Enabling *Storage for AI* Benchmarking through MLPerf Storage

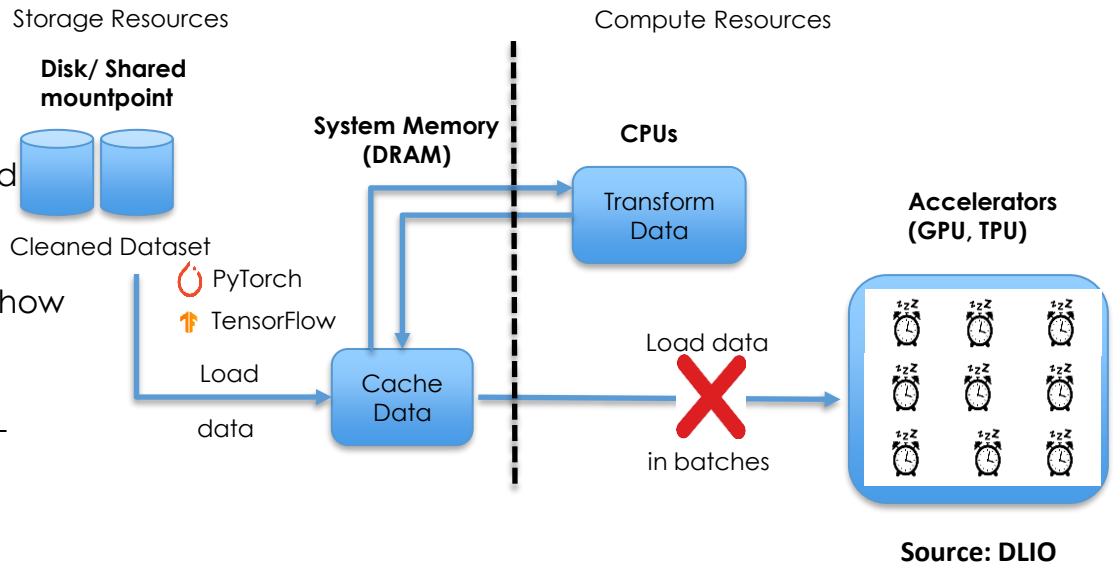
Sept 24th, 2025

Storage for AI

- **Storage-compute gap**
  - 30-50%
    - Typical GPU Utilization
    - GPUs idle waiting for data
  - TB → PBs
    - Dataset growth
    - Exponential data expansion
  - $30-50K
    - Per GPU cost
    - Wasted on idle GPU cycles
  - Training: 2-4x, Inference: 10-30x
    - H100 vs A100 Speed
    - Storage must keep pace

MLPerf Storage Benchmark

- Evolution of SSD Performance Evaluation:
  - Traditional focus:
    - Sequential read/ write
    - Random read/write
- Diverse storage demands introduced by AI and GPU-based workloads
- Standardized benchmark suite that evaluates how efficiently storage systems can supply AI/ML pipeline training data efficiently to GPUs
  - Replay operations to storage from CPU – avoid large GPU clusters
  - Understand storage bottlenecks in AI workloads and propose optimizations
  - Current: Designed to evaluate systems with NVIDIA A100 and H100 GPUs

