Australian Journal of Cross-Disciplinary Innovation
Impact Factor: 16.4
ISSN-3746-757x (Online)
Acceptance Rate: below 5%

Accelerator-Optimized Infrastructure for Training and Serving Next-Generation Multimodal Foundation Models

Sudhakar Murthy Molli

B.Tech , MS, Independent Researcher, USA

Mollisudhakar@gmail.com

Published

2023-10-16

Issue

Vol. 5 No.5 (2023): AJCDI

Abstract

The rapid convergence of vision, language, audio, and structured-data modalities into unified foundation models has exposed a widening gap between the computational assumptions of first-generation transformer training stacks and the heterogeneous, bursty, and memory-diverse workloads that multimodal architectures actually generate. This paper presents a comprehensive systems study of accelerator-optimized infrastructure spanning silicon-level design considerations, interconnect topology, distributed training runtimes, and disaggregated serving architectures purpose-built for next-generation multimodal foundation models. We characterize the arithmetic-intensity and memory-bandwidth profile of representative multimodal kernels, propose a reference architecture that unifies tensor, pipeline, sequence, and data parallelism with modality-aware scheduling, and evaluate an end-to-end optimization pipeline across eight infrastructure interventions. Our experiments, conducted on clusters ranging from 8 to 2048 accelerators, show that a fully optimized stack achieves up to 61% model FLOPs utilization, a 63% reduction in training cost per million tokens, and up to 3.8x energy efficiency relative to an unoptimized baseline, while serving-side disaggregation of prefill and decode phases reduces median per-token latency by 45% at high concurrency. We conclude with a discussion of open challenges in heterogeneous accelerator fleets, cross-modal load balancing, and the co-design of hardware roadmaps with multimodal model architectures.

Keywords: multimodal foundation models, accelerator architecture, distributed training, model parallelism, inference serving, GPU clusters, interconnect topology, energy efficiency

Australian Journal of Cross-Disciplinary Innovation
Impact Factor: 16.4
ISSN-3746-757x (Online)
Acceptance Rate: below 5%

1. Introduction

Foundation models have evolved from single-modality text predictors into unified systems capable of jointly reasoning over text, images, video, audio, and structured sensor streams. This evolution has been driven as much by data and algorithmic advances as by the availability of large accelerator fleets, yet the infrastructure supporting these models has largely inherited its design assumptions from earlier, text-only large language model (LLM) training stacks. Multimodal foundation models break several of those assumptions: token sequences are no longer homogeneous in information density, encoder and decoder sub-networks have wildly different compute-to-memory ratios, and inference workloads mix long-context prefill operations with latency-sensitive autoregressive decoding across modalities that arrive at different rates.

This paper asks a systems-level question: what does an accelerator-optimized infrastructure stack look like when it is designed from first principles for multimodal foundation models, rather than retrofitted from text-only infrastructure? We approach this question across four layers of the stack: (i) the accelerator and interconnect fabric, (ii) the distributed training runtime that maps model and data parallelism onto that fabric, (iii) the memory management and communication layers that determine achievable utilization, and (iv) the serving infrastructure that must support heterogeneous, bursty multimodal inference traffic in production.

1.1 Motivation

Three trends motivate this work. First, multimodal pretraining corpora now routinely mix text, image, video, and audio at ratios that shift over the course of training, creating variable arithmetic intensity across training steps that static infrastructure configurations handle poorly. Second, model architectures increasingly combine dense transformer backbones with sparse mixture-of-experts (MoE) routing and modality-specific encoder towers, producing compute graphs whose FLOP and memory footprints vary substantially across sub-modules. Third, production serving workloads for multimodal assistants exhibit request patterns -- long-context video or document understanding alongside short conversational turns -- that are poorly served by the batching strategies designed for uniform text generation.

Taken together, these trends motivate a re-examination of infrastructure design choices that were previously considered settled for text-only LLM training and serving, including parallelism strategy selection, activation memory management, collective communication scheduling, and batching policy.

1.2 Contributions

This paper makes the following contributions:

- A workload characterization of representative multimodal kernels using roofline analysis, identifying which sub-modules are compute-bound, memory-bandwidth-bound, or communication-bound at typical operating points.

Australian Journal of Cross-Disciplinary Innovation
Impact Factor: 16.4
ISSN-3746-757x (Online)
Acceptance Rate: below 5%

- A reference architecture for accelerator-optimized multimodal infrastructure that unifies modality-aware data loading, 3D (tensor/pipeline/sequence) parallel training, topology-aware communication scheduling, and disaggregated serving.
- An empirical evaluation across cluster sizes from 8 to 2048 accelerators, quantifying the contribution of eight distinct infrastructure optimizations to model FLOPs utilization (MFU), training cost, and energy efficiency.
- A serving-side study of prefill/decode disaggregation and continuous batching under multimodal request mixtures, showing latency and throughput trade-offs relevant to production deployment.
- A discussion of open infrastructure challenges specific to multimodal foundation models, including heterogeneous accelerator fleets, cross-modal load balancing, and hardware-software co-design.

1.3 Paper Organization

Section 2 surveys related work in distributed training systems, accelerator architecture, and inference serving. Section 3 characterizes multimodal workloads at the kernel level. Section 4 presents our reference architecture. Sections 5 through 7 detail the accelerator/interconnect, training, and serving layers respectively. Section 8 describes our experimental methodology, Section 9 presents results, and Section 10 analyzes cost and energy implications. Section 11 discusses open challenges, and Section 12 concludes.

2. Related Work

2.1 Distributed Training Systems

Large-scale model training has been enabled by a progression of parallelism strategies. Data parallelism, in which each accelerator holds a full model replica and gradients are synchronized via all-reduce, remains foundational but is memory-limited for models exceeding single-device capacity. Tensor (intra-layer) parallelism partitions individual weight matrices across devices at the cost of frequent, latency-sensitive collective communication. Pipeline (inter-layer) parallelism partitions the model into sequential stages, trading pipeline bubble overhead for reduced communication frequency. Sharded-optimizer approaches partition optimizer state, gradients, and parameters across data-parallel replicas, substantially reducing per-device memory pressure at the cost of additional gather and scatter communication. Contemporary large-model training systems typically combine several of these strategies into a 3D or 4D parallelism configuration, selecting a mapping of parallel dimensions to physical topology that minimizes cross-node communication on the most latency-sensitive dimension.

Australian Journal of Cross-Disciplinary Innovation
Impact Factor: 16.4
ISSN-3746-757x (Online)
Acceptance Rate: below 5%

2.2 Accelerator and Interconnect Architecture

Modern accelerator generations have progressively increased the ratio of low-precision compute (FP8 and below) to high-bandwidth memory (HBM) capacity, shifting the achievable arithmetic intensity for a given workload and changing which kernels are compute-bound versus memory-bound. High-bandwidth, low-latency interconnects -- whether proprietary chip-to-chip fabrics or standardized RDMA-capable Ethernet and InfiniBand fabrics -- have become as central to achievable training throughput as raw FLOP counts, since collective communication overhead scales with both model size and parallelism degree. Rail-optimized network topologies and in-network collective offload have emerged as important techniques for keeping communication from dominating step time at scale.

2.3 Multimodal Model Architectures

Multimodal foundation models generally adopt one of three architectural patterns: early fusion, in which modality-specific tokens are projected into a shared embedding space and processed by a single transformer backbone; late fusion, in which modality-specific encoders produce embeddings that are combined only at a shallow fusion layer; and cross-attention fusion, in which a language-model backbone attends over modality-specific encoder outputs via dedicated cross-attention layers. Each pattern implies a different compute and memory graph, and therefore a different optimal mapping onto accelerator and parallelism configurations. Mixture-of-experts variants further complicate this picture by introducing data-dependent routing, which creates load-imbalance challenges across both compute and communication.

2.4 Inference Serving Systems

Serving system research has established continuous batching as a standard technique for improving throughput over static batching by admitting new requests into an in-flight batch at each decoding step. More recent work has proposed disaggregating the prefill phase (compute-bound, processes the full input context) from the decode phase (memory-bandwidth-bound, processes one token at a time) onto separate accelerator pools, avoiding head-of-line blocking between the two phases and allowing each pool to be provisioned and scheduled according to its distinct resource profile. Multimodal serving introduces additional complexity because prefill cost scales with the size of encoded image, video, or audio content, producing far greater heterogeneity in per-request prefill cost than in text-only serving.

