International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

Beyond Bigger Models: Multi-Axis Scaling for Large Language Models

Reeshav Kumar

Abstract: For years, the dominant prescription for building capable large language models (LLMs) was deceptively simple: make the model bigger. That logic produced real results with GPT-3's 175 billion parameters, delivering in-context learning across dozens of benchmarks. However, the assumption that parameter count is the primary lever for system quality is no longer tenable with Chinchilla demonstrating that most landmark models were significantly undertrained relative to their size. What followed was not a replacement of the scaling paradigm but a proliferation of it. Modern LLM systems are now shaped by nine distinct scaling axes: pretraining scale, compute-optimality, sparse conditional computation, retrieval and memory augmentation, long-context modeling, post-training alignment, parameter-efficient adaptation (PEFT), reasoning-time compute, and serving infrastructure optimization. This paper evaluates each axis against four operational dimensions (capability, cost, latency, and governance burden) and presents a product decision framework to minimize enterprise failure modes by selecting the most efficient scaling interventions. The central finding is that the AI systems that consistently outperform in production are those built on a coherent portfolio of axis choices, not those with the largest base model.

Keywords: *Large Language Models, Scaling Laws, Multi-Axis Scaling, Sparse Mixture-of-Experts, Retrieval-Augmented Generation, Post-Training Alignment, Inference Optimization, AI Governance*

1. Introduction

The architectural bet behind GPT-3 was straightforward, and it paid off. A 175-billion parameter autoregressive transformer, trained on web-scale text, could perform in-context learning across a remarkable breadth of tasks without any task-specific fine-tuning [2]. Kaplan et al. formalized the underlying logic through empirical scaling laws, characterizing a smooth power-law relationship — loss decreasing as approximately N^-0.076 with parameter count and D^-0.095 with data volume — that held across seven orders of magnitude [3]. The inference seemed obvious: scale the model, and capability follows. For several years, that was a productive heuristic.

The Chinchilla paper changed the terms of the argument. Hoffmann et al. demonstrated that the power-law scaling relationship had been misapplied by most frontier labs: for a fixed compute budget, training a smaller model on proportionally more tokens consistently outperformed training a larger model on fewer tokens [4]. The optimal ratio, Chinchilla established, is approximately 20 training tokens per parameter. By that standard, GPT-3 — trained on roughly 300 billion tokens against 175 billion parameters — was undertrained by a factor of more than ten relative to its compute-optimal point. DeepMind's Gopher, at 280 billion parameters and 300 billion training tokens, was similarly misallocated. Chinchilla's 70-billion parameter model, trained on 1.4 trillion tokens, outperformed both on most benchmarks. Parameter count, it turned out, was a poor proxy for model quality.

What emerged from that realignment was not a single corrected scaling recipe but an increasingly multi-dimensional design space. Organizations deploying LLMs in production face constraints that benchmark scores do not reflect: response latency bounded by user experience requirements, per-query cost constrained by unit economics, knowledge currency constrained by update cycles, and output behavior constrained by regulatory and organizational policy. Each of these constraints points to a different axis of the system — and optimizing any one axis without accounting for the others often produces a system that fails on the dimensions that were ignored. A model that scores

Independent Researcher, USA
ORCID ID: https://orcid.org/0009-0006-4522-8799

well on reasoning benchmarks but costs forty times as much to serve per query cannot be deployed at scale. A model with excellent factual recall that cannot be updated without a full retraining run is a liability in fast-moving domains.

This paper presents a multi-axis framework for analyzing and designing LLM systems for enterprise deployment. The framework identifies nine distinct scaling axes and evaluates each against four dimensions: capability gain, cost impact, latency profile, and governance burden. Section 2 defines the full taxonomy. Sections 3 through 5 analyze the axes in depth, grouped by where in the model lifecycle they operate. Section 6 presents the product decision framework and governance considerations. Section 7 concludes with implications for systems engineering practice.

2. The Multi-Axis Scaling Taxonomy

A scaling axis, as used in this paper, is any controllable dimension along which additional resources — compute, memory, data, or engineering investment — produce measurable improvement in LLM system behavior. The definition is intentionally broad: it encompasses interventions at pretraining time, post-training time, inference time, and serving infrastructure, because each of these stages contributes to the observed behavior of the deployed system and each carries its own economic and governance profile. Treating only one or two of these stages as "real" scaling conflates architectural research with systems engineering.

The nine axes form a loose lifecycle sequence. Pretraining scale and compute-optimality operate on the raw training run. Sparse conditional computation and retrieval augmentation operate on the model architecture and memory design. Long-context modeling spans both architecture and serving. Post-training alignment and PEFT operate after the base model is established. Reasoning-time compute operates at inference. Serving infrastructure optimization operates at the deployment layer. The axes are largely independent in their mechanisms — LoRA adaptation does not require any change to the retrieval pipeline, and speculative decoding does not require any change to the alignment technique — but they interact in their combined operational profiles. A system that adds reasoning-time compute without addressing serving infrastructure may find that extended reasoning traces exceed latency budgets. A system that deploys RAG without alignment review may introduce provenance risks that governance frameworks require to be tracked.

Four dimensions evaluate each axis throughout this paper. Capability refers to improvement in task success rate, error reduction, or output quality on the relevant task distribution. Cost refers to the incremental training, inference, or operational expenditure introduced by the axis. Latency refers to the axis's effect on the system's ability to meet response-time service level objectives (SLOs). Governance burden refers to the additional evaluation, monitoring, provenance tracking, or security review that the axis introduces. A lever that dramatically improves benchmark capability but doubles serving cost and triples governance overhead may not be the right investment for a cost-sensitive or compliance-heavy deployment. Table 1 summarizes the nine axes against all four dimensions.

Table 1. Multi-Axis Scaling Levers: Evaluation Across Four Operational Dimensions

Scaling Axis	Capability Impact	Cost Impact	Latency Impact	Governance Burden
Dense pretraining	High — foundational capability floor	Very high — dominant training cost	Negligible at inference	Medium — provenance tracking
Compute-optimality	High — same compute, lower loss	Medium — token budget reallocation	Negligible	Low
Sparse MoE	High — larger effective capacity	Medium — lower per-token FLOP	Low — routing overhead	High — expert load monitoring

Scaling Axis	Capability Impact	Cost Impact	Latency Impact	Governance Burden
Retrieval / RAG	High — freshness and grounding	Low — no retraining needed	Medium — retrieval latency added	High — data provenance, PII risk
Long-context modeling	Medium — more input capacity	High — quadratic attention cost	High — long decode sequences	Medium
Post-training alignment	High — compliance and usability	Medium — human or AI feedback	Negligible	High — bias, refusal auditing
PEFT (LoRA / QLoRA)	Medium — domain specialization	Very low — <1% of full fine-tune	Negligible	Low
Reasoning-time compute	High — complex task accuracy	High — per-query compute scales	High — extended generation	Medium — output variance
Serving optimization	Indirect — enables other axes	Low — hardware reuse	High reduction — key benefit	Low

3. Training-Time and Memory-Time Scaling

Dense pretraining establishes the capability floor on which every other scaling axis builds. GPT-3's 175 billion parameters delivered few-shot performance across 42 benchmark tasks [2]; PaLM extended the demonstration to 540 billion parameters across multilingual, code, and mathematical reasoning tasks [18]; GPT-4 added the insight that scaling behavior could be predicted from smaller runs, making frontier planning more tractable [19]. Each of these results confirmed that pretraining scale generates broad, transferable capability gains. What Chinchilla established is that those gains are achievable at lower parameter counts when the token budget is calibrated correctly — a 70-billion parameter model trained on 1.4 trillion tokens outperformed a 280-billion parameter model trained on 300 billion tokens, using the same total compute [4]. LLaMA 3 operationalized this principle at the open-access tier, training an 8-billion parameter model on over 15 trillion tokens and producing competitive performance against models with five to ten times as many parameters [20].

Sparse Mixture-of-Experts (MoE) architectures represent the most consequential structural innovation since the transformer itself. The core insight of Switch Transformer was that model capacity and per-token inference cost need not scale together: by routing each input token to a small subset of specialized expert sub-networks, the model can maintain a very large total parameter count while computing over only a fraction of those parameters for any given forward pass [5]. Switch Transformer demonstrated a pre-training speedup of more than 7x over a comparably sized T5 dense model at equivalent compute, using a 1.6-trillion parameter sparse model. Mixtral 8x7B extended this design to a production-viable form: 46.7 billion total parameters with 12.9 billion active per token, matching or exceeding Llama 2 70B on most standard benchmarks at substantially lower inference cost [6]. The trade-offs are real — expert routing introduces load-balancing complexity, communication overhead in distributed training, and serving infrastructure requirements beyond those of dense models — but the capacity-per-compute advantage has made sparse architectures the dominant design at the frontier.

Retrieval-augmented generation (RAG) addresses what pretraining fundamentally cannot: the staleness of parametric knowledge. A pretrained

model encodes facts as they existed in its training corpus; as those facts change, the model has no mechanism to update without retraining. RAG separates the knowledge storage problem from the language modeling problem by pairing the model with a non-parametric external index that is queried at inference time [7]. The retrieved documents are concatenated with the user query and passed to the model as context, conditioning generation on current, verifiable, and citable information without any modification to model weights. The operational advantage is significant: knowledge currency can be maintained by updating the retrieval index on any schedule the use case requires, at a fraction of the cost of retraining. GraphRAG extends the paradigm to enterprise-scale corpora by constructing a community-detection graph over the document collection, enabling global synthesis queries — "what themes appear across these 10,000 documents?" — that localized chunk retrieval cannot answer [8]. For enterprise applications, standard RAG and GraphRAG cover complementary query types: local and relational, respectively.

Long-context modeling introduces a different set of trade-offs. Recent models have demonstrated effective reasoning across context windows extending into the millions of tokens [9], an order-of-magnitude expansion from the 4,096-token windows typical in 2022. The capability gain is real: longer contexts allow the model to reason over full documents, multi-document sets, or extended conversation histories without truncation. The cost is equally real: standard attention computation scales quadratically with sequence length, making long-context processing substantially more expensive than short-context processing at equivalent model size. Critically, Liu et al. demonstrated that performance degrades for information positioned in the middle of long input sequences — models attend reliably to the beginning and end of a context but lose track of information sandwiched between them [10]. In production systems, this finding argues against naive full-context approaches: retrieval followed by selective long-context reasoning consistently outperforms either approach used in isolation, preserving the synthesis benefits of large context windows while mitigating the attention reliability degradation that plagues unconstrained long-context use. Table 2 situates these developments chronologically with quantitative results.

Table 2. Key LLM Scaling Milestones (2017–2025) with Quantitative Contributions

Year	Model / Work	Quantitative Contribution
2017	Transformer (Vaswani et al.)	Replaced LSTM with attention; enabled linear parallelization across sequence length
2020	GPT-3 (Brown et al.)	175B parameters; few-shot performance across 42 NLP benchmarks without fine-tuning
2020	Scaling Laws (Kaplan et al.)	Loss ~ N^-0.076 (params), D^-0.095 (data); power-law validated across 7 orders of magnitude
2022	Chinchilla (Hoffmann et al.)	Optimal: ~20 tokens per parameter; 70B model on 1.4T tokens outperformed 280B Gopher
2022	Switch Transformer (Fedus et al.)	7x+ pre-training speedup vs T5 at equivalent compute; 1.6T parameter sparse model
2022	InstructGPT (Ouyang et al.)	1.3B aligned model preferred over 175B GPT-3 by human raters on ~70% of prompts
2022	LoRA (Hu et al.)	Trainable parameters reduced ~10,000x; GPU memory reduced 3x vs full fine-tuning
2023	vLLM/PagedAttention (Kwon et al.)	Up to 24x throughput improvement over HuggingFace Transformers; near-zero memory waste
2024	Mixtral 8x7B (Jiang et al.)	12.9B active / 46.7B total params; matches or exceeds Llama 2

Year	Model / Work	Quantitative Contribution
		70B on most benchmarks
2025	DeepSeek-R1 (Guo et al.)	79.8% on AIME 2024; 97.3% on MATH-500; competitive with OpenAI o1 via RL-only training

4. Post-Training Alignment and Inference-Time Compute

Pretraining produces a model that predicts text — which is not the same as a model that is useful. The gap between these two states, it turns out, is substantial and consequential. InstructGPT quantified the gap precisely: when human raters were asked to compare the outputs of a 1.3-billion parameter aligned model against the outputs of the raw 175-billion parameter GPT-3 on the same prompt distribution, they preferred the smaller aligned model on approximately 70% of prompts [11]. A model more than 100 times smaller, once aligned, was more useful to the people it was meant to serve. That result has not been superseded. It established that post-training is not an optional polish step but a primary determinant of whether a capable pretrained model becomes a deployable product.

The alignment toolkit has expanded considerably since InstructGPT. Chung et al. demonstrated that instruction tuning across a diverse, large-scale task collection — including chain-of-thought (CoT) exemplars — improved zero-shot generalization on held-out tasks significantly, and that the gains scaled with both the number of tasks and model size [21]. Constitutional AI introduced a self-critique mechanism in which the model evaluates and revises its own outputs against a stated set of principles, reducing the dependence on direct human preference labels and making the alignment process more scalable at the frontier [22]. Direct Preference Optimization (DPO) simplified the pipeline further: by reformulating preference alignment as a direct supervised objective over pairwise comparisons, DPO eliminates the separate reward model and proximal policy optimization (PPO) reinforcement learning stage entirely, achieving comparable alignment quality at substantially lower computational and implementation cost [23]. These three methods are not mutually exclusive — they address different parts of the alignment problem and can be composed in production pipelines. Table 3 compares the principal alignment and adaptation methods across their key dimensions.

Table 3. Post-Training Alignment and Adaptation Methods: Comparative Overview

Method	Training Signal	RL Stage	Relative Cost	Key Empirical Result
InstructGPT / RLHF	Human pairwise preferences	PPO required	High	1.3B aligned > 175B raw GPT-3 (human raters)
Constitutional AI	AI-generated critique + revision	RLAIF	Medium	Harmlessness improved without human labelers at scale
DPO	Pairwise preferences — direct	None	Low	Eliminates reward model; comparable alignment quality
LoRA / QLoRA adaptation	Task demonstrations	None	Very low	QLoRA: 65B model fine-tuned on single 48GB GPU

Method	Training Signal	RL Stage	Relative Cost	Key Empirical Result
Chain-of-Thought + RLVR	Verifiable correctness signals	Yes	Medium	DeepSeek-R1: 79.8% AIME 2024 via RL alone

Parameter-efficient fine-tuning (PEFT) makes domain specialization economically accessible to organizations without frontier training resources. Low-Rank Adaptation (LoRA) addresses the adaptation problem by observing that the weight updates required to specialize a pretrained model for a narrow task tend to have low intrinsic rank [12]. Rather than updating all model weights during fine-tuning — which requires storing and computing gradients for billions of parameters — LoRA injects pairs of trainable low-rank matrices into the frozen weight matrices of the pretrained model, reducing the number of trainable parameters by approximately 10,000x and GPU memory requirements by 3x relative to full fine-tuning. QLoRA extends this to quantized base models: by loading the pretrained model in 4-bit precision and applying LoRA adapters in 16-bit, Dettmers et al. demonstrated that a 65-billion parameter model could be fine-tuned on a single 48-gigabyte GPU — a task that would have required multiple A100s under standard fine-tuning [13]. The practical implication is that organizations operating outside the frontier training tier can still achieve strong, domain-specific performance through PEFT adaptation of publicly available base models.

Reasoning-time compute has emerged as the most recently recognized major scaling axis. Wei et al. demonstrated that prompting a large model to produce intermediate reasoning steps before a final answer — chain-of-thought prompting — dramatically improved performance on multi-step reasoning tasks: three-digit addition accuracy increased from 18.8% to 98.8% at sufficient model scale, and gains appeared systematically across arithmetic, symbolic, and commonsense reasoning categories [14]. Wang et al. showed that sampling multiple independent CoT reasoning traces and selecting the most consistent final answer (self-consistency) improved accuracy further, treating generation as a sampling problem rather than a greedy search [15]. Recent training-level work pushed the principle into the model itself: OpenAI's reasoning-focused model line demonstrated that training models to engage in extended deliberation before responding — essentially internalizing CoT at the training level through reinforcement learning — produces consistent capability gains on hard tasks [24]. DeepSeek-R1 made the mechanism explicit: through reinforcement learning from verifiable rewards alone, without any supervised CoT examples, the model learned to generate long reasoning traces that achieved 79.8% on AIME 2024 and 97.3% on the MATH-500 benchmark, competitive with OpenAI o1 [25]. The architectural implication is fundamental: capability is no longer fixed at the end of pretraining. It scales within individual queries, subject to latency and cost constraints.

5. Serving Infrastructure and Hardware Co-Design

The serving layer is where the capabilities built across all other scaling axes either become viable in production or collapse under economic and latency constraints. Most early LLM serving systems managed the key-value (KV) cache — the memory structure that stores attention states for previously generated tokens during autoregressive decoding — as contiguous memory blocks allocated at request initiation. This approach leads to severe fragmentation: because the final length of a generated response is not known in advance, serving systems must either over-allocate (wasting GPU memory) or under-allocate (causing request failures). Kwon et al. addressed this systematically with PagedAttention, which manages KV cache memory in variable-sized, non-contiguous pages analogous to virtual memory in operating systems [16]. The vLLM serving system, which implements PagedAttention, demonstrated throughput improvements of up to 24x over HuggingFace Transformers and up to 3.5x over FasterTransformer under equivalent hardware, with near-zero KV cache memory waste. At enterprise serving volumes — millions of queries per day — a 24x throughput improvement translates directly to a 24x reduction in serving hardware cost at

constant throughput, or a 24x increase in throughput at constant hardware cost.

Speculative decoding targets a complementary bottleneck: the sequential nature of autoregressive token generation. Standard autoregressive decoding generates exactly one token per forward pass through the large target model, regardless of how straightforward the continuation is. Leviathan et al. addressed this by introducing a small, fast draft model that proposes multiple candidate tokens in parallel; the large target model then verifies all candidate tokens in a single forward pass and accepts those that match its own distribution [17]. For straightforward continuations — common phrases, code completions, formulaic transitions — the draft model's proposals are frequently accepted, allowing the system to generate multiple tokens per large-model forward pass and reducing wall-clock latency without any change to the output distribution. The technique is composable with PagedAttention: the two address different bottlenecks (memory fragmentation versus sequential generation latency), and their benefits are largely additive in production deployments.

At the hardware level, recent GPU architectures have explicitly reoriented around the operational profile of deployed LLM systems rather than the profile of the original training run [26]. Post-training iteration, inference-time compute, and retrieval-augmented serving at scale now represent larger total expenditure than the original pretraining run for most enterprise deployments, and hardware design has followed that shift. MLPerf Inference v5.0 extended its evaluation scope to include time-to-first-token and per-output-token throughput metrics alongside traditional offline accuracy metrics, reflecting the recognition that production system quality is measured in latency and serving economics, not benchmark scores alone [27]. FlashAttention addresses the attention computation bottleneck through IO-aware kernel design: by fusing the attention computation into a single GPU kernel that avoids materializing the full attention matrix in high-bandwidth memory, it reduces memory traffic by up to 4x and delivers end-to-end training speedups of 15–20% on BERT and 3x on GPT-2 compared to standard implementations [28]. Taken together, these serving and hardware advances determine which combinations of scaling axes are economically deployable — making serving infrastructure a gating constraint on the rest of the multi-axis stack.

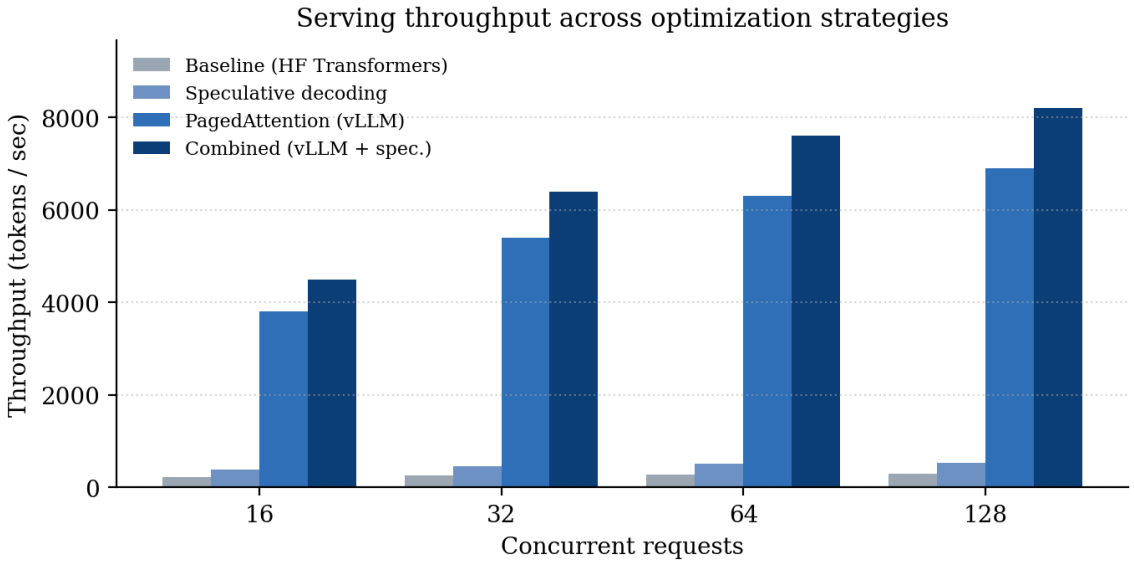

Fig. 1. Serving throughput (tokens/sec) for baseline HuggingFace Transformers, speculative decoding, PagedAttention (vLLM), and the combined configuration under 16, 32, 64, and 128 concurrent requests. Values are indicative; precise ratios depend on hardware, model size, and request distribution.

6. Governance, Evaluation, and Product Decision Framework

The nine scaling axes are not equally available to every organization, and they are not equally safe to deploy without governance controls. Treating axis selection as a purely technical portfolio problem misses the governance dimension that determines what is permissible in production. The National Institute of Standards and Technology (NIST) AI Risk Management Framework (AI RMF 1.0)

frames risk management as a lifecycle-wide activity — one that begins at data collection, runs through training and evaluation, and continues through deployment and monitoring — not a one-time compliance check performed at launch [29]. NIST's Generative AI profile (NIST AI 600-1) extends this to LLM-specific risks: hallucination, harmful content generation, data provenance failures, and emergent capabilities that evade pre-deployment testing [30]. For multi-axis systems, governance burden accumulates across axes: retrieval introduces data provenance and personally identifiable information (PII) risks; alignment introduces bias evaluation and refusal auditing requirements; reasoning-time compute introduces output variance and verification challenges. Systems engineers must account for the total governance burden of the axis combination, not just the governance profile of each axis in isolation.

ISO/IEC 42001:2023 establishes AI governance as a management system with continuous monitoring obligations, analogous in structure to ISO 9001 for quality management [30]. This framing matters for multi-axis LLM systems because it positions governance as an ongoing operational function rather than a one-time compliance gate. The OWASP LLM Top 10 catalogs application-level risks specific to LLM deployments: prompt injection (external inputs manipulating model behavior), insecure output handling (unvalidated model outputs passed to downstream systems), sensitive information disclosure, training data poisoning, and overreliance [31]. Each risk category maps onto specific axis combinations — retrieval-augmented systems face distinct prompt injection and sensitive information disclosure risks; alignment-reliant systems face overreliance risks if the alignment policy is treated as a complete behavioral specification rather than a first-order control. A mature multi-axis deployment design addresses OWASP risks at the architecture level, not as an afterthought.

Aggregate evaluation metrics are insufficient for enterprise AI systems where failure modes are heterogeneous across deployment contexts. Conneau et al. demonstrated that cross-lingual model improvements at the aggregate level can conceal significant degradation for specific language groups when model capacity is held constant [32]. BIG-bench established a parallel phenomenon at the task level: broad benchmark averages can mask substantial variance in difficulty tiers, with some model families improving on easy tasks while regressing on hard ones [33]. Enterprise AI evaluation must therefore be disaggregated — by geography, language, customer segment, task difficulty tier, and risk category — with particular attention to the segments where failure consequences are highest. A model that improves globally while degrading on the queries that represent the highest regulatory risk or highest revenue concentration is not an improvement in any operationally meaningful sense.

The product decision framework synthesizes the technical and governance analysis into a structured diagnostic. The entry point is the observed failure mode. When the failure is factual staleness or inaccuracy on recent events, retrieval augmentation is the first intervention — standard RAG for local passage retrieval, GraphRAG for cross-document synthesis tasks [7], [8]. When the failure is policy non-compliance or output inconsistency, post-training alignment is the appropriate axis — DPO for pipelines that require simplicity and cost efficiency, Constitutional AI for pipelines that require scalable, principle-based control [22], [23]. When the failure is domain-specific accuracy degradation on narrow, stable task distributions where labeled data is available, PEFT adaptation via QLoRA provides the best capability-per-cost ratio — single-GPU fine-tuning of a 65-billion parameter model is now feasible [13]. When the failure is concentrated on high-stakes, computationally bounded tasks — complex reasoning, multi-step analysis, scientific synthesis — reasoning-time compute allocation is appropriate, with CoT and self-consistency providing the primary mechanism [14], [15]. When the failure is serving economics — throughput, latency, or hardware utilization — PagedAttention and speculative decoding must be addressed before any other axis investment, as a serving bottleneck limits the deployment viability of every other axis [16], [17]. Table 4 presents the full framework in structured form.

Table 4. Product Decision Framework: Enterprise Failure Mode to Scaling Axis Mapping

Enterprise Failure Mode	Recommended Axis	Primary Technique	Expected Outcome
Factual staleness / inaccuracy on recent events	Retrieval / RAG	GraphRAG over indexed corpus	Grounded, citable, up-to-date outputs
Policy non-compliance or inconsistent refusals	Post-training alignment	DPO or Constitutional AI	Auditable, consistent behavioral policy
Domain-specific accuracy gaps with stable data	PEFT adaptation	QLoRA on labeled domain examples	Cost-efficient specialization — single GPU feasible
Errors concentrated on high-stakes complex tasks	Reasoning-time compute	CoT + self-consistency sampling	Improved accuracy on hard inputs at higher per-query cost
Serving cost exceeds unit economics at scale	Serving optimization	PagedAttention + speculative decoding	Up to 24x throughput; significant latency reduction
Cross-document synthesis failures in enterprise search	GraphRAG	Graph-indexed corpus with community detection	Global query answering across large document sets

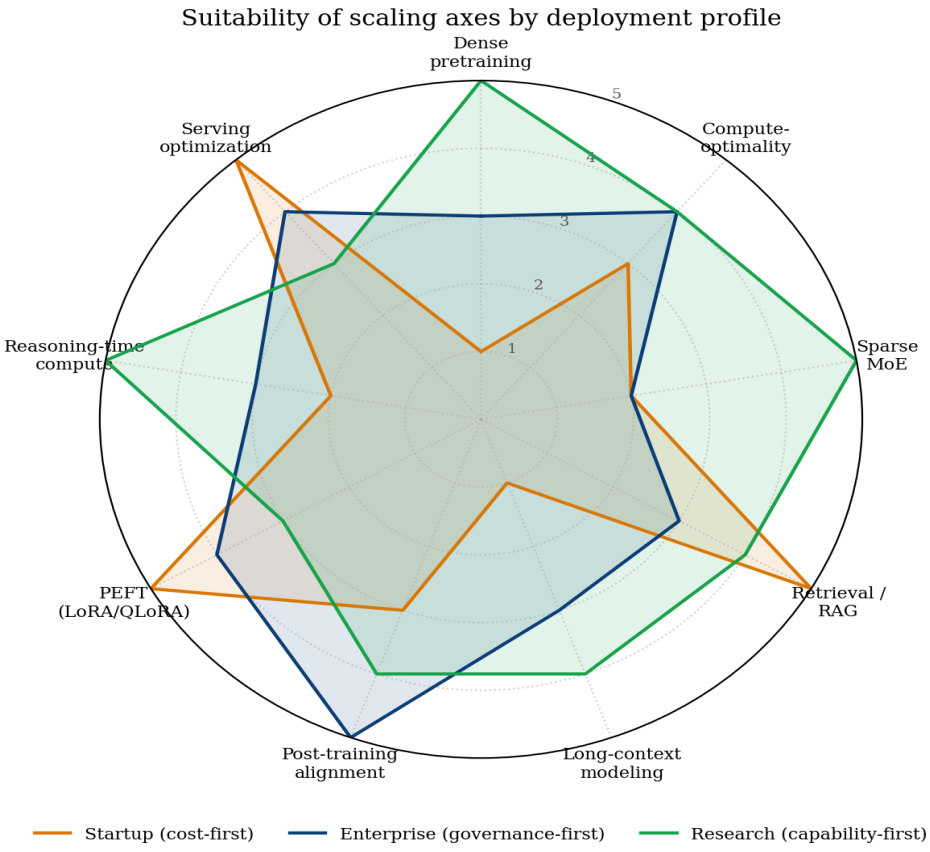

Fig. 2. Relative suitability of the nine scaling axes across three deployment profiles — startup (cost-first), enterprise (governance-first), and research (capability-first). Scores (1–5) are qualitative and derived from the operational trade-offs summarized in Table 1.

7. Conclusion

The central argument of this paper is that the most consequential design decisions in enterprise LLM deployment are no longer about selecting the largest available base model — they are about identifying which combination of scaling axes most efficiently addresses the actual failure modes observed in production, within the constraints of the deployment context. The evidence assembled here supports this claim across every dimension examined. Chinchilla demonstrated that compute-optimal training at 20 tokens per parameter produces better models than parameter maximization under equivalent compute [4]. Switch Transformer and Mixtral showed that sparse MoE architectures can deliver larger effective capacity at lower per-token inference cost [5], [6]. RAG and GraphRAG demonstrated that knowledge freshness and cross-document synthesis are better addressed through retrieval architecture than through larger parametric models [7], [8]. InstructGPT established that alignment transforms the utility of a model independently of its size — a 1.3-billion parameter aligned model outperforming a 175-billion parameter unaligned one on user-facing tasks [11]. LoRA and QLoRA made domain adaptation accessible to organizations without frontier training resources [12], [13]. Reasoning-time compute, as demonstrated by DeepSeek-R1's 79.8% on AIME 2024, showed that capability continues to scale within individual requests through principled inference-time allocation [25]. PagedAttention demonstrated that serving infrastructure is a first-order economic lever, not a deployment detail [16].

For systems engineers and product leaders, this multi-axis view has a practical implication that single-axis reasoning misses: the right question is not "which model should we use?" but "which axis combination addresses our failure mode profile within our operational envelope?" Those are different questions, and they lead to different investments. An organization experiencing serving economics pressure should address PagedAttention before evaluating larger base models. An organization experiencing domain accuracy failures should evaluate QLoRA adaptation before evaluating alignment techniques designed for a different class of failure. An organization experiencing policy compliance failures should address post-training alignment before increasing retrieval coverage. The framework presented here provides a structured basis for making those diagnostic decisions without defaulting to the comfortable but often incorrect heuristic of "use a bigger model."

Several open problems remain at the intersection of multi-axis scaling and systems engineering practice. First, axis interaction effects are not well characterized empirically: how PEFT adaptation affects alignment properties, how reasoning-time compute interacts with retrieval latency under tight SLOs, and how governance burden accumulates across combined axis deployments are all areas where the current literature offers limited direct guidance. Second, the governance frameworks cited here — NIST AI RMF, ISO/IEC 42001, OWASP LLM Top 10 — were developed largely against single-model deployment patterns and will require extension to address the full complexity of multi-axis systems. Third, as reasoning-time compute scales and long-context windows expand, serving infrastructure will face increasing pressure to evolve beyond current PagedAttention and speculative decoding approaches. These open problems define the near-term research and engineering agenda for multi-axis LLM system design.

References

[1] A. Vaswani et al., "Attention is all you need," in Advances in Neural Information Processing Systems, vol. 30, 2017. [Online]. Available: https://doi.org/10.48550/arXiv.1706.03762

[2] T. Brown et al., "Language models are few-shot learners," in Advances in Neural Information Processing Systems, vol. 33, pp. 1877–1901, 2020. [Online]. Available: https://doi.org/10.48550/arXiv.2005.14165

[3] J. Kaplan et al., "Scaling laws for neural language models," arXiv preprint arXiv:2001.08361, 2020. [Online]. Available: https://doi.org/10.48550/arXiv.2001.08361

[4] J. Hoffmann et al., "Training compute-optimal large language models," arXiv preprint arXiv:2203.15556, 2022. [Online]. Available: https://doi.org/10.48550/arXiv.2203.15556

[5] W. Fedus, B. Zoph, and N. Shazeer, "Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity," Journal of Machine Learning Research, vol. 23, no. 120, pp. 1–39, 2022. [Online]. Available: https://jmlr.org/papers/v23/21-0243.html

[6] A. Q. Jiang et al., "Mixtral of experts," arXiv preprint arXiv:2401.04088, 2024. [Online]. Available: https://doi.org/10.48550/arXiv.2401.04088
[7] P. Lewis et al., "Retrieval-augmented generation for knowledge-intensive NLP tasks," in Advances in Neural Information Processing Systems, vol. 33, pp. 9459–9474, 2020. [Online]. Available: https://doi.org/10.48550/arXiv.2005.11401
[8] D. Edge et al., "From local to global: A graph RAG approach to query-focused summarization," arXiv preprint arXiv:2404.16130, 2024. [Online]. Available: https://doi.org/10.48550/arXiv.2404.16130
[9] Google DeepMind, "Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context," arXiv preprint arXiv:2403.05530, 2024. [Online]. Available: https://doi.org/10.48550/arXiv.2403.05530
[10] N. F. Liu et al., "Lost in the middle: How language models use long contexts," Transactions of the Association for Computational Linguistics, vol. 12, pp. 157–173, 2024. [Online]. Available: https://doi.org/10.1162/tacl_a_00638
[11] L. Ouyang et al., "Training language models to follow instructions with human feedback," in Advances in Neural Information Processing Systems, vol. 35, pp. 27730–27744, 2022. [Online]. Available: https://doi.org/10.48550/arXiv.2203.02155
[12] E. J. Hu et al., "LoRA: Low-rank adaptation of large language models," in International Conference on Learning Representations, 2022. [Online]. Available: https://doi.org/10.48550/arXiv.2106.09685
[13] T. Dettmers et al., "QLoRA: Efficient finetuning of quantized LLMs," in Advances in Neural Information Processing Systems, vol. 36, 2023. [Online]. Available: https://doi.org/10.48550/arXiv.2305.14314
[14] J. Wei et al., "Chain-of-thought prompting elicits reasoning in large language models," in Advances in Neural Information Processing Systems, vol. 35, pp. 24824–24837, 2022. [Online]. Available: https://doi.org/10.48550/arXiv.2201.11903
[15] X. Wang et al., "Self-consistency improves chain of thought reasoning in language models," in International Conference on Learning Representations, 2023. [Online]. Available: https://doi.org/10.48550/arXiv.2203.11171
[16] W. Kwon et al., "Efficient memory management for large language model serving with PagedAttention," in Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles, pp. 611–626, 2023. [Online]. Available: https://doi.org/10.1145/3600006.3613165
[17] Y. Leviathan, M. Kalman, and Y. Matias, "Fast inference from transformers via speculative decoding," in Proceedings of the 40th International Conference on Machine Learning, pp. 19274–19286, 2023. [Online]. Available: https://doi.org/10.48550/arXiv.2211.17192
[18] A. Chowdhery et al., "PaLM: Scaling language modeling with pathways," Journal of Machine Learning Research, vol. 24, no. 240, pp. 1–113, 2023. [Online]. Available: https://doi.org/10.48550/arXiv.2204.02311
[19] OpenAI, "GPT-4 technical report," arXiv preprint arXiv:2303.08774, 2023. [Online]. Available: https://doi.org/10.48550/arXiv.2303.08774
[20] AI@Meta, "The Llama 3 herd of models," arXiv preprint arXiv:2407.21783, 2024. [Online]. Available: https://doi.org/10.48550/arXiv.2407.21783
[21] H. W. Chung et al., "Scaling instruction-finetuned language models," Journal of Machine Learning Research, vol. 25, no. 70, pp. 1–53, 2024. [Online]. Available: https://doi.org/10.48550/arXiv.2210.11416
[22] Y. Bai et al., "Constitutional AI: Harmlessness from AI feedback," arXiv preprint arXiv:2212.08073, 2022. [Online]. Available: https://doi.org/10.48550/arXiv.2212.08073
[23] R. Rafailov et al., "Direct preference optimization: Your language model is secretly a reward model," in Advances in Neural Information Processing Systems, vol. 36, 2023. [Online]. Available: https://doi.org/10.48550/arXiv.2305.18290
[24] OpenAI, "Learning to reason with LLMs," OpenAI Blog, Sep. 2024. [Online]. Available: https://openai.com/index/learning-to-reason-with-llms/
[25] D. Guo et al., "DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning," arXiv preprint arXiv:2501.12948, 2025. [Online]. Available: https://doi.org/10.48550/arXiv.2501.12948
[26] NVIDIA Corporation, "NVIDIA Blackwell architecture technical brief," NVIDIA, 2024. [Online]. Available:

https://resources.nvidia.com/en-us-blackwell-architecture
[27] MLCommons, "MLPerf inference v5.0 results," MLCommons, 2025. [Online]. Available: https://mlcommons.org/benchmarks/inference-datacenter/
[28] T. Dao et al., "FlashAttention: Fast and memory-efficient exact attention with IO-awareness," in Advances in Neural Information Processing Systems, vol. 35, pp. 16344–16359, 2022. [Online]. Available: https://doi.org/10.48550/arXiv.2205.14135
[29] NIST, "Artificial intelligence risk management framework (AI RMF 1.0)," National Institute of Standards and Technology, NIST AI 100-1, 2023. [Online]. Available: https://doi.org/10.6028/NIST.AI.100-1
[30] NIST, "Artificial intelligence risk management framework: Generative artificial intelligence profile," National Institute of Standards and Technology, NIST AI 600-1, 2024. [Online]. Available: https://doi.org/10.6028/NIST.AI.600-1
[31] ISO/IEC, "ISO/IEC 42001:2023 — Information technology — Artificial intelligence — Management system," International Organization for Standardization, 2023. [Online]. Available: https://www.iso.org/standard/81230.html
[32] OWASP, "OWASP top 10 for large language model applications v1.1," OWASP Foundation, 2023. [Online]. Available: https://owasp.org/www-project-top-10-for-large-language-model-applications/
[33] A. Conneau et al., "Unsupervised cross-lingual representation learning at scale," in Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pp. 8440–8451, 2020. [Online]. Available: https://doi.org/10.18653/v1/2020.acl-main.747
[34] A. Srivastava et al., "Beyond the imitation game: Quantifying and extrapolating the capabilities of language models," Transactions on Machine Learning Research, 2023. [Online]. Available: https://doi.org/10.48550/arXiv.2206.04615