Toward Sustainability-Aware LLM Inference on Edge Clusters

Kolichala Rajashekar, Nafiseh Sharghivand, Radu Prodan
Department of Computer Science
University of Innsbruck, Austria

Reza Farahani
Institute of Information Technology
University of Klagenfurt, Austria

Abstract

Large language models (LLMs) require substantial computational resources, leading to significant carbon emissions and operational costs. Although training is energy-intensive, the long-term environmental burden arises from inference, amplified by the massive global query volume. Cloud-based inference offers scalability but suffers from latency and bandwidth constraints due to centralized processing and continuous data transfer. Edge clusters instead can mitigate these limitations by enabling localized execution, yet they face trade-offs between performance, energy efficiency, and device constraints. This short paper presents a sustainability-aware LLM inference for edge clusters comprising NVIDIA Jetson Orin NX (8GB) and Nvidia Ada 2000 (16GB) devices. It aims to balance inference latency and carbon footprint through carbon- and latency-aware routing strategies, guided by empirical benchmarking of energy consumption and execution time across diverse prompts and batch (i.e., group of prompts) configurations. We compared baseline greedy strategies to carbon-aware and latency-aware strategies in prompt routing to specific hardware based on benchmarking information. Experimental evaluation shows that a batch size of four prompts achieves a trade-off between throughput, energy efficiency, while larger batches risk GPU memory saturation.

Keywords

Sustainability, Large Language Models, LLM inference, Carbon Footprint, Edge Computing.

1 Introduction

Large language models (LLMs) require immense computational resources, driving both high carbon emissions and substantial operational costs. Although LLM training is notoriously energy-intensive, the inference phase, where models process user prompts into responses, poses a greater long-term sustainability challenge due to the massive, continuous global query volume [1, 13, 21]. Cloud servers provide vast computational and memory resources; however, processing all inference in the cloud requires extensive data transmission, which can degrade real-time performance under varying bandwidth availability [11, 22]. Moreover, inference for large-scale models such as GPT-4 can emit, on a daily basis, a carbon footprint comparable to a significant fraction of their one-time training emissions, intensifying sustainability concerns amid accelerating AI adoption [9, 13]. To address these challenges, modern computing architectures increasingly integrate centralized cloud resources with distributed edge devices, where edge instances handle latency-sensitive and lightweight tasks locally, reducing response time and associated emissions, while cloud systems manage compute-intensive workloads at scale [10, 12, 17].

This hybrid paradigm is particularly crucial for LLM inference, where prompt complexity varies widely. For example, simple factual questions may require only modest computational effort, while tasks involving multi-step reasoning or tool use demand substantially greater processing power. However, current LLM inference systems still rely on coarse-grained heuristics, e.g., routing all reasoning-heavy or mathematical prompts to high-capacity models, ignoring hardware heterogeneity, inter-device communication latency, or dynamic energy profiles [4, 8, 16]. These oversights lead to inefficient resource utilization, elevated carbon emissions, and degraded performance in edge deployments.

This short paper introduces sustainable LLM inference on edge clusters composed of NVIDIA Jetson Orin NX (8GB) and Nvidia Ada 2000 (16GB) as representative edge hardware. We propose carbon- and latency-aware strategies informed by extensive benchmarking across diverse LLM prompts from established prompt datasets [1]. Through empirical evaluation of energy consumption, carbon emissions, and end-to-end latency, our results show that these strategies reduce emissions by up to 35 % and improve execution speed by 2–3x compared to greedy baselines. Furthermore, we analyze the effect of batch size, i.e., the number of prompts processed in parallel during a single inference pass to amortize computational overhead, across configurations of 1, 4, and 8. Experimental results shows that batch size of 4 provides the balance between end-to-end latency and energy efficiency, while a batch size of 8 increases GPU utilization at the cost of higher latency. These findings highlight the importance of benchmarking-driven, sustainability-aware LLM inference to enable efficient and environmentally responsible deployments on edge server clusters.