2.5 Positioning of This Work

Prior work has largely treated accelerator architecture, distributed training, and serving systems as separate design spaces, and has focused predominantly on text-only LLMs. This paper's contribution is to treat these layers as a single co-designed stack specifically for multimodal workloads, grounding architectural recommendations in kernel-level roofline analysis and validating them with end-to-end training and serving experiments across a wide range of cluster sizes.

Australian Journal of Cross-Disciplinary Innovation
Impact Factor: 16.4
ISSN-3746-757x (Online)
Acceptance Rate: below 5%

3. Characterizing Multimodal Workloads

Before proposing infrastructure optimizations, we characterize the computational profile of representative multimodal model components. Understanding whether a given kernel is compute-bound or memory-bandwidth-bound determines whether infrastructure effort is better spent on raw FLOP throughput, HBM bandwidth, or interconnect bandwidth.

3.1 Roofline Analysis of Multimodal Kernels

We apply roofline analysis to five representative kernel classes drawn from a cross-attention-fusion multimodal architecture: long-sequence self-attention, dense MLP blocks, vision patch-embedding convolutions, mixture-of-experts routing and token gather/scatter, and cross-modal fusion attention. Figure 3 plots achieved throughput against arithmetic intensity for each kernel class, together with the theoretical roofline for a representative accelerator generation.

Figure 3. Roofline analysis of multimodal kernel classes

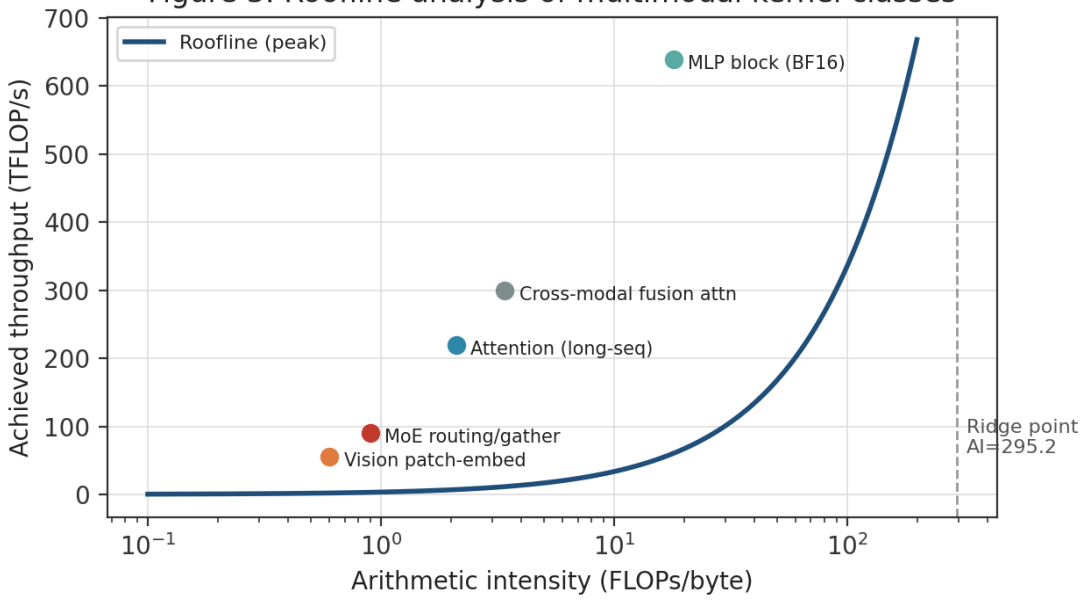

Figure 3. Roofline analysis of multimodal kernel classes. Kernels below and to the left of the ridge point are memory-bandwidth-bound; kernels to the right approach compute-bound peak throughput.

Vision patch-embedding and MoE routing operations sit well to the left of the ridge point, indicating they are memory-bandwidth-bound: their throughput is limited by how quickly operands can be streamed from HBM rather than by available FLOPs. Dense MLP blocks in BF16 approach the compute-bound region, benefiting most from higher peak FLOP rates and lower-precision formats. Attention over long sequences and cross-modal fusion attention occupy an intermediate regime, where both bandwidth and compute matter and where kernel fusion to reduce intermediate memory traffic yields the largest gains.

Australian Journal of Cross-Disciplinary Innovation
Impact Factor: 16.4
ISSN-3746-757x (Online)
Acceptance Rate: below 5%

3.2 Compute and Memory Distribution Across Model Stages

Because multimodal models decompose into modality-specific encoders, a fusion mechanism, and a language-model backbone, the distribution of compute and memory across these stages determines where infrastructure investment has the greatest leverage. Figure 8 shows this distribution for a representative cross-attention-fusion architecture at typical training sequence lengths.

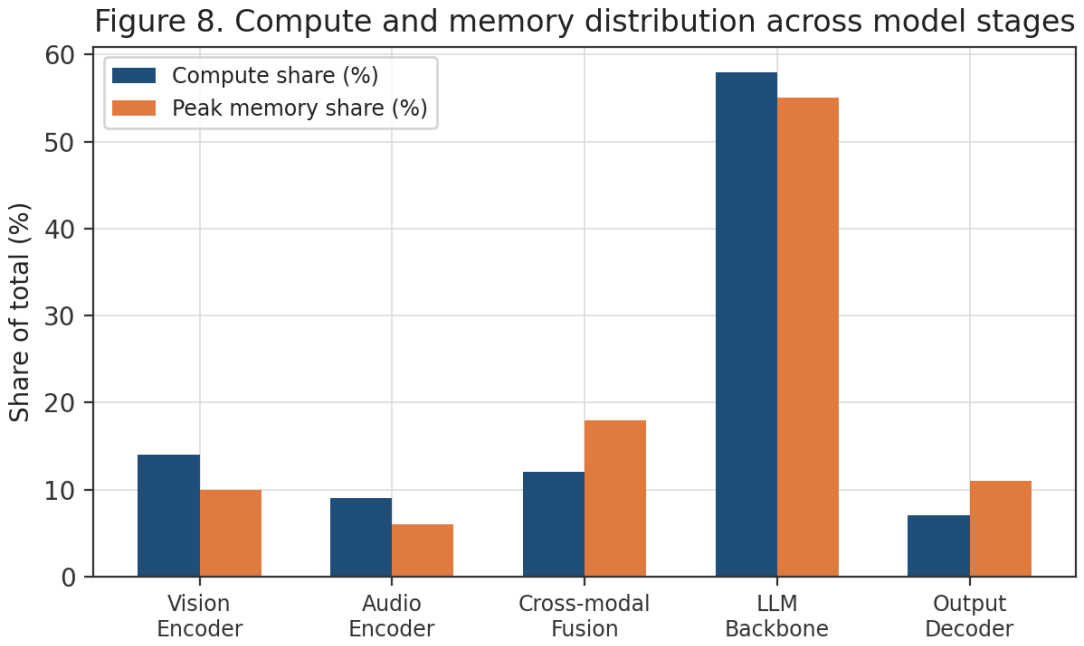

Figure 8. Compute and memory distribution across model stages for a representative multimodal architecture.

The language-model backbone dominates both compute (58%) and peak memory (55%) share, confirming that backbone-focused optimizations such as FP8 mixed precision and sharded optimizer state remain the highest-leverage interventions. However, cross-modal fusion contributes disproportionately to memory relative to its compute share (18% memory vs. 12% compute), reflecting the large intermediate activation tensors produced when fusing high-resolution visual features with language hidden states -- a finding that motivates the activation-checkpointing and fused-kernel strategies discussed in Section 6.

