Energy Use of AI Inference: Efficiency Pathways and Test-Time Compute

Felipe Oviedo, Fiodar Kazhamiaka, Esha Choukse, Allen Kim, Amy Luers, Melanie Nakagawa, Ricardo Bianchini, Juan M. Lavista Ferres

Microsoft

As AI inference scales to billions of queries and emerging reasoning and agentic workflows substantially increase token demand, reliable estimates of per-query energy use are increasingly important for capacity planning, emissions accounting, and efficiency prioritization. Yet many public estimates tend to be inconsistent and systematically overstate energy use, because they extrapolate from limited benchmarks and fail to reflect the efficiency gains achievable in at-scale deployments. In this perspective, we introduce a bottom-up methodology to estimate the per-query energy of large-scale LLM systems based on token throughput estimation. For models running on an H100 node under realistic workloads, GPU utilization and PUE constraints, we estimate a median energy per query of 0.34 Wh (IQR: 0.18-0.67) for frontier-scale models (>200 billion parameters). These results are consistent with measurements using production-scale configurations and show that non-production estimates and assumptions can overstate energy use by 4-20×. Extending to test-time scaling scenarios with 15× more tokens per typical query, the median energy rises 13-fold to 4.32 Wh, indicating that targeting efficiency in this regime will likely deliver the largest fleetwide savings. We explored how to target efficiency interventions to maximize impact. By quantifying achievable efficiency gains at the model, serving platform, and hardware levels, we find that individual levers yield median reductions of 1.5-3.5× in energy per query, while combined advances can plausibly deliver 8-20× reductions. To illustrate the systemlevel impact, we estimate the baseline daily energy use of a deployment serving 1 billion queries to be 0.8 GWh/day. If 10% are long queries, demand could grow to 1.8 GWh/day. With targeted efficiency interventions, it falls to 0.9 GWh/day—similar to the energy footprint of web search at that scale. This echoes how data centers historically tempered energy growth through efficiency gains during the internet and cloud build-up.

CCS CONCEPTS • Artificial Intelligence • Inference • Large Language Models • Energy efficiency

1 INTRODUCTION

LLM (Large Language Model) inference is one of the fastest-growing sources of computing demand, with energy use already approaching that of web search and other mature digital services [1], [2], [3]. As a result, many stakeholders are seeking reliable figures on how much energy is consumed during inference at a granular level (e.g., energy per textual query) to provide a firm basis for energy projections, compare serving approaches, and relate efficiency considerations to engineering and product trade-offs. Despite this need, public reports of "energy per query" differ by more than an order of magnitude [2], [4].

Two issues drive this discrepancy. First, measurement boundaries vary: some account only for GPU draw, others include CPUs or application overhead. Second, many benchmarks fail to capture realistic production conditions such as batching, concurrency, and steady-state serving [5], [6], [7]. Model-based estimates are also highly sensitive to assumptions about hardware specifications, parameter counts, and token lengths [8], [9], [10]. Direct measurements, though more precise, are often tied to non-production models or small-batch tests [6]. These settings tend to overestimate energy use relative to large, high-concurrency deployments, which handle most AI inference workloads.

An added challenge to characterizing inference energy is that per-query figures cannot be interpreted in isolation. The energy used in inference also depends on how energy demand scales when billions of queries are served, and how interventions at the model, serving software, and hardware layers can moderate that growth. Recent model provider disclosures offer a more realistic baseline [2], [11] by measuring large-scale systems but remain limited by confidentiality. Typically, these reports only include a single "representative" or "average query", masking how energy use changes with query length distributions and model and serving configurations. This limitation is significant: test-time scaling (reasoning models) and agentic workflows are increasingly routine, and their long outputs can drive a disproportionate share of aggregate energy demand. Without this system context, averages give a misleading picture of energy use in inference. Furthermore, such metrics fail to provide guidance on where future efficiency efforts should focus.

With these limitations in mind, we propose a bottom-up estimation approach to fill the gaps. Our framework aligns with production-scale disclosures but adds two contributions: (i) a decomposition of the interacting determinants of energy per query—model, serving platform, and hardware—to provide insight into practical efficiency interventions; and (ii) explicit modeling of token-length distributions, showing how long inference increases energy per query.

2 ESTIMATING ENERGY PER QUERY IN LARGE-SCALE DEPLOYMENTS

To estimate the energy use of text queries, we focus on at-scale deployments where nodes are saturated through batching, high concurrency, and optimized serving. LLM inference has two stages: prefill, where the input sequence is processed once in parallel, and decoding, where output tokens are generated step by step. Because decoding cannot be parallelized across tokens, it dominates energy use in most real-world workloads—a trend amplified in reasoning and agentic workflows [12].

Our estimation framework is designed around these conditions. We propose a bottom-up Monte Carlo simulation that captures these interacting factors for various open-source models. For a fixed GPU node, the per-query energy E_{query} (in Wh) is estimated as:

$$E_{\text{query}} = \frac{\text{PUE}}{3.6} \left(\frac{P_{\text{node}} \cdot L_{\text{eff}}}{TPS(L_{\text{in}}, L_{\text{out}})} \right) [1]$$

Here, P_{node} is the steady-state power draw per GPU node during inference. For our analysis, we assume models are hosted on 8xH100 GPUs at FP8 precision, as in the NVIDIA DGX H100 architecture [13], or 10xH100 (in the case of DeepSeek-R1), with a maximum power draw P_{max} of 11.3 kW or 14.1 kW (linearly scaled), respectively. For highly utilized nodes, average power utilization has been reported around 70% during inference [12], [4]. Accordingly, we model P_{node} as a log-normal distribution centered at $0.7 \cdot P_{\text{max}}$ with P5–95 range of $0.4 \cdot P_{\text{max}} - 0.9 \cdot P_{\text{max}}$. The aggregated effect of idle nodes and overhead is explored later. PUE is the power usage effectiveness of the AI data center, modeled as a log-normal distribution with P5–P95 range of 1.05–1.40, consistent with hyperscaler public reports. The constant 3.6 converts kW to Wh.

