The End of the One-Model Era

Aman Pathak
Associate Researcher, ENineHQ Technologies PVT LTD
aman@eninehq.com

May 27, 2026

Abstract

Generalist large language models represent a transitional architecture whose limitations become increasingly apparent under rigorous technical scrutiny. The scaling laws that motivated the paradigm are reaching structural constraints: the cross-entropy loss improvement from doubling compute has fallen below 0.05 nats for frontier models, data stock is projected for exhaustion between 2026 and 2032, and the transformer architecture is formally bounded by TC0 expressiveness limitations. Controlled evaluations across legal, medical, and financial domains reveal hallucination rates of 58 to 88 percent for generalist models on domain-specific queries. A roofline model analysis shows that autoregressive decode for a 220-billion-parameter active model achieves an arithmetic intensity of only 2.0 FLOPs/byte on H100 hardware, two orders of magnitude below the compute-bound threshold of 330 FLOPs/byte, making inference costs structurally dominated by memory bandwidth rather than computation. Domain-adapted models circumvent this constraint: a 7-billion-parameter specialized model requires 35 times less memory-bandwidth time per token while matching or exceeding generalist accuracy on target tasks. The emergence of parameter-efficient fine-tuning, quantized deployment, and Mixture-of-Experts routing indicates that the industry is converging on specialized computation rather than monolithic generalization. This synthesis argues that the future of applied natural language processing lies in an ecosystem of narrow, task-optimized systems.

Conceptual Limitations of the Generalist Paradigm

François Chollet provided an early theoretical framework for understanding these limitations in his 2019 paper on the measurement of intelligence. Chollet formalized intelligence as the efficiency with which a system acquires new skills through experience, measured by the ARC (Abstraction and Reasoning Corpus) benchmark. He demonstrated through controlled experiments that the performance of large language models on standardized benchmarks measures skill acquisition rather than general intelligence: skill consists of the ability to retrieve and apply memorized patterns from a training distribution, while general intelligence requires the capacity to adapt to novel situations using minimal data. His core result was that these two capacities are orthogonal and that the scaling of model parameters primarily increases the former without addressing the latter [1].

The scaling laws that motivated the generalist paradigm merit closer formal examination. Kaplan et al. established that the cross-entropy loss of a transformer follows a power-law relationship with model parameters and dataset size: $L(N, D) = L_0 + (N_c/N)^{\alpha_N} + (D_c/D)^{\alpha_D}$, where $\alpha_N \approx 0.076$ and $\alpha_D \approx 0.103$ for the parameter and data exponents [2]. Hoffmann et al. at DeepMind subsequently demonstrated that this formulation had significantly overestimated the optimal parameter-to-data ratio. Their Chinchilla scaling law established that for compute-optimal training, parameters and tokens scale as $N_{\text{opt}} \propto C^{1/2}$ and $D_{\text{opt}} \propto C^{1/2}$ for a compute budget C, meaning most pre-2022

models were undertrained by factors of 4–10 [3]. The practical consequence was a wave of smaller, better-trained models that matched larger predecessors.

The diminishing returns evident in recent model generations are consistent with the Chinchilla framework applied to saturated benchmarks. Marcus documented that the MMLU improvement from GPT-3 to GPT-4 was approximately 27 percentage points, while the improvement from GPT-4 to GPT-4 Turbo over the subsequent eight months was fewer than 3 points [4]. Under the Chinchilla scaling law, the compute investment required for each additional point of MMLU improvement follows a power-law escalation: reducing the loss by a fixed increment demands a compute multiplier of $1/(1 - \alpha_N - \alpha_D) \approx 1.22$ in the exponent. After the first 80 percent of achievable performance has been captured, each remaining point costs more than the entire preceding training run. This creates an incentive structure where further pre-training investment for marginal benchmark gains is increasingly difficult to justify compared with domain-specific fine-tuning at a fraction of the compute cost.

This distinction between skill acquisition and reasoning has been confirmed by rigorous empirical work. In October 2024, a team at Apple published a study using GSM-NoOp, a modified version of the grade-school mathematics dataset GSM8K in which controlled irrelevant textual variations were introduced. The perturbations included name substitution, numerical variation, clause insertion, and narrative restructuring while preserving mathematical structure. The study found that changing proper names within problems altered accuracy by approximately 10 percent, that adding irrelevant clauses reduced accuracy by up to 30 percent on GPT-4 and Claude 3.5, and that the models showed systematic inconsistency: the same mathematical operation would succeed or fail depending on semantically irrelevant surface features. The authors concluded that observed performance was better explained by sophisticated pattern matching against training distribution analogues than by abstract reasoning [5]. Gary Marcus characterized these results as demonstrating that reliable agentic systems cannot be constructed on a foundation where semantically irrelevant input perturbations produce different outputs [6].

A complementary line of theoretical research has identified fundamental expressiveness limitations in the transformer architecture itself. Merrill and Sabharwal demonstrated that the transformer with log-precision computations lies within the complexity class TC0, the set of problems solvable by constant-depth threshold circuits with polynomially many wires [7]. This places provable limits on the computations transformers can perform: they cannot reliably execute function composition requiring super-constant depth, they cannot model hierarchical structures with long-range dependencies that exceed their effective receptive field, and they cannot solve problems requiring counting or parity operations without chain-of-thought augmentation, which effectively adds computational depth through sequential token generation. These limitations are architectural rather than parametric. No increase in scale within the transformer framework can overcome a TC0 expressiveness bound. This finding implies that for any domain requiring reliable multi-step reasoning, the model must either be supplemented with external computation (tools, code execution, symbolic solvers) or be trained on a sufficiently narrow distribution that the required reasoning path is memorized, which is exactly the specialization thesis.