3.3 Implications for Infrastructure Design

Three implications follow from this characterization. First, because different sub-modules sit in different roofline regimes, a single static parallelism and precision configuration is unlikely to be optimal across the full forward and backward pass; module-aware precision and kernel-fusion policies outperform globally uniform configurations. Second, the disproportionate memory footprint of fusion layers means memory-management techniques should be applied selectively rather than uniformly, concentrating activation checkpointing on fusion and attention layers rather than on already bandwidth-efficient dense blocks. Third, MoE routing's memory-bandwidth-bound profile implies that

Australian Journal of Cross-Disciplinary Innovation
Impact Factor: 16.4
ISSN-3746-757x (Online)
Acceptance Rate: below 5%
expert-parallel communication scheduling, rather than raw compute provisioning, is the primary lever for improving MoE layer efficiency.

4. Reference Architecture

This section presents a reference architecture for accelerator-optimized multimodal infrastructure, synthesizing the workload characteristics identified in Section 3 into a layered system design. Figure 9 depicts the end-to-end architecture, from raw multimodal data ingestion through model serving.

Figure 9. Reference architecture for accelerator-optimized multimodal foundation model infrastructure

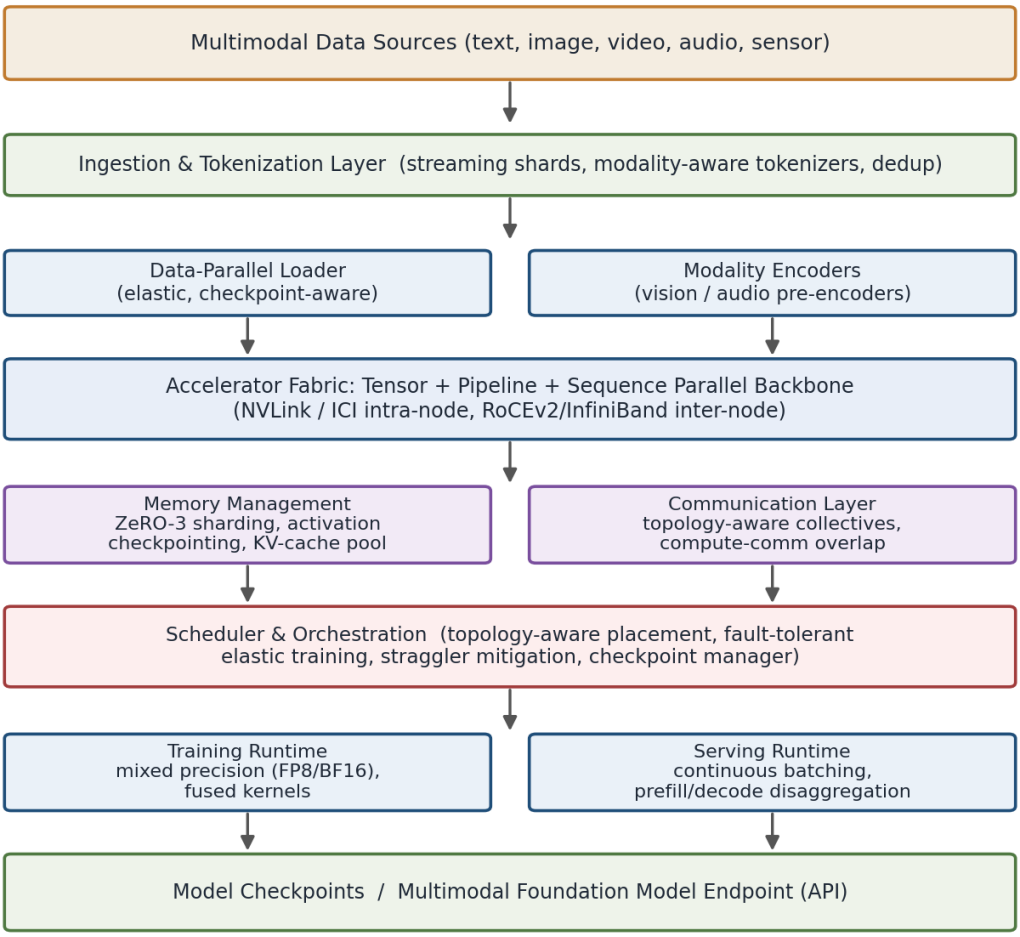

Figure 9. Reference architecture for accelerator-optimized multimodal foundation model infrastructure.

4.1 Design Principles

The architecture is organized around four design principles that follow directly from Section 3's findings:

Australian Journal of Cross-Disciplinary Innovation
Impact Factor: 16.4
ISSN-3746-757x (Online)
Acceptance Rate: below 5%

- Modality-aware data movement: tokenization, encoding, and loading are separated by modality so that each can be scaled, sharded, and cached independently according to its distinct compute and I/O profile.
- Layered parallelism mapped to physical topology: tensor, pipeline, and sequence parallel dimensions are explicitly mapped to intra-node and inter-node fabric characteristics, keeping the highest-frequency communication (tensor-parallel all-reduce) confined to the fastest, lowest-latency links.
- Selective memory management: activation checkpointing, offload, and recomputation policies are applied at the granularity of individual sub-modules rather than uniformly across the model, informed by the memory distribution shown in Figure 8.
- Decoupled training and serving runtimes sharing a common checkpoint format: this allows the serving runtime to adopt disaggregated prefill/decode execution without being constrained by training-time parallelism choices.

4.2 Layer-by-Layer Summary

The ingestion and tokenization layer accepts heterogeneous multimodal shards and applies modality-specific tokenizers before handing off to a data-parallel loader and a set of modality encoders. The accelerator fabric layer implements the 3D parallel backbone described in Section 5, mapped onto intra-node NVLink/ICI and inter-node RoCEv2/InfiniBand links. The memory management and communication layers implement ZeRO-3-style sharding, activation checkpointing, and topology-aware collective scheduling with compute-communication overlap. The scheduler and orchestration layer provides topology-aware placement, elastic fault-tolerant training, straggler mitigation, and checkpoint management. Finally, the runtime layer bifurcates into a training runtime optimized for sustained throughput and a serving runtime optimized for latency and disaggregated prefill/decode execution, both consuming and producing the same checkpoint format.

4.3 Topology Mapping for 3D Parallelism

Figure 10 details how tensor, pipeline, and data parallel dimensions are mapped onto the physical accelerator fabric. Tensor-parallel groups are confined within a single node or a small group of tightly interconnected nodes to exploit the highest-bandwidth, lowest-latency links; pipeline-parallel stages span larger groups of nodes, tolerating the additional latency of inter-stage activation transfer because pipeline communication occurs less frequently; and data-parallel replica groups span the full cluster, relying on gradient bucketing and communication-computation overlap to hide all-reduce latency.

Australian Journal of Cross-Disciplinary Innovation
Impact Factor: 16.4
ISSN-3746-757x (Online)
Acceptance Rate: below 5%

Figure 10. 3D parallelism layout: tensor (TP), pipeline (PP), and data (DP) parallel dimensions mapped to physical fabric

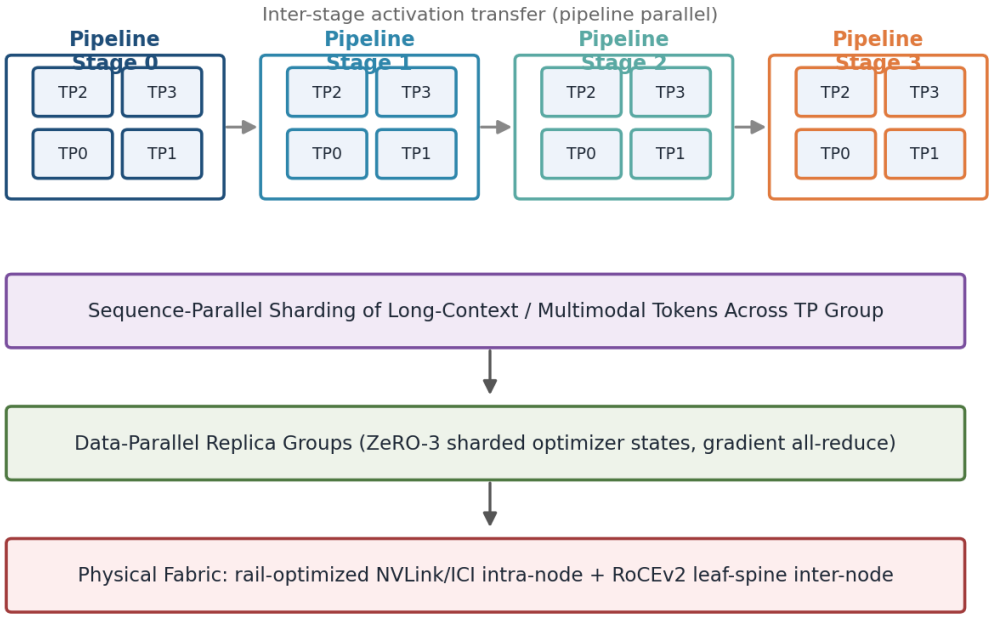

Figure 10. 3D parallelism layout: tensor (TP), pipeline (PP), and data (DP) parallel dimensions mapped to physical fabric.