 $L_{\rm in}$, $L_{\rm out}$ denote the number of input and output tokens for a query. We model a variety of workloads by sampling $L_{\rm in}$, $L_{\rm out}$, choosing discrete values for $L_{\rm in}$ and sampling $L_{\rm out}$ from an exponential distribution. We analyze two regimes: i) Traditional, common in conversational queries with no intensive test-time scaling [14], with median $L_{\rm out} = 300$ tokens (IQR: 129-618), and ii) Test-time scaling, consisting of long queries common in reasoning models such as OpenAI o3 [15] or DeepSeek-R1 [16] or long agentic workflows, with median $L_{\rm out} = 5,000$ (IQR: 2,040-9,717) [17]. $L_{\rm eff}$ denotes the effective token length and scales the energy by the number of tokens in the query. Since the output tokens dominate energy use, we approximate $L_{\rm eff} \sim L_{\rm out}$ and fix $L_{\rm in} = 500$ in our main analysis. In the Supplemental (S.III), we explore additional $L_{\rm in}$ and $L_{\rm eff}$ configurations.

TPS is the token throughput per second. TPS varies according to the model architecture, deployment configuration, concurrency, and latency constraints. In our case, since we are interested in modeling energy usage in a steady-state deployment, we take advantage of the existing benchmark data for H100 using NVIDIA's TensorRT-LLM [18], [19] considering parallelism of 8 (10 in DeepSeek-R1) and continuous batching. As presented in S.I. in Supplemental, TPS for the models of interest hosted on 8xH100s has been measured for multiple $L_{\rm in}$, $L_{\rm out}$ configurations. The reported TPS in these benchmarks approximates the throughput of a fully saturated node with high concurrency. For our simulation, $TPS(L_{\rm in}, L_{\rm out})$ is a piecewise log-linear regression model trained on this curated data, as detailed in S.II in Supplemental. The log-linear model is trained in available benchmark data of up to 4,000 output tokens. For longer outputs, we hold TPS at the plateau rather than modeling the gradual decline due to growing KV (Key-Value) and attention overhead. This simplification likely underestimates energy for very long generation but provides a conservative baseline. Although correlated, for simplicity, we treat TPS as independent of $P_{\rm node}$ in the main analysis, and explore a linear dependency in the S.IV in Supplemental, leading to similar results.

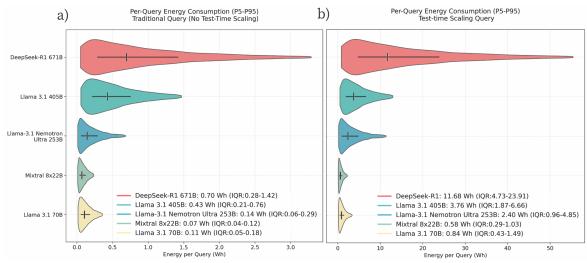


Figure 1: P5-P95 energy per query for several LLMs in two regimes, after sampling 10,000 queries (L_{out} , P_{node} and PUE configurations). a) Standard query, with 300 median output tokens (IQR: 129-618). b) Test-time scaling query, with median 5,000 median output tokens (IQR: 2,040-9,717). In both cases, we assume 500 input tokens and estimate token throughput for realistic deployments on an H100 node. Bars represent median and interquartile range.

To estimate $E_{\rm query}$, we sample 10,000 random configurations of $L_{\rm out}$, $P_{\rm node}$ and PUE, and compute the corresponding TPS for a given model. Figure 1 summarizes the results for five dense and mixture-of-expert (MoE) models: DeepSeek-R1 671 B, Llama 3.1 405B, Llama 3.1 Nemotron Ultra 235 B, Mixtral 8x22B, and Llama 3.1 70B. In the traditional regime, Figure 1a, our Monte Carlo estimates align with the measured energy of production-optimized deployments. For example, Llama 3.1 405B consumed a median of 0.43 Wh per query, consistent with ML.ENERGY leaderboard [4] with median of 0.548 Wh on an 8xH100 node (0.931 Wh using BF16 quantization, approximately 0.931/1.7=0.548 Wh using FP8). Similarly, Mixtral 8x22B was estimated to consume a median of 0.07 Wh per query, close to the ML.ENERGY estimate of 0.092 Wh. When in-production deployments are not considered, the per-query estimates are substantially larger: AI Energy Score [6] reports ~0.86 Wh per query for Llama 3.1 70B (average of 1.72 Wh for 1,000 queries with FP16 quantization, ~1.72/2=0.86 Wh at FP8), under similar assumptions IEA [20] estimated ~1.25 Wh per-query for Mixtral 8x22B and ~2.25 Wh per-query for DeepSeek-R1. These latter estimates did not consider basic in-production optimization techniques, including batching and inference acceleration, leading to 4–20× overestimation of energy per query. Benchmarking to these numbers or extrapolating them to large-scale inference is not representative of the energy use of widely used AI products.

Our approach provides an estimate for the energy of test-time scaling queries in Figure 1b. As expected, the per-query energy scales approximately linearly with the total output tokens. The energy cost of very long queries can be substantial, but those models that maintain high throughput for long queries, such as Mixtral 8x22B, seem to have a scaling advantage at the same number of parameters.

3 LINE-OF-SIGHT REDUCTIONS IN ENERGY USE

To analyze energy implications and efficiency opportunities, we defined a "Baseline" energy usage scenario by sampling the estimated energy use of the three models above 200B parameters (Figure 1a: DeepSeek-R1 671B, Llama 3.1 405B, Llama 3.1 Nemotron Ultra 253B). This parameter threshold is based on the size of popular, frontier-scale models, including open weight models like DeepSeek-R1 [16], Llama 3.1 405B [21], Qwen 3 [22], and is also a reasonable range for proprietary models such as GPT-40 or Claude 3.5 Sonnet [23]. As presented in Figure 2, the median energy per query in a baseline of models over 200B parameters is 0.34 Wh (IQR: 0.18–0.67). This is consistent with recent estimates for a typical chatbot query in a highly-optimized deployment: 0.3 Wh for a typical GPT-40 query of 500 output tokens (heuristic estimate based on FLOPs of a 200B model) [10], 0.421 Wh for GPT-40 query with 300 output tokens (based on user-side token throughput) [8], 0.34 Wh for the "average energy use of a ChatGPT query", as disclosed by Sam Altman [11], and 0.24 Wh for "the typical Gemini Apps query" [2]. Although not a standardized comparison due to a diversity of products, models, workloads and measurement approaches, this energy range is substantially lower than previous and widely reported estimates and makes it comparable to that commonly cited for web search, 0.3 Wh [24]. Various factors have led to this improvement compared to the previous generation of models: reduced size of new frontier models (shift from Kaplan to Chinchilla scaling laws, leading to smaller models trained with more data) [23], hardware efficiency gains [25], and model and serving improvements.

