SNIA DEVELOPER CONFERENCE

SDC 24

BY Developers FOR Developers

September 16-18, 2024
Santa Clara, CA

Storage in the era of large-scale AI computing

What we already know (or) not?

Pratik Mishra

AMD

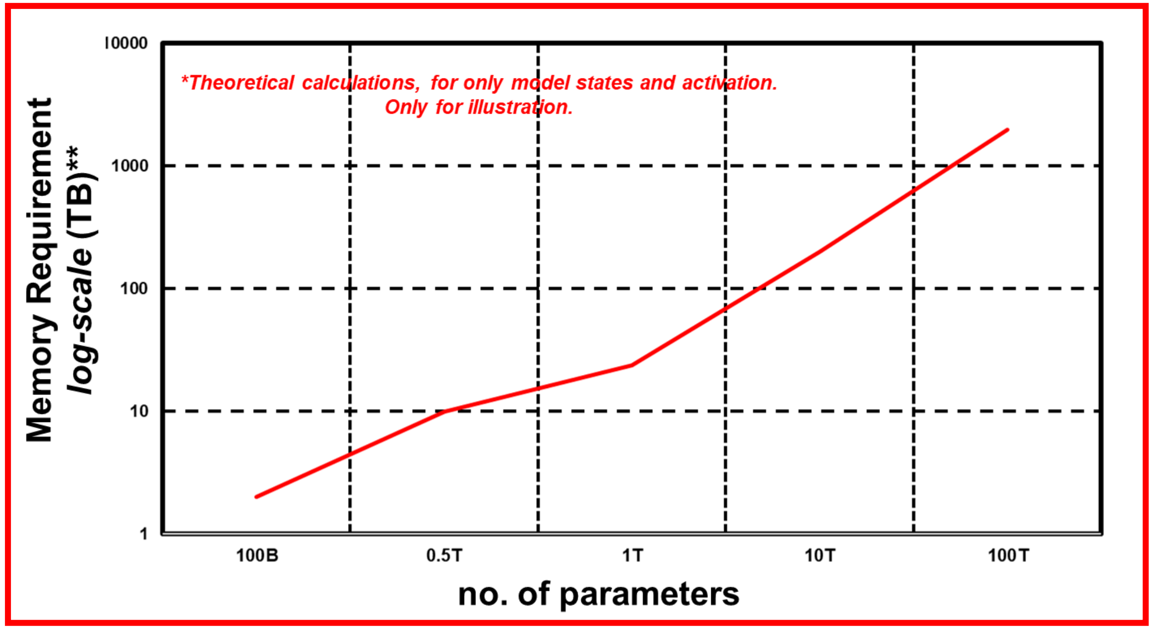
Table of Contents

- The Problem
- ***Storage is the key player in AI life-cycle***
  - Data Ingestion and Pre-processing
  - AI Training
  - Emerging AI Inference
  - IO Blender
- Final Thoughts
- Conclusion
- Copyrights and Disclaimer

Disclaimer: *Please refer to the Copyrights and Disclaimer in the presentation. The information provided in this talk/presentation is based on understanding/knowledge of systems and collected through public available data, publications and other forms of media. We have tried to cite most relevant sources. We (the authors and associated organization) owe no responsibility towards the content's accuracy or claims, and they should be viewed as personal viewpoints/opinions to cater open discussions.*

The Problem: ***Just Another AI presentation***

- **5Vs** (<u>V</u>olume, <u>V</u>elocity, <u>V</u>alue, <u>V</u>ariety, <u>V</u>eracity)
- AI requirements are becoming *“more”* multi-modal
  - Text, images, videos, etc. (Everything!)
  - *Data is also becoming sparser*.
- Models and data (+metadata) cannot fit in a single system.



***Calculations based on FP16 precision AI Training.*

Memory never outgrows the requirements of data.

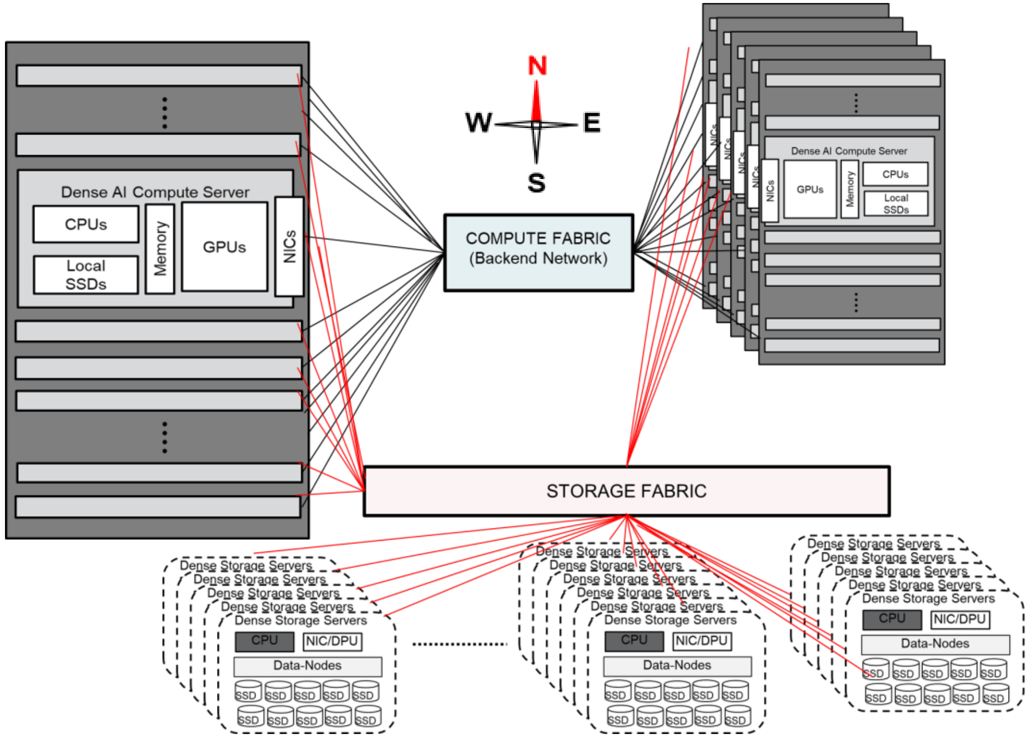
Data Movement: The necessary Evil!

