The Implementation Gap:

Inducing Variation in LLM Inference-time Energy Efficiency

for Fixed Computational Workloads

Henry Baker

Masters of Data Science for Public Policy

2025

Supervisor: Prof. Lynn Kaack

Word count: 7668

Abstract

As Large Language Models (LLMs) proliferate across digital services, inference-time energy consumption has become a key sustainability concern. Efforts to characterise energy efficiency based on analytical compute metrics such as FLOPs, neglect downstream system implementation decisions and deployment pipelines that critically determine empirical energy costs

This study conducts a grid search over 2,112 configurations of LLaMA-3.2–1B and 3B models to empirically assess the energy effects of five deployment-level parameters: batching, numerical precision, decoder sampling, tensor parallelism, and latency conditions - all evaluated under a fixed computational workload on a shared GPU cluster. These parameters interact with measures of throughput and device utilisation, indicating that energy efficiency is partially determined by a given system's position relative to its throughput–efficiency frontier.

The comprehensive grid search revealed substantial variability in energy efficiency, with the most extreme cases ranging by up to 516.5-fold for the 1B model and 293.5-fold for the 3B model, corresponding to coefficients of variation of 127.7% and 123.5%, respectively. While these extreme figures cannot be interpreted as directly achievable gains in real-world production, building in some further assumptions, this study tentatively suggests unconstrained deployments might unrealistically achieve energy costs 10%-15% of those plausibly expected from the same model during typical production.

These findings support calls for standardised, end-to-end benchmarking frameworks that extend beyond static model metrics to capture full-stack implementations. Such benchmarks would equip policymakers and practitioners with actionable insights for deploying LLM services both effectively and sustainably.

Contents

Abbreviations

AI	Artificial Intelligence
ML	Machine Learning
LLM	Large Language Model
NLP	Natural Language Processing
FLOP	Floating Point Operation
SLO	Service Level Objective

1 Problem Statement

The adoption of LLMs across varied digital services has led to increased scrutiny over their energy costs, as data-centre demand from AI workloads is projected to more than quadruple by 2030 (IEA, 2025) to account for 9.1-11.7% of total energy demand in the US by 2030 (Georgiou et al., 2022; Shehabi et al., 2024). Moreover, recent industry reports indicate that inference-time consumption now dominates AI energy usage. Google and Meta reported 60% and 70% of their respective AI-driven energy consumption was inference-related (Pasek et al., 2023; Patterson et al., 2021), as was 80-90% of Amazon Web Services' ML cloud compute (Amazon, 2019). Accurately estimating and reducing inference-time energy is vital to broader AI sustainability efforts.

Within the ML community, Strubell et al. (2019) first drew widespread academic attention to the substantial carbon costs of NLP model training, and Schwartz et al. (2020) proceeded to formalise *Green AI* as an efficiency-oriented alternative to the prevailing *Red AI* paradigm of performance at any cost. This framing catalysed a wave of research quantifying the environmental impacts of ML (García-Martín et al., 2019; Henderson et al., 2020; Lacoste et al., 2019; Patterson et al., 2021). Parallel efforts have sought to identify model-level drivers of energy consumption (Cai et al., 2017; Luccioni et al., 2024; Tripp et al., 2024), while systems-level research has expanded upon van Wynsberghe (2021)'s *Sustainable AI* framework, advocating for more holistic lifecycle-wide approaches to AI sustainability (Gupta et al., 2021; Kaack et al., 2022; Rolnick et al., 2023; Wu et al., 2025). Complementary to this, technical research has advanced energy efficiency of both training and inference through algorithmic, software, and hardware-level innovations focused on throughput and latency optimisations (Aminabadi et al., 2022; Chien et al., 2023; Dettmers et al., 2022; Kwon et al., 2023; Leviathan et al., 2023; Samsi et al., 2023; Stojkovic, Choukse, et al., 2024; Stojkovic, Zhang, Goiri, et al., 2024; Tripp et al., 2024).

Yet the practice of benchmarking LLM inference-time energy efficiency remains under-considered. Indeed, which empirical quantities make for practical, valid and robust measures of a model's energy consumption, and how to measure them, remains an open question. A common simplifying assumption underlying various *Green AI* efficiency metric proposals, is to take the number of FLOPs required per generated token as a proxy for inference-time energy costs (Henderson et al., 2020; Patterson et al., 2021; Schwartz et al., 2020). This assumption is closely aligned with parameter-counting heuristics that dominate model selection processes and continues to influence emerging AI policy discourse writ large (Luccioni et al., 2024).

As measures of theoretical computational complexity, FLOP-based metrics face growing criticism for their limitation in capturing empirical energy consumption (Desislavov et al., 2023; Fischer, Jakobs, & Morik, 2023; Gupta et al., 2021; Luccioni et al., 2024; Tripp et al., 2024). Specifically, FLOP counts are analytically computed as deterministic functions of model architecture and input-output characteristics (or number of complete forward passes required), often expressed as:

$$\text{FLOPs} = f(\text{number of parameters, input length, output length})$$

FLOPs quantify the number of arithmetic operations required to generate a given output, but they do not capture the energy efficiency with which those operations are executed. While correlated, FLOPs and inference-time energy consumption are conceptually distinct. The latter is a mediated empirical measure, shaped both by upstream (of FLOPs) development-phase choices, which define the static computation graph - such as model architecture and parameterisation - and downstream deployment-phase choices, which govern how efficiently that graph is executed in practice. Emerging evidence indicates that implementation-level factors can induce substantial variation in a deployment's energy consumption, even when FLOPs counts remain constant. Two systems with the same theoretical compute requirement may exhibit markedly different energy profiles. This makes raw FLOPs an incomplete and potentially misleading proxy for real-world energy costs.

Notably, recent initiatives such as Hugging Face's *AI Energy Score* Leaderboard ("AI is set to drive surging electricity demand from data centres while offering the potential to transform how the energy sector works - News", 2025) opt for direct empirical energy measures. Nevertheless, despite growing recognition of the limitations of theoretical proxies, FLOP-counts (and less frequently, multiply-accumulation operations (MAC) counts) remain widely used in both academic and industry contexts for estimating energy costs, due to their ease of computation from static model specifications. However, this research demonstrates that reporting FLOPs-per-token in isolation fails to capture the substantial range of empirical variation observable in inference-time energy consumption across different implementation configurations, nor provides information as to the underlying mechanisms driving this variance.

Moreover, FLOP-counting narrows policy attention to immutable model attributes, conceptually restricting the scope of intervention to the moment of model selection. This framing overlooks a wide array of downstream system-level implementation decisions and tradeoffs that shape the energy efficiency of real-world deployments (Gupta et al., 2021; Luccioni et al., 2024). From a benchmarking perspective, the neglect of implementation-level variation translates through to a lack of standardised test-time controls. This gap creates opportunities for motivated actors to present artificially efficient performance metrics by testing under unrealistic configurations, misaligned with production constraints. Ultimately, these dynamics leave, consumers and policymakers with little insight into the true energy cost of querying an LLM, or the trade-offs being made on their behalf (Gupta et al., 2021; Luccioni et al., 2024).

In this research, a subset of these implementation decisions are treated as tunable parameters to evaluate how deployment choices influence inference efficiency. The comprehensive grid search identifies substantial variability in energy efficiency - up to 516.5-fold for the LLaMA-3.2-1B model and 293.5-fold for the 3B model

- with corresponding coefficients of variation of 127.7% and 123.5%, respectively (see Table 2). While the explored parameter space includes many impractical and unrealistic configurations - meaning these extreme figures should not be interpreted as directly achievable gains in real-world production - the results underscore the importance of standardised benchmarking methods representative of real-world operational contexts.

2 Paper Positioning

2.1 Research Gap

Despite increasing attention to the environmental costs of AI, there remains a significant empirical gap in understanding the joint effects of deployment-phase decisions on LLM inference-time energy consumption. This research integrates previously fragmented technical insights into a coherent policy-oriented framework that highlights implications for standardising LLM energy benchmarking. In doing so, it exposes the cumulative - and occasionally interactive - energy effects of decisions made during system deployment.

By identifying parameters that influence inference-time energy efficiency independently of FLOPs, this study proposes a minimum initial set of controls for future energy benchmarking efforts. Within the defined search space it demonstrates the degree of variation in energy outcomes possible for a given FLOP-count; and by further simulating a series of implementation scenarios, it underscores the fragility of recent FLOP-based benchmarking proposals and the need for more granular policy attention to the energy trade-offs embedded in deployment decisions.

2.2 Research Question

This study investigates how inference-time energy efficiency varies with changes to a pre-identified set of readily-accessible deployment-time parameters, asking:

> *How do implementation decisions affect empirical energy consumption during LLM inference*?

Focusing on batch size, tensor parallelism, numerical precision, decoder sampling, and latency conditions, the study shows that even with fixed theoretical computational workloads, actual energy costs can vary substantially.

3 Methodology

This study presents an empirical analysis of LLM inference-time efficiency, holding computational workload constant (ensuring fixed FLOPs-per-token) and measuring energy-per-token under a range of deployment configurations. The configuration space is explored through a within-model grid search, following Fischer, Jakobs, and Morik (2023).

The full experimental setup is defined by a task configuration C, and an execution environment E, where:

$$C = T \times D \times M \qquad \text{(Task, Dataset, Model)}$$

$$E = S \times A \qquad \text{(Software, Hardware)}$$

The task (T) is held constant: a causal language modelling task with 500-token input and output sequences. Both are deterministically truncated to eliminate variation from sequence-length effects (which would constitute workload variation). The dataset (D) is a stratified subset of 128 prompts from Hugging Face's *AI Energy Score* benchmark corpus, sampled from WikiText (Wikipedia), OSCAR (web text), and UltraChat (conversational transcripts) (HuggingFace, 2024).

The model (M) includes both architecture and configuration. Two models are evaluated - LLaMA 3.2–1B and 3B - representing lightweight, open-weight LLMs. Within each model, a grid search is conducted over selected parameters: tensor parallelism (1-4 GPUs), batch size (1-64 prompts), numerical precision and quantisation (FP32, FP16, INT8, INT4), decoder strategy and stochasticity (greedy, vanilla-multinomial, Top-p, Top-k), and simulated latency conditions (including delay severity, and constant vs bursty patterns). This is a non-exhaustive list of available implementation variables, selected based on (i) accessibility to LLM service providers, and (ii) prior evidence of energy effect size.

$$C = T \times D \times M = 1 \times 1 \times (2 \times 1056)$$

The execution environment (E) - encompassing both hardware (A) and software (S) - is also held constant. All experiments were run on (A) a shared single node server equipped with 4x NVIDIA A100-PCIE-40GB GPUs and 128x AMD EPYC 7742 64-core CPUs; running (S) Ubuntu Linux 20.04, with PyTorch 2.5.1 as the computational backend, orchestrated using Accelerate 1.4.0 launchers. A detailed description of the Python measurement package developed for this research is included in Appendix: Developed Python Package.

$$E = S \times A = 1 \times 1$$

Thus an experiment X is the 5 tuple (T, D, M, S, A):

$$X = C \times E = 2112 \text{ configurations}$$

To mitigate environmental noise from operating on a shared server (e.g., fluctuating server load, thermal throttling), the full configuration space X was executed two times across non-contiguous experimental cycles over separate weeks. Each cycle entailed a full traversal of X, with execution times staggered across days and hours to reduce temporal confounding. Runs were stratified by model and randomly sequenced to minimise ordering effects.

Each run was initiated via a fresh command-line invocation of Hugging Face's Accelerate launcher to fully reinitialise the distributed environment. Three dummy forward passes were used to trigger lazy initialisations (discarded from measurement), and a teardown protocol followed each run to clear all distributed states (caches, weights, and process groups) to prevent cross-run contamination. Failed runs were logged and retried up to three times before exclusion.

Raw per-process metrics were logged at the batch level: energy consumption, power draw, GPU/CPU/RAM utilisation, and latency. These were aggregated to produce run-level metrics: total energy, average power, utilisation, memory allocation, tokens generated, and throughput. FLOPs were analytically estimated by run using a representative prompt, to validate constancy. Further derived metrics include: energy-per-token, FLOPs-per-token, tokens-per-joule, FLOPs-per-joule, token- and query-latency, and end-to-end throughput.

While some recent works instrument node-level wattmeters to isolate low-level hardware effects (Tripp et al., 2024), this study adopts a system-level perspective consistent with the benchmarking needs of LLM service providers and policymakers. As such, energy measurements were obtained via software profilers (CodeCarbon), reflecting the emerging standard in ML energy analysis. These tools offer ±10–15% accuracy (Anthony et al., 2020; Desislavov et al., 2023).

4 Related Work & Research Design

4.1 Batching

One well-documented lever for improving inference-time energy efficiency is batching strategy. Batching of input queries improves energy efficiency by amortising fixed overheads, parallelising memory and compute operations, and maximising utilisation of the GPU's streaming multiprocessors. This reduces idle time between operations and pushes the device closer to its thermal design power (TDP) (Yang et al., 2024).

Both empirical investigations (Mavromatis et al., 2024; Stojkovic, Choukse, et al., 2024; Yang et al., 2024) and more theoretically-oriented modeling efforts (Cai et al., 2017) consistently demonstrate that energy consumed per sample decreases as batch size increases - up to a saturation point beyond which efficiency gains plateau.

The dominant mechanism by which batching (and other implementation-level optimisations) affects energy efficiency is through improving system throughput - the rate at which output tokens are generated - relative to power draw.

$$\text{Throughput} = \frac{\text{Total output tokens}}{\text{Total inference time}}$$

The full extent of the interactive relationships between throughput, instantaneous power draw, device utilisation and energy-efficiency is complex, conditional, and sometimes ambiguously-signed (and out of scope of this analysis). However, with respect to batching, Yang et al. (2024) find that increasing batch size drives up average GPU power usage, but improves energy per input - as more work is completed in less time with fewer wasted cycles. Higher power corresponds to greater energy efficiency, conditional that it reflects reduced latency and higher throughput. The optimal batch size often lies just below the point of full GPU utilisation.

Batching is thus a necessary implementation control for FLOPs to be considered a meaningful indicator of energy-efficiency. Henderson et al. (2020) found FLOPs to be uncorrelated with energy when using a batch size of one, whereas later work

(Yang et al., 2024) highlighted that unbatched inference leaves the GPU under-utilised, distorting the FLOPs-energy relationship and biasing comparisons in favour of larger models.

In practice, batching is constrained by latency requirements. Smaller, energy-inefficient batches are required to meet SLOs in low-latency interactive applications, resulting in a tradeoff (Samsi et al., 2023). Production systems therefore use dynamic or continuous batching (temporally aggregating concurrent user requests) to increase utilisation without breaching latency budgets. Yet inefficient batching and request scheduling remain persistent sources of energy waste in deployed LLM services (Stojkovic, Choukse, et al., 2024; Stojkovic, Zhang, Goiri, et al., 2024; Stojkovic et al., 2025). A well-calibrated batching strategy must consider both the implemented model's throughput characteristics and the underlying hardware's utilisation limits. This trade-off between energy efficiency and latency performance is invisible to naive analytically-determined proxies.

In this study, I adopt fixed, statically defined batches to isolate the marginal effect of batch size and eliminate confounding from real-time scheduling variability. Batch sizes are systematically varied from single-request inference up to 64 inputs.

4.2 Numerical Precision & Quantisation Effects

Modern accelerators support lower-precision arithmetic modes (FP16, BF16, INT8, INT4). Cutting per-operation bit requirements reduces memory bandwidth demands and enables faster execution with lower power draw. As with batching, reducing numerical precision increases energy efficiency principally by increasing arithmetic throughput relative to power draw.

Narang et al. (2018) report that switching to mixed-precision training (FP16) yields substantial energy savings, with He et al. (2022) reporting efficiency improvements of up to 24.2%. Sinha et al. (2022) find that post-training quantisation to ultra-low-bit formats (INT8 and INT4) further reduced power draw by 20–60% and lowered thermal footprints by 2–20°C, while Wu et al. (2018) report energy savings of up to 5× against a full precision baseline. These improvements are conditional on underlying framework-level optimisations that more effectively map integer operations to hardware-accelerated kernels (Alizadeh & Castor, 2024).

However, reduced precision is not without trade-offs, with marginal energy gains from more aggressive quantisation diminishing, accompanied by greater risk of degraded model performance. Without careful calibration or quantisation-aware fine-tuning, ultra-low-precision deployment can impair key metrics such as perplexity, latency, and output reliability (Latotzke et al., 2022a; Yao et al., 2023). Once again, this trade-off between energy efficiency and acceptable performance degradation is invisible to naive analytically-determined efficiency metrics.

In this study, I evaluate four precision formats: FP32, FP16, INT8, and INT4. The floating-point models are executed using native support in PyTorch, while integer quantisation is applied post-training using the 'bitsandbytes' library on pre-trained Hugging Face model checkpoints.

NB: There is a lack of consensus on how to compute FLOPs for quantised models (Yao et al., 2023). Integer operations (IntOps) differ fundamentally from floating-point operations in both execution and representation, and quantisation may also alter the computation graph via kernel fusion or reordered memory access patterns. Therefore, interpreting IntOps as direct FLOP-equivalents is technically inaccurate. Nevertheless, to maintain comparability across configurations, this study treats FLOPs-per-token as constant for all precisions, and treats precision as an ordinal axis (from FP32 to INT4).

4.3 Decoder Sampling Strategy

While a (decoder-based) model's static computation graph determines the primary computational cost of inference, the choice of decoding strategy can subtly influence per-token energy costs. This dimension remains underexplored in the empirical literature, yet there are algorithmic grounds to expect small but non-negligible effects.

Production deployments typically choose between greedy decoding (argmax), beam search (parallel forward passes), and stochastic sampling (vanilla-multinomial, Top-k, Top-p). The latter sampling-based methods alter how the next token is selected from the model's output logits without changing the core forward pass; by contrast, beam search incurs multiple forward passes per generation step, scaling roughly linearly with beam width b. To maintain the design constraint of fixed FLOPs-per-token, beam search is excluded from this study, though it would be expected to exhibit the highest energy costs among common decoding strategies.[1]

In all decoding modes considered here, each token still requires one full forward pass. Observed variation in energy stems from lightweight differences in the post-logit sampling overheads: respectively sampling from either the k most probable tokens, or from the smallest set of tokens whose cumulative probability equals or exceeds p (nucleus-decoding). These post-processing differences represent second-order effects relative to core matrix-multiplication inference costs, nevertheless they do introduce variation in runtime and memory access patterns. By highlighting greater numbers of required arithmetic and memory-transfer operations for a given token count, algorithmic complexity offers an approximation for a theoretical understanding of the energy effects of decoding overheads. Greedy decoding has the lowest complexity in selecting the argmax logit in $O(V)$; vanilla-multinomial sampling also has $O(V)$ complexity but requires random number generation and more irregular access; Top-k decoding requires sorting and truncating the logits in $O(V + k \log k)$; and

[1]As shown by Poddar et al. (2025), the primary energy effect of stochastic decoding strategies arises indirectly, through increases in output length and generation time under variable stopping criteria. Similarly, Luccioni et al. (2024) find prompt content to have minimal impact on energy cost aside from through output length. However, to isolate per-token efficiency effects arising from intrinsic differences in decoder runtime and sampling overhead, and to preserve a fixed FLOPs-per-token basis for comparability, all outputs in this study are deterministically truncated to 500 tokens. This approach blocks (dominant) length-induced energy variation.

Top-p introduces cumulative probability computations and sorting in $O(V \log V)$, and is associated with higher memory access irregularity and cache churn (Holtzman et al., 2020).

Empirical evidence on the energy cost of these decoding strategies is sparse. In the only dedicated study located, Nik et al. (2025, preprint) report Top-k ($k = 50$) to have +10–15% additional energy costs per token (relative to greedy); Top-p ($p = 0.9$) to have +20–25%; and Beam search ($b = 5$) to evaluate to approximately 4× the energy costs of greedy. Beyond this, no peer-reviewed work has empirically measured per-token energy by decoding strategy, with most studies focusing instead on latency or FLOP-based cost approximations.

In this study, I conduct a hierarchical grid search over three decoder parameters: temperature, k, and p. To save resources I do not conduct a full grid search between decoder parameters with other variables, but take a fixed configuration point (given by Appendix: Baseline Config) implemented with greedy decoding (temperature ≈ 0), then for each temperature variation, I evaluate vanilla-multinomial sampling alongside all specified Top-k and Top-p configurations. Parameter bounds were selected to encompass and slightly exceed standard production defaults:

Decoder Parameter	**Evaluated** Range	**Production** Standard	**Reference**
Temperature	0 – 1.4	0.7 – 1.0	(Singh, 2023)
k	0 – 500	40 – 100	(Singh, 2023)
p	0.1 – 0.98	0.8 – 0.95	(OpenAI-Developer-Community, 2023)

Table 1: Decoder-parameter evaluation ranges and production standards.

4.4 Tensor Parallelism & GPU Resource Allocation

Serving large LLMs at scale necessitates distributed inference architectures, typically implemented through tensor parallelism, pipeline parallelism, and GPU multiplexing. Tensor and pipeline parallelism distribute the model's computation and memory footprint across multiple GPUs (either within- or across-nodes) while multiplexing executes multiple model instances concurrently on a shared device(s) Weng et al. (2022). The literature on distributed computing and data-centre systems engineering focuses on efficiently managing these processes through low-level runtime strategies - such as scheduling algorithms and dynamic sharding - that optimise throughput and memory contention, to scale inference across large heterogeneous GPU clusters (Samsi et al., 2023).

