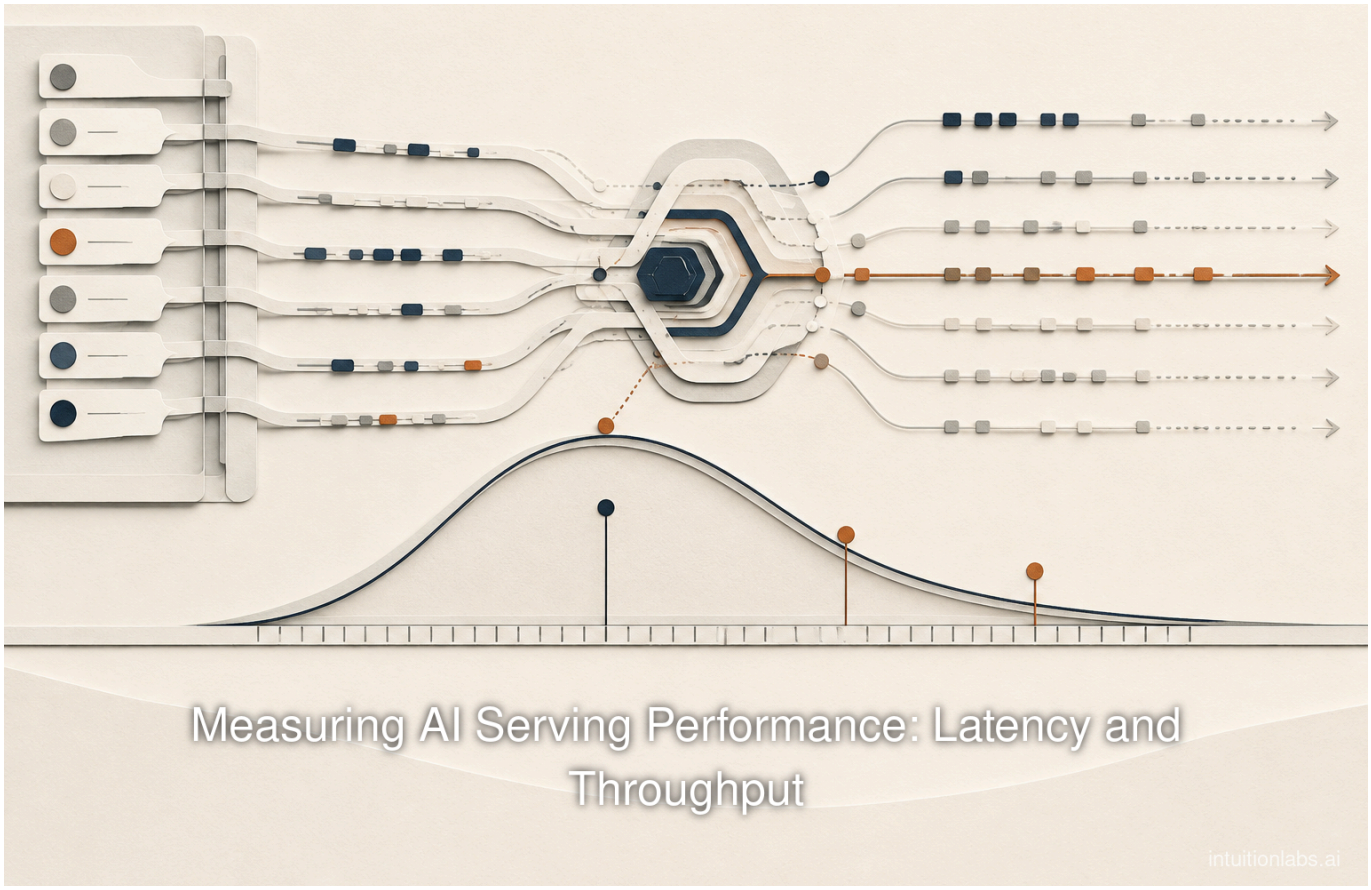
Measuring AI Serving Performance: Latency and Throughput

IntuitionLabs.ai · Adrien Laurent · 2026-09-05 · ai inference latency · llm throughput benchmark · time to first token · tokens per second benchmark · p99 tail latency · llm serving optimization · mlperf inference · inference sla metrics



