Small Prompts, Big Energy and CO_2 Impact: Benchmarking Ollama LLMs on CPU and GPU

Ana Kolovska, Marjan Gusev, Dimitar Mileski

Ss. Cyril and Methodius University, Faculty of Computer Science and Engineering, Skopje, North Macedonia,
ana.kolovska@students.finki.ukim.mk {marjan.gushev, dimitar.mileski}@finki.ukim.mk,

***Abstract*—Energy efficiency is a crucial challenge when deploying Large Language Models (LLMs). Electricity usage and related CO_2 emissions can differ greatly depending on model architecture, parameter size, prompt length, and inference hardware. In this study, we evaluate 31 popular Ollama models across CPU and GPU inference, resulting in 60 testing scenarios. Energy and carbon metrics were gathered using the NVML and CodeCarbon libraries, providing insights into the environmental impact of LLM inference in data center settings.**

***Index Terms*—LLM, Ollama, Energy Efficiency, Carbon Footprint, Electricity Consumption, CPU Inference, GPU Inference, Data Centers, Environmental Impact**

I. Introduction

The deployment of Large Language Models (LLMs) poses growing concerns about energy efficiency and environmental sustainability. As model complexity increases, inference demands substantial computational resources, leading to higher electricity consumption and carbon emissions. While frameworks such as Hugging Face Transformers, TensorFlow Serving and NVIDIA TensorRT improve scalability and throughput; however, fewer studies have addressed the energy and carbon implications of LLM inference. GPUs offer greater energy efficiency than CPUs due to their parallel processing capabilities. However, the environmental impact of model execution, especially within containerized and multi-hardware environments, remains insufficiently benchmarked [1]. This motivates a systematic evaluation of energy consumption and carbon footprint during LLM inference.

We evaluated 31 Ollama [2] models across two inference configurations (GPU and CPU) within 60 evaluation scenarios. Energy consumption and carbon footprint were analyzed using 10 metrics collected with NVIDIA's NVML and CodeCarbon libraries, enabling a systematic assessment of energy efficiency during LLM inference.

The paper is further organized as follows. Section II reviews related work on energy-efficient LLM inference. Section III describes our experimental setup, including the selection of 31 Ollama models, the GPU and CPU configurations, and the use of NVML and CodeCarbon. Section IV presents results from 60 evaluation scenarios across 10 collected metrics. Section V presents a detailed discussion of the results. Finally, Section VI concludes the paper and outlines directions for future work.

II. Related work

The growing deployment of LLMs has drawn significant attention to their environmental and energy impacts. Previous research has highlighted the ever-increasing carbon footprint of LLM inference and training, underscoring the need for systematic methods to measure energy efficiency across different hardware platforms. While much focus has been placed on training efficiency, the inference stage, where LLMs are most often used in real-world applications, remains less studied, even though it is a major factor in operational costs and emissions.

Many research efforts have sought to fill this gap by developing frameworks for energy-aware monitoring and benchmarking. Strubell et al. [1] first quantified the significant carbon emissions from deep learning models, sparking further interest in measurement methods. More recently, CodeCarbon has been introduced as a lightweight, open-source tool to estimate emissions during machine learning workloads [3]. However, its reliance on generalized hardware profiles and country-level emission factors can limit its accuracy when benchmarked against real-world inference scenarios. The MELODI framework [4] builds on this by combining real-time monitoring with model performance evaluation, demonstrating that emissions are linked not only to hardware utilization but also to prompt complexity and inference latency.

Several approaches have also examined detailed energy tracking methods. The study [5] suggests improvements at both the algorithm and system levels, such as dynamic batching and quantization, to decrease the energy footprint. Similarly, EcoServe [6] highlights energy-aware scheduling, demonstrating that workload placement across different hardware (CPU, GPU, and accelerators) can lead to significant reductions in emissions.

Another area of research has explored real-time power measurement tools. The paper [7] argues that emission calculators like CodeCarbon may underestimate short-lived inference tasks because they do not capture transient energy spikes. Instead, the authors suggest direct hardware-based measurements (e.g., NVIDIA's NVML for GPU monitoring), allowing for more accurate comparisons between CPU and GPU workloads.