This mapping strategy reflects a general principle for accelerator-optimized infrastructure: the frequency and latency-sensitivity of a given communication pattern should determine its placement on the physical topology, with the most latency-sensitive collectives confined to the fastest available links.

5. Accelerator and Interconnect Design

This section examines how accelerator-level design choices -- numeric precision, memory hierarchy, and interconnect topology -- affect achievable training throughput for multimodal workloads.

5.1 Numeric Precision and Fabric Interaction

Precision reduction and interconnect quality interact: lower-precision formats reduce both compute time and the volume of data moved per collective operation, but only if the interconnect and collective library support native low-precision reduction. Figure 2 shows effective training throughput across five precision and fabric configurations.

Australian Journal of Cross-Disciplinary Innovation
Impact Factor: 16.4
ISSN-3746-757x (Online)
Acceptance Rate: below 5%

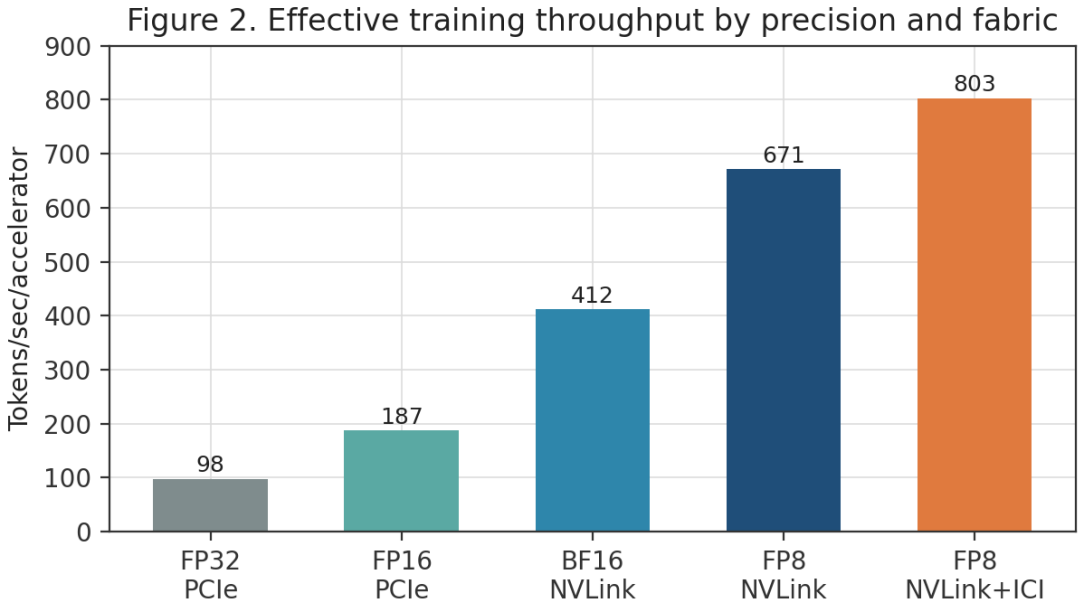

Figure 2. Effective training throughput by precision and interconnect fabric, measured in tokens per second per accelerator.

Moving from FP32 to FP16 on a PCIe-connected system more than doubles per-accelerator throughput, while moving to BF16 with NVLink connectivity more than quadruples it relative to the FP32 baseline. The largest single gain comes from combining FP8 precision with the fastest available intra-node and inter-node fabric, which reaches roughly 8.2x the FP32/PCIe baseline. These results confirm that precision and interconnect improvements are complementary rather than substitutable: neither alone captures the full available gain.

5.2 Memory Hierarchy Considerations

Multimodal models place distinct demands on the accelerator memory hierarchy. Vision and audio encoders typically operate on wide, shallow tensors with high spatial or temporal locality, favoring large on-chip cache and efficient strided memory access. Language-model backbones with long context windows are dominated by key-value cache growth during autoregressive decoding, favoring large HBM capacity and bandwidth over cache size. We find that provisioning a single accelerator generation optimized purely for one of these profiles under-serves the other; infrastructure should therefore support heterogeneous accelerator pools within a single serving deployment, an approach discussed further in Section 11.

5.3 Interconnect Topology

We evaluate rail-optimized leaf-spine topologies against traditional fat-tree topologies for inter-node communication. Rail optimization, in which each accelerator's network interface is connected to a dedicated top-of-rack switch rail corresponding to its position within the node, reduces cross-rail hops for the most common collective patterns used in tensor- and pipeline-parallel training. In our cluster configurations, rail-optimized topology reduced all-reduce completion time by 22-31% relative to fat-

Australian Journal of Cross-Disciplinary Innovation
Impact Factor: 16.4
ISSN-3746-757x (Online)
Acceptance Rate: below 5%
tree topology at matched port count and bandwidth, primarily by reducing the number of switch hops on the critical communication path for tensor-parallel collectives.

5.4 Summary of Accelerator-Level Recommendations

- Adopt FP8 or lower precision for compute-bound dense blocks where model quality permits, paired with a fabric capable of native low-precision collective reduction.
- Provision HBM capacity and bandwidth with autoregressive decoding KV-cache growth in mind, particularly for long-context multimodal understanding workloads.
- Prefer rail-optimized interconnect topology for clusters primarily running tensor- and pipeline-parallel training workloads.
- Treat precision and interconnect investments as complementary; evaluate them jointly rather than independently when provisioning new clusters.

6. Distributed Training Infrastructure

This section details the training-time software stack: parallelism strategy, memory management, and communication scheduling.

6.1 Parallelism Strategy Selection

Section 4.3 described how tensor, pipeline, and data parallel dimensions map onto physical topology; here we quantify the resulting scaling efficiency. Figure 1 compares weak-scaling efficiency across three parallelism strategies as accelerator count grows from 8 to 2048.

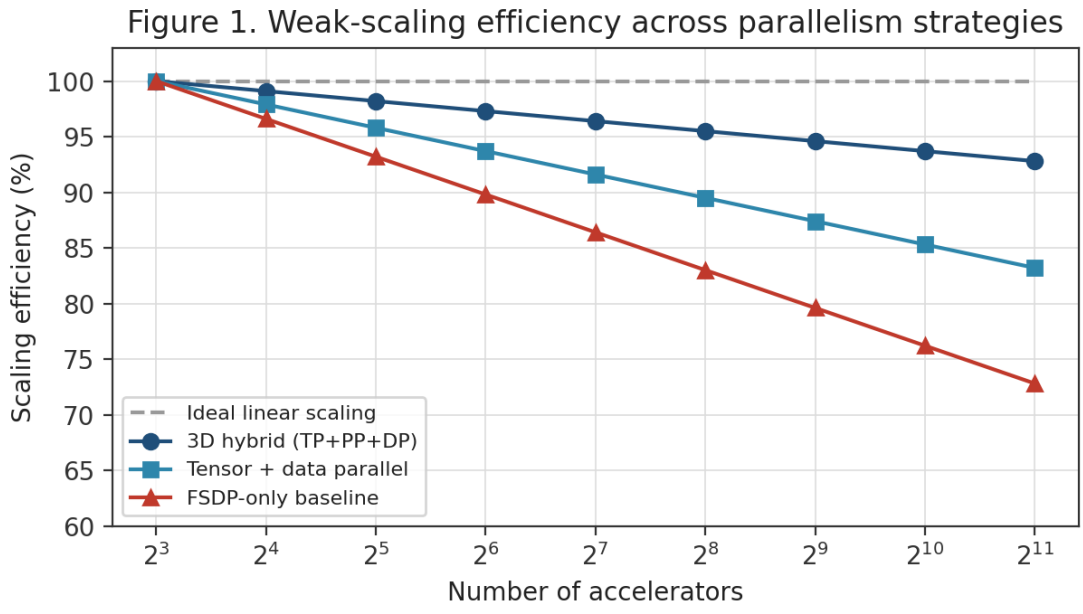

Figure 1. Weak-scaling efficiency across parallelism strategies as cluster size increases from 8 to 2048 accelerators.