Executive Summary

Measuring **AI inference** performance requires separating latency from throughput rather than relying on a single headline "**tokens per second**" figure. Practitioners track five distinct metrics: **Time to First Token (TTFT)**, the delay before any output appears, defined by vLLM as "the time from sending a request to receiving its first streamed output" (docs.vllm.ai); **Time Per Output Token (TPOT)**, also called Inter-Token Latency; **end-to-end request latency**; aggregate **throughput** in tokens or requests per second; and **goodput**, throughput restricted to requests meeting a latency service level objective (SLO), a concept vLLM's own benchmarking code traces to the academic DistServe system (raw.githubusercontent.com). Because a mean latency figure hides exactly the requests that hurt user experience, Google's Site Reliability Engineering guidance warns that "a simple average can obscure these tail latencies" (sre.google), which is why benchmarks report **p50, p95, and p99** percentiles instead.

Reproducible measurement depends on open-source tools including vLLM's benchmarking utilities, NVIDIA's GenAI-Perf, and Anyscale's LLMPerf, all of which expose concurrency, timeout, and input/output token-length distributions as explicit, documented parameters (github.com), plus the MLCommons-governed **MLPerf Inference** suite, whose v6.0 results were announced on April 1, 2026 (mlcommons.org). Vendor-reported figures, such as NVIDIA's claimed "98,443" tokens per second on eight Blackwell B200 GPUs (developer.nvidia.com), AMD partner Mangoboost's "around 103K tokens/sec" on a four-node MI300X cluster (rocm.blogs.amd.com), and Cerebras's claimed "up to 750 output tokens per second" (cerebras.ai), must be distinguished from independently measured results such as those published by third-party trackers like Artificial Analysis (artificialanalysis.ai).

Optimization techniques including continuous batching (measured at up to a "36.9x throughput improvement" in the academic Orca system (usenix.org)), PagedAttention-style KV-cache management, speculative decoding, quantization, and disaggregated prefill/decode serving each target a specific bottleneck rather than uniformly improving every metric. Commercial pricing increasingly reflects this: AWS Bedrock charges a "75% premium" for its Priority tier (aws.amazon.com) over Standard, while inference costs overall are falling roughly "10x every year" for equivalent model quality, according to venture firm a16z (a16z.com), even as the global AI inference market is projected to grow from roughly $97 billion to $254 billion by 2030 (grandviewresearch.com). For any organization deploying AI systems, the reliable path is to define a latency SLO first, then measure throughput and cost against it using a documented, reproducible method, rather than accepting an unconstrained headline benchmark at face value.

Introduction and Background

Every deployment of a **large language model (LLM)** eventually confronts the same question: is it actually fast enough. A model that scores well on accuracy benchmarks can still fail in production if a chatbot takes ten seconds to start responding, or if a batch job that looked fine in a demo collapses under fifty concurrent users. Answering that question requires more than a single headline number. It requires a consistent vocabulary for **latency** (how long a request takes) and **throughput** (how much work a system completes per unit time), plus a repeatable method for measuring both under realistic, concurrent load.

This guide defines the metrics that matter for **AI inference** (the process of running a trained model to generate outputs, as opposed to training it), the open-source tools practitioners use to measure them, and the tradeoffs that make a single "**tokens per second**" figure an unreliable basis for a purchasing or architecture decision. As of September 2026, the inference-serving stack, spanning frameworks such as **vLLM**, **NVIDIA TensorRT-LLM**, and **SGLang**, has converged on a shared set of definitions for time to first token, inter-token latency, and percentile-based tail latency, even as vendors continue to publish throughput claims using inconsistent baselines.

IntuitionLabs, a life-sciences and artificial intelligence (AI) consultancy founded in 2023, advises regulated enterprises on deploying generative AI systems, including AI-assisted chatbots and analytics tools built on the Veeva platform ecosystem. The firm states that it provides "consulting, implementation, and custom development services across the Veeva Development Cloud" (intuitionlabs.ai) and that it "leverage[s] AI and Generative AI for various applications, including predictive analytics, process automation, intelligent chatbots" (intuitionlabs.ai). For an advisory firm helping clients select and deploy such systems, understanding the difference between a vendor's marketing benchmark and a measured, load-tested result is a practical necessity, not an academic exercise. This report is organized to be reproducible: each method described states what was measured, on what inputs, and under what assumptions, so a reader could run the same test again.

Core Metrics: Defining Latency and Throughput in AI Serving

Inference performance is not one number. Practitioners generally track five distinct metrics, and conflating them is the most common source of misleading comparisons.