Carbon-aware optimization techniques are also discussed in related literature. The article from [8] emphasizes how workload scheduling based on renewable energy availability

This is accepted version. Published version DOI: 10.1109/TELFOR67910.2025.11314365

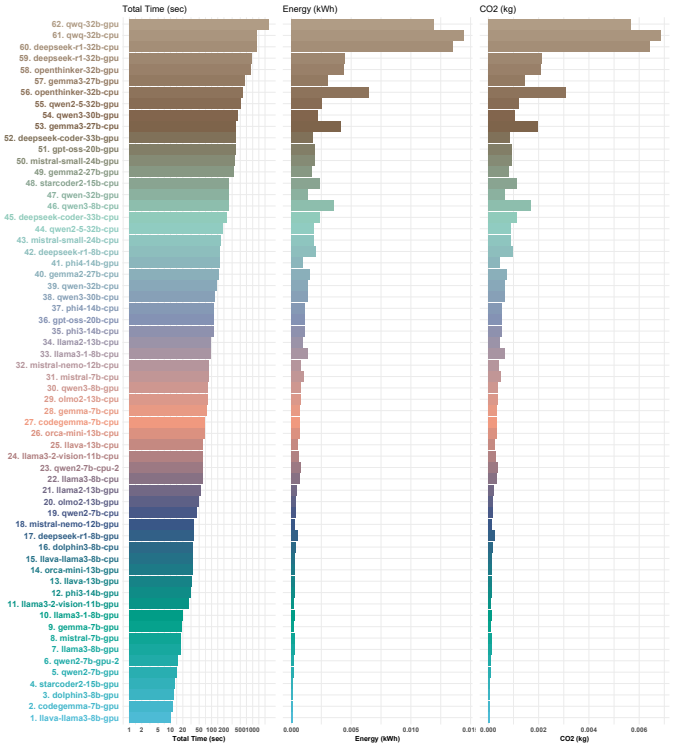


Fig. 1: NVML - Total Time (s), Energy (kWh), and CO_2 (kg) footprint comparison for all evaluated models.

can lower the net footprint of AI workloads. Additionally, [9] examines strategies such as workload deferral and carbon-intensity-based routing to enhance sustainability without compromising user experience.

Finally, comparative studies of CPU versus GPU execution offer insights into the efficiency trade-offs relevant to our work. While GPUs provide higher throughput for parallelizable workloads, recent studies [10] indicate that CPUs can sometimes be more energy efficient for small-batch or latency-sensitive inference. Other research, including [7], emphasizes how hardware selection interacts with model size and request patterns to influence emissions.

Previous research has shown that precise, hardware-aware energy and emission tracking is crucial for sustainable AI efforts. Existing tools like CodeCarbon offer a starting point but fall short in capturing the behavior of short-lived LLM inference workloads. Recent frameworks suggest real-time monitoring and carbon-aware scheduling. However, few have explored these issues within the context of Ollama, a popular inference framework that allows flexible deployment on both CPU and GPU.

III. Methods

This study creates a benchmarking framework to assess the **energy consumption and carbon footprint** of Ollama-based LLM inference on both CPU and GPU. By combining *software-level estimations* with *hardware-level monitoring*, we measure the environmental impact of inference across models of different sizes and architectures.

A. Model Selection

A total of **31 LLMs available through Ollama** were selected for evaluation. These models span multiple families (e.g., LLaMA, DeepSeek, Gemma, Mistral ...) and differ in parameter scale:

- **Small to medium models**: 7B–8B parameters (optimized for lower memory footprint).
- **Intermediate models**: 14B parameters (balanced between performance and resource demand).
- **Large models**: 30B parameters (high accuracy but high computational cost).

B. Creating CPU and GPU Variants

To ensure a fair comparison, we created two separate versions of each model: one configured to run on the GPU and the other forced to run on the CPU. This was achieved using **Ollama's Modelfile system**.