While the primary engineering objective of parallelism is to support 'efficient' serving (with respect to latency and scale) of large models under high demand, with regards to energy consumption it matters how computation is distributed across the system. These second-order energy implications are conditional: on the one hand, parallelism can increase throughput by distributing workload more evenly across devices; on the other, it can introduce communication and synchronisation overheads in the form of collective barriers, inter-device data transfers, and attention cache exchanges (Li et al., 2022). Parallelism's net energy impact depends on the system's position relative to its throughput-efficiency frontier (Yang et al., 2024).

Additional system-level characteristics such as memory access patterns, GPU-device partitioning, cache locality, and hardware heterogeneity, further influence distributed inference's energy profile (Tripp et al., 2024; Weng et al., 2022). To mitigate these overheads, modern data-centres employ granular GPU sharing and context-switching mechanisms, such as NVIDIA's Multi-Instance GPU (MIG) and Multi-Process Service (MPS) to improve device-level utilisation and lower energy costs (Li et al., 2022; Wilkins et al., 2024a). However, this study adopts a naïve tensor parallelism scheme for experimental tractability, whereby available GPUs are treated as a single homogeneous cluster, and the number of cores exposed to the distributed environment is systematically varied. This omits industry-standard inference frameworks (e.g. vLLM, DeepSpeed, TensorRT, ONNX) that provide advanced parallelism and fine-grained execution control.

4.5 Latency Constraints and Power Management

Latency management is critical for deployed LLMs, especially in SLO-bound interactive applications such as chatbots. Optimising a given system towards common deployment metrics such as Time-to-First-Token (TTFT), Time-Between-Tokens (TBT), and system-level throughput (tokens-per-second), typically involves making tradeoffs against energy efficient implementations.

Latency constraints arise from two distinct sources (computational/service) vs communication/wait delay). Decomposing latency management into constituent subproblems highlights their distinctive constraining dynamics, and the associated tradeoffs in reducing each.

Computational latency is inherent to the inference workload itself, reflecting limits in arithmetic throughput and memory bandwidth. Its reduction involves aggressive scheduling and smaller (energy-inefficient) batch sizes, driving GPUs towards alternating between bursts of high power usage and underutilisation - both energy inefficient states (Henderson et al., 2020). Strategies like power-capping and dynamic voltage-frequency scaling (DVFS; more common in CPUs than GPUs) can mitigate energy consumption under computational latency constraints, but typically at the cost of increased latency and reduced throughput (Kakolyris et al., 2024; Samsi et al., 2023; Stojkovic et al., 2025). Improvements here involve movements along a throughput-power pareto frontier.

Conversely, latency arising from non-computational/communication sources such as network congestion, scheduling jitter, burstiness, or temporarily clustered and uneven query distribution, represents inefficiencies external to computation (Chien et al., 2023; Stojkovic, Zhang, Goiri, et al., 2024). Mitigation strategies such as request coalescing (under SLO-aware queuing) and workload co-location, typically improve hardware utilisation, reduce idle power draw, and increase throughput and energy efficiency (Stojkovic, Zhang, Goiri, et al., 2024; Wilkins et al., 2024a). Improvements here effectively move deployments closer to the throughput-power efficiency frontier.

Ultimately the latency-energy relationship is governed by a system's position with respect to its throughput capacity. Deployed LLMs are typically throughput-efficient (i.e. on the frontier) where measured latency reflects computation bottlenecks. This typically means that in deployed settings, latency-reducing performance gains degrade energy-efficiency, introducing a further trade-off dimension between runtime- vs energy-efficiency, invisible to analytical efficiency metrics (Yang et al., 2024).

This study simulates communication/wait latency (whereas computational latency is inherent to the inference process across all experiments). A structured grid search pairs randomised per-request delays defined by specified min-max delay bounds, with five burst sizes (0–20 concurrent queries), and four inter-burst intervals (0–6 milliseconds). These are inserted into the forward pass using scripted delay mechanisms.

5 Analysis of Results

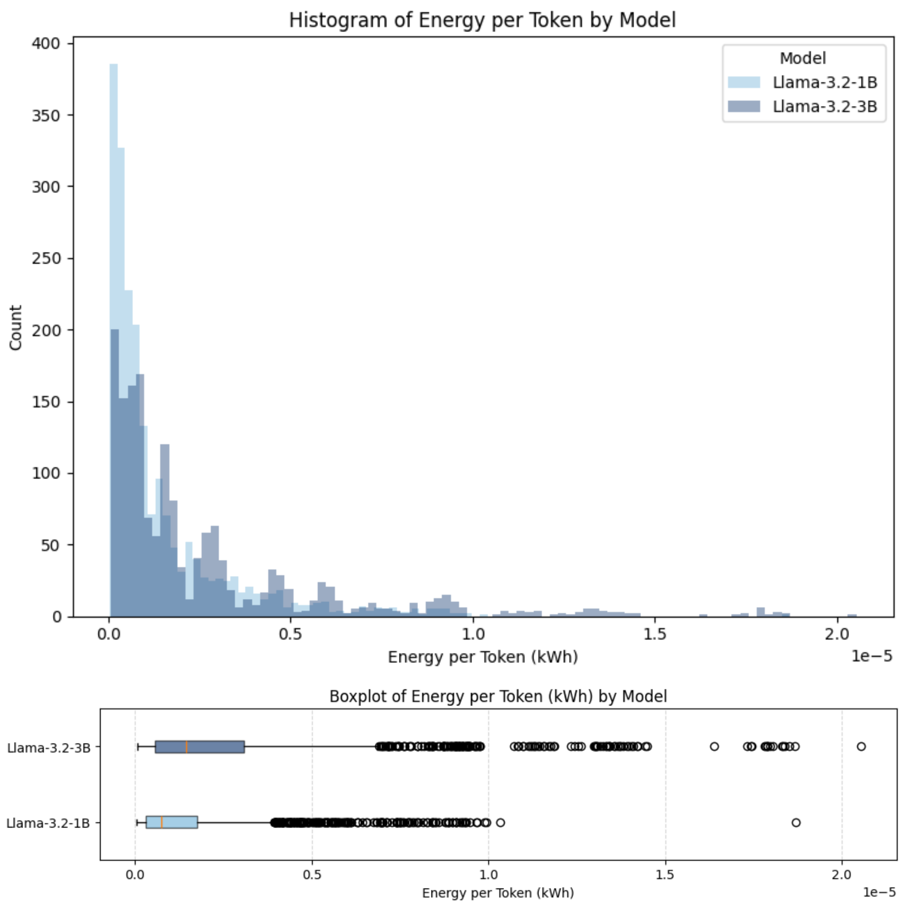


Figure 1: Distribution of grid search energy outcomes

Systematic variation of implementation parameters reveals substantial intra-model variability in inference-time energy efficiency under a fixed FLOPs constraint. As shown in Figure 1, both models exhibit positively skewed energy outcome distributions, with long right tails capturing high-energy outliers. While the two models show considerable overlap in their energy profiles - highlighting that deployment choices can outweigh architectural complexity - the 3B model incurs a higher median energy-per-token (reflecting its greater computational and memory demands) and exhibits larger absolute variance. However, when normalised by each model's mean, the 1B model's coefficient of variation is computed at 127.7%, compared with 123.5% for the 3B model. Likewise, while the 1B model's absolute maximum/minumum spread is almost 60% wider than the 3B, its clipped 95/5-percentile fold-change

Model	μ (kWh/token)	σ (kWh/token)	CV (%)
1B	1.44×10^{-6}	1.84×10^{-6}	127.7
3B	2.64×10^{-6}	3.26×10^{-6}	123.5

(a) Mean, standard deviation, and coefficient of variation.

Model	max/min fold	95/5 fold	% reduction	95/5 % reduction
1B	516.5×	61.0×	99.8%	98.4%
3B	293.5×	51.0×	99.7%	98.0%

(b) Fold-change and percent-reduction statistics.

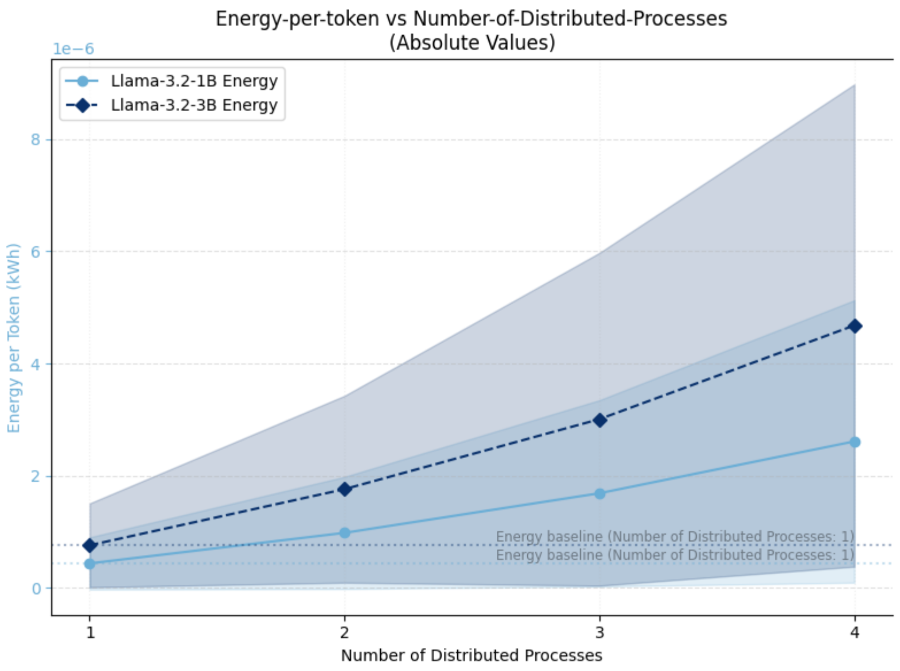
Table 2: Energy outcomes summary for 1B and 3B models: (a) core variability metrics; (b) fold-change and reduction metrics.

(61-fold, against 51-fold for the 3B model), indicates that in relative terms smaller models can be just as (if not more) sensitive to implementation-level decisions. (See Table 2 for full details.)

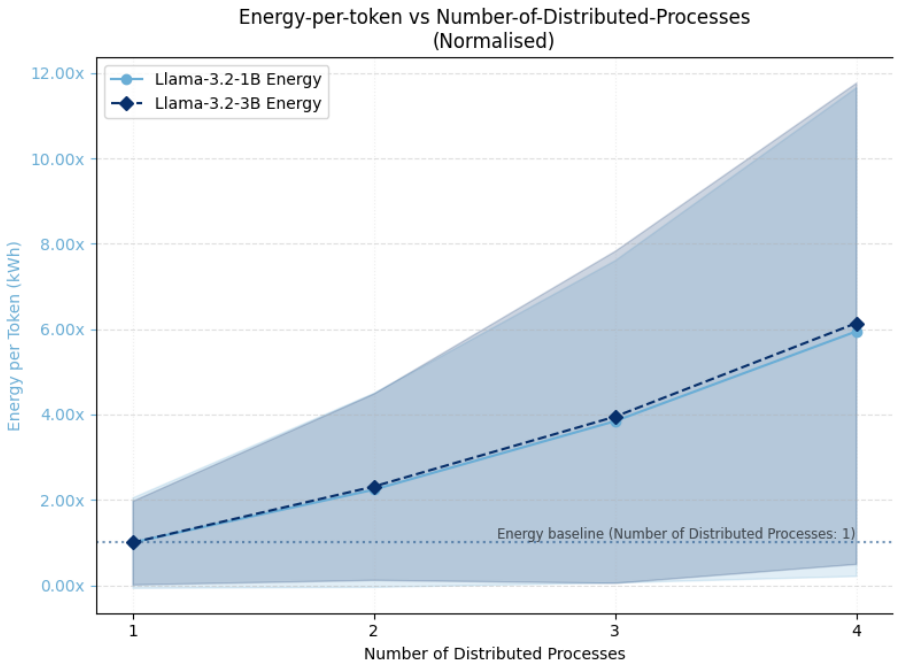
Note: For the following plots visualising variation associated with each implementation parameter, one-standard-deviation intervals are computed from the full grid-search results. These intervals are calculated across repeated runs, capturing both (minor) measurement noise, and the total (major) observed variation in energy-per-token across all configurations at that parameter value (including interactions with other settings).

5.0.1 Tensor Parallelism

As shown in Figure 2, increasing the number of processes over which model layers are distributed leads to a moderately super-linear increase in energy-per-token. Degree of tensor distribution has the largest effect of all tested parameter, with both models more than doubling with the addition of a single extra process, then scaling up to 4x and 6x under 3 and 4 processes respectively. Both model's variance in energy outcomes across the entire grid search also grows with parallelism, reflecting both



(a) Absolute values



(b) Normalised

Figure 2: Tensor parallelism (number of processes) vs energy outcomes

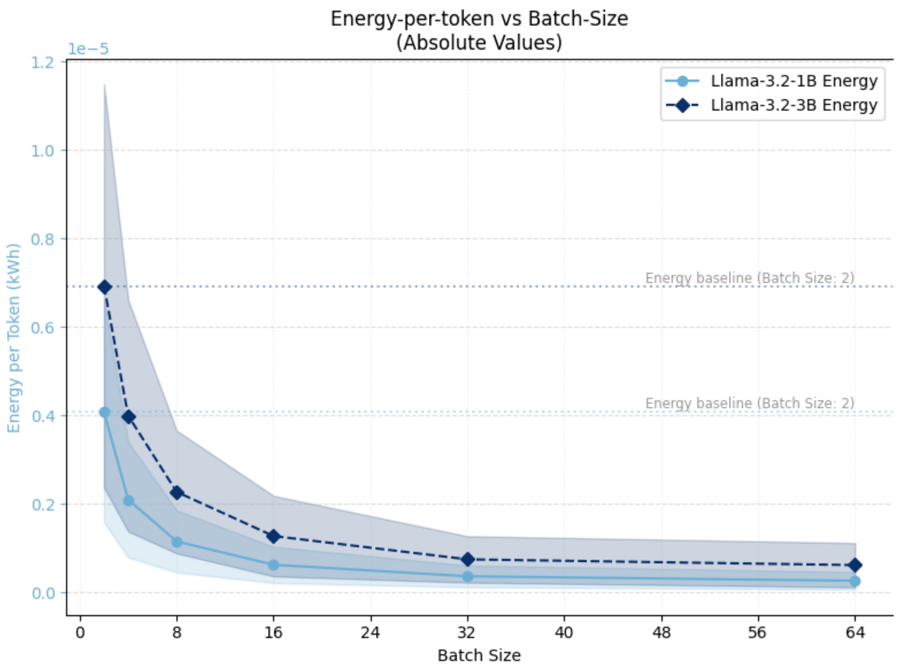
scale and increased instability and interaction effects under communication-heavy, fragmented scheduling.

This behaviour reflects a naive unoptimised implementation of tensor parallelism and is consistent with known limitations when model size is small relative to available GPU capacity. As (underutilised) GPUs are added, not only does another device need powering, but additionally energy overheads from inter-device communication and coordination increase disproportionately to realised throughput gains - hence the moderately super-linear pattern. This is supported by Figures 12e & 27, which shows no meaningful increase in throughput as parallelism increases, and further confirmed in the moderating effect of increasing batch size on parallelism (Figure 4a) - which by increasing throughput and improving device utilisation partially offset these efficiency losses.

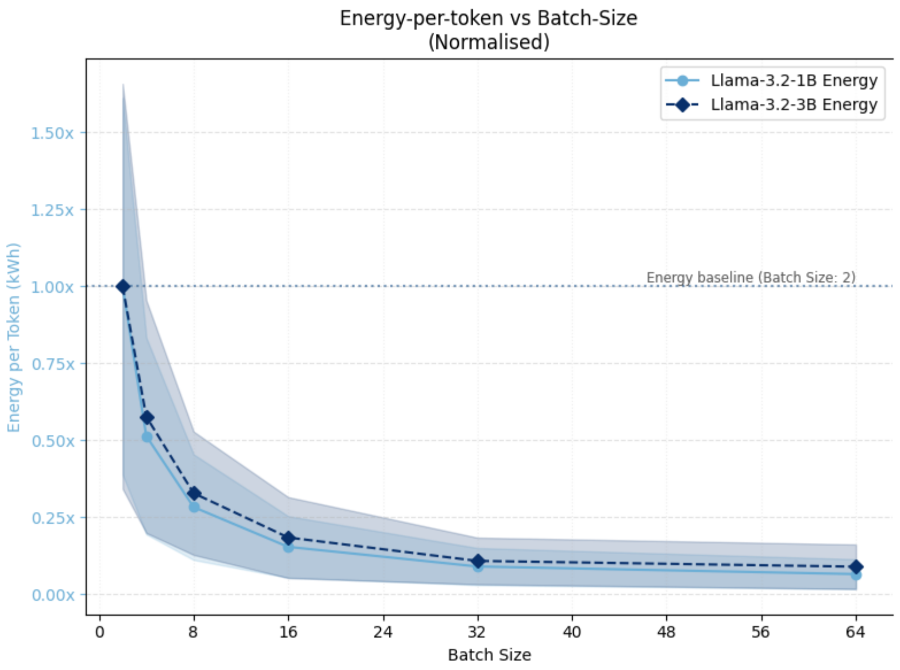
Leveraging more sophisticated parallel execution strategies would likely alter the scaling dynamics observed in these experiments, flattening the energy-efficiency curves and shifting the throughput-efficiency frontier outward.

5.0.2 Batch Size

Figure 3 reveals a steep inverse relationship between batch size and energy-per-token: in the small-batch regime, fixed overheads dominate, while beyond a certain point the curve plateaus, reflecting compute and memory-bandwidth saturation. The lower bound is reached for both models at around 10% of the double-batch baseline (so a range of 10x, excluding input single batches). Likewise, Figures 12a and 25 show energy-per-token falling as throughput rises, demonstrating that batching-driven efficiency gains stem primarily from throughput improvements. This throughput-sensitivity is further evidenced by the moderation effect of tensor parallelism (here a proxy for per-device utilisation) in Figure 4.

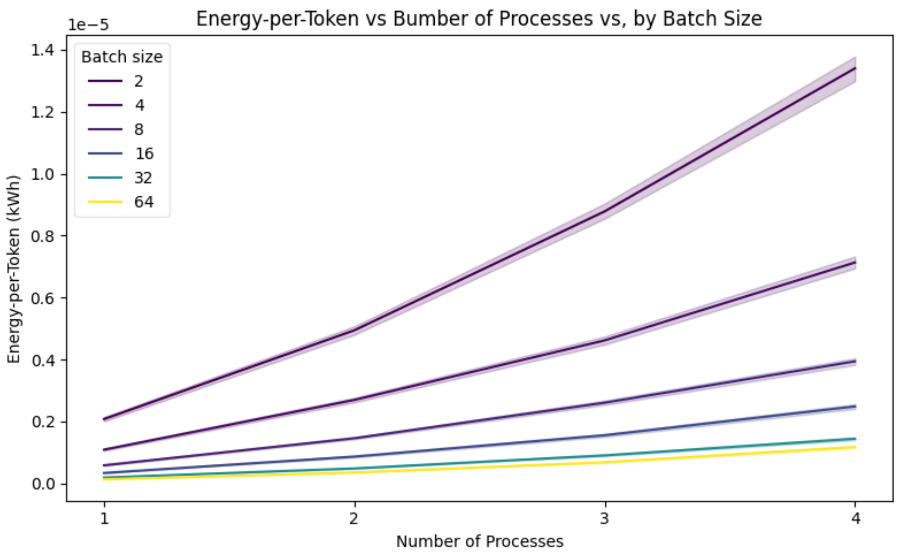


(a) Absolute values

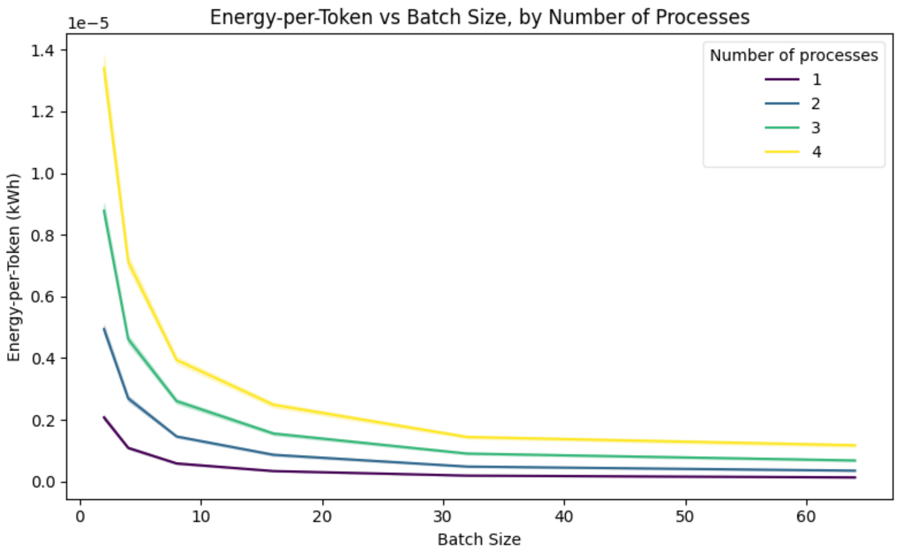


(b) Normalised

Figure 3: Batch size vs energy outcomes



(a) Batch size → number of processes



(b) Number of processes → batch size

Figure 4: Interaction effects between degree of tensor parallelisation and batch size

5.0.3 Numerical Precision

Energy-per-token does not decrease monotonically with precision reductions (Figure 5). Transitioning from FP32 to mixed FP16 yields energy efficiency gains of 35-40% for both models, consistent with findings by Narang et al. (2018) and He et al. (2022). However, the efficiency losses of 5-10% across the quantised models suggest that realised energy savings from quantisation is conditional on backend-level integration (Alizadeh & Castor, 2024; Yang et al., 2024).