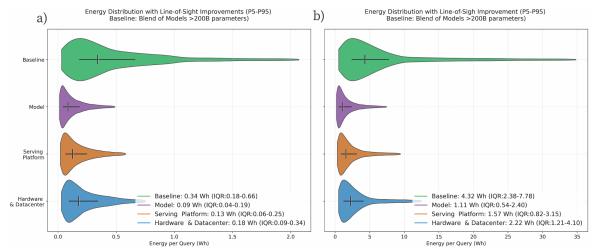


Figure 2 Baseline and efficiency gains for LLM >200B parameters, after sampling 10,000 queries (L_{out} , P_{node} and PUE configurations). P5-P95 energy per query in two regimes: a) Traditional query, with 300 median output tokens (IQR: 129-618). b) Test-time scaling query, with median 5,000 median output tokens (IQR: 2,040-9,717). In both cases, we assume 500 input tokens and estimate token throughput for realistic deployments on an H100 node. Bars represent median and interquartile range.

In the test-time scaling regime, sampling of models over 200B parameters led to a median estimate of 4.32 Wh per query (IQR: 2.38–7.38) with over 10 Wh in a significant portion of queries. Given this substantial footprint, even a small portion of test-time scaling queries or token-intensive agentic workflows can have a major impact on the total energy consumption of AI inference. From our perspective, this represents a key frontier for energy efficiency.

For this purpose, we define several of these line-of-sight energy efficiency opportunities and quantify their potential impact. We modify Eq. 1, defining a multiplier α to scale-up token throughput (α ·TPS) or energy-per-query (1/ α); given

a fixed compute node, α is defined as a log-normal distribution with P5-P95 ranges estimated from literature to capture uncertainty of the impact of each optimization. We categorize these improvements into:

- i) Model ($\alpha = [1.5, 10]$): Algorithm and architecture improvements have been the largest drivers of energy efficiency for inference workloads. Based on recent techniques, we estimate line-of-sight improvements of 1.5–10× for next generation deployments.
 - Developments in model distillation have led to energy reductions of 5–10x [2]. Notably, model distillation not only leads to more FLOP-efficient models but also allows larger batch sizes with the same hardware, potentially reducing embodied emissions. Distillation has been proven useful for reasoning models: smaller models with longer inference have been shown to outperform larger non-reasoning models, e.g., 7 to 70B reasoning model distilled from DeepSeek-R1 demonstrated comparable performance to models three to five times larger [16], and carefully distilled Qwen 3 reasoning models outperformed larger models [22]. In certain areas, small language models (SML) can reduce energy by one to two orders of magnitude for token-intensive tasks such as math, coding, or specific agentic workflows compared to models above 100 billion parameters [26], [27], [28].
 - Model quantization already has substantial impact in production: FP8 in H100 already delivers 1.5–2×
 TPS over FP16. Emerging low-bit quantization techniques supported in newer hardware can plausibly
 increase inference TPS by 1.3–3× if model performance holds steady [29], [30]. Although quantized
 models could allow efficient parallelism and request management in large deployments, technical
 challenges for running heavily-quantized models with large batches must be addressed [31].
 - MoE has demonstrated success in reducing inference compute compared to equivalent dense models. Multiple MoE inference optimization approaches are emerging, including optimized expert parallelism (e.g., DeepSpeed-MoE [32]), dynamic gating and load balancing (e.g., Tutel [33]), expert offloading/caching (e.g., MoE-Lightning [34]). These approaches report 1.3–8× inference speedups depending on baseline and workload. Additionally, attention operations and architectures have been optimized for efficient inference, such as FlashAttention [35] or Multi-Head Latent Attention [16], and long sequence generation. Focusing on efficient reasoning models, Llama-Nemotron [19] demonstrated 1.9–4× TPS compared to DeepSeek-R1 using neural architecture search, selective attention removal and feedforward network optimization. Hybrid attention models using variations of linear or sliding attention have achieved gains for test-time scaling [36], [37], e.g., MiniMax-M1 [37] consumes 4× less FLOPs generating 100K tokens compared to DeepSeek-R1.
- ii) Serving platform and workload management ($\alpha = [1.5, 5]$): Optimizing model deployments to service-level objectives has a significant impact on the energy and carbon footprint of inference [1], [38], [39]. Furthermore, large-scale AI systems allow efficiency gains by workflow management, for example routing models according to query complexity. We estimate line-of-sight improvements of $1.5-5\times$ due to these interventions.
 - Adaptive serving based on service-level objectives demonstrated over 50% reduction in energy consumption [38]. Disaggregated serving allows to separate the prefill and decoding inference stages, and optimize each stage with different resources, caching or accelerators [40], [41], leading to 1.4—

- 2.3× TPS improvement [38]. Due to disaggregation, better resource utilization in the decoding stage is especially impactful in long generation and agentic workflows with TPS gains of 1.4–4× in distributed settings [41], [42], especially when combined with dynamic GPU allocation, smart routing, and low-latency communication [39].
- KV cache management and quantization are critical for serving, techniques, such as PQCache, KVQuant, and LServe, have demonstrated long sequence generation with over 1.7× inference speedup [36], [43], [44]. Speculative decoding has demonstrated 2–3× TPS speedups [45].
- Automated or user-triggered model routing can have a large impact in token throughput and memory and token utilization and thus has become popular in the next-generation of models, such as GPT-5. This is particularly useful for reasoning models: we estimate that models that allow explicitly switching of reasoning capabilities can reduce per-query energy by 5x or more due to reduced query length. Smart model routing can actively maintain response performance and reduce per-query energy and cost, e.g., routing with continual learning has been able to maintain 97% response quality with ~2–4x cost reduction [46], [47]. Dynamic filtering of low-quality reasoning traces, as done by DeepConf [48], has demonstrated up to 99.9% accuracy with ~5x query length reduction.
- iii) Hardware and datacenter (α = [1.5, 2.5]): Hardware and datacenter advances are expected to remain a foundation of inference speed-ups. NVIDIA's Blackwell architecture delivers a substantial improvement in TPS per watt compared to H100/H200 GPUs. In MLPerf Inference v5.0 running Llama 3.1 405B, a single Blackwell B200 achieved up to 2.8–3.4× higher TPS per GPU than H100 [49]. SemiAnalysis notes that Blackwell improves TFLOPS per GPU-watt by 47-82% relative to H100 at the same quantization level [25]. Beyond commodity GPUs, custom AI inference chips (e.g., ASIC, FPGA) have TPS per W gains of around 5–20× [50], in part due to better performance in the memory-bound decoding stage, a critical capability for test-time scaling. Furthermore, power management techniques such as oversubscription, voltage/frequency scaling and power capping can lead to up to allocating 30% more servers in existing clusters, and 20% peak power reduction [12]. At the datacenter level, cooling optimization can provide additional gains, e.g., cold-plate and immersion cooling can reduce energy demand by 15–20% compared to air cooling [51]. Taken together, without considering improved quantization or custom hardware, these data points support a TPS per GPU-W increase of 1.5–2.5×.

