Rearchitecting Datacenter Lifecycle for AI: A TCO-Driven Framework

Jovan Stojkovic
University of Illinois
Urbana-Champaign, USA
jovans2@illinois.edu

Chaojie Zhang
Microsoft Azure Research
Redmond, USA
chaojiezhang@microsoft.com

Íñigo Goiri
Microsoft Azure Research
Redmond, USA
inigog@microsoft.com

Ricardo Bianchini
Microsoft Azure
Redmond, USA
ricardob@microsoft.com

Abstract

The rapid rise of large language models (LLMs) has driven an enormous demand for AI inference infrastructure, mainly powered by high-end GPUs. While these accelerators offer immense computational power, they incur high capital and operational costs due to frequent upgrades, dense power consumption, and cooling demands, making total cost of ownership (TCO) for AI datacenters a critical concern for cloud providers.

Unfortunately, traditional datacenter lifecycle management (designed for general-purpose workloads) struggles to keep pace with AI's fast-evolving models, rising resource needs, and diverse hardware profiles. In this paper, we rethink the AI datacenter lifecycle scheme across three stages: building, hardware refresh, and operation. We show how design choices in power, cooling, and networking *provisioning* impact long-term TCO. We also explore *refresh* strategies aligned with hardware trends. Finally, we use *operation* software optimizations to reduce cost.

While these optimizations at each stage yield benefits, unlocking the full potential requires rethinking the entire lifecycle. Thus, we present a holistic lifecycle management framework that coordinates and co-optimizes decisions across all three stages, accounting for workload dynamics, hardware evolution, and system aging. Our system reduces the TCO by up to 40% over traditional approaches. Using our framework we provide guidelines on how to manage AI datacenter lifecycle for the future.

1 Introduction

Generative LLMs are reshaping industries, from education [9] and healthcare [90] to software development [28] and scientific research [117]. Their rapid adoption is driven by the ability to perform complex reasoning, summarization, and interactive tasks with minimal supervision, creating unprecedented demand for scalable AI inference infrastructure [58].

Modern LLM inference relies on high-end GPUs (*e.g.*, NVIDIA's A100 [77] and H100 [76]) which offer strong performance but come with steep financial and infrastructure costs. For example, a single NVIDIA DGX H100 server can exceed $200,000 [18] and draw up to 10.2kW [89, 100], pushing demands for power and cooling capacity far beyond those of traditional CPU servers [75]. To support these workloads, cloud providers have built specialized datacenters for high-throughput inference, making AI-serving one of the most resource-intensive and costly datacenter operations [74].

Researchers have proposed software and hardware techniques that improve the performance [2, 19, 57, 71, 89, 113, 119, 121] or energy-efficiency [93, 97, 99, 100] of LLM inference clusters. However, for providers, the key challenge is minimizing the TCO over the datacenter lifecycle, spanning CapEx (*e.g.*, infrastructure build-out) and OpEx (*e.g.*, energy) while meeting performance expectations from their users.

Traditional datacenter practices (*e.g.*, regular refresh cycles [41] and conservative provisioning [46, 61, 98, 115]) fall short for AI workloads, where models grow rapidly in complexity and scale [24], hardware has higher cost and infrastructure demands [18, 50], and inference is highly sensitive to latency and model quality [99, 100].

Our Work. To address this challenge, we first break down datacenter lifecycle into stages: *build*, *IT provisioning*, and *operation*, each with distinct costs and optimization opportunities. The *build* stage covers initial infrastructure setup, including provisioning decisions about power topology (*e.g.*, flat [7] *vs.* hierarchical [26, 115]), cooling technology (*e.g.*, air *vs.* liquid [32, 43]), and network configurations (*e.g.*, NVLink [85] *vs.* Ethernet). The *IT provisioning* stage governs when and how to decommission and upgrade hardware. The *operation* stage focuses on runtime management: workload placement, scheduling, and software optimizations.

We introduce a framework that rethinks and rearchitects the entire lifecycle for AI datacenters, by exploring alternative strategies across all stages and identifies the most cost-effective combination. In *build*, we compare emerging infrastructure designs to understand and balance long-term scalability, efficiency, and performance. In *IT provisioning*, we evaluate when to adopt new hardware and retire old systems, assessing the impact on TCO given AI's distinct model and hardware characteristics. In *operation*, we assess the impact of software techniques (*e.g.*, model migration, LLM inference disaggregation, and workload scheduling) over lifecycle TCO.

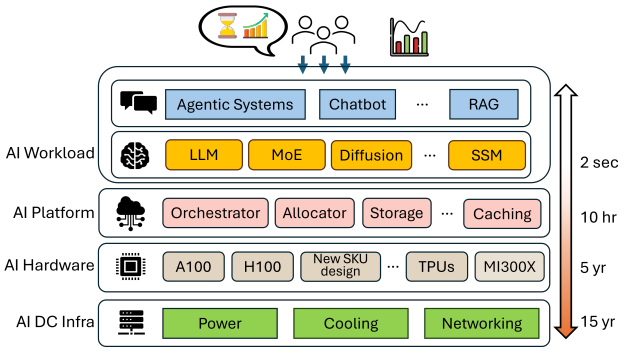


Figure 1. Hosting AI workloads from models to hardware and supporting datacenter infrastructure.

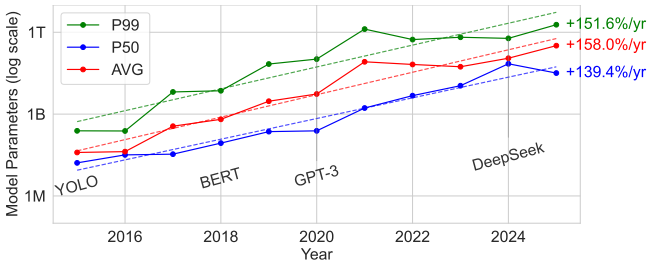


Figure 2. The P50, P99, and average size of the most popular AI models published in the last decade.

These stages are interdependent and our framework captures cross-stage interactions to support lifecycle-aware decisions. Leveraging workload growth trends, hardware road maps, and cost models, it projects future scenarios and selects strategies that satisfy performance, fault-tolerance, and accuracy requirements. For example, investing in a larger power-sharing domain is more expensive at *build* time but provides flexibility for future *IT provisioning* and improves utilization during *operation*.

We build our TCO model using open-source LLMs, public hardware specs, and detailed cost data from public sources. Stage-specific optimizations reduce TCO by 15% (*build*), 23% (*IT provisioning*), and 19% (*operation*). Our cross-stage strategy achieves up to a 40% reduction. Looking ahead, we identify emerging cross-stage opportunities and provide guidelines for adapting AI datacenter lifecycle management to future model and hardware trajectories.

Summary. This paper makes three main contributions:

- Characterization of LLM workload and GPU performance-power interactions across datacenter lifecycle stages.
- Evaluation of stage-wise strategies that improve efficiency and cross-generation compatibility.
- The first-of-its-kind framework for lifecycle-aware, cross-stage optimization of AI datacenters.

2 Hosting AI Workloads

Figure 1 shows the stack required to host AI workloads within a cloud provider: from datacenter infrastructure and specialized hardware to the workloads. To minimize the TCO of AI datacenters, we start at the top of this stack by analyzing AI workloads and reasoning about the demands they place on the underlying hardware and infrastructure.

2.1 AI Workloads

Nowadays, cloud providers host a wide range of AI workloads, spanning large language models (LLMs), vision and multimodal models, speech, recommendation systems, and classical deep neural networks (DNNs) [16, 42, 55]. These workloads differ substantially in compute complexity, memory footprint, performance and accuracy targets, and input modalities. The largest difference is between training and inference workloads: training demands high-bandwidth memory, fast interconnects, and fault-tolerant checkpointing, while inference workloads range from latency-sensitive, memory-bound LLMs at small batch sizes to throughput-oriented vision and recommendation pipelines.

In this paper, we focus on LLM inference, which is rapidly becoming the dominant workload in AI datacenters [16, 55, 88, 89]. For this workload, the most critical factors for datacenter build and provisioning are the size and architecture of the models (which drive compute and memory needs) and the user demand (the load that the system must sustain).

AI Model Trends. These workloads have rapidly evolved in scale, architecture, and demand over the past decade.

Scale. Models have grown dramatically in size, demanding far more compute, memory, and interconnect bandwidth. Larger models drive higher FLOP requirements for inference and require larger on-device memory with greater bandwidth to store weights and intermediate results. Once models exceed the capacity of a single GPU, multi-GPU setups become necessary, connected by high-bandwidth, low-latency links to exchange activations efficiently. Figure 2 shows this trend: model sizes have increased exponentially over the past decade, from Google Neural Machine Translation's ≈200M parameters in 2016 [110], to GPT-3's 175B parameters in 2020 [10], and most recently to Llama 4 Behemoth, which exceeds 2T parameters [69].

However, recent trends indicate a slowdown, with growth shifting to linear [25] and potentially becoming sublinear or even flat in the medium term. This plateau reflects diminishing returns from traditional LLM scaling and reduced gains in training efficiency. Several emerging approaches are attracting attention as alternatives to pure model scaling. For example, distillation transfers knowledge into smaller, more efficient models [112]; and reasoning models leverage time-test-compute to enhance capabilities without relying on parameter growth [13, 39].

Architecture. Model architecture largely determines how much computation and memory bandwidth an inference workload

needs. Transformer-based models dominate current deployments [104]. Their attention layers scale quadratically in compute and memory with sequence length, while large-kernel matrix multiplications (GEMM) demand high FLOP throughput and sustained memory bandwidth. Emerging alternatives like state-space models (SSMs) [38] replace attention with convolution-like operations, reducing memory footprint and improving scalability for long contexts.

Mixture-of-Experts (MoE) [95] activate only a subset of experts per query, lowering the average compute cost but increasing the memory and network demands due to expert sharding. Complementing these designs, fine-tuning techniques such as low-rank adaptation (LoRA) [47] update only a small subset of parameters, reducing memory and bandwidth needs for model updates.

Despite architectural differences, the core computation across modern AI models reduces to large matrix multiplications. This commonality enables a unified performance modeling framework for AI inference—unlike traditional general-purpose datacenters, where heterogeneous workloads (*e.g.*, databases, web services, and scientific applications) made unified modeling impractical. Even as emerging applications adopt agentic systems that orchestrate multiple LLMs to solve complex, multi-step tasks [1, 12], they still rely on foundational models for each component stage, preserving a consistent underlying computational structure.

User Demand. The global AI market is projected to grow from $638B in 2024 to over $3.68T by 2034 [123], with U.S. generative AI expected to see a 36.3% CAGR through 2030 [33]. This growth is driving increased inference workloads, which already dominate AI operational costs [88]. Unlike training, inference incurs higher cumulative costs due to continuous, large-scale deployment serving millions of queries daily [58]. Cloud providers like Microsoft, Amazon, and Google report 15–25% year-over-year growth in AI workloads [56, 91, 108], reflecting rising user demand and the shift toward scalable, cost-efficient inference infrastructure.

2.2 Hardware for AI Workloads

Accelerators. AI inference workloads are both compute- and memory-intensive, relying heavily on GEMM and attention mechanisms. These kernels require high floating-point throughput and high-bandwidth memory (HBM) to sustain performance. Modern accelerators (*e.g.*, GPUs [4, 76, 77], TPUs [55], and NPUs [73, 111]) combine massive parallelism with optimized memory systems that offer both large capacity and sufficient bandwidth to feed compute efficiently.

Interconnects. AI servers rely on high-speed interconnects to fully utilize accelerators. These links—electrical or optical, over copper, active optical cables, or co-packaged optics—differ in bandwidth, latency, and energy efficiency. *Intra-node* connections (PCIe, NVLink [85]) enable fast device communication within a server, while *inter-node* networks (InfiniBand, RoCE [86]) are critical for scaling large models across servers. Efficient collective operations (*e.g.*, all-reduce, all-to-all) are essential to synchronize activations, gradients, and KV-cache data without stalling computation [30, 81].

2.3 Datacenter Infrastructure for AI Hardware

The high performance of modern AI accelerators comes with high demands on supporting infrastructure. Power, cooling, and networking requirements for large-scale AI deployments far exceed those of traditional datacenters.

Power. High-end accelerators consume hundreds of watts per device. For example, an NVIDIA DGX H100 server with eight GPUs has a thermal design power (TDP) of 10.2kW [76]. As a result, rack densities can range from several kilowatts to over 100 kW per rack [17]. Sustaining such loads requires robust *power delivery* systems, including high-capacity uninterruptible power supplies (UPS) and power distribution unit (PDU) topologies, often with oversubscription to balance utilization and provisioning costs [115]. In addition, rapid workload fluctuations can induce large transient currents [14, 62], requiring careful electrical design and monitoring to maintain stability.

Cooling. The heat output of dense AI clusters quickly exceeds the practical limits of traditional air cooling [99]. To maintain performance and reliability, operators increasingly adopt advanced *cooling* solutions such as rear-door heat exchangers, direct-to-chip liquid cooling, and, in ultra-dense deployments, full liquid cooling [79]. While these technologies enable sustained operation, they also require specialized facility layouts, coolant distribution systems, and higher maintenance complexity.

Networking. Large-scale AI inference also stresses *networking* infrastructure. Models using tensor, pipeline, or expert parallelism rely on low-latency, high-bandwidth communication between thousands of accelerators [55]. This drives adoption of specialized network topologies (*e.g.*, fat-tree, dragonfly) and high-performance interconnect technologies such as InfiniBand and RoCE [86], with precise bandwidth provisioning to prevent collective communication from becoming the bottleneck. The capital cost of network switches, optical modules, and cabling for such deployments can represent a significant share of overall system expenditure.

In combination, these infrastructure requirements make AI datacenter builds substantially more complex and capital-intensive than traditional deployments.

3 Datacenter Lifecycle

Based on these AI workloads, hardware, and infrastructure trends, we examine the datacenter lifecycle to identify opportunities for reducing TCO. Table 1 breaks the lifecycle into: *build*, *IT provisioning*, and *operate*. These stages help us explore how traditional lifecycle policies must be revisited to address the scale, density, and performance demands of

Stage	Description	Timeline
Build	Site selection and facility construction	15–30 years
IT provision	IT hardware deployment and upgrades	4–6 years
Operate	Workload scheduling, resource management	Per inference

Table 1. Lifecycle stages for datacenter infrastructure.

AI. On this foundation, we develop a TCO model to evaluate costs across design choices over the datacenter's lifetime.

3.1 Stages of a Datacenter Lifecycle

Build. This stage is where the datacenter facility itself is designed and constructed. Decisions made here set long-lived constraints on utility capacity, substation feeds, power distribution topology (*e.g.*, flat *vs.* hierarchical), cooling technology (air *vs.* liquid), floor space, and network fabric. These choices determine the maximum achievable rack density, define fault domains, and affect how easily the facility can accommodate future upgrades such as liquid cooling or higher-voltage power buses. Networking decisions (*e.g.*, Ethernet *vs.* InfiniBand, optical link reach, and oversubscription ratios) impact job scaling efficiency and the cost of east–west traffic, which is critical for large AI models distributed across accelerators.

IT provisioning. This stage defines when and how to introduce new accelerators and retire or repurpose older ones. These decisions balance several factors, including performance-per-watt improvements from newer hardware, cost of newer hardware, software maturity and compatibility, depreciation schedules, and risk of underutilized power or cooling. IT provisioning may involve mixed-generation GPU pools with placement constraints or repurposing older GPUs to give them a "second life" on workloads with lower performance requirements (*e.g.*, fine-tuning or batch analytics).

Operate. In this stage, decisions focus on where to place model instances, how to schedule queries to instances, and how to execute queries efficiently. Placement strategies account for hardware heterogeneity, ensuring that each model runs on the accelerator generation that provides the best performance-to-cost tradeoff. Scheduling considers factors such as service-level objectives (SLOs), model routing based on query complexity, and price-aware policies that shift flexible workloads across time or geography. Execution leverages AI-specific optimizations, including dynamic batching, quantization, speculative decoding, distillation, and model disaggregation, to reduce cost per query while meeting SLOs.

Traditional Approach. Table 2 outlines the lifecycle of general-purpose datacenters, which are primarily powered by CPUs and designed to support diverse workloads.

Build. Traditional datacenters rely on a conservative, uniform infrastructure. Power distribution usually follows a hierarchical topology: from the colo-level to rows, and then to individual racks, with each level having its own power caps [115]. Cooling is primarily air-based, and networking is implemented with standard Ethernet [8, 34].

Stage	Traditional Approach Characteristics
Build	Hierarchical power distrib; Air cooling; Ethernet network.
IT provision	Fixed per-server lifecycle; New server generations released every 2–3 years; Gradual replacement.
Operate	Services tied to fixed hardware configurations; Instances migrated to new hardware when released; Legacy applications remain on old servers.

Table 2. Overview of the traditional approach to manage the lifecycle of a traditional datacenter for general-purpose serving CPU-based workloads.

IT provisioning. Servers follow a fixed lifecycle, with each generation operating for a defined period before decommissioning. New hardware is typically released every 2–3 years, and operators replace older servers in a phased manner according to this schedule.

Operate. Services run on servers with a specific hardware configuration. When new hardware is introduced, new services migrate to the latest generation, while legacy applications remain on older servers. This approach prioritizes stability and predictability but reduces flexibility to exploit hardware heterogeneity or tailor performance to specific workloads.

Rearchitecting for AI. As shown in Section 2, AI workloads challenge traditional datacenter design. Modern accelerators have higher power and thermal demands, increasing the value of liquid cooling and high-density racks. Space and density constraints needs favor scale-up architectures like NVLink Switch-based designs [85].

Memory capacity and bandwidth have grown to support larger model contexts, enabling more complex workloads but raising per-node costs. Efficient memory provisioning is essential. Similarly, *interconnect* performance across servers is critical for parallel efficiency (low-bandwidth or high-latency links can severely limit scaling).

Trade-offs between cost and performance must be evaluated both within each stage (build, IT provisioning, and operate) and across them. For example, separating prefill and decode across servers [89] enables using heterogeneous hardware and influences *build* and *refresh* strategies.

3.2 Total Cost of Ownership (TCO) Model

We use a *comprehensive TCO model* that enables automated, end-to-end lifecycle analysis by capturing workload dynamics, hardware evolution, and system aging. It accounts for costs across lifecycle stages and supports exploration of how workload trends, hardware roadmaps, and infrastructure decisions impact total cost.

Model overview. Table 3 summarizes the components of the TCO, breaking them down into *CapEx* and *OpEx*:

$$\text{TCO} = \text{CapEx} + \text{OpEx}$$

Category	Component	Description	Example Cost ($)
CapEx	IT	Servers, racks, accelerators, storage	$375k/server [18]
	Networking	Fabric switches, optics, structured cabling	$2000/server [59, 85]
	Building	Site preparation, building shell, land, electrical and mechanical base infrastructure	$0.5/ft^2 [29]
	Power	Power infrastructure (Switchgear, transformers, UPS, PDUs, busbars, rack distribution)	$7.0/W [26, 116]
	Cooling	Cooling infrastructure (chillers, CRAH/CRAC units, pumps, piping, liquid loops, airflow)	$2.5/W [103, 116]
OpEx	Networking	Port licenses, optics replacement, networking component power	$600/server [59, 82, 84]
	Energy	IT load scaled by PUE, utility tariffs, demand charges	$20–40/MWh [35]
	Maintenance	Spares, repairs, monitoring, water/treatment, field-replaceable units, failure-rate	$5000/server [102]
	Software	Licenses, support contracts	$200/server

Table 3. TCO components for an example datacenter with DGX H100 [18] servers.

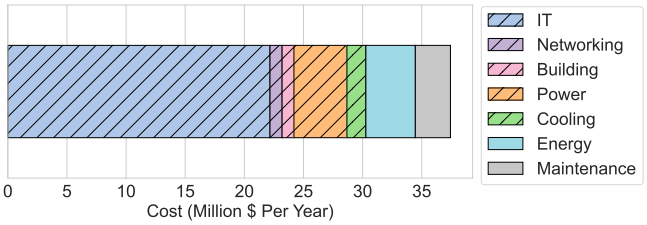


Figure 3. TCO breakdown for a 10MW AI datacenter.

$$\text{CapEx} = \text{CapEx}_{\text{Facility}} + \text{CapEx}_{\text{Power}} + \text{CapEx}_{\text{Cool}} + \text{CapEx}_{\text{Net}} + \text{CapEx}_{\text{IT}}$$

$$\text{OpEx} = \text{OpEx}_{\text{energy}} + \text{OpEx}_{\text{M\&R}} + \text{OpEx}_{\text{network}} + \text{OpEx}_{\text{software}}$$

For the annualized TCO, CapEx amortizes long-term infrastructure and IT over their useful lives, OpEx captures variable and recurring operational costs over a full year.

Figure 3 shows the annual datacenter TCO breakdown, divided into CapEx and OpEx categories for a representative user demand and model size projected for 2025. At 75% average utilization, this 10MW datacenter [115] sustains roughly 500 H100 servers, consuming 70 GWh of energy per year for operation. GPU servers drive IT CapEx and dominate costs, followed by energy-related OpEx. On the other hand, building construction and maintenance contribute the least.

Capital Expenses (CapEx). This is the upfront costs for acquiring, building, or upgrading long-term datacenter assets, including facility, IT equipment (such as servers and racks), and networking infrastructure. Facility, network, and IT assets are amortized over 15–30, 7–10, and 3–5 years respectively, using straight-line or declining-balance depreciation. CapEx is often normalized per delivered kW or per accelerator to facilitate design comparisons.

IT Infrastructure. It consists of compute servers equipped with CPUs and accelerators such as GPUs, TPUs, or NPUs, along with racks and storage devices like NVMe. Modern servers may also include CXL-attached memory to expand capacity. These components provide the core computational and storage resources of the datacenter.

Networking Infrastructure. It connects servers, storage, and other datacenter resources. Fabric switches route traffic between racks, while optical transceivers and structured cabling provide high-bandwidth, low-latency connections across the facility. This infrastructure ensures efficient communication for distributed workloads.

Building. It includes the physical infrastructure required to support a datacenter. These foundational elements ensure a safe and efficient environment for all equipment.

Power Infrastructure. Electrical systems deliver reliable power to racks and IT equipment, including all power devices and connections to ensure stable and redundant power distribution throughout the datacenter.

Cooling Infrastructure. Mechanical systems cool datacenters to maintain safe operating temperatures, with core infrastructure such as chillers and pumps. Together, these systems prevent overheating and enable high-performance operation.

Operational Expenses (OpEx). This is the ongoing costs of running and maintaining a datacenter. We model utilization-sensitive costs (*e.g.*, energy) based on workload mix and scheduling policies, reflecting how activity levels impact overall consumption. In contrast, utilization-insensitive costs (*e.g.*, fixed maintenance, software contracts, and leases) are treated as per-rack or per-site constants, providing a stable baseline that does not fluctuate with workload intensity.

Network Operations. Covers the ongoing costs of maintaining the datacenter network, including licensing fees for switch ports and the replacement of failed components. Costs are influenced by network size, redundancy, and utilization patterns, as higher traffic and denser topologies can increase wear and require more frequent upgrades or maintenance.

Peak power. Datacenters often pay monthly fees based on their highest instantaneous power draw. Utilities charge this way because the grid must be provisioned for these peaks: generators, transformers, and transmission lines all need capacity for the maximum load, even if brief. Peaks can also threaten grid stability and require fast-response resources like spinning reserves. By tying costs to peak usage, utilities incentivize datacenters to smooth load spikes, easing stress on the grid and lowering overall infrastructure costs.

Energy. They include the electricity consumed by IT equipment as well as the supporting infrastructure (*e.g.*, cooling and power distribution systems). These costs, typically billed on a monthly basis, account for datacenter's IT hardware

utilization and *power usage effectiveness* (PUE), which measures the ratio of total facility energy to energy used by IT equipment. A higher PUE indicates more energy spent on overheads such as cooling and power conversion.

Maintenance & Repairs. Includes both preventive and corrective maintenance of mechanical, electrical, and IT systems. Costs are primarily driven by the expected failure rates of components such as cooling and power components, servers, storage devices, and networking equipment. This category also covers other maintenance ensuring that the datacenter remains operational and reliable over time.

Other. Recurring expenses such as software licenses, support contracts, and land or lease payments. These costs are largely fixed and do not vary with workload or utilization.

4 TCO-Driven Lifecycle Framework

To evaluate the long-term economics of AI datacenters, we develop *a cross-stage, TCO-driven framework* that spans the 15-year datacenter lifecycle, covering *build*, *IT provisioning*, and *operation*. Our framework couples workload dynamics, model evolution, hardware road maps, and infrastructure costs to project scenarios and assess alternative policies, allowing us to identify the best policies both within each stage and across stages. We validate our methodology against real-world observations.

4.1 Modeling Assumptions

Timeline. We model a lifecycle of 15 years starting from 2015 and ending in 2030. The methodology generalizes to longer or shorter horizons, but uncertainty increases the further we project. For validation, we fit model outputs against current-day trends (*i.e.*, 2025), then forecast costs and fleet composition for the following five years.

Workload. We focus on LLM inference as the dominant AI datacenter workload [11]. Training workloads would follow a similar methodology but differ in modeling, as they are heavier in both computation and communication. We use input traces from DynamoLLM [100], which exhibit diurnal patterns, and assume a baseline of 100K requests per second. Following Section 2.1, we apply a 15% annual growth rate [56, 91, 108], which implies over 200K RPS after five years.

AI Models. Based on 2015–2025 parameter scaling trends (Section 2.1), we assume linear growth in model size through 2030, with alternative scenarios for accelerated (exponential) or slowed (sub-linear) scaling. Providers are assumed to adopt new models via *smooth migration*, gradually transitioning workloads rather than abrupt cutovers [45, 80]. We also assume future LLMs follow the LLaMA design lineage [67], *i.e.*, decoder-only transformer architectures with consistent layer organization, attention mechanisms, and parameterization strategies. This assumption reflects the convergence of recent frontier models, which primarily differ in scale.

Hardware. Hardware projections include FLOPS, memory bandwidth, TDP, and cost, with linear growth trends [44]. We also model delays between the announcement of a new GPU [78] and its actual mass availability in cloud providers [6, 15, 94] (*e.g.*, B200 had a delay between 6 months and a year).

Performance. We develop a roofline model tailored to LLM inference across diverse hardware configurations. Our validation against known model==hardware pairings shows that it aligns with prior work [40, 114] and with profiling results from real workloads. The model captures how hardware limits (compute throughput and memory bandwidth) interact with workload characteristics (arithmetic intensity and memory footprint) during LLM inference).

Our model derives the arithmetic intensity and memory footprint analytically from the LLM architecture and parameter count, making the predictions reproducible, validated against profiling-based approaches. Extending the model to new GPUs requires only the peak FLOPs and memory bandwidth (GB/s) of the device, while applying it to new LLMs requires recomputing theoretical compute and memory requirements from the model's structure. The roofline model predicts the time-to-first-token (TTFT) and time-between-tokens (TBT) latencies for a given hardware, model, and request load (*e.g.*, H200 running Llama3 at 10 requests per second will have a TTFT of 200 ms and a TBT of 50 ms).

We then increase the load until requests exceed an SLO of 400 ms for TTFT and 100 ms for TBT [100]. The resulting goodput is the maximum request rate (RPS) the system can sustain without violating the SLO. This approach identifies, for each hardware–model pair, the utilization point at which latency begins to degrade. Using this SLO, we provision the minimal number of GPUs required to serve the load and calculate the corresponding GPU utilization.

4.2 Optimization Goal

Cost. The framework integrates both CapEx (IT hardware, networking, building, power, cooling) and OpEx (networking, energy, maintenance, software). CapEx is amortized over useful lifetimes (*e.g.*, 15–30 years for facilities, 3–5 years for GPUs), while OpEx captures recurring operational costs. This enables evaluation of trade-offs such as upfront investment in cooling vs. deferred savings in refresh.

Carbon. A similar methodology applies to carbon emissions, including embodied carbon from construction and hardware, and operational carbon from electricity use. We leave this extension to future work.

4.3 Lifecycle Evaluation

Baseline Timeline. Figure 4 shows the simulated deployment timeline of an AI fleet under the baseline policy, which follows the traditional datacenter lifecycle approach (Table 2). The figure shows how model release cycles and hardware availability shape the fleet composition over time, with the

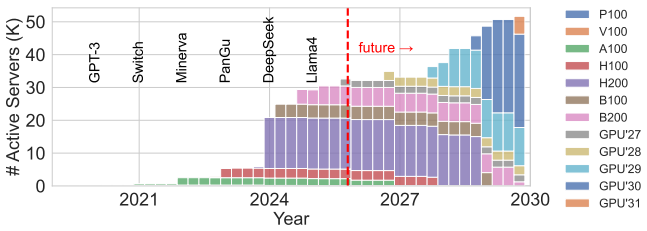


Figure 4. Server count by GPU type over time in an AI fleet following the traditional baseline in Table 2. Includes the release dates of major AI models.

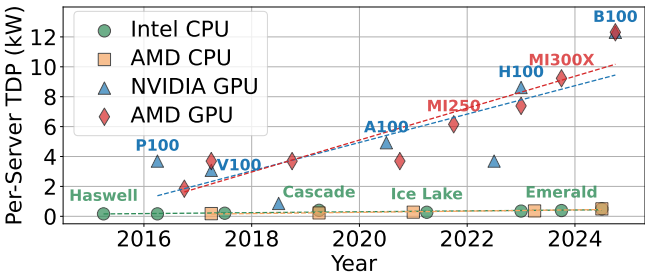


Figure 5. Per-server TDP across Intel and AMD CPUs, and NVIDIA and AMD GPUs over the years.

release dates of notable large models marked for reference. The simulation begins in 2015 with 50 P100 GPU servers sustaining an initial steady-state load of 100K RPS. At this point, the total annual TCO of the AI datacenter is ≈$0.2M.

As user demand and model size grow (*i.e.*, 15% year-over-year), the fleet scales gradually. By 2024, traffic reaches 350K RPS, coinciding with the release of DeepSeek V3, a 671B-parameter model [21, 64], which triggers a major hardware refresh with H200 GPUs. This refresh increases the total server count to 25K to meet performance targets, and the annual TCO rises to ≈$0.3B, reflecting the added hardware, expanded datacenter infrastructure, and higher operating expenses. Peaks in server deployments align with major LLM launches, highlighting the close connection between AI model roadmaps and datacenter economics.

Solution Approach. We run Monte Carlo simulations [70] to capture uncertainty in workload growth, model scaling, hardware availability, and cost projections. Each simulation samples these distributions, yielding a range of outcomes for TCO under different policies (*e.g.*, aggressive vs. delayed server refresh). We use the traditional approach in Table 2 as a baseline. By comparing distributions, we identify which decisions dominate long-term cost and where flexibility provides the greatest value.

5 Building Efficient AI Infrastructure

The first stage in the datacenter lifecycle is building the infrastructure, which includes: (1) the physical building that houses the datacenter, (2) the power delivery that supplies electricity to servers and other equipment, (3) the cooling that removes the heat generated by servers, and (4) the networking that interconnects servers.

The physical building is relatively static, while power, cooling, and networking requirements [26] must anticipate shifts in workload demand and hardware capabilities over decades of operation. Emerging AI workloads, with their extreme power and thermal densities and distinct communication patterns, are already reshaping the design space across these dimensions. We revisit these design choices using our TCO-driven framework to build a holistic understanding over the full lifecycle and validate our approach comparing it to the directions already being pursued in practice.

	Feature	Power domain: Per-PDU	Per-UPS	Per-DC
CapEx	Stranding	Lower	Medium	Higher
	Complexity	Higher	Medium	Lower
OpEx	Maintenance	Higher	Medium	Lower
Other	Fault isolation	Excellent	Good	Poor

Table 4. Comparison of power delivery infrastructure designs. Green: good, yellow: moderate, red: poor.

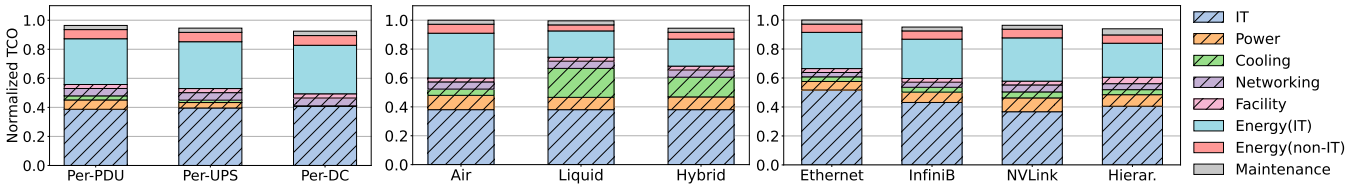
5.1 Power Infrastructure

Traditional Approach. Traditional datacenters adopt a hierarchical power distribution, balancing fault isolation, maintainability, and power stranding [107, 109, 115]. An Automatic Transmission Switch (ATS) directs power from the grid to multiple Uninterruptible Power Supplies (UPS) with redundancy. Each UPS supports a fraction of the total load and feeds downstream into multiples Power Distribution Units (PDUs), distributing power to server rows and racks.

Server and rack provisioning must respect capacity limits within each power sharing domain defined by the hierarchy. When the budget at a power domain (*e.g.*, per-PDU) is X and each server consumes Y, only $\lfloor X/Y \rfloor$ servers can be deployed. The residual capacity, $X - Y \times \lfloor X/Y \rfloor$, becomes stranded power (*i.e.*, power fragmentation).

Rearchitecting for AI. AI accelerators (*e.g.*, GPUs and TPUs) push power density far beyond historical norms. Figure 5 shows that the TDP of high-end GPU servers has significantly outgrown the power demand of high-end CPU servers. For example, while NVIDA DGX H100 server with 8 GPUs needs 10.2kW [76], Intel Emerald Rapids server with 64 cores needs only 385W [50].

While this surge exacerbates power fragmentation, flatter power distribution architectures can pool power across broader domains to reduce stranding [7, 92]. However, these designs involve complicated trade-offs in fault tolerance, design and maintenance complexity, and TCO, we introduce simplifications by removing second-order effects and practical constraints (*e.g.* supply chains) and summarize these trade-offs qualitatively in Table 4. For example, while a flat



(a) TCO *vs.* power infrastructure. **(b)** TCO *vs.* cooling infrastructure. **(c)** TCO *vs.* networking infrastructure.

Figure 6. TCO *vs.* infrastructure designs during the *build* stage.

	Feature	Air	Hybrid	Liquid
CapEx	Complexity	Lower	Medium	Higher
OpEx	Energy efficiency	Lower	Medium	Higher
	Maintenance	Lower	Medium	Higher
Other	High-dense racks	Lower	Medium	Higher
	Noise level	Higher	Medium	Lower

Table 5. Comparison of cooling infrastructure designs.

	Feature	Ethernet	InfiniBand	NVLink	Hierarchical
CapEx	Cost	Lower	Medium	Higher	Medium
OpEx	Energy	Lower	Medium	Higher	Higher
	Maintain	Lower	Medium	Higher	Medium
Perf	Bandwidth	Lower	Higher	Higher	Higher
	Latency	Higher	Lower	Lower	Medium

Table 6. Comparison of networking infrastructure designs.

per-DC power delivery minimizes stranded capacity, it demands greater redundancy and more intricate maintenance planning. Though a flat per-DC power delivery architecture might not be feasible in practice today, our TCO-driven analysis in Figure 6a shows that, with simplified modeling of associated costs and hardware requirements, it reduces lifecycle TCO by 4.2% compared to conventional hierarchical designs [115], motivating new designs enabling a flatter power sharing domains in real-world distribution schemes.

5.2 Cooling Infrastructure

Traditional Approach. Most datacenters rely on air-based cooling systems [20, 23, 48, 66, 99], where chillers or adiabatic cooling units deliver cold air through air handling units (AHUs). These AHUs circulate the air via raised floors or hot/-cold aisle containment using fans. Cooling capacity is provisioned conservatively to meet peak thermal loads. Power usage effectiveness (PUE) is optimized through careful airflow management and economization. Modern datacenters achieve typical PUE values between 1.1 and 1.3 [3, 31, 72].

Rearchitecting for AI. High-density GPU racks generate far more heat than CPU-based system (often 4–8× higher per rack [79]). Meeting these loads requires significantly higher airflow rates, lower inlet temperatures, and more fan energy, pushing air cooling beyond its physical and economic limits.

As a result, liquid cooling (*e.g.*, cold plates or immersion [32, 53]) is increasingly considered for future dense GPU deployments [7, 79]. While upfront complexity and CapEx are higher, OpEx is reduced through improved heat transfer, reduced chiller load, and lower fan power, summarized in Table 5. Hybrid designs, combining liquid for AI racks and air for low-density racks, balance cost, density, and maintainability. Specifically, a hybrid design could be a mix of 75% cold plates and 25% air cooling [43], corresponding to the high and low density loads with a resulting Power Usage Effectiveness (PUE) of 1.15. Our evaluation in Figure 6b shows that hybrid cooling yields the lowest TCO with 9% reduction over the full lifecycle.

5.3 Networking Infrastructure

Traditional Approach. General-purpose datacenters use multi-tier Ethernet fabrics (*e.g.*, leaf–spine) with moderate oversubscription at higher tiers [8, 34]. This design is cost-effective for CPU-based workloads with modest inter-node bandwidth and latency needs.

Rearchitecting for AI. AI workloads impose far greater network demands than general-purpose datacenters. Emerging practice starts adopting hierarchical designs for AI inference workloads to match communication patterns [60]: NVLink [85] for intra-server to support tensor parallelism (TP) [96], and lower-cost networks for pipeline parallelism [96], which exchanges data less frequently and is employed across servers. We evaluate four network designs: (1) all-accelerator connectivity over Ethernet, (2) all over InfiniBand [5], (3) all over NVLink [85], and (4) a *hierarchical* approach (NVLink within a server, InfiniBand within a rack, and Ethernet across racks). Table 6 shows different cost–performance trade-offs across networking strategies. Figure 6c shows that the hierarchical design delivers the best balance, reducing TCO by 6% relative to a flat high-performance network to achieve the workload latency requirements. By aligning interconnect performance with workload communication patterns, hierarchical networking ensures scalability without over-provisioning expensive, low-latency links.

5.4 Lessons

AI workloads fundamentally challenge legacy datacenter designs, creating unprecedented demands on power, cooling, and networking. As accelerator power densities increase and communication patterns grow more complex, traditional hierarchical power distribution, air-based cooling, and uniform network fabrics become increasingly inadequate; both

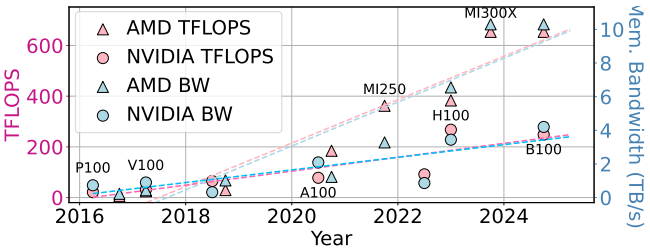


Figure 7. Evolution of AMD and NVIDIA GPUs showing TFLOPS (left axis) and memory bandwidth (right) over time.

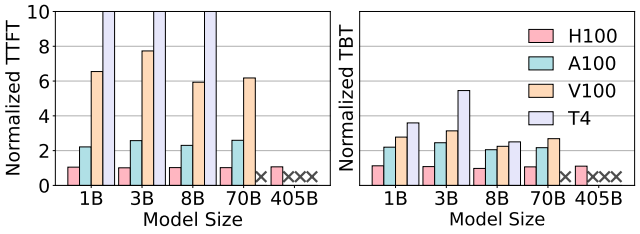


Figure 8. TTFT and TBT latencies for different sizes of Llama-3 LLM across GPU generations normalized to H200.

in terms of cost efficiency and workload performance. These rapidly evolving requirements have spurred a wave of infrastructure innovations. By evaluating design choices from the ground up using our TCO-driven framework over the full datacenter lifecycle, we show how emerging solutions (*e.g.*, flatter power delivery architectures, hybrid cooling systems, and hierarchical network designs) can meet long-term efficiency and cost objectives.

6 Provisioning AI Hardware

Once the datacenter infrastructure is built, the next step is managing the hardware lifecycle: deciding when and how to retire outdated components and introduce new ones. Careful refresh planning ensures the system continues to meet performance, efficiency, and scalability demands.

6.1 Current Hardware AI Trends

GPU Release Cycle. GPUs from vendors like NVIDIA and AMD have evolved significantly. Starting with the NVIDIA P100 (Pascal), which offered modest FP16/FP32 throughput, modern GPUs such as the B200 deliver vastly higher performance through advanced tensor cores, increased memory bandwidth, and optimizations in sparsity and mixed precision. Figure 7 shows the progression of compute capability (TFLOPS) and memory bandwidth of NVIDIA datacenter GPUs from 2016 to 2025. Compute throughput has increased nearly 12×, while memory bandwidth (critical for large-scale and sparse model inference) has grown over 5×. These trends are even more pronounced for AMD GPUs [4].

GPU vendors are aggressively releasing new architectures every year (even multiple generations per year) to meet the rapidly evolving demands of AI. This pace contrasts the traditional 2–3 year cadence of CPU releases. Similar trends hold for other specialized AI hardware such as TPUs [55] and LPUs [37], which also follow fast-paced release cycles.

Cost. The cost of GPU servers has also grown substantially. For example, the NVIDIA P100 was priced at around $9K per GPU (roughly $90K per server), whereas the modern H100 costs about $30K per GPU (over $350K per server) [18]. AMD's MI series accelerators follow a similar trend, with newer generations carrying significantly higher price tags. In contrast, CPU server costs have risen more moderately over the same period. For example, from around $7K per server for Intel's Haswell [51] systems in 2014 to approximately $12K for the latest Granite Rapids [49] generation in 2025.

Performance. We evaluate how the performance of AI inference workloads with different model sizes and architectures scales across GPU generations. We ran experiments using real hardware on vLLM [57] and compare against the roofline model, which shows within 5% of errors for Llama3 [68] models. All runs use a 2K sequence length and batch size of 8. We measure TTFT and TBT in milliseconds, normalized to the H200 baseline per model. We also combine performance with cost and power (*i.e.*, Perf/$ and Perf/Watt).

Model Size. We evaluate scalability using Llama3 [68] models from 1B to 405B parameters on NVIDIA GPUs: T4, V100, A100, H100, and H200. We configure TP [96] with the smallest value that fits each model across most GPUs (*e.g.*, TP1 for 1B/3B, TP4 for 8B, TP8 for 70B/405B). Some setups fail on older GPUs due to memory limits. Figure 8 shows that older GPUs remain viable for small models. The decode phase (TBT) is less sensitive to GPU generation than the prefill phase, reflecting its lower compute demands [89].

Model Architecture. We evaluate the impact of model sparsity by comparing sparse Qwen3 models (30B A3B and 235B A22B) with dense Llama3 models of similar sizes. Figure 9 shows that sparse models scale better on older GPUs, maintaining competitive accuracy while outperforming dense counterparts in latency. For example, Qwen3-235B-A22B matches Llama3-70B in accuracy but degrades less on older hardware (though it requires nearly twice the GPU memory). This highlights the value of sparsity-aware designs for extending the utility of legacy hardware.

Figure 10 compares transformer-based (Llama3-3B) with state-space-based (Mamba-2.8B). State-space models are more hardware-efficient: for 2K sequences using TP1, Llama3 runs 7.7× slower on V100 than on H200, while Mamba slows down by only 3.6×. This shows the architectural compatibility of state-space models with older or less performant GPUs.

Model Efficiency. We combine model performance and hardware cost and power usage to evaluate model efficiency on different generations of hardware. While the goodput per

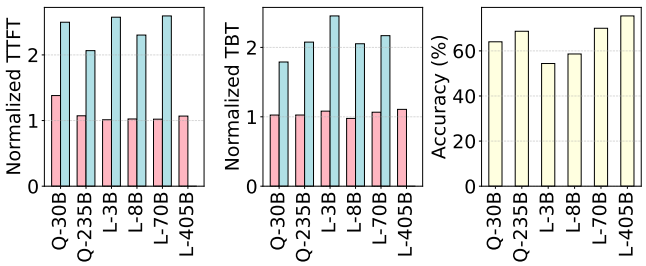


Figure 9. Latencies and accuracies for dense (Llama3) and sparse models (Qwen3) across GPU generations normalized to H200. Pink and blue bars for H100 and A100, respectively.

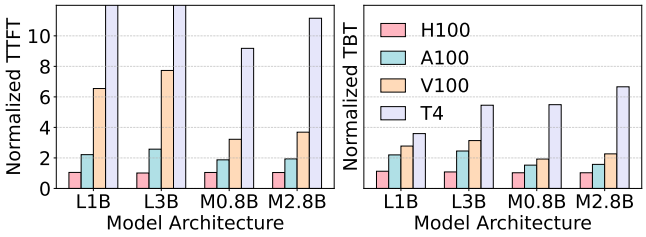


Figure 10. Latencies for transformer (Llama3) and state space (Mamba) models across GPUs normalized to H200.

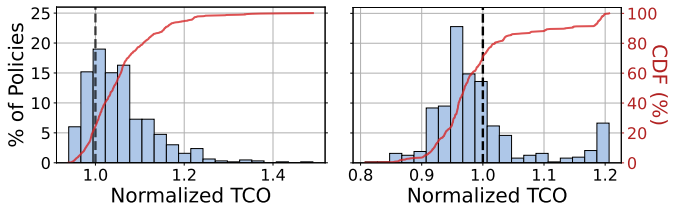
Watt of large models degrades substantially on older hardware, the gap is much smaller for smaller models. For instance, running Llama3-70B on an A100 yields roughly 3× lower goodput per Watt compared to an H100, whereas running Llama-1B on an A100 is only about 18% lower. In terms of goodput per Watt per dollar, the A100 actually outperforms the H100 by 8–23% for the 1B, 3B, and 8B models, but delivers about 2× lower efficiency for Llama-70B. In fact, for smaller models, even the V100 is on par with the H100, achieving performance per Watt per dollar within 5% of it.

6.2 Refresh Policies

Traditional Approach. General-purpose datacenters typically follow a steady CPU refresh cycle, with servers remaining in service for about five years before replacement [106]. This baseline policy strikes a balance between capital and operational efficiency.

Alternative strategies include extending server lifetimes to reduce CapEx (at the expense of higher energy use and maintenance) or shortening lifetimes to deploy more efficient hardware sooner, increasing capital costs but improving energy and space efficiency. Skipping intermediate generations is another option when current hardware meets workload needs and newer gains are marginal.

We use our framework to evaluate the policies to provision and refresh the servers in an AI datacenter. Figure 11a shows the TCO distribution of all feasible refresh policies normalized to baseline policy value. Most of the policies lie on the right side of the distribution, indicating the 5-year



(a) General-purpose datacenters. **(b)** AI datacenters.

Figure 11. TCO distribution for various hardware refresh policies in general-purpose and AI datacenters.

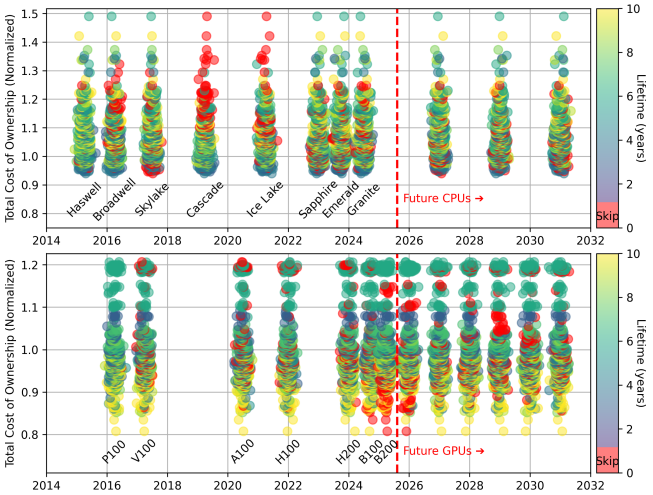


Figure 12. TCO changing refresh policy for each hardware generation in general-purpose and AI datacenters.

baseline remains among the most cost-effective strategies for general-purpose datacenters.

Figure 12 shows the same data split by hardware generation. The top of Figure 12 presents a per-CPU-generation view, showing how varying the refresh policies of each individual hardware generation, from 0 years (skipping that generation entirely) to 10 years, impacts normalized TCO. For general-purpose datacenters, except for Skylake (where skipping was preferable), the refresh policies of 4–6 years lifetime have the lowest TCOs.

Rearchitecting for AI. The dynamics of AI accelerators and workloads differ substantially from those of general-purpose datacenters, rendering the traditional refreshes sub-optimal. Figure 11b shows that many alternative refresh strategies can reduce TCO by 15–20% compared to the baseline.

Hardware Trend. Figure 12 shows the TCO changing the refresh cycle (and skipping) for each hardware generation. Three dominant hardware-driven trends emerge. First, *newer is much better*: when new GPUs provide substantial efficiency gains, early decommissioning of older hardware is justified, and datacenters benefit from upgrading as soon as the next generation is available. For example, transitioning from NVIDIA V100 to A100 GPUs is beneficial.

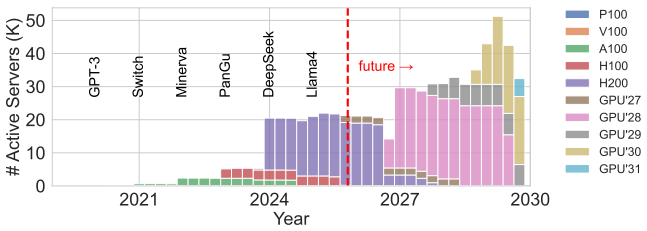


Figure 13. Server count by GPU type over time in an AI fleet following the optimal refresh policy for minimizing TCO.

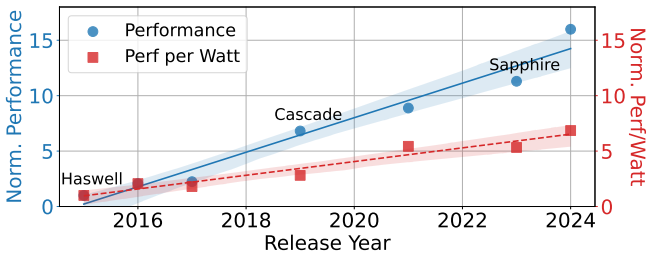


Figure 14. Performance and performance/Watt for DCPerf applications [101] across generations of Intel servers.

Second, *older is competitive*: The generation breakdown shows that for most cases extending their hardware lifespan beyond 6 years is cost-effective. This was the case for all generations before B100 GPUs.

Third, *newer is similar or worse*: when new GPUs offer marginal or negative efficiency gains, it is better to extend the life of existing hardware and skip intermediate generations. For example, it is more cost effective to skip B100/B200 GPUs.

Workload Evolution. If models grow significantly year over year (as seen in past GPT-family releases [87]) refresh cycles must be accelerated to provision more efficient hardware that can handle the increased compute needs. Conversely, if model sizes stabilize or decrease, extending hardware lifetime becomes more cost-effective.

Example Timeline. Figure 13 shows the deployment of GPU server generations under the optimal refresh policy to minimize TCO. The policy skips some generations (*e.g.*, B100, B200) entirely, as extending the service life of H100 and H200 GPUs proves more cost-effective. When a newer generation delivers substantial performance and efficiency gains, the policy triggers earlier decommissioning and demand-driven purchases to match workload growth and model sizes. Compared to the baseline policy in Figure 4, this approach yields a smoother, more balanced mix of old and new hardware, and at times even a modest reduction in total server count due to improved GPU efficiency (*e.g.*, in early 2027).

6.3 Lessons

Fixed refresh intervals are no longer sufficient for AI workloads. Unlike traditional datacenters, where hardware efficiency and workload demands change gradually, AI accelerators and models evolve at a much faster pace. Some generations deliver dramatic performance gains, while others bring only marginal improvements. These shifts are further amplified by rapid increases in power and thermal densities, hardware costs, and release frequency, changing the trade-offs between raw performance and efficiency.

Therefore, AI datacenters must adopt flexible hardware refresh strategies that respond to evolving hardware efficiency and workload trends. This may involve aggressively retiring older GPUs when new generations deliver significant efficiency gains, while extending the life of existing hardware (or skipping intermediate generations) when improvements are limited. By aligning refresh decisions with model scaling and operational cost trends, infrastructure can evolve more intelligently, maximizing performance while controlling costs.

7 Operating an AI Datacenter

Once hardware is provisioned, the challenge becomes sustaining high utilization while meeting SLOs across a diverse, evolving fleet. This demands software techniques to orchestrate diverse workload types, hardware generations, and different performance-cost trade-offs, all shaped by earlier building and refreshing decisions.

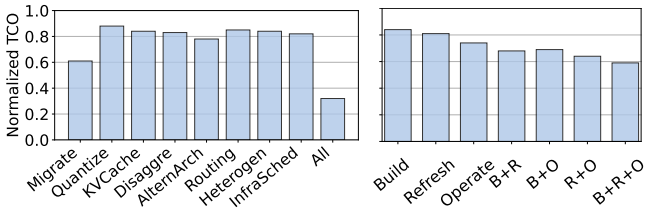
Traditional Approach. In general-purpose datacenters, workloads typically run on homogeneous hardware pools with uniform deployment strategies, from bare-metal servers and VMs to microservices and serverless. Most workloads are generally deployed on the latest hardware, while some workloads may remain on legacy systems until decommissioned [36]. Software stacks are often tuned for predictable, stable performance, requiring little adaptation to hardware diversity or rapid workload changes.

To quantify this, we use the DCPerf benchmark suite [101], which includes representative production-grade applications (*e.g.*, key-value stores, databases, and video processing) to evaluate performance and power efficiency across multiple Intel server generations. Figure 14 shows aggregated performance and performance per Watt, indicating that both throughput and power efficiency scale nearly linearly with newer servers. This justifies the common practice of migrating workloads to the latest generation.

Rearchitecting for AI. As discussed in Section 6.1, AI workloads and hardware trends diverge significantly from those of general-purpose datacenters. Unlike CPUs, we do not observe uniform or linear performance improvements of AI inference workloads across GPU generations, and the benefits of new hardware vary considerably across different models and use-cases. Hence, applying the same direct-migration strategy used in traditional datacenters can be suboptimal for AI workloads. Instead, achieving operational efficiency for AI requires software optimizations that bridge the gap

Optimization Technique	Description	TCO Impact
Smooth Model Migration [45, 80]	Gradual migration from old to newer models upon releases	Avoid rapid hardware procurement
Model Quantization [63, 118]	Lower precision to reduce compute/memory	Lower hardware demand and cost per inference
KV-Cache Management [27, 83]	Optimize storage and reuse of KV cache	Increase older hardware reuse
Disaggregation [83, 89, 105, 120, 122]	Split distinct phases onto different hardware	Extend useful life of heterogeneous generation
Alternative Architectures	Mixture-of-Experts [95], State-Space-Models [38]	Increase older hardware reuse
Model Routing [22, 52]	Direct workloads to the most efficient model variant	Increase older hardware reuse
Heterogeneity-Aware Scheduling [54, 65]	Map workloads to optimal/available hardware generation	Defer refresh costs
Infrastructure-Aware Scheduling [99, 100]	Exploit headroom within infra capacity envelopes	Improve infrastructure efficiency

Table 7. Operation stage software optimizations that introduce new cross-stage optimization opportunities.



(a) TCO *vs.* software optimization. **(b)** TCO *vs.* stages.

Figure 15. TCO for optimization during *operation* and for stages, normalized to the baseline without optimizations.

between evolving models, growing user demands, and a heterogeneous fleet, while controlling cost and sustaining SLOs.

Optimizations. Table 7 summarizes some key operation optimizations that influence build and refresh strategies. *Model migration*, *quantization*, and *KV-cache management* reduce compute and memory pressure. *Disaggregation* leverages fast interconnects and aligns workload phases with suitable hardware generations. *Heterogeneity-aware* routing, placement, and scheduling adapt workloads in real time to match hardware with workload characteristics. *Infrastructure-aware scheduling* optimizes for datacenter constraints (*e.g.*, power and cooling) via software controls (*e.g.*, dynamic sharding), linking operational policies to build-time decisions and enabling lifecycle-wide optimization.

TCO savings. These optimizations extend the life of older hardware, defer costly refreshes, and improve infrastructure value. Figure 15a shows TCO reductions of 12–39% per strategy compared to the baseline. *Smoothen model migration* delivers the largest gains by reducing post-release compute load. *Disaggregation* and *infrastructure-aware scheduling* achieve strong savings without modifying load, by leveraging heterogeneous fleets and aligning software with datacenter constraints. Combining *all* optimizations yields over 60% TCO reduction. While not fully additive due to overlaps, the cumulative impact is substantial.

7.1 Lessons

Fixed, uniform operational strategies are insufficient for rapidly evolving models and heterogeneous hardware generations. Software-driven techniques (*e.g.*, model migration smoothing, disaggregation, and heterogeneity-aware scheduling) shift operations from a reactive process to a proactive, cost-optimized stage of the datacenter lifecycle. These optimizations extend the useful life of older hardware, leverage heterogeneous fleets, and dynamically align workloads with available infrastructure to capture efficiency gains. By closing the loop between operations, hardware refresh, and build stages, these techniques turn rigid hardware timelines into adaptive, lifecycle-aware policies.

8 Cross-Stage Optimizations

Optimizing individual stages for AI reduces TCO, but the largest gains come from cross-stage rearchitecting the end-to-end AI datacenter lifecycle. We outline the current cross-stage strategies and explore emerging opportunities.

8.1 Existing Cross-Stage Optimizations

IT provisioning → Build. AI hardware trends are reshaping the datacenter build. Power hierarchies are flattening to support high-density accelerators, cooling is shifting from air to liquid [79], and networking is moving beyond Ethernet to InfiniBand and NVLink fabrics [85]. These choices raise initial build costs but ease future refreshes, extending the usable lifetime of each deployment.

Operate → IT provisioning. Heterogeneity-aware scheduling helps repurposing older GPUs for workloads better suited to their capabilities: compute-intensive phases (*e.g.*, prefill or large models) run on newer GPUs, while memory- or bandwidth-bound phases (*e.g.*, decode or smaller models) are offloaded to older generations [65, 89]. This strategy smooths refresh costs and maintains high utilization, turning hardware upgrades into opportunities for redistribution and continued value rather than premature hardware retirement.

Build → Operate. Infrastructure decisions made at build time shape operational flexibility. Scheduling frameworks translate these choices into software controls that smooth demand, enable safe hardware derating, and sustain efficiency. Conversely, coordinated derating of servers and power devices within the power hierarchy allows oversubscription

and denser deployments; effectively "upgrading" infrastructure at runtime without new physical buildouts [88, 99].

Compound TCO Benefits. Figure 15b shows that cross-stage strategies amplify impact across the lifecycle. Optimizing individual stages yields 20–30% savings, while combining build and refresh policies pushes savings beyond 35%. A holistic approach can reduce TCO by over 40%. Assuming hardware and model sizes grow linearly (center cells in Table 8), the best infrastructure to *build* includes flat power delivery, hybrid liquid-air cooling, and a hierarchical networking with Ethernet, Infiniband, and NVLink. For hardware *refresh*, it is best to extend server lifetimes to five years and adopt new hardware as it becomes available. During *operation*, the best strategy is to combine *all* available optimizations.

Hardware Growth \ Model Growth	Slow →	Medium ↗	Fast ↑
→	Per-PDU Air IB	Per-PDU Air NVLink	Per-PDU Air NVLink
↗	Flat Hybrid IB	Flat Hybrid Hierarchy	Flat Liquid Hierarchy
↑	Flat Liquid IB	Flat Hybrid IB	Flat Hybrid Hierarchy
→	8 years Skip	8 years Skip	8 years Skip
↗	4 years Buy new	5 years Buy new	4 years Skip
↑	4 years Buy new	4 years Buy new	3 years Buy new
→	All Optimizations	Migration + Quantization	Migration + Quantization
↗	Disaggregation	All Optimizations	Migration + Quantization
↑	Disagg. + Hetero.	Disaggregation	All Optimizations

Table 8. Optimal cross-stage strategies based on model and hardware trends under exponential user demand growth. The rows represent *build*, *refresh*, and *operate* stages. Color gradient shows degree of lifecycle adaptation required.

8.2 Opportunities for Cross-Stage Optimizations

Looking ahead, several opportunities emerge when infrastructure, hardware, and software are explicitly co-designed with lifecycle interplay in mind.

Infrastructure. Today's software supports heterogeneous fleets, but build and refresh strategies can better leverage heterogeneity. Rack-level provisioning with mixed-generation accelerators or general-purpose compute reduces interconnect bottlenecks and power fragmentation. Combined with heterogeneous derating, these setups adapt efficiently to workload demands. While they require upfront investment in fine-grained telemetry and control, they offer substantial long-term efficiency gains.

Hardware. Emerging AI accelerators have traditionally shaped datacenter infrastructure design. Looking ahead, future accelerators should be designed not only for performance but also for long-term compatibility with the existing infrastructure. Lower power density and moderated TDP simplify power delivery and cooling, reducing fragmentation. For example, accelerators could combine high-power SMs to handle the compute-bound prefill phase with Processing-in-Memory (PIM) units tailored for the memory-bound decode phase, enabling more efficient execution within the same server.

Operation. Techniques such as KV-cache management or new model architectures (*e.g.*, MoEs) shift the balance between compute and memory needs, reshaping both refresh priorities and placement strategies. Cross-stage planning anticipates these shifts by provisioning memory or storage servers that support multiple GPU generations.

8.3 Future Trends

As AI models and hardware keep evolving, lifecycle management must adapt not only to these trends individually but also to their compounded effects. We analyze model scaling and hardware trajectories to offer guidance for future AI datacenter lifecycles. Table 8 shows the optimal strategies across *build*, *IT provisioning*, and *operate* stages, tailored to projected growth patterns in both hardware and models. We explore *slow* (sub-linear), *medium* (linear), and *fast* (exponential) growth of model size/complexity and hardware.

AI Hardware. When hardware performance scales rapidly, frequent *refresh* cycles are advantageous, justifying earlier adoption and earlier decommissioning. However, if performance gains slow or power efficiency declines, it becomes more beneficial to extend refresh intervals and design infrastructure for long-term use. Lower-power, lower-density hardware can also extend infrastructure lifetimes by reducing the need for costly upgrades to power and cooling systems.

AI Model Size. So far, we have assumed that AI models will grow linearly. Larger models amplify compute, memory, and networking requirements, accelerating refresh cadence and demanding infrastructure explicitly designed for modular expansion. If model size growth slows down, operators can rely on more cost-effective infrastructures (*e.g.*, InfiniBand networking). Conversely, if models continue to scale rapidly, more expensive infrastructures become necessary (*e.g.*, NVLink), alongside more aggressive software optimizations (*e.g.*, model migration and quantization).

8.4 Lessons

Cross-stage optimization reframes lifecycle management as a continuum rather than a sequence. Build-time choices, refresh strategies, and operation policies compound to drive long-term efficiency. Looking ahead, future datacenters should be increasingly co-designed across infrastructure, hardware, and software; with lifecycle-aware thinking at the core of scalable, cost-effective AI.

9 Conclusion

The rapid growth of AI workloads has outpaced traditional datacenter management. AI datacenters need flexible designs, advanced cooling, and software for heterogeneous hardware. While stage-specific improvements (build, IT provisioning, operate) help, the biggest gains, up to 40% TCO reduction, come from a holistic, cross-stage approach that anticipates workload dynamics and hardware trends, enabling cloud providers to scale efficiently and control costs.

References

[1] Deepak Bhaskar Acharya, Karthigeyan Kuppan, and B Divya. 2025. Agentic ai: Autonomous intelligence for complex goals–a comprehensive survey. *IEEe Access* (2025).

[2] Keivan Alizadeh, Iman Mirzadeh, Dmitry Belenko, Karen Khatamifard, Minsik Cho, Carlo C Del Mundo, Mohammad Rastegari, and Mehrdad Farajtabar. 2024. LLM in a flash: Efficient Large Language Model Inference with Limited Memory. *arXiv preprint arXiv:2312.11514* (2024).

[3] Amazon AWS. 2025. Power usage effectiveness. https://sustainability.aboutamazon.com/products-services/aws-cloud.

[4] AMD. 2025. AMD Instinct™ GPUs. https://www.amd.com/en/products/accelerators/instinct.html.

[5] InfiniBand Trade Association. 2001. InfiniBand Architecture Specification. In *InfiniBand Trade Association*. https://www.infinibandta.org/specs/ Version 1.0.

[6] Microsoft Azure. 2025. Microsoft and NVIDIA accelerate AI development and performance. https://azure.microsoft.com/en-us/blog/microsoft-and-nvidia-accelerate-ai-development-and-performance/.

[7] Ricardo Bianchini, Christian Belady, and Anand Sivasubramaniam. 2024. Datacenter power and energy management: past, present, and future. *IEEE Micro* (2024).

[8] Kashif Bilal, Samee U Khan, Limin Zhang, Hongxiang Li, Khizar Hayat, Sajjad A Madani, Nasro Min-Allah, Lizhe Wang, Dan Chen, Majid Iqbal, et al. 2013. Quantitative comparisons of the state-of-the-art data center architectures. *Concurrency and Computation: Practice and Experience* 25 (2013).

[9] Jairus Bowne. 2024. Using Large Language Models in Learning and Teaching. https://biomedicalsciences.unimelb.edu.au/study/dlh/assets/documents/large-language-models-in-education/llms-in-education.

[10] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners. arXiv:2005.14165 [cs.CL] https://arxiv.org/abs/2005.14165

[11] Foundation Capital. 2024. *Why 2024 Will Be the Year of Inference.* https://foundationcapital.com/insights/why-2024-will-be-the-year-of-inference Accessed: 2025-09-26.

[12] Gohar Irfan Chaudhry, Esha Choukse, Haoran Qiu, Íñigo Goiri, Rodrigo Fonseca, Adam Belay, and Ricardo Bianchini. 2025. Murakkab: Resource-Efficient Agentic Workflow Orchestration in Cloud Platforms. *arXiv preprint arXiv:2508.18298* (2025).

[13] Qiguang Chen, Libo Qin, Jinhao Liu, Dengyun Peng, Jiannan Guan, Peng Wang, Mengkang Hu, Yuhang Zhou, Te Gao, and Wanxiang Che. 2025. Towards reasoning era: A survey of long chain-of-thought for reasoning large language models. *arXiv preprint arXiv:2503.09567* (2025).

[14] Esha Choukse, Brijesh Warrier, Scot Heath, Luz Belmont, April Zhao, Hassan Ali Khan, Brian Harry, Matthew Kappel, Russell J Hewett, Kushal Datta, et al. 2025. Power stabilization for AI training datacenters. *arXiv preprint arXiv:2508.14318* (2025).

[15] Google Cloud. 2025. Introducing A4X VMs powered by NVIDIA GB200. https://cloud.google.com/blog/products/compute/new-a4x-vms-powered-by-nvidia-gb200-gpus.

[16] Joel Coburn, Chunqiang Tang, Sameer Abu Asal, Neeraj Agrawal, Raviteja Chinta, Harish Dixit, Brian Dodds, Saritha Dwarakapuram, Amin Firoozshahian, Cao Gao, Kaustubh Gondkar, Tyler Graf, Junhan Hu, Jian Huang, Sterling Hughes, Adam Hutchin, Bhasker Jakka, Guoqiang Jerry Chen, Indu Kalyanaraman, Ashwin Kamath, Pankaj Kansal, Erum Kazi, Roman Levenstein, Mahesh Maddury, Alex Mastro, Siji Medaiyese, Pritesh Modi, Jack Montgomery, Satish Nadathur, Amit Nagpal, Ashwin Narasimha, Maxim Naumov, Eleanor Ozer, Jongsoo Park, Poorvaja Ramani, Harikrishna Reddy, David Reiss, Deboleena Roy, Sathish Sekar, Arushi Sharma, Pavan Shetty, Aravind Sukumaran-Rajam, Eran Tal, Mike Tsai, Shreya Varshini, Richard Wareing, Olivia Wu, Xiaolong Xie, Jinghan Yang, Hangchen Yu, Tanmay Zargar, Zitong Zeng, Feixiong Zhang, Ajit Matthews, Xun Jiao, Jiyuan Zhang, Emmanuel Menage, Truls Edvard Stokke, and Mohammed Sourouri. 2025. Meta's Second Generation AI Chip: Model-Chip Co-Design and Productionization Experiences. In *Proceedings of the 52nd Annual International Symposium on Computer Architecture (ISCA '25)*. Association for Computing Machinery, New York, NY, USA, 1689–1702. https://doi.org/10.1145/3695053.3731409

[17] CoreSite. 2025. AI and the Data Center: Driving Greater Power Density. https://www.coresite.com/blog/ai-and-the-data-center-driving-greater-power-density.

[18] Cyfuture Cloud. 2025. What is the Price of an NVIDIA DGX H100 System? https://cyfuture.cloud/kb/gpu/what-is-the-price-of-an-nvidia-dgx-h100-system.

[19] Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. 2022. FlashAttention: Fast and Memory-Efficient Exact Attention with IO-Awareness. *arXiv preprint arXiv:2205.14135* (2022).

[20] Hafiz M Daraghmeh and Chi-Chuan Wang. 2017. A review of current status of free cooling in datacenters. *Applied Thermal Engineering* 114 (2017), 1224–1239.

[21] DeepSeek. 2025. Introducing DeepSeek-V3. https://api-docs.deepseek.com/news/news1226.

[22] Dujian Ding, Ankur Mallick, Chi Wang, Robert Sim, Subhabrata Mukherjee, Victor Ruhle, Laks VS Lakshmanan, and Ahmed Hassan Awadallah. 2024. Hybrid llm: Cost-efficient and quality-aware query routing. *arXiv preprint arXiv:2404.14618* (2024).

[23] Energy.gov. 2024. Evaporative Coolers. https://www.energy.gov/energysaver/evaporative-coolers.

[24] Epoch AI. 2024. Data on Notable AI Models. https://epoch.ai/data/notable-ai-models Accessed: 2025-06-02.

[25] Ege Erdil. 2024. Frontier language models have become much smaller. https://epoch.ai/gradient-updates/frontier-language-models-have-become-much-smaller.

[26] Xiaobo Fan, Wolf-Dietrich Weber, and Luiz Andre Barroso. 2007. Power provisioning for a warehouse-sized computer. In *Proceedings of the 34th Annual International Symposium on Computer Architecture (ISCA '07)*. https://doi.org/10.1145/1250662.1250665

[27] Bin Gao, Zhuomin He, Puru Sharma, Qingxuan Kang, Djordje Jevdjic, Junbo Deng, Xingkun Yang, Zhou Yu, and Pengfei Zuo. 2024. Cost-Efficient Large Language Model serving for multi-turn conversations with CachedAttention. In *2024 USENIX Annual Technical Conference (USENIX ATC 24)*. 111–126.

[28] GitHub. 2024. The world's most widely adopted AI developer tool. https://github.com/features/copilot.

[29] Inigo Goiri, Kien Le, Jordi Guitart, Jordi Torres, and Ricardo Bianchini. 2011. Intelligent Placement of Datacenters for Internet Services. In *Proceedings of the 31st International Conference on Distributed Computing Systems*. https://doi.org/10.1109/ICDCS.2011.19

[30] Raja Gond, Nipun Kwatra, and Ramachandran Ramjee. 2025. TokenWeave: Efficient Compute-Communication Overlap for Distributed LLM Inference. arXiv:2505.11329 [cs.DC] https://arxiv.org/abs/2505.11329

[31] Google. 2025. Growing the internet while reducing energy consumption. https://datacenters.google/efficiency/.

[32] Google Cloud. 2025. Enabling 1 MW IT racks and liquid cooling at OCP EMEA Summit. https://cloud.google.com/blog/topics/systems/enabling-1-mw-it-racks-and-liquid-cooling-at-ocp-emea-summit.

Accessed on August 20, 2025.

[33] Grand View Research. 2025. U.S. Generative AI Market Size & Trends. https://www.grandviewresearch.com/industry-analysis/us-generative-ai-market-report.

[34] Albert Greenberg, James Hamilton, David A Maltz, and Parveen Patel. 2008. The cost of a cloud: research problems in data center networks. In *SIGCOMM*.

[35] GridStatus.io Team. 2025. GridStatus: Real-Time Power Grid Status Monitoring. https://www.gridstatus.io/. Accessed: 2025-09-29.

[36] Tim Grieser, Joeph Pucciarelli, Jean Bozman, and Randy Perry. 2010. *Managing the Server Migration Process: The HP Approach to Reducing Operational Costs*. Technical Report. HP.

[37] Groq. 2025. What is a Language Processing Unit? https://groq.com/blog/the-groq-lpu-explained.

[38] Albert Gu, Karan Goel, and Christopher Ré. 2022. Efficiently Modeling Long Sequences with Structured State Spaces. arXiv:2111.00396 [cs.LG] https://arxiv.org/abs/2111.00396

[39] Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. 2025. DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning. *arXiv:2501.12948* (2025).

[40] Wenzhe Guo, Joyjit Kundu, Uras Tos, Weijiang Kong, Giuliano Sisto, Timon Evenblij, and Manu Perumkunnil. 2025. System-performance and cost modeling of Large Language Model training and inference. arXiv:2507.02456 [cs.AR] https://arxiv.org/abs/2507.02456

[41] Udit Gupta, Mariam Elgamal, Gage Hills, Gu-Yeon Wei, Hsien-Hsin S. Lee, David Brooks, and Carole-Jean Wu. 2022. ACT: designing sustainable computer systems with an architectural carbon modeling tool. In *Proceedings of the 49th Annual International Symposium on Computer Architecture (ISCA '22)*.

[42] Bowen He, Xiao Zheng, Yuan Chen, Weinan Li, Yajin Zhou, Xin Long, Pengcheng Zhang, Xiaowei Lu, Linquan Jiang, Qiang Liu, Dennis Cai, and Xiantao Zhang. 2023. DxPU: Large-scale Disaggregated GPU Pools in the Datacenter. *ACM Trans. Archit. Code Optim.* 20, 4, Article 55 (Dec. 2023), 23 pages. https://doi.org/10.1145/3617995

[43] Ali Heydari, Bahareh Eslami, Vahideh Radmard, Fred Rebarber, Tyler Buell, Kevin Gray, Sam Sather, and Jeremy Rodriguez. 2022. Power Usage Effectiveness Analysis of a High-Density Air-Liquid Hybrid Cooled Data Center *(International Electronic Packaging Technical Conference and Exhibition, Vol. ASME 2022 International Technical Conference and Exhibition on Packaging and Integration of Electronic and Photonic Microsystems)*. V001T01A014. https://doi.org/10.1115/IPACK2022-97447

[44] Marius Hobbhahn, Lennart Heim, and Gökçe Aydos. 2023. Trends in Machine Learning Hardware. https://epoch.ai/blog/trends-in-machine-learning-hardware.

[45] Cheng-Yu Hsieh, Chun-Liang Li, Chih-Kuan Yeh, Hootan Nakhost, Yasuhisa Fujii, Alexander Ratner, Ranjay Krishna, Chen-Yu Lee, and Tomas Pfister. 2023. Distilling Step-by-Step! Outperforming Larger Language Models with Less Training Data and Smaller Model Sizes. arXiv:2305.02301 [cs.CL] https://arxiv.org/abs/2305.02301

[46] Chang-Hong Hsu, Qingyuan Deng, Jason Mars, and Lingjia Tang. 2018. SmoothOperator: Reducing Power Fragmentation and Improving Power Utilization in Large-Scale Datacenters. In *Proceedings of the 23rd International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*.

[47] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Lu Wang, and Weizhu Chen. [n. d.]. LoRA: Low-Rank Adaptation of Large Language Models. *arXiv:2106.09685* ([n. d.]).

[48] Urs Hölzle. 2022. Our commitment to climate-conscious data center cooling. https://blog.google/outreach-initiatives/sustainability/our-commitment-to-climate-conscious-data-center-cooling/.

[49] Intel. 2025. Intel® Xeon® 6980P Processor. https://www.intel.com/content/www/us/en/products/sku/240777/intel-xeon-6980p-processor-504m-cache-2-00-ghz/specifications.html.

[50] Intel. 2025. Intel® Xeon® Platinum 8593Q Processor. https://www.intel.com/content/www/us/en/products/sku/237252/intel-xeon-platinum-8593q-processor-320m-cache-2-20-ghz/specifications.html.

[51] Intel. 2025. Intel® Xeon® Processor E7-8890 v3. https://www.intel.com/content/www/us/en/products/sku/84685/intel-xeon-processor-e78890-v3-45m-cache-2-50-ghz/specifications.html.

[52] Kunal Jain, Anjaly Parayil, Ankur Mallick, Esha Choukse, Xiaoting Qin, Jue Zhang, Íñigo Goiri, Rujia Wang, Chetan Bansal, Victor Rühle, et al. 2024. Intelligent Router for LLM Workloads: Improving Performance Through Workload-Aware Load Balancing. *arXiv preprint arXiv:2408.13510* (2024).

[53] Majid Jalili, Ioannis Manousakis, Íñigo Goiri, Pulkit A. Misra, Ashish Raniwala, Husam Alissa, Bharath Ramakrishnan, Phillip Tuma, Christian Belady, Marcus Fontoura, and Ricardo Bianchini. 2021. Cost-Efficient Overclocking in Immersion-Cooled Datacenters. In *Proceedings of the 48th Annual International Symposium on Computer Architecture (ISCA)*.

[54] Youhe Jiang, Ran Yan, Xiaozhe Yao, Yang Zhou, Beidi Chen, and Binhang Yuan. 2024. HexGen: Generative Inference of Large Language Model over Heterogeneous Environment. *arXiv preprint arXiv:2311.11514* (2024).

[55] Norm Jouppi, George Kurian, Sheng Li, Peter Ma, Rahul Nagarajan, Lifeng Nai, Nishant Patil, Suvinay Subramanian, Andy Swing, Brian Towles, Clifford Young, Xiang Zhou, Zongwei Zhou, and David A Patterson. 2023. TPU v4: An Optically Reconfigurable Supercomputer for Machine Learning with Hardware Support for Embeddings. In *Proceedings of the 50th Annual International Symposium on Computer Architecture (ISCA '23)*.

[56] Keith Kirkpatrick and Daniel Newman. 2025. Microsoft Q3 FY 2025: Azure and AI Services Drive Strong Growth. https://futurumgroup.com/insights/microsoft-q3-fy-2025-earnings-beat-on-strong-cloud-and-ai-services-growth/.

[57] Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient Memory Management for Large Language Model Serving with PagedAttention. In *Proceedings of the Symposium on Operating Systems Principles (SOSP)*.

[58] Marina Lammertyn. 2024. 60+ ChatGPT Statistics And Facts You Need to Know in 2024. https://blog.invgate.com/chatgpt-statistics.

[59] Chuan Li. 2025. Tesla A100 Server Total Cost of Ownership Analysis. https://lambda.ai/blog/tesla-a100-server-total-cost-of-ownership.

[60] Jinhao Li, Jiaming Xu, Shan Huang, et al. 2025. Large Language Model Inference Acceleration: A Comprehensive Hardware Perspective. *arXiv preprint arXiv:2410.04466* (2025). https://arxiv.org/abs/2410.04466

[61] Shaohong Li, Xi Wang, Xiao Zhang, Vasileios Kontorinis, Sreekumar Kodakara, David Lo, and Parthasarathy Ranganathan. 2020. Thunderbolt: Throughput-Optimized, Quality-of-Service-Aware Power Capping at Scale. In *Proceedings of the 14th USENIX Symposium on Operating Systems Design and Implementation (OSDI)*.

[62] Yuzhuo Li, Mariam Mughees, Yize Chen, and Yunwei Ryan Li. 2024. The Unseen AI Disruptions for Power Grids: LLM-Induced Transients. arXiv:2409.11416 [cs.AR] https://arxiv.org/abs/2409.11416

[63] Yujun Lin, Haotian Tang, Shang Yang, Zhekai Zhang, Guangxuan Xiao, Chuang Gan, and Song Han. 2024. Qserve: W4a8kv4 quantization and system co-design for efficient llm serving. *arXiv preprint arXiv:2405.04532* (2024).

[64] Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. 2024. DeepSeek-V3 Technical Report. *arXiv:2412.19437* (2024).

[65] Yixuan Mei, Yonghao Zhuang, Xupeng Miao, Juncheng Yang, Zhihao Jia, and Rashmi Vinayak. 2025. Helix: Serving large language models

over heterogeneous gpus and network via max-flow. In *Proceedings of the International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*.

[66] Meta. 2024. 2024 Sustainability Report. https://sustainability.atmeta.com/2024-sustainability-report/.

[67] Meta. 2024. Llama2-70B. https://huggingface.co/meta-llama/Llama-2-70b.

[68] Meta. 2024. Llama3-70B. https://huggingface.co/meta-llama/Meta-Llama-3-70B-Instruct.

[69] Meta. 2025. The Llama 4 herd: The beginning of a new era of natively multimodal AI innovation. https://ai.meta.com/blog/llama-4-multimodal-intelligence/.

[70] Nicholas Metropolis and Stanislaw Ulam. 1949. The Monte Carlo Method. *J. Amer. Statist. Assoc.* 44, 247 (1949), 335–341.

[71] Xupeng Miao, Chunan Shi, Jiangfei Duan, Xiaoli Xi, Dahua Lin, Bin Cui, and Zhihao Jia. 2024. SpotServe: Serving Generative Large Language Models on Preemptible Instances. In *Proceedings of the International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*.

[72] Microsoft. 2025. Measuring datacenter energy and water use to improve Microsoft Cloud sustainability. https://datacenters.microsoft.com/sustainability/efficiency/.

[73] Seungjae Moon, Junseo Cha, Hyunjun Park, and Joo-Young Kim. 2025. Hybe: GPU-NPU Hybrid System for Efficient LLM Inference with Million-Token Context Window. In *Proceedings of the 52nd Annual International Symposium on Computer Architecture (ISCA '25)*.

[74] Jesse Noffsinger, Maria Goodpaster, Mark Patel, Haley Chang, Pankaj Sachdeva, and Arjita Bhan. 2025. The cost of compute: A $7 trillion race to scale data centers. https://www.mckinsey.com/industries/technology-media-and-telecommunications/our-insights/the-cost-of-compute-a-7-trillion-dollar-race-to-scale-data-centers.

[75] NVIDIA. 2024. Cooling and Airflow Optimization. https://docs.nvidia.com/dgx-superpod/design-guides/dgx-superpod-data-center-design-h100/latest/cooling.html.

[76] NVIDIA. 2024. DGX H100: AI for Enterprise. https://www.nvidia.com/en-us/data-center/dgx-h100/.

[77] NVIDIA. 2024. Introduction to the NVIDIA DGX A100 System. https://docs.nvidia.com/dgx/dgxa100-user-guide/introduction-to-dgxa100.html.

[78] NVIDIA. 2024. NVIDIA Blackwell Platform Arrives to Power a New Era of Computing. https://nvidianews.nvidia.com/news/nvidia-blackwell-platform-arrives-to-power-a-new-era-of-computing.

[79] NVIDIA. 2025. Chill Factor: NVIDIA Blackwell Platform Boosts Water Efficiency by Over 300x. https://blogs.nvidia.com/blog/blackwell-platform-water-efficiency-liquid-cooling-data-centers-ai-factories/.

[80] NVIDIA. 2025. LLM Model Pruning and Knowledge Distillation with NVIDIA NeMo Framework. https://developer.nvidia.com/blog/llm-model-pruning-and-knowledge-distillation-with-nvidia-nemo-framework/.

[81] NVIDIA. 2025. NVIDIA Collective Communications Library (NCCL). https://developer.nvidia.com/nccl.

[82] NVIDIA. 2025. NVIDIA ConnectX-7 Adapter Cards User Manual. https://docs.nvidia.com/networking/display/connectx7vpi/specifications.

[83] NVIDIA. 2025. NVIDIA Dynamo Platform. https://developer.nvidia.com/dynamo.

[84] NVIDIA. 2025. NVIDIA Quantum-2 InfiniBand Platform. https://www.nvidia.com/en-eu/networking/quantum2/.

[85] NVIDIA. 2025. NVLink and NVLink Switch. https://www.nvidia.com/en-us/data-center/nvlink/.

[86] NVIDIA. 2025. RDMA over Converged Ethernet (RoCE). https://docs.nvidia.com/networking/display/winofv55054000/rdma+over+converged+ethernet+(roce).

[87] OpenAI. 2025. Models. https://platform.openai.com/docs/models.

[88] Pratyush Patel, Esha Choukse, Chaojie Zhang, Íñigo Goiri, Brijesh Warrier, Nithish Mahalingam, and Ricardo Bianchini. 2024. Characterizing Power Management Opportunities for LLMs in the Cloud. In *Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*.

[89] Pratyush Patel, Esha Choukse, Chaojie Zhang, Aashaka Shah, Íñigo Goiri, Saeed Maleki, and Ricardo Bianchini. 2024. Splitwise: Efficient generative LLM inference using phase splitting. In *Proceedings of the 51st Annual International Symposium on Computer Architecture (ISCA)*.

[90] Cheng Peng, Xi Yang, Aokun Chen, Kaleb Smith, Nima PourNejatian, Anthony Costa, Cheryl Martin, Mona Flores, Ying Zhang, Tanja Magoc, Gloria Lipori, Mitchell Duane, Naykky Ospina, Mustafa Ahmed, William Hogan, Elizabeth Shenkman, Yi Guo, Jiang Bian, and Yonghui Wu. 2023. A study of generative large language model for medical research and healthcare. *npj Digital Medicine* (2023).

[91] Sundar Pichai. 2025. Q2 earnings call: CEO's remarks. https://blog.google/inside-google/message-ceo/alphabet-earnings-q2-2025/#introduction.

[92] Varun Sakalkar, Vasileios Kontorinis, David Landhuis, Shaohong Li, Darren De Ronde, Thomas Blooming, Anand Ramesh, James Kennedy, Christopher Malone, Jimmy Clidaras, et al. 2020. Data Center Power Oversubscription with a Medium Voltage Power Plane and Priority-Aware Capping. In *Proceedings of the International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*.

[93] Siddharth Samsi, Dan Zhao, Joseph McDonald, Baolin Li, Adam Michaleas, Michael Jones, William Bergeron, Jeremy Kepner, Devesh Tiwari, and Vijay Gadepally. 2023. From Words to Watts: Benchmarking the Energy Costs of Large Language Model Inference. In *Proceedings of the High Performance Extreme Computing Conference (HPEC)*.

[94] Amazon Web Services. 2025. Amazon EC2 P6e-GB200 UltraServers and P6-B200 instances. https://aws.amazon.com/blogs/aws/new-amazon-ec2-p6e-gb200-ultraservers-powered-by-nvidia-grace-blackwell-gpus-for-the-highest-ai-performance/.

[95] Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. 2017. Outrageously Large Neural Networks: The Sparsely-Gated Mixture-of-Experts Layer. arXiv:1701.06538 [cs.LG] https://arxiv.org/abs/1701.06538

[96] Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. 2019. Megatron-LM: Training Multi-Billion Parameter Language Models Using Model Parallelism. *arXiv:1909.08053* (2019).

[97] Jovan Stojkovic, Esha Choukse, Chaojie Zhang, Íñigo Goiri, and Josep Torrellas. 2024. Towards Greener LLMs: Bringing Energy-Efficiency to the Forefront of LLM Inference. *arXiv preprint arXiv:2403.20306* (2024).

[98] Jovan Stojkovic, Pulkit Misra, Íñigo Goiri, Sam Whitlock, Esha Choukse, Mayukh Das, Chetan Bansal, Jason Lee, Zoey Sun, Haoran Qiu, Reed Zimmermann, Savyasachi Samal, Brijesh Warrier, Ashish Raniwala, and Ricardo Bianchini. 2024. SmartOClock: Workload- and Risk-Aware Overclocking in the Cloud. In *Proceedings of the International Symposium on Computer Architecture (ISCA)*.

[99] Jovan Stojkovic, Chaojie Zhang, Íñigo Goiri, Esha Choukse, Haoran Qiu, Rodrigo Fonseca, Josep Torrellas, and Ricardo Bianchini. 2025. TAPAS: Thermal-and power-aware scheduling for LLM inference in cloud platforms. In *ASPLOS*. 1266–1281.

[100] Jovan Stojkovic, Chaojie Zhang, Íñigo Goiri, Josep Torrellas, and Esha Choukse. 2025. DynamoLLM: Designing LLM Inference Clusters for Performance and Energy Efficiency. In *Proceedings of the IEEE*

International Symposium on High Performance Computer Architecture (HPCA).

[101] Wei Su, Abhishek Dhanotia, Carlos Torres, Jayneel Gandhi, Neha Gholkar, Shobhit Kanaujia, Maxim Naumov, Kalyan Subramanian, Valentin Andrei, Yifan Yuan, and Chunqiang Tang. 2025. DCPerf: An Open-Source, Battle-Tested Performance Benchmark Suite for Datacenter Workloads. In *Proceedings of the 52nd Annual International Symposium on Computer Architecture (ISCA'25).*

[102] Team Uvation. 2025. Breaking Down the AI server data center cost. https://uvation.com/articles/breaking-down-the-ai-server-data-center-cost#maintenance-amp-support-contracts.

[103] Wendy Torell. 2025. Liquid vs. Air Cooling. Which is the Capex winner? https://blog.se.com/datacenter/architecture/2020/02/24/liquid-vs-air-cooling-which-is-the-capex-winner/.

[104] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is All You Need. *Advances in Neural Information Processing Systems (NeurIPS)* (2017).

[105] Bin Wang, Bojun Wang, Changyi Wan, Guanzhe Huang, Hanpeng Hu, Haonan Jia, Hao Nie, Mingliang Li, Nuo Chen, Siyu Chen, et al. 2025. Step-3 is Large yet Affordable: Model-system Co-design for Cost-effective Decoding. *arXiv preprint arXiv:2507.19427* (2025).

[106] Jaylen Wang, Daniel S Berger, Fiodar Kazhamiaka, Celine Irvene, Chaojie Zhang, Esha Choukse, Kali Frost, Rodrigo Fonseca, Brijesh Warrier, Chetan Bansal, et al. 2024. Designing cloud servers for lower carbon. In *ISCA*.

[107] Xiaorui Wang, Ming Chen, Charles Lefurgy, and Tom W Keller. 2009. SHIP: Scalable Hierarchical Power Control for Large-Scale Data Centers. In *Proceedings of the 18th International Conference on Parallel Architectures and Compilation Techniques (PACT).*

[108] Kitty Wheeler. 2025. How Amazon Achieved Rocketing Sales and Growth From AI. https://aimagazine.com/news/how-amazon-achieved-rocketing-sales-and-growth-from-ai.

[109] Qiang Wu, Qingyuan Deng, Lakshmi Ganesh, Chang-Hong Hsu, Yun Jin, Sanjeev Kumar, Bin Li, Justin Meza, and Yee Jiun Song. 2016. Dynamo: Facebook's Data Center-Wide Power Management System. In *Proceedings of the International Symposium on Computer Architecture (ISCA).*

[110] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation. *CoRR* abs/1609.08144 (2016). http://arxiv.org/abs/1609.08144

[111] Daliang Xu, Hao Zhang, Liming Yang, Ruiqi Liu, Gang Huang, Mengwei Xu, and Xuanzhe Liu. 2025. Fast On-device LLM Inference with NPUs. In *Proceedings of the 30th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 1 (ASPLOS '25).*

[112] Xiaohan Xu, Ming Li, Chongyang Tao, Tao Shen, Reynold Cheng, Jinyang Li, Can Xu, Dacheng Tao, and Tianyi Zhou. 2024. A survey on knowledge distillation of large language models. *arXiv preprint arXiv:2402.13116* (2024).

[113] Gyeong-In Yu, Joo Seong Jeong, Geon-Woo Kim, Soojeong Kim, and Byung-Gon Chun. 2022. Orca: A Distributed Serving System for Transformer-Based Generative Models. In *Proceedings of the 16th USENIX Symposium on Operating Systems Design and Implementation (OSDI).*

[114] Zhihang Yuan, Yuzhang Shang, Yang Zhou, Zhen Dong, Zhe Zhou, Chenhao Xue, Bingzhe Wu, Zhikai Li, Qingyi Gu, Yong Jae Lee, Yan Yan, Beidi Chen, Guangyu Sun, and Kurt Keutzer. 2024. LLM Inference Unveiled: Survey and Roofline Model Insights. arXiv:2402.16363 [cs.CL] https://arxiv.org/abs/2402.16363

[115] Chaojie Zhang, Alok Gautam Kumbhare, Ioannis Manousakis, Deli Zhang, Pulkit A. Misra, Rod Assis, Kyle Woolcock, Nithish Mahalingam, Brijesh Warrier, David Gauthier, Lalu Kunnath, Steve Solomon, Osvaldo Morales, Marcus Fontoura, and Ricardo Bianchini. 2021. Flex: High-Availability Datacenters with Zero Reserved Power. In *Proceedings of the 48th Annual International Symposium on Computer Architecture (ISCA).*

[116] Mary Zhang. 2025. How Much Does it Cost to Build a Data Center. https://dgtlinfra.com/how-much-does-it-cost-to-build-a-data-center/.

[117] Yu Zhang, Xiusi Chen, Bowen Jin, Sheng Wang, Shuiwang Ji, Wei Wang, and Jiawei Han. 2024. A Comprehensive Survey of Scientific Large Language Models and Their Applications in Scientific Discovery. arXiv:2406.10833 [cs.CL] https://arxiv.org/abs/2406.10833

[118] Yilong Zhao, Chien-Yu Lin, Kan Zhu, Zihao Ye, Lequn Chen, Size Zheng, Luis Ceze, Arvind Krishnamurthy, Tianqi Chen, and Baris Kasikci. 2024. Atom: Low-bit quantization for efficient and accurate llm serving. *Proceedings of Machine Learning and Systems* 6 (2024), 196–209.

[119] Youpeng Zhao Zhao, Di Wu Wu, and Jun Wang. 2024. ALISA: Accelerating Large Language Model Inference via Sparsity-Aware KV Caching. In *Proceedings of the 51st Annual International Symposium on Computer Architecture (ISCA).*

[120] Yinmin Zhong, Shengyu Liu, Junda Chen, Jianbo Hu, Yibo Zhu, Xuanzhe Liu, Xin Jin, and Hao Zhang. 2024. DistServe: Disaggregating Prefill and Decoding for Goodput-optimized Large Language Model Serving. In *18th USENIX Symposium on Operating Systems Design and Implementation (OSDI).*

[121] Zhe Zhou, Xuechao Wei, Jiejing Zhang, and Guangyu Sun. 2022. PetS: A Unified Framework for Parameter-Efficient Transformers Serving. In *Proceedings of the USENIX Annual Technical Conference (USENIX ATC).*

[122] Ruidong Zhu, Ziheng Jiang, Chao Jin, Peng Wu, Cesar A Stuardo, Dongyang Wang, Xinlei Zhang, Huaping Zhou, Haoran Wei, Yang Cheng, et al. 2025. MegaScale-Infer: Serving Mixture-of-Experts at Scale with Disaggregated Expert Parallelism. *arXiv preprint arXiv:2504.02263* (2025).

[123] Shivani Zoting. 2025. Artificial Intelligence (AI) Market Size and Growth 2025 to 2034. https://www.precedenceresearch.com/artificial-intelligence-market.