- ❑ Build bigger and wider distributed systems $$$...$$.
  - ❑ *Scale-up and scale-out.*

- ❑ Most focus has been in E-W traffic (i.e., compute).
  - ❑ *Parallelism, collective optimizations, batching, data-types, quantization, etc.*

Large amounts of data must move across inter + intra nodes, servers, racks, servers, data-centers.

- ❑ Storage can easily become the bottleneck and *GPUs critical compute resources need to wait for data.*

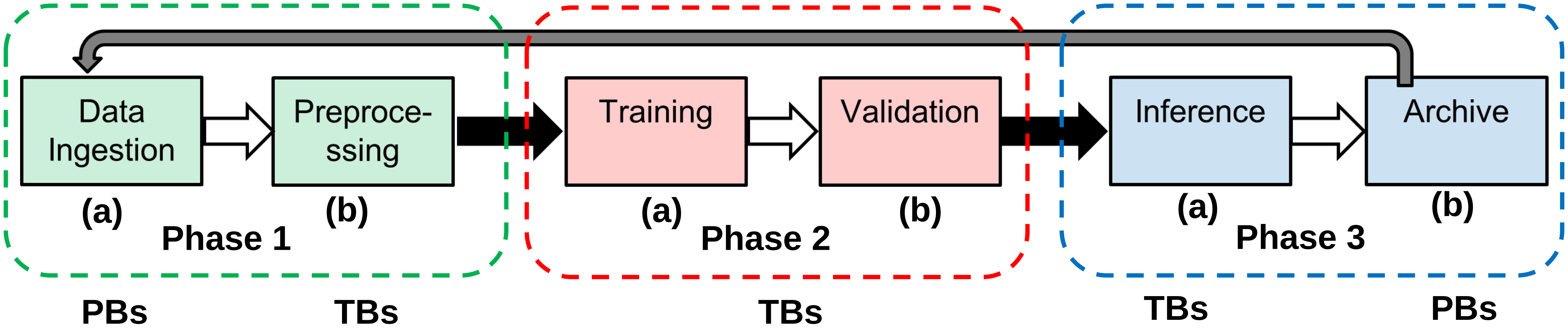


Storage despite being a key-player in AI, is often over-looked [1] and least talked about [2].

[1] Gartner (February 14, 2024): *Top Storage Recommendations to Support Generative AI*
[2] Engineering at Meta (March 12, 2024): Building Meta's GenAI Infrastructure :

**Storage is used inter-changeably to imply the storage ecosystem (devices, nodes, servers, storage networks).*

Lifecycle of Data in AI



(a) Ingest massive ***objects*** via bulk-insert (streams) across data-sources/clouds/data-centers, etc.

(b) Multiple transform pipelines (ETL) of data to prepare "*tensors*" for training. – Annotation, indexing and search intensive.

(a) Loading model, batches of data for parallel training; update weights and parameters, while persisting *checkpoints*. ***Repeat (epoch)***.

(b) Validate the model parameters and gradients. ***Replays.***

(a) Loading trained model parameters, and process inference queries for real time output generation (store).

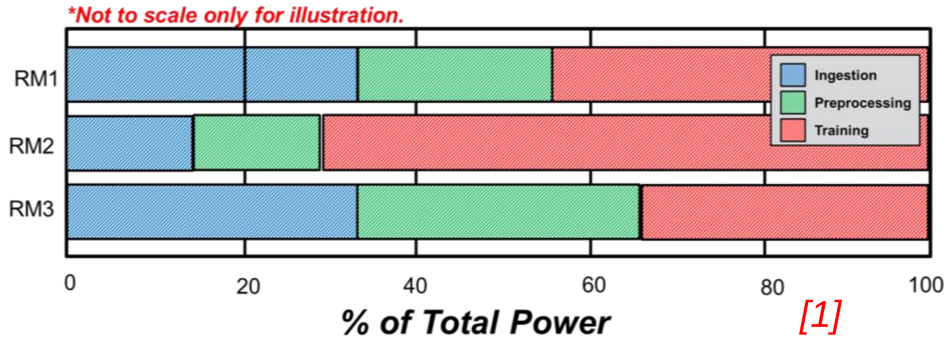
(b) Lifecycle management of data; retain training data and model for long.

Challenge: Maximize GPU/compute utilization and reduction of stalls due to storage.

Phase 1: Data Ingestion and Pre-processing

- ***Data Ingestion***: Aggregate data (text, images, videos, etc.), shapes, sizes, and store them in various formats efficiently for pre-processing.
  - Log aggregation via Kafka like-streams, or streams on top of RocksDB [1].
  - ETL type pipelines and features (mostly, sparse) are updated often.

(a) PB-scale capacity; Object/file protocol; Concurrent W throughput; High queue depth; Encryption.



[1]

Ingestion and Pre-processing consume more power than Training itself at a Hyperscalar infrastructure [1].

- ***Pre-processing***: Continuously transform raw data into pre-processed tensors for Training jobs to consume as efficiently (low latency, high throughput).
  - Iteratively read samples (features) dynamically in different formats.
  - Raw data is transformed into training samples.
    - Filtering, decryption, reconstructions, and multiple format transformations [1] (multiple ETL jobs).

(b) High R/W throughput; Sequential Writes, Advanced Compression/De-compression; Metadata heavy, High queue depth.