The **GPU version** was created from the base model using the default configuration with the Modelfile specifying `FROM [model name]` and `PARAMETER num_gpu 1`, and the command `ollama create [model name]_gpu -f /path/to/Modelfile`. The **CPU version** was created by explicitly disabling GPU allocation using `PARAMETER num_gpu 0` in the Modelfile, and the command `ollama create [model name]_cpu -f /path/to/Modelfile`.

This procedure was repeated for all 31 models, ensuring that each had a `_cpu` and `_gpu` variant. By doing so, CPU and GPU inference paths were isolated, allowing direct comparison of energy consumption and emissions under identical workloads.

C. Benchmark Prompts

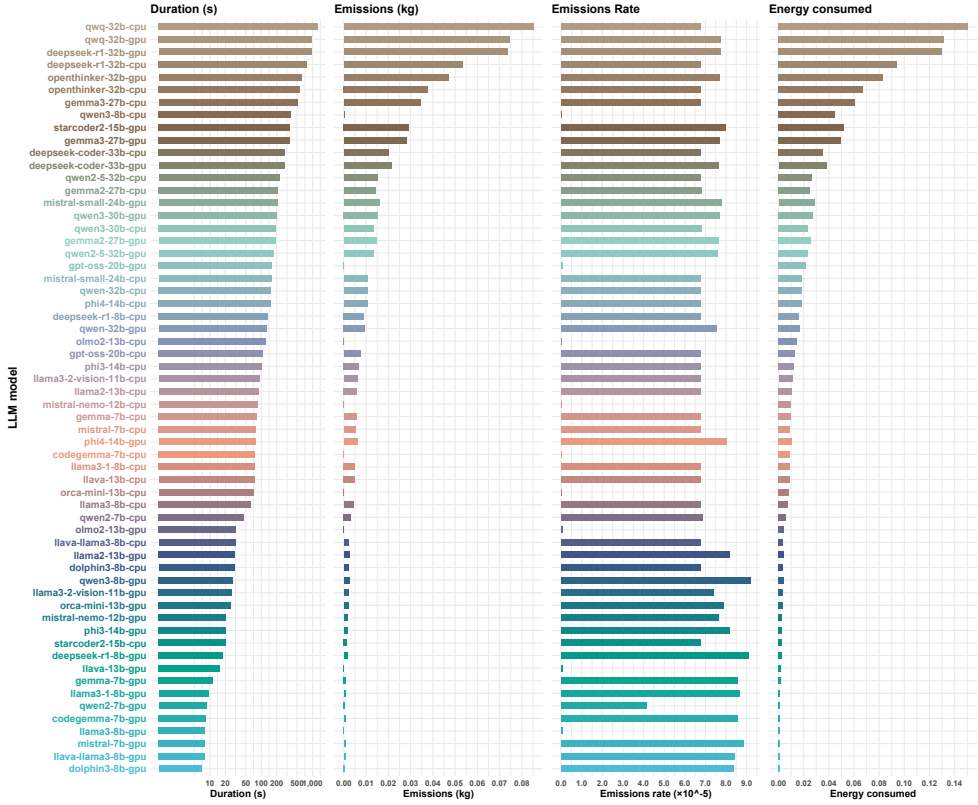
To standardize evaluation, a set of prompts was created that reflects common LLM use cases. Each experiment included two phases:

- **Warm-up phase:** 3 prompts (e.g. greetings, simple factual queries) used to preload model weights into memory and stabilize runtime performance.
- **Evaluation phase:** 5 prompts covering reasoning, summarization, question answering, and creative text generation. These were designed to stress models differently depending on size and architecture.