- **Time to First Token (TTFT)**: the delay between sending a request and receiving the first token of the response. vLLM's benchmarking documentation defines it as "the time from sending a request to receiving its first streamed output" (docs.vllm.ai), while NVIDIA's GenAI-Perf tool defines it almost identically as the "time between when a request is sent and when its first response is received" (docs.nvidia.com). TTFT is dominated by the **prefill phase**, in which the model processes the full input prompt in parallel before generating any output.
- **Time Per Output Token (TPOT)**, also called **Inter-Token Latency (ITL)**: the average time between successive output tokens once generation has started. vLLM calculates it "once per request, excluding the first token, and then aggregated across requests" (docs.vllm.ai), and its internal Prometheus metrics separately expose this as the interval "between first (after the most recent SCHEDULED) and last NEW_TOKENS" events during decoding (docs.vllm.ai). NVIDIA's GenAI-Perf computes ITL as the "time between intermediate responses for a single request divided by the number of generated tokens of the latter response" (docs.nvidia.com). TPOT reflects the **decode phase**, generating one token at a time.
- **End-to-end request latency**: the full wall-clock time a client experiences, from request submission to the final token. vLLM's benchmarking code defines it as "the time taken on the client side from sending a request to receiving a complete response" (raw.githubusercontent.com), and its server-side metrics define it as "the interval between frontend arrival_time and the frontend receiving the final token" (docs.vllm.ai). Approximately, end-to-end latency equals TTFT plus TPOT multiplied by the number of output tokens minus one.
- **Throughput**, measured two ways: aggregate **tokens per second** across a whole benchmark run, defined by GenAI-Perf as "total number of output tokens from benchmark divided by benchmark duration" (docs.nvidia.com), and **requests per second (RPS)**, the count of completed responses per unit time.
- **Goodput**: throughput restricted to requests that also satisfy a latency service level objective (SLO). The vLLM benchmarking tool implements this directly, letting an operator "specify service level objectives for goodput as "KEY:VALUE"" for TTFT, TPOT, and end-to-end latency thresholds (raw.githubusercontent.com), a concept the tool traces to the **DistServe** research paper on disaggregated inference serving.

Table 1 below summarizes these metrics, their unit of measurement, and the serving-stage each one primarily reflects.

Metric	What it measures	Typical unit	Serving phase reflected
TTFT	Delay before the first output token arrives	Milliseconds or seconds	Prefill (prompt processing)
TPOT / ITL	Average delay between subsequent tokens	Milliseconds per token	Decode (token generation)
End-to-end latency	Total client-perceived request time	Seconds	Prefill + decode combined

Metric	What it measures	Typical unit	Serving phase reflected
Aggregate throughput	Total output tokens generated across all requests, per second	Tokens/second	System-wide, load-dependent
Request throughput	Completed requests per second	Requests/second (RPS) or queries/second (QPS)	System-wide, load-dependent
Goodput	Throughput restricted to SLO-compliant requests	Requests/second meeting a latency bound	System-wide, quality-adjusted

As the table illustrates, no single row substitutes for the others: a system can post excellent aggregate tokens-per-second while individual users experience unacceptable TTFT, because aggregate throughput is, in the words of one independent technical analysis, "a capacity metric, not an experience metric" (clickhouse.com). Independent benchmark aggregator **Artificial Analysis** formalizes this split in its published methodology, defining TTFT as "the time in seconds between sending a request to the service or system and receiving the first token of the response" (artificialanalysis.ai) and tracking "output speed" separately as "the average number of tokens received per second, after the first token is received" (artificialanalysis.ai).

Benchmarking Methodology: How to Measure Performance Under Controlled Load

A reproducible inference benchmark requires four explicit choices: the tool, the workload shape, the concurrency profile, and the timeout and warm-up handling. This section documents each, using open-source tools whose scripts can be inspected and rerun.

Benchmarking tools

Three open-source tools are discussed here. **vLLM**'s own `vllm bench serve` command load-tests any OpenAI-compatible endpoint, though the project's maintainers now recommend a successor, noting "we recommend GuideLLM, an established performance benchmarking framework with live progress updates and automatic report generation" (docs.vllm.ai). **NVIDIA's GenAI-Perf**, part of the Triton Inference Server ecosystem, lets an operator directly control "the type of load to generate (number of concurrent requests, request rate)" (docs.nvidia.com), though NVIDIA states in its own documentation that "GenAI-Perf is being phased out. We are no longer actively developing new features for GenAI-Perf" (docs.nvidia.com) in favor of a successor project. **LLMPerf**, maintained by Anyscale under the Ray project, "spawns a number of concurrent requests to the LLM API and measures the inter-token latency and generation throughput per request and across concurrent requests" (github.com).