[1] Zhao, Mark, et al. "Understanding data storage and ingestion for large-scale deep recommendation model training: Industrial product." Proceedings of the 49th annual international symposium on computer architecture. 2022.
[2] Zhao, Mark, et al. "RecD: Deduplication for end-to-end deep learning recommendation model training infrastructure." *Proceedings of Machine Learning and Systems* 5 (2023): 754-767.

Data Ingestion, Preprocessing - Issues

- Raw data is transformed into training samples (tensors): heavy ***filtering***.
  - Training jobs are diverse and geo-distributed.
  - Data is usually sparse, and operations involve high reductions ratios (*filtering*).

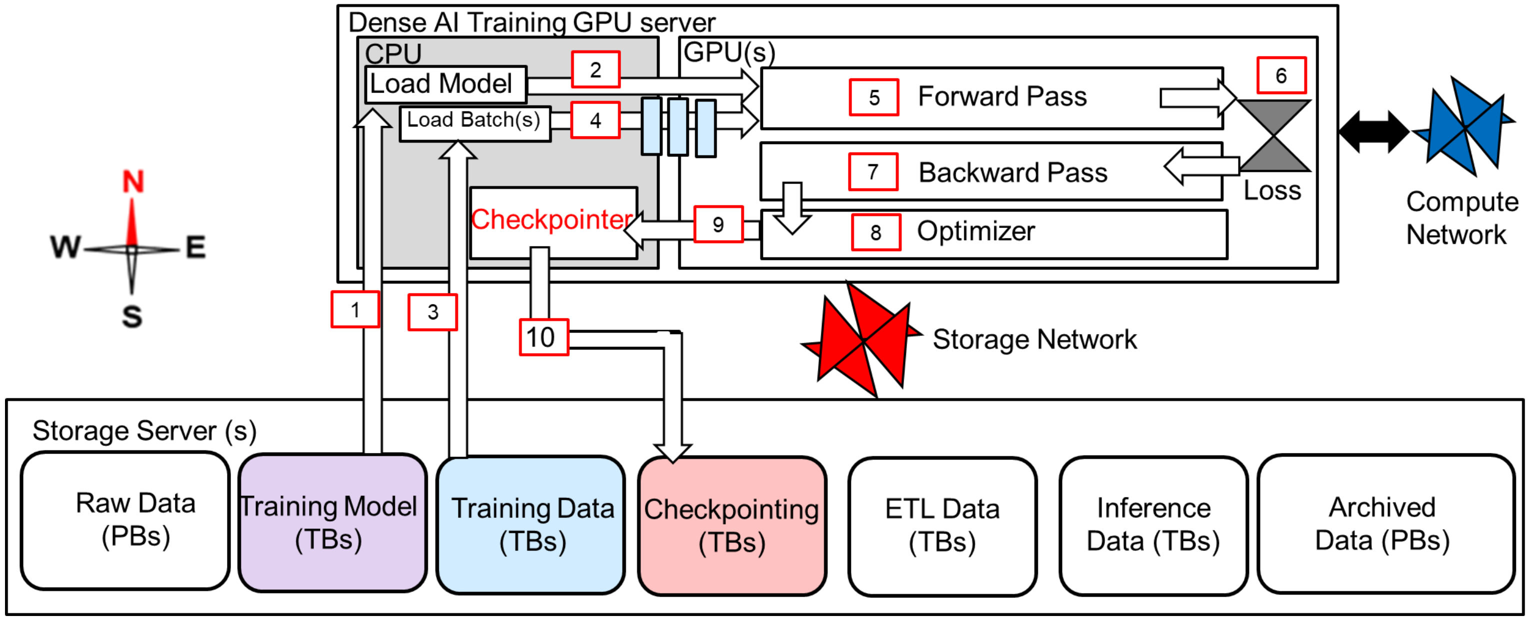
- When *Preprocessing* executed by Training Nodes (CPUs) can cause ***GPUs stall for the data.***
  - High IOPS from storage servers.
  - Bottlenecked by front-end resources (CPU, Memory)
  - NICs are highly over-subscribed (@line-rate)

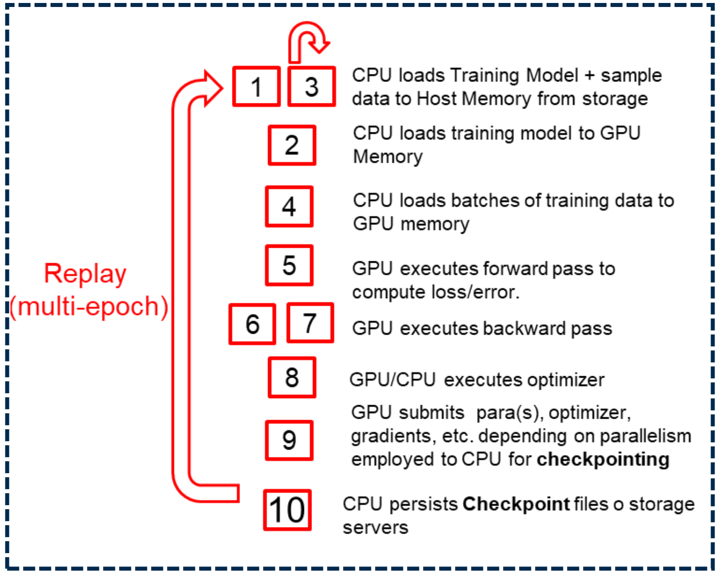
IO Tax

Therefore, it is extremely critical to have DIP pipelines in infrastructure which is highly optimized for storage and retrieval of training data.

[1] Zhao, Mark, et al. "Understanding data storage and ingestion for large-scale deep recommendation model training: Industrial product." Proceedings of the 49th annual international symposium on computer architecture. 2022.
[2] Zhao, Mark, et al. "RecD: Deduplication for end-to-end deep learning recommendation model training infrastructure." *Proceedings of Machine Learning and Systems* 5 (2023): 754-767.