2 Motivation Example

To illustrate the need for sustainability-aware LLM inference, we deployed two quantized Gemma models on an edge cluster: Gemma-3-1B-it-qat on an NVIDIA Jetson Orin NX (8GB) of GPU memory and Gemma-3-12B-it-qat on an NVIDIA Ada 2000 (16GB). For a cloud baseline, we used the Google Gemini 2.0 Flash API [20]. We used Ollama and evaluated four representative prompts (P1–P4), summarized in Table 1, which cover reasoning, generative writing, and factual lookup. We used a judge model (cloud) that rates expected reasoning depth and token footprint, calculating prompt complexity scores (CS); scores are normalized to [0,1], higher is harder.

Performance analysis: We compare the performance of deployed LLM models using four key metrics: *inference time* (IT), *time-to-first-token* (TTFT), *tokens-per-second* (TPS), and *time-per-output-token* (TPOT) [2, 15]. As shown in Fig. 1, the Gemma-3-12B model achieves the shortest TTFT but incurs higher IT and TPOT on longer prompts, whereas Gemma-3-1B provides a more balanced efficiency profile. The cloud-based Gemini 2.0 Flash API delivers superior IT

[1] https://huggingface.co/datasets

https://doi.org/10.34749/3061-1008.2025.9

Table 1: Prompts used in evaluation (CS: complexity score [0–1] from a judge model; higher is harder).

ID	Prompt	CS
P1	A group of five friends (Alice, Bob, Carol, David, Emily) are trying to decide who will buy tickets for a concert, prepare snacks, drive, and pick up drinks. Alice hates driving. Bob can only pick up drinks if he's not preparing snacks. Carol loves concerts and wants to buy tickets. David can only drive if Emily prepares snacks. Emily will not pick up drinks. Each friend must take exactly one task, and each task must be assigned to exactly one friend. Assign the tasks to each friend and explain your logical deduction step by step.	0.47
P2	Write a short story, approximately 500 words, about a sentient, self-repairing antique grandfather clock that secretly orchestrates minor, benevolent 'time anomalies' in a quiet, forgotten library. Introduce a skeptical new librarian who slowly uncovers the clock's secret. The story must include: The clock's motivation for its actions. Three distinct 'time anomalies' are caused. A moment of direct, non-verbal communication between the clock and the librarian. A surprising twist where the librarian, instead of exposing the clock, aids its efforts for an unexpected reason.	0.39
P3	What is the boiling point of water at standard atmospheric pressure?	0.08
P4	Who painted the Mona Lisa?	0.07

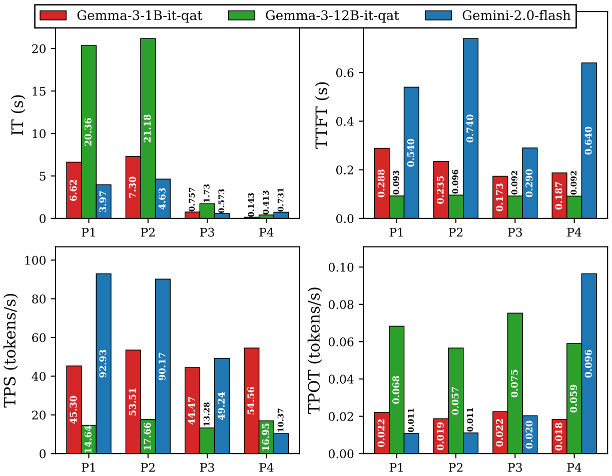

Figure 1: Comparison of inference performance metrics, IT, TTFT, TPS, and TPOT, across NVIDIA Jetson Orin NX (8 GB), Ada 2000 (16 GB), and the Gemini 2.0 Flash cloud API.

and TPS for complex prompts (P1, P2) but underperforms on simpler factual queries (P4), indicating bandwidth and dispatch overheads.