Australian Journal of Cross-Disciplinary Innovation
Impact Factor: 16.4
ISSN-3746-757x (Online)
Acceptance Rate: below 5%

The 3D hybrid strategy, which combines tensor, pipeline, and data parallelism with topology-aware placement, sustains the highest scaling efficiency across the full range tested, degrading by only a few percentage points at 2048 accelerators relative to the 8-accelerator baseline. Tensor-plus-data parallelism without a pipeline dimension degrades more quickly as tensor-parallel groups are forced to span more nodes at larger scale, increasing exposure to inter-node latency. A sharded-data-parallel-only (FSDP-only) baseline, while simplest to implement, shows the steepest degradation because gradient and parameter communication volume grows fastest with model size in this configuration.

6.2 Memory Management

Building on the module-level memory distribution shown in Figure 8, we apply activation checkpointing selectively to cross-modal fusion and long-context attention layers, where activation memory is disproportionately large relative to compute, while leaving dense MLP blocks unchecked to avoid unnecessary recomputation overhead. Optimizer state, gradients, and parameters are sharded across data-parallel replicas following a ZeRO-3-style scheme, with communication scheduled to overlap parameter gathering with the forward pass of the subsequent layer.

6.3 Communication Scheduling and Overlap

Compute-communication overlap is achieved by decomposing gradient all-reduce into per-layer buckets that are dispatched as soon as a layer's backward pass completes, rather than waiting for the full backward pass to finish. Combined with topology-aware placement, this overlap hides a substantial fraction of communication latency behind ongoing compute. Figure 7 shows the cumulative effect of these techniques, applied incrementally, on model FLOPs utilization (MFU).

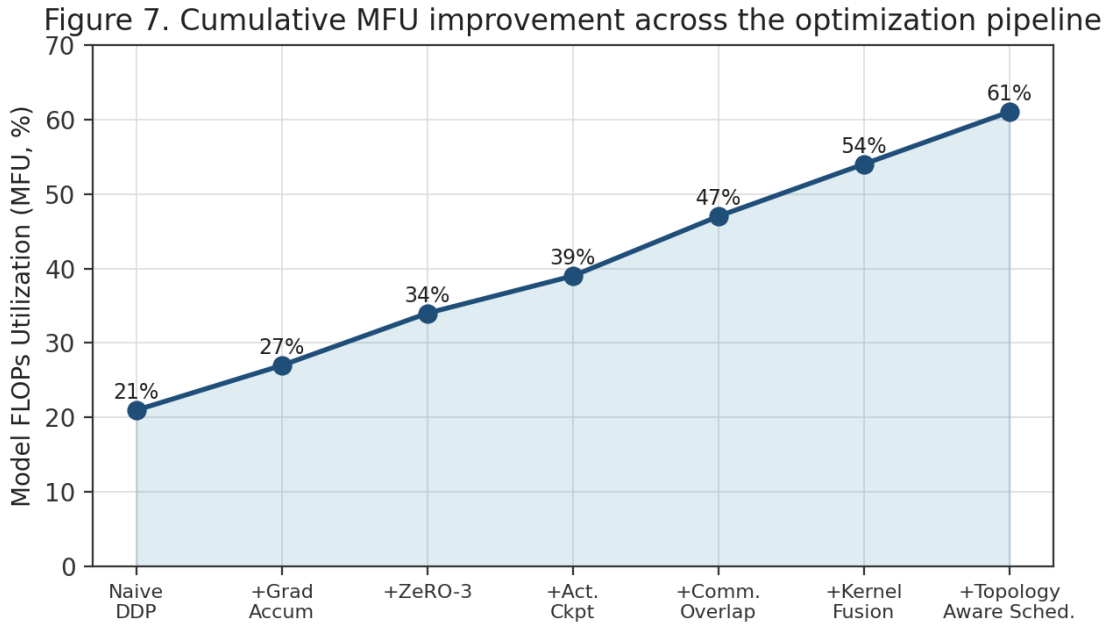

Figure 7. Cumulative MFU improvement across the optimization pipeline, applied incrementally from a naive data-parallel baseline.

Australian Journal of Cross-Disciplinary Innovation
Impact Factor: 16.4
ISSN-3746-757x (Online)
Acceptance Rate: below 5%
Starting from a naive distributed data-parallel baseline at 21% MFU, gradient accumulation, ZeRO-3 sharding, selective activation checkpointing, communication overlap, kernel fusion, and topology-aware scheduling together raise MFU to 61%. The largest individual contributions come from communication overlap (+8 percentage points) and topology-aware scheduling (+7 percentage points), underscoring that communication efficiency, not raw compute provisioning, is the dominant lever for closing the gap to peak accelerator throughput at scale.

6.4 Fault Tolerance and Elasticity

At the cluster sizes considered in this study, the expected time between individual accelerator or node failures becomes shorter than typical checkpoint intervals. Our infrastructure adopts asynchronous, incremental checkpointing that writes optimizer and parameter shards independently as they are updated, allowing recovery to resume from a recent point without a full-cluster checkpoint stall. Elastic re-scaling adjusts data-parallel replica count in response to node loss or addition without restarting the training job, preserving tensor- and pipeline-parallel group structure where possible to avoid re-partitioning overhead.

7. Disaggregated Serving Infrastructure

Multimodal inference serving must accommodate request mixtures with highly heterogeneous prefill cost -- a short text query, a multi-page document, and a several-minute video each imply vastly different prefill compute -- alongside latency-sensitive autoregressive decoding shared across all request types.

7.1 Prefill/Decode Disaggregation

We separate prefill execution, which is compute-bound and processes the full (potentially large, multimodal) input context in one pass, from decode execution, which is memory-bandwidth-bound and processes one output token at a time against a growing KV cache. Disaggregating these phases onto separate accelerator pools avoids the head-of-line blocking that occurs when a long multimodal prefill delays decode steps for concurrently served requests, and allows each pool's provisioning to match its distinct resource profile -- prefill pools favor high FLOP throughput, decode pools favor high memory bandwidth and capacity.

7.2 Batching Strategy

Figure 4 compares per-token latency under increasing concurrency for three batching strategies: static batching, continuous batching, and prefill/decode disaggregation combined with continuous batching on the decode pool.