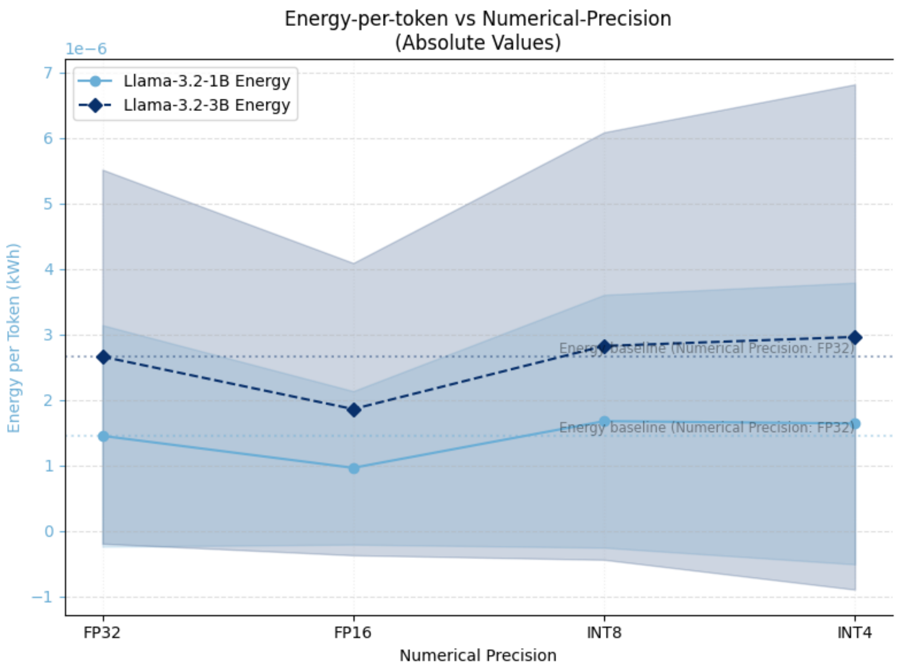
5.0.4 Decoding Strategy

Within-mode variation yielded little of substantive interest, with no discernible pattern from varying values of k (Figure 6) and p (Figure 7) across all temperature levels. Larger, more legible plots are included in Appendix: Decoder Plots.

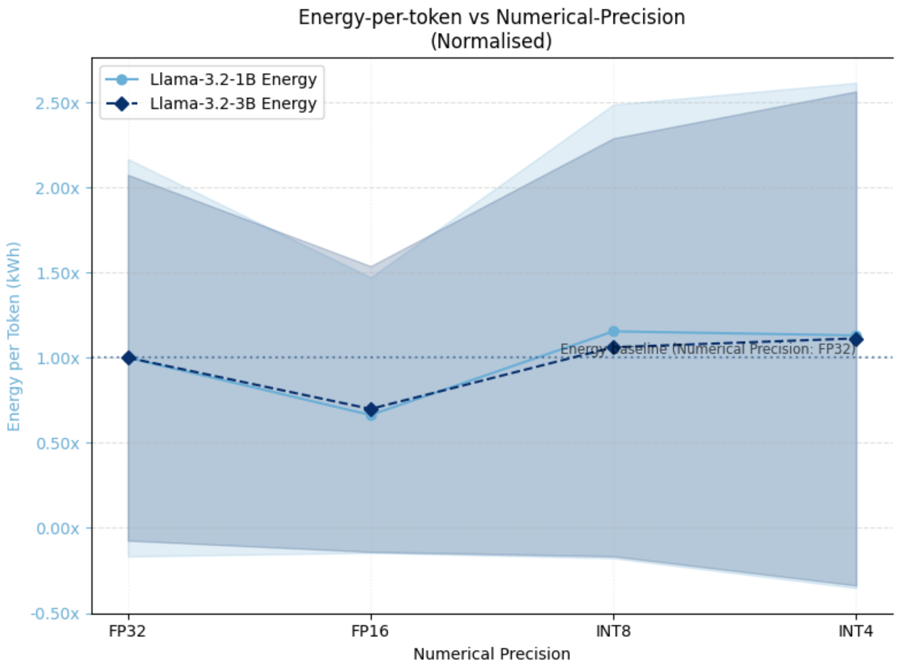
Between-mode variation exhibited clear, albeit minor, differences - with all curves remaining within 5% of the baseline (Figure 8). Greedy decoding - at temperature 0, where all modes converge by definition - exhibited lowest energy-per-token costs, except for certain vanilla-multinomial sampling values in the 3B model (likely an artifact; of little substantive interest). For all sampling strategies, there was no obvious trend across temperature values after the initial dispersion from the greedy baseline; of the stochastic strategies, vanilla-multinomial sampling was most efficient, followed by Top-k, then Top-p - consistent with algorithmic expectations.

5.0.5 Simulated Latency Conditions

Figures 9 and 10 show that energy efficiency steadily degrades as simulated latency increases, with larger declines under bursty conditions - albeit with modest overall effect sizes. Because this latency models exogenous communication delays (network congestion or scheduling jitter) rather than compute bottlenecks, the efficiency loss reflects reduced throughput: GPUs remain powered longer but perform less useful work per unit time. In the 1B model, this decline is linear and monotonic (Figures

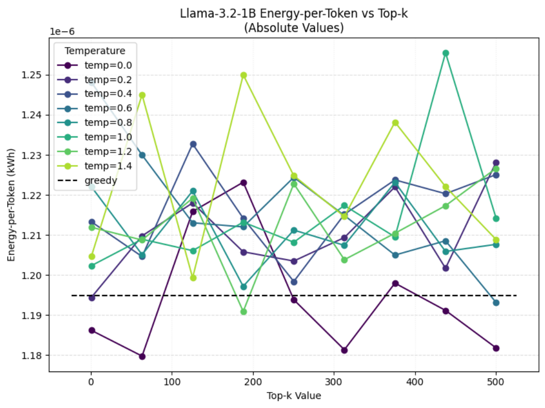


(a) Absolute values

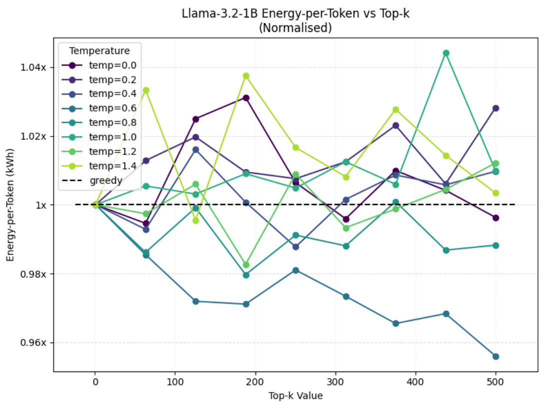


(b) Normalised

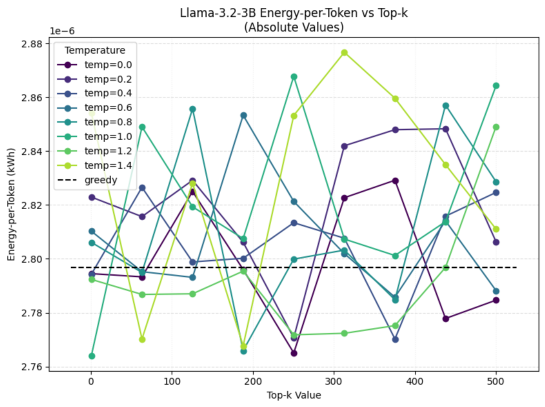
Figure 5: Numerical precision vs energy outcomes



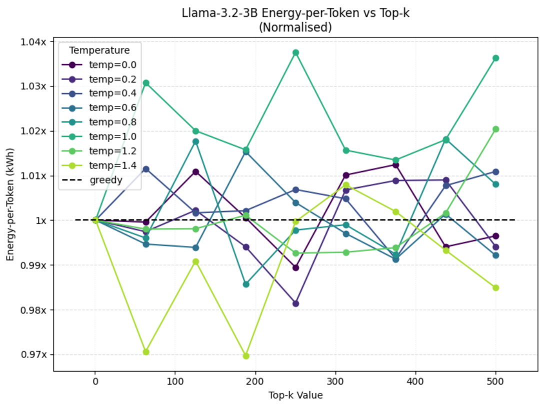
(a) 1B (absolute)



(b) 1B (normalised)

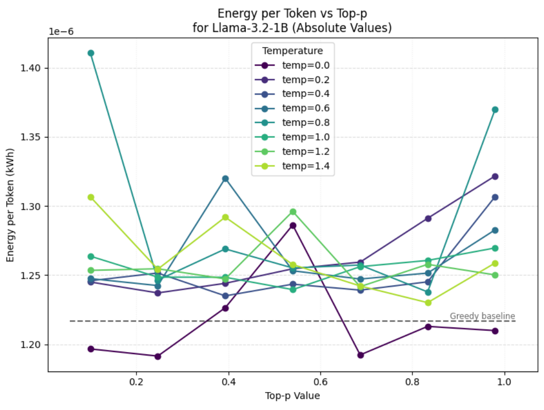


(c) 3B (absolute)

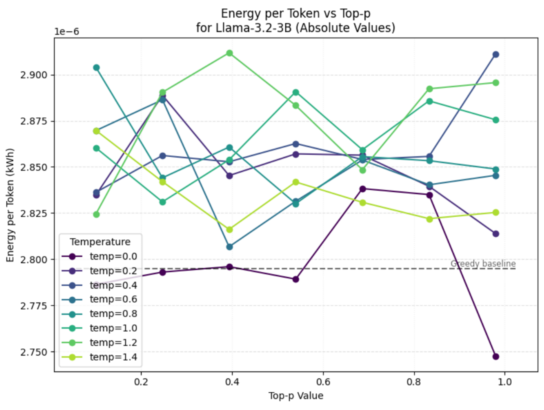


(d) 3B (normalised)

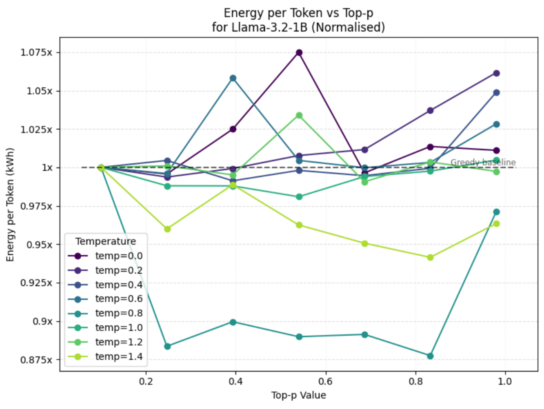
Figure 6: Top–k within-mode energy outcomes



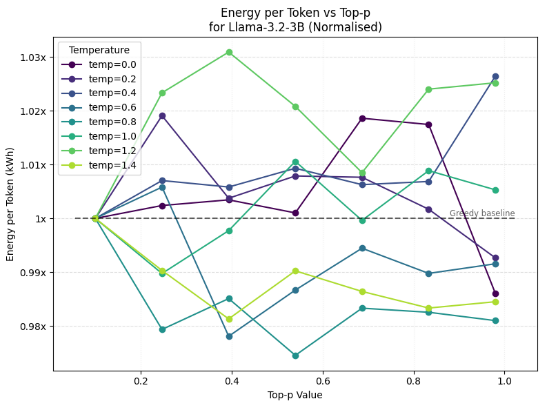
(a) 1B (absolute)



(b) 3B (absolute)

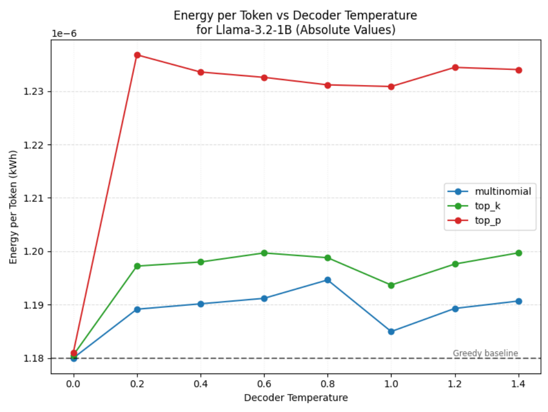


(c) 1B (normalised)

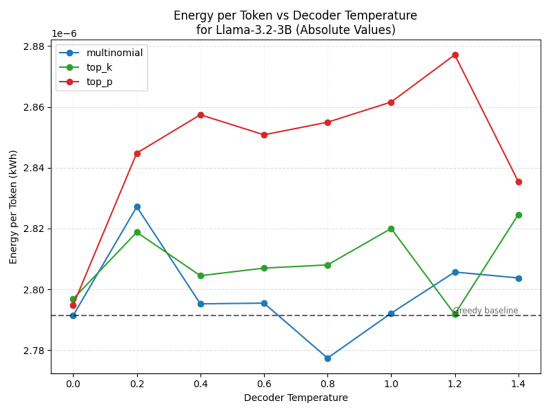


(d) 3B (normalised)

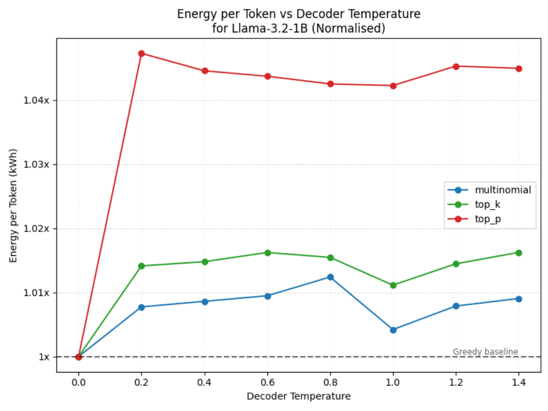
Figure 7: Top-p within-mode energy outcomes



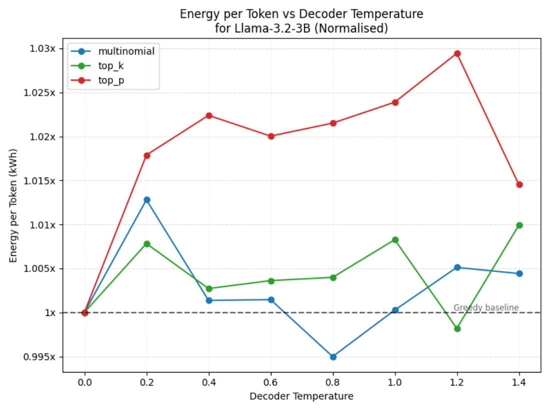
(a) 1B absolute



(b) 3B absolute



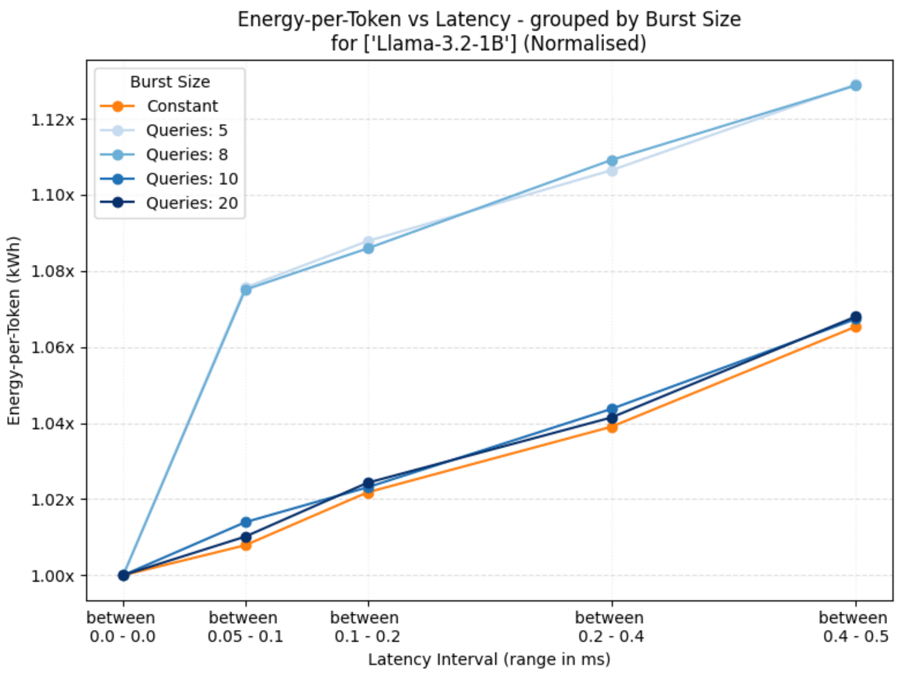
(c) 1B normalised



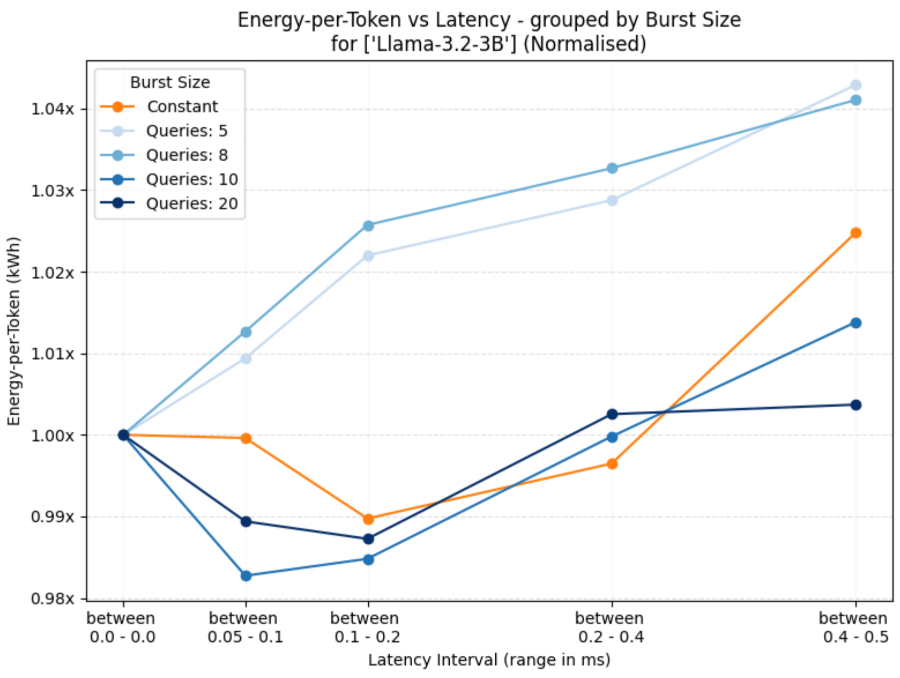
(d) 3B normalised

Figure 8: Comparison of between decoder-mode energy outcomes.

9a and 10a); in the 3B model (Figures 14b and 10b), the curve is more irregular, likely due to more interactions among device utilisation, memory access patterns, and synchronisation overhead at higher compute intensity.

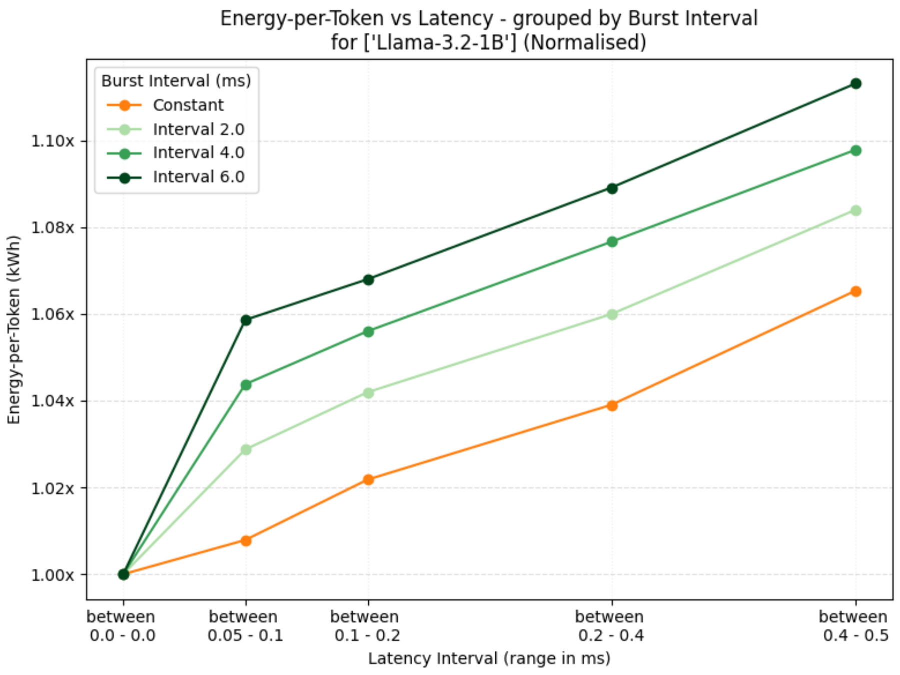


(a) 1B

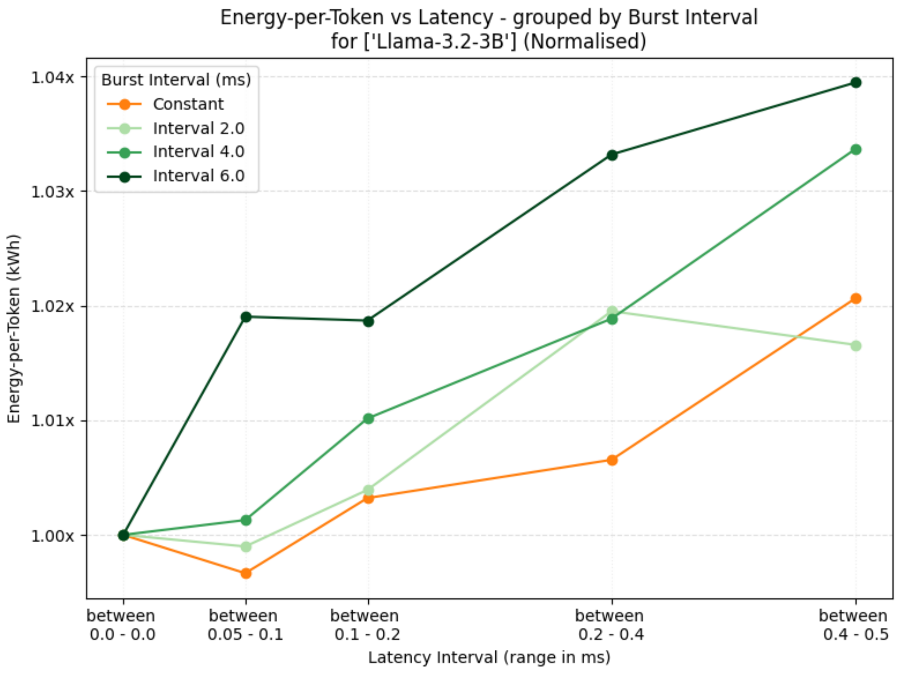


(b) 3B

Figure 9: Delay interval vs energy outcomes, by burst size.