Sustainability analysis: Fig. 2 shows the measured carbon footprint (in CO_2eq) and power draw (in watts) obtained using the JetPack SDK [2] and the PyNVML library [3]. Gemma-3-1B emits roughly one-tenth the carbon of Gemma-3-12B on reasoning prompts (P1, P2), while both models exhibit low emissions on simpler ones (P3, P4). These results demonstrate that hardware-aware and model-adaptive inference can substantially reduce energy consumption and carbon emissions, underscoring the potential of sustainability-oriented deployment strategies for LLMs on edge clusters.

Key takeaway: Sustainability-aware LLM inference on edge clusters demands balancing performance, energy, and carbon efficiency.

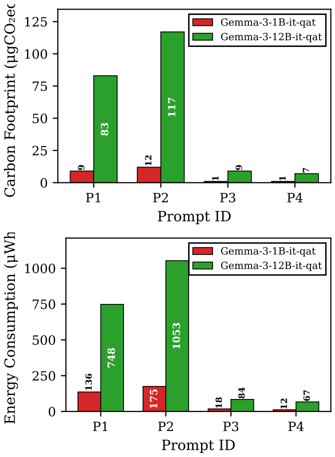

Figure 2: Carbon footprint and energy consumption for Gemma3-1B-it and Gemma3-12B-it models across prompts P1–P4.

Relying solely on either compact edge models or large cloud-based LLMs is suboptimal. Instead, hardware- and model-aware LLM inference is crucial for minimizing emissions while maintaining responsiveness. As shown in Fig. 1, lightweight models such as Gemma-3-1B achieve low latency for simple prompts, whereas Fig. 2 confirms substantial energy and carbon savings. For complex reasoning tasks, larger models, such as Gemma-3-12B or the Gemini 2.0 Flash API, offer superior output quality, underscoring the need for workload distribution across the edge–cloud continuum.

3 Benchmark Evaluation

We assessed our edge cluster, introduced in Section 2, using a mixed dataset that integrates prompts from multiple publicly available sources. The prompts spans diverse domains, including math reasoning (GSM8K) [7], extractive question answering (SQuAD) [19], dialogue summarization (DialogSum) [5], Python coding instructions [3], multiple-choice science reasoning (ARC-Challenge) [6], long-form summarization of arXiv papers, multi-turn dialogue continuation [18], and general long-form summarization [14]. From this composite benchmark of approximately 5000 prompts, we sampled 500 representative inputs to measure both end-to-end inference latency and the corresponding carbon footprint across the edge cluster. Table 2 summarizes the benchmark results, reporting average performance metrics across all evaluated configurations. These results provide an overview of the latency–energy trade-offs and serve as a practical guide for resource allocation in edge–cloud deployments. We implemented two LLM inference strategies:

(i) Carbon-aware: Assigns each prompt to the model with the measured lower carbon footprint, prioritizing emission reduction even if it increases latency.

(ii) Latency-aware: Employs a greedy heuristic that sorts prompts by decreasing average latency and assigns them to minimize total end-to-end execution time.

Baselines include assigning all prompts exclusively to either the Jetson Orin NX (8 GB) or the Ada 2000 (16 GB). The LLM inference results for batch sizes of 1, 4, and 8, summarized in Table 3, reveal

[2] https://developer.nvidia.com/embedded/jetpack

[3] https://pypi.org/project/pynvml/

Table 2: Average inference metrics across edge devices and batch configurations.