Phase 2: AI Training





For Hyperscalar Llama 3 405B pre-training on 16K GPUs, the Model FLOPs Utilization is ~ **41%** [1]. (BF16; 4D parallelism TP=8, CP=1, PP=16, DP =128)

- Extremely high memory req. (TBs) – (a) model states; (b) activations; (c) training data; (d) checkpoints.
- Highly bursty (intense) and periodic (line-rate of host-NICs 400Gbps), etc. [2].
- Both E-W traffic (collectives) and N-S traffic (loads and checkpoints).

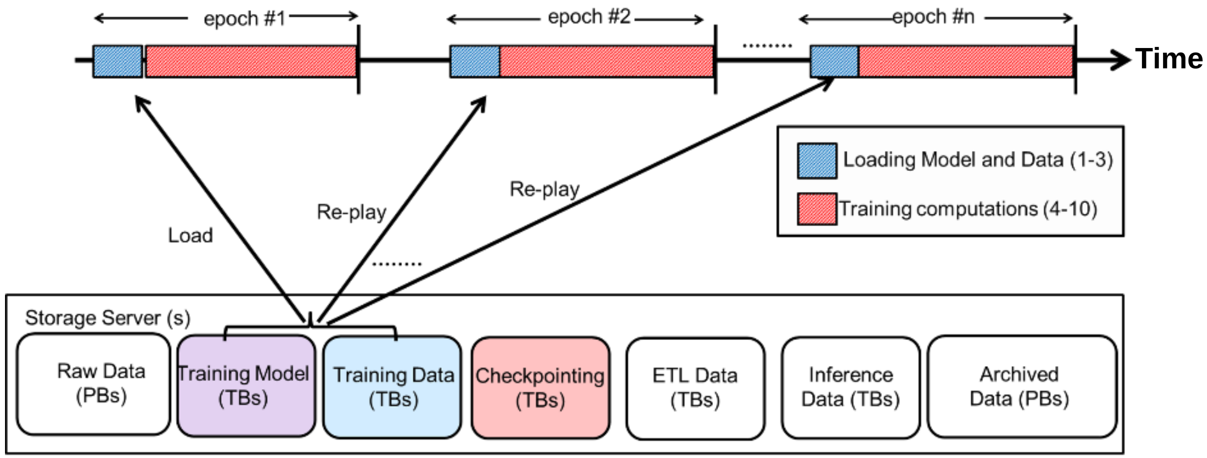
[1] Dubey, Abhimanyu, et al. "The llama 3 herd of models." *arXiv preprint arXiv:2407.21783* (2024).
[2] Qian, Kun, et al. "Alibaba hpn: A data center network for large language model training." *Proceedings of the ACM SIGCOMM 2024 Conference*. 2024.

Model and Data Load(s)

- Model Load significant (TBs), but once/epoch.
- Training samples loaded in batches to Training Nodes.
  - For 1T para. model, O(800TB)* of data required for high-end training.

High Read throughput required, Large # of small file random reads (metadata heavy).

- *Data Load (Preparation) incurs high data-center* tax.
  - Host resources and network highly over-subscribed.
  - DLRM data has high-scope for de-duplication [2].
  - Diverse functions, mostly filtering ops, (de)-compression, etc.



**Highly simplistic representation of epoch-based training for model and data loads (re), not all functions are shown. Only for illustration.*

Highly read latency sensitive.
Spectrum of IO sizes (batches, mini-batches, re-plays)

Near-Storage Computation ???

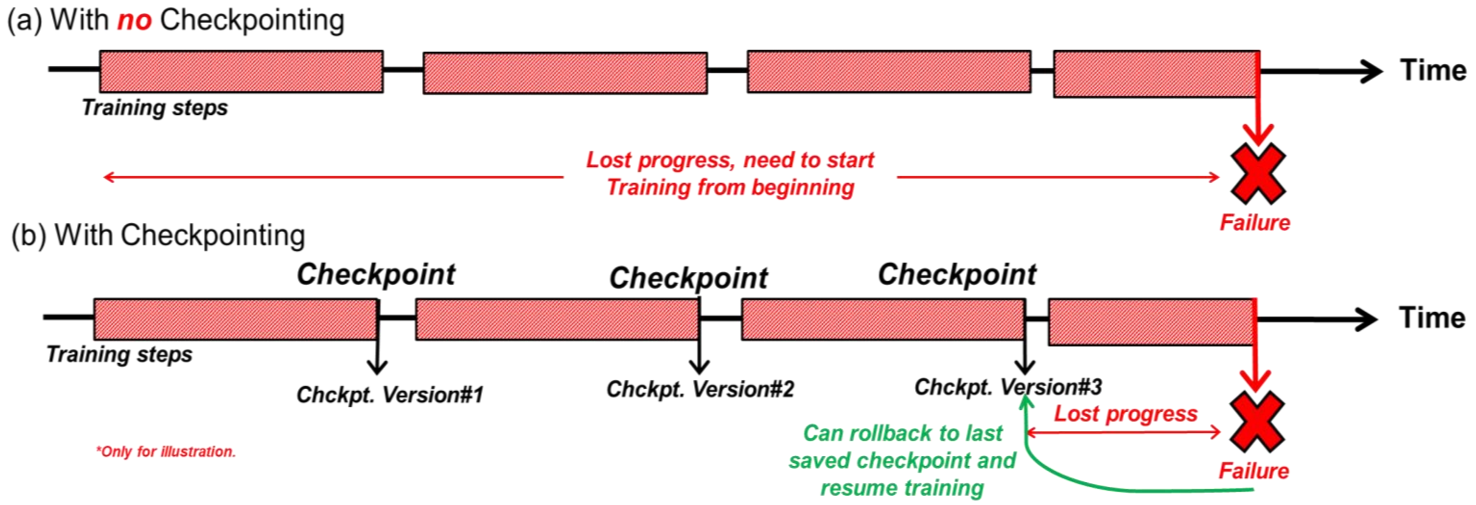
Significant GPU stalls due to inefficient load pipelines [1,2].