(a) 1B



(b) 3B

Figure 10: Delay interval vs energy outcomes, by burst interval.

Burstiness further exacerbates energy and throughput inefficiency. Longer burst intervals degrade throughput and increase energy cost (Figure 10), however the relationship between burst structure and energy use is not uniform. On this model-hardware setup, smaller burst sizes (5 or 8 queries) exhibit consistently higher inefficiencies, whereas larger bursts (10 or 20 queries) resemble constant latency efficiency profiles (Figure 9). In line with findings by Stojkovic, Choukse, et al. (2024) and Stojkovic, Zhang, Goiri, et al. (2024), this suggests modern GPUs exploit burst-locality to regain uninterrupted stretches of computation, effectively treating larger bursts as pseudo-batches.

5.0.6 Throughput & Device Utilisation

Prior work has considered the role of throughput and device utilisation rates in determining inference-time energy efficiency (Fischer, Jakobs, & Morik, 2023; Samsi et al., 2023; Yang et al., 2024). Routinely monitored in production environments, these metrics serve as system- and device-level indicators of runtime efficiency, capturing how effectively compute is translated into useful output. This study suggests extending that logic, proposing that together they capture computational and non-computational aspects of energy efficiency in deployed LLM systems.

A system's overall throughput capacity is a function of model architecture (e.g., layer depth, kernel operations, memory access patterns), hardware constraints, and implementation decisions (Fischer, Jakobs, & Morik, 2023). Within this capacity, the observed throughput rate characterises the *realised* productivity of a system. The distance between the two governs key energy-performance trade-offs for the system as a whole.

In contrast, device utilisation indicates how closely specific hardware modules are operating to their respective performance ceilings - offering practical diagnostics of system bottlenecks.

While this study does not formalise the precise functional form of the interac-

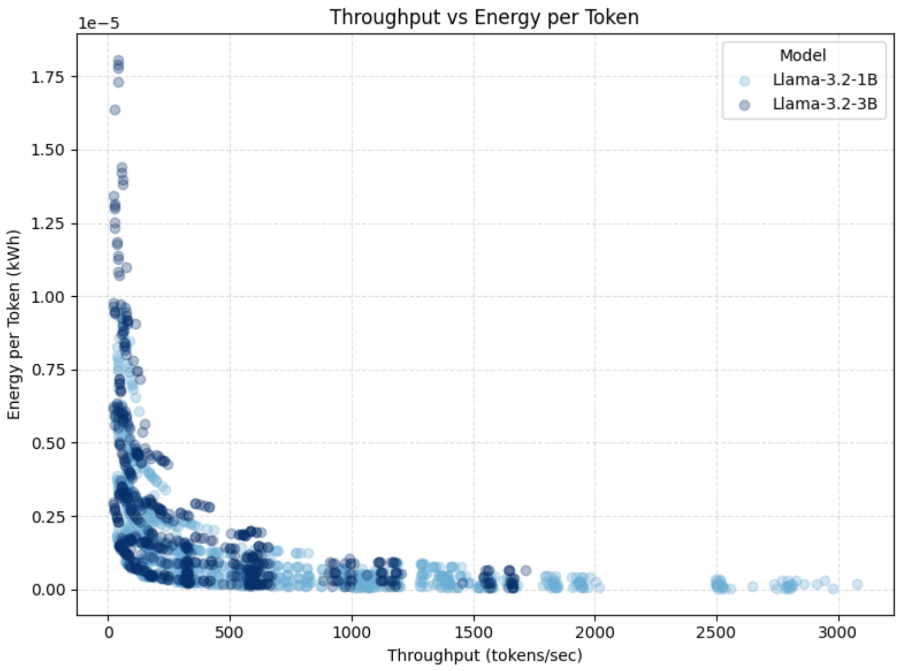
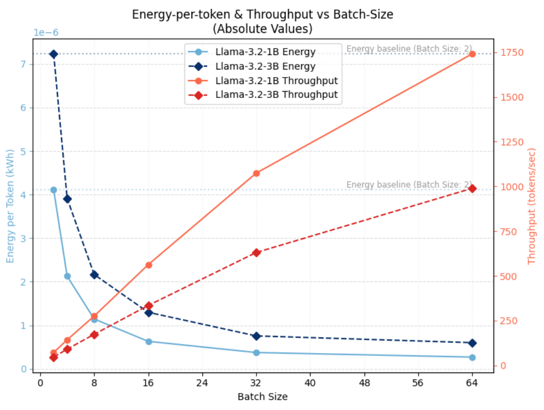


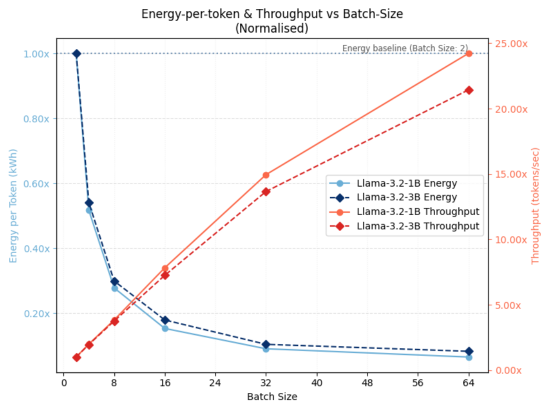
Figure 11: Throughput vs energy outcome

tions between device utilisation, power draw, system throughput capacity, throughput rate, and energy costs (an area of open research), Figure 11 illustrates that for systems characterised by throughput-inefficiency (operating at low throughput, within their throughput-power pareto frontier) small increases yield large reductions in energy-per-token. Beyond an inflection point — indicating proximity to the system's throughput-efficiency frontier — further throughput gains no longer improve energy efficiency, but instead require proportionate increases in power draw. This convex non-linearity allows the optimal throughput configuration set to be conceptualised as a constrained optimisation problem under SLO conditions (Samsi et al., 2023; Stojkovic, Zhang, Goiri, et al., 2024).

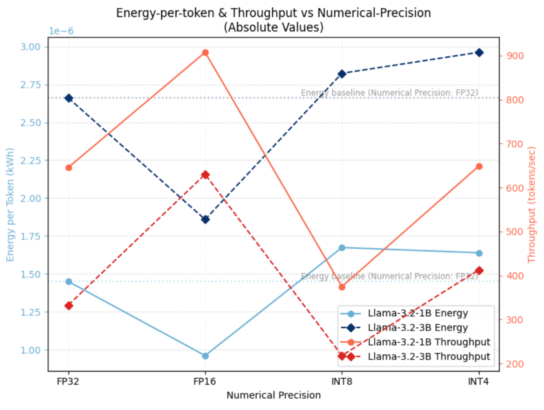
In the throughput-inefficient regime (below the frontier), configurations that increase throughput - such as larger batch sizes (Figures 12a, 25) or lower precision (Figures 12c, 26) - yield substantial energy-per-token improvements. By contrast, adding more GPUs (Figures 12e, 27) shows little benefit until the system is already operating near peak utilisation. Optimising system implementation and scaling carry throughput-sensitive dynamics.



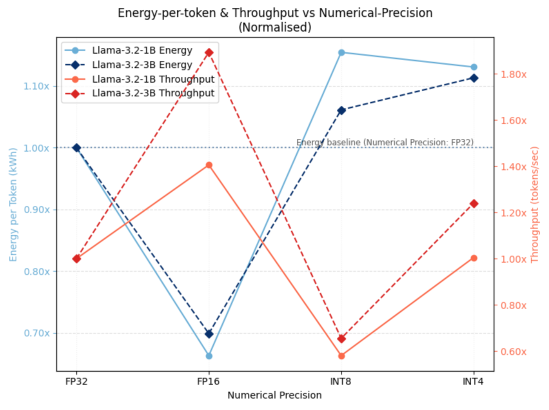
(a) Batch size (absolute)



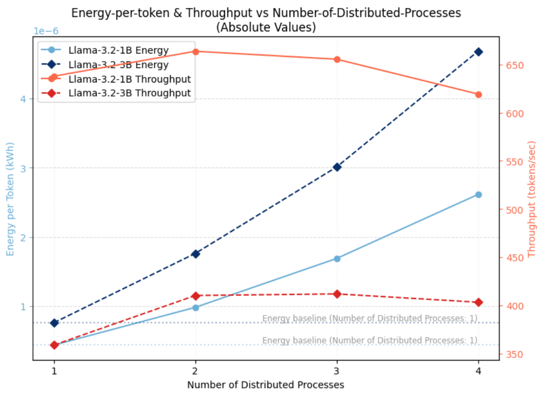
(b) Batch size (normalised)



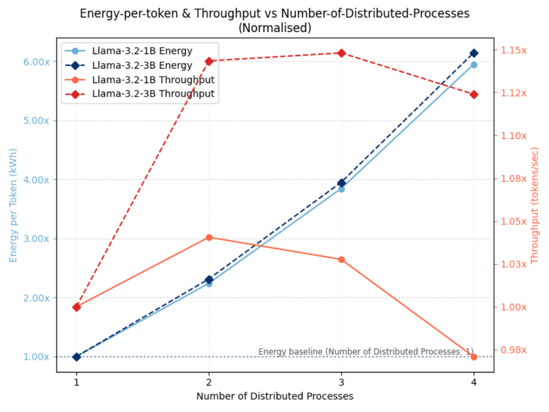
(c) Precision (absolute)



(d) Precision (normalised)

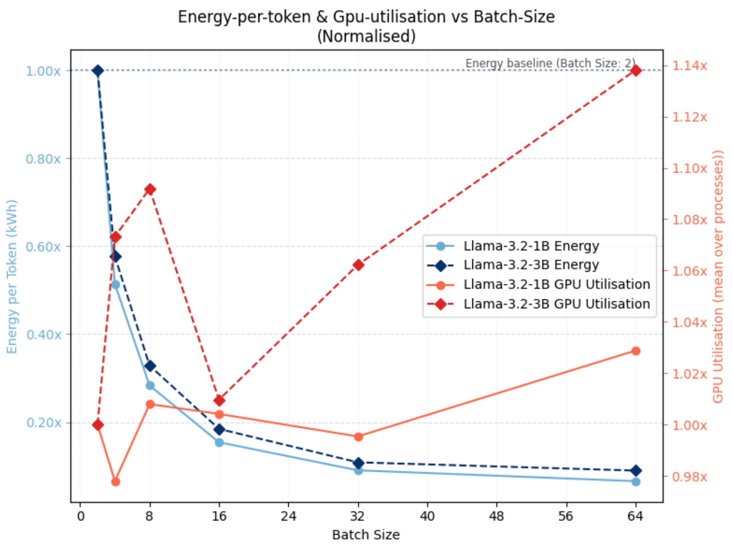


(e) Num. of procs (absolute)

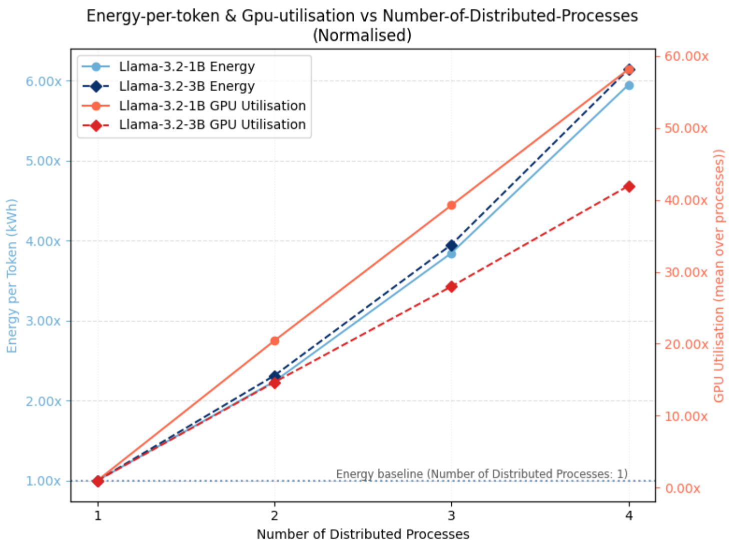


(f) Num. of procs (normalised)

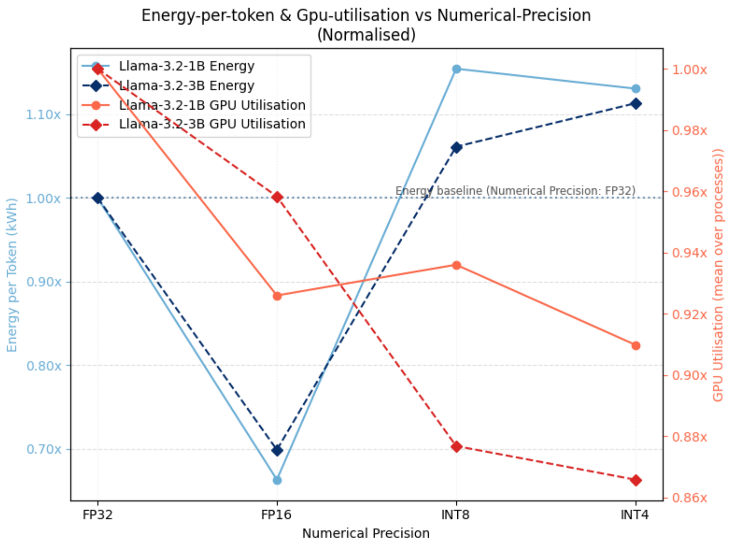
Figure 12: y_1-energy-per-token; y_2-throughput, for three implementation parameters. The changing relationship between energy and throughput in each reflects the changing trade-offs as the system transitions from a throughput-inefficient to a throughput-efficient regime. Larger plots in Appendix: Throughput Plots.



(a) Batch size



(b) Tensor parallelism



(c) Numerical precision

Figure 13: y_1-energy outcomes; y_2-device-utilisation, vs three implementation parameters.

By contrast, Figure 13 demonstrates that per-device utilisation captures a different set of non-computational inefficiencies that pull performance inside the efficiency frontier. In efficient deployments, utilisation should remain consistently high; persistent underutilisation indicates specific sources of energy waste - as is often the case in deployed systems where utilisation rates have been reported as low as 23–27% (Samsi et al., 2023).

The utilisation data collected here - measured at the device-level in a shared server environment - are noisy and reflect aggregate activity across concurrent users, so over-interpretation is avoided. Nevertheless, it is of note that the negatively-signed relationship between utilisation and increased numerical precision in Figure 13c is surprisingly consistent - even when energy and throughput increased under quantisation - reinforcing that realised quantisation-related energy savings are conditional on backend integration. Figure 13a's lack of a clear utilisation-energy relationship in the low batch regime (where communication bottlenecks dominate) also reinforces that high utilisation is a practical diagnostic and a necessary-but-not-sufficient condition for efficient inference.

From a policy perspective, throughput and device utilisation offer more insightful characterisation of runtime energy dynamics than analytical metrics such as FLOPs, and a deployment-sensitive basis for benchmarking deployed systems. These metrics are routinely logged in production, integrate both architectural and operational factors, and support systematic comparison across LLM deployments. Moreover, better understanding of the topology of a system's throughput and its determinants (an area of open research), supports *Green AI*-aligned technical research.

6 Discussion of Results

6.1 Implementation Matters

Deployment decisions substantially affect inference-time energy outcomes - to an extent under-acknowledged in *sustainable AI* discourse. This study shows that variation induced by implementation *can* well exceed that attributable to model size alone.[2] As such, LLM energy efficiency should be conceptualised not at the level of abstract model architectures, but as a property of implemented systems shaped by concrete deployment choices.

Across the full grid search, at the most extreme observed energy-per-token variation reached 516.5-fold for the LLaMa-3.2-1B model and 293.5-fold of the 3B model, reducing to 61-fold and 51-fold when considering only the 5th–95th percentiles - with coefficients of variation of 127.7% and 123.5% (see Figure 1 and Table 2). However, these raw statistics include implementations associated with configurations unrealistic for practical deployment. To address this I further simulate six plausible deployment scenarios reflecting distinct real-world constraints (details in Appendix 17).[3] Within these scenarios, energy efficiency outcomes ranged by 4.3-fold (CV: 61.35%) for the 1B model, and 4.9-fold (CV: 58%) for the 3B model. Table 3 further contextualises this variation at the 3B scale by comparing energy-per-query against familiar consumer electronics.

[2]Although considerable external validity concerns emphasise the *'can'* element of this statement, and more robust statistical analysis is needed to delineate the conditional interaction effects.

[3]I acknowledge these statistics are *illustrative*, not definitive: what constitutes a 'plausible deployment scenario' is multifaceted and confounded in ways my limited-dimensionality configurations cannot capture. More robustly, representative weighting of the grid search results could map the experimental findings to real-world deployment distributions - here, out of scope, so these rough-and-ready figures will suffice for now.

		Number of equivalent 300-token responses	
Appliance	**Charge**	**Least-efficient implementation**	**Most-efficient implementation**
iPhone 16	Full charge	7	19
MacBook Pro (M3 2023)	Full charge	40	116
WiFi Router	24 hr	64	186
HD Streaming	1 hr	35	103
Google Search	1 query	0.13	0.39
Electric Kettle	1 litre boil	129	67
Electric Shower	10 mins	663	1934

Table 3: Number of responses to match the energy consumption of a given appliance (3B model scale). Details deriving these calculations provided in Appendix: Energy Appliance Calculations. For illustrative purposes only, non-definitive.

6.2 Parameter Effects & Trade-offs

While the precise mechanisms driving inference-time energy consumption are complex, this study identifies key tunable parameters: batching, tensor parallelism, and numerical precision. Even amongst this initial set, interaction effects and performance trade-offs underscore the need for joint optimisation frameworks and throughput-aware scaling. By way of example, aggressive quantisation and deterministic decoding degrade generation quality (Yang et al., 2024), while larger batch sizes reduce responsiveness of real-time applications but improve efficiency (Figure 14a) - more so for highly-distributed inference environments. Conversely, decoding strategy has minimal direct energy implications, suggesting LLM providers can select sampling strategies on application-specific criteria rather than efficiency concerns.

6.3 Implications for Energy Benchmarking Standards

Prevailing conceptualisations of LLM energy efficiency emphasise analytical over empirical measures, and model attributes over system dynamics. Early benchmarking frameworks neglect the role of implementation-level factors as necessary controls

(Fischer, Jakobs, & Morik, 2023). Consistent with recent recommendations (Desislavov et al., 2023; S. Luccioni, 2024), this study highlights the need for direct empirical energy measurements and standardised reporting of key deployment configurations to ensure the validity and robustness of energy benchmarking practices.

By way of an example to illustrate the extent of possible distortion by motivated actors at test-time, I additionally simulate an over-optimised deployment, designed to maximise throughput and minimise energy consumption without regard for practical deployment constraints (details in Appendix 17). Given the strong assumption that the other configurations in this appendix reasonably represent 'plausible deployments', comparing this idealised configuration to the production-like set mean tentatively indicates that unconstrained optimising for benchmark test-time can yield energy costs around 10.69% (1B) and 13.25% (3B) of those expected from the same model during production.[4]

Finally, to assess whether *ceteris paribus* FLOP-based metrics remain informative for between-model energy-efficiency comparisons (if not for whole-of-system benchmarking), the FLOPs-energy relationship conditional on identified controls, warrants further investigation. Absent such validation, energy benchmarking frameworks would benefit from normalisation with respect to throughput, a metric already routinely monitored in industry for SLO compliance, that captures both model-intrinsic computational characteristics and extrinsic implementation decisions - and when jointly considered alongside device-utilisation, offers comprehensive insight into the inference-time efficiency of a deployed system.

[4]As with the illustrative figures above, these results lack rigour and are intended to demonstrate that benchmarking outcomes *can* be systematically distorted in the absence of standardised implementation controls. The magnitude of these effects is likely overstated due to the validity limitations discussed below.

7 Threats to Validity

7.1 Internal Validity

The naively implemented optimisation strategies, and student-user restrictions around on-device shard-loading prevented full exploitation of GPUs to saturation. This likely overstates the negative impact of tensor-parallel scaling and understates batch size effects on energy efficiency. Similarly, quantisation's effects are negatively signed, contradicting prior findings and suggesting a need for quantisation-aware deployment before evaluation (Sinha et al., 2022; Wu et al., 2018)

All experiments were conducted on a shared GPU cluster, where concurrent user workloads introduced uncontrolled variability in runtime, device utilisation, and system-level power draw. While temporal stratification, randomisation, repeated runs, and aggregation were employed to mitigate these effects, such measures offer only partial control over environmental noise. Moreover, the returned utilisation metrics do not isolate resource usage attributable to my experimental workloads, but instead reflect aggregate usage across all active processes on shared hardware. This lack of user-specific identification likely contributes to the anomalous patterns observed in the plotted relationships in Figure 13.

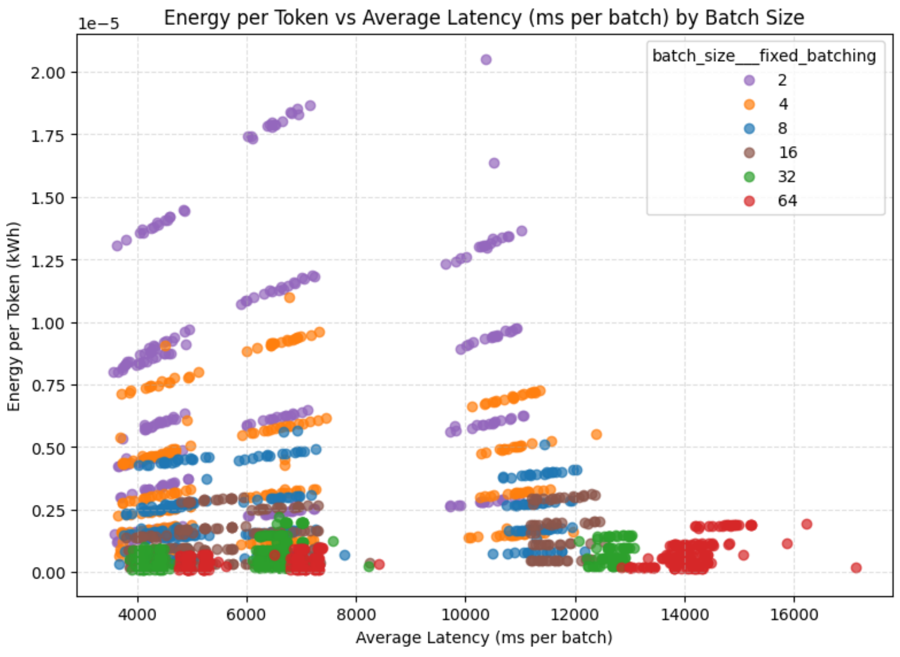
7.2 External Validity

The generalisability of reported findings are severely constrained.

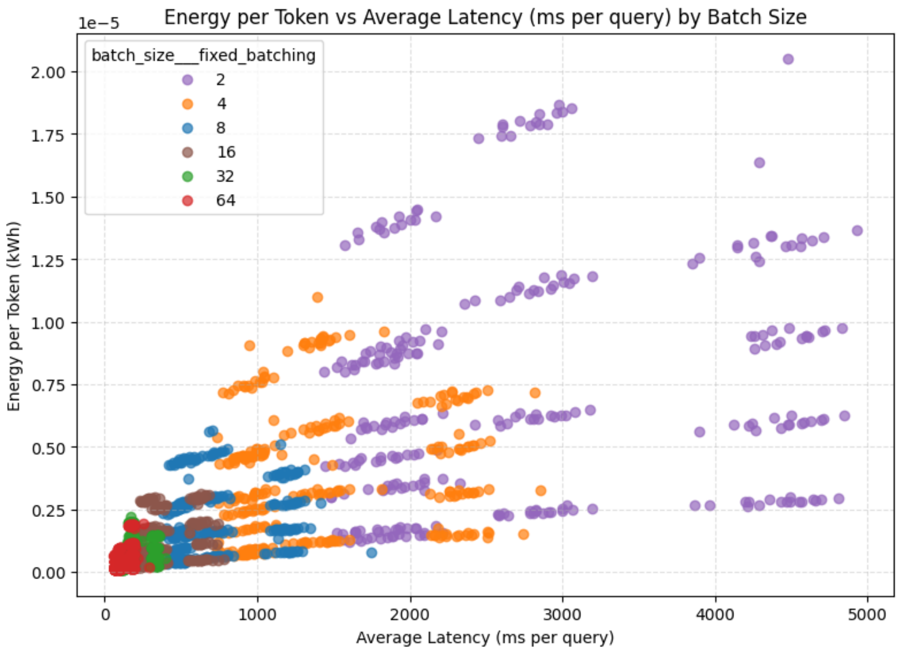
This study evaluates only two relatively small, open-weight models on a four-GPUs A100 cluster, without inclusion of industry-standard optimisation techniques. These configurations do not represent API-gated frontier-scale deployments typical of commercial LLM services. For comparison, GPT-4 reportedly operates 8 'experts' (component models) each of 220B parameters (totalling $\approx$1.76T parameters) (Howarth, 2024), while newer, more efficient accelerators such as NVIDIA H100s offer up to 3$\times$ higher FLOPs and substantially greater memory bandwidth (Liang et al., 2023). Moreover, experiments rely on the standard PyTorch inference stack and Hugging Face's `Accelerate` for orchestration; this omits the algorithmic optimisations and process control exposed in dedicated compilation toolchains such as vLLM, DeepSpeed, ONNX Runtime, or TensorRT. (By way of example, static batching was used to isolate the marginal effects of batch size, whereas dynamic/adaptive batching - an industry standard - would likely moderate and diminish observable batching effect sizes in production contexts, and perhaps even reverse the findings herein (Samsi et al., 2023; Wilkins et al., 2024a); under static batching small (inefficient) batches are needed for low *per-batch* latency (see Figure 14a); however if we consider *per-query* latency patterns, the opposite effect is seen Figure 14 - dynamic batching would support this dynamic.) As most production systems are baseline engineered to operate near their throughput-efficiency frontier, the naive baseline configurations here likely exaggerate the scope of achievable optimisation through deployment tuning.

At the same time, to preserve the constant FLOPs-per-token constraint, the study excludes multi-pass inference tasks such as chain-of-thought prompting, mixture-of-experts, retrieval-augmented generation, and self-consistency decoding - all of which alter both FLOP counts and runtime dynamics. These emerging use cases involve conditional computation paths and iterative reasoning loops, and may yield distinct efficiency profiles with potentially greater scope for energy improvement.

These limitations cut in both directions. On one hand, the restricted search space and naive implementations likely understate the potential gains achievable via more sophisticated deployment strategies. On the other, the low baseline efficiency and simplified runtime environment likely overstate the marginal impact of individual parameters when generalised to well-optimised, production-grade systems. The latter effect is expected to strongly dominate.



(a) per-batch latency (static batching effects)



(b) per-query latency (indicative of dynamic batching effects)

Figure 14: Latency (per-batch/per-query) vs energy outcomes, by batch size. Indicating that with dynamic batching, large concurrent batch sizes would be optimal, with minimal tradeoffs involved.

8 Future Work

Future work should prioritise addressing external validity constraints by executing experiments on dedicated compute clusters, scaling to larger model families, and integrating production-grade inference frameworks. Such extensions would also support assessment of cross-hardware generalisability.

Additionally, broader and deeper implementation-parameter coverage - such as load shaping, burst smoothing, dynamic power management, thermal- and cache-aware scheduling, token-aware routing, fine-grained sharding control, and pipeline parallelism, among others, warrant further investigation. Complementarily, finer-grained temporal analysis of power draw, device utilisation, and resource allocation dynamics is required to characterise instantaneous and cumulative energy behaviour more precisely.

Relaxing the fixed-FLOPs constraint would permit statistical modelling of the conditional FLOPs-energy relationship, enabling analysis of how runtime, throughput, instantaneous power, and device utilisation jointly determine energy outcomes. It would also allow inclusion of multi-pass inference regimes into the analysis. The input-dependent execution paths of these conditional computing architectures opens avenues for data-centric profiling of prompt complexity, structural uncertainty, and reasoning depth - all critical for considering the design of energy benchmarking datasets.

Finally, integrating these findings with sustainable systems research and life-cycle assessment (LCA) models would situate inference energy use within a broader environmental impact framework.

9 Conclusion

This study addresses a critical empirical gap regarding the impact of deployment-level decisions on LLM inference-time energy efficiency. It demonstrates that analytic FLOPs-based metrics inadequately represent actual energy usage, failing to capture significant variation induced by practical implementation choices. Experimental results from controlled parameter variations - including batch size, tensor parallelism, numerical precision, decoding strategy, and simulated latency - reveal substantial intra-model variability in energy efficiency, consistent across models. Particularly, batch sizing and tensor parallelism showed marked energy-efficiency sensitivity; in contrast, numerical precision demonstrated efficiency gains to be dependent on backend integration, while decoder strategies exhibited minimal direct impacts.

The study further emphasises that throughput and device utilisation metrics - routinely captured in industry contexts - offer useful empirical indicators for evaluating and diagnosing deployment (in)efficiency. Integrating system-level metrics sensitive to both architectural and operational factors into benchmarking, shifts the focus from abstract model comparisons to practical, holistic evaluation of deployed systems. This whole-system perspective mitigates the risk of misleading efficiency claims and aligns benchmarking practices with wider *Green AI* objectives. The results herein clearly indicate a risk of distorted efficiency claims arising from standalone reliance on convenient analytical proxies and underline the necessity for standardised empirical benchmarks reflecting realistic operational conditions.

Future research should expand on these findings by incorporating larger model scales, dedicated infrastructure, and production-grade inference frameworks. Moreover, a broader exploration of deployment parameters and finer-grained analyses of power and utilisation dynamics would enrich benchmarking precision, ultimately contributing to more sustainable and informed LLM deployments.

10 Appendix: Developed Python Package

10.1 GitHub URLs

1. Python Package: https://github.com/henrycgbaker/thesis
2. Analysis of Results: https://github.com/henrycgbaker/thesis_analysis

10.2 Description of package

The core work of this thesis was in developing a Python package that measures the energy outcomes of deployed Hugging Face models. At its core, the package separates responsibilities into distinct layers: top-level experiment drivers, orchestration utilities, core experiment logic, and configuration modules.

The first layer consists of the top-level driver scripts (`MAIN_*.py`), which interleave to handle controlled sweeps over variables. Of these, a master script coordinates the entire suite, invoking each driver in sequence or parallel as needed.

Beneath the drivers lies the `experiment_orchestration_utilities` package, which provides an object-oriented `ExperimentRunner` class and workflow definitions. The `ExperimentRunner` initialises experiment metadata, logging, and output directories, manages the full lifecycle of each run, and ensures proper cleanup and error handling. Complementary helper functions handle batch launches and resource management, enabling fully parallel or distributed execution across GPUs.

The orchestration scripts call on a series of helper functions in the `core_experiment_utils` to implement the logic that actually powers the experiments. These modules manage unitary tasks such as loading and placing models on devices, processing and tokenising prompts, executing inference, and capturing both performance and energy consumption metrics. Dedicated submodules further derive additional metrics and aggregate all results to a single experiment-run level. Finally, results are serialised to structured JSON, CSV, and metadata files, creating a standardised output format for downstream analysis.

Configuration in this framework is handled through a suite of modules that define default settings, model lists, controlled-variable sets, scenario definitions, and grid-search ranges. A base `Config` class loads and validates parameters, while utility functions allow for dynamic overrides from the command line. By editing a single config file, users can control every aspect of an experiment—from batch size and random seed to the names of models under test and the energy-monitoring library in use.

To get started, users install the required packages in the `requirements.txt` file and prepare a configuration file (or define parameters in the `configs` subpackage for automated generation of configurations across a given search space). Running a custom experiment involves invoking one of the `MAIN_*.py` scripts with the `--config` flag pointing to the desired configuration, however I suggest starting with the `MAIN_run_experimental_suite.py` and defining the necessary parameters in the relevant `*_config.py` file in the `config directory`, so that the rest of the workflow is already handled programmatically. Outputs are collected under the `results/` directory, where they are directly saved as an input into a collected CSV, with additional generated subfolders for run-level raw inference outputs, computed metrics, energy logs, and experiment metadata.

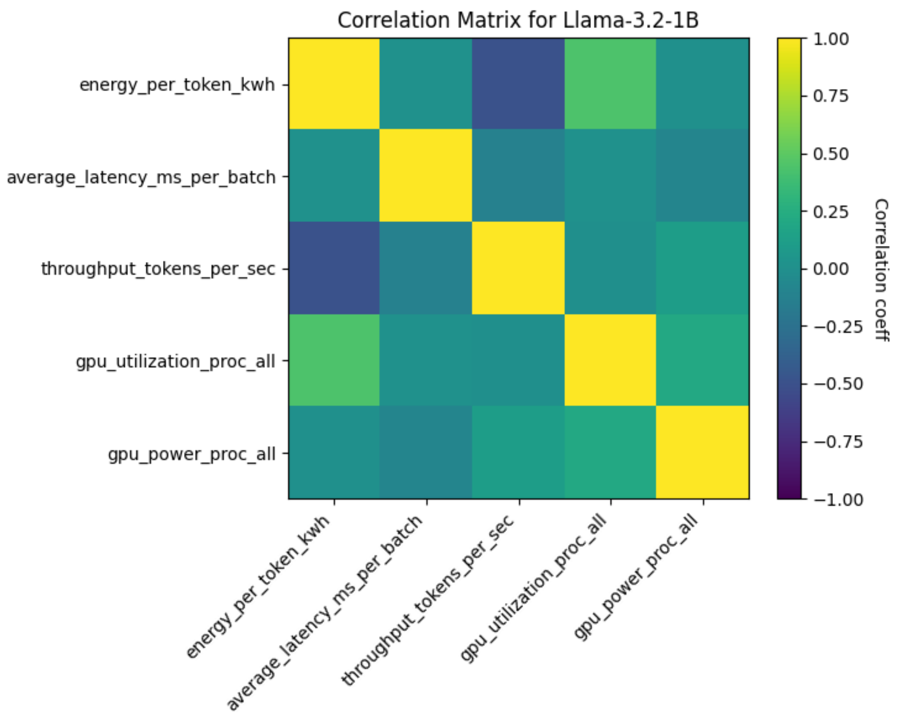
Extending the framework is a matter of adding or customising modules. Researchers can integrate new models by updating the model loader and configuration, implement additional metrics by adding functions to the metrics subpackages, or design entirely new workflows (i.e. beyond standard text generation tasks) by subclassing the `ExperimentRunner` and defining custom steps.

11 Appendix: Energy Appliance Calculations

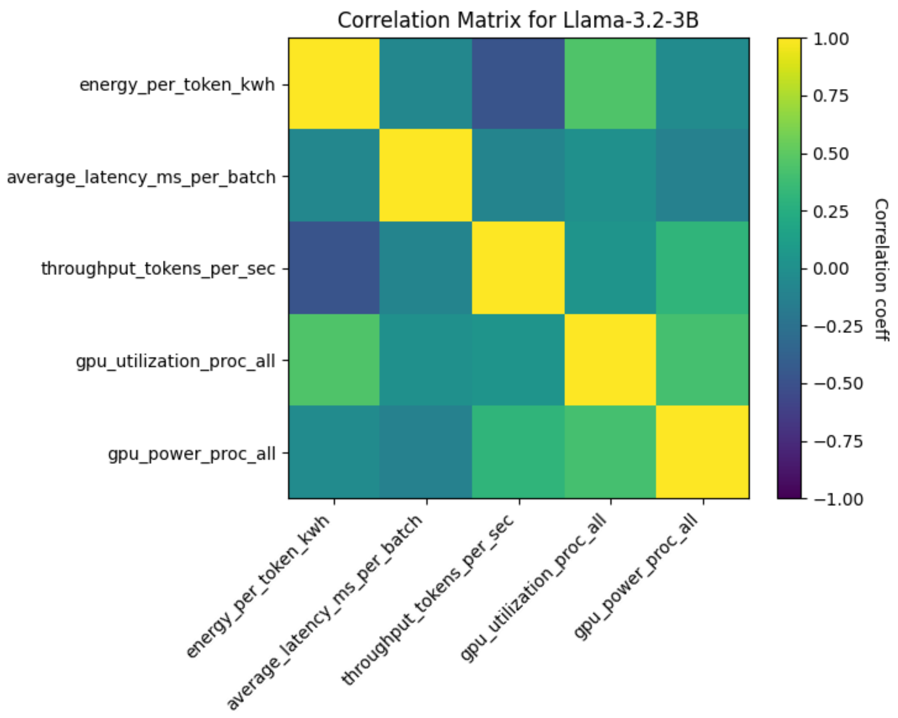
Table 4: Energy consumption of common appliances per use

Appliance	**Usage**	**Energy (kWh)**	**Justification**	**Source**
iPhone 16	Full charge	~ 0.015	Battery ~13.7 Wh; inefficiencies $\rightarrow \sim 0.015$kWh	macworld.com
MacBook Pro 2023 (M3)	Full charge	~ 0.09	99.6Wh $\rightarrow$ 0.0996kWh	support.apple.com
Wi-Fi Router	24h	~ 0.144	6W $\times$ 24h = 144Wh = 0.144kWh	energyusecalculator.com
HD Streaming	1h	~ 0.08	IEA: 0.077kWh per hour of HD	carbonbrief.org
Google Search	One query	~ 0.0003	0.0003kWh (0.3Wh) per search	cloudforecast.io
Electric Kettle	Boil 1L	~ 0.10	2–3kW $\times$ 3min (0.05h) 0.115kWh per litre	electricalfaultsfixed.com
Electric Shower	10min	~ 1.5	9kW $\times$ (10/60h) = 1.5kWh	basengreen.com

12 Appendix: Initial Correlation Plots



(a) Correlation Matrix, 1 B



(b) Correlation Matrix, 3 B

Figure 15: Comparison of feature–feature correlations for the 1 B vs. 3 B decoder model (tot robustly modelled)

13 Appendix: Miscellaneous Plots

I did not have time/space to include several plots that I had begun work on - here are some early works that I found interesting, but did not fully develop.

Note: these plots contain both 1B and 3B models together and do not visually distinguish between the two - ideally I would have separated these out with different markers and mean lines, which I expect would render interesting patterns more visible.