Hardware	Batch Size	E2E Latency (s)	TTFT (s)	TPOT (s)	Token Count	Tokens/s (Throughput)	Energy (kWh)	Carbon (kgCO2e)
Ada 2000 16GB	1	3.39	0.26	0.03	69.62	20.54	6.35e-05	4.38e-06
	4	14.58	12.07	0.02	56.83	3.90	5.05e-05	3.49e-06
	8	26.82	24.00	0.03	63.97	2.39	5.73e-05	3.96e-06
Jetson 8GB	1	13.06	0.36	0.061	148	11.33	1.79e-05	1.23e-06
	4	15.08	1.13	0.063	149	9.88	4.89e-06	3.37e-07
	8	14.12	4.87	0.057	136	9.63	5.12e-06	3.53e-07

clear trade-offs between execution time and carbon footprint. For a batch size of 1, assigning all prompts to the NVIDIA Jetson (8GB) yields a total execution time of 1873.13 s with a carbon footprint of 0.000209 kg CO_2e, whereas using the NVIDIA Ada (16GB) reduces execution time to 1354.25 s but increases emissions to 0.000300 kg CO_2e. The *carbon-aware* strategy achieves the minimum footprint by directing roughly 85 % of prompts to the Jetson device, leveraging its energy efficiency for low-token tasks such as sentiment analysis. However, this causes high end-to-end (E2E) latency due to load imbalance from compute-intensive tasks such as Python coding. In contrast, the *latency-aware* strategy minimizes total E2E latency to 580.34 s by balancing workload distribution, assigning complex tasks to the Ada device, while maintaining a moderate carbon footprint of 0.000247 kg CO_2e. These results highlight the complementary roles of both devices: the Jotson device excels in lightweight workloads, whereas the Ada instance dominates in memory- and compute-intensive inference.

For a batch size of 4, the *Jetson-only* baseline achieves an execution time of 649.6 s with a carbon footprint of 0.000071 kg CO_2e, while the *Ada-only* configuration reduces execution time to 568.4 s but increases emissions to 0.000103 kg CO_2e. The *carbon-aware* strategy lowers emissions by approximately 33 % (0.000069 kg CO_2e) by routing around 80 % of prompts to the Ada device. However, E2E latency remains high due to memory constraints affecting compute-intensive tasks such as Python coding. In contrast, the *latency-aware* strategy achieves the shortest E2E latency, about twice as fast as the *Jetson-only* baseline, while maintaining a moderate carbon footprint of 0.000085 kg CO_2e, benefiting from balanced workload assignment that exploits the Ada device's efficiency for high-token prompts. Overall, a batch size of 4 provides a strong trade-off between throughput and energy efficiency, although minor accuracy degradation on the Ada device indicates limitations in handling larger model states.

At a batch size of 8, the *Jetson-only* baseline shows an execution time of 609 s with a carbon footprint of 0.000057 kg CO_2e, while the *Ada-only* setup reduces execution time to 533.6 s but increases emissions to 0.000084 kg CO_2e. The *carbon-aware* strategy yields the lowest footprint (0.000055 kg CO_2e) by routing approximately 75 % of prompts to the Jetson device. However, its E2E latency (552.4 s) remains elevated due to instability on high-token workloads. In contrast, the *latency-aware* strategy minimizes E2E latency to 266.8 s, roughly twice as fast as the *Jetson-only* baseline, while maintaining a moderate footprint of 0.000070 kg CO_2e, benefiting from the Ada device's greater stability in long-form summarization and other memory-intensive tasks. Although a batch size of 8 maximizes throughput, it introduces instability and accuracy degradation on the Jetson device, indicating that larger-memory configurations are preferable for high-batch inference workloads.

Table 3: Comparison of different LLM inference strategies across batch sizes 1, 4, and 8.

Strategy	Total E2E latency (s)	Total Carbon Footprint (kgCO2e)
	Batch Size 1	
All on Jetson (8GB)	1873.13	0.000209
All on Ada (16GB)	1354.25	0.000300
Carbon-Aware	1674.86	0.000204 (lowest)
Latency-Aware	580.34 (lowest)	0.000247
	Batch Size 4	
All on Jetson (8GB)	649.6	0.000071
All on Ada (16GB)	568.4	0.000103
Carbon-Aware	590.2	0.000069 (lowest)
Latency-Aware	284.2 (lowest)	0.000085
	Batch Size 8	
All on Jetson (8GB)	609.0	0.000057
All on Ada (16GB)	533.6	0.000084
Carbon-Aware	552.4	0.000055 (lowest)
Latency-Aware	266.8 (lowest)	0.000070