*Estimating, 200 tokens/ parameter required for training with 4 bytes/token.

[1] Zhao, Mark, et al. "Understanding data storage and ingestion for large-scale deep recommendation model training: Industrial product." Proceedings of the 49th annual international symposium on computer architecture. 2022.
[2] Zhao, Mark, et al. "RecD: Deduplication for end-to-end deep learning recommendation model training infrastructure." *Proceedings of Machine Learning and Systems* 5 (2023): 754-767.

Phase 2: AI Training – Checkpointing

- Training jobs typically run for weeks and months.
- Checkpointing is the critical mechanism of saving snapshots and vital information of the model.



- Crash on any GPU could be extremely ***expensive*** (time, money, power, resources, etc.).
  - Customer of a hyperscalar, checkpointing every hour training with a 3K GPU cluster, rollback costs **$30K** [2].

Fault-tolerance 101: *With ever lowering MTBF(s) checkpointing frequency will increase [1,2].*

- Also used for:
  - *Hardware refresh, Resource re-balancing, Fine-tuning, early-kill (if error rates go up), increase accuracy, etc.*

[1] Qian, Kun, et al. "Alibaba hpn: A data center network for large language model training." Proceedings of the ACM SIGCOMM 2024 Conference. 2024.
[2] Dubey, Abhimanyu, et al. "The llama 3 herd of models." *arXiv preprint arXiv:2407.21783* (2024).

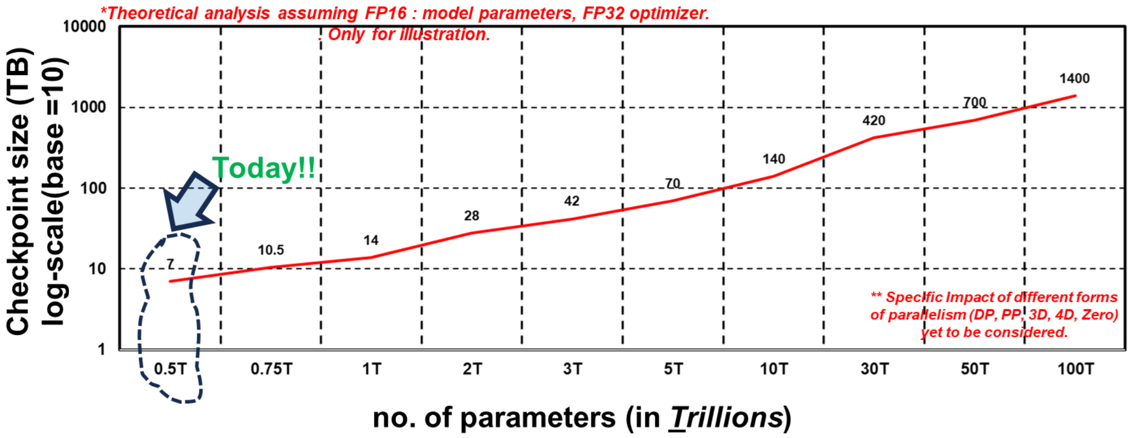
Checkpointing: What is involved?

❑ **C**heckpointing = **S**erialization + **P**ersistence

❑ **S**erialization : Create tensor file compatible structures, ***quantize data***, augment metadata for ease of reconstruction during checkpoint restore (loads).

❑ **P**ersistance: Write tensor serialized quantized files to remote persistent storage.

- ❑ Remote storage : scalability, high-availability.
- ❑ *no. of files and size also depends on parallelism shard*.
- ❑ Checkpointing creates sequence of writes to file.



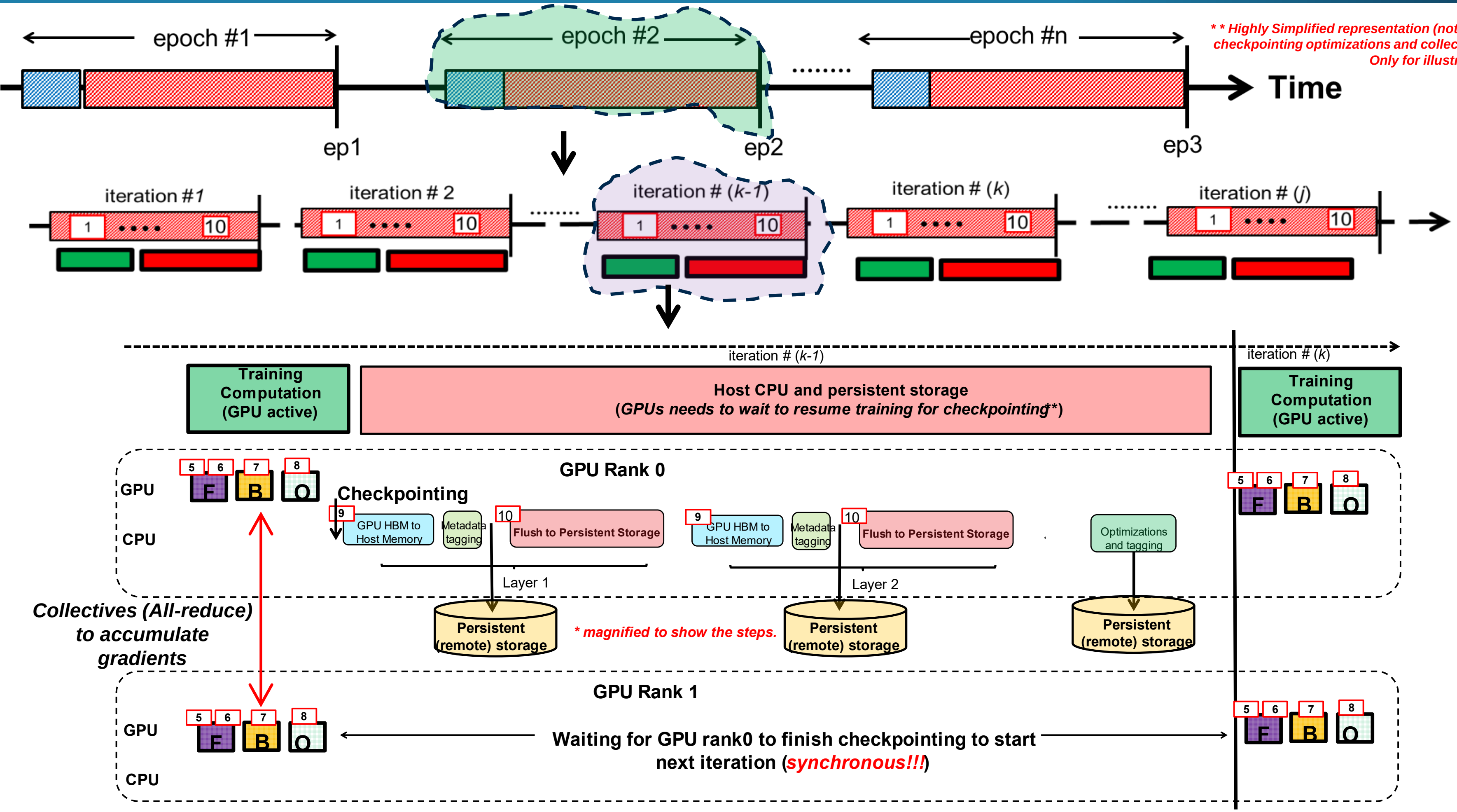
What goes into a checkpoint ?