Figure 2 summarizes the estimated impact of these interventions, considering each category as non-additive to others. In the traditional regime, hardware gains or improved serving could reduce the energy footprint of the typical query by half; model improvements could lead to even a larger reduction (~4x) but are conditional on model performance staying competitive. In the reasoning regime, the energy reductions are similar for those interventions but notably the long-tail P95 energy consumption is reduced. Model and serving platform optimization can have substantial impact (~3-4x), opening a pathway for intensive use of test-time scaling: an optimized reasoning query with a median of 5,000 tokens could be served at around 1 Wh, comparable to much smaller queries without optimization. Adding these efficiencies, we observe that the proposed line-of-sight interventions could lead to a reduction of energy per query of 8–20x or more (for example, 1.5–2x due to hardware in combination with 3–4x due to model improvements and/or 2–3x for serving and workflow optimization), a gain consistent with those reported in large-scale production systems [2].

4 ENERGY USE OF A BILLION QUERIES PER DAY

Based on the discussed scenarios, we extrapolate the energy consumption of serving 1 billion queries per day. The billion queries scale aligns with usage patterns reported across leading conversational AI platforms [52], [53], such as ChatGPT at 2.5 billion queries per day as of July, 2025 [53], and is comparable to around 10 billion daily queries served by dominant web search platforms [54]. Using our proposed framework, we sampled the energy for serving 1 billion queries in the baseline scenario and compute the total energy of serving those queries. As our estimate is performed at the node-level, we scale our total energy estimate by a factor b=1.33, accounting for non-uniform demand profile during the day, node redundancy, and InfiniBand [55], as described in the S.V in Supplemental. Figure 3a presents the total energy use of 1 billion traditional queries per day. The naïve "Baseline" scenario leads to 0.81 GWh, whereas a very conservative line-of-sight improvement ($\alpha = [1.5,3]$) results in less than half the total energy use. As reference, a single 30 MW H100 data center operating at constant 80% utilization will consume under 0.6 GWh of energy per day, comparable to the "Baseline" scenario. Taking commonly reported energy use per web search results in 0.3 GWh for 1 billion web searches. Thus, we observe that our conservative line-of-sight improvement for AI inference reduces the traditional AI inference energy footprint to the range of web search, while AI remains substantially more useful. In contrast to past claims [2], [6], [20], given that inference can be distributed across geographies and electrical grids, this is a large but manageable load: even if AI inference would scale up to 10 billion queries per day to be on par with the magnitude of web search, this will result in around 1.32 TWh/year, or about ~6% of the current energy use by large hyperscalers such as Microsoft or Google or ~1% of the current US data center electricity consumption [20], [56]. The largest strain of new AI loads in grids is not caused by the energy use of inference, but by training loads and the rate of AI adoption and concentrated capacity buildup, which are more challenging to distribute and optimize.

A potential source of concern, however, is the test-time scaling regime. In Figure 3b, we model a mixed inference scenario where 90% of queries are traditional, while 10% are long-inference queries. A naïve estimate, not accounting for improvements, results in multiplicative energy growth: 10% of test-time scaling queries can more than double the daily energy footprint of serving 1 billion queries in the "Baseline" scenario. However, line-of-sight efficiency improvements, including model routing and workflow management, can effectively reduce this energy growth with limited reliance on algorithmic breakthroughs. In the optimized scenario with 10% of test-time scaling queries, this results in a total energy consumption of 0.89 GWh for 1 billion queries, equivalent to the daily energy consumption to power roughly 0.4% of television watching in US households [57]. Thus, although test-time scaling can substantially raise energy consumption, its multiplicative effect can be contained through hardware and serving optimizations—and because cost per token is the dominant driver of AI adoption, efficiency gains in long inference should be expected and actively pursued.

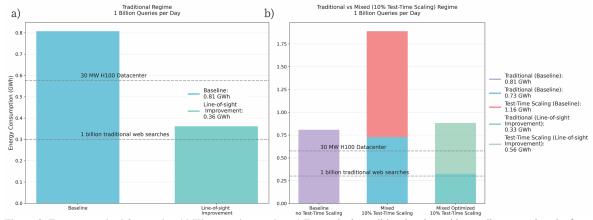


Figure 3: Energy required for serving 1 billion queries per day. a) Energy in the traditional regime with a median output length of 300 tokens (IQR: 129-618). Baseline estimate (model > 200B parameters) is compared to a conservative improved scenario with efficiency gains of $\alpha = 1.5-3$. b) Energy in the traditional regime compared to a mixed scenario with 10% test-time scaling queries with median of 5,000 output tokens (IQR: 2,040-9,717). Although very long inference has a large effect on energy, the conservative improved scenario with efficiency gains of $\alpha = 1.5-3$ moderates energy growth.

Historically, projections of digital infrastructure energy use have repeatedly overestimated demand by ignoring rapid gains in efficiency and system-level optimization amid multiplicative demand growth [56]. For instance, a 1999 forecast warned that the internet could consume 50% of the U.S. grid by 2010, but actual data center usage peaked around 2% [56]. Another high-profile projection in 2015 estimated data centers would consume 8,000 TWh/year by 2030—more than 25% of global electricity—but updated estimates now suggest 600–800 TWh, or about 2% [56]. In our opinion, the current surge in AI inference demand may seem unprecedented, but it follows the same historical pattern: while absolute energy use will grow, efficiency gains in hardware, software and deployment strategies at scale can moderate its long-term energy footprint.

5 CONCLUSIONS

In this work, we proposed a bottom-up methodology to estimate the energy use of AI inference. By combining node-level power parameters, utilization factors, PUE distributions, and empirically fitted token throughput under realistic serving conditions, our estimates align with disclosures from optimized production environments. This framework enables us to quantify how existing optimization techniques and token-length distributions shape overall energy consumption.

This study has several limitations. First, our simulation focuses on single-node inference with relatively high utilization, and does not fully capture inter-node orchestration overhead, ramp-up dynamics, token throughput and power utilization correlation, or network latency in distributed systems. Second, we assume an idealized workload distribution and token-length spectrum for both traditional and long inference, which may differ across real-world enterprise and consumer deployments. Third, our modeling of prefill costs in long-context queries is limited; in extreme long input scenarios (e.g., 100 thousand tokens), prefill will be more significant. Finally, we assume flat token throughput for very long generations, even though real systems often show decline; this likely leads to an underestimate of energy in such cases.

Nevertheless, our analysis leads to three central conclusions, each with direct technical and policy implications:

 Non-production energy use per query is overestimated. Energy estimates or measurements based on small-scale or non-optimized deployments overestimate the energy use of inference by 4–20×. For

- policymakers and stakeholders, this means that alarmist claims drawn from narrow benchmarks should be avoided. Instead, disclosures and estimates should emphasize ranges of energy use that reflect workload and serving diversity, utilization, and deployment scale. Comparisons must be performed with a systems approach, in the context of products, rather than relying on comparing isolated models.
- 2. More and longer inference increases energy use, but efficiency gains can limit the increase. Rapid AI adoption, test-time scaling and agentic workflows substantially raise the inference energy use, but their impact can be managed. The largest efficiency gains are expected from algorithmic advances—particularly due to small/distilled LLMs and inference-optimized architectures—and from intelligent model routing, which can prevent unnecessary use of large models. Hardware improvements and serving optimization also contribute meaningfully. Together, these levers can plausibly yield 8–20× line-of-sight reductions. Realizing these efficiencies promptly will be critical to moderating the energy impact of AI inference as it scales.

ACKNOWLEDGMENTS

The authors thank William Chappell, Lucas Meyer, Rahul Dodhia and Karin Strauss for useful discussion.

REFERENCES

- [1] Y. Li, Z. Hu, E. Choukse, R. Fonseca, G. E. Suh, and U. Gupta, "EcoServe: Designing Carbon-Aware Al Inference Systems," Mar. 15, 2025, arXiv: arXiv:2502.05043. doi: 10.48550/arXiv.2502.05043.
- [2] C. Elsworth et al., "Measuring the environmental impact of delivering Al at Google Scale".
- [3] U. Gupta et al., "Chasing Carbon: The Elusive Environmental Footprint of Computing," Oct. 28, 2020, arXiv: arXiv:2011.02839. doi: 10.48550/arXiv.2011.02839.
- [4] J.-W. Chung et al., "The ML.ENERGY Benchmark: Toward Automated Inference Energy Measurement and Optimization," May 09, 2025, arXiv: arXiv:2505.06371. doi: 10.48550/arXiv.2505.06371.
- [5] A. S. Luccioni, S. Viguier, and A.-L. Ligozat, "Estimating the Carbon Footprint of BLOOM, a 176B Parameter Language Model," Nov. 03, 2022, arXiv: arXiv:2211.02001. doi: 10.48550/arXiv.2211.02001.
- [6] "Al Energy Score," Al Energy Score. Accessed: Sept. 16, 2025. [Online]. Available: https://huggingface.github.io/AlEnergyScore/
- [7] S. Samsi et al., "From Words to Watts: Benchmarking the Energy Costs of Large Language Model Inference," Oct. 04, 2023, arXiv: arXiv:2310.03003. doi: 10.48550/arXiv.2310.03003.
- [8] N. Jegham, M. Abdelatti, L. Elmoubarki, and A. Hendawi, "How Hungry is AI? Benchmarking Energy, Water, and Carbon Footprint of LLM Inference".
- [9] "The growing energy footprint of artificial intelligence ScienceDirect." Accessed: Sept. 16, 2025. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2542435123003653
- [10] J. You, "How much energy does ChatGPT use?," Epoch Al. Accessed: June 19, 2025. [Online]. Available: https://epoch.ai/gradient-updates/how-much-energy-does-chatgpt-use
- [11] "Sam Altman," Sam Altman. Accessed: June 19, 2025. [Online]. Available: https://blog.samaltman.com/
- [12] P. Patel et al., "Characterizing Power Management Opportunities for LLMs in the Cloud," in *Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 3*, La Jolla CA USA: ACM, Apr. 2024, pp. 207–222. doi: 10.1145/3620666.3651329.
- [13] "Introduction to NVIDIA DGX H100/H200 Systems NVIDIA DGX H100/H200 User Guide." Accessed: Sept. 23, 2025. [Online]. Available: https://docs.nvidia.com/dgx/dgxh100-user-guide/introduction-to-dgxh100.html
- [14] W.-L. Chiang et al., "Chatbot Arena: An Open Platform for Evaluating LLMs by Human Preference".
- [15] "Introducing OpenAI o3 and o4-mini." Accessed: Sept. 16, 2025. [Online]. Available: https://openai.com/index/introducing-o3-and-o4-mini/
- [16] DeepSeek-Al et al., "DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning," Jan. 22, 2025, arXiv: arXiv:2501.12948. doi: 10.48550/arXiv.2501.12948.
- [17] L. Emberson, "LLM responses to benchmark questions are getting longer over time," Epoch Al. Accessed: Sept. 23, 2025. [Online]. Available: https://epoch.ai/data-insights/output-length
- [18] NVIDIA, "TensorRT-LLM/docs/source/performance/perf-overview.md at release/0.19 · NVIDIA/TensorRT-LLM," GitHub. Accessed: Sept. 23, 2025. [Online]. Available: https://github.com/NVIDIA/TensorRT-LLM/blob/release/0.19/docs/source/performance/perf-overview.md
- [19] A. Bercovich et al., "Llama-Nemotron: Efficient Reasoning Models," June 30, 2025, arXiv: arXiv:2505.00949. doi: 10.48550/arXiv.2505.00949.
- [20] "EnergyandAl.pdf." Accessed: June 19, 2025. [Online]. Available: https://iea.blob.core.windows.net/assets/34eac603-ecf1-464f-b813-2ecceb8f81c2/EnergyandAl.pdf
- [21] A. Grattafiori et al., "The Llama 3 Herd of Models," Nov. 23, 2024, arXiv: arXiv:2407.21783. doi: 10.48550/arXiv.2407.21783.
- [22] A. Yang et al., "Qwen3 Technical Report," May 14, 2025, arXiv: arXiv:2505.09388. doi: 10.48550/arXiv.2505.09388.

- [23] E. Erdil, "Frontier language models have become much smaller," Epoch Al. Accessed: Sept. 16, 2025. [Online]. Available: https://epoch.ai/gradient-updates/frontier-language-models-have-become-much-smaller
- [24] U. Hölzle, S. V. President, and Operations, "Powering a Google search," Official Google Blog. Accessed: Sept. 23, 2025. [Online]. Available: https://googleblog.blogspot.com/2009/01/powering-google-search.html
- [25] "Nvidia Blackwell Perf TCO Analysis B100 vs B200 vs GB200 NVL72," SemiAnalysis. Accessed: Aug. 25, 2025. [Online]. Available: https://semianalysis.com/2024/04/10/nvidia-blackwell-perf-tco-analysis/
- [26] H. Xu et al., "Phi-4-Mini-Reasoning: Exploring the Limits of Small Reasoning Language Models in Math," Apr. 30, 2025, arXiv: arXiv:2504.21233. doi: 10.48550/arXiv.2504.21233.
- [27] R. Liu et al., "Can 1B LLM Surpass 405B LLM? Rethinking Compute-Optimal Test-Time Scaling," Feb. 10, 2025, arXiv: arXiv:2502.06703. doi: 10.48550/arXiv.2502.06703.
- [28] G. Team et al., "Gemma 3 Technical Report," Mar. 25, 2025, arXiv: arXiv:2503.19786. doi: 10.48550/arXiv.2503.19786.
- [29] "Introducing NVFP4 for Efficient and Accurate Low-Precision Inference," NVIDIA Technical Blog. Accessed: Aug. 26, 2025. [Online]. Available: https://developer.nvidia.com/blog/introducing-nvfp4-for-efficient-and-accurate-low-precision-inference/
- [30] Y. Liu et al., "VPTQ: Extreme Low-bit Vector Post-Training Quantization for Large Language Models," Oct. 22, 2024, arXiv: arXiv:2409.17066. doi: 10.48550/arXiv.2409.17066.
- [31] E. Frantar, R. L. Castro, J. Chen, T. Hoefler, and D. Alistarh, "MARLIN: Mixed-Precision Auto-Regressive Parallel Inference on Large Language Models," in *Proceedings of the 30th ACM SIGPLAN Annual Symposium on Principles and Practice of Parallel Programming*, Las Vegas NV USA: ACM, Feb. 2025, pp. 239–251. doi: 10.1145/3710848.3710871.
- [32] S. Rajbhandari et al., "DeepSpeed-MoE: Advancing Mixture-of-Experts Inference and Training to Power Next-Generation Al Scale".
- [33] C. Hwang et al., "Tutel: Adaptive Mixture-of-Experts at Scale," Proceedings of Machine Learning and Systems, vol. 5, pp. 269–287, Mar. 2023.
- [34] S. Cao et al., "MoE-Lightning: High-Throughput MoE Inference on Memory-constrained GPUs," Nov. 18, 2024, arXiv: arXiv:2411.11217. doi: 10.48550/arXiv.2411.11217.
- [35] J. Shah, G. Bikshandi, Y. Zhang, V. Thakkar, P. Ramani, and T. Dao, "FlashAttention-3: Fast and Accurate Attention with Asynchrony and Low-precision".
- [36] S. Yang et al., "LServe: Efficient Long-sequence LLM Serving with Unified Sparse Attention," Apr. 21, 2025, arXiv: arXiv:2502.14866. doi: 10.48550/arXiv.2502.14866.
- [37] MiniMax et al., "MiniMax-M1: Scaling Test-Time Compute Efficiently with Lightning Attention," June 16, 2025, arXiv: arXiv:2506.13585. doi: 10.48550/arXiv.2506.13585.
- [38] J. Stojkovic, C. Zhang, Í. Goiri, J. Torrellas, and E. Choukse, "DynamoLLM: Designing LLM Inference Clusters for Performance and Energy Efficiency," Aug. 01, 2024, arXiv: arXiv:2408.00741. doi: 10.48550/arXiv.2408.00741.
- [39] "NVIDIA Dynamo," NVIDIA. Accessed: June 19, 2025. [Online]. Available: https://www.nvidia.com/en-us/ai/dynamo/
- [40] P. Patel et al., "Splitwise: Efficient generative LLM inference using phase splitting," May 20, 2024, arXiv: arXiv:2311.18677. doi: 10.48550/arXiv.2311.18677.
- [41] C. Hu et al., "MemServe: Context Caching for Disaggregated LLM Serving with Elastic Memory Pool," Dec. 21, 2024, arXiv: arXiv:2406.17565. doi: 10.48550/arXiv.2406.17565.
- [42] "Dynamo 0.4 Delivers 4x Faster Performance, SLO-Based Autoscaling, and Real-Time Observability," NVIDIA Technical Blog. Accessed: Aug. 27, 2025. [Online]. Available: https://developer.nvidia.com/blog/dynamo-0-4-delivers-4x-faster-performance-slo-based-autoscaling-and-real-time-observability/
- [43] C. Hooper et al., "KVQuant: Towards 10 Million Context Length LLM Inference with KV Cache Quantization," May 28, 2025, arXiv: arXiv:2401.18079. doi: 10.48550/arXiv.2401.18079.
- [44] H. Zhang et al., "PQCache: Product Quantization-based KVCache for Long Context LLM Inference," Mar. 30, 2025, arXiv: arXiv:2407.12820. doi: 10.48550/arXiv.2407.12820.
- [45] C. Chen, S. Borgeaud, G. Irving, J.-B. Lespiau, L. Sifre, and J. Jumper, "Accelerating Large Language Model Decoding with Speculative Sampling," Feb. 02, 2023, arXiv: arXiv:2302.01318. doi: 10.48550/arXiv.2302.01318.
- [46] X. Wang et al., "MixLLM: Dynamic Routing in Mixed Large Language Models," Feb. 09, 2025, arXiv: arXiv:2502.18482. doi: 10.48550/arXiv.2502.18482.
- [47] D. Ding et al., "Hybrid LLM: Cost-Efficient and Quality-Aware Query Routing," Apr. 22, 2024, arXiv: arXiv:2404.14618. doi: 10.48550/arXiv.2404.14618.
- [48] Y. Fu, X. Wang, Y. Tian, and J. Zhao, "Deep Think with Confidence," Aug. 21, 2025, arXiv: arXiv:2508.15260. doi: 10.48550/arXiv.2508.15260.
- [49] "NVIDIA: MLPerf AI Benchmarks," NVIDIA. Accessed: Aug. 25, 2025. [Online]. Available: https://www.nvidia.com/en-us/data-center/resources/mlperf-benchmarks/
- [50] J. Li et al., "Large Language Model Inference Acceleration: A Comprehensive Hardware Perspective," Jan. 23, 2025, arXiv: arXiv:2410.04466. doi: 10.48550/arXiv.2410.04466.
- [51] "Using life cycle assessment to drive innovation for sustainable cool clouds | Nature." Accessed: Sept. 20, 2025. [Online]. Available: https://www.nature.com/articles/s41586-025-08832-3
- [52] "Microsoft to spend record \$30 billion this quarter as Al investments pay off | Reuters." Accessed: Sept. 02, 2025. [Online]. Available: https://www.reuters.com/business/microsoft-spend-record-30-billion-this-quarter-ai-investments-pay-off-2025-07-30/?utm_source=chatgpt.com
- [53] M. Allen, "Altman plans D.C. push to 'democratize' Al economic benefits," Axios. Accessed: Sept. 02, 2025. [Online]. Available: https://www.axios.com/2025/07/21/sam-altman-openai-trump-dc-fed
- [54] "AI, personalization and the future of shopping," Google. Accessed: Sept. 02, 2025. [Online]. Available: https://blog.google/products/ads-commerce/ai-personalization-and-the-future-of-shopping/
- [55] "Specifications," NVIDIA Docs. Accessed: Sept. 23, 2025. [Online]. Available: https://docs.nvidia.com/networking/display/QM97X0PUB/Specifications
- [56] G. Kamiya and V. C. Coroamă, "Data Centre Energy Use: Critical Review of Models and Results".

[57] A. Luers, "Net zero needs AI — five actions to realize its promise," Nature, vol. 644, no. 8078, pp. 871–873, Aug. 2025, doi: 10.1038/d41586-025-02641-4.

SUPPLEMENTAL

I. TPS Benchmark

Model	TP Size	Quantization	Tokens per Second (TPS)	Input Length	Output Length	Source
Llama 3.1 405B	8	FP8	2050.00	500	2000	llamaNemotron Article
Llama 3.1 405B	8	FP8	480.00	5000	500	llamaNemotron Article
Llama 3.1 405B	8	FP8	3732.40	128	128	tensorRT-LLM-May2025 v019
Llama 3.1 405B	8	FP8	4572.23	128	2048	tensorRT-LLM-May2025 v019
Llama 3.1 405B	8	FP8	2911.42	128	4096	tensorRT-LLM-May2025 v019
Llama 3.1 405B	8	FP8	3661.85	500	2000	tensorRT-LLM-May2025 v019
Llama 3.1 405B	8	FP8	2963.36	1000	1000	tensorRT-LLM-May2025 v019
Llama 3.1 405B	8	FP8	3253.17	1000	2000	tensorRT-LLM-May2025 v019
Llama 3.1 405B	8	FP8	3089.16	1024	2048	tensorRT-LLM-May2025 v019
Llama 3.1 405B	8	FP8	448.89	2048	128	tensorRT-LLM-May2025 v019
Llama 3.1 405B	8	FP8	2139.94	2048	2048	tensorRT-LLM-May2025 v019
Llama 3.1 405B	8	FP8	579.14	5000	500	tensorRT-LLM-May2025 v019
Llama 3.1 405B	8	FP8	370.26	20000	2000	tensorRT-LLM-May2025 v019
Llama 3.1 70B	8	FP8	14686.01	128	128	tensorRT-LLM-May2025 v019
Llama 3.1 70B	8	FP8	17463.99	128	2048	tensorRT-LLM-May2025 v019
Llama 3.1 70B	8	FP8	12598.55	128	4096	tensorRT-LLM-May2025 v019
Llama 3.1 70B	8	FP8	14630.41	500	2000	tensorRT-LLM-May2025 v019
Llama 3.1 70B	8	FP8	11082.48	1000	1000	tensorRT-LLM-May2025 v019
Llama 3.1 70B	8	FP8	11108.11	1000	2000	tensorRT-LLM-May2025 v019
Llama 3.1 70B	8	FP8	11028.32	1024	2048	tensorRT-LLM-May2025 v019
Llama 3.1 70B	8	FP8	1840.12	2048	128	tensorRT-LLM-May2025 v019
Llama 3.1 70B	8	FP8	8772.88	2048	2048	tensorRT-LLM-May2025 v019
Llama 3.1 70B	8	FP8	2399.78	5000	500	tensorRT-LLM-May2025 v019
Llama 3.1 70B	8	FP8	1568.84	20000	2000	tensorRT-LLM-May2025 v019
Mixtral 8x22B	8	FP8	21876.08	128	128	tensorRT-LLM-Feb2025 v017
Mixtral 8x22B	8	FP8	25150.03	128	2048	tensorRT-LLM-Feb2025 v017