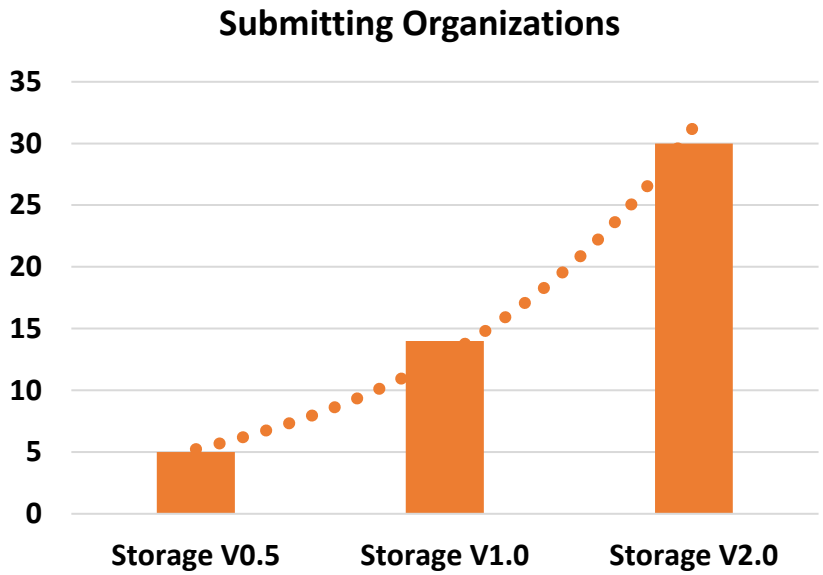


Source: DLIO

AI Workloads for Storage Benchmarking: V2.0

Uses AI Industry tools and frameworks to demonstrate Storage subsystems capabilities

1. **Training Benchmarks**: Data ingest for 3 models emulating A100 and H100 accelerators → UNET3D, ResNet50, CosmoFlow
2. **Checkpointing:** 4 llama3 models → 8B, 70B, 405B, 1T
   - Model loading time
   - Checkpoint save latency and recovery time
3. **[PREVIEW] VectorDB for RAG Workload:**
   - Milvus DB, DiskANN support
   - Throughput QPS
   - Latency



Training Workloads

UNET3D:

- Bandwidth-intensive test → Opens large files in small batches and reads them sequentially

ResNet50:

- Concurrently reading many small files within a large number of larger files
- Consists of smaller and more random I/O

CosmoFlow:

- Larger sample size compared to ResNet50, single sample per file

Model (Area)	Dataset Seed	Data Format and Reader	Minimum Accelerator Utilization Requirement	H100 Computation time (s)
UNET3D (Vision: Medical Image Segmentation)	KiTS 19 (143 MB/sample)	1 sample per file NumPy .npz PyTorch Reader	90%	0.323
ResNet50 (Vision: Image Classification)	ImageNet (150 KB/sample)	10,000 samples per file TFRecord TensorFlow → >100MB file	90%	0.224
CosmoFlow (Scientific: Cosmology parameter prediction)	CosmoFlow N-body simulation (2 MB/sample)	1 sample per file TFRecord TensorFlow	70%	0.0035

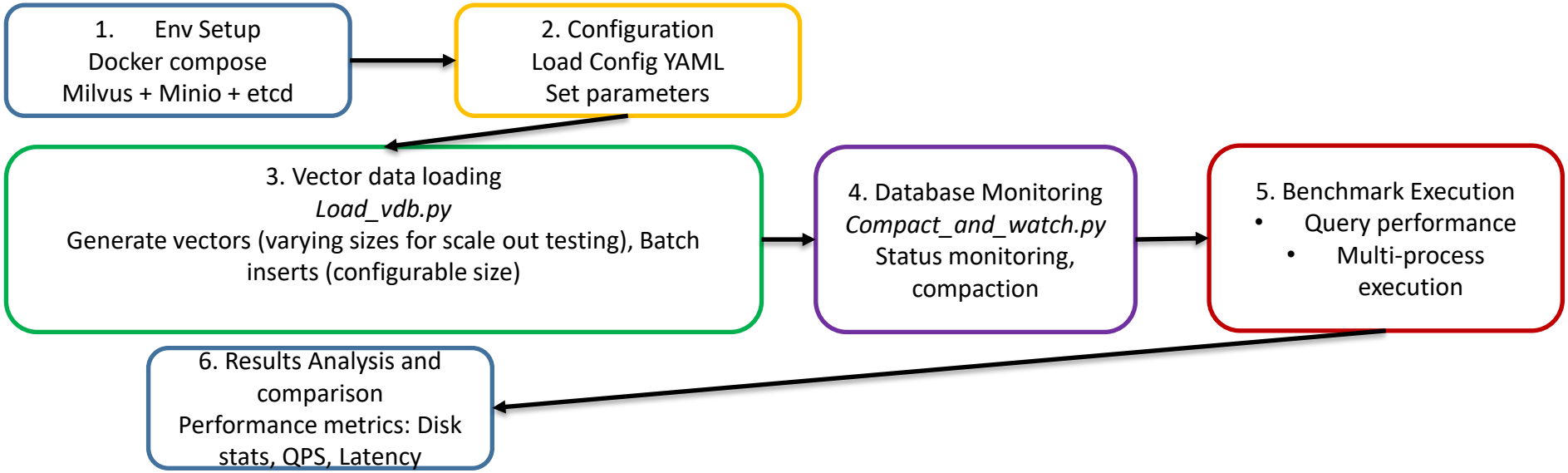
Checkpointing I/O: Supported models

Underlying I/O generator supports most of transformer models of different sizes, by adjusting following parameters:

- model.num_layers,
- model.transformer.hidden_size,
- model.transformer.ffn_hidden_size
- model.transformer.vocab.size

	8B	70B	405B	1T
Hidden dimension	4096	8,192	16,384	25,872
FFN size	14336	28,672	53,248	98,304
num_attention_heads	32	128	128	192
num_kv_heads	8	8	8	32
Num layers	32	80	126	128
Parallelism	1x1x 8(DP)	8x8(DP)	8x32x2(DP)	8x64x2(DP)
Num_GPUs	8	64	576	1024
ZeRO	3	3	1	1
Checkpoint size	105 GB	912 GB	5.29 TB	18 TB

VectorDB Benchmark Workflow



Key Configuration Parameters

Dataset Settings

- Vectors: Variable size (500K, 5M, 10M, 100M ..)
- Dimensions: Configurable (1536 default)
- Distribution: uniform/ normal

Database and Index Settings

- Type: DiskANN default (configurable)
- Metric: COSINE
- Connections (M) and Expansion factor (ef)
- Collection to create

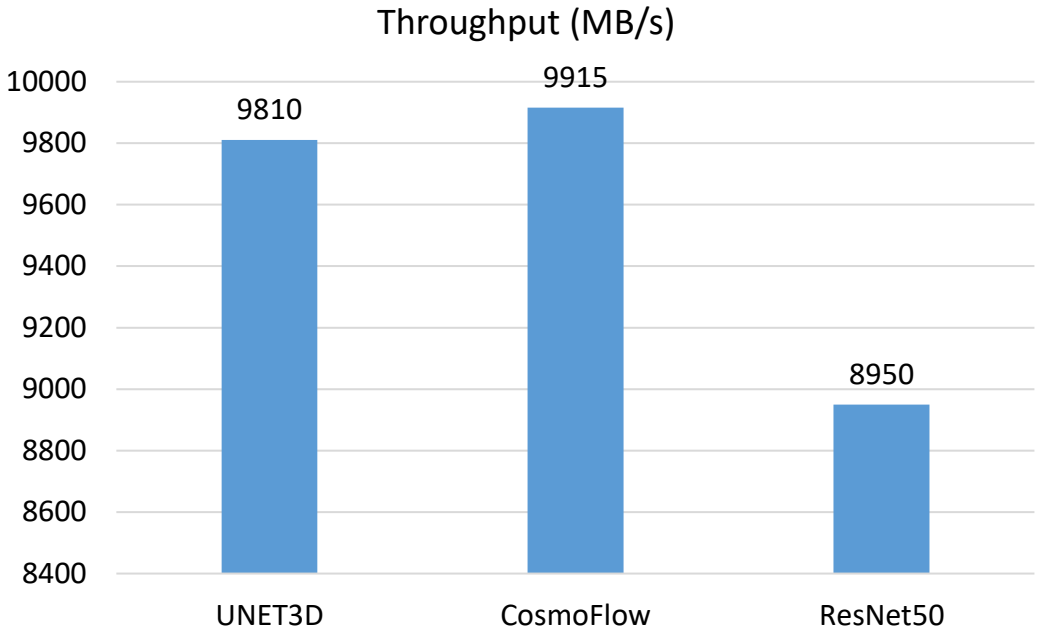
Benchmark Settings:

- Processes
- Batch size
- Runtime
- Chunk size for memory management

Training Workloads Analysis

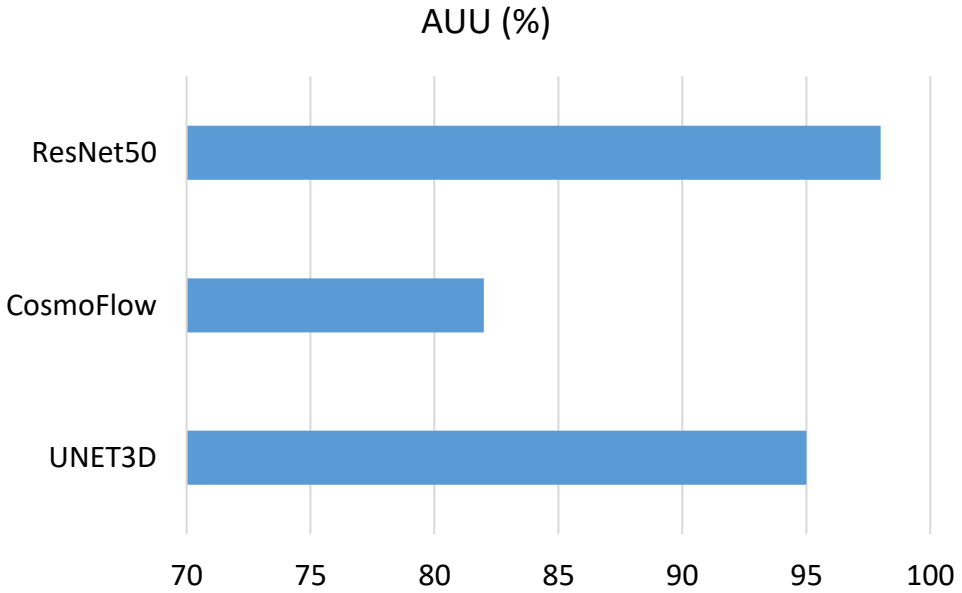
Training Throughput

- Maximum Gen5 SSD throughput by workload (averaged across 5 epochs of training)
- Readers threads:
  - UNET3D: 16
  - CosmoFlow: 8
  - ResNet50: 8
- H100s accelerators:
  - UNET3D: 4
  - CosmoFlow: 14
  - ResNet50: 54

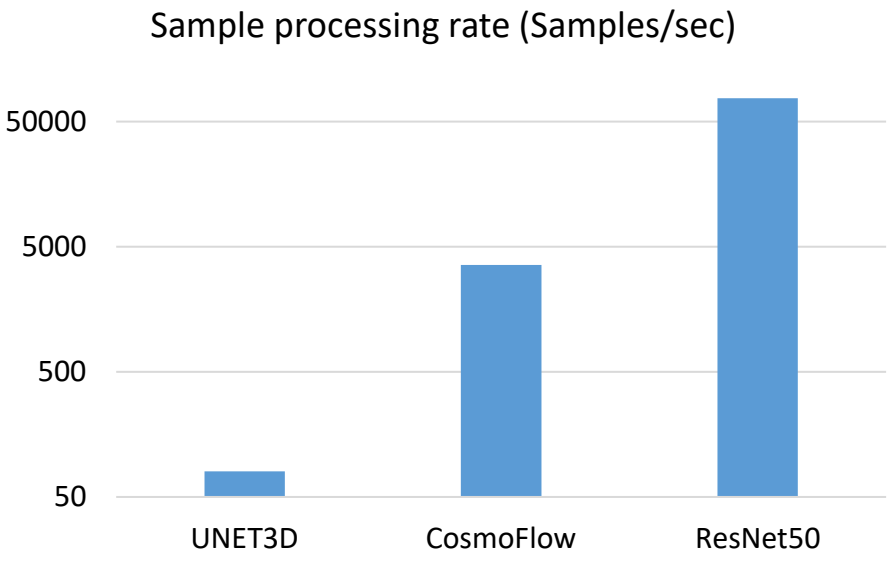


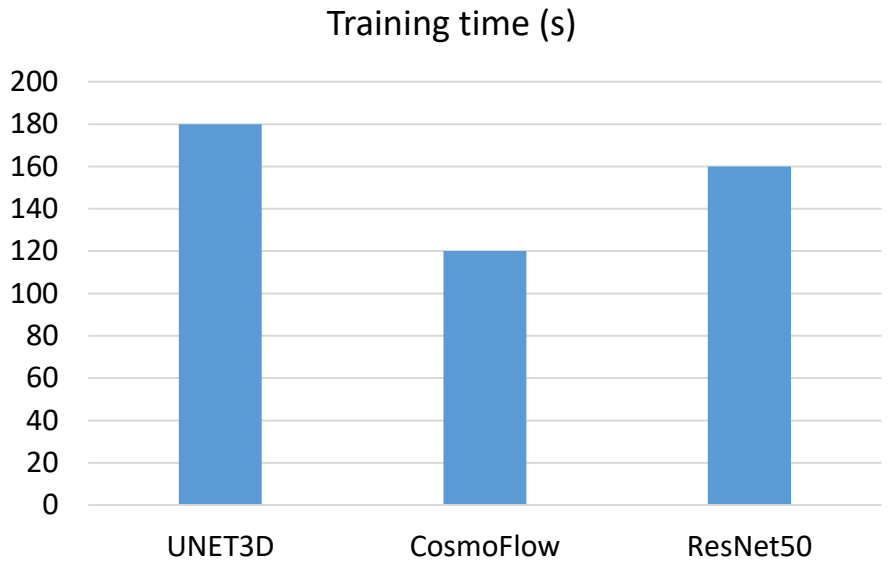
Accelerator Utilization (AUU%)

- AU per workload when the number of accelerators is fixed
- Key Observations:
  - Due to SSD's load handling capability, increasing accelerators lead to inefficiencies → Longer overall execution times
  - Real-world:
    - In Multi-gpu environment, a higher AU → Shorter training time → Reduced GPU Idle time



Sample Processing rate and Execution time

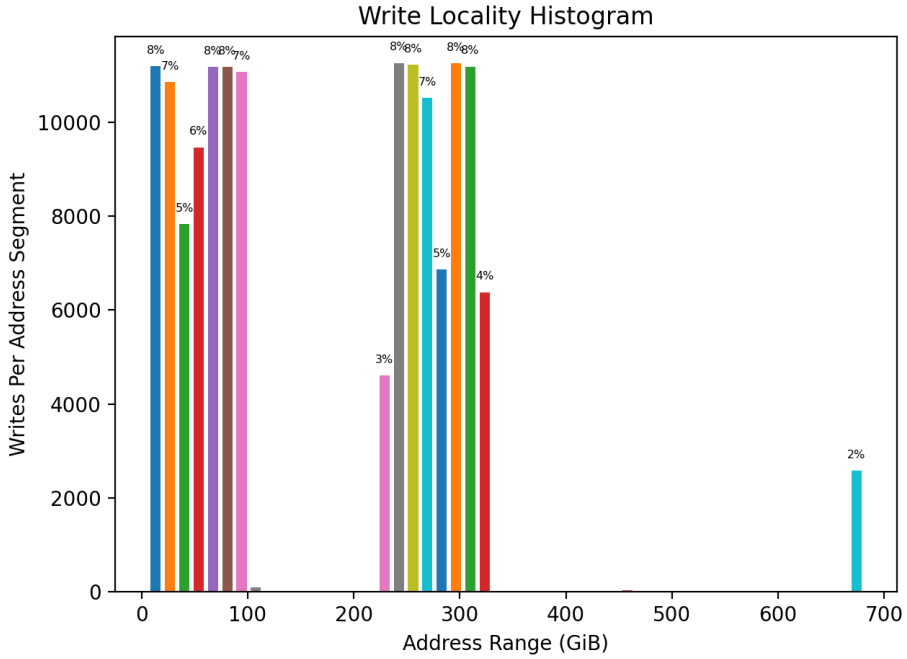


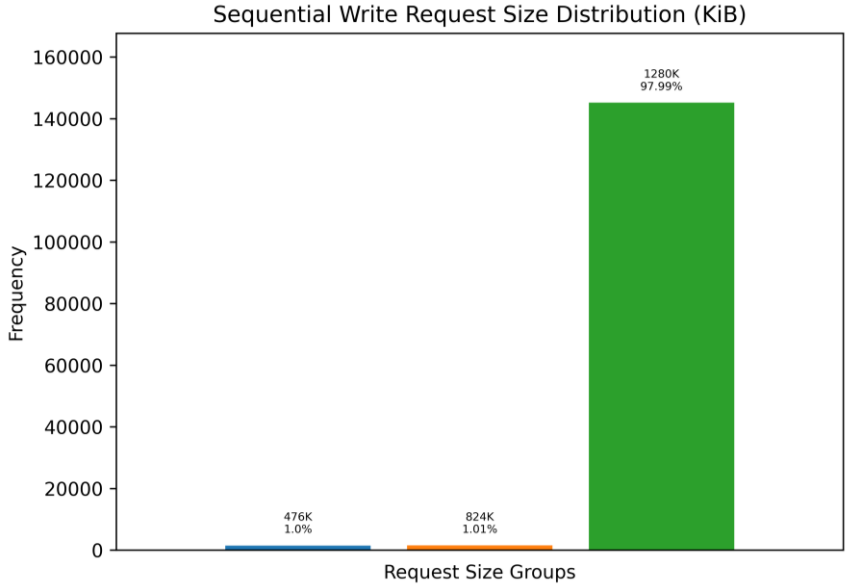


Checkpointing Analysis

Checkpointing Characterization at SSD-level

Model	Load time (s)	Save time (s)	Load throughput (GB/a)	Save Throughput (GB/s)
llama-8B	9.69	8.78	10.82	11.97
llama-70B	10.29	9.31	11.07	12.38

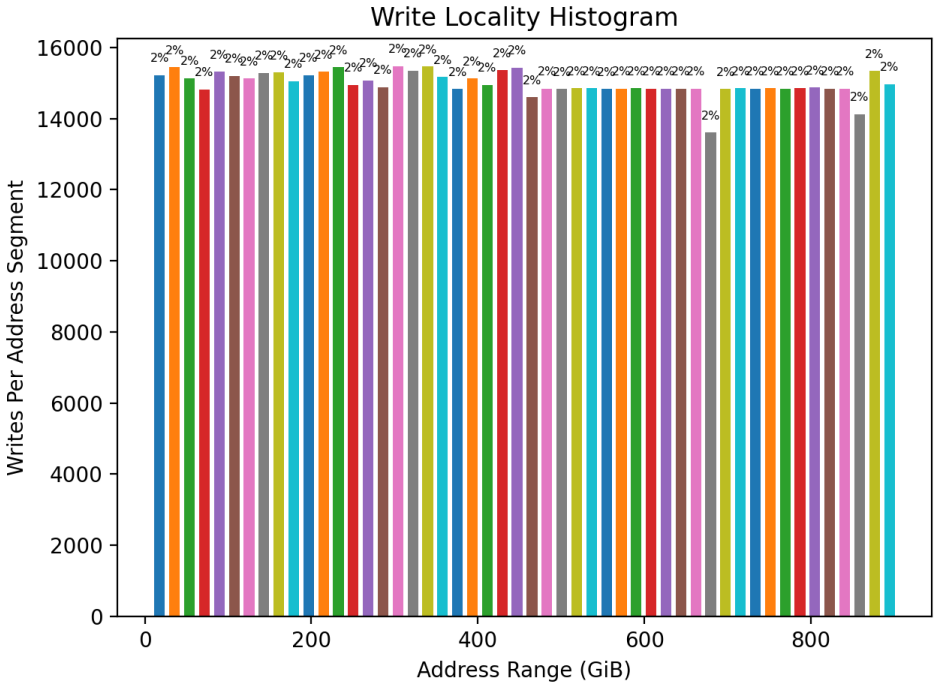


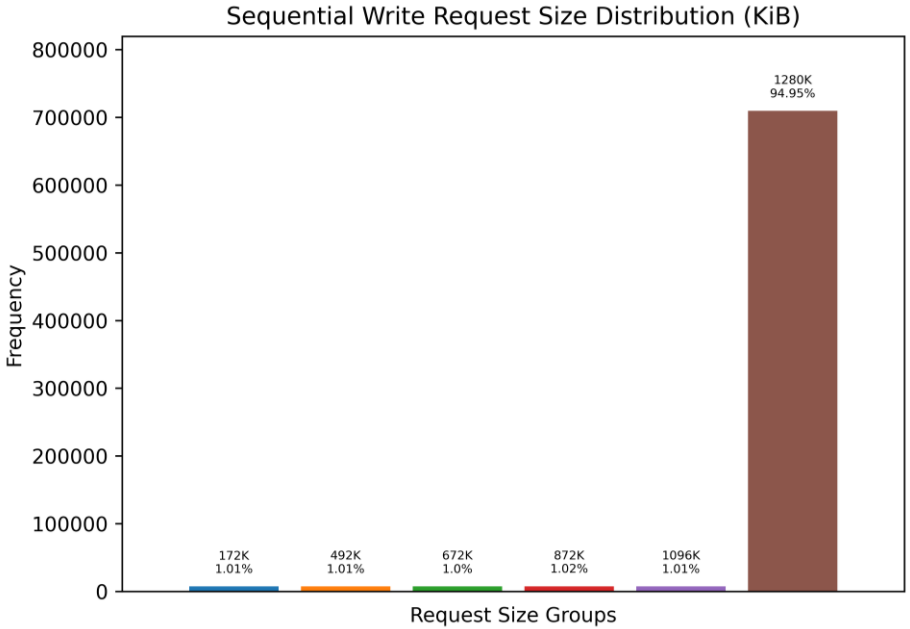


- Llama3- 8B model checkpointing
  - Total I/O Count: 148732
  - Sequential I/Os: 148195
  - Sequential percentage: ~99.7
  - Mean & Median request size: ~1MB