❑ Model parameters (weights, biases), optimizer state (momentum, variance, gradients), and *may contain* - metadata- data type, size, reader state information (iterator), GPU rank, parallelism, etc.

With growing model sizes, checkpointing frequency and checkpointing size grows exponentially gets more distributed and complicated (persisting and restore).

[1] Qian, Kun, et al. "Alibaba hpn: A data center network for large language model training." Proceedings of the ACM SIGCOMM 2024 Conference. 2024.
[2] Dubey, Abhimanyu, et al. "The llama 3 herd of models." *arXiv preprint arXiv:2407.21783* (2024).

Checkpointing: From ground up!



** Highly Simplified representation (not actual). Complex interactions, checkpointing optimizations and collectives not shown for simplicity. Only for illustration

Training is paused and constricted by the slowest TrainingNode - Storage path.

Inefficient, synchronous, compute, network and storage infrastructure agnostic checkpointing can lead to increasing training time, wasting data-centric resources.

Training is paused and GPUs across ranks need to wait for the checkpoints to persist.

Checkpointing – Impact on Infrastructure @ scale

- ❑ Checkpointing footprint on Training Nodes (/GPU) and storage subsystem is massive.
  - ❑ With growing model size, total checkpoint size and /GPU size grows.
  - ❑ For hyperscalars, *30GB/GPU [1], Llama 3 training - 1MB-4GB/GPU [2].*

High R/W bandwidth with tight latency.

- ❑ Persistence and restores getting tougher with complex interactions.
  - ❑ *Data-Parallelism (DP), Tensor Parallelism* (TP), *3D parallelism* (TP, PP, DP), *now 4D parallelism, etc.*

- ❑ *For storage systems: manage network and storage BW for persisting multiple large checkpoints concurrently from different models being run in the data-center (each TBs).*
  - ❑ Highly bursty and periodic (@NIC line-rate): saturates storage fabric for Llama 3 training [2].
  - ❑ Unpredictable tail-latencies + multi-tenancy: SLA misses.
  - ❑ Storage NICs over-subscribed: need for efficient rate-limiting schedulers to reduce stalls and failures.

The goal for storage ecosystem (compute, network, storage subsystem) should be maximizing GPU BW utilization and minimizing the time to load and store checkpoints.

[1] Qian, Kun, et al. "Alibaba hpn: A data center network for large language model training." Proceedings of the ACM SIGCOMM 2024 Conference. 2024.
[2] Dubey, Abhimanyu, et al. "The llama 3 herd of models." *arXiv preprint arXiv:2407.21783* (2024).

Phase 3: AI Inference

- Most inference deploying infrastructures are latency sensitive to for providing real-time output for user-queries.

- Requires reliable and fast deployment of the model by loading trained models efficiently and with minimal *time-to-deploy*.

- Storage subsystems need to provide strict low-latency SLA guarantees for multi-tenant environments, especially for batched LLMs.
  - Characterized mostly by small random read IOs.
  - Same data is shared by multiple jobs (*performance isolation*).
  - *Requires Scale-out storage (for sharing between GPUs) and high availability with high performance*.

Scale-out, shared, high-performance storage with low-latency, high R bandwidth for saturating Inference node GPUs occupied with data.