The scaling paradigm faces an additional structural constraint on the supply of training data. The 2024 Epoch AI analysis estimated that the stock of publicly available human-generated text has a size of approximately 310 trillion tokens, with current growth rates of high-quality text data at roughly 5 percent per year. Under current training scaling trends, this stock will be exhausted

between 2026 and 2032 [8]. Synthetic data generated by other models offers only a partial solution: the marginal value of synthetic tokens decreases with each generation due to distributional drift and mode collapse, and the risk of model collapse (where successive generations of synthetic data amplify sampling errors) is well-documented for autoregressive models. This constraint applies with particular force to domain-specialized applications: the availability of high-quality annotated data in medicine, law, and engineering is several orders of magnitude smaller than the general web corpus, and the cost of expert annotation per token is higher by factors of 100–1000.

Economic Constraints and Pricing Distortions

The inference cost of frontier language models imposes a structural constraint on their deployment at scale. Reiner Pope, a former Google TPU architect, provided a detailed analysis of the batch economics underlying LLM inference. A model such as DeepSeek V3, with approximately 700 billion parameters, requires loading its full weight matrix from memory for every token generated. When a single user is served, the cost of this memory fetch is borne entirely by that request. When hundreds of users are batched together, the cost is amortized across the batch. The consequence is that latency and cost are in direct tension: low-latency inference requires small batches, which implies high per-request cost [9]. This tension is inherent to the transformer decoder architecture and does not diminish with scale.

Quantitative Analysis of Inference Costs

The cost of serving a frontier model decomposes into three analytically distinct components: memory transfer of model parameters, matrix computations of the forward pass, and key-value cache maintenance. Each scales differently with architecture, sequence length, and batch size.

A transformer decoder with N active parameters, hidden dimension d, and L layers must read its weight matrices from HBM for every token. At FP8 precision, the standard format for production inference, the weight data transferred per token is N bytes. The H100 GPU provides approximately 3 TB/s of HBM3 bandwidth across 80 GB of HBM3 memory with 6144-bit interface width [10]. The minimum per-token latency in the memory-bound regime is:

$$t_{\text{mem}} = \frac{N}{B}$$

Evaluating with $N = 220 \times 10^9$ (the estimated active parameter count of GPT-4 [11]) and $B = 3 \times 10^{12}$ bytes/second:

$$t_{\text{mem}} = \frac{2.2 \times 10^{11}}{3 \times 10^{12}} \approx 0.073 \text{ s}$$

This yields approximately 14 tokens per second as a hard lower bound for a single-user request. The bound is architectural: it is determined by the ratio of parameter count to memory bandwidth. The dense predecessor GPT-3, with 175 billion parameters, had an equivalent bound of approximately 17 tokens per second; the transition to Mixture-of-Experts has not materially improved single-user latency because the active parameter count remains comparable.

A complete characterization requires the roofline model, which relates arithmetic intensity I (FLOPs per byte of memory traffic) to achievable throughput. For a single decode step:

$$I = \frac{\text{FLOPs}}{\text{bytes}} = \frac{2N}{N + M_{\text{kv}}}$$

where $2N$ is the FLOP count (one matrix-vector multiply) and the denominator is total memory traffic. When I is below the hardware's compute-to-bandwidth ratio $R = \Pi/B$, the operation is memory-bound. For the H100 with $\Pi = 989 \times 10^{12}$ FLOP/s and $B = 3 \times 10^{12}$ bytes/s, $R \approx 330$ FLOPs/byte. At GPT-4 scale with negligible initial KV cache:

$$I = \frac{4.4 \times 10^{11}}{2.2 \times 10^{11}} = 2.0 \text{ FLOPs/byte}$$

This is two orders of magnitude below the roofline ridge. The operation enters the compute-bound regime only when batch size exceeds approximately $R/I \approx 165$ concurrent requests, which itself consumes commensurately more memory for KV cache storage.

The KV cache imposes a separate memory constraint that scales linearly with sequence length. For a model with hidden dimension d and L layers, the cache per sequence of length S is:

$$M_{\text{kv}} = 4LdS$$

in bytes, for key and value tensors at FP16. For GPT-4-class architectures with $d \approx 16384$ and $L \approx 120$:

$$M_{\text{kv}} = 4 \times 120 \times 16384 \times S \approx 7.9 \times 10^6 \times S \text{ bytes}$$

A 128,000-token context window requires approximately 1 GB of KV cache per request. At one million tokens, this rises to roughly 8 GB per request. In a production system serving hundreds of concurrent long-context requests, the aggregate KV cache can consume terabytes of HBM, requiring additional GPUs solely for cache storage. These GPUs contribute to hardware cost without increasing throughput.

Several architectural innovations mitigate but do not eliminate this bottleneck. Grouped Query Attention (GQA) shares key and value heads across query groups: for a model with h total heads and g query groups, the KV cache is reduced by a factor of h/g. LLaMA 2 and PaLM 2 employ GQA with $g = 8$, producing an 8× reduction [12]. FlashAttention tiles the attention computation through the GPU SRAM hierarchy, avoiding materialization of the full $S \times S$ attention matrix in HBM and producing 2–4× wall-clock speedups [13]. PagedAttention, implemented in the vLLM serving system, manages the KV cache in fixed-size blocks analogous to virtual memory paging, eliminating fragmentation and increasing effective batch sizes by 2–4× [14]. Despite these optimizations, the underlying scaling trend is unchanged: the KV cache requirement grows as $O(LdSB)$ and cannot be reduced by more than a constant factor without changing model architecture.

Speculative decoding offers a complementary latency-mitigation technique orthogonal to memory optimization. A small draft model generates candidate tokens at low cost, and the large target model

verifies them in parallel. The expected speedup is approximately $1/(1-\beta+\beta\gamma)$ where β is the fraction of steps using the draft and γ is the token acceptance rate [15]. Empirical acceptance rates of 0.7–0.9 using 1–7 billion parameter draft models yield 2–3× latency improvements on GPT-4-class targets without reducing total FLOPs.