Mixtral 8x22B	8	FP8	18387.40	128	4096	tensorRT-LLM-Feb2025 v017
Mixtral 8x22B	8	FP8	21421.86	500	2000	tensorRT-LLM-Feb2025 v017
Mixtral 8x22B	8	FP8	16573.24	1000	1000	tensorRT-LLM-Feb2025 v017
Mixtral 8x22B	8	FP8	2794.97	2048	128	tensorRT-LLM-Feb2025 v017
Mixtral 8x22B	8	FP8	12244.93	2048	2048	tensorRT-LLM-Feb2025 v017
Mixtral 8x22B	8	FP8	3645.27	5000	500	tensorRT-LLM-Feb2025 v017
Mixtral 8x22B	8	FP8	2227.63	20000	2000	tensorRT-LLM-Feb2025 v017
DeepSeek-R1	10	FP8	886.00	1024	1024	tensorRT-LLM-DeepSeekR1
Llama-3.1 Nemotron Ultra 253B	8	FP8	5200.00	500	2000	llamaNemotron Article
Llama-3.1 Nemotron Ultra 253B	8	FP8	720.00	5000	500	llamaNemotron Article
DeepSeek-R1	10	FP8	1300.00	500	2000	llamaNemotron Article
DeepSeek-R1	10	FP8	378.00	5000	500	llamaNemotron Article

II. TPS regression

To estimate tokens-per-second (TPS) as a function of input and output lengths, we used a piecewise regression approach. We fit a regression of the form:

$$log(TPS) = \beta_0 + \beta_1 log(L_{in}) + \beta_2 log(L_{out})$$

where $L_{\rm in}$ is the input token length and $L_{\rm out}$ is the output token length. We chose a log-linear function as it provided a better fit than linear regression.

Predictions are transformed back to the original scale as:

$$\widehat{\text{TPS}}(L_{\text{in}}, L_{\text{out}}) = e^{(\beta_0 + \beta_1 \log L_{\text{in}} + \beta_2 \log L_{\text{out}})}$$

Predictions were capped at the maximum observed TPS_{max} as L_{out} increases for a fixed L_{in} :

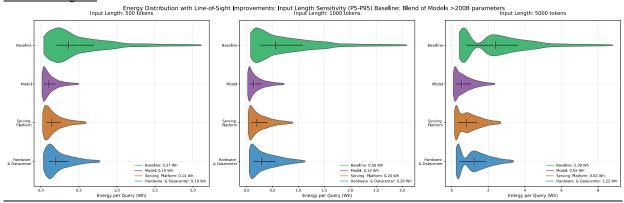
$$\widehat{TPS} = \min(\widehat{TPS}, TPS_{max})$$

This capping captures the plateau observed in the benchmarked range but does not model the throughput decline seen in very long generations, so our framework likely underestimates energy use in such cases.

III. Effect of query input length on energy use

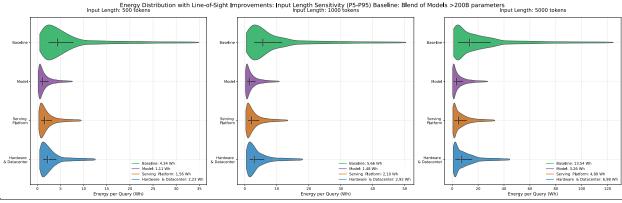
In these sections, we explore the effect increasing the input length from Lin = 500 to Lin = 1000 and Lin = 5000 for all queries and sampling from an exponential distribution L_{out} . Instead of assuming $L_{eff} \sim L_{out}$, we define $L_{eff} = L_{in} + L_{out}$.

Traditional regime:



When setting Lin = 500, the distribution is very close (\sim 10%) to our original estimate assuming Leff = Lout. Lin = 1000 increases the energy per query, but with a lower effect than increasing Lout. When Lin = 5000, the energy distribution becomes bimodal because long prefill dominates queries with short outputs.

Test-time scaling regime:

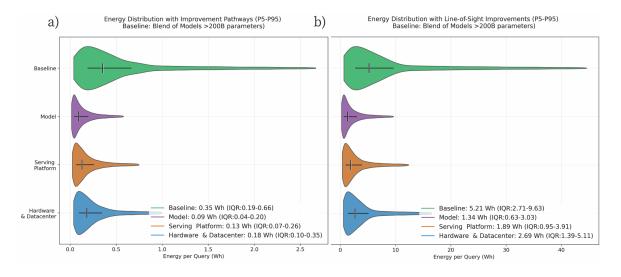


When setting Lin = 500, the distribution is very close (~1%) to our original estimate assuming Leff = Lout. Lin = 1000 increases the energy per query, but with a lower effect than increasing Lout. When Lin = 5000, the energy distribution increases but it does not become bimodal as in the traditional case, as prefill is not dominant.

IV. Effect of TPS and P_{node} proportionality

We explore the effect of increasing P_{node} with increasing TPS. For this we assume P_{node} increases linearly with TPS starting from the idle power $P_{idle} = 2.7 \text{ kW}$ to $0.9 \cdot P_{max}$

The following figure summarizes the effect on energy per query distributions. For the traditional regime (a) in the baseline scenario, the median energy per query increased only slightly compared to the original analysis with steady-state power as an independent variable (~3% increase). For the test-time scaling regime (b) the median energy per query increased by 20%, however this is a small effect compared to the effect of longer inference and model size.



V. Estimating scaling factor $\beta = 1.33$

We estimate the additional number of idle nodes based on a non-uniform usage profile. For this, we assume the daily sinusoidal load with range 50–100%, average 75%. We assume node peak power P_{max} = 11.3 kW, idle power P_{idle} = 2.7 kW, leading to dynamic power P_{dyn} = 8.6 kW

Each node at full load for 24 hours:

$$E_{\text{node}}^{\text{uniform}} = P_{max} \times 24 = 11.3 \times 24 = 271.2 \text{ kWh/}$$

Each node with sinusoidal load (mean utilization 0.75) consumes per day:

$$E_{\text{node}}^{\sin} = P_{\text{idle}} \times 24 + P_{\text{dyn}} \times 24 \times \overline{f} = 219.6 \text{ kWh/day}$$

So, the scaling factor for utilization is:

$$\beta = \frac{1}{f} \times \frac{E_{\text{node}}^{\sin}}{E_{\text{node}}^{\text{uniform}}} = \frac{1}{0.75} \times \frac{219.6}{271.2} = 1.08$$

 $Assuming \ 10\% \ node \ redundancy \ and \ 12\% \ additional \ energy \ consumed \ by \ InfiniBand \ interconnection, \ we \ get:$

$$\beta = 1.10 \times 1.08 \times 1.12 = 1.33$$