Emerging AI Inference : RAGs & VectorDBs

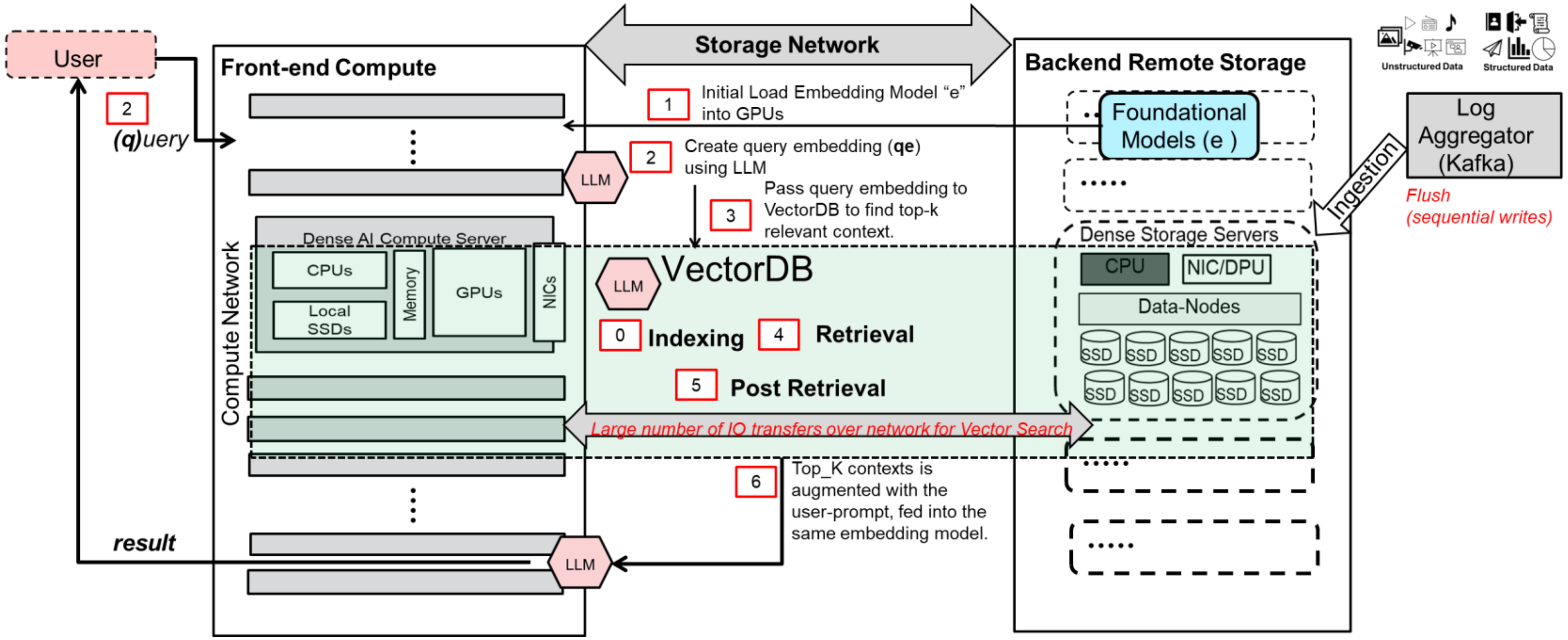
- LLMs during Generation suffer from hallucinations, and inaccurate information.
  - Models trained at prior timestamp, time-varying information either won't be present (or) becomes irrelevant when inference queries arrive.
  - Businesses would like domain-specific context and still do not want expose internal data to foundation models.

- Therefore, RAG (**R**etrieval **A**ugmented **G**eneration).
  - **A**ugments external information (or large corpus) and user-queries to **R**etrieve most relevant information (*top_k*) as context to foundational LLMs for **G**enerating most relevant/accurate response.

- RAG applications have huge storage footprint.
  - GenAI jobs are becoming “more” mutli-modal – images, text, videos, etc. {***Objects***}
  - Continuous data ingestion (via kafka streams), indexing (embeddings), and real-time augmentation, and inference (retrieval/filtering), etc.

Memory can never be enough to fit the needs of data + metadata.

RAG and VectorDB: In action and challenges



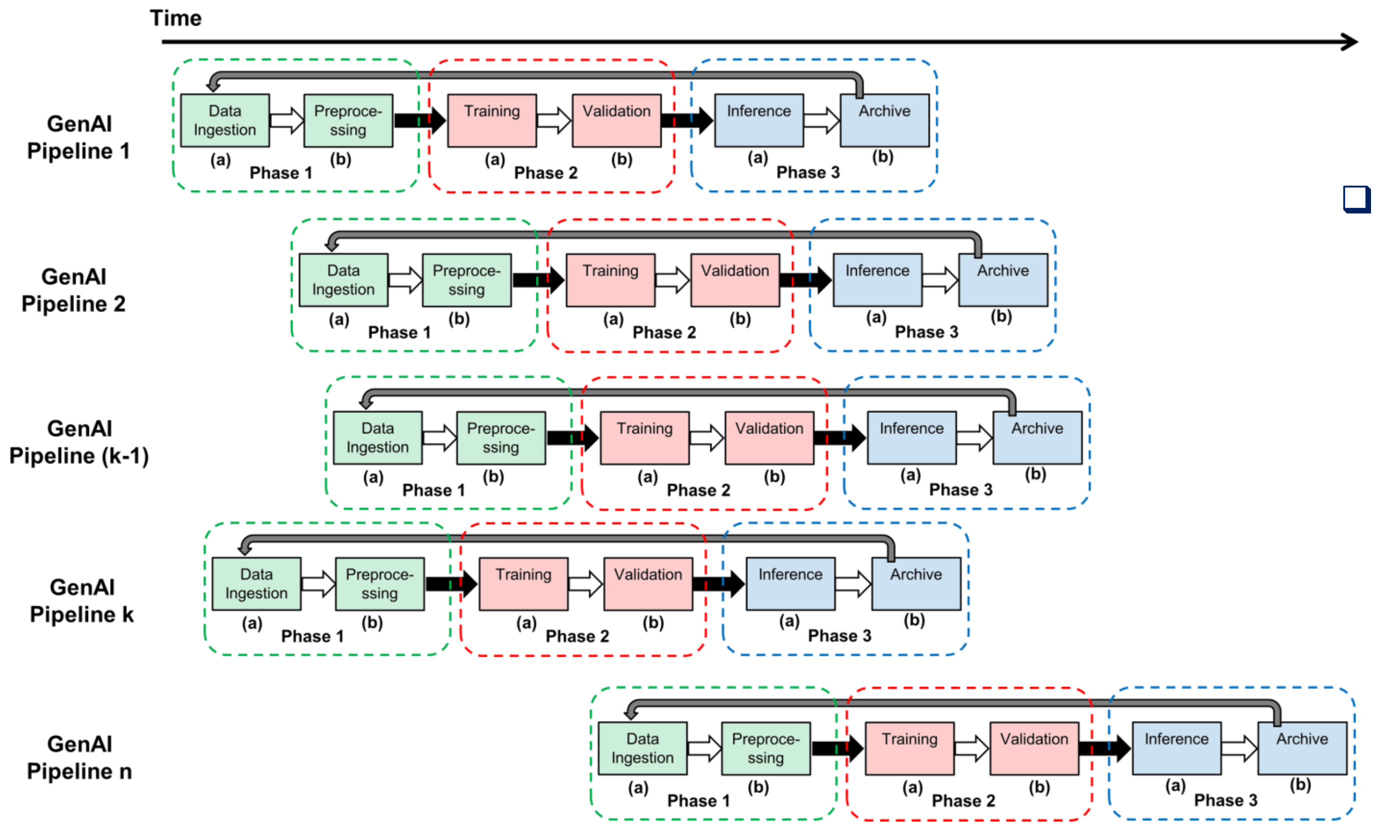
- Data ingestion (logs via streams) : *sequential writes (high BW). Object storage preferred.*
- Embedding model (a) loads: *high read BW and low latency for fast deployment.*
- *Indexing* : create embedding vectors (clustering). *high read/write BW.*