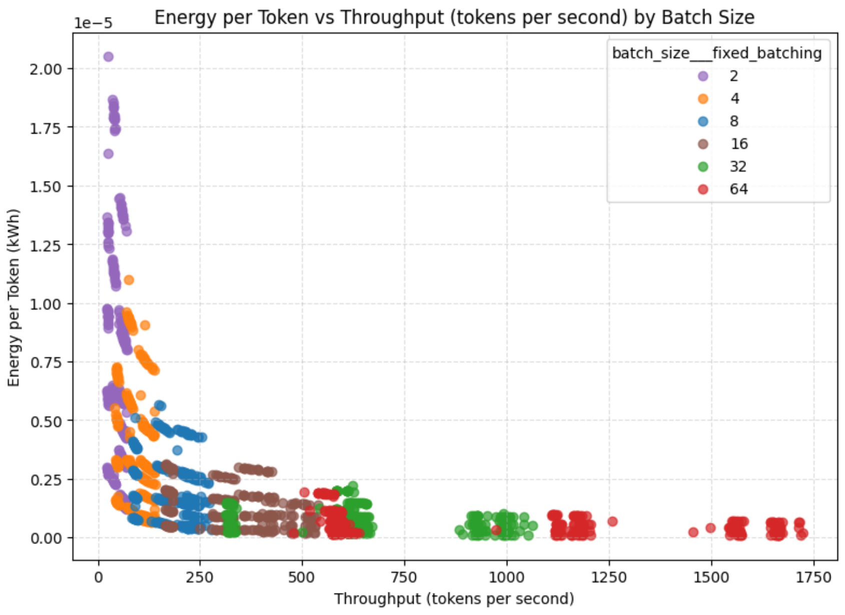
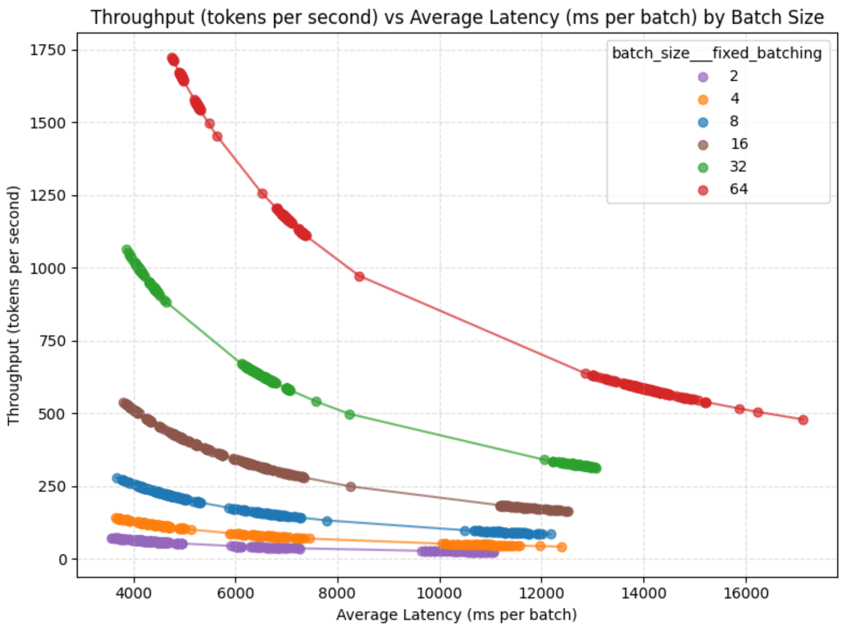
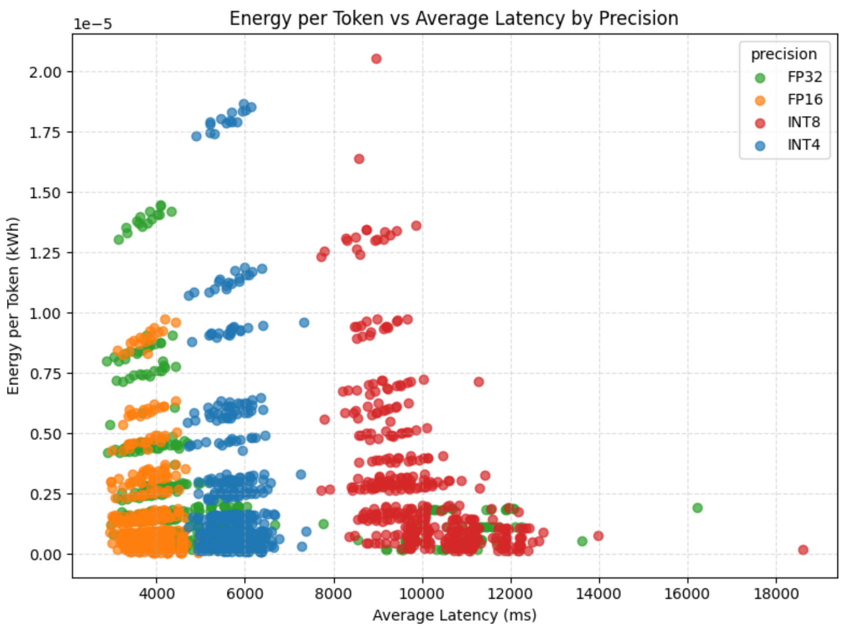
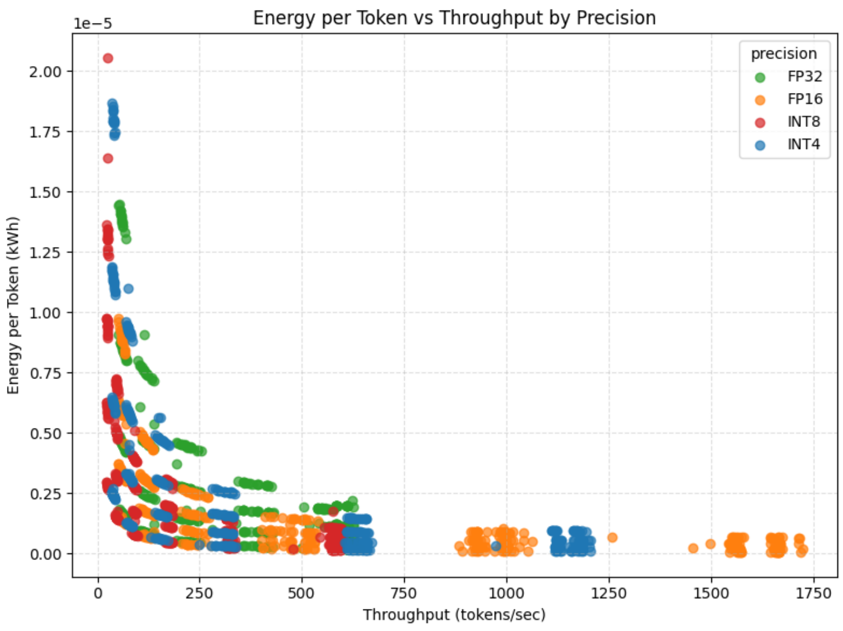
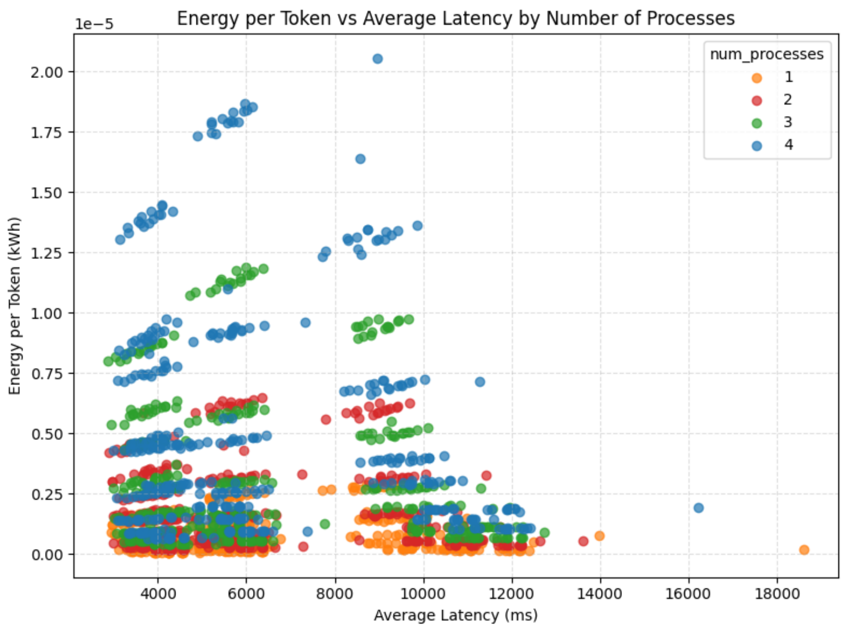


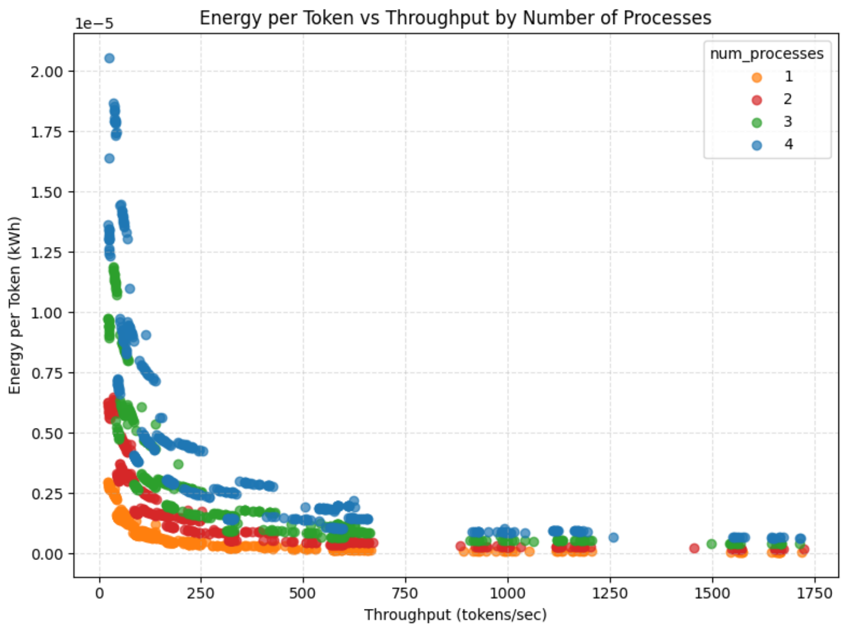
Figure 16: That there's this clear batching distinctions without distinguishing between models, suggests that batch size matters more than model complexity (at the 1B vs 3B scale)




Energy per Token vs Average Latency by Precision
1e−5
Energy per Token (kWh)
Average Latency (ms)
precision
FP32
FP16
INT8
INT4


Energy per Token vs Throughput by Precision
1e−5
Energy per Token (kWh)
Throughput (tokens/sec)
precision
FP32
FP16
INT8
INT4


Energy per Token vs Average Latency by Number of Processes
1e−5
Energy per Token (kWh)
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
4000
6000
8000
10000
12000
14000
16000
18000
Average Latency (ms)
num_processes
● 1
● 2
● 3
● 4


Energy per Token vs Throughput by Number of Processes
1e−5
Energy per Token (kWh)
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
0
250
500
750
1000
1250
1500
1750
Throughput (tokens/sec)
num_processes
● 1
● 2
● 3
● 4

14 Appendix: Baseline Config

Listing 1: Base experiment configuration

```
base_config = { \\
    "config_name": None,            \\
    "suite": None,                   \\
    "controlled_variation": {},     \\
    "scenario_info": {},            \\
    "cycle_id": None,               \\

    "model_name": "TinyLlama/TinyLlama-1.1B-Chat-v1.0", \\
    "is_encoder_decoder": False, \\
    "task_type": "text_generation", \\
    "inference_type": "pure_generative", \\
    "gpu_list": [0, 1, 2, 3], \\
    "backend": "pytorch", \\
    "save_outputs": True, \\

    # Default values (overridden by the grid)
    "max_input_tokens": 128,
    "max_output_tokens": 128,
    "min_output_tokens": 128,
    "num_input_prompts": 128,
    "decode_token_to_text": True,
    "num_processes": 4,
    "batching_options": {
        "batch_size___fixed_batching": 16,
        "adaptive_batching": False,
        "adaptive_max_tokens": 0,
        "max_batch_size___adaptive_batching": 0,
    },
    "sharding_config": {
        "fsdp_config": {
            "use_orig_params": False,
            "cpu_offload": False
        },
        "sharding_strategy": "NO_SHARD"
    },
    "query_rate": 1.0,
    "latency_simulation": {
        "simulate": False,
        "delay_min": 0,
        "delay_max": 0,
        "simulate_burst": False,
        "burst_interval": 0.0,
        "burst_size": 0
    },
    "decoder_config": {
        "decoding_mode": None,
        "decoder_temperature": 1.0,
        "decoder_top_k": None,
        "decoder_top_p": None
    },
    "fp_precision": "float32",
    "quantization_config": {
        "quantization": None,
        "load_in_8bit": None,
        "load_in_4bit": None,
    }}
```

15 Appendix: Decoder Plots

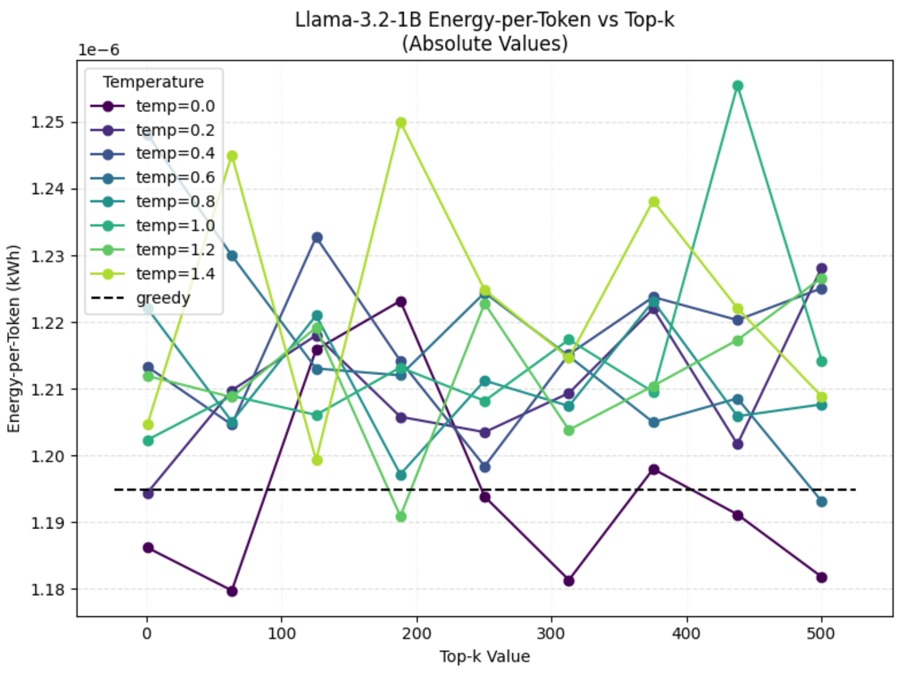


Figure 17: Top–k vs energy outcomes for 1B (absolute).

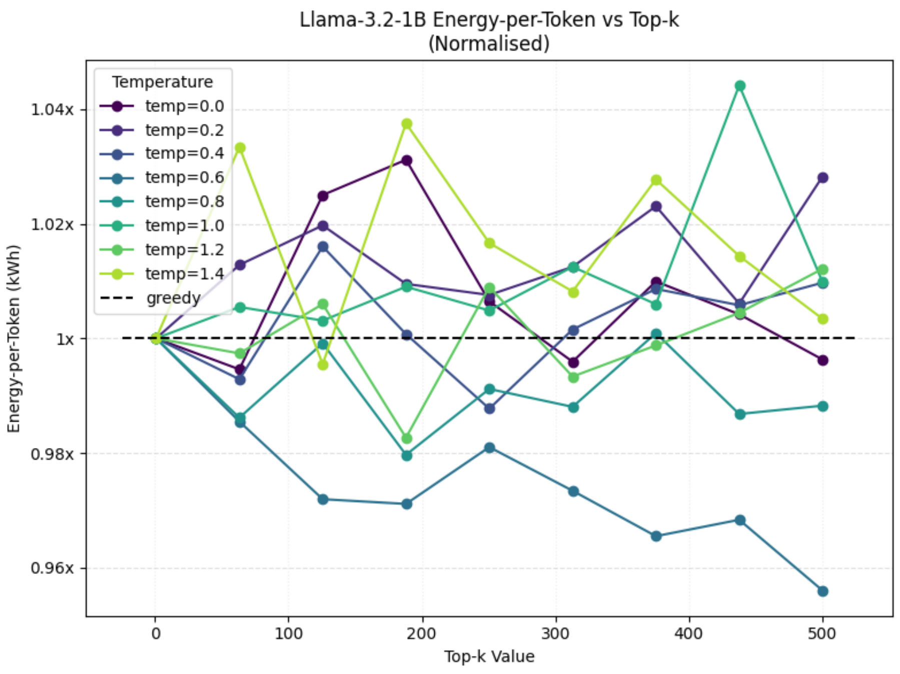


Figure 18: Top–k vs energy outcomes for 1B (normalised).

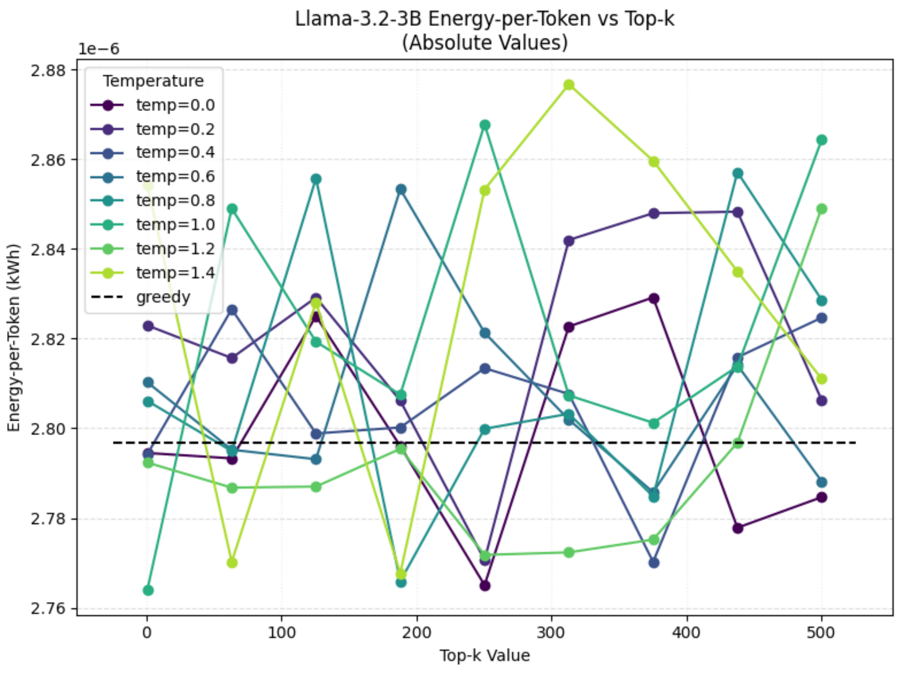


Figure 19: Top–k vs energy outcomes for 3B (absolute).

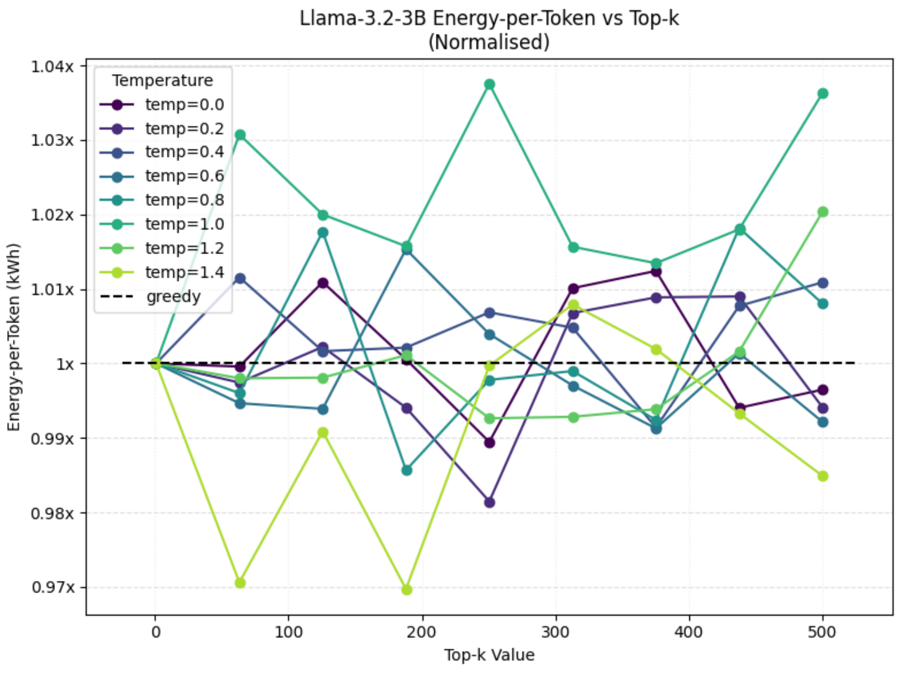


Figure 20: Top–k vs energy outcomes for 3B (normalised).

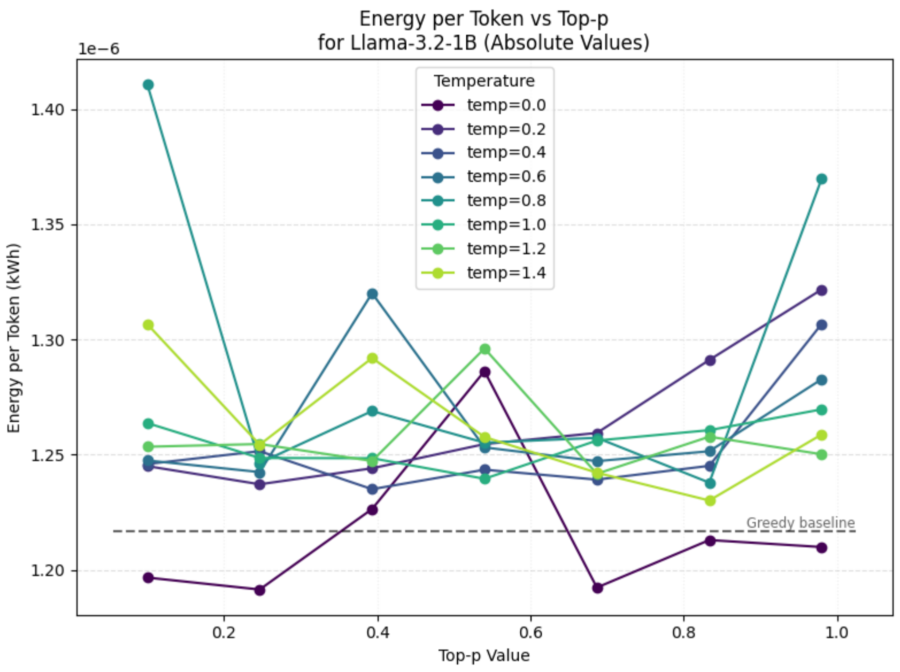


Figure 21: Top–p vs energy outcomes for 1B (absolute).

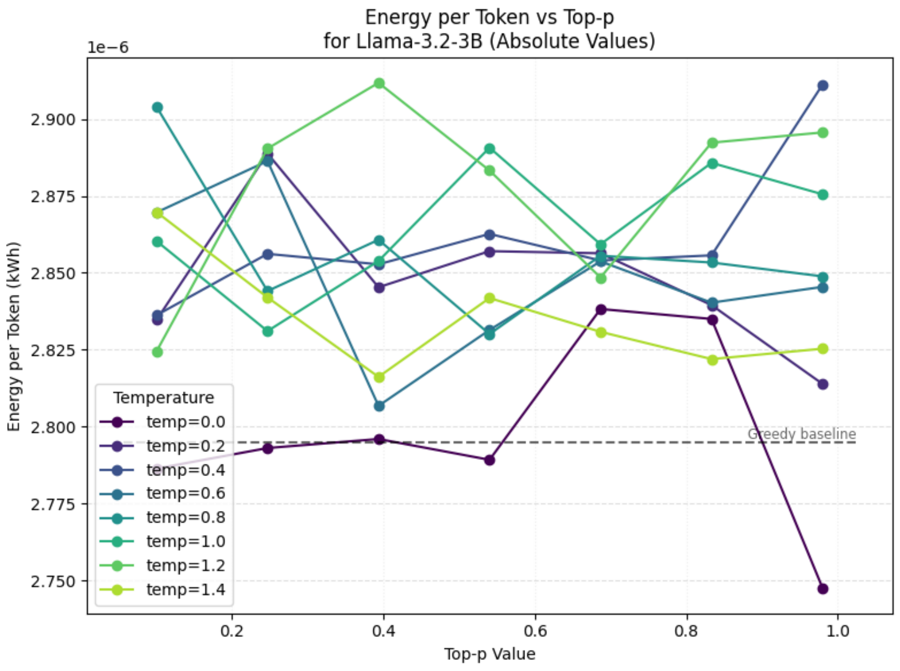


Figure 22: Top–p vs energy outcomes for 3B (absolute).

figures/decoder_topp_norm_1b.png

Figure 23: Top–p vs energy outcomes for 1B (normalised).

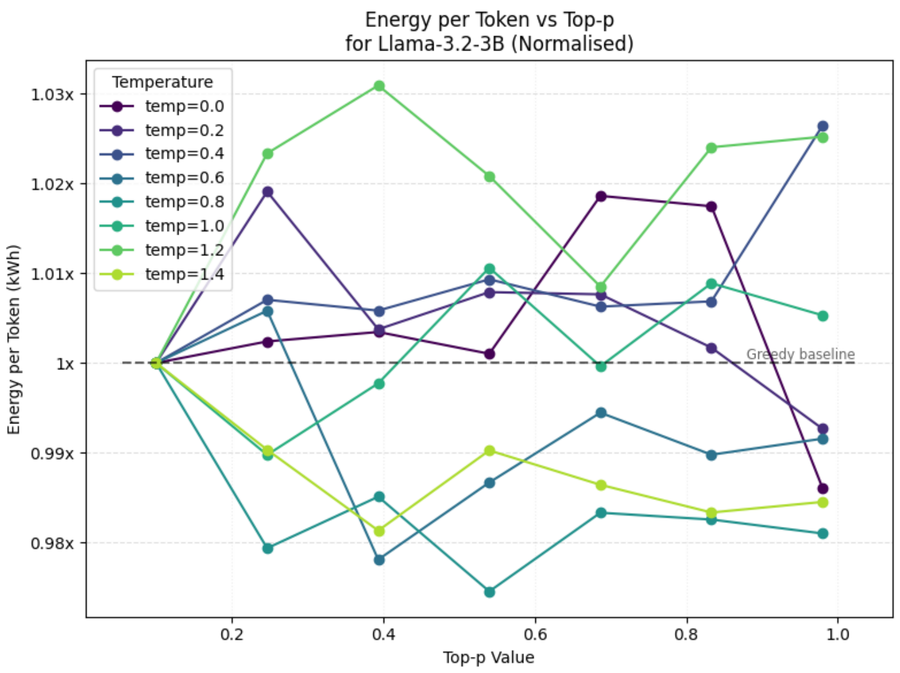


Figure 24: Top–p vs energy outcomes for 3B (normalised).

16 Appendix: Throughput Plots

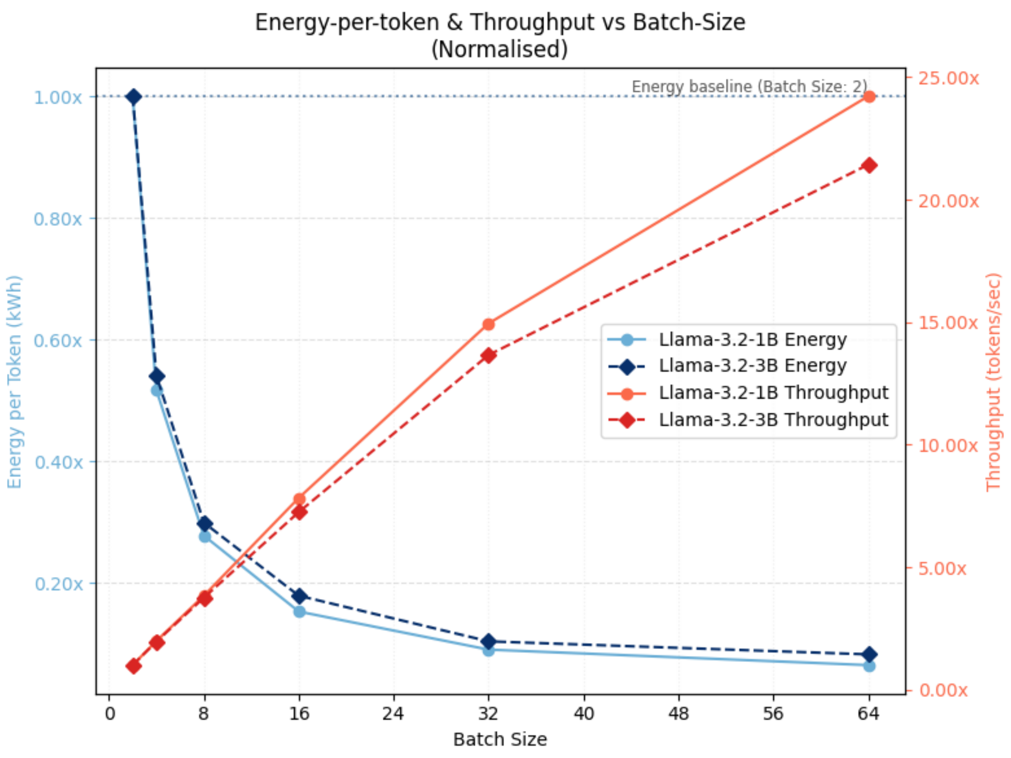


Figure 25: Batch size (normalised)

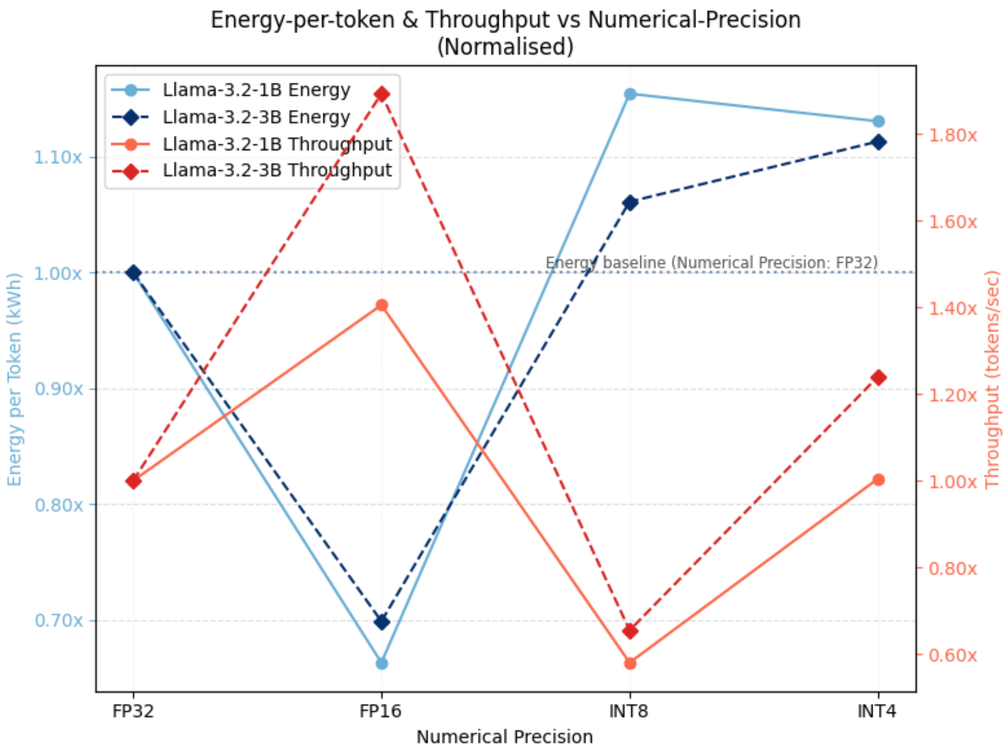


Figure 26: Precision (normalised)

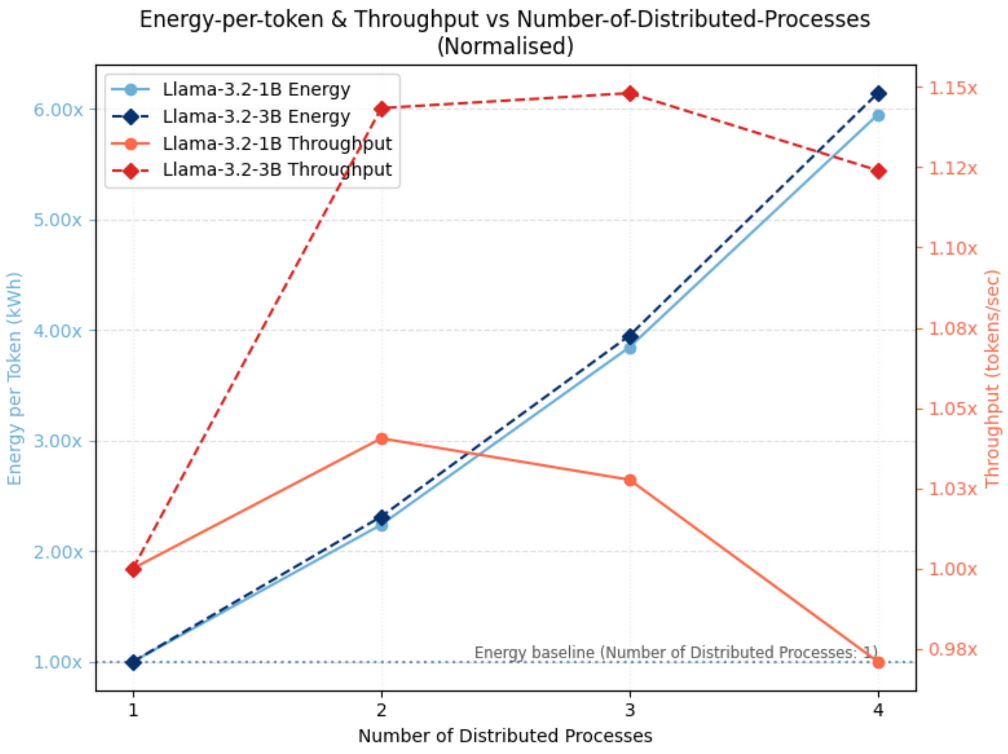


Figure 27: Number of processes (normalised)

17 Appendix: Simulated Scenario Configurations

Table 5: Overview of scenario configurations

Scenario	Batch	Precision	Quantization	Decoding	Latency Sim?	Delay (s)	Burst?	Realistic
A1_ideal	128	float16	None	greedy	No	–	No	No
R1	4	float16	None	top-k (k=500)	Yes	0.01–0.10	No	Yes
R2	8	float16	None	top_p (p=0.9)	Yes	0.01–0.10	No	Yes
R3	16	-	4-bit	multinomial	Yes	0.05–0.50	Yes (3s, 6)	Yes
R4	8	-	8-bit	top_p (p=0.8)	Yes	0.05–0.50	Yes (3s, 6)	Yes
R5	10	float32	None	multinomial	Yes	0.30–0.80	Yes (4s, 5)	Yes
R6	16	float32	None	top_k (k=400)	Yes	0.30–0.80	Yes (4s, 5)	Yes

Note: For all scenarios, the number of GPUs and parallel processes is held constant at four. Although deploying on four GPUs is excessive for the LLaMA-3.2-1B and 3B models studied here - and leads to device under-utilisation - this choice was deliberately made to reflect typical production deployments. Larger-scale LLMs typical of industry, generally require small GPU clusters for efficient inference serving, thus employing four GPUs yields energy consumption results (slightly) more representative of realistic production environments - a (pragmatic, and problematic step for internal-validity) step towards enhancing external validity.

References

AI is set to drive surging electricity demand from data centres while offering the potential to transform how the energy sector works - News. (2025, April). Retrieved May 16, 2025, from https://www.iea.org/news/ai-is-set-to-drive-surging-electricity-demand-from-data-centres-while-offering-the-potential-to-transform-how-the-energy-sector-works

Alizadeh, N., & Castor, F. (2024). Green AI: A Preliminary Empirical Study on Energy Consumption in DL Models Across Different Runtime Infrastructures [arXiv:2402.13640 [cs]]. *Proceedings of the IEEE/ACM 3rd International Conference on AI Engineering - Software Engineering for AI*, 134–139. https://doi.org/10.1145/3644815.3644967

Amazon. (2019, December). Amazon EC2 Update [Section: Amazon EC2]. Retrieved May 16, 2025, from https://aws.amazon.com/blogs/aws/amazon-ec2-update-inf1-instances-with-aws-inferentia-chips-for-high-performance-cost-effective-inferencing/

Aminabadi, R. Y., Rajbhandari, S., Awan, A. A., Li, C., Li, D., Zheng, E., Ruwase, O., Smith, S., Zhang, M., Rasley, J., & He, Y. (2022). DeepSpeed- Inference: Enabling Efficient Inference of Transformer Models at Unprecedented Scale [ISSN: 2167-4337]. *SC22: International Conference for High Performance Computing, Networking, Storage and Analysis*, 1–15. https://doi.org/10.1109/SC41404.2022.00051

Anderson, T., Belay, A., Chowdhury, M., Cidon, A., & Zhang, I. (2022, January). Treehouse: A Case For Carbon-Aware Datacenter Software [arXiv:2201.02120 [cs]]. https://doi.org/10.48550/arXiv.2201.02120

Anthony, L. F. W., Kanding, B., & Selvan, R. (2020, July). Carbontracker: Tracking and Predicting the Carbon Footprint of Training Deep Learning Models [arXiv:2007.03051 [cs]]. https://doi.org/10.48550/arXiv.2007.03051

Axberg, T. (2022). Deriving an Natural Language Processing inference Cost Model with Greenhouse Gas Accounting: Towards a sustainable usage of Machine Learning.

Barbierato, E., & Gatti, A. (2024). Toward Green AI: A Methodological Survey of the Scientific Literature. *IEEE Access*, *12*, 23989–24013. https://doi.org/10.1109/ACCESS.2024.3360705

Brownlee, A. E., Adair, J., Haraldsson, S. O., & Jabbo, J. (2021). Exploring the Accuracy – Energy Trade-off in Machine Learning. *2021 IEEE/ACM International Workshop on Genetic Improvement (GI)*, 11–18. https://doi.org/10.1109/GI52543.2021.00011

Budennyy, S. A., Lazarev, V. D., Zakharenko, N. N., Korovin, A. N., Plosskaya, O. A., Dimitrov, D. V., Akhripkin, V. S., Pavlov, I. V., Oseledets, I. V., Barsola, I. S., Egorov, I. V., Kosterina, A. A., & Zhukov, L. E. (2022). eco2AI: Carbon Emissions Tracking of Machine Learning Models as the First Step Towards Sustainable AI. *Doklady Mathematics*, *106*(S1), S118–S128. https://doi.org/10.1134/S1064562422060230

Cai, E., Juan, D.-C., Stamoulis, D., & Marculescu, D. (2017). NeuralPower : Predict and Deploy Energy-Efficient Convolutional Neural Networks.

Canziani, A., Paszke, A., & Culurciello, E. (2017, April). An Analysis of Deep Neural Network Models for Practical Applications [arXiv:1605.07678 [cs]]. https://doi.org/10.48550/arXiv.1605.07678

The carbon footprint of streaming video: Fact-checking the headlines – Analysis. (2020, December). Retrieved May 17, 2025, from https://www.iea.org/commentaries/the-carbon-footprint-of-streaming-video-fact-checking-the-headlines

Castaño, J., Martínez-Fernández, S., Franch, X., & Bogner, J. (2023). Exploring the Carbon Footprint of Hugging Face's ML Models: A Repository Mining Study [arXiv:2305.11164 [cs]]. *2023 ACM/IEEE International Symposium on*

Empirical Software Engineering and Measurement (ESEM), 1–12. https://doi.org/10.1109/ESEM56168.2023.10304801

Chen, Y.-H., Krishna, T., Emer, J., & Sze, V. (2016). Eyeriss: An Energy-Efficient Reconfigurable Accelerator for Deep Convolutional Neural Networks.

Chien, A. A., Lin, L., Nguyen, H., Rao, V., Sharma, T., & Wijayawardana, R. (2023). Reducing the Carbon Impact of Generative AI Inference (today and in 2035). *Proceedings of the 2nd Workshop on Sustainable Computer Systems*, 1–7. https://doi.org/10.1145/3604930.3605705

Chitty-Venkata, K. T., Raskar, S., Kale, B., Ferdaus, F., Tanikanti, A., Raffenetti, K., Taylor, V., Emani, M., & Vishwanath, V. (2024, October). LLM-Inference-Bench: Inference Benchmarking of Large Language Models on AI Accelerators [arXiv:2411.00136 [cs]]. https://doi.org/10.48550/arXiv.2411.00136

Da Silva Barros, T., Ferre, D., Giroire, F., Aparicio-Pardo, R., & Perennes, S. (2024). Scheduling Machine Learning Compressible Inference Tasks with Limited Energy Budget. *Proceedings of the 53rd International Conference on Parallel Processing*, 961–970. https://doi.org/10.1145/3673038.3673106

De Vries, A. (2023). The growing energy footprint of artificial intelligence. *Joule*, *7*(10), 2191–2194. https://doi.org/10.1016/j.joule.2023.09.004

Decoding Strategies: How LLMs Choose The Next Word. (n.d.). Retrieved April 13, 2025, from https://assemblyai.com/blog/decoding-strategies-how-llms-choose-the-next-word

Dehghani, M., Arnab, A., Beyer, L., Vaswani, A., & Tay, Y. (2022, March). The Efficiency Misnomer [arXiv:2110.12894 [cs]]. https://doi.org/10.48550/arXiv.2110.12894

Desislavov, R., Martínez-Plumed, F., & Hernández-Orallo, J. (2023). Trends in AI inference energy consumption: Beyond the performance-vs-parameter laws of deep learning. *Sustainable Computing: Informatics and Systems*, *38*, 100857. https://doi.org/10.1016/j.suscom.2023.100857

Dettmers, T., Lewis, M., Belkada, Y., & Zettlemoyer, L. (2022). LLM.int8(): 8-bit Matrix Multiplication for Transformers at Scale.

Dodge, J., Prewitt, T., Tachet Des Combes, R., Odmark, E., Schwartz, R., Strubell, E., Luccioni, A. S., Smith, N. A., DeCario, N., & Buchanan, W. (2022). Measuring the Carbon Intensity of AI in Cloud Instances. *2022 ACM Conference on Fairness, Accountability, and Transparency*, 1877–1894. https://doi.org/10.1145/3531146.3533234

Ebert, K., Alder, N., Herbrich, R., & Hacker, P. (2024, October). AI, Climate, and Regulation: From Data Centers to the AI Act. https://doi.org/10.2139/ssrn.4980340

Electricity usage of a Wi-Fi Router - Energy Use Calculator. (n.d.). Retrieved May 17, 2025, from https://energyusecalculator.com/electricity_wifirouter.htm

Exclusive: Uncovering the Real Battery Capacities of the iPhone 16 Series. (2024, September). Retrieved May 17, 2025, from https://blogdoiphone.com/en/news/exclusive-uncovering-the-real-battery-capacities-of-the-iphone-16-series/

Fernandez, J., Kahn, J., Na, C., Bisk, Y., & Strubell, E. (2023, December). The Framework Tax: Disparities Between Inference Efficiency in NLP Research and Deployment [arXiv:2302.06117 [cs]]. https://doi.org/10.48550/arXiv.2302.06117

Fernandez, J., Wehrstedt, L., Shamis, L., Elhoushi, M., Saladi, K., Bisk, Y., Strubell, E., & Kahn, J. (2024, November). Hardware Scaling Trends and Diminishing Returns in Large-Scale Distributed Training [arXiv:2411.13055 [cs]]. https://doi.org/10.48550/arXiv.2411.13055

Fischer, R., Jakobs, M., & Morik, K. (2023, April). Energy Efficiency Considerations for Popular AI Benchmarks [arXiv:2304.08359 [cs]]. https://doi.org/10.48550/arXiv.2304.08359

Fischer, R., Jakobs, M., Mücke, S., & Morik, K. (2023). A Unified Framework for Assessing Energy Efficiency of Machine Learning [Series Title: Communications in Computer and Information Science]. In I. Koprinska, P. Mignone, R. Guidotti, S. Jaroszewicz, H. Fröning, F. Gullo, P. M. Ferreira, D. Roqueiro, G. Ceddia, S. Nowaczyk, J. Gama, R. Ribeiro, R. Gavaldà, E. Masciari, Z. Ras, E. Ritacco, F. Naretto, A. Theissler, P. Biecek, … S. Pashami (Eds.), *Machine Learning and Principles and Practice of Knowledge Discovery in Databases* (pp. 39–54, Vol. 1752). Springer Nature Switzerland. https://doi.org/10.1007/978-3-031-23618-1_3

García-Martín, E., Rodrigues, C. F., Riley, G., & Grahn, H. (2019). Estimation of energy consumption in machine learning. *Journal of Parallel and Distributed Computing*, *134*, 75–88. https://doi.org/10.1016/j.jpdc.2019.07.007

Georgiou, S., Kechagia, M., Sharma, T., Sarro, F., & Zou, Y. (2022). Green AI: Do deep learning frameworks have different costs? *Proceedings of the 44th International Conference on Software Engineering*, 1082–1094. https://doi.org/10.1145/3510003.3510221

Gu, J., Cho, K., & Li, V. O. (2017). Trainable Greedy Decoding for Neural Machine Translation. *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 1968–1978. https://doi.org/10.18653/v1/D17-1210

Guan, X., Bashir, N., Irwin, D., & Shenoy, P. (2023). WattScope: Non-intrusive application-level power disaggregation in datacenters. *Performance Evaluation*, *162*, 102369. https://doi.org/10.1016/j.peva.2023.102369

Gupta, U., Kim, Y. G., Lee, S., Tse, J., Lee, H.-H. S., Wei, G.-Y., Brooks, D., & Wu, C.-J. (2021). Chasing Carbon: The Elusive Environmental Footprint of Computing [ISSN: 2378-203X]. *2021 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*, 854–867. https://doi.org/10.1109/HPCA51647.2021.00076

He, X., Sun, J., Chen, H., & Li, D. (2022). Campo: Cost-Aware Performance Optimization for Mixed-Precision Neural Network Training.