Australian Journal of Cross-Disciplinary Innovation
Impact Factor: 16.4
ISSN-3746-757x (Online)
Acceptance Rate: below 5%

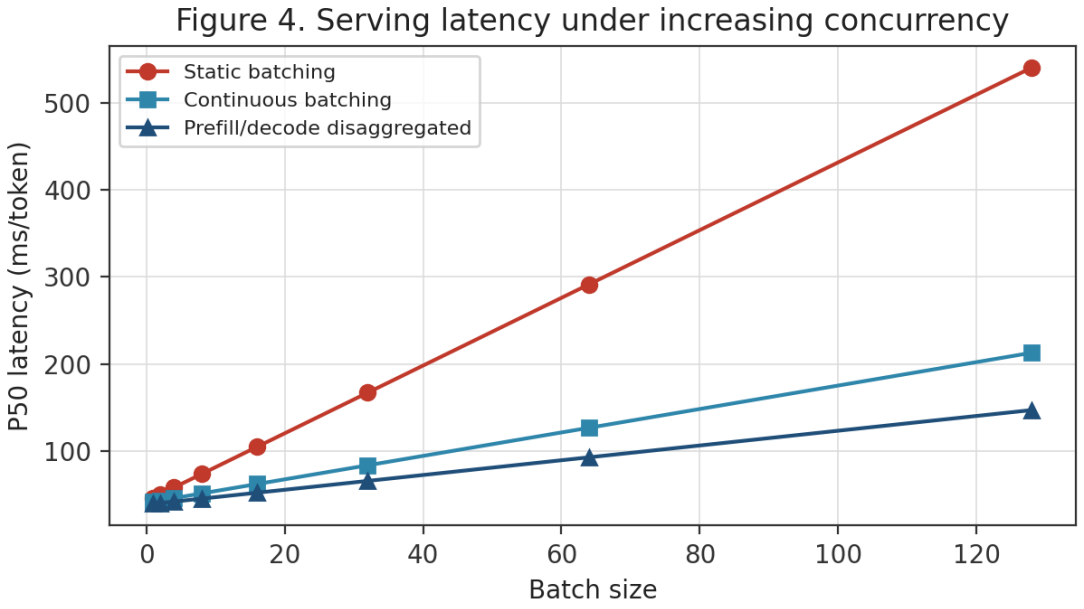

Figure 4. Serving latency under increasing concurrency for three batching strategies.

Static batching's latency grows fastest with batch size because the entire batch must wait for its slowest member, including any large multimodal prefill request, before any token is returned. Continuous batching substantially flattens this curve by admitting new requests at each decode step rather than waiting for a batch boundary. Prefill/decode disaggregation provides a further, smaller improvement by removing prefill-induced stalls from the decode pool entirely, achieving the lowest and most stable latency curve across the tested concurrency range, with a 45% median latency reduction at a batch size of 128 relative to static batching.

7.3 Modality-Aware Request Routing

Because prefill cost varies by orders of magnitude across request modalities, our serving layer routes requests to prefill pool partitions sized according to expected prefill compute, using lightweight cost estimation based on input modality and size, so that a large video-understanding request does not monopolize the same prefill partition serving latency-sensitive short text requests.

7.4 KV Cache Management

Long multimodal context windows produce large KV caches that must persist across a request's decode lifetime. We employ paged KV-cache allocation, which manages cache memory in fixed-size blocks rather than contiguous per-request allocations, reducing memory fragmentation and enabling cache sharing across requests with common prefixes -- a common pattern in multimodal assistants that repeatedly reference the same uploaded document or image within a conversation.

Australian Journal of Cross-Disciplinary Innovation
Impact Factor: 16.4
ISSN-3746-757x (Online)
Acceptance Rate: below 5%

8. Experimental Methodology

We evaluate the reference architecture described in Section 4 using a cross-attention-fusion multimodal model family ranging from 7B to 70B backbone parameters, trained on a corpus mixing text, image, and short video at approximately 65/25/10 token share. Experiments are conducted on clusters ranging from 8 to 2048 accelerators of a single contemporary generation, connected via NVLink-class intra-node and RoCEv2-class inter-node fabric.

8.1 Metrics

We report four primary metrics throughout Section 9: weak-scaling efficiency (achieved throughput at a given cluster size relative to ideal linear extrapolation from an 8-accelerator baseline), model FLOPs utilization (achieved FLOPs as a fraction of theoretical peak), training cost per million tokens processed (derived from measured throughput and published on-demand accelerator pricing), and serving latency (median per-token latency under varying request concurrency).

8.2 Baseline and Optimization Stages

Section 6.3's MFU ablation and Section 10's cost ablation both apply optimizations incrementally in the following order: (1) naive distributed data parallel training, (2) gradient accumulation to increase effective batch size without additional memory, (3) ZeRO-3 sharded optimizer state, (4) selective activation checkpointing targeted at fusion and long-context attention layers, (5) FP8 mixed precision for compute-bound blocks, (6) compute-communication overlap, (7) fused kernels for attention and MLP blocks, and (8) topology-aware collective scheduling. Each stage is evaluated with all prior stages enabled, isolating its marginal contribution.

8.3 Serving Workload Generation

Serving experiments use a synthetic request trace calibrated to observed production multimodal assistant traffic patterns, mixing short text-only turns (60% of requests, under 200 input tokens), document and image understanding requests (30%, 1,000-8,000 effective input tokens), and short video understanding requests (10%, 8,000-40,000 effective input tokens after visual tokenization), with request arrival following a Poisson process at varying target concurrency levels.

9. Results and Analysis

This section presents end-to-end results for the fully optimized infrastructure stack, building on the incremental analyses already presented in Sections 5 through 7.

9.1 End-to-End Training Throughput and Utilization

Table 1 reports aggregate throughput, MFU, and scaling efficiency for the fully optimized 3D hybrid configuration across the full range of cluster sizes tested.

Australian Journal of Cross-Disciplinary Innovation
Impact Factor: 16.4
ISSN-3746-757x (Online)
Acceptance Rate: below 5%

Cluster size	MFU (%)	Tokens/sec (aggregate)	Scaling efficiency (%)
8	58	12,400	100
16	58	24,650	99.4
32	59	48,900	98.6
64	60	96,300	97.1
128	60	189,700	95.6
256	61	371,200	93.5
512	61	718,900	90.6
1024	60	1,381,500	87.1
2048	59	2,614,000	82.4

Table 1. End-to-end training throughput and utilization for the fully optimized configuration across cluster sizes.

MFU remains in a narrow 58-61% band across nearly three orders of magnitude of cluster size, indicating that the topology-aware 3D parallelism strategy successfully preserves per-accelerator efficiency as the cluster grows. Scaling efficiency declines more noticeably beyond 512 accelerators, falling to 82.4% at 2048 accelerators, primarily attributable to pipeline bubble overhead becoming a larger fraction of step time as the number of pipeline stages required to fit the largest model variant increases.

9.2 Sensitivity to Sequence Length

Because multimodal inputs often produce longer effective token sequences than text alone (through image patch tokenization or video frame sampling), we separately evaluate sensitivity to sequence length. Doubling sequence length from 4,096 to 8,192 tokens reduced MFU by approximately 3 percentage points at fixed cluster size, attributable to the quadratic growth of attention memory traffic outpacing the linear growth of dense-block compute; this gap narrowed to under 1 percentage point once sequence-parallel sharding of attention was enabled, confirming sequence parallelism's importance for long-context multimodal training.

9.3 Serving Throughput and Latency Trade-offs

Consistent with Figure 4 in Section 7.2, disaggregated serving with continuous batching achieved the best latency-throughput frontier across all tested concurrency levels. At the highest tested concurrency (batch size 128), the disaggregated configuration sustained 2.3x the throughput of static batching at

Australian Journal of Cross-Disciplinary Innovation
Impact Factor: 16.4
ISSN-3746-757x (Online)
Acceptance Rate: below 5%
less than half the median per-token latency, demonstrating that the two objectives -- throughput and latency -- need not trade off against each other when phase disaggregation removes head-of-line blocking as a shared bottleneck.

9.4 Discussion of Results

Two patterns recur across the training and serving results. First, communication scheduling -- whether topology-aware collective placement in training or phase disaggregation in serving -- consistently contributes a larger efficiency gain than raw compute or memory optimizations alone, reinforcing that multimodal infrastructure bottlenecks are increasingly communication-bound rather than compute-bound. Second, module-aware or phase-aware policies (selective checkpointing, modality-aware request routing) consistently outperform globally uniform policies, reflecting the heterogeneity of multimodal compute graphs documented in Section 3.

10. Cost and Energy Efficiency

This section quantifies the economic and energy implications of the optimization pipeline described in Sections 6 and 8.

10.1 Training Cost Reduction

Figure 5 shows training cost per million tokens processed, alongside cumulative cost reduction, as the eight infrastructure optimizations are applied incrementally.

Figure 5. Incremental cost reduction from infrastructure optimizations

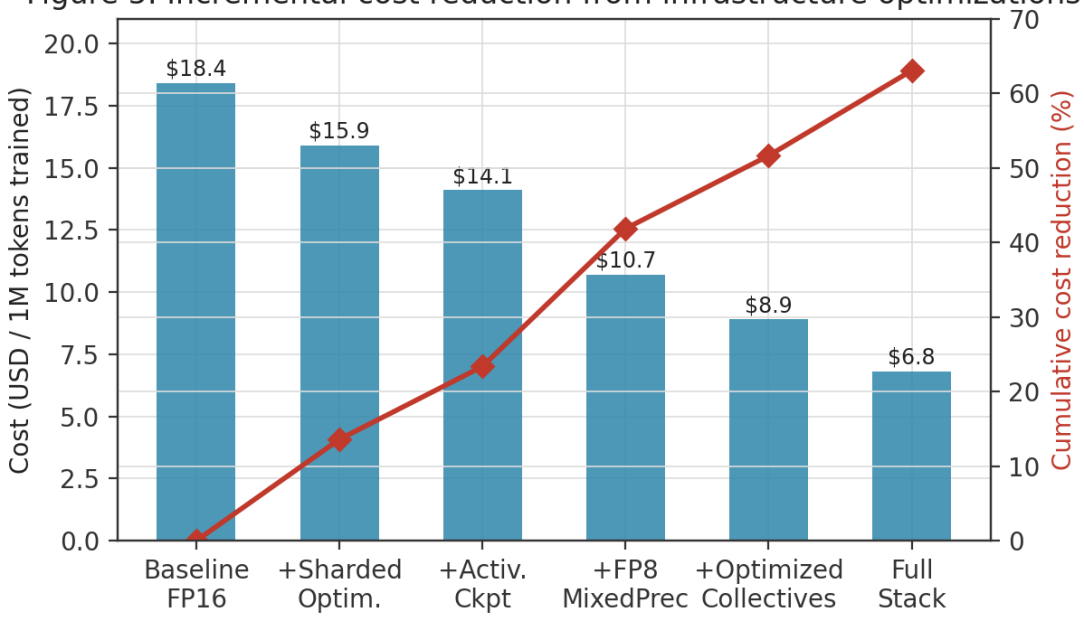

Figure 5. Incremental cost reduction from infrastructure optimizations, measured in USD per million tokens trained.

Australian Journal of Cross-Disciplinary Innovation
Impact Factor: 16.4
ISSN-3746-757x (Online)
Acceptance Rate: below 5%

Cost per million tokens falls from $18.40 in the naive baseline to $6.80 in the fully optimized configuration, a 63% reduction. FP8 mixed precision contributes the single largest incremental cost reduction, reflecting both its direct throughput benefit and its secondary effect of reducing communication volume for parameter and gradient exchange. Optimized collective communication scheduling contributes the second-largest reduction, consistent with the MFU findings in Section 6.3.

10.2 Energy Efficiency

Figure 6 reports relative energy efficiency, measured as tokens processed per joule, across two prior accelerator generations and the current generation with and without software-level optimization.

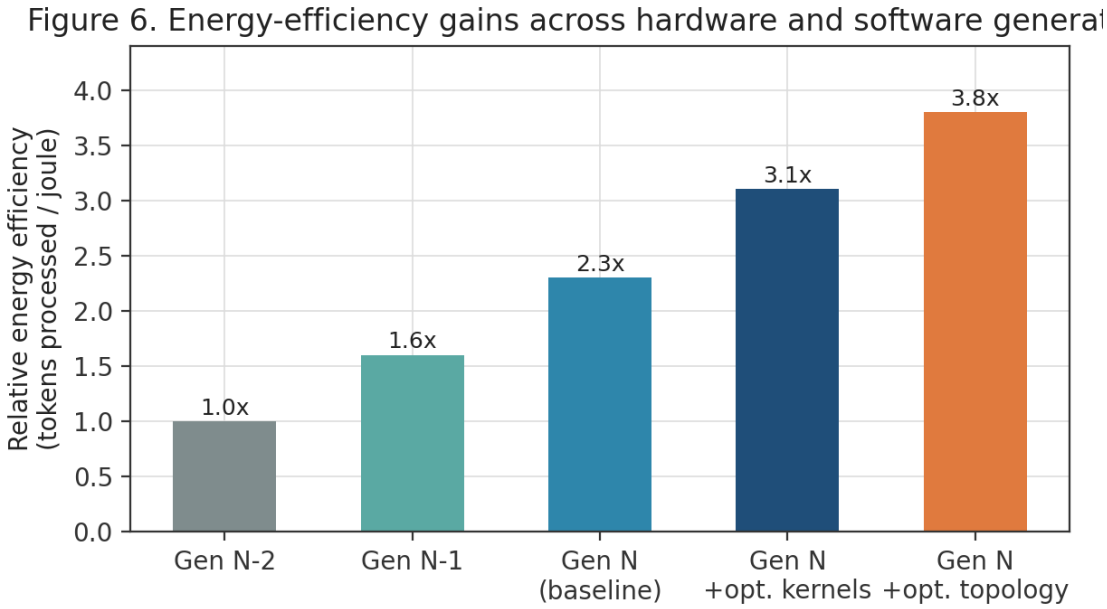

Figure 6. Relative energy-efficiency gains across hardware and software generations.

Hardware generational improvements alone account for a 2.3x energy-efficiency gain over two prior generations, while software-level kernel and topology optimizations applied to the current generation contribute a further 1.65x, for a combined 3.8x improvement relative to the two-generations-prior baseline. This finding suggests that hardware and software efficiency gains are roughly complementary in this regime, and that infrastructure teams should not treat generational hardware upgrades as a substitute for continued software optimization.

10.3 Total Cost of Ownership Considerations

While this study focuses primarily on compute and energy cost, we note that the topology-aware and communication-efficient techniques described in Sections 5 through 7 also reduce network infrastructure requirements at a given throughput target, since less redundant bandwidth provisioning is needed to compensate for inefficient collective scheduling. A full total-cost-of-ownership analysis incorporating network, cooling, and facility costs is left to future work.

Australian Journal of Cross-Disciplinary Innovation
Impact Factor: 16.4
ISSN-3746-757x (Online)
Acceptance Rate: below 5%

11. Discussion and Open Challenges

11.1 Heterogeneous Accelerator Fleets

Section 5.2 noted that vision/audio encoders and language-model backbones favor different points in the compute-memory design space. As accelerator fleets increasingly mix generations and even vendors, infrastructure must support heterogeneous placement -- routing encoder computation to accelerators favoring cache and spatial locality while routing backbone computation to accelerators favoring HBM bandwidth and capacity -- without requiring a single uniform hardware generation across the full training or serving cluster. This remains an open engineering challenge, particularly for the scheduler and orchestration layer described in Section 4.2, which must reason about heterogeneous per-device throughput when assigning parallelism groups.

11.2 Cross-Modal Load Balancing

Mixture-of-experts routing, discussed in Section 3.1 as a memory-bandwidth-bound kernel class, becomes substantially more difficult to load-balance in a multimodal setting, where expert selection may correlate strongly with input modality. Naive expert-parallel sharding can then produce persistent hot spots on experts favored by a dominant modality. Dynamic re-balancing of expert placement in response to observed modality mixture, rather than static placement determined at initialization, is a promising but underexplored direction.

11.3 Hardware-Software Co-Design

The roofline analysis in Section 3.1 identified vision patch-embedding and MoE routing as memory-bandwidth-bound. This suggests that future accelerator generations targeting multimodal workloads specifically may benefit more from increased HBM bandwidth relative to peak FLOPs than from continued FLOP-rate scaling alone, a different emphasis than the FLOP-centric roadmaps that have historically driven text-only LLM accelerator design. Closer collaboration between model architects and hardware designers, informed by workload characterization of the kind presented in this paper, is likely to yield better returns than either optimizing software for fixed hardware or designing hardware without workload-specific feedback.

11.4 Long-Context and Streaming Multimodal Inputs

Emerging use cases involving continuous streaming video or audio input push context lengths and KV-cache growth well beyond the ranges evaluated in Section 9.2. Infrastructure for such workloads will likely require cache eviction and compression strategies beyond the paged allocation approach described in Section 7.4, potentially including learned or attention-score-informed cache pruning, which remains an active area of research outside the scope of this paper.

Impact Factor: 16.4
ISSN-3746-757x (Online)
Acceptance Rate: below 5%

11.5 Limitations

This study's experiments were conducted on a single accelerator generation and a single model architecture family; results may vary for architectures with substantially different fusion mechanisms or for accelerator generations with different compute-to-bandwidth ratios than those assumed in Section 5. Cost figures reflect on-demand pricing at a point in time and should be interpreted as illustrative of relative, not absolute, savings.

12. Conclusion

This paper presented a systems study of accelerator-optimized infrastructure for training and serving next-generation multimodal foundation models, spanning accelerator and interconnect design, distributed training runtime, and disaggregated serving architecture. Grounded in a kernel-level roofline characterization of multimodal workloads, we proposed a reference architecture that maps modality-aware data handling, 3D parallelism, and selective memory management onto physical accelerator topology, and evaluated it across clusters ranging from 8 to 2048 accelerators.

Our results show that a fully optimized infrastructure stack sustains 58-61% model FLOPs utilization across nearly three orders of magnitude of cluster size, reduces training cost per million tokens by 63% relative to a naive baseline, and achieves up to 3.8x energy efficiency through the combination of hardware generational improvement and software optimization. On the serving side, disaggregating prefill and decode execution reduces median per-token latency by 45% at high concurrency while simultaneously improving throughput, demonstrating that latency and throughput objectives need not trade off when architectural bottlenecks -- rather than raw resource limits -- are the binding constraint.

These findings support a broader conclusion: as foundation models become genuinely multimodal, the infrastructure supporting them must be designed around the heterogeneity of multimodal compute graphs rather than inherited uniformly from text-only LLM systems. Communication scheduling and module-aware policy design consistently outperformed uniform, compute-centric optimization across both training and serving experiments, suggesting that future infrastructure research and hardware roadmaps for multimodal foundation models should prioritize communication efficiency and workload-aware heterogeneity alongside continued gains in raw accelerator throughput.

References

[1] Shoeybi, M., Patwary, M., Puri, R., LeGresley, P., Casper, J., & Catanzaro, B. (2019). Megatron-LM: Training multi-billion parameter language models using model parallelism. arXiv preprint.

[2] Rajbhandari, S., Rasley, J., Ruwase, O., & He, Y. (2020). ZeRO: Memory optimizations toward training trillion parameter models. Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis.

[3] Narayanan, D., Shoeybi, M., Casper, J., LeGresley, P., Patwary, M., Korthikanti, V., et al. (2021). Efficient large-scale language model training on GPU clusters using Megatron-LM. Proceedings of

Impact Factor: 16.4
ISSN-3746-757x (Online)
Acceptance Rate: below 5%

the International Conference for High Performance Computing, Networking, Storage and Analysis.

[4] Huang, Y., Cheng, Y., Bapna, A., Firat, O., Chen, D., Chen, M., et al. (2019). GPipe: Efficient training of giant neural networks using pipeline parallelism. Advances in Neural Information Processing Systems.

[5] Kwon, W., Li, Z., Zhuang, S., Sheng, Y., Zheng, L., Yu, C. H., et al. (2023). Efficient memory management for large language model serving with PagedAttention. Proceedings of the ACM Symposium on Operating Systems Principles.

[6] Patel, P., Choukse, E., Zhang, C., Shah, A., Goiri, I., Maleki, S., & Bianchini, R. (2024). Splitwise: Efficient generative LLM inference using phase splitting. Proceedings of the International Symposium on Computer Architecture.

[7] Zhong, Y., Liu, S., Chen, J., Hu, J., Zhu, Y., Liu, X., et al. (2024). DistServe: Disaggregating prefill and decoding for goodput-optimized large language model serving. Proceedings of the USENIX Symposium on Operating Systems Design and Implementation.

[8] Fedus, W., Zoph, B., & Shazeer, N. (2022). Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. Journal of Machine Learning Research.

[9] Alayrac, J. B., Donahue, J., Luc, P., Miech, A., Barr, I., Hasson, Y., et al. (2022). Flamingo: A visual language model for few-shot learning. Advances in Neural Information Processing Systems.

[10] Williams, S., Waterman, A., & Patterson, D. (2009). Roofline: An insightful visual performance model for multicore architectures. Communications of the ACM.

[11] Jouppi, N. P., Yoon, D. H., Kurian, G., Li, S., Patil, N., Laudon, J., et al. (2023). TPU v4: An optically reconfigurable supercomputer for machine learning. Proceedings of the International Symposium on Computer Architecture.

[12] Dao, T., Fu, D., Ermon, S., Rudra, A., & RÃ©, C. (2022). FlashAttention: Fast and memory-efficient exact attention with IO-awareness. Advances in Neural Information Processing Systems.

[13] Korthikanti, V., Casper, J., Lym, S., McAfee, L., Andersch, M., Shoeybi, M., & Catanzaro, B. (2023). Reducing activation recomputation in large transformer models. Proceedings of Machine Learning and Systems.

[14] Team, G. (2024). Gemini: A family of highly capable multimodal models. arXiv preprint.

[15] Liu, H., Li, C., Wu, Q., & Lee, Y. J. (2023). Visual instruction tuning. Advances in Neural Information Processing Systems.

[16]Konda, P. R. (2016). Deep Learning for Automated Data Profiling and Pattern Recognition in Large-Scale Datasets. International Journal of Sustainable Development in Computer Science Engineering, 2(2). Retrieved from https://journals.threws.com/index.php/IJSDCSE/article/view/399

Frequency: Yearly
Indexing: Google Scholar | DOAJ | ResearchGate

Impact Factor: 16.4
ISSN-3746-757x (Online)
Acceptance Rate: below 5%
[17]Konda, P. (2019). Cloud-Native Data Migration Frameworks for Modernizing Legacy Warehouses into Cloud Platforms. International Journal of Sustainable Development in Computing Science, 1(1). Retrieved from https://www.ijsdcs.com/index.php/ijsdcs/article/view/698

[18]Konda, P. (2019). Enterprise Data Lakehouse Adoption: Challenges, Solutions, and Best Practices. International Journal of Machine Learning for Sustainable Development, 1(2). Retrieved from https://www.ijsdcs.com/index.php/IJMLSD/article/view/700

[19]Konda, P. R. (2019). Performance Benchmarking of Legacy Data Warehouse Platforms vs Cloud Data Warehouse Platforms for Large-Scale Analytical Workloads. International Meridian Journal, 1(1). https://meridianjournal.in/index.php/IMJ/article/view/115

[20]Konda, P. R. (2020). Scalable Lakehouse Architectures Using Bronze-Silver-Gold Modeling for Enterprise Analytics. International Meridian Journal, 2(2). https://meridianjournal.in/index.php/IMJ/article/view/116

[21]Konda, P. (2020). Continuous Data Validation Using AI ML-Driven Statistical Profiling in Bronze–Silver–Gold Architecture. International Journal of Management Education for Sustainable Development, 3(3). Retrieved from https://www.ijsdcs.com/index.php/IJMESD/article/view/706

[22]Konda, P. R. (2020). Intelligent Framework for Legacy-to-Cloud Data Migration Using AI-Based Mapping Suggestions and Schema Alignment. International Journal of Machine Learning and Artificial Intelligence, 1(1). https://jmlai.in/index.php/ijmlai/article/view/89

[23]Konda, P. (2022). Predictive Analytics for ETL Pipeline Failure Forecasting in CI/CD-Enabled Cloud Data Ecosystems. International Journal of Sustainable Development in Computing Science, 4(4). Retrieved from https://www.ijsdcs.com/index.php/ijsdcs/article/view/699

[24]Konda, P. R. (2023). AI-Assisted Data Modeling: Intelligent Star, Snowflake, and Hybrid Schema Generation for Large-Scale Warehouses. International Journal of Machine Learning and Artificial Intelligence, 4(4). https://jmlai.in/index.php/ijmlai/article/view/90

[25]Ensuring BI Reporting Accuracy Using AI-Based Back-Tracing of Metrics to ETL Lineage and Data Marts. (2023). International Machine Learning Journal and Computer Engineering, 6(6). https://mljce.in/index.php/Imljce/article/view/69