The total cost per request can be expressed as:

$$C = \frac{n_{\text{gpu}}\, c_{\text{gpu}}\, t_{\text{gen}}}{3600}$$

For a single-user request generating 1,000 tokens at the memory-bound rate of 14 tokens per second, $t_{\text{gen}} \approx 71$ seconds. A production configuration that holds active weights and KV cache for a batch of concurrent requests requires 16 H100 GPUs. At approximately $3 per GPU-hour:

$$C = \frac{16 \times 3 \times 71}{3600} \approx \$0.95 \text{ per request}$$

Networking overhead, tensor parallelism communication costs, and scheduler inefficiencies each add 20–50 percent in production deployments. Tensor parallelism requires all-reduce operations across GPUs for each transformer layer: the communication volume per all-reduce is $4d$ bytes per layer, and the total communication time across p GPUs is approximately $4Ld/B_{\text{nvlink}}$ where $B_{\text{nvlink}} \approx 900$ GB/s for H100 NVLink.

Power consumption adds a further cost. Each H100 GPU has a TDP of 700 W, and facility PUE factors of 1.3–1.6 are standard, yielding 910–1120 W per GPU including cooling [10]. For the 16-GPU configuration:

$$P = 16 \times 0.7 \times 1.4 \approx 15.7 \text{ kW}$$

At $0.08 per kWh, the energy cost is approximately $1.26 per hour, adding roughly $0.09 per request.

The economics of pre-training impose a further structural constraint. A Chinchilla-optimal training run for a 1.8-trillion-parameter model requires approximately 2×10^{25} FLOPs. At 30 percent utilization on H100 GPUs (989 TFLOPS peak):

$$t_{\text{train}} = \frac{2 \times 10^{25}}{0.3 \times 9.89 \times 10^{14}} \approx 6.7 \times 10^{10} \text{ GPU-seconds}$$

or approximately 2,100 H100-years. At $3 per GPU-hour, the compute cost exceeds $55 million. Including data acquisition, curation, networking, storage, and the cost of failed runs, the total exceeds $100 million. This capital barrier creates a supply constraint that the market for domain-specific models does not face.

The central implication for deployment decisions is that inference costs are dominated by the fixed overhead of loading large weight matrices from memory, which does not vary with query complexity. A 1.8-trillion-parameter model costs approximately the same per token for a trivial query as for a complex reasoning task, because the dominant cost is the memory transfer of parameters rather than the computation required for inference. Domain-specialized models circumvent this constraint: a 7-billion-parameter model requires $t_{\text{mem}} \approx 0.002$ seconds per token, approximately 35 times

faster, and can run on a single consumer GPU. The inference cost advantage of specialization is multiplicative with the accuracy advantages documented in Section 3.

Production deployment data confirm this analysis. The engineering team at Foresight Data reported spending $90,000 on AWS compute over two weeks while evaluating a generative AI approach to company name matching, estimating that a full production implementation would cost approximately $3 million annually for a task their existing deterministic pipeline already performed reliably [16]. The engineering team at Vera removed all OpenAI dependencies from their internal stack in 2024, citing scaling limitations, latency variability, absence of version control, and cost. After migrating to open-weight models deployed on their own infrastructure, internal costs decreased by over 100 percent [17].

A further distortion in the market for generalist models arises from the pricing strategies of major API providers. The Stanford AI Index documented that the inference cost for GPT-3.5-equivalent queries fell from approximately $20 per million tokens in November 2022 to $0.07 per million tokens by October 2024, a reduction factor of approximately 280 [18]. This price trajectory reflects intentional subsidy rather than underlying cost improvement. OpenAI's operational costs, including estimated daily expenditures of $700,000 for ChatGPT in 2023, substantially exceed API revenue at these price levels. A simple marginal-cost pricing model at $3 per GPU-hour and 14 tokens per second per H100 yields an inference cost of approximately $0.06 per million tokens for compute alone, which is close to the API price, but this excludes amortized training cost, networking, GPU idle time, and the overhead of serving infrastructure, each of which multiplies the true economic cost. Several firms that initially pursued generalist model development have abandoned the approach. Aleph Alpha ceased model development in September 2024, with CEO Jonas Andrulis stating that maintaining a sovereign European LLM did not constitute a viable business model and that required investment could not be justified [19]. Two of the six major Chinese AI unicorns have scaled back pre-training to redirect resources toward application development [20]. Microsoft has been integrating smaller proprietary models including Phi-4 into Microsoft 365 Copilot, and Gartner reported that most enterprise adopters of 365 Copilot have not progressed beyond the pilot phase [21]. These indicators suggest that current generalist API pricing does not reflect long-run equilibrium costs.

Empirical Evidence for Domain Specialization

Comparative evaluations across multiple domains show that models fine-tuned or trained on domain-specific data consistently outperform generalist models on in-distribution benchmarks, often by substantial margins while requiring fewer computational resources.