Henderson, P., Hu, J., Romoff, J., Brunskill, E., Jurafsky, D., & Pineau, J. (2020). Towards the Systematic Reporting of the Energy and Carbon Footprints of Machine Learning.

Hogade, N., & Pasricha, S. (2024). Game-Theoretic Deep Reinforcement Learning to Minimize Carbon Emissions and Energy Costs for AI Inference Workloads in Geo-Distributed Data Centers. *IEEE Transactions on Sustainable Computing*, 1–14. https://doi.org/10.1109/TSUSC.2024.3520969

Holtzman, A., Buys, J., Du, L., Forbes, M., & Choi, Y. (2020, February). The Curious Case of Neural Text Degeneration [arXiv:1904.09751 [cs]]. https://doi.org/10.48550/arXiv.1904.09751

Howarth, J. (2024, August). Number of Parameters in GPT-4 (Latest Data). Retrieved May 14, 2025, from https://explodingtopics.com/blog/gpt-parameters

Hu, Q., Sun, P., Yan, S., Wen, Y., & Zhang, T. (2021). Characterization and Prediction of Deep Learning Workloads in Large-Scale GPU Datacenters [arXiv:2109.01313 [cs]]. *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, 1–15. https://doi.org/10.1145/3458817.3476223

HuggingFace. (2024, December). AIEnergyScore/text_generation · Datasets at Hugging Face. Retrieved May 11, 2025, from https://huggingface.co/datasets/AIEnergyScore/text_generation

HuggingFace. (2025, February). AI Energy Score. Retrieved May 11, 2025, from https://huggingface.co/AIEnergyScore

Husom, E. J., Goknil, A., Shar, L. K., & Sen, S. (2024, July). The Price of Prompting: Profiling Energy Use in Large Language Models Inference [arXiv:2407.16893 [cs]]. https://doi.org/10.48550/arXiv.2407.16893

IEA. (2025). Energy and AI – Analysis. Retrieved May 20, 2025, from https://www.iea.org/reports/energy-and-ai

Ji, S., Yang, Z., Jones, A. K., & Zhou, P. (2024, August). Advancing Environmental Sustainability in Data Centers by Proposing Carbon Depreciation Models [arXiv:2403.04976 [cs]]. https://doi.org/10.48550/arXiv.2403.04976

Kaack, L. H., Donti, P. L., Strubell, E., Kamiya, G., Creutzig, F., & Rolnick, D. (2022). Aligning artificial intelligence with climate change mitigation. *Nature Climate Change*, *12*(6), 518–527. https://doi.org/10.1038/s41558-022-01377-7

Kakolyris, A. K., Masouros, D., Vavaroutsos, P., Xydis, S., & Soudris, D. (2024, August). SLO-aware GPU Frequency Scaling for Energy Efficient LLM Inference Serving [arXiv:2408.05235 [cs]]. https://doi.org/10.48550/arXiv.2408.05235

Kettle - Wikiwand. (n.d.). Retrieved May 17, 2025, from https://www.wikiwand.com/en/articles/Kettle

Kwon, W., Li, Z., Zhuang, S., Sheng, Y., Zheng, L., Yu, C. H., Gonzalez, J. E., Zhang, H., & Stoica, I. (2023, September). Efficient Memory Management for Large Language Model Serving with PagedAttention [arXiv:2309.06180 [cs]]. https://doi.org/10.48550/arXiv.2309.06180

Lacoste, A., Luccioni, A., Schmidt, V., & Dandres, T. (2019, November). Quantifying the Carbon Emissions of Machine Learning [arXiv:1910.09700 [cs]]. https://doi.org/10.48550/arXiv.1910.09700

Latotzke, C., Balim, B., & Gemmeke, T. (2022a, October). Post-Training Quantization for Energy Efficient Realization of Deep Neural Networks [arXiv:2210.07906 [cs]]. https://doi.org/10.48550/arXiv.2210.07906

Latotzke, C., Balim, B., & Gemmeke, T. (2022b, October). Post-Training Quantization for Energy Efficient Realization of Deep Neural Networks [arXiv:2210.07906 [cs]]. https://doi.org/10.48550/arXiv.2210.07906

Leviathan, Y., Kalman, M., & Matias, Y. (2023). Fast Inference from Transformers via Speculative Decoding [ISSN: 2640-3498]. *Proceedings of the 40th International Conference on Machine Learning*, 19274–19286. Retrieved May 11, 2025, from https://proceedings.mlr.press/v202/leviathan23a.html

Li, D., Chen, X., Becchi, M., & Zong, Z. (2016). Evaluating the Energy Efficiency of Deep Convolutional Neural Networks on CPUs and GPUs. *2016 IEEE International Conferences on Big Data and Cloud Computing (BDCloud), Social Computing and Networking (SocialCom), Sustainable Computing and Communications (SustainCom) (BDCloud-SocialCom-SustainCom)*, 477–484. https://doi.org/10.1109/BDCloud-SocialCom-SustainCom.2016.76

Li, G., Sun, Z., Wang, Q., Wang, S., Huang, K., Zhao, N., Di, Y., Zhao, X., & Zhu, Z. (2023). China's green data center development:Policies and carbon reduction technology path. *Environmental Research*, *231*, 116248. https://doi.org/10.1016/j.envres.2023.116248

Li, Patel, T., Samsi, S., Gadepally, V., & Tiwari, D. (2022). MISO: Exploiting Multi-Instance GPU Capability on Multi-Tenant Systems for Machine Learning [arXiv:2207.11428 [cs]]. *Proceedings of the 13th Symposium on Cloud Computing*, 173–189. https://doi.org/10.1145/3542929.3563510

Liang, P., Bommasani, R., Lee, T., Tsipras, D., Soylu, D., Yasunaga, M., Zhang, Y., Narayanan, D., Wu, Y., Kumar, A., Newman, B., Yuan, B., Yan, B., Zhang, C., Cosgrove, C., Manning, C. D., Ré, C., Acosta-Navas, D., Hudson, D. A., … Koreeda, Y. (2023, October). Holistic Evaluation of Language Models [arXiv:2211.09110 [cs]]. https://doi.org/10.48550/arXiv.2211.09110

Luccioni, A. S., & Hernandez-Garcia, A. (2023, February). Counting Carbon: A Survey of Factors Influencing the Emissions of Machine Learning [arXiv:2302.08476 [cs]]. https://doi.org/10.48550/arXiv.2302.08476

Luccioni, A. S., Viguier, S., & Ligozat, A.-L. (2022, November). Estimating the Carbon Footprint of BLOOM, a 176B Parameter Language Model [arXiv:2211.02001 [cs]]. https://doi.org/10.48550/arXiv.2211.02001

Luccioni, Jernite, Y., & Strubell, E. (2024). Power Hungry Processing: Watts Driving the Cost of AI Deployment? *The 2024 ACM Conference on Fairness, Accountability, and Transparency*, 85–99. https://doi.org/10.1145/3630106.3658542

Luccioni, S. (2024). Energy Scores for AI Models. Retrieved February 19, 2025, from https://huggingface.co/blog/sasha/energy-star-ai-proposal

MacBook Pro - Tech Specs. (n.d.). Retrieved May 17, 2025, from https://www.apple.com/macbook-pro/specs/

Maliakel, P. J., Ilager, S., & Brandic, I. (2025, January). Investigating Energy Efficiency and Performance Trade-offs in LLM Inference Across Tasks and DVFS Settings [arXiv:2501.08219 [cs]]. https://doi.org/10.48550/arXiv.2501.08219

Mavromatis, I., Katsaros, K., & Khan, A. (2024, June). Computing Within Limits: An Empirical Study of Energy Consumption in ML Training and Inference [arXiv:2406.14328 [cs]]. https://doi.org/10.48550/arXiv.2406.14328

Milojevic-Dupont, N., Hans, N., Kaack, L. H., Zumwald, M., Andrieux, F., De Barros Soares, D., Lohrey, S., Pichler, P.-P., & Creutzig, F. (2020). Learning from urban form to predict building heights (F. Biljecki, Ed.). *PLOS ONE*, *15*(12), e0242010. https://doi.org/10.1371/journal.pone.0242010

Mlco2/codecarbon [original-date: 2020-05-12T14:44:03Z]. (2025, May). Retrieved May 11, 2025, from https://github.com/mlco2/codecarbon

Narang, S., Diamos, G., Elsen, E., Micikevicius, P., Alben, J., Garcia, D., Ginsburg, B., Houston, M., Kuchaiev, O., Venkatesh, G., & Wu, H. (2018). MIXED PRECISION TRAINING.

Ni, W., Hu, X., Du, H., Kang, Y., Ju, Y., & Wang, Q. (2024). CO2 emission-mitigation pathways for China's data centers. *Resources, Conservation and Recycling*, *202*, 107383. https://doi.org/10.1016/j.resconrec.2023.107383

Nik, A., Riegler, M. A., & Halvorsen, P. (2025a, February). Energy-Conscious LLM Decoding: Impact of Text Generation Strategies on GPU Energy Consumption [arXiv:2502.11723 [cs]]. https://doi.org/10.48550/arXiv.2502.11723

Nik, A., Riegler, M. A., & Halvorsen, P. (2025b, February). Energy-Conscious LLM Decoding: Impact of Text Generation Strategies on GPU Energy Consumption [arXiv:2502.11723 [cs]]. https://doi.org/10.48550/arXiv.2502.11723

OpenAI-Developer-Community. (2023, July). Temperature, top_p and top_k for chatbot responses - Prompting [Section: Prompting]. Retrieved April 22, 2025, from https://community.openai.com/t/temperature-top-p-and-top-k-for-chatbot-responses/295542

Paccou, R., & Wijnhoven, F. (2024). Artificial Intelligence and Electricity.

Pasek, A., Vaughan, H., & Starosielski, N. (2023). The world wide web of carbon: Toward a relational footprinting of information and communications technology's climate impacts. *Big Data & Society*, *10*(1), 20539517231158994. https://doi.org/10.1177/20539517231158994

Patterson, D., Gonzalez, J., Le, Q., Liang, C., Munguia, L.-M., Rothchild, D., So, D., Texier, M., & Dean, J. (2021). Carbon Emissions and Large Neural Network Training.

Poddar, S., Koley, P., Misra, J., Ganguly, N., & Ghosh, S. (2025, February). Towards Sustainable NLP: Insights from Benchmarking Inference Energy in Large Language Models [arXiv:2502.05610 [cs]]. https://doi.org/10.48550/arXiv.2502.05610

Proceedings of the Workshop on Design, Deployment, and Evaluation of Network-assisted Video Streaming. (n.d.). Retrieved May 11, 2025, from https://dl.acm.org/doi/proceedings/10.1145/3488662

Rodrigues, C. F., Riley, G., & Luján, M. (2018). SyNERGY: An energy measurement and prediction framework for Convolutional Neural Networks on Jetson TX.

Rolnick, D., Donti, P. L., Kaack, L. H., Kochanski, K., Lacoste, A., Sankaran, K., Ross, A. S., Milojevic-Dupont, N., Jaques, N., Waldman-Brown, A., Luccioni, A. S., Maharaj, T., Sherwin, E. D., Mukkavilli, S. K., Kording, K. P., Gomes, C. P., Ng, A. Y., Hassabis, D., Platt, J. C., … Bengio, Y. (2023). Tackling Climate Change with Machine Learning. *ACM Computing Surveys*, *55*(2), 1–96. https://doi.org/10.1145/3485128

rw5603. (2024, October). How Much Energy Do Google Search and ChatGPT Use? Retrieved May 17, 2025, from https://www.rwdigital.ca/blog/how-much-energy-do-google-search-and-chatgpt-use/

Samsi, S., Zhao, D., McDonald, J., Li, B., Michaleas, A., Jones, M., Bergeron, W., Kepner, J., Tiwari, D., & Gadepally, V. (2023, October). From Words to Watts: Benchmarking the Energy Costs of Large Language Model Inference [arXiv:2310.03003 [cs]]. https://doi.org/10.48550/arXiv.2310.03003

Savazzi, S., Rampa, V., Kianoush, S., & Bennis, M. (2023). An Energy and Carbon Footprint Analysis of Distributed and Federated Learning. *IEEE Transactions on Green Communications and Networking*, *7*(1), 248–264. https://doi.org/10.1109/TGCN.2022.3186439

Schwartz, R., Dodge, J., Smith, N. A., & Etzioni, O. (2020). Green AI. *Communications of the ACM*, *63*(12), 54–63. https://doi.org/10.1145/3381831

Selvan, R., Bhagwat, N., Anthony, L. F. W., Kanding, B., & Dam, E. B. (2022). Carbon Footprint of Selecting and Training Deep Learning Models for Medical Image Analysis [arXiv:2203.02202 [eess]]. https://doi.org/10.1007/978-3-031-16443-9_49

Sewerin, S., Kaack, L. H., Küttel, J., Sigurdsson, F., Martikainen, O., Esshaki, A., & Hafner, F. (2023). Towards understanding policy design through text-as-data approaches: The policy design annotations (POLIANNA) dataset. *Scientific Data*, *10*(1), 896. https://doi.org/10.1038/s41597-023-02801-z

Shehabi, A., Smith, S., Sartor, D., Brown, R., Herrlin, M., Koomey, J., Masanet, E., Horner, N., Azevedo, I., & Lintner, W. (2024). *United States Data Center Energy Usage Report* (tech. rep. No. LBNL–1005775, 1372902). https://doi.org/10.2172/1372902

Shi, T., Wu, Y., Liu, S., & Ding, Y. (2024, December). GreenLLM: Disaggregating Large Language Model Serving on Heterogeneous GPUs for Lower Carbon Emissions [arXiv:2412.20322 [cs]]. https://doi.org/10.48550/arXiv.2412.20322

Singh, V. (2023, September). A Guide to Controlling LLM Model Output: Exploring Top-k, Top-p, and Temperature Parameters. Retrieved April 22, 2025, from https://ivibudh.medium.com/a-guide-to-controlling-llm-model-output-exploring-top-k-top-p-and-temperature-parameters-ed6a31313910

Sinha, P., Tummala, S. S., Purkayastha, S., & Gichoya, J. W. (2022). Energy Efficiency of Quantized Neural Networks in Medical Imaging.

Stevens, I. (2024, May). Regulating AI: The Limits of FLOPs as a Metric. Retrieved April 24, 2025, from https://medium.com/@ingridwickstevens/regulating-ai-the-limits-of-flops-as-a-metric-41e3b12d5d0c

Stocker, V., Mariotte, N., Ullrich, A., & Rejeski, D. (2024). ICT Sustainability Reporting Strategies of Large Tech Companies: Changes in Format, Scope, and Content. *SSRN Electronic Journal*. https://doi.org/10.2139/ssrn.4927128

Stojkovic, J., Choukse, E., Zhang, C., Goiri, I., & Torrellas, J. (2024, March). Towards Greener LLMs: Bringing Energy-Efficiency to the Forefront of LLM Inference [arXiv:2403.20306 [cs]]. https://doi.org/10.48550/arXiv.2403.20306

Stojkovic, J., Zhang, C., Goiri, Í., Choukse, E., Qiu, H., Fonseca, R., Torrellas, J., & Bianchini, R. (2025, January). TAPAS: Thermal- and Power-Aware Scheduling for LLM Inference in Cloud Platforms [arXiv:2501.02600 [cs]]. https://doi.org/10.48550/arXiv.2501.02600

Stojkovic, J., Zhang, C., Goiri, Í., Torrellas, J., & Choukse, E. (2024, August). DynamoLLM: Designing LLM Inference Clusters for Performance and Energy Efficiency [arXiv:2408.00741 [cs]]. https://doi.org/10.48550/arXiv.2408.0074 1

Strubell, E., Ganesh, A., & McCallum, A. (2019, June). Energy and Policy Considerations for Deep Learning in NLP [arXiv:1906.02243 [cs]]. https://doi.org /10.48550/arXiv.1906.02243

Strubell, E., Ganesh, A., & McCallum, A. (2020). Energy and Policy Considerations for Modern Deep Learning Research. *Proceedings of the AAAI Conference on Artificial Intelligence*, *34*(09), 13693–13696. https://doi.org/10.1609/aaai.v 34i09.7123

Tang, Y., Cheng, X., & Lu, W. (2022, December). Improving Complex Knowledge Base Question Answering via Question-to-Action and Question-to-Question Alignment. In Y. Goldberg, Z. Kozareva, & Y. Zhang (Eds.), *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing* (pp. 137–147). Association for Computational Linguistics. https://doi.org/1 0.18653/v1/2022.emnlp-main.10

Tiutiulnikov, M., Lazarev, V., Korovin, A., Zakharenko, N., Doroshchenko, I., & Budennyy, S. (2023). Eco4cast: Bridging Predictive Scheduling and Cloud Computing for Reduction of Carbon Emissions for ML Models Training. *Doklady Mathematics*, *108*(S2), S443–S455. https://doi.org/10.1134/S10 64562423701223

Tripp, C. E., Perr-Sauer, J., Gafur, J., Nag, A., Purkayastha, A., Zisman, S., & Bensen, E. A. (2024, March). Measuring the Energy Consumption and Efficiency of Deep Neural Networks: An Empirical Analysis and Design Recommendations [arXiv:2403.08151 [cs]]. https://doi.org/10.48550/arXiv.2403.08151

Understanding Electric Shower kilowatt (kW) Ratings by Mira Showers. (n.d.). Retrieved May 17, 2025, from https://www.mirashowers.co.uk/blog/mira-recommends/kw-ratings-explained

van Wynsberghe, A. (2021). Sustainable AI: AI for sustainability and the sustainability of AI. *AI and Ethics*, *1*(3), 213–218. https://doi.org/10.1007/s43681-021-00043-6

Verdecchia, R., Sallou, J., & Cruz, L. (2023). A systematic review of Green ¡span style="font-variant:small-caps;"¿AI¡/span¿. *WIREs Data Mining and Knowledge Discovery*, *13*(4), e1507. https://doi.org/10.1002/widm.1507

Wang, X., Na, C., Strubell, E., Friedler, S., & Luccioni, S. (2023). Energy and Carbon Considerations of Fine-Tuning BERT [arXiv:2311.10267 [cs]]. *Findings of the Association for Computational Linguistics: EMNLP 2023*, 9058–9069. https://doi.org/10.18653/v1/2023.findings-emnlp.607

Wang, Y., Chen, Y., Li, Z., Kang, X., Tang, Z., He, X., Guo, R., Wang, X., Wang, Q., Zhou, A. C., & Chu, X. (2024, June). BurstGPT: A Real-world Workload Dataset to Optimize LLM Serving Systems [arXiv:2401.17644 [cs]]. https://doi.org/10.48550/arXiv.2401.17644

Weng, Q., Xiao, W., Yu, Y., Wang, W., Wang, C., He, J., Li, Y., Zhang, L., Lin, W., & Ding, Y. (2022). MLaaS in the Wild: Workload Analysis and Scheduling in Large-Scale Heterogeneous GPU Clusters.

Wilkins, G., Keshav, S., & Mortier, R. (2024a). Hybrid Heterogeneous Clusters Can Lower the Energy Consumption of LLM Inference Workloads. *The 15th ACM International Conference on Future and Sustainable Energy Systems*, 506–513. https://doi.org/10.1145/3632775.3662830

Wilkins, G., Keshav, S., & Mortier, R. (2024b, July). Offline Energy-Optimal LLM Serving: Workload-Based Energy Models for LLM Inference on Heterogeneous Systems [arXiv:2407.04014 [cs]]. https://doi.org/10.48550/arXiv.2407.04014

Wu, Hua, I., & Ding, Y. (2025, February). Unveiling Environmental Impacts of Large Language Model Serving: A Functional Unit View [arXiv:2502.11256 [cs]]. https://doi.org/10.48550/arXiv.2502.11256

Wu, Li, G., Chen, F., & Shi, L. (2018, February). Training and Inference with Integers in Deep Neural Networks [arXiv:1802.04680 [cs]]. https://doi.org/10.48550/arXiv.1802.04680

Yang, Z., Adamek, K., & Armour, W. (2024, December). Double-Exponential Increases in Inference Energy: The Cost of the Race for Accuracy [arXiv:2412.09731 [cs]]. https://doi.org/10.48550/arXiv.2412.09731

Yao, Z., Wu, X., Li, C., Youn, S., & He, Y. (2023, May). ZeroQuant-V2: Exploring Post-training Quantization in LLMs from Comprehensive Study to Low Rank Compensation [arXiv:2303.08302 [cs]]. https://doi.org/10.48550/arXiv.2303.08302

Statement of Authorship

I hereby confirm and certify that this master thesis is my own work. All ideas and language of others are acknowledged in the text. All references and verbatim extracts are properly quoted and all other sources of information are specifically and clearly designated. I confirm that the digital copy of the master thesis that I submitted on 22/05/2025 is identical to the printed version I submitted to the Examination Office.

Date: **22/05/2025**

Name: **Henry Baker**

Student ID: **228755**

Signature: