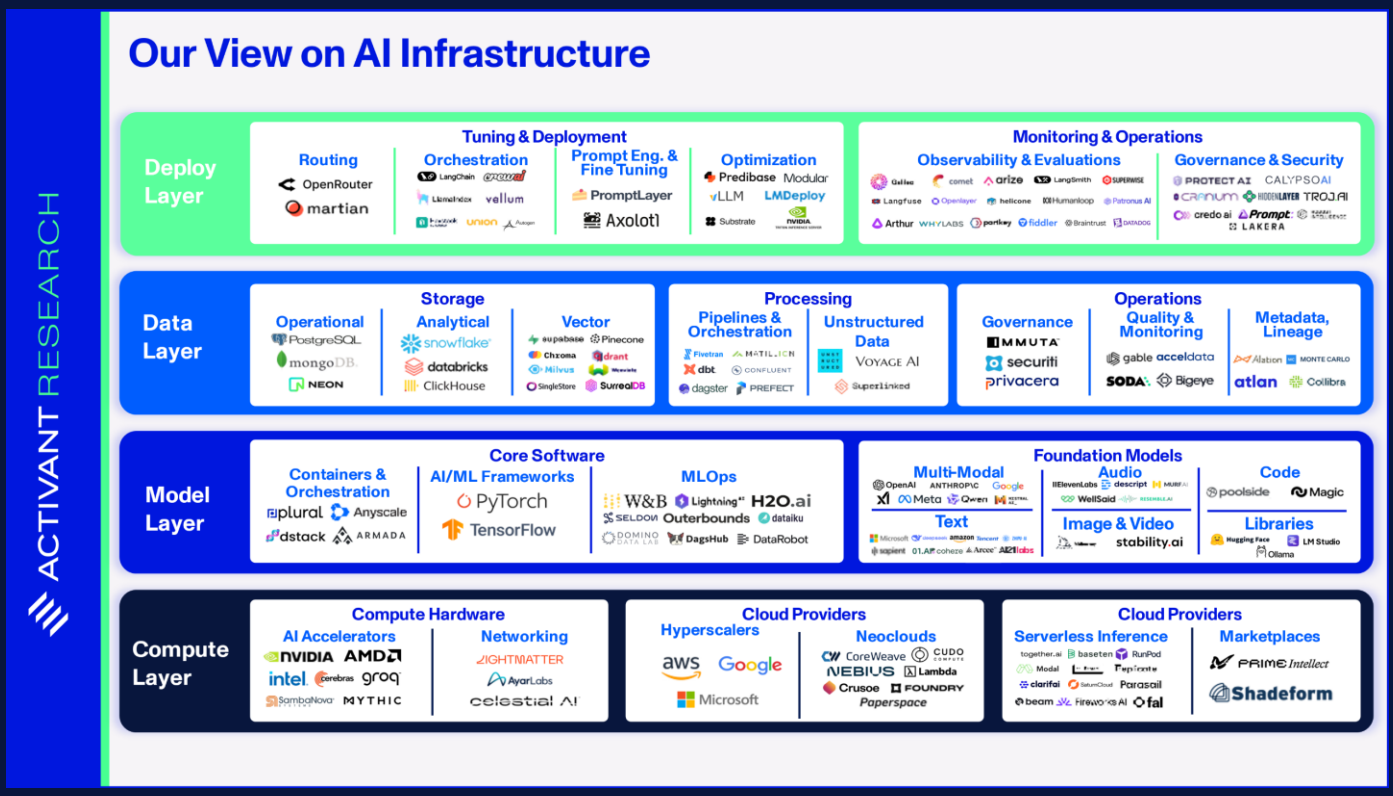
ACTIVANT RESEARCH

AI Infrastructure: Compute (4/4)

The inference opportunity: will software rule the infra layer?



Q1 2026

Jono Vickery

AI Everywhere is the Inference Opportunity

In our previous article, we explored Neoclouds, the bare metal providers capturing multi-billion-dollar contracts for stable, predictable training workloads. But once a model has been trained, it must serve the real world, known as model inference. This article is about the companies making model inference into a simple API call rather than a deep hardware problem.

As AI diffuses across the economy, inference is expected to dwarf training as a share of AI workloads, with some estimates suggesting that it will account for up to 80% of all AI workloads.[1] Where training is about a few companies training a few large models, **inference, by contrast, is about every AI company, feature, chat interaction, and agent action running continuously across the world.** The inference providers have a massive opportunity ahead of them.

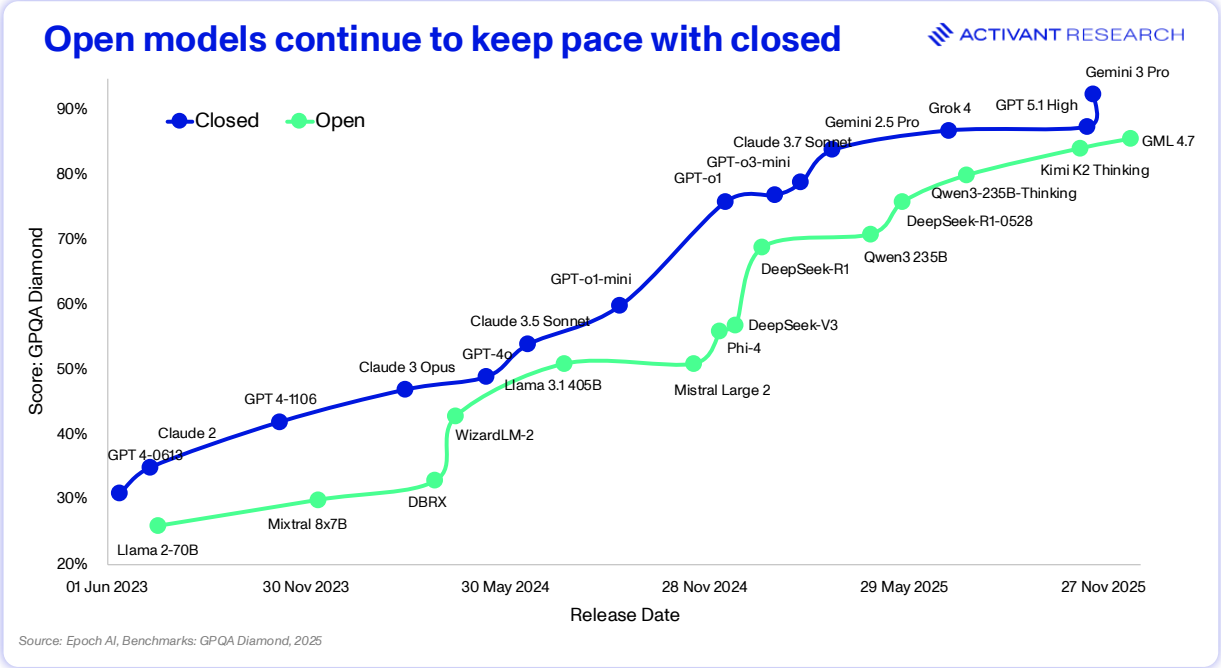
While companies like Google, OpenAI and Anthropic will benefit significantly from this shift, demand for open source models creates space for a new class of providers. Serverless inference providers make it dead simple to run an open source model in production, eliminating painful cold starts and transforming the economic model from pay-per-hour to pay-per-million-tokens. Many see this market as a race to the bottom: reselling commodity hardware in the form of tokens, the price of which continuously fall.

However, a tight technical talent market could mean that software-level optimizations become the moat, allowing some providers to create unmatched cost efficiencies, consolidate market share and lock in developers with advanced, high-value workloads like fine-tuning. **As AI workloads continue to shift from training to inference, the $450bn AI infrastructure opportunity becomes less about access to the best hardware and more about delivering tokens through a delightful developer experience, with leading edge performance and cost leadership.**

The Inference Gap: Slow, Complex Clouds

Getting started with AI models as a software developer is easy. A first response from the OpenAI API requires just six lines of code.[2] While moving from this “hello world” example to full-featured apps does of course result in more complexity, the point is that the **closed models** (OpenAI, Anthropic, xAI) offer a way for developers to build AI without thinking about infrastructure at all. They abstract away complexity.

But the future of AI is increasingly open, and increasingly complex as we highlighted in our research on Open Source Generative AI. We’re seeing AI deployments make greater use of open models, compound AI systems and enterprise-specific fine-tuning. Open source models continue to narrow the performance gap with closed source systems and often offer advantages in cost, speed and fine-tuning capabilities.



For these compound AI deployments, however, developers risk running into significant infrastructure complexity and latency. Running an open source model is not as simple as hitting the download button on Hugging Face. Developers who try to "roll their own" infrastructure on generic cloud providers quickly slam into two formidable walls: **complexity** and the **cold start.**

Complexity, the DevOps Tax

To deploy an open source model on one of the hyperscale clouds, developers need to provision GPU capacity (selecting specific GPU types), attach high throughput storage for the model weights, and package the model runtime (CUDA drivers, PyTorch version) into a docker image. They'll need to think about autoscaling for bursty workloads as well as observability and monitoring.

All those technical buzzwords are likely to send most developers racing back to the six lines of code with OpenAI and that's the point. But it doesn't end there. Driver versions need to be maintained and orchestrated across heterogenous hardware (H100s, A100s, T4s, L4s, etc.) and through the entire chain (CUDA hardware drivers through to Nvidia software kernels).

Even if one can get it all to work, it might still be slow.

The Cold Start Problem

When a cloud resource is not in use, the provider will spin it down to save costs and free up capacity. For a new request, they need to create a new execution environment from scratch, which is the "cold start." Cold starts bite due to the time spent provisioning a container, downloading the code package, starting the model runtime and executing the code. For traditional functions, cold starts measure between 5 and 30 seconds but when loading an LLM (~20GB+ of storage), they can extend beyond 10 minutes.[3,4]

The issue is, of course, that consumers are extremely impatient. It's now famous that Amazon found every 100-ms delay in page load cost their ecommerce site 1% of sales.[5] For a start-up trying to retain users on a new AI-native app, that 10 minutes could spell death.

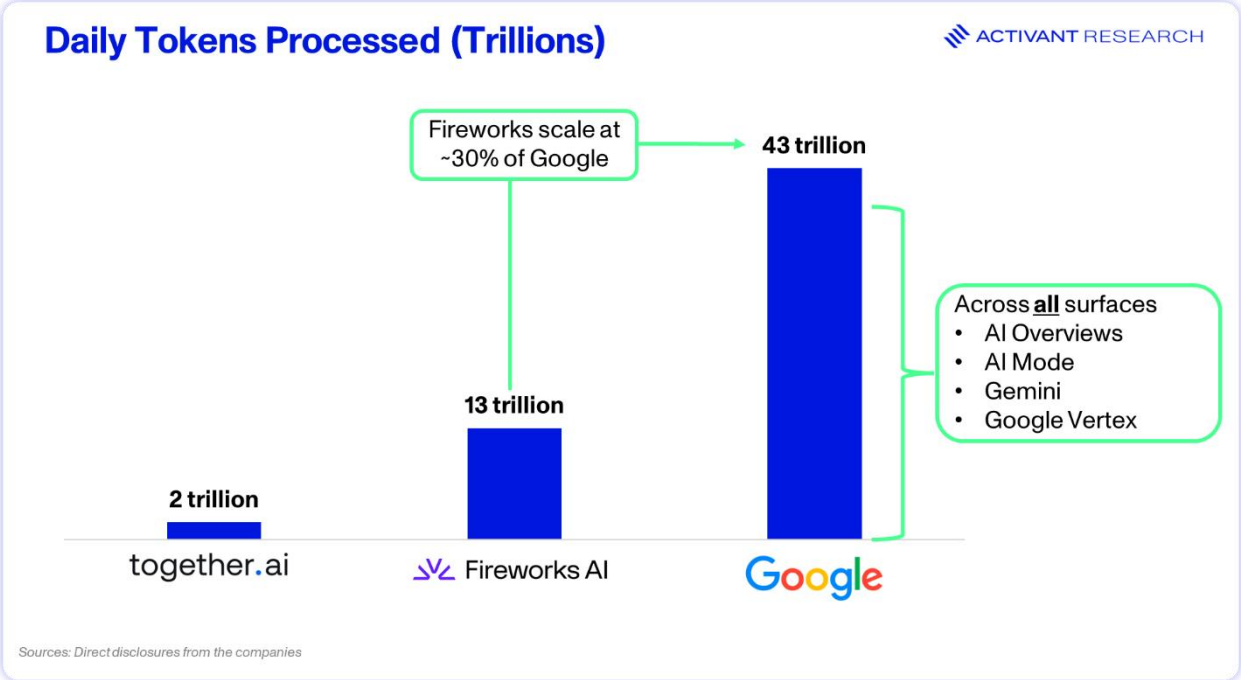
However, if your infrastructure capacity is always, or at least mostly, warm — meaning model weights have been loaded into memory recently — then cold starts are not an issue. The problem for most AI companies is that their workloads are **bursty.** While training represents long-running, steady-state processing, inference workloads fluctuate wildly based on user activity such as spikes in image generators following a new viral trend. AI companies need their infrastructure to scale back down to zero after these spikes to avoid the high cost of idle hardware, which means that they will always face the cold start problem.

The inference gap left a clear playbook for new entrants: provide a delightful developer experience for using open source models and solve the cold start problem.

Enter Serverless Inference

That's exactly what serverless inference did to enter what was otherwise a locked-up cloud market. **They provided leading-edge performance with no cold starts and rewrote the economics of AI compute — charging only per token, automatically scaling from zero to thousands of concurrent users and back again, and removing the need for costly infrastructure management.**

Today, the category is a very real and massive infrastructure layer underpinning the AI ecosystem. Fireworks AI processes 13 trillion tokens per day, about 30% of what Google processes across all their surfaces, including search.[6,7] Fal's specialized media-generation tooling reaches over 1 million developers, and Together AI had already surpassed $100 million in ARR in early 2024.[8,9]



Delighting Developers

OpenAI made it possible for developers to start inferencing their models with just six lines of code. Together AI and Fireworks AI allow you to use **that exact same code** to run more than 200 open source models.[10] These providers adopt the OpenAI API schema, enabling apps built on GPT models to switch to open source simply by changing the "base URL" API endpoint, while keeping the rest of the codebase the same.

In effect, these companies allow developers to run open source LLMs and image generators without provisioning, managing or even thinking about infrastructure. From the developer's perspective, **the experience is serverless**. AI teams never need to navigate GPU scarcity, container orchestration, driver compatibility, or cold start latency again.

But it wasn't just delighting developers that made these tools a must-have, there was also a crucial business model innovation: pricing.

Innovating Pricing

Serverless inference providers charge per token, or per second of compute, not per GPU hour.

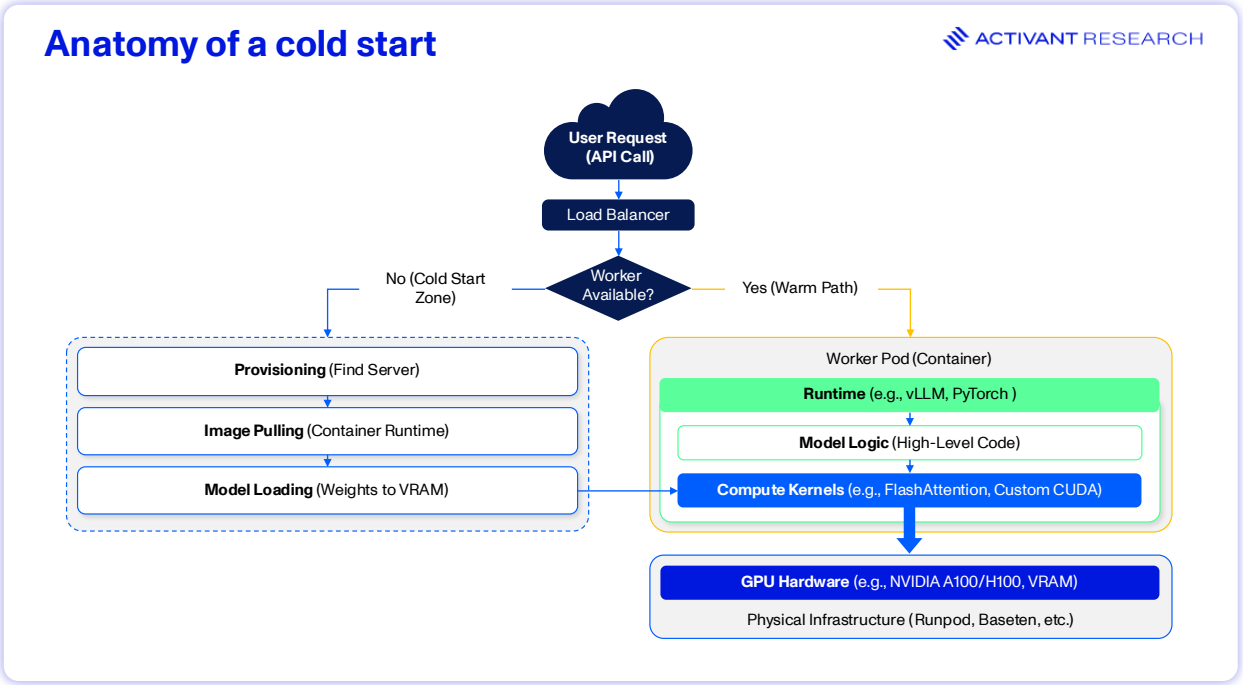
This matters more than it might initially seem. On AWS or GCP, GPU rental is billed hourly. If the first 15 minutes of that hour are consumed downloading model weights for a large open source model, the full 60 minutes is still billed. For companies with unpredictable, bursty inference workloads, this pricing model can become prohibitively expensive.

When infrastructure can scale down to zero during idle periods and only charge for actual usage, the unit economics transform completely. BentoML's scale-to-zero capabilities allowed Neurolabs to reduce compute costs by 70%.[11] Patreon cut ML infrastructure costs by nearly $600,000 annually by leveraging Baseten's autoscaling and rightsizing features.[12]

These companies have also largely eliminated the cold start, further improving both cost efficiency and performance.

Solving Cold Starts and Squeezing Out Performance

As we noted earlier, cold starts are bottlenecked by loading large models into memory for the first time. To solve this, providers need to either keep resources warm for their customers (eroding their margins in the process) or completely rethink the architecture of the infrastructure stack. The former plays a role, but we've also seen immense innovation on the latter in the form of custom runtimes, file systems and kernel optimization.[13,14] The graphic below illustrates this clearly:



For example, Runpod's Flashboot technology achieves 95% of cold starts (P95) in under 2.3 seconds with a combination of **state retention** (retaining a worker that has spun down for a short period, allowing it to be revived rather than booted from scratch); **warm container pooling** (keeping high-demand models active) and **model caching** (loading the model weights on the machine's disk and reducing load time).[15] Beam uses a similar state retention technique called "snapshotting."

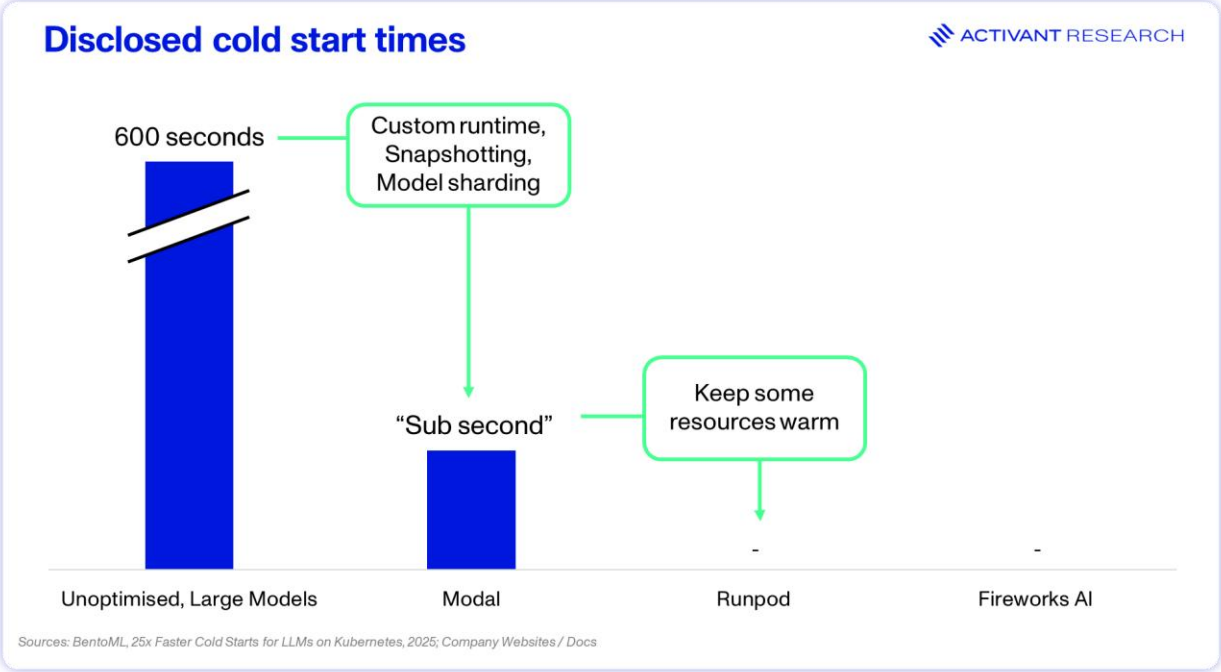
In the above examples, most of the benefit is obtained from a combination of keeping idle resources warm or changing the model loading strategy, such as from a retained state. Modal Labs solved the problem by completely redesigning the runtime.

Custom Runtimes and File Systems

The AI Worker pod or container is typically built using Docker and/or Kubernetes, with Docker commanding over 80% market share in containerization.[16] However, these technologies are general-purpose and not optimized for AI workloads. Docker containers can be bloated (often 5GB – 15GB in size) and load elements sequentially, contributing to slow startup times.

To overcome these limitations, Modal Labs built a proprietary file system and container runtime engine, written in Rust, to replace Docker/Kubernetes. The file system uses content-addressed, lazy loading, which only pulls the required bytes. If a model needs only the first 100MB of a 10GB

file to begin initialization, Modal loads only those 100MB – eliminating the model download phase of a cold start.[17] In addition, the runtime engine also makes use of snapshotting, which reduces cold start times for models such as Mistral 3 from ~118 seconds to ~12 seconds.[18]



Cutting cold starts from ~10 minutes to, in many cases, under one second is the fundamental shift that makes this segment possible. With that in place, optimizing the kernel squeezes out the final inch of performance.

Kernel-level Optimization

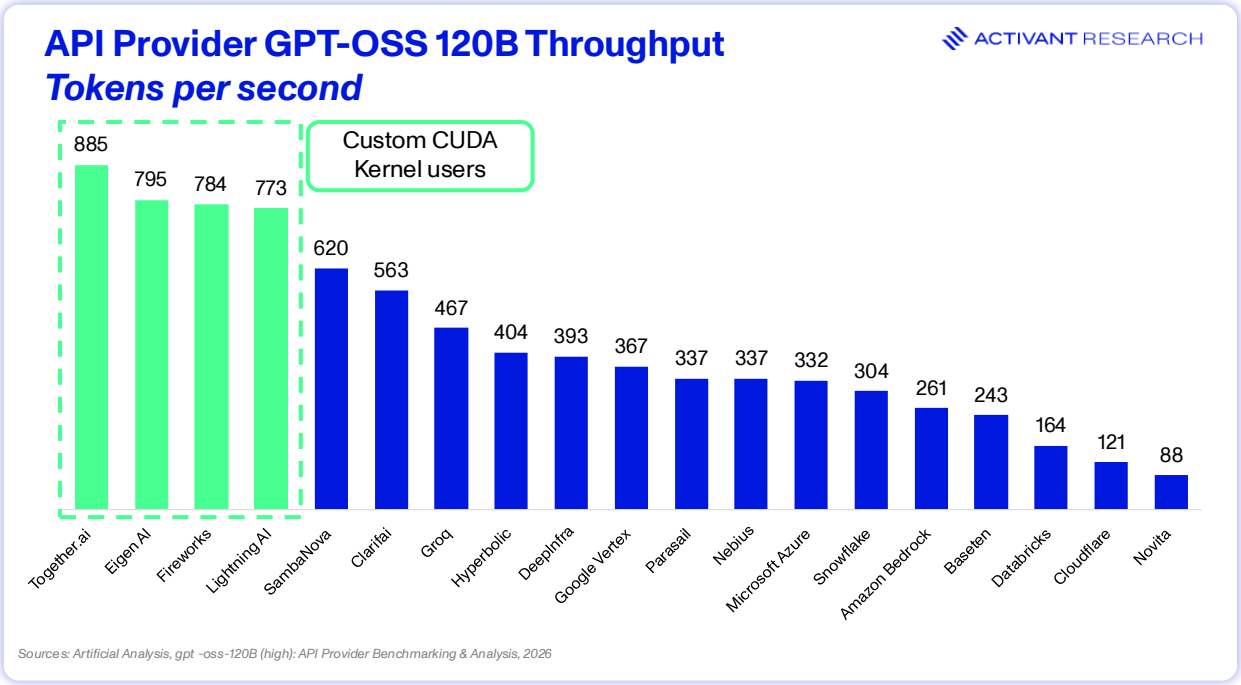
Infrastructure engineers can use standard kernels, like those provided by Nvidia as part of their software platform CUDA, but companies like Fireworks AI and Together AI write kernels from scratch specifically to drive performance.

The problem is that most AI inference is **memory-bound:** the GPU can spend 90% of its time waiting for data to arrive from memory and only 10% doing math.[19] Optimizing the kernel is about solving this memory bottleneck. Flash Attention, the open source kernel used by Together AI, leverages the ultra-fast on-chip memory (SRAM) rather than the default main memory (HBM). Flash Attention 3 achieves GPU utilization of 75%, a dramatic increase from 10% without kernel optimization.[20]

Together also uses speculative decoding, a technique where a smaller, faster "draft" model predicts the next few tokens while the larger "target" model verifies them in parallel. If the draft is correct, multiple tokens can be generated in a single step. In Together's ATLAS version, the draft model is updated in real time, becoming more intelligent as more tokens are processed.[21]

Fireworks, on the other hand, focus on quantization, using 4-bit floating precision. This allows them to fit models 4x larger into the same memory footprint or run standard models 4x faster compared with standard 16-bit precision. With **Quantization-Aware Training (QAT)**, the reduction in calculation precision has no impact on output accuracy.[22]

The result is that with the same Nvidia hardware, these kernel-optimized providers can process significantly more tokens than competitors.

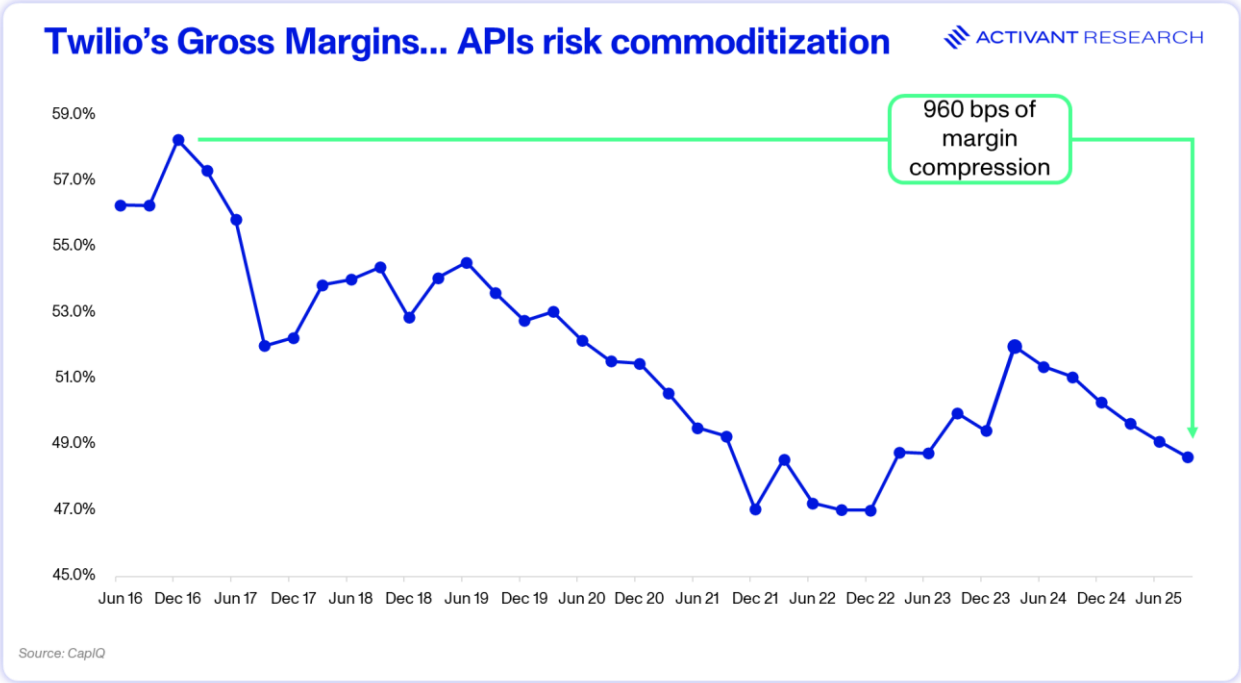


Serverless inference has innovated pricing, made open source models ridiculously easy to use, and squeezed out every last drop of performance. They've created highly compelling products, but have they built enduring businesses?

The Infrastructure Paradox

As the technology industry has seen time and again, solving a highly sophisticated infrastructure problem and packaging it up as an API can quickly shift from "magic" to commodity.

Take Twilio, for example. They were an early darling of the "API economy," trading at over 30x EV/Revenue in early 2021.[23] The business packaged a network of global carriers into a simple API to SMS customers. However, carrier fees prevented the company from reaching true SaaS margins and competition from companies like MessageBird and Sinch further pressured profitability.. Over time, SMS APIs became a commodity and Twilio had to pivot into higher margin SaaS like Customer Data Platforms to turn the business around.



Fireworks and peers, who have made accessing an open source model feel just like magic, face a similar risk of long-term commoditization:

1. **Perfect competition:** Today, the "product" offered by serverless inference providers is effectively a standardized output of an open-source model delivered via an OpenAI-compatible API. Switching costs are low and price discovery is high, a set up that many would describe as a race to the bottom. Aggregators like OpenRouter, which dynamically route queries to the lowest-cost provider, will only accelerate this risk.
2. **Graduation risk:** For small start-ups or teams deploying an AI app for the first time, using an infrastructure provider instead of rolling your own feels like a no brainer. But as apps scale or go viral, the API markup may exceed the cost of hiring a dedicated team to manage the infrastructure. Sophisticated customers can "graduate" off the API, effectively churning.

We estimate API providers cost ~1.8x DIY

ACTIVANT RESEARCH

Theoretical costing: GPT-OSS 120B	Best	Base	Worst	Notes
System Throughput Calculation				
Memory Requirement (GB)	59	59	59	*117 billion parameters, 0.5 bytes per param (4-bit quantization)*
(/) Nvidia H100 Memory Bandwidth (GB/s)	3,350	3,350	3,350	*nvidia.com/data-center/h100/*
Time per batch (Seconds)	**0.02**	**0.02**	**0.02**	
Batch Size	128	128	64	*Assumed based on 20GB VRAM headroom*
Theoretical throughput (tokens/second)	**7,330**	**7,330**	**3,665**	
(x) Achieved throughput	*50%*	40%	30%	*Assumed. Note: sources cite rates as low as 10%*
Actual Throughput (tokens/second)	**3,665**	**2,932**	**1,099**	
Annual Cost Calculation				
Assumed Daily Token Volume ('billions)	100	100	100	*Theoretical scaled technology business*
(/) Seconds per day	86,400	86,400	86,400	
Required Token throughput per second ('millions)	**1.2**	**1.2**	**1.2**	
(/) Actual Throughput (tokens/second)	3,665	2,932	1,099	
(x) Bursty Workload Buffer	1.25x	1.25x	1.25x	*25% Idle resource headroom to prevent cold starts*
Required GPU Capacity	**395**	**493**	**1,316**	
(x) Price / hour (GPU Rental)	$1.8	$1.8	$1.8	*CUDO Compute on demand H100 price, 14 Jan 2026*
(x) Hours / yr	8,760	8,760	8,760	
Annual Cost: DIY Infrastructure ($'millions)	**$6.2**	**$7.8**	**$20.7**	
API Cost Comparison				
Assumed Daily Token Volume ('billions)	100	100	100	
(x) Days/yr	365	365	365	
(x) Median Cost /mTokens (blended input/output)	$0.30	$0.40	$0.50	*Artificial Analysis, 14 Jan 2026*
Annual Cost: API Provider ($'millions)	**$10.95**	**$14.60**	**$18.25**	
Cost Differential ($'millions)	***$4.7***	***$6.8***	***($2.5)***	
API provider multiple on DIY infrastructure	***1.76x***	***1.88x***	***0.88x***	

As modelled above, a scaled technology business processing 100 billion tokens per day on a low-cost model like GPT-OSS 120B would spend just under $7 million more using the API than managing the software infrastructure (on rented GPUs) themselves. This calculation could easily flip the build-vs-buy decision in favor of build, but there is one major risk**. In the worst case, where the DIY approach achieves ~30% of maximum throughput, it's cheaper to use the API provider.**

The amount of alpha available in optimizing GPU efficiency is what sets this market apart from Twilio's SMS API, and it's where differentiated providers can drive sustainable gross margins.

It's All About Cost (Efficiency)

1. The Risk of Inefficiency Will Limit Graduation

Don't forget, actual GPU utilization rates have been cited as low as 10%, so achieving 40 – 50% of maximum throughput rates requires highly sophisticated staff, knowledgeable in the latest techniques and inference-serving packages. Each time a new model architecture or hardware cycle is released, teams may need to rewrite all their optimizations again.

Some may argue that open source packages like vLLM allow most teams to achieve strong utilization rates. For example, vLLM uses continuous batching to keep GPU memory utilization above 90% and even supports speculative decoding.[24] However, vLLM can introduce latency for

each user, at the cost of high memory utilization.[25] Further, vLLM is not self-driving: developers need to know how to configure items like speculative decoding, and ensure that they use it in a way that does not worsen latency.

The reality is that optimizing between these complex packages and doing it well, will only make sense, or be possible, for the largest and most sophisticated organizations. This leads to a critical thesis on serverless inference: **the graduation problem is largely irrelevant for the vast majority of potential customers.**

2. Performance Optimization Can Protect Gross Margins

The same math that validates serverless inference also explains how leading firms could protect gross margins. Companies with custom CUDA kernels and other critical optimizations, can drive more tokens than competitors out of the same GPU, in the same amount of time. It's a critical form of operating leverage that lets leading firms run profitably at prices where competitors are breakeven.

In a world where individuals with the right AI expertise are being offered $1bn+ pay packages, talent becomes a competitive advantage. **For serverless inference, the barrier to entry is low, but the barrier to the leading edge (custom kernels, proprietary runtimes) is extremely high.** Tri Dao, who wrote the open source kernel optimization Flash Attention is now Chief Scientist at Together AI. Similarly, most contributors to open source vLLM have moved to large research labs, or started companies.

If the ability to drive performance per dollar is concentrated in a few firms like Together, Fireworks and Modal Labs, they can cut prices to gain share or maintain higher margins at the same prices, deploying excess cash to expand their product suites and capture customers.

And at scale, with a large and diversified customer base, individual customer workloads may spike, but aggregate workloads can become smooth and predictable, a benefit of workload pooling. Companies that reach this scale can keep fewer idle resources warm while maintaining the same customer experience, further boosting profitability.

Activant Modeled Inference Gross Margins: *Sensitivity*

ACTIVANT RESEARCH

Throughput Achieved as % of GPU Maximum

GPU Capacity Buffer for Bursty Workloads	10%	20%	30%	40%	50%	60%	70%	80%
1.30x	(196%)	(48%)	1%	26%	41%	51%	58%	63%
1.25x	(184%)	(42%)	5%	29%	43%	53%	59%	64%
1.20x	(173%)	(36%)	9%	32%	45%	55%	61%	66%
1.15x	(161%)	(31%)	13%	35%	48%	56%	63%	67%
1.10x	(150%)	(25%)	17%	37%	50%	58%	64%	69%
1.05x	(139%)	(19%)	20%	40%	52%	60%	66%	70%

Lower required capacity buffer drives gross margin

Higher GPU utilization drives gross margin

Source: Activant Capital Model; Uses GPT-OSS 120B economics, assuming $0.30 blended price per mTokens

Ultimately, these software optimizations are the difference between being a low margin hardware reseller, and a high margin software business. The real risk for these providers then is that while they optimize the software stack on Nvidia GPUs, a completely different set of hardware upends the inference market.

The Hardware Risk

Software-centric inference providers are fighting a war against the physical constraints of memory bandwidth — the bottleneck that occurs when moving data between compute units and HBM. Quantization and speculative decoding mitigate these effects, but hardware-centric providers like Cerebras and Groq address it structurally, bypassing HBM entirely.

Take Cerebras for example. After struggling to break into the market for training, due to a weak software ecosystem and bespoke requirements that made it tough for server OEMs like Dell and Supermicro to adopt, they're making a major push for the inference market.

Their wafer-scale engine (effectively many reticle-sized chips stitched together on a single wafer), is 56x larger than a standard GPU and drives all communications on the silicon itself. Data no longer travels back and forth to HBM; instead it resides entirely on the chip with fast SRAM, amounting to 9,000x more on-chip memory bandwidth than an Nvidia H100.[26,27] This architecture

removes the significant memory bottlenecks that many are optimizing for on Nvidia GPUs and means that Cerebras' inference API processes over 3,000 tokens/second for GPT-OSS 120B, 3.5x that of Together AI.[28]

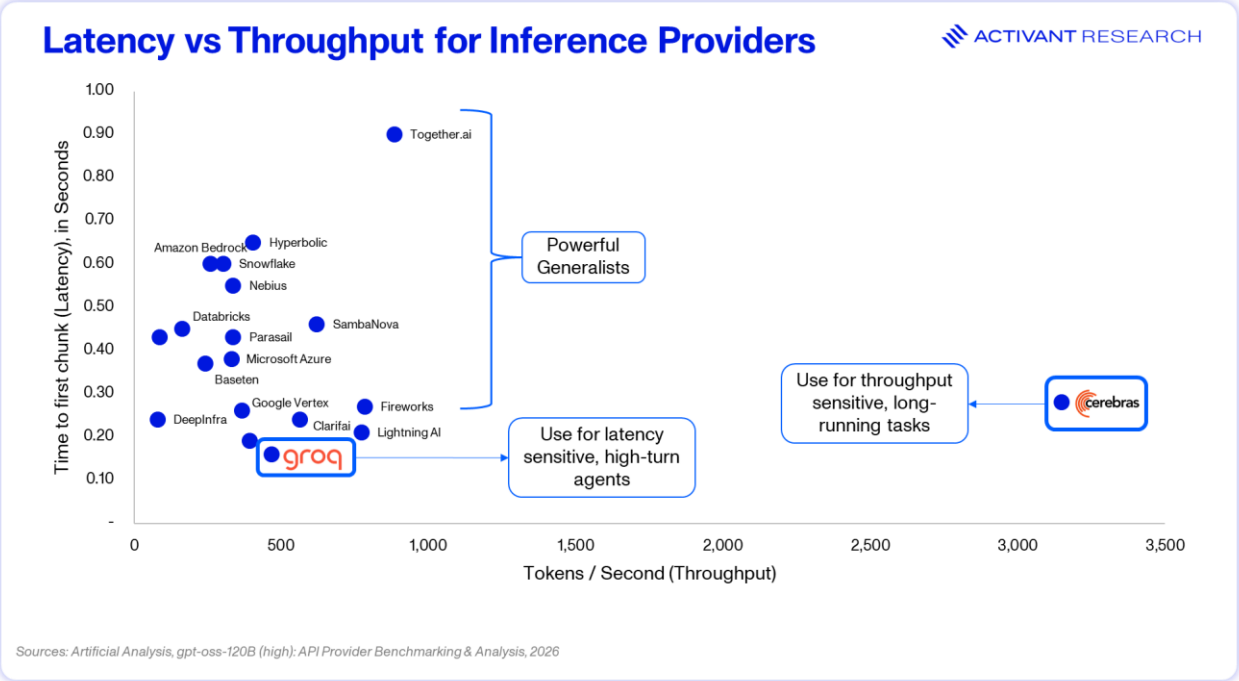
Such speed positions Cerebras well for extremely token-heavy tasks, like coding agents that write entire applications. That's how OpenAI sees it, citing coding as a key driver for their recent $10bn Cerebras purchase.[29]

Similarly, Groq's Language Processing Unit (LPU) uses no HBM and only on-chip SRAM. Like the famous Google TPU, it makes use of deterministic execution. Their software compiler controls every mathematical operation, rather than the traditional approach where the hardware execution runtime makes these decisions on chip. While known to be extremely difficult, when done well this approach can remove latency experienced by Nvidia customers. Groq provides 60% lower time-to-first-token (latency) than the median API provider and is known for remarkable *latency consistency*, driven by deterministic execution.[30]

There are, however, important trade-offs. The AI/ML ecosystem is heavily built on Nvidia software, including its CUDA ecosystem and a broad library of software packages. As a result, almost any model runs on Nvidia hardware instantly. By contrast, Cerebras and Groq support only three and eight models released in the last 12 months, respectively.[31,32]

Both companies also transform the economics of serving inference. Replacing HBM with SRAM optimizes for memory *bandwidth* but sacrifices memory *capacity* – you can go fast but hold less data. Therefore, both Cerebras and Groq are best for single batch (one user at a time) workloads, while companies optimizing Nvidia chips can serve batches of 64, 128 or even 248 users concurrently on the same hardware, resulting in less capex to reach scale. Further, more chips are required to be chained together to serve large models. It's estimated that hosting one instance of a 1 Trillion parameter model, like Kimi K2 would require upfront capex on Groq hardware of $30 million, compared to just $400k on Nvidia.[33]

While hardware-centric players bring unique strengths to the market that will capture specific workloads, we expect their economic models and lack of model breadth make it unlikely that they will take a significant market share in inference workloads.



Negating the Hardware Risk with DevOps Lock-In

Cementing the specialist hardware providers position as powerful but niche providers, is the need for fine-tuning and model customization. Fine tuning requires storing not just model weights but also optimizer states, gradients, and activations, **making fine-tuning (even LoRa), far more memory intensive than pure inference**. Hardware choices that prioritize SRAM rather than HBM for latency optimization (like Groq and Cerebras) are inefficient for these workloads.

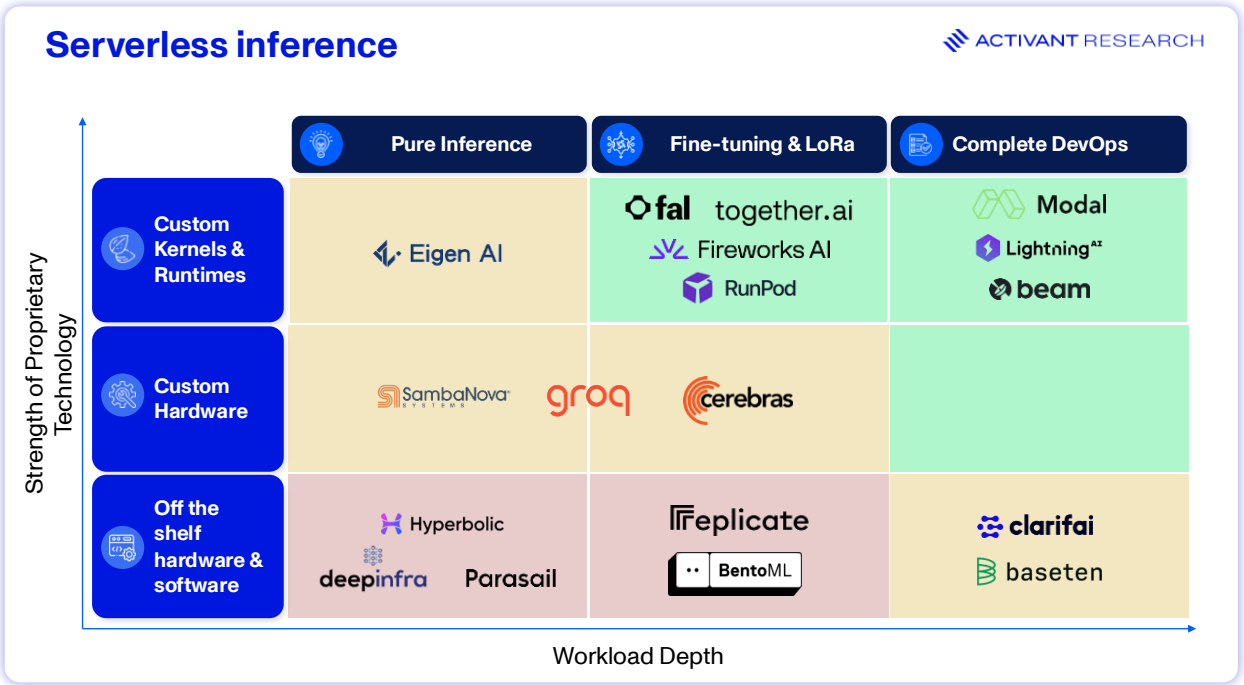
The use of methods like ***deterministic*** execution further compounds this challenge: updating model weights requires Groq to do a full recompilation at the software layer, a much slower process Nvidia's GPUs which execute ***dynamically***.

By contrast, Fireworks provides Reinforcement Fine-Tuning (RFT), which teaches models behaviors rather than static facts, allowing fine-tuned open source models to outperform state of the art closed models.[34] Together allows customers to manage hundreds of fine-tuned adaptations on base models, while charging only base model inference prices to run them.

These fine-tuning workloads add higher value service lines to the inference provider business model and make it harder for developers to port their models and workflows to competing providers.

Modal and Beam both provide Agentic Sandboxes, which allow agents to execute untrusted workflows and code in a secure perimeter. Clarifai's mesh workflow engine can chain multiple models and logical operations (e.g., OCR -> Translation -> Sentiment) into a single API call. Replicate's Cog and Baseten's Truss are both open source model packaging frameworks that allow developers to "write once, run anywhere," while still optimizing for their own infrastructure. Modal and Beam also allow developers to define infrastructure with decorators directly inside of python coding environments. Decorators make custom code easy to accelerate with GPUs, creating a natural lock-in to the provider who's logic was used to build the program.

These higher value and developer centric solutions suggest that it will be the software native inference providers, optimizing flexible Nvidia GPUs, who will lead the inference market. They will continue to capture higher value workloads like fine-tuning and custom development, driving stickiness and moving past perfect competition.



Closing thoughts

We believe that prioritizing software-level optimizations and deep developer lock-in, combined with added value services presents the strategy that will not just drive the success of serverless inference providers but also let them hold onto software-like economics as they ride the wave of a $450bn+ TAM shifting from training to inference.

Of course, as we close out this series, it's worth mentioning that AI remains one of the most rapidly changing and dynamic spaces in the history of both software and hardware. The future is always undecided, and we'll be watching many critical factors from the continued demonstration of the scaling laws and various physical bottlenecks to the rise of small language models, diffusion language models, semiconductor innovations and the strategic gambits of key players, like Nvidia's recent acquisition of Groq.

Endnotes

[1] Gartner, The Future of Cloud, 2026

[2] OpenAI, Developer QuickStart, 2026

[3] Reyanshicodes on Medium, How We Cut Spring Boot Cold Start Time by 85% in Kubernetes, 2025

[4] BentoML, 25x Faster Cold Starts for LLMs on Kubernetes, 2025

[5] Forbes, Why Brands Are Fighting Over Milliseconds, 2016

[6] Lin Qiao (Fireworks CEO), LinkedIn Post, 2025

[7] Google Cloud on Youtube, Gemini at Work 2025, 2025

[8] Fal.ai, Home Page, 2026

[9] Read The Signal, Together AI reached 100m ARR in less than 10 months, 2024

[10] Together AI, Build with Leading open-source Models, 2026

[11] BentoML, Neurolabs Accelerates Time to Market by 9 Months and Saves up to 70% with BentoML, 2024

[12] Baseten, Patreon saves nearly $600k/year in ML resources with Baseten, 2025

[13] **Runtime:** a specialized engine responsible for executing the software package (including model weights and related code) on the underlying hardware. The runtime is critical for decisions on memory management, batching strategy, and parallelization.

[14] **Compute Kernel**: runs on the hardware to perform specialized calculations directly on the hardware e.g. CUDA cores for Nvidia GPUs or Tensor Cores for Google TPUs. Kernels can also improve memory management, parallelization and calculation precision (quantization).

[15] Runpod, Introducing FlashBoot: 1-Second Serverless Cold-Start, 2023; *P95: 95% of all cold starts are at or below the indicated time*

[16] 6sense, Docker, 2026

[17] YouTube, Creating our Own Kubernetes & Docker to Run Our Data Infrastructure | Modal, 2023

[18] Modal Labs, Modal + Mistral 3: 10x faster cold starts with GPU snapshotting, 2025

[19] Paul Elvinger, et al., Understanding GPU Resource Interference One Level Deeper, 2025

[20] Together AI, FlashAttention-3: Fast and Accurate Attention with Asynchrony and Low-precision, 2024

[21] Together AI, AdapTive-LeArning Speculator System (ATLAS): A New Paradigm in LLM Inference via Runtime-Learning Accelerators, 2025

[22] Fireworks AI, FireAttention V4: Industry-Leading Latency and Cost Efficiency with FP4, 2025

[23] CapIQ

[24] Kwon, et al, Efficient Memory Management for Large Language Model Serving with PagedAttention, 2023

[25] Prabhu, et al, vAttention: Dynamic Memory Management for Serving LLMs without PagedAttention, 2025

[26] Cerebras, Cerebras Wafer Scale Engine 3, 2026

[27] Nvidia, H100, 2026

[28] Artificial Analysis, gpt-oss-120B (high): API Provider Benchmarking & Analysis, 2026

[29] WSJ, OpenAI Forges Multibillion-Dollar Computing Partnership With Cerebras, 2026

[30] Artificial Analysis, gpt-oss-120B (high): API Provider Benchmarking & Analysis, 2026

[31] Cerebras, pricing, Jan 2026

[32] Groq, pricing, Jan 2026

[33] Huatai Securities, NVDA's Next Mellanox: Groq's Latency-Critical LPU for Agentic AI, 2026

[34] Fireworks, Fireworks RFT: Build AI agents with fine-tuned open models that outperform frontier closed models, 2025