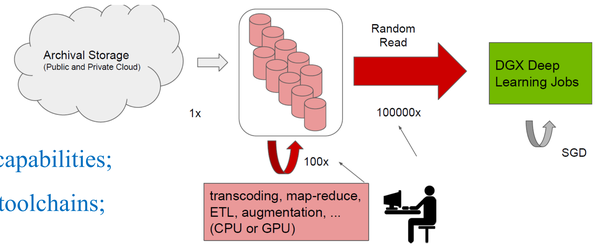
High-performance I/O for large-scale deep learning

Alex Aizman, Gavin Maltby, Thomas Breuel
NVIDIA, USA

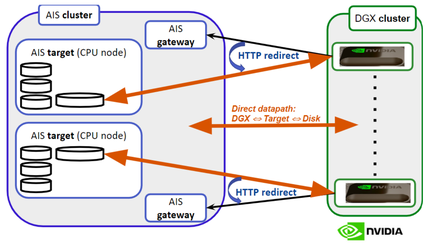
Motivation

- Training deep learning (DL) models on **petascale datasets** is essential for achieving state-of-the-art performance.
- DL jobs iterate repeatedly through random permutations of training datasets.
- Overall, the requirements to storage include:
  - ✓ use of standard protocols and formats;
  - ✓ easy migration of existing DL models and datasets;
  - ✓ scalability to run at speeds close to combined hardware capabilities;
  - ✓ easy setup, predictable performance; compatibility with toolchains;
  - ✓ easy integration with Kubernetes
  - ✓ support of MapReduce style preprocessing (e.g., parallel resharding).



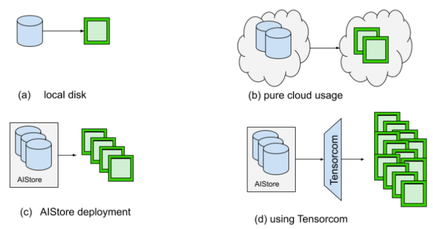
AIStore: https://github.com/NVIDIA/aistore

- Is fully open-sourced, currently at version 2.6;
- Runs on any commodity hardware;
- Includes easy-to-use CLI;
- Scales-out linearly with no downtime and no limitations;
- Supports S3-like HTTP REST API;
- Can be mounted as a Linux filesystem, supports access to AIS objects as files;
- Natively integrates with Amazon S3 and Google Cloud Storage (GCS), can be deployed as a fast tier for GCS and S3;
- Can be populated from Cloud on-demand and/or via *prefetch* and *download* APIs;
- Can be used as a standalone highly-available protected storage;
- Auto-rebalances upon changes in cluster membership, drive failures, bucket renames;
- Supports n-way mirroring (RAID-1), RS erasure coding, end-to-end data protection;
- Includes MapReduce extension for massively parallel resharding of very large datasets;
- Features open format and freedom to copy/move data at any time using familiar tools.



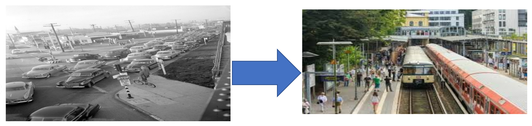
WebDataset: https://github.com/tmbdev/webdataset

- Is based on the standard PyTorch 1.2 IterableDataset;
- Supports datasets comprised of tar files (aka shards);
- Enables archival representation of the original training data.



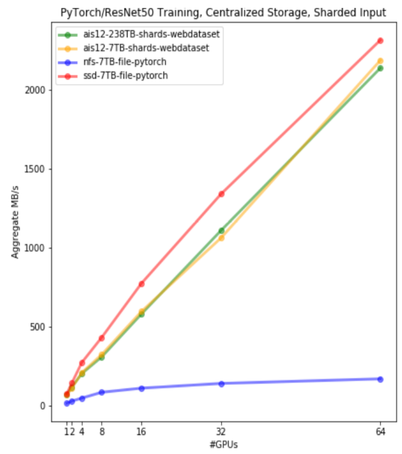
(RE)sharding

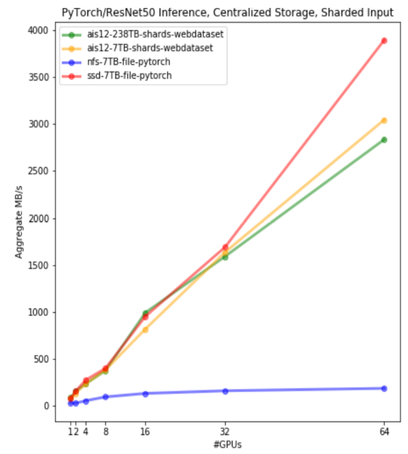
- The problem of *small files* (e.g., *in Hadoop*) is well known.
- AIStore provides a module (called dSort) to reshard datasets in parallel for a given (customizable) sorting order and output size
- While WebDataset can read from any input streams containing tar shards.



Linear scale under DL workloads

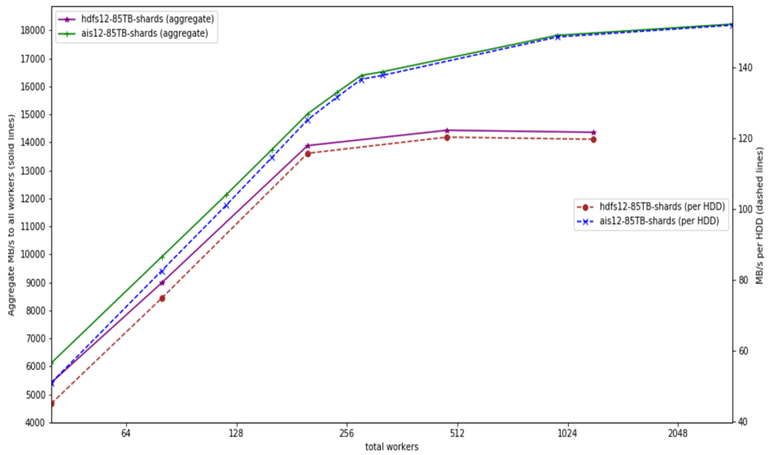
- We compared DL performance using PyTorch-based ResNet-50 models and different storage backends.
- All datasets were derived from ImageNet by uniformly duplicating existing samples under randomly generated file names.





Maximum data delivery rate

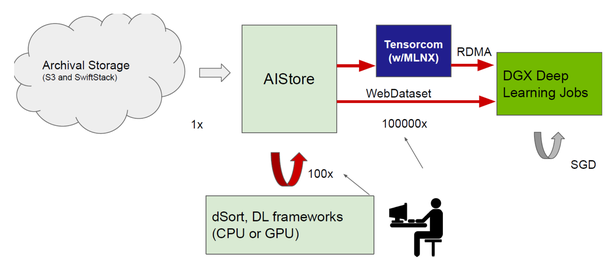
- How many deep learning clients can a given storage cluster configuration support at a reasonable data rate?
- The benchmark runs on an 85TB dataset comprising 68,000 x 1.25GB input shards, each shard containing, on average, 8,600 input images for a total of 588 million images.
- HDFS utilizes its default 128MB block size while AIS is configured to store entire shards on the same 120 HDDs (of 12 clustered nodes).
- Both HDFS and AIS maintain 3 replicas of each of the 68K shards.
- A 40-worker load is run on a single GPU node, then 80 workers on 2 nodes, and so on up 280 workers over 7 compute nodes and a final additional result of 8 nodes with 120 and with 360 workers each.
- The benchmark selects input shards at random, reads an entire shard, and discards the read data



AIStore delivers 18GB/s aggregated throughput, or 150MB/s per each of the 120 hard drives – effectively, a hardware-imposed limit. HDFS, on the other hand, falls below AIS, with the gap widening as the number of DataLoader workers grows.

Summary and next steps

- Existing distributed filesystems (including Google's GFS and Hadoop's HDFS) do not satisfy the performance and usability requirements of deep learning.
- We have developed an open-standards (HTTP, POSIX) based solution.
- AIStore scales linearly with each added drive, is DL-friendly, supports easy ad-hoc setup and configuration, and performs under training and MapReduce type workloads.
- Next, we'll be adding support for RDMA and TensorFlow, integration with Nvidia DALI, and more.



NVIDIA