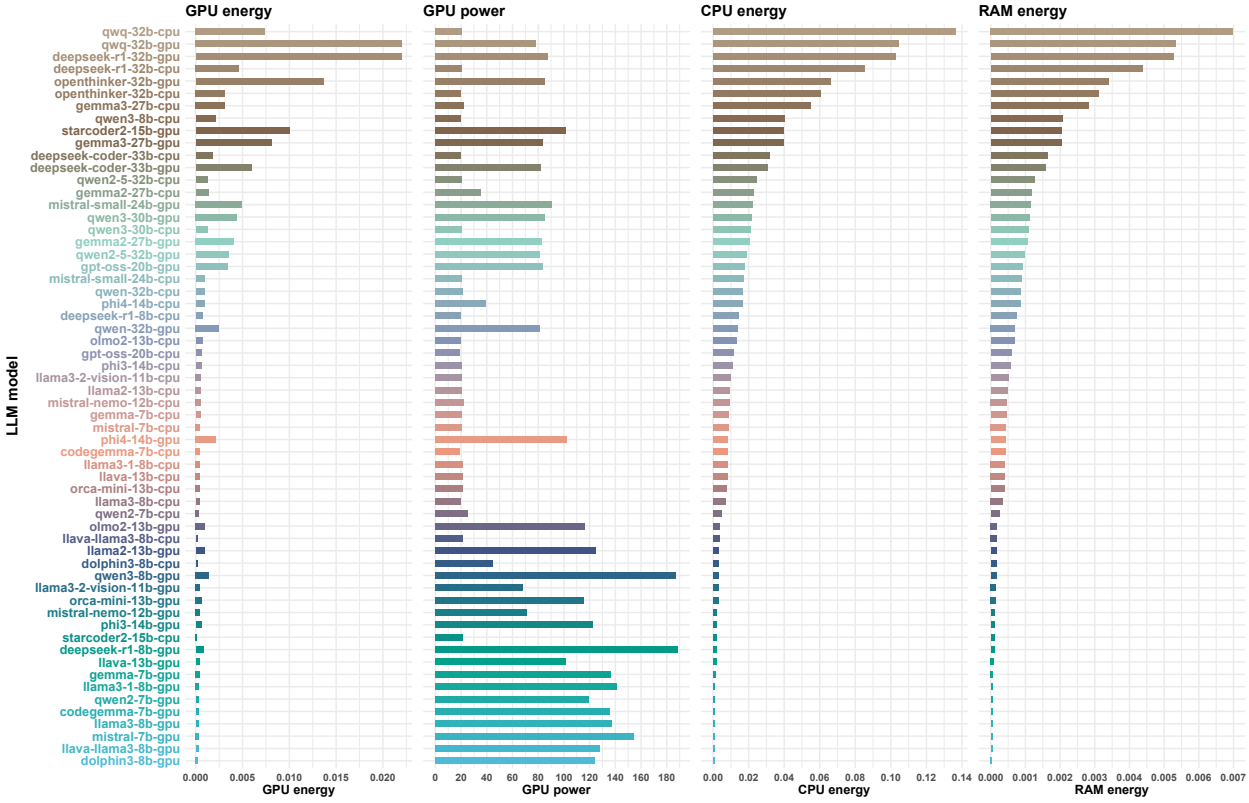
This follows approaches in MELODI [11], where workload diversity and warm-up cycles are critical for fair energy benchmarking.

D. Execution Framework

All experiments were automated using Python scripts that launched the appropriate Ollama model (CPU or GPU variant) via the `subprocess` module. Each inference sequence (3 warm-up + 5 evaluation prompts) was run twice, once with the CPU version and once with the GPU version. Results (outputs, runtime, and hardware metrics) were logged for later analysis.



(a) Duration, Emission, Emission Rate, and Energy Consumed comparison for all evaluated models.



(b) GPU Energy, GPU Power, CPU Energy, and RAM Energy comparison for all evaluated models.

Fig. 2: Code Carbon

E. Energy and Carbon Measurement

Two complementary approaches were employed to capture energy and emissions:

- **CodeCarbon**: a software-based tool that estimates energy consumption and CO_2 emissions using hardware profiles and regional carbon intensity data.
- **NVIDIA NVML (via `pynvml`)**: a hardware-level monitoring library that captures real-time GPU utilization, memory usage, and power draw.

The combination of CodeCarbon and NVML enables both *emission estimation* and *fine-grained hardware monitoring*, addressing limitