Benchmarking Storage with AI Workloads

Presented by
Devasena Inupakutika, Charles Lofton, Bridget Davis
Samsung Semiconductor Inc.
Motivation

- Growing production datasets: 10s, 100s of petabytes
- Samsung’s datacenter storage and memory products
- Research involving the impact of storage on AI/ML pipelines is limited
- How to showcase Samsung datacenter product’s impact to real world workloads?
Introduction

- Benchmarking essential to evaluating storage systems:
  - Storage needs for large machine learning datasets are growing
- Evaluating storage for AI workloads is challenging
  - Real-world AI training requires specialized hardware
  - System resources stressed by AI application
- Do AI workloads benefit from high performance storage systems?
- Is there a realistic method to showcase high performance storage for AI workloads?
- Can the test methods be easily implemented and reproducible?
Introduction

- Benchmark datasets are smaller whereas data is the moving force of AI algorithms
- Real-world production workloads demands huge data (both for training and generation during streaming)
- Empirical study to understand how AI workloads utilize storage devices through I/O patterns
AI Workloads I/O Characterization

- Better understanding of AI I/O profiles
- Provides insights on the design and configuration of storage systems
- Main aspects under consideration:
  - I/O Rates
  - Throughput Rates
  - Randomness
  - Locality of reference
  - I/O size distribution
  - % Reads vs Writes
Blocktrace Analysis of AI Workloads

- Gives deeper insight into I/O profile
- The block report generated by “btt” provides detail about each I/O:
  - Command (read or write), precise timestamp, starting LBA, ending LBA
  - From the above data we can derive details about:
    - Randomness: If starting address of I/O “B” equals ending address of I/O “A”, I/O is sequential
    - Read/write ratios
    - I/O size distribution: Ending LBA minus starting LBA equals block size in sectors
    - Locality of reference: Some address ranges are accessed more frequently than others
Rule of Thumb

- **AI workloads are computation bound**
  - Loading a 200KB image takes \(~200\text{us}\)
  - Classify a image takes \(~10\text{ms}\)

- **Parallelize AI jobs to saturate I/O**
  - Use a cluster of GPUs
  - Keep every GPU busy
I/O intensive Methodologies

Benchmarking AI workloads in a customer representative scenarios
Limiting Memory

- To accurately model realistic workload with very large training dataset requirement
  - Readily available benchmark datasets are small and fit in memory
  - Goal is to stress storage in a small realistic test environment
- Control Dataset size to memory ratio
  - e.g. MLPerf ImageNet dataset (150 GB)
  - Docker memory limit options

<table>
<thead>
<tr>
<th>Dataset Size (GB)</th>
<th>System Memory (GB)</th>
<th>Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>150</td>
<td>768</td>
<td>1:5</td>
</tr>
<tr>
<td>150</td>
<td>64</td>
<td>2.5:1</td>
</tr>
</tbody>
</table>
Simultaneous Data Ingestion and Training

- Normally, training is not run in isolation
- Multiple models to be trained
- Realistic scenario: data ingest and training happen together
Training in parallel

- **Training parallelism:**
  - Storage to meet the needs of concurrent data ingest of different training jobs

- **Hyper-parameter tuning:**
  - Run tens of hundreds of instances of the same training job with different configuration of the model
Inference: Streaming applications

- Inference is more likely I/O bound
  - Training has 3x computations compared to Inferencing
    - Forward propagation, backward propagation, and weight updates
  - Less CPU bound implies possibility of I/O bound
I/O Challenges for Streaming applications

- Large amount of concurrent input data volume
  - One 4K 30 fps video stream: 45Mbps (~6MBps)
    - 1000 video streams: 45Gbps (~6GBps)
  - Massive intermediate data from different stages in a pipeline

- Video processing pipeline
  - Videos are split into frames
  - Stages are isolated into containers
  - One stage consume frames from last stage
  - Frames are passed through Apache Kafka with replicas
Test System

Hardware Components | Details
--- | ---
**GPU** | 8x Nvidia Tesla V100S, 32 GB

**CPU** | Intel Xeon Platinum 8268, 2.9 GHz, 2 Sockets, 2 threads per core, 96 (24*2*2) total cores, 768 GB System Memory

**Storage** | Local: 1 Samsung PM9A3 (3.49 TiB) drive per host; PCI Express Gen4 x 4 interface U.2 (EXT4 file system)

Software Components | Details
--- | ---
**Ubuntu** | 20.04 focal

**Tensorflow (tensorflow-gpu)** | MLPerf- Version: 2.4.1

**Docker** | Version: 20.10.12

**CUDA Toolkit** | Version: CUDA-11.2

**FIO** | Version: 3.26-59

**ResNet50 v1.5 model** | Distributed multi-GPU training with ImageNet ILSVRC2012 dataset

**OpenMPI** | Version: 3.0.0

**Horovod** | Version: 0.24.2

For inference testbed:
- **Compute node cluster**
  - Kubernetes

- **Storage (message broker) cluster**
  - Kafka (Helm charts)
# Dataset and Model details

<table>
<thead>
<tr>
<th>Task</th>
<th>Model</th>
<th>Framework</th>
<th>Dataset details</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Image classification training</strong></td>
<td>ResNet50</td>
<td>Tensorflow-gpu: 2.4.1</td>
<td>ImageNet-1k</td>
</tr>
<tr>
<td><strong>Video streaming and recognition: Inference through Image classification model</strong></td>
<td>ResNet50</td>
<td>Tensorflow-gpu: 2.11.0</td>
<td>1. Videos:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>a. Big Buck Bunny, Frame rate:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>24FPS, Resolution: 1920 x 1080, Size: 45 MB, Duration: 09:56 min</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>b. Costa Rica, Frame rate:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>60FPS, Resolution: 3840 x 2160, Size: 1.13 GB, Duration: 05:13 min</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2. ImageNet-1k Validation dataset</td>
</tr>
</tbody>
</table>
Impact of Limiting Memory
Baseline vs Limited memory: Disk profiles

- Disk throughput is substantially increased $\rightarrow 48x$
- Training time does not change much when limiting memory $\rightarrow$ with faster/ performant storage

<table>
<thead>
<tr>
<th>Metric</th>
<th>Baseline</th>
<th>Limited Memory</th>
</tr>
</thead>
<tbody>
<tr>
<td>Avg. IOPS</td>
<td>23</td>
<td>2,244</td>
</tr>
<tr>
<td>Avg. Throughput (MiB/s)</td>
<td>5.84</td>
<td>280.46</td>
</tr>
<tr>
<td>Avg. Block Size (KiB)</td>
<td>169.55</td>
<td>170.23</td>
</tr>
<tr>
<td>Avg. Response time (μs)</td>
<td>203.63</td>
<td>185.91</td>
</tr>
<tr>
<td>Training time (minutes)</td>
<td>364</td>
<td>357</td>
</tr>
</tbody>
</table>

* Zero values are discarded from disk metric statistics calculation in the tables. Disk I/O, Throughput, Block sizes, Response time, CPU and GPU utilization % are average values.
System resources

- Baseline and Limiting memory exhibit comparable performance
I/O Profile: Resnet50 Single-Model Training

<table>
<thead>
<tr>
<th>I/O</th>
<th>Read Pct.</th>
<th>Random Pct.</th>
<th>Average IOPS</th>
<th>Minimum Read Request (KiB)</th>
<th>Median Read Request (KiB)</th>
<th>Maximum Read Request (KiB)</th>
<th>Mean Read Request (KiB)</th>
<th>Standard Deviation (KiB)</th>
<th>Minimum Write Request (KiB)</th>
<th>Median Write Request (KiB)</th>
<th>Maximum Write Request (KiB)</th>
<th>Mean Write Request (KiB)</th>
<th>Standard Deviation (KiB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total</td>
<td>99.94%</td>
<td>83.88%</td>
<td>639</td>
<td>4</td>
<td>128</td>
<td>256</td>
<td>171</td>
<td>60</td>
<td>4</td>
<td>8</td>
<td>108</td>
<td>16</td>
<td>16</td>
</tr>
<tr>
<td>Random</td>
<td>99.96%</td>
<td>100%</td>
<td>536</td>
<td>4</td>
<td>128</td>
<td>256</td>
<td>177</td>
<td>62</td>
<td>4</td>
<td>8</td>
<td>108</td>
<td>13</td>
<td>13</td>
</tr>
<tr>
<td>Sequential</td>
<td>99.85%</td>
<td>0%</td>
<td>103</td>
<td>4</td>
<td>128</td>
<td>256</td>
<td>135</td>
<td>30</td>
<td>4</td>
<td>4</td>
<td>44</td>
<td>19</td>
<td>18</td>
</tr>
</tbody>
</table>

- Nearly 100% read, 84% random, with I/O sizes ranging from 4K to 256K
Trace statistics: I/O plots and locality histogram

- Random and Sequential reads within a relatively narrow address range
- High locality of reference
Trace statistics: I/O Request Sizes

- Random reads ranged from 4K to 256K, but more than 99% were either 128K or 256K (left).
- Random write I/O sizes were more diverse (right). Sequential I/O size distribution was similar.
Simultaneous Data Ingestion and Training
## Baseline vs Limited memory: Disk profiles

<table>
<thead>
<tr>
<th>Metric</th>
<th>Baseline</th>
<th>Limited Memory</th>
</tr>
</thead>
<tbody>
<tr>
<td>Avg. IOPS</td>
<td>25054</td>
<td>25035</td>
</tr>
<tr>
<td>Avg. Throughput (MiB/s)</td>
<td>3162.59</td>
<td>3181.91</td>
</tr>
<tr>
<td>Avg. Block Size (KiB)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Read:</td>
<td>169.8</td>
<td>Read: 170.4</td>
</tr>
<tr>
<td>Write:</td>
<td>128</td>
<td>Write: 128</td>
</tr>
<tr>
<td>Avg. Response time (ms)</td>
<td>79.418</td>
<td>75.48</td>
</tr>
<tr>
<td>Training time (minutes)</td>
<td>373.15</td>
<td>373</td>
</tr>
</tbody>
</table>

* Zero values are discarded from disk metric statistics calculation in the tables. Disk I/O, Throughput, Block sizes, Response time, CPU and GPU utilization % are average values.
System resources

- GPU utilization unaffected:
  - GPU not handling data ingestion operations
- CPU-IOWait increases:
  - Parallel data ingestion
### I/O Characterization

<table>
<thead>
<tr>
<th>I/O</th>
<th>Read Percent</th>
<th>Random Percent</th>
<th>Average IOPS</th>
<th>Minimum Read (KiB)</th>
<th>Median Read (KiB)*</th>
<th>Mean Read (KiB)</th>
<th>Read Std. Dev. (KiB)</th>
<th>Minimum Write (KiB)</th>
<th>Median Write (KiB)</th>
<th>Maximum Write (KiB)</th>
<th>Mean Write (KiB)</th>
<th>Write Std. Dev. (KiB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>0.33%</td>
<td>95.47%</td>
<td>24,714</td>
<td>4</td>
<td>256</td>
<td>247</td>
<td>46</td>
<td>4</td>
<td>128</td>
<td>508</td>
<td>128</td>
<td>6</td>
</tr>
<tr>
<td>Limited Memory</td>
<td>1.78%</td>
<td>93.86%</td>
<td>24,786</td>
<td>4</td>
<td>256</td>
<td>245</td>
<td>52</td>
<td>4</td>
<td>128</td>
<td>508</td>
<td>128</td>
<td>7</td>
</tr>
</tbody>
</table>

* Also Maximum Read

**Baseline**
- I/O profile is mostly write and mostly random
- Primary difference between baseline and limited memory is in the read profile
- In baseline training run, disk reads occur primarily in the first epoch because the entire data set fits in memory
- In limited memory run, reads from disk occur during all training epochs

**Limited Memory**

![Read Scatterplot](Baseline.png)

![Read Scatterplot](Limited Memory.png)
Trace statistics: Write I/O plots and locality

- Writes are ~95% random, but locality of reference is high
Training in Parallel
Parallel models training: Disk profiles

<table>
<thead>
<tr>
<th>Containers/Parallel Models</th>
<th>1</th>
<th>2</th>
<th>4</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>GPUs per training workload</td>
<td>8</td>
<td>4</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>Batch Size</td>
<td>1024</td>
<td>1024</td>
<td>1024</td>
<td>512</td>
</tr>
<tr>
<td>Disk I/O</td>
<td>1658.3</td>
<td>1679.94</td>
<td>2805.26</td>
<td>1245.34</td>
</tr>
<tr>
<td>Disk Throughput (MiB/s)</td>
<td>276.55</td>
<td>419.56</td>
<td>351.32</td>
<td>310.72</td>
</tr>
<tr>
<td>Block (KiB)</td>
<td>169.55</td>
<td>253.71</td>
<td>127.31</td>
<td>254.2</td>
</tr>
<tr>
<td>Response time (μs)</td>
<td>203.63</td>
<td>304.57</td>
<td>162.71</td>
<td>195.88</td>
</tr>
<tr>
<td>Training time (minutes)</td>
<td>364</td>
<td>258.2</td>
<td>441</td>
<td>682</td>
</tr>
</tbody>
</table>

* Zero values are discarded from disk metric statistics calculation in the tables. Disk I/O, Throughput, Block sizes, Response time, CPU and GPU utilization % are average values.
System resources

- CPU and GPU utilization increases with number of read-intensive training workloads
I/O Characterization

<table>
<thead>
<tr>
<th></th>
<th>1 Model</th>
<th>2 Models</th>
<th>4 Models</th>
<th>8 Models</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total Reads</td>
<td>794,262</td>
<td>509,876</td>
<td>1,084,946</td>
<td>509,674</td>
</tr>
<tr>
<td>Mean Read Request</td>
<td>170 KiB</td>
<td>256 KiB</td>
<td>128 KiB</td>
<td>256 KiB</td>
</tr>
<tr>
<td>Median Read Request</td>
<td>128 KiB</td>
<td>256 KiB</td>
<td>128 KiB</td>
<td>256 KiB</td>
</tr>
<tr>
<td>Randomness</td>
<td>83.9%</td>
<td>95.4%</td>
<td>74.8%</td>
<td>92.6%</td>
</tr>
<tr>
<td>Locality Bands</td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>Percent of I/O received by 10% address space</td>
<td>99%</td>
<td>63%</td>
<td>98%</td>
<td>62%</td>
</tr>
</tbody>
</table>

- 2-models and 8-models parallel training similarities
- Average request size increased from 256 blocks to 512 blocks (256 KiB)
- 8-models training is 100% read, with randomness increasing from 75% (4-models) to 92%
Two- and eight-models show several bands of activity distributed across drive’s address range.
Trace statistics: Locality

- Highest locality of reference in single model training: 6% address space receiving > 99% reads
- Two- and eight-models have reads more distributed across the drive’s address range
**Trace statistics: I/O Request Sizes**

- **Single model**: Random read request sizes ranged from 4KiB to 256KiB
  - Mainly either 4KiB or 256KiB
- **Four models**: Most reads are 128 KiB

Eight models
Inference: Streaming workload
## Data Ingestion Disk Metrics

<table>
<thead>
<tr>
<th>Metric/Concurrent Streams</th>
<th>300, 24 FPS Videos, 3 RF (6 partitions) - 1 topics</th>
<th>300, 24 FPS Videos, 3 RF (6 partitions) - 3 topics</th>
<th>300, 60 FPS Videos, 3 RF (6 partitions) - 1 topic</th>
<th>300, 60 FPS Videos, 3 RF (6 partitions) - 3 topics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Avg. IOPS</td>
<td>4471.79</td>
<td>7327.74</td>
<td>27637.63</td>
<td>13234</td>
</tr>
<tr>
<td>Avg. Throughput (MiB/s)</td>
<td>46.77</td>
<td>152.69</td>
<td>407.75</td>
<td>306.63</td>
</tr>
<tr>
<td>Avg. Block Size (KiB)</td>
<td>Read: 110.87 Write: 11.69</td>
<td>Read: 44 Write: 18</td>
<td>Read: 157.7 Write: 13.2</td>
<td>Read: 125 Write: 21.18</td>
</tr>
<tr>
<td>Avg. Response time (μs)</td>
<td>838.37</td>
<td>1489.38</td>
<td>975.29</td>
<td>1223.09</td>
</tr>
</tbody>
</table>

- Frame extraction from 300 concurrent streams and publish to topic: ~27K IOPS
- Disk I/O and Throughput increase with great parallelism
CPU overhead increased with increasing partitions from 3 to 6 but remained constant with further increase to 12 partitions.

Videos with higher frame rate (FPS) and resolution showed relatively higher CPU utilization.
Data Ingestion I/O Characterization

<table>
<thead>
<tr>
<th>I/O</th>
<th>Read Percent</th>
<th>Random Percent</th>
<th>Average IOPS</th>
<th>Minimum Write (KiB)</th>
<th>Median Write (KiB)</th>
<th>Maximum Write (KiB)</th>
<th>Mean Write (KiB)</th>
<th>Std. Dev. (KiB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>30 Streams</td>
<td>0.08%</td>
<td>71.43%</td>
<td>281</td>
<td>4</td>
<td>4</td>
<td>764</td>
<td>32</td>
<td>96</td>
</tr>
<tr>
<td>100 Streams</td>
<td>0.54%</td>
<td>69.92%</td>
<td>422</td>
<td>4</td>
<td>8</td>
<td>764</td>
<td>64</td>
<td>140</td>
</tr>
</tbody>
</table>

- Nearly 100% write, ~70% random

- Writes more widely distributed across SSD’s address range with increased streams

Standard deviation suggests high diversity of write sizes
Trace statistics: Locality of reference and I/O sizes distribution

- Write locality high both for 30 and 100 streams with 6% address space receiving 87% and 93% writes respectively.

- Random write request size distribution was quite varied
- 70% of random writes were 28K or less, but the remaining 30% ranged up to 764K
System Implications and Discussion

- The majority of the workloads studied were primarily random, with relatively high locality of reference
  - Suitable for testing optimizations such as read caching and write coalesce
- Some workloads (e.g. inference streaming) exhibited a very diverse write I/O size distribution
  - Useful “real-world” benchmarking tool for challenging high performance storage systems
Conclusion

▪ Simultaneous data ingestion and training, and inference were particularly effective benchmarks
  ▪ These approaches present challenging, “real-world” workloads to storage

▪ Our testing indicates that high-performance storage allows I/O-intensive and computationally-intensive portions of the AI pipeline to run in parallel with minimal impact on training and inference times.
Thank You!
Backup Slides
## Summary statistics

| Workload Description | Read Percentage | Random Percentage | Average IOPS | Minimum Read Request (KiB) | Median Read Request (KiB) | Maximum Read Request (KiB) | Mean Read Request (KiB) | Standard Deviation (KiB) | Minimum Write Request (KiB) | Median Write Request (KiB) | Maximum Write Request (KiB) | Mean Write Request (KiB) | Standard Deviation (KiB) | Random Read Operations | Random Write Operations | Sequential Read Operations | Sequential Write Operations | Trace Length Seconds |
|----------------------|----------------|------------------|--------------|-----------------------------|---------------------------|---------------------------|------------------------|--------------------------|-----------------------------|---------------------------|---------------------------|-------------------------|--------------------------|-------------------------|-------------------------|-------------------------|-------------------------|
| ResNet50 Training Single Model | 99.94% | 83.88% | 639 | 4 | 128 | 256 | 171 | 60 | 8 | 108 | 16 | 16 | 666,340 | 265 | 127,922 | 194 | 1,244 |
| ResNet50 Training Two Models | 100.00% | 95.43% | 600 | 4 | 256 | 256 | 256 | 6 | 4 | 4 | 4 | 138 | 6 | 46,231,316 | 1,312 | 1,824,854 | 744 | 20,822 |
| ResNet50 Training Two Models LM | 100.00% | 96.20% | 2,308 | 4 | 256 | 256 | 256 | 172 | 113 | 4 | 4 | 4 | 4 | 138 | 6 | 46,231,316 | 1,312 | 1,824,854 | 744 | 20,822 |
| ResNet50 Training Four Models | 99.95% | 74.79% | 890 | 4 | 128 | 128 | 128 | 2 | 4 | 4 | 4 | 128 | 11 | 20 | 811,309 | 471 | 273,637 | 52 | 1,220 |
| ResNet50 Training Eight Models | 100.00% | 92.59% | 257 | 4 | 256 | 256 | 256 | 7 | 0 | 0 | 0 | 0 | 0 | 741,924 | 0 | 37,746 | 0 | 1,983 |

**Inference Baseline, Video Streaming, Ingestion Phase (30 Streams, 3 Partitions)**
- 0.08% | 71.43% | 281 | 4 | 128 | 128 | 102 | 50 | 4 | 4 | 764 | 32 | 96 | 773 | 720,927 | 40 | 288,605 | 3,599 |

**Inference Baseline, Video Streaming, Ingestion Phase (100 Streams, 3 Partitions)**
- 0.54% | 69.92% | 422 | 4 | 128 | 128 | 118 | 32 | 4 | 8 | 764 | 64 | 140 | 8,016 | 1,054,351 | 260 | 456,703 | 3,599 |

**Simultaneous Data Ingestion and Training (5 Epochs)**
- 0.33% | 95.47% | 24,714 | 4 | 256 | 256 | 247 | 46 | 4 | 128 | 508 | 128 | 6 | 574,458 | 175,355,092 | 33,960 | 8,305,481 | 7,456 |

**Simultaneous Data Ingestion and Training (5 Epochs Limited Memory)**
- 1.78% | 93.86% | 24,786 | 4 | 256 | 256 | 245 | 52 | 4 | 128 | 508 | 128 | 6 | 2,879,201 | 157,200,319 | 154,185 | 10,321,862 | 6,881 |

**Training with Checkpointing Every 100 Steps**
- 93.27% | 92.61% | 165 | 4 | 256 | 256 | 255 | 14 | 4 | 16 | 1,280 | 431 | 567 | 507,355 | 12,527 | 16,214 | 25,255 | 3,408 |

**Training with Checkpointing Every 1252 Steps (Default Interval)**
- 99.68% | 96.78% | 151 | 4 | 256 | 256 | 256 | 7 | 4 | 16 | 1,280 | 134 | 362 | 501,256 | 297 | 15,348 | 1,351 | 3,484 |

**BERT 2000-Step Default Checkpoint Interval PM983**
- 0.22% | 4.38% | 26 | 4 | 128 | 128 | 126 | 15 | 4 | 128 | 128 | 128 | 5 | 69 | 2,740 | 74 | 61,185 | 2,511 |

**BERT 2000-Step Default Checkpoint Interval PM9A3**
- 0.11% | 60.38% | 43 | 4 | 128 | 256 | 166 | 66 | 4 | 8 | 1,280 | 36 | 176 | 215 | 164,878 | 92 | 108,218 | 6,395 |

**BERT 2000-Step Default Checkpoint Interval PM9A3 + Preconditioning + New FS**
- 0.23% | 0.49% | 2 | 4 | 128 | 256 | 129 | 89 | 4 | 1,280 | 1,280 | 1,127 | 326 | 9 | 16 | 5,113 | 2,163 |

**BERT 2000-Step Default Checkpoint Interval PM9A3 + New FS + Pytorch Framework**
- 0.00% | 3.47% | 181 | 0 | 0 | 0 | 0 | 0 | 0 | 4 | 508 | 1,280 | 579 | 443 | 7,382 | 0 | 205,078 | 1,176 |

**BERT 2000-Step Limited Memory Default Checkpoint Interval PM983**
- 0.27% | 3.63% | 26 | 4 | 128 | 128 | 126 | 5 | 4 | 128 | 128 | 128 | 5 | 107 | 2,149 | 60 | 59,818 | 2,380 |

**BERT 2000-Step Limited Memory Default Checkpoint Interval PM9A3**
- 0.12% | 58.17% | 45 | 4 | 128 | 256 | 169 | 63 | 4 | 8 | 1,280 | 36 | 174 | 219 | 158,072 | 106 | 113,707 | 6,110 |

**BERT 2000-Step With 250-Step Checkpoint Interval PM983**
- 0.10% | 3.70% | 106 | 4 | 128 | 128 | 123 | 25 | 4 | 128 | 128 | 128 | 5 | 133 | 965,119 | 254,328 | 2,504 |

**BERT 2000-Step With 250-Step Checkpoint Interval PM9A3**
- 0.08% | 57.94% | 131 | 4 | 128 | 256 | 172 | 64 | 4 | 8 | 1,280 | 89 | 285 | 196 | 202,814 | 99 | 147,279 | 2,680 |

**BERT 2000-Step With Simultaneous Data Ingestion PM983**
- 0.05% | 97.63% | 4,470 | 4 | 128 | 128 | 7 | 20 | 4 | 128 | 128 | 127 | 8 | 17,135 | 33,601,880 | 1,471 | 814,030 | 7,704 |

**BERT 2000-Step With Simultaneous Data Ingestion PM9A3**
- 0.04% | 99.32% | 24,311 | 4 | 4 | 256 | 10 | 31 | 4 | 128 | 1,280 | 127 | 12 | 6,949 | 62,821,436 | 16,860 | 411,639 | 2,402 |
Please take a moment to rate this session.

Your feedback is important to us.