- ***R**etrieval: filtering + search for Vector Search Similarity to find the top_k contexts.*
  - *Transfer large no. of files (data and index) across the network from storage to inference GPU servers.*

  Large amounts of data transfers over network with very high reduction ratio.
  - Storage → CPU BW (requires high storage BW); CPU→GPU BW; GPU-GPU BW for data copy and reduction.

Reduction of data-movement and Host-Storage read BW expansion is essential for faster real-time response.

IO Blender



- Typically, GenAI deployments are shared and used by multiple tenants.
  - Distinct IO features for individual phases are not usually observed, but a *mixed IO profile*.
  - Different phases of multiple AI pipelines execute in parallel.
  - *Performance isolation* to every client (and phase) is quintessential.
  - Extremely *metadata heavy – large number of file ops*.

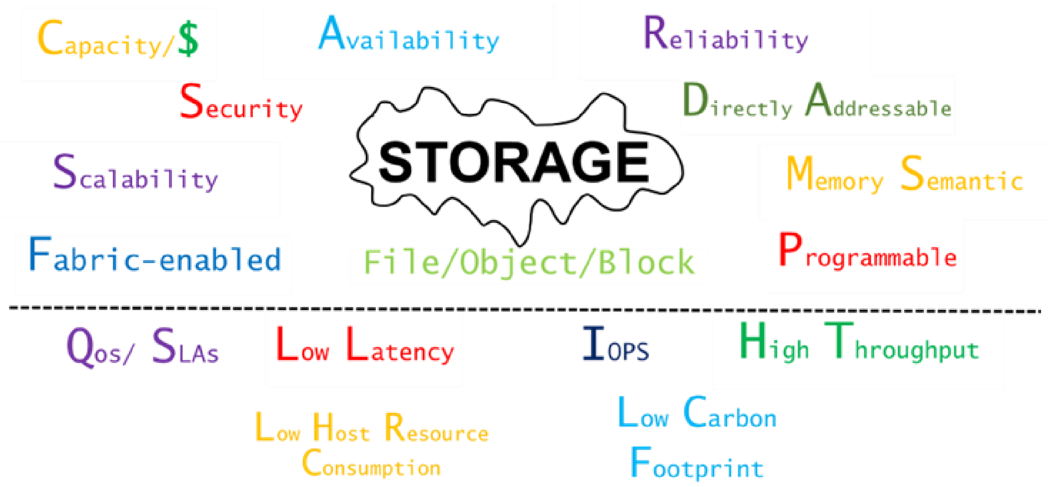
For maximal GPU utilization, performance isolation and SLA guarantees should be ensured by the storage ecosystem.

~~Final~~ Penultimate Thoughts !

- Requirements of AI from storage is nearly *everything and anything* (**just name it or just think, and you need it**).
  - ***Primarily***: $/capacity, RAS, bandwidth (R/W), low-latency, power budget, highly performant metadata management, etc.

- There will be a lot of data, required all the time, at the fastest rate as possible across multiple subsystems.

- Assume there will be failures in the system all the time. Embrace for the traffic storm to/fro storage.



A ***unified storage*** *will be required with performance isolation to capture the needs of the different phases of the AI pipeline(s) (refer image) for large-scale infrastructure.*

Final Thoughts ! (~~Finally,~~ not yet.)

- Complete re-design of end-to-end (GPU optimized) infrastructure.
  - Inter+ Intra GPU node interactions @ scale : *UALink*.
  - Efficient transport mechanism : *Ultra-Ethernet*.
  - Highly optimized GPU-Storage interactions.
    - Direct RDMA like services@scale.
    - Inter-operable Accelerator (GPU) direct interfaces to all kinds of storage.

***Main objective is keeping the highly performant GPUs occupied with datasets at the correct time*.**

- Domain-specific and programmable HW-SW co-design across the entire compute, network and storage stack.
  - *Compute Everywhere and Anywhere*: CPU offloads, in-network computations (NIC), near storage computations (DPUs, CPUs), etc.

- AI Storage topology considerations – converged or not?
  - Balance Storage Foreground vs Background traffic.
  - Define the appropriate transport for storage over NVMe-oF.

Final Thoughts ! (*Finally.*)

Reduce the data-entropy tax, and maximize the utilization of compute, network and storage resources to satiate (or try) the advances of large-scale AI computing for today and tomorrow.

Conclusion

Storage despite being a key-player in AI, is often over-looked and (least talked about **).

As an industry, need to come together to change this **(period!)

COPYRIGHT AND DISCLAIMER

AMD