Workload shape

A defensible benchmark specifies input and output token-length distributions rather than a single fixed value, since real traffic varies. LLMPerf's command-line interface exposes this directly, letting an operator configure a run's mean and standard deviation for both prompt and completion lengths, for example with a flag such as "--mean-input-tokens 550" (github.com) alongside matching output-token parameters. The same interface exposes **timeout**

and **concurrency** as first-class parameters, for instance "--num-concurrent-requests 1" in its documented example (github.com), meaning a benchmark run must state, at minimum, how many simulated users were active simultaneously and how long a stalled request was allowed to run before being counted as a failure.

Concurrency and its effect on throughput versus latency

Increasing concurrent users raises aggregate throughput but slows the response each individual user sees, because more requests compete for the same compute and memory bandwidth. Databricks' engineering team documents this directly: "if we process 16 user queries concurrently, we'll have higher throughput compared to running the queries sequentially, but we'll take longer to generate output tokens for each user" (databricks.com). Anyscale's own benchmarking work adds a second dimension: how a fixed pool of GPUs is partitioned into serving replicas changes which metric improves, finding that "8 replicas with one GPU each is likely to be the lowest TTFT" configuration compared with fewer, larger replicas (anyscale.com) even when aggregate throughput is not maximized by that layout.

Industry-standard methodology

The **MLPerf Inference** suite, maintained by the MLCommons consortium, exists precisely to standardize these choices across submitters. Its rules require deterministic randomness, stating "all random numbers must be based on fixed random seeds and a deterministic random number generator" (raw.githubusercontent.com), and its "Server" scenario, meant to emulate realistic online traffic, generates arrivals so that "LoadGen sends new queries to the SUT (system under test) according to a Poisson distribution" (raw.githubusercontent.com) rather than a constant rate. Submitters must tune the target query rate against a latency bound; MLCommons' own Llama 2 70B benchmark documentation notes that this target "must be determined manually. It is usually around 80% of the Offline QPS, but on some systems, it can drop below 50%" (docs.mlcommons.org), showing that the Server target QPS is usually around 80% of Offline QPS but can drop below 50% on some systems.

A caution applies across all these tools: repeated runs against the same server can be distorted by caching effects. vLLM's own documentation warns that "repeating vllm bench serve against the same server can reuse prompts left in the prefix cache and inflate throughput" (docs.vllm.ai), and LLMPerf's maintainers state plainly that "the results may vary with the load" (github.com), meaning no single benchmark run at one concurrency level characterizes a system's full performance envelope.

Tail Latency, SLAs, and Why P99 Matters

A mean or median latency figure hides exactly the requests that most affect user experience. Google's **Site Reliability Engineering (SRE)** book, an industry-standard reference for online-service reliability, states plainly that "a simple average can obscure these tail latencies, as well as changes in them" (sre.google), and that tail behavior worsens specifically under load, "an effect exacerbated at high load by queuing effects" (sre.google). This is why AI serving benchmarks report **percentiles**, most commonly **p50** (median), **p95**, and **p99** (the worst 1 percent of requests), rather than a single average. Anyscale's LLMPerf methodology reflects this directly, stating the team is "interested not just in the mean TTFT, but the distribution: the P50, P90, P95 and P99" (anyscale.com).

The SRE book's broader framework defines three related terms worth distinguishing: a **Service Level Indicator (SLI)** is "a carefully defined quantitative measure of some aspect of the level of service" (sre.google) such as p99 latency; a **Service Level Objective (SLO)** is a target value for that indicator (for example, "p99 TTFT under 2 seconds"); and a **Service Level Agreement (SLA)** is a contractual commitment, typically with financial consequences, built on top of an SLO.

In practice, major cloud AI providers currently publish **uptime SLAs** far more often than **latency SLAs**. **Amazon Web Services (AWS)** commits only that it "will use commercially reasonable efforts to make Amazon Bedrock available" (aws.amazon.com) at a stated monthly uptime percentage, with service credits scaling up so that uptime "less than 95.0%" (aws.amazon.com) triggers the maximum 100 percent credit. Google Cloud's Vertex AI SLA blends the SLO and SLA terms in its own contract language, stating that "the Covered Service will provide a Monthly Uptime Percentage to Customer as follows (the "Service Level Objective" or "SLO")" (cloud.google.com) and committing to 99.5 percent uptime for "custom model online prediction for models deployed on 2 or more nodes" (cloud.google.com). Microsoft's Azure AI Foundry documentation is more explicit that latency guarantees are tier-dependent, listing a "defined latency target per model" (learn.microsoft.com) only for its Priority and Provisioned deployment tiers, while its Standard and Batch tiers carry none. Anthropic, meanwhile, manages capacity through rate limiting rather than a published latency SLA, using a token-bucket approach in which "a rate of 60 requests per minute (RPM) might be enforced as 1 request per second" (docs.anthropic.com) rather than allowing a burst, and it offers a distinct "Priority Tier" with its own rate-limit headers, including "anthropic-priority-input-tokens-limit" (docs.anthropic.com), for customers who need more predictable latency.

At the serving-infrastructure layer, request queuing is a direct, tunable lever on tail latency. **NVIDIA Triton Inference Server**'s dynamic batcher documentation explains that an operator can trade a small amount of queuing delay for higher throughput, since "the dynamic batcher will delay sending the batch as long as no request is delayed longer than the configured max_queue_delay_microseconds value" (raw.githubusercontent.com). Anyscale's own production engineering work reports a case where a batching change improved both metrics simultaneously, describing "how continuous batching enables 23x throughput in LLM inference while reducing p50 latency" (anyscale.com), though such simultaneous gains depend on the specific bottleneck being addressed and should not be assumed to generalize to every workload.

Optimization Techniques and Their Performance Tradeoffs

Serving-framework engineers have developed several distinct techniques to improve latency and throughput, each addressing a different bottleneck in the inference pipeline.

- **Continuous (iteration-level) batching**: introduced in the academic **Orca** system, this approach "schedules execution at the granularity of iteration" rather than waiting for an entire batch of requests to finish before starting new ones (usenix.org), and the paper reports a measured "36.9x throughput improvement at the same level of latency" against NVIDIA's FasterTransformer baseline on a GPT-3 175B model (usenix.org). Databricks separately measured that this style of batching "can achieve 10x-20x better throughput than dynamic batching" in its own testing (databricks.com). NVIDIA's developer blog explains the underlying problem this solves: in naive static batching, "all requests in the batch must wait until the longest request is finished" (developer.nvidia.com), wasting GPU cycles on already-finished sequences.

- **PagedAttention and key-value (KV) cache management**: the vLLM project's research paper found that prior systems wasted enormous memory to fragmentation, measuring that "only 20.4% - 38.2% of the KV cache memory is used to store the actual token states" in earlier designs (arxiv.org). By managing memory in non-contiguous pages, analogous to operating-system virtual memory, the paper reports that "vLLM improves the throughput of popular LLMs by 2-4x with the same level of latency" (arxiv.org) compared to prior serving systems. vLLM's own project documentation now lists "continuous batching of incoming requests, chunked prefill, prefix caching" (docs.vllm.ai) among its core built-in techniques.
- **Speculative decoding**: a smaller, faster "draft" model or lightweight decoding head proposes several tokens at once, which the full model verifies in parallel. NVIDIA's TensorRT-LLM documentation explains the mechanism directly: because verifying several drafted tokens costs little more than generating one, "the combination of both these allows speculative decoding to result in reduced latency" (nvidia.github.io) whenever enough drafted tokens are accepted.
- **Quantization**: reducing the numerical precision of model weights, for example from 16-bit to 8-bit or 4-bit representations. Hugging Face's documentation frames the core tradeoff as "storing the weights in a lower precision while trying to preserve as much accuracy as possible" (huggingface.co). NVIDIA reports that its INT4 AWQ (Activation-aware Weight Quantization) format enables running "Falcon-180B on a single H200 GPU with INT4 AWQ, and 6.7x faster Llama-70B over A100" hardware (nvidia.github.io), a vendor-reported figure specific to that model and hardware pairing.
- **Disaggregated prefill and decode serving**: a newer architecture that runs the compute-bound prefill phase and the memory-bandwidth-bound decode phase on separate GPUs rather than the same one. The academic **DistServe** system, which vLLM's own benchmarking code cites as the origin of the "goodput" concept, reports that this separation lets it "serve 7.4x more requests or 12.6x tighter SLO, compared to state-of-the-art systems" under the same latency constraints (arxiv.org). NVIDIA's own disaggregated-serving documentation explains why colocation hurts: running both phases on one GPU "can lead to interference where context processing delays token generation, increasing token-to-token latency (TPOT)" (nvidia.github.io).

These techniques are implemented across a small set of serving frameworks that are frequently compared to one another: **vLLM**, **NVIDIA TensorRT-LLM**, **SGLang** (which describes itself as "designed for low-latency, high-throughput inference with RadixAttention, prefix caching, and multi-GPU parallelism" (docs.sglang.ai)), Hugging Face's **Text Generation Inference (TGI)**, and **NVIDIA Triton Inference Server**, a general-purpose model server that "provides an optimized cloud and edge inferencing solution" (github.com) not limited to language models. All five continue to add overlapping features, so a framework choice should be validated against a specific model and hardware target rather than assumed from general reputation.

Finally, the decode phase's fundamental bottleneck is worth stating plainly, because it explains why most of these techniques exist: NVIDIA's own optimization guide notes that "the speed at which the data (weights, keys, values, activations) is transferred to the GPU from memory dominates the latency, not how fast the computation actually happens" (developer.nvidia.com) during decoding. This is why increasing raw compute (FLOPS) alone rarely fixes token-generation latency, and why memory-bandwidth-oriented architectures feature prominently in vendor throughput claims.

Data Analysis and Evidence

Quantitative claims in this field must be read carefully, because vendor-reported numbers and independently measured numbers frequently use different models, hardware configurations, and load levels.

MLPerf Inference, the closest available standard for cross-vendor comparison, announced v6.0 results on April 1, 2026 (mlcommons.org).

Within the v5.0 round, NVIDIA reported (as a vendor claim, corroborated by the official MLCommons submission IDs) that its "GB200 NVL72 delivers the highest Llama 3.1 405B performance per GPU, showing 2.8x faster offline and 3.4x faster in server mode" (developer.nvidia.com) than an eight-GPU H200 system, and published a direct tokens-per-second table showing 8x Blackwell **B200** GPUs reaching "Llama 2 70B Tokens/sec 98,443 | 98,858" combined tokens per second versus "33,072 | 34,988" for 8x H200 GPUs (developer.nvidia.com), a roughly threefold gain. NVIDIA separately disclosed a non-competitive figure obtained outside the official round, "achieving an unverified result of 869,203 tokens/second" (developer.nvidia.com) on an older benchmark version, a figure that should not be compared directly to MLCommons-audited results. On the competing hardware side, AMD's partner Mangoboost reported in the same official round that a "4-node MI300X cluster" achieved "the highest ever Llama 2 70B Offline performance of around 103K tokens/sec" (rocm.blogs.amd.com), and AMD's own blog states its first MI325X submission was directly comparable in performance to NVIDIA's single-node H200 entry in the same round (amd.com).

Vendor claims outside the audited MLPerf process are common and must be labeled as such. **Cerebras** states that as of August 27, 2026 it serves an OpenAI model in an "Ultrafast mode ... at up to 750 output tokens per second" on its wafer-scale hardware (cerebras.ai), and separately claims serving "OpenAI GPT OSS 120B at 3,000 tok/s" (cerebras.ai) on its public endpoint, both self-reported figures rather than MLPerf-audited results. **Groq** attributes its speed to a memory-architecture difference, stating its "on-chip SRAM has memory bandwidth upwards of 80 terabytes/second, while GPU off-chip HBM clocks in at about eight terabytes/second" (groq.com), an architectural claim rather than an audited benchmark. A fuller comparative treatment of these three alternative-hardware vendors, including deployment and cost considerations, is available in IntuitionLabs' report on Cerebras, SambaNova, and Groq. The report presents the result as a Groq marketing-materials case study involving an unnamed online retail company, rather than as an independently measured benchmark.

Independent, cross-provider measurement remains comparatively rare. **Artificial Analysis**, one of the few continuously updated independent benchmarks, maintains live, per-provider comparisons of the same underlying model hosted by different infrastructure operators, tracking "output speed" as "the average number of tokens received per second, after the first token is received" (artificialanalysis.ai) for each one separately, illustrating that even identical model weights can perform differently across hosting infrastructure, load balancing, and regional deployment. A peer-reviewed-adjacent 2025 analysis on arXiv cautions that many published metrics are themselves flawed, warning that "metric normalizations that hide substantial performance variability like generation stalls" can lead "to misleading conclusions" (arxiv.org) about a system's real-world performance, reinforcing that a single tokens-per-second figure, however it was obtained, should never be the sole basis for a procurement decision.

Case Studies and Real-World Examples

Character.AI's production optimization (measured engineering results)

In a November 21, 2024 engineering post, Character.AI's infrastructure team published measured, self-reported results from a custom int8 attention kernel deployed in its production LLM serving stack, reporting gains of "up to 10% in prefill" (blog.character.ai) speed and larger gains during decoding, illustrating that low-level kernel engineering, not just model or hardware choice, materially affects both TTFT and TPOT at scale.

Performance-tiered pricing as an implicit throughput and latency benchmark

Provider pricing should be assessed at the model and serving-path level rather than treated as a universal speed or reliability surcharge. AWS Bedrock states that "priority tier pricing is at 75% premium to Standard tier pricing" (aws.amazon.com), while its lower-priority "Flex" tier is discounted, and its Provisioned Throughput pricing lists Cohere Command at $49.50 per hour with no commitment, $39.60 with a one-month commitment, and $23.77 with a six-month commitment (aws.amazon.com). Fireworks AI documents three serving paths that route and price differently: Standard is the default, Priority is selected with `service_tier: "priority"`, and Fast is selected by switching the `model` ID to a Fast variant (docs.fireworks.ai), and Together AI structures its entire platform around "Serverless Inference" (together.ai) alongside separate provisioned-throughput and fully dedicated tiers.

Table 2 below summarizes how three inference providers tie pricing directly to a performance tier, illustrating that "cost per token" and "speed" are not independent variables in commercial AI serving.

Provider	Documented performance tiers	Pricing relationship	Observed
AWS Bedrock	Standard, Priority, Flex, Provisioned Throughput	Priority costs 75% more than Standard; Flex costs 50% less (aws.amazon.com)	September 2026
Fireworks AI	Standard, Priority, Fast	Fixed percentage premiums for faster serving paths (docs.fireworks.ai)	September 2026
Together AI	Serverless, Provisioned Throughput, Dedicated	Dedicated and provisioned tiers trade higher fixed cost for guaranteed capacity (together.ai)	September 2026

This pattern, an interpretive reading of the table above, means that a buyer evaluating "tokens per second" claims should also ask which pricing tier produced that number; a Standard-tier benchmark result is not representative of what a Priority-tier deployment of the same model will deliver, and vice versa.

Implications and Future Directions

Two trends are likely to shape how AI serving performance is measured and reported over the next several years. First, standardized, latency-gated benchmarking is spreading beyond MLPerf's audited rounds into everyday commercial practice, as the AWS, Fireworks, and Together AI pricing tiers above demonstrate; expect more providers to formalize explicit **latency-tiered pricing** rather than a single flat per-token rate. Second, the

disaggregated prefill and decode architecture demonstrated by DistServe and now supported natively in NVIDIA TensorRT-LLM is likely to become the default for large-scale deployments, since it directly targets the TPOT-degrading interference that colocated serving causes (nvidia.github.io).

Cost trends reinforce why measurement discipline matters more, not less, over time. Venture firm **a16z** documented what it calls "LLMflation," finding that "for an LLM of equivalent performance, the cost is decreasing by 10x every year" (a16z.com), a decline that makes raw cost-per-token an increasingly unstable basis for architecture decisions compared to a fixed latency or throughput requirement. At the same time, enterprise spending on generative AI is accelerating: Menlo Ventures' 2025 enterprise survey of 495 U.S. AI decision-makers found that enterprise generative AI spending rose from $11.5 billion in 2024 to $37 billion in 2025, "a 3.2x year-over-year increase" (menlovc.com), while research firm Grand View Research separately estimated the "global AI inference market size was estimated at USD 97.24 billion in 2024 and is projected to reach USD 253.75 billion by 2030" (grandviewresearch.com). MarketsandMarkets, using a different methodology, projected the same market would grow "from USD 106.15 billion in 2025 ... to USD 254.98 billion by 2030" (prnewswire.com), a figure notably close to Grand View's independent 2030 estimate despite the differing base years, which lends some cross-validation to both projections even though neither firm discloses identical methodology. IDC separately tracks the infrastructure side of this spending, projecting that "AI infrastructure spending will reach $497 billion in 2026, representing approximately 56% year-over-year growth" (idc.com).

For regulated industries such as life sciences, where IntuitionLabs operates as an implementation partner rather than an infrastructure vendor, this combination of falling per-token cost and rising deployment scale raises the practical stakes of measurement discipline: an AI-assisted workflow, such as the generative sales-operations or intelligent chatbot applications the firm has described building (intuitionlabs.ai), needs a documented TTFT and tail-latency target agreed with the business owner before deployment, not a vendor's headline tokens-per-second figure taken at face value.

Frequently Asked Questions (FAQs)

What is a good time to first token (TTFT) benchmark for a chatbot?

There is no single universal figure; a defensible answer is a percentile, not an average, and it depends on the model size, hardware, and concurrency at measurement time. OpenAI's own latency guidance identifies **streaming** the response as "the single most effective approach, as it cuts the waiting time to a second or less" (platform.openai.com) from the user's perceived perspective, even when the underlying TTFT is unchanged, because the user sees partial output immediately.

Which matters more: LLM inference throughput or latency?

Both, but for different reasons: throughput (tokens per second, requests per second) determines infrastructure cost per user served, while latency (TTFT, TPOT) determines whether an individual interaction feels responsive. A high-throughput, high-concurrency configuration and a low-latency, low-concurrency configuration are frequently different serving setups, as the Anyscale replica-count findings cited earlier illustrate (anyscale.com); the correct target depends on whether the application is interactive (favor latency) or batch (favor throughput).

What is tail latency and why does p99 AI serving performance matter?

Tail latency refers to the slowest fraction of requests, typically reported at the 95th or 99th percentile. Google's SRE guidance is direct that averages obscure exactly this behavior (sre.google), and in a production system with thousands of daily requests, even a 1 percent tail-latency failure rate translates into a meaningful number of frustrated users per day.

How do input (prompt) tokens affect latency compared to output tokens?

In one Anyscale LLMPerf experiment for Llama 2 70B on Anyscale Endpoints, comparing 550 input tokens with 3,500 input tokens, each additional input token added 0.3–0.7 ms to end-to-end time, compared with 30–60 ms for each output token—approximately 1% of the impact (anyscale.com). That configuration-specific result is not a universal ratio: report prompt and output lengths, concurrency, cache conditions, TTFT, and TPOT for the workload tested.

What are common inference SLA metrics for AI systems?

Uptime percentage is the most commonly contracted metric today, as the AWS Bedrock and Google Vertex AI SLA language above shows; explicit numeric latency SLAs remain rare outside of premium or provisioned tiers, where Microsoft Azure AI Foundry, for instance, offers a defined latency target only above the Standard tier (learn.microsoft.com).

Why is a single tokens-per-second benchmark number insufficient?

Because it conflates prefill and decode performance, ignores concurrency, and rarely discloses the latency bound (if any) under which it was measured. As the independent arXiv critique of inference benchmarking practice states, metric choices that hide "generation stalls" or similar variability "lead to misleading conclusions" (arxiv.org) about how a system will behave in production, and both MLPerf's own methodology and OpenAI's guidance for developers "groups them into seven principles that represent a high-level taxonomy" (platform.openai.com) of latency drivers precisely because no single number captures the whole picture.

Conclusion

Measuring AI inference performance correctly requires separating five distinct quantities: time to first token, time per output token, end-to-end latency, throughput, and goodput, and reporting each as a distribution, not an average. The open-source tooling to do this reproducibly already exists, from vLLM's benchmarking utilities and NVIDIA's GenAI-Perf to Anyscale's LLMPerf and the MLCommons-governed MLPerf Inference suite, each of which documents its own concurrency, timeout, and workload-shape parameters so a result can be independently reproduced. The evidence gathered here also shows a persistent gap between vendor-reported and independently measured figures. NVIDIA, AMD, Cerebras, and Groq each publish favorable numbers for their own hardware, and cross-checking those claims against MLPerf's audited results or an independent tracker such as Artificial Analysis remains the only reliable way to compare them.

For organizations evaluating AI serving infrastructure, the practical takeaway is to define a latency SLO first (a target TTFT and TPOT at a stated percentile and concurrency level), then measure throughput and cost against that constraint, rather than optimizing an unconstrained tokens-per-second figure that may not hold once real, concurrent traffic and a genuine latency requirement are applied. As inference costs continue falling roughly tenfold per year by some measures (a16z.com) and enterprise deployment scales accordingly, the discipline of publishing a reproducible method, not just a result, is what will separate a durable benchmark from a marketing claim.
