as `read`, `write`, `open`, as well as `mmap` calls, block I/O accesses, CPU, memory, and accelerator use. We are collecting traces for workloads with different I/O profiles, based on the MLPerf Training [8] benchmark reference implementations and datasets. Specifically, we are focusing on computer vision, natural language processing, and recommender systems workloads, collecting traces during the training phase. The current system is single-node, with the data residing in local storage. In the second stage of the work, we intend to switch the focus to data cleaning and pre-processing for these three workload types. Finally, we intend to expand the work to a multi-node setup.

To account for different memory to dataset size ratios we will scale up the datasets (e.g., through data replication and adding noise), and we will limit the main memory size (e.g., through `cgroups`). Note that, while still required as the base criteria to generate faithful traces, the data quality and accuracy of the trained models are less relevant to our work, as we only focus on understanding I/O patterns at different stages of the workload.

Based on the trace analysis, we will build a synthetic I/O workload generator. The workload generator will accurately reproduce I/O patterns for representative ML workloads. We make it a central design point for the workload generator to be user-friendly to non-ML researchers and practitioners. In particular, we take inspiration from the `fio` [4] interface, which is familiar to storage researchers. Finally, one of our design goals is to enable users to run the workload generator without having to use ML accelerators. One possible way to achieve this is artificially introducing delays between the I/O calls, to simulate the accelerator compute time.

Impact. A key question this work will help answer is how to provision a balanced training cluster that is not bottlenecked on storage or compute for a complex mix of workloads. Furthermore, a storage-focused benchmark will accelerate the research and development of specialized storage systems for ML. Finally, the detailed trace analysis will uncover compelling research directions at the intersection of storage and ML.

REFERENCES

[1] eBPF. ebpf.io/.

[2] SPECstorage Solution. spec.org/storage2020/.

[3] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, and M. Kudlur. TensorFlow: A system for Large-Scale machine learning. In *Proceedings of the 12th USENIX symposium on operating systems design and implementation (OSDI), pp. 265-283*, 2016.

[4] J. Axboe. fio. github.com/axboe/fio.

[5] C. Beattie, J. Z. Leibo, D. Teplyashin, T. Ward, M. Wainwright, H. Küttler, A. Lefrancq, S. Green, V. Valdés, A. Sadik, and J. Schrittwieser. Deepmind Lab. *arXiv preprint arXiv:1612.03801*, 2016.

[6] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba. OpenAI Gym. *arXiv preprint arXiv:1606.01540*, 2016.

[7] C. Coleman, D. Narayanan, D. Kang, T. Zhao, J. Zhang, L. Nardi, P. Bailis, K. Olukotun, C. Ré, and M. Zaharia. Dawnbench: An End-to-end Deep Learning Benchmark and Competition. *Proceedings of Training, 100(101), p.102*, 2017.

[8] P. Mattson, C. Cheng, G. Diamos, C. Coleman, P. Micikevicius, D. Patterson, H. Tang, G.-Y. Wei, P. Bailis, V. Bittorf, D. Brooks, D. Chen, D. Dutta, U. Gupta, K. Hazelwood, A. Hock, X. Huang, D. Kang, D. Kanter, N. Kumar, J. Liao, D. Narayanan, T. Oguntebi, G. Pekhimenko, L. Pentecost, V. Janapa Reddi, T. Robie, T. St John, C.-J. Wu, L. Xu, C. Young, and M. Zaharia. MLPerf Training Benchmark. In *Proceedings of Machine Learning and Systems (MLSys), 2, pp. 336-349*, 2020.

[9] D. Murray, J. Simsa, A. Klimovic, and I. Indyk. tf.data: A Machine Learning Data Processing Framework. *Proceedings of the VLDB Endowment 14(12), 2945-2958*, 2021.

[10] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, and A. Desmaison. Pytorch: An Imperative Style, High-performance Deep Learning Library. *Proceedings of Advances in neural information processing systems (NeurIPS), 32*, 2019.

[11] M. Zhao, N. Agarwal, A. Basant, B. Gedik, S. Pan, M. Ozdal, R. Komuravelli, J. Pan, T. Bao, H. Lu, S. Narayanan, J. Langman, K. Wilfong, H. Rastogi, C.-J. Wu, C. Kozyrakis, and P. Pol. Understanding Data Storage and Ingestion for Large-Scale Deep Recommendation Model Training. In *Proceedings of International Symposium on Computer Architecture (ISCA)*, 2022.