Cross-batch analysis reveals that overall latency decreases with larger batch size, as parallel token generation reduces TPOT. However, TTFT increases significantly, noticeably, limiting responsiveness in real-time applications. The carbon footprint per prompt declines with batching, since energy costs are amortized across multiple inputs. The Ada device consistently achieves higher accuracy, particularly at batch size 8, where the Jetson device exhibits errors due to memory saturation. Overall, batch size 4 offers the best trade-off between latency, carbon footprint, and accuracy, whereas batch size 8 maximizes throughput but demands at least 16GB of GPU memory for stable operation. The Jetson instance remains well-suited for lightweight, low-token tasks, while the Ada device excels in high-token, high-batch inference scenarios.

4 Conclusion and Future Work

This paper presented an empirical analysis of sustainability-aware LLM inference on heterogeneous edge server clusters. Our results show that increasing and balancing prompt batch sizes provides a clear trade-off between latency and carbon footprint, making it suitable for mixed workloads. In contrast, a larger batch size maximizes throughput but requires more memory for stability. In our experiments, the carbon-aware strategy reduces emissions by up to 35 % by leveraging the Jetson device efficiency for low-token tasks, while the latency-aware strategy achieves 2-3x faster execution times through balanced load distribution. These findings highlight the importance of dynamic, complexity-aware LLM inference for optimizing performance and minimizing environmental impact. Future work will investigate scalability for unseen prompts and adaptive edge-server selection to advance sustainable LLM inference.

https://doi.org/10.34749/3061-1008.2025.9

Acknowledgment

This work received funding from the Horizon Europe research and innovation program (grant agreement 101189771, DataPACT) and the Austrian Research Promotion Agency (FFG, grant agreement 909989, AIM AT Stiftungsprofessur für Edge AI).

References

[1] Zoha Azimi, Reza Farahani, Christian Timmerer, and Radu Prodan. 2025. Towards an Energy-Efficient Video Processing Tool with LLMs. In *Proceedings of the 4th Mile-High Video Conference*. 89–90.

[2] Kayhan Behdin, Yun Dai, Ata Fatahibaarzi, Aman Gupta, Qingquan Song, Shao Tang, Hejian Sang, Gregory Dexter, Sirou Zhu, Siyu Zhu, et al. 2025. Efficient AI in Practice: Training and Deployment of Efficient LLMs for Industry Applications. *arXiv preprint arXiv:2502.14305* (2025).

[3] Tarun Bisht. 2021. python_code_instructions_18k: A Large-Scale Dataset of Python Code Instructions. Dataset on Hugging Face. https://huggingface.co/datasets/iamtarun/python_code_instructions_18k_alpaca.

[4] Shuhao Chen, Weisen Jiang, Baijiong Lin, James Kwok, and Yu Zhang. 2024. Routerdc: Query-based router by dual contrastive learning for assembling large language models. *Advances in Neural Information Processing Systems* 37 (2024), 66305–66328.

[5] Yulong Chen, Yang Liu, Liang Chen, and Yue Zhang. 2021. DialogSum: A Real-Life Scenario Dialogue Summarization Dataset. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*. Association for Computational Linguistics, Online, 5062–5074. doi:10.18653/v1/2021.findings-acl.449

[6] Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have Solved Question Answering? Try ARC, the AI2 Reasoning Challenge. *arXiv:1803.05457v1* (2018).

[7] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training Verifiers to Solve Math Word Problems. *arXiv preprint arXiv:2110.14168* (2021).

[8] Dujian Ding, Ankur Mallick, Chi Wang, Robert Sim, Subhabrata Mukherjee, Victor Ruhle, Laks VS Lakshmanan, and Ahmed Hassan Awadallah. 2024. Hybrid llm: Cost-efficient and quality-aware query routing. *arXiv preprint arXiv:2404.14618* (2024).

[9] Reza Farahani, Zoha Azimi, Christian Timmerer, and Radu Prodan. 2024. Towards AI-Assisted Sustainable Adaptive Video Streaming Systems: Tutorial and Survey. *arXiv preprint arXiv:2406.02302* (2024).

[10] Reza Farahani, Narges Mehran, Sashko Ristov, and Radu Prodan. 2024. Heftless: A Bi-objective Serverless Workflow Batch Orchestration on the Computing Continuum. In *2024 IEEE International Conference on Cluster Computing (CLUSTER)*. IEEE, 286–296.

[11] Reza Farahani and Radu Prodan. [n. d.]. Serverless Large Language Models: Edge vs. Cloud Deployment Trade-offs. In *Book of Abstracts*. 29.

[12] Reza Farahani and Radu Prodan. 2025. EnergyLess: An Energy-Aware Serverless Workflow Batch Orchestration on the Computing Continuum. In *2025 IEEE 18th International Conference on Cloud Computing (CLOUD)*. IEEE.

[13] Zhenxiao Fu, Fan Chen, Shan Zhou, Haitong Li, and Lei Jiang. 2024. LLMCO2: Advancing Accurate Carbon Footprint Prediction for LLM Inferences. *arXiv preprint arXiv:2410.02950* (2024).

[14] Karl Moritz Hermann, Tomáš Kočiský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching Machines to Read and Comprehend. In *Proceedings of NIPS*. Introduces the CNN/DailyMail summarization dataset.

[15] Kunal Jain, Anjaly Parayil, Ankur Mallick, Esha Choukse, Xiaoting Qin, Jue Zhang, Íñigo Goiri, Rujia Wang, Chetan Bansal, Victor Rühle, et al. 2024. Intelligent router for llm workloads: Improving performance through workload-aware scheduling. *arXiv preprint arXiv:2408.13510* (2024).

[16] Aly M Kassem, Bernhard Schölkopf, and Zhijing Jin. 2025. How Robust Are Router-LLMs? Analysis of the Fragility of LLM Routing Capabilities. *arXiv preprint arXiv:2504.07113* (2025).

[17] Jiaxing Li, Chi Xu, Lianchen Jia, Feng Wang, Cong Zhang, and Jiangchuan Liu. 2024. EACO-RAG: Towards Distributed Tiered LLM Deployment using Edge-Assisted and Collaborative RAG with Adaptive Knowledge Update. *arXiv preprint arXiv:2410.20299* (2024).

[18] Yanran Li, Hui Su, Xiaoyu Shen, Wenjie Li, Ziqiang Cao, and Shuzi Niu. 2017. Dailydialog: A manually labelled multi-turn dialogue dataset. *arXiv preprint arXiv:1710.03957* (2017).

[19] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ Questions for Machine Comprehension of Text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, Jian Su, Kevin Duh, and Xavier Carreras (Eds.). Association for Computational Linguistics, Austin, Texas, 2383–2392. arXiv:1606.05250 [cs.CL] doi:10.18653/v1/D16-1264

[20] Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, et al. 2025. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786* (2025).

[21] Clovis Varangot-Reille, Christophe Bouvard, Antoine Gourru, Mathieu Ciancone, Marion Schaeffer, and François Jacquenet. 2025. Doing More with Less–Implementing Routing Strategies in Large Language Model-Based Systems: An Extended Survey. *arXiv preprint arXiv:2502.00409* (2025).

[22] Zheming Yang, Yuanhao Yang, Chang Zhao, Qi Guo, Wenkai He, and Wen Ji. 2024. Perllm: Personalized inference scheduling with edge-cloud collaboration for diverse llm services. *arXiv preprint arXiv:2405.14636* (2024).

https://doi.org/10.34749/3061-1008.2025.9