In medicine, Google's Med-PaLM 2, developed through fine-tuning of PaLM 2 on medical examination data, achieved 86.5 percent on MedQA (USMLE). This represented an improvement of more than 19 percentage points over the generalist PaLM 2. In pairwise evaluations by physicians, Med-PaLM 2 responses were preferred over physician-generated answers on eight of nine clinical utility axes [22]. The subsequent Med-Gemini model achieved 91.1 percent on the same benchmark through an uncertainty-guided search strategy and outperformed GPT-4V on multimodal medical benchmarks by an average relative margin of 44.5 percent [23]. The progression illustrates the importance of domain-specific inference techniques: Med-Gemini's uncertainty-guided search generates multiple candidate answers and selects based on model confidence, producing monotonic accuracy improvements proportional to search width. This method is effective within the bounded scope of medical questions but impractical for general-purpose deployment, where confidence calibration across all possible

Table 1: Comparative performance of generalist and domain-specialized models across selected benchmarks

Domain	Generalist Model	Specialized Model	Advantage
Medicine (MedQA)	GPT-4: 81.4%	Med-Gemini: 91.1%	+9.7 pp
Medicine (multimodal)	GPT-4V: baseline	Med-Gemini: +44.5%	44.5% rel.
Finance (public)	GPT-NeoX: 51.90	BloombergGPT: 62.51	+10.6 pts
Finance (sentiment)	GPT-NeoX: 29.23 F1	BloombergGPT: 62.47 F1	+33.2 F1
Software (supply chain)	GPT-4-turbo: 84.15%	Phi-3 mini: 95.86%	+11.7 pp
Software (cost/req)	GPT-4-turbo: 42c	Phi-3 mini: 0.19c	223×
Protein folding	RFdiffusion	AlphaFold2: 92.4 GDT	+substantial

queries is infeasible. GPT-4 with the Medprompt prompting strategy reached 90.2 percent on MedQA, but this required a domain-specific orchestration layer of few-shot exemplars and chain-of-thought prompts that must be designed, validated, and maintained for the medical domain [24]. Both approaches confirm that domain-specific intervention is necessary.

In finance, BloombergGPT, a 50-billion-parameter model trained on 363 billion tokens of financial text (SEC filings, earnings transcripts, EDGAR filings, financial news) supplemented with 345 billion tokens of general-domain text from The Pile, outperformed public baseline models by 10.6 points on aggregated financial benchmarks. On internal sentiment analysis tasks, the advantage widened to 33.2 points in weighted F1 score [25]. The training methodology reveals the data economics of specialization: the total training budget was approximately 1.3×10^{22} FLOPs (roughly 2,000 A100 GPU-days), whereas a generalist model of comparable size would require an order of magnitude more compute with no guarantee of equivalent financial task performance. The domain-specific data, though smaller in total volume, is more densely informative for the target distribution, yielding higher marginal benefit per training token.

In software engineering, Microsoft researchers evaluated Phi-3 mini, a 3.8-billion-parameter model, on cloud supply chain fulfillment tasks. After fine-tuning on one thousand examples per task, Phi-3 mini achieved 95.86 percent accuracy compared with 84.15 percent for GPT-4-turbo. The inference cost per request was 0.19 cents for Phi-3 mini compared with 42.32 cents for GPT-4-turbo [26]. The legal domain presents a contrasting case: the hallucination rate of 58 percent observed in GPT-4 on legal queries indicates that domain-specific training remains necessary where factual accuracy is a professional requirement [27]. The learning efficiency advantage is substantial: a model fine-tuned on a few thousand domain-specific examples can match or exceed the accuracy of a generalist model trained on trillions of tokens, because the fine-tuning distribution is narrower and the task definition is more constrained.

The principle of domain specialization extends beyond language to other modalities. DeepMind's AlphaFold2 predicted protein structures from amino acid sequences at accuracy competitive with experimental methods, trained exclusively on the Protein Data Bank of approximately 170,000 structures [28]. Meta's ESM-2, a protein language model trained on 250 million sequences using a masked language modeling objective, achieved comparable structure prediction accuracy on several benchmarks through an architecture with rotated attention biases optimized for biological sequence data [29]. These examples demonstrate that domain specialization is often the only viable path to acceptable performance in domains with high-dimensional structured output spaces.

Table 1 summarizes these comparisons. The consistent pattern across all domains is that specialization

improves accuracy while reducing computational cost.

Infrastructure for Specialization with Open Weights, Local Deployment, and Community Distribution

The feasibility of domain specialization depends on infrastructure for distributing, fine-tuning, and deploying models outside the API paradigm. The Hugging Face Hub has become the primary platform, hosting over 2 million public model checkpoints as of late 2025, spanning base architectures, domain-fine-tuned variants, and quantized deployments. The platform provides standardized model cards, dataset integration, and community benchmarking. Libraries such as PEFT, Transformers, and Unsloth provide the software stack for parameter-efficient fine-tuning [30].

The dominant method for efficient fine-tuning is Low-Rank Adaptation (LoRA), which constrains the weight update to a low-rank decomposition. For a pre-trained weight matrix $W_0 \in \mathbb{R}^{d \times k}$, LoRA parameterizes the update as:

$$W = W_0 + BA$$

where $B \in \mathbb{R}^{d \times r}$, $A \in \mathbb{R}^{r \times k}$, and $r \ll \min(d, k)$. During fine-tuning, W_0 is frozen and only A and B are trained. The rank r controls the expressiveness of the adaptation: for $r = 8$ and $d = 4096$, the number of trainable parameters is $2dr = 65,536$ per layer, compared with $d^2 \approx 16.8$ million for full fine-tuning, a reduction factor of approximately 256 [31]. LoRA can be applied to any linear projection in the transformer (query, key, value, output projections, and feed-forward layers) independently, giving practitioners fine-grained control over which components of the model are adapted.

QLoRA extends this approach by quantizing the frozen base model to 4-bit precision using the NormalFloat4 (NF4) data type, which optimally distributes quantization levels according to a zero-mean normal distribution of weights [32]. The quantized weights are stored as 4-bit integers and dequantized on the fly during the forward pass. The memory requirement for a 7-billion-parameter model drops from approximately 14 GB (FP16) to approximately 4 GB (NF4), enabling fine-tuning on a single consumer GPU with 8–12 GB of VRAM. The combination of LoRA with NF4 quantization enables fine-tuning of models up to 65 billion parameters on a single workstation GPU, reducing the capital barrier to entry from millions of dollars in cloud compute to the cost of consumer hardware.