Llama3-70B Checkpointing I/O

- Total I/O Count: 749377
- Sequential I/Os: ~99.8%





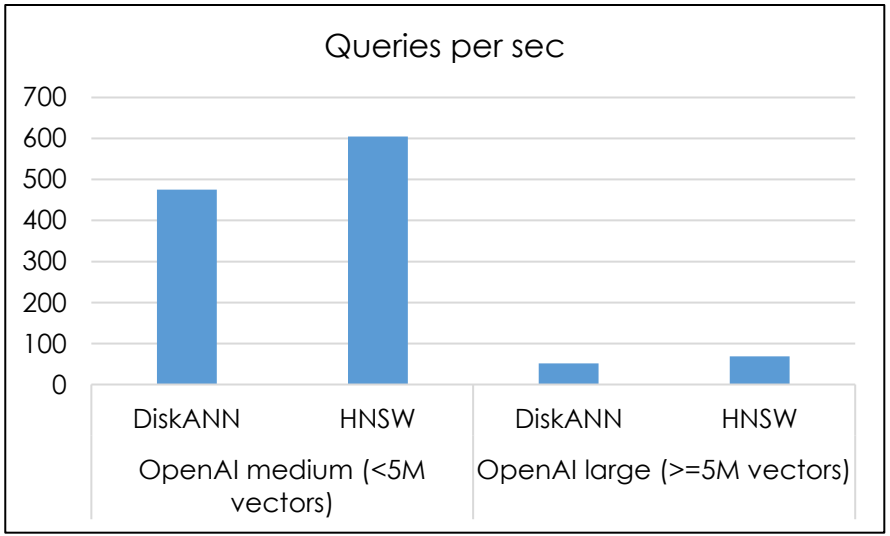
Mean & Median request size: ~1MB

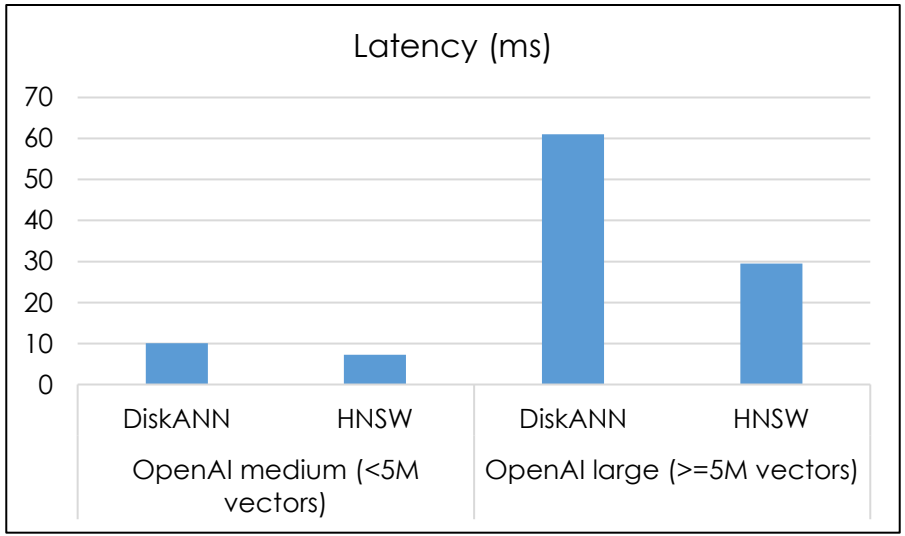
Initial VectorDB Analysis

Milvus Performance Results

- Throughput in QPS
- P99 latency
- Recall rate

Dataset	Index	Queries per sec	Latency (ms)	Recall rate
OpenAI medium (500K vectors)	DiskANN	475	10.1	0.965
	HNSW	605	7.3	0.971
OpenAI large (5M vectors)	DiskANN	52	61	0.988
	HNSW	69	29.5	0.989





Milvus VectorDB I/O Analysis

- Storage workload characteristics
  - Data ingestion and post-insert phase involved mix of reads and writes (predominantly writes)
  - Query phase mainly involved random read operations: I/O size of 8K
- Milvus exhibits an OLTP-like workload with distinct phases (ingest, optimization, query)
- Index impact
  - DiskANN → higher storage I/O
  - HNSW → memory-bound

Workload Phase	Metric	Value
Data Ingestion	Workload	Mixed read-write, Writes predominant
	IO Size	128K
Query	Workload	Reads
	IO Size	8K

MLPerf Storage V3 Features Development

Call to action

- **VDB-Bench**
  - Multi-client and vectorDB multi-node support
  - Enhance I/O profiling for scale-out dataset
  - Dynamic-indexing support
  - GPU-enhanced indexing and vector search
- **S3 for AI**
  - Establish S3 as a viable storage choice for AI workloads
    - AI/ML Workload driver – DLIO
      - MLPerf workload driver with S3 support
    - AI/ML I/O Library – s3DLIO
      - S3 I/O operations
      - S3 I/O tracing/ capture
    - S3 Test tool – warp-replay
      - S3 I/O replay of trace/ capture
      - S3 performance testing

Thank you!