The viability of local deployment has increased substantially through weight-only quantization techniques that reduce model precision at inference time. GPTQ uses optimal brain quantization, a second-order method that approximates the optimal rounding of weights by minimizing the squared error relative to a calibration dataset [33]. AWQ (Activation-aware Weight Quantization) identifies a small fraction of weights that are disproportionately important for model quality by analyzing activation magnitudes and protects them during quantization [34]. GGUF, the format underlying llama.cpp, supports mixed-precision quantization where different layers are quantized to different bit widths based on their sensitivity to quantization error. The practical result is that a 70-billion-parameter model quantized to 4-bit weights requires approximately 35 GB of memory, fitting on a single NVIDIA RTX 6000 Ada or Apple M2 Ultra with 192 GB of unified memory.

Local deployment addresses a limitation of the generalist API model that is frequently overlooked. Applications in healthcare, legal services, finance, and defense process data that cannot be transmitted

to external API endpoints under regulatory frameworks including HIPAA, GDPR, and attorney-client privilege rules. A locally deployable specialized model resolves this constraint without requiring the compromises associated with on-premise cloud deployments. The economic motivation is equally significant: a fine-tuned 7-billion-parameter model running on local hardware incurs no per-token cost after the initial training investment.

Architectural and Market Convergence on Specialization

The architectural direction of frontier models indicates that even the laboratories responsible for the generalist paradigm are adopting specialization as a design principle. Mixture-of-Experts architectures have become the dominant configuration for state-of-the-art systems, including DeepSeek V3, Gemini 1.5, Mixtral 8x7B, and Llama 4. The MoE formulation replaces each feed-forward layer with a set of E expert networks $E_1, \ldots, E_E$ and a gating function $G : \mathbb{R}^d \to \mathbb{R}^E$ that routes each input token to the top-k experts:

$$y = \sum_{i=1}^{E} G(x)_i \cdot E_i(x), \quad G(x) = \text{TopK}(\text{softmax}(W_g x), k)$$

where $W_g \in \mathbb{R}^{E \times d}$ is the gating weight matrix and TopK retains only the k largest logits, zeroing the rest [35]. The sparsity factor k/E determines the computational cost: for Mixtral 8x7B with $E = 8$ and $k = 2$, only 25 percent of expert parameters are activated per token, yet the model has access to the full capacity of all 8 experts through the routing mechanism. An auxiliary load-balancing loss is added during training to prevent token collapse, where the gating network routes most tokens to a small subset of experts:

$$\mathcal{L}_{\text{bal}} = \alpha \cdot E \sum_{i=1}^{E} f_i \cdot P_i$$

where f_i is the fraction of tokens routed to expert i and P_i is the average gate probability assigned to expert i [36]. The coefficient α controls the trade-off between routing specialization and load balance.

Controlled empirical comparisons confirm the efficiency advantage of MoE over dense architectures. A study at Apple across model scales from 6.4 to 29.6 billion parameters found that MoE configurations formed a consistently higher speed-accuracy frontier than dense alternatives [37]. A separate EMNLP 2024 study reported that MoE achieves approximately 16.4 percent greater data utilization efficiency under fixed compute budgets [38]. The theoretical foundation for this advantage is that token-dependent routing provides a form of conditional computation: each input is processed by a subset of parameters specialized for its features, rather than by the same dense transformation applied uniformly. This is specialization at the parameter level.

The same principle of conditional computation applies at the system level through retrieval-augmented generation. RAG decomposes the generation task into retrieval and synthesis: for an input query q, a retriever selects the top-k documents from a corpus $\mathcal{D}$ according to a relevance score $s(q, d_i)$, and the language model generates an output conditioned on both the query and the retrieved documents:

$$P(y \mid q) = \sum_{d \in \mathrm{TopK}(s(q,\mathcal{D}))} P_{\mathrm{LM}}(y \mid q, d) \cdot P_{\mathrm{ret}}(d \mid q)$$

This architecture externalizes knowledge storage to a structured corpus while the language model provides linguistic competence [39]. The effect is that the system can be specialized to a domain by replacing the retrieval corpus rather than retraining the model. Domain-specific RAG systems in legal, medical, and technical support contexts achieve higher factual accuracy than unaided generalist models because the retrieval step constrains the model to produce outputs grounded in the retrieved evidence.

Agentic pipelines represent a further layer of system-level specialization. Rather than a single model performing all reasoning steps, multiple specialized models are composed through orchestration layers that route subtasks to appropriate models. Common patterns include the ReAct framework, which interleaves reasoning traces with action execution in a feedback loop, and directed acyclic graph orchestration, where the output of one specialist model becomes the input of another. Andrej Karpathy characterized this trajectory as one where foundation models serve as a substrate for application-layer specialization through context engineering, orchestration, and task-specific fine-tuning [40]. The architectural pattern mirrors the economic logic of specialization at the system level.

Market evidence supports this trajectory. Salespeak trained a 14-billion-parameter model using QLoRA on a single A10G GPU at a total compute cost of approximately $25. The resulting model performs comparably to GPT-4 on real-time sales dialogue while executing 37 percent faster [41]. Intercom's Fin Apex processes over one million conversations per week with a custom support model, replacing a generalist API at substantially reduced cost. Y Combinator's Summer 2024 cohort analysis indicated that over 75 percent of funded AI startups were building vertical, task-specific applications rather than general-purpose models, and the accelerator has argued that vertical AI agents represent a market opportunity an order of magnitude larger than the SaaS model [42, 43].

Perspectives on the Generalist-Specialist Trajectory

Researchers and practitioners with direct experience across multiple phases of AI development have expressed converging views on the limitations of the generalist paradigm. Their assessments draw on operational rather than theoretical experience.

Andrej Karpathy described current LLMs in a 2025 Y Combinator talk as systems with anterograde amnesia: they cannot consolidate knowledge across sessions and require explicit programming of working memory for each interaction. He characterized the trajectory as one in which foundation models serve as a substrate for application-layer specialization rather than as complete solutions [44]. In his annual LLM review, he predicted that durable competitive advantage would accrue to organizations that orchestrate teams of fine-tuned models for specific verticals, not to those that optimize a single generalist system [45].

Yann LeCun has been more direct. In a lecture at IIT Madras, he advised students that work on LLMs would not lead to human-level intelligence and that the field required fundamentally different approaches [46]. He enumerated four capabilities current models lack: understanding of the physical world, persistent memory, reasoning in any meaningful sense, and hierarchical planning, each present

in domestic cats [47]. While directed at the longer-term question of human-level intelligence, these observations apply directly to domain-specific deployment, where these capabilities are often required.

François Chollet distinguished between the memorization of reasoning patterns that powers LLM performance and the adaptive program synthesis that genuine reasoning requires [48]. His ARC-AGI benchmark was explicitly designed to measure the latter by presenting novel visual reasoning problems that cannot be solved through pattern matching against training data. The persistent gap between human and machine performance on ARC (humans achieve 85 percent, while the best LLM-based systems score below 40 percent) quantifies the distance between skill acquisition and general intelligence. Gary Marcus has argued that the structural inability of transformer architectures to perform abstract symbol manipulation represents a fundamental constraint that scaling alone cannot resolve [6, 4]. Bender et al. characterized LLMs as stochastic parrots that lack a coherent model of the world and whose apparent competence is a reflection of the statistical regularities in their training distribution [49]. Each of these critiques converges on the same conclusion: the competence of generalist models is bounded by their training distribution, and reliable performance outside that distribution requires domain-specific intervention.

Conclusion

The generalist large language model has been a productive research direction, demonstrating that neural language models can achieve broad competence across diverse tasks. But the evidence reviewed here indicates that the generalist approach is converging on structural limitations that specialization addresses directly.

The scaling laws that motivated the paradigm face data exhaustion, diminishing returns on saturated benchmarks, and expressiveness bounds inherent to the transformer architecture. A roofline analysis of inference costs reveals that autoregressive decode is memory-bandwidth-bound by two orders of magnitude for frontier models, making per-token cost independent of query complexity and creating a structural economic advantage for smaller specialized models. Domain-adapted models achieve higher accuracy on their target tasks at inference costs lower by factors of 35–200, confirmed across medicine, finance, software engineering, law, and protein folding. Parameter-efficient fine-tuning with LoRA and QLoRA reduces the capital barrier for domain adaptation to consumer hardware costs, while quantization techniques enable local deployment of models up to 70 billion parameters on single workstations.

The architectural direction of the field reflects a recognition that routing inputs to specialized computation is more efficient than dense processing. MoE routing at the parameter level, retrieval augmentation at the system level, and agentic pipelines at the deployment level all instantiate the same principle: conditional computation, where the subset of parameters or systems activated is chosen based on the input, is more efficient than processing all inputs through the same transformation.

The implication for practitioners is that procurement decisions should be based on task-specific evaluation rather than general benchmark scores, that total cost of ownership for API-based deployment should account for the likelihood of price corrections as venture capital subsidies are withdrawn, and that local deployment of fine-tuned models should be considered wherever regulatory compliance, latency requirements, or cost predictability are priorities. The future of applied language technology belongs not to a single generalist system but to a distributed ecosystem of specialized models, each optimized for the contexts in which it operates.

References

[1] François Chollet. *On the Measure of Intelligence*. 2019. arXiv: 1911.01547 [cs.AI]. URL: https://arxiv.org/abs/1911.01547 (visited on 05/27/2026).

[2] Jared Kaplan et al. *Scaling Laws for Neural Language Models*. Established power-law scaling of loss with parameters and data. 2020. arXiv: 2001.08361 [cs.LG].

[3] Jordan Hoffmann et al. *Training Compute-Optimal Large Language Models*. Chinchilla scaling laws: parameters and tokens should scale equally with compute budget. 2022. arXiv: 2203.15556 [cs.CL].

[4] Gary Marcus. *Evidence That LLMs Are Reaching a Point of Diminishing Returns*. Substack. 2024. URL: https://garymarcus.substack.com/p/evidence-that-llms-are-reaching-a (visited on 05/27/2026).

[5] Mehrdad Farajtabar et al. *GSM-NoOp: No Evidence of Formal Reasoning in Language Models*. Apple ML Research study showing pattern-matching rather than reasoning in math problems. 2024. arXiv: 2410.05229 [cs.CL]. URL: https://arxiv.org/abs/2410.05229 (visited on 05/27/2026).

[6] Gary Marcus. *LLMs Don't Do Formal Reasoning — And That Is a HUGE Problem*. Substack. 2024. URL: https://garymarcus.substack.com/p/llms-dont-do-formal-reasoning-and (visited on 05/27/2026).

[7] William Merrill and Ashish Sabharwal. *The Parallelism Tradeoff: Limitations of Log-Precision Transformers*. Demonstrates log-precision transformers lie within complexity class TC0. 2023. arXiv: 2207.00729 [cs.CL].

[8] Pablo Villalobos et al. *Will We Run Out of Data? Limits of LLM Scaling Based on Human-Generated Text*. Projects public text data exhaustion by 2026–2032. 2024. arXiv: 2211.04325 [cs.CL]. URL: https://arxiv.org/abs/2211.04325 (visited on 05/27/2026).

[9] Reiner Pope. *Batch Size Economics: Why AI Costs 1,000X More*. LinkedIn (via Gauhar Junnarkar). 2026. URL: https://www.linkedin.com/posts/gauharjunnarkar_aiinfrastructure-llminference-machinelearning-activity-7458272395808854017-rTP- (visited on 05/27/2026).

[10] NVIDIA Corporation. *NVIDIA H100 Tensor Core GPU Architecture*. Tech. rep. H100 specifications: 989 TFLOPS FP8, 80 GB HBM3, 3 TB/s bandwidth, 700 W TDP. 2024. URL: https://resources.nvidia.com/en-us-tensor-core-gpu-datasheet (visited on 05/27/2026).

[11] Dylan Patel and Afzal Ahmad. *The Secret to GPT-4's Success: 1.8 Trillion Parameters, 220 Billion Active*. Analysis of GPT-4 MoE architecture estimating 1.8T total and 220B active parameters. SemiAnalysis. 2024. URL: https://www.semianalysis.com/p/gpt-4-architecture-infrastructure (visited on 05/27/2026).

[12] Joshua Ainslie et al. *GQA: Training Generalized Multi-Query Transformer Models from Multi-Head Checkpoints*. Grouped Query Attention reduces KV cache by factor of h/g. 2023. arXiv: 2305.13245 [cs.LG].

[13] Tri Dao et al. *FlashAttention: Fast and Memory-Efficient Exact Attention with IO-Awareness*. Tiled attention computation avoiding full SxS attention matrix in HBM. 2022. arXiv: 2205.14135 [cs.LG].

[14] Woosuk Kwon et al. *Efficient Memory Management for Large Language Model Serving with PagedAttention*. PagedAttention for KV cache management in vLLM serving system. 2023. arXiv: 2309.06180 [cs.LG].

[15] Yaniv Leviathan, Matan Kalman, and Yossi Matias. *Fast Inference from Transformers via Speculative Decoding*. Draft-then-verify framework for 2-3x latency improvement. 2023. arXiv: 2211.17192 [cs.LG].

[16] Foresight Data Inc. *We Spent $100k on AWS in 2 Weeks Playing with Gen AI*. LinkedIn. 2024. URL: https://www.linkedin.com/pulse/we-spent-100k-aws-2-weeks-playing-gen-ai-foresight-data-inc-cqdle (visited on 05/27/2026).

[17] Vera Engineering Blog. *Goodbye, OpenAI*. Medium. 2024. URL: https://medium.com/@askverablog/goodbye-openai-b4dcaebb9840 (visited on 05/27/2026).

[18] Stanford University Human-Centered AI (HAI). *AI Index Report 2025*. Documents inference cost drop for GPT-3.5-class models from $20 to $0.07 per million tokens (Nov 2022–Oct 2024). 2025. URL: https://hai.stanford.edu/ai-index/2025 (visited on 05/27/2026).

[19] Kim M. Scheurenbrand. *Aleph Alpha Quits AI Model Race*. The Decoder. 2024. URL: https://the-decoder.com/aleph-alpha-quits-ai-model-race/ (visited on 05/27/2026).

[20] 36Kr English. *Two of China's "AI Tigers" Abandon Pre-Trained Models to Pursue App-Driven Growth*. 2024. URL: https://kr-asia.com/two-of-chinas-ai-tigers-abandon-pre-trained-models-to-pursue-app-driven-growth (visited on 05/27/2026).

[21] BW Businessworld. *Microsoft Reduces Reliance On OpenAI For 365 Copilot Amid Cost, Speed Concerns*. 2024. URL: https://www.businessworld.in/article/microsoft-reduces-reliance-on-openai-for-365-copilot-amid-cost-speed-concerns-542965 (visited on 05/27/2026).

[22] Karan Singhal et al. "Towards Physician-Level Medical Question Answering with Large Language Models". In: *Nature Medicine* 30 (2023). Med-PaLM 2 achieves 86.5% on USMLE MedQA dataset, pp. 2503–2513. URL: https://www.nature.com/articles/s41591-024-03423-7 (visited on 05/27/2026).

[23] Khaled Saab et al. *Capabilities of Gemini Models in Medicine*. Med-Gemini achieves 91.1% on MedQA, outperforms GPT-4V by 44.5% on multimodal medical benchmarks. 2024. arXiv: 2404.18416 [cs.CL]. URL: https://arxiv.org/abs/2404.18416 (visited on 05/27/2026).

[24] Harsha Nori et al. *Can Generalist Foundation Models Outcompete Special-Purpose Tuning? Case Study in Medicine*. GPT-4 with Medprompt reaches 90.2% on MedQA USMLE benchmark. 2023. arXiv: 2311.16452 [cs.CL]. URL: https://arxiv.org/abs/2311.16452 (visited on 05/27/2026).

[25] Shijie Wu et al. *BloombergGPT: A Large Language Model for Finance*. Domain-specific finance model outperforms generalists by 10+ points on financial benchmarks. 2023. arXiv: 2303.17564 [cs.LG]. URL: https://arxiv.org/abs/2303.17564 (visited on 05/27/2026).

[26] Jinesh Doshi et al. *Small Language Models for Application Interactions: A Case Study*. Phi-3 mini (3.8B) beats GPT-4-turbo on supply chain task at less than 1/200th cost. 2024. arXiv: 2405.20347 [cs.CL]. URL: https://arxiv.org/abs/2405.20347 (visited on 05/27/2026).

[27] Matthew Dahl et al. *Large Legal Fictions: Profiling Legal Hallucinations in Large Language Models*. Published in Stanford Journal of Law & AI. Found LLMs hallucinate 58–88% on legal case queries across GPT-4, PaLM 2, and Llama 2. 2024. URL: https://dho.stanford.edu/wp-content/uploads/Hallucinations_JLA.pdf (visited on 05/27/2026).

[28] John Jumper et al. "Highly Accurate Protein Structure Prediction with AlphaFold". In: *Nature* 596 (2021). AlphaFold2 achieves experimental-competitive accuracy on CASP14 protein structure prediction, pp. 583–589. URL: https://www.nature.com/articles/s41586-021-03819-2 (visited on 05/27/2026).

[29] Zeming Lin et al. "Evolutionary-Scale Prediction of Atomic-Level Protein Structure with a Language Model". In: *Science* 379 (2023). ESM-2 protein language model achieves AlphaFold2-comparable structure prediction, pp. 1123–1130. URL: https://www.science.org/doi/10.1126/science.ade2574 (visited on 05/27/2026).

[30] Lucain Pouget et al. *huggingface_hub v1.0: Five Years of Building the Foundation of Open Machine Learning*. Hugging Face Hub reaches 2M+ public model checkpoints. 2025. URL: https://huggingface.co/blog/huggingface-hub-v1 (visited on 05/27/2026).

[31] Edward J. Hu et al. *LoRA: Low-Rank Adaptation of Large Language Models*. Parameter-efficient fine-tuning via low-rank decomposition delta W = BA. 2021. arXiv: 2106.09685 [cs.CL].

[32] Tim Dettmers et al. *QLoRA: Efficient Finetuning of Quantized Language Models*. NF4 quantization combined with LoRA for 4-bit fine-tuning. 2023. arXiv: 2305.14314 [cs.LG].

[33] Elias Frantar et al. *GPTQ: Accurate Post-Training Quantization for Generative Pre-Trained Transformers*. Optimal brain quantization for LLM weight compression to 3-4 bits. 2022. arXiv: 2210.17323 [cs.LG].

[34] Ji Lin et al. *AWQ: Activation-aware Weight Quantization for LLM Compression and Acceleration*. Activation-aware weight quantization protecting salient 1% of weights. 2024. arXiv: 2306.00978 [cs.LG].

[35] Noam Shazeer et al. *Outrageously Large Neural Networks: The Sparsely-Gated Mixture-of-Experts Layer*. Original MoE formulation with top-k gating and auxiliary load-balancing loss. 2017. arXiv: 1701.06538 [cs.LG].

[36] William Fedus, Barret Zoph, and Noam Shazeer. *Switch Transformers: Scaling to Trillion Parameter Models with Simple and Efficient Sparsity*. Switch Transformer routing: top-1 expert selection with load-balancing regularization. 2022. arXiv: 2101.03961 [cs.LG].

[37] A. Chung et al. *Revisiting MoE and Dense Speed-Accuracy Comparisons for LLM Training*. Apple study finding MoE consistently outperforms dense models across scales. 2024. arXiv: 2405.15052 [cs.LG]. URL: https://arxiv.org/abs/2405.15052 (visited on 05/27/2026).

[38] Tong Zhu et al. "Scaling Laws Across Model Architectures: A Comparative Analysis of Dense and MoE Models in Large Language Models". In: *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. MoE shows 16.37% improvement in data utilization efficiency over dense models. 2024. URL: https://aclanthology.org/2024.emnlp-main.319.pdf (visited on 05/27/2026).

[39] Patrick Lewis et al. *Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks*. RAG framework combining dense retriever with seq2seq generator. 2020. arXiv: 2005.11401 [cs.CL].

[40] Andrej Karpathy. *Ex-Tesla AI Chief Andrej Karpathy Shares Four Tips for AI Startups Competing with OpenAI*. The Decoder. 2025. URL: https://the-decoder.com/ex-tesla-ai-chief-andrej-karpathy-shares-four-tips-for-ai-startups-competing-with-openai/ (visited on 05/27/2026).

[41] Salespeak. *We're Training Our Own LLM. Here's What It Actually Takes*. 2024. URL: https://salespeak.ai/blog/building-our-own-llm-what-it-actually-takes (visited on 05/27/2026).

[42] Y Combinator. *Vertical AI Agents Could Be 10X Bigger Than SaaS*. 2024. URL: https://rosetta.to/u/ycombinator/vertical-ai-agents-could-be-10x-bigger-than-saas (visited on 05/27/2026).

[43] Alex Bozhin. *Y Combinator Summer 2024: AI Surge and Unprecedented Startup Valuations*. LinkedIn. 2024. URL: https://www.linkedin.com/pulse/y-combinator-summer-2024-ai-surge-unprecedented-startup-alex-bozhin-of25e (visited on 05/27/2026).

[44] Andrej Karpathy. *Software Is Changing (Again)*. YC talk describing LLMs as coworkers with anterograde amnesia. Y Combinator. 2025. URL: https://www.youtube.com/watch?v=LCEmiRjPEtQ (visited on 05/27/2026).

[45] Andrej Karpathy. *2025 LLM Year in Review*. 2025. URL: https://karpathy.bearblog.dev/year-in-review-2025/ (visited on 05/27/2026).

[46] Yann LeCun. *Do Not Work on LLMs If You Are Interested in Human-Level Intelligence: Meta Chief AI Scientist Yann LeCun*. Times of India. 2024. URL: https://timesofindia.indiatimes.com/city/chennai/do-not-work-on-llms-if-you-are-interested-in-human-level-intelligence-meta-chief-ai-scientist-yann-lecun/articleshow/114475059.cms (visited on 05/27/2026).

[47] Yann LeCun. *Meta's AI Chief Yann LeCun on AGI, Open-Source, and AI Risk*. TIME. 2024. URL: https://time.com/6694432/yann-lecun-meta-ai-interview/ (visited on 05/27/2026).

[48] François Chollet and Mike Knoop. *LLMs Won't Lead to AGI*. Dwarkesh Podcast. 2024. URL: https://www.dwarkesh.com/p/francois-chollet (visited on 05/27/2026).

[49] Emily M. Bender et al. *On the Dangers of Stochastic Parrots: Can Language Models Be Too Big?* LLMs characterized as stochastic parrots lacking coherent world models. 2021. URL: https://dl.acm.org/doi/10.1145/3442188.3445922 (visited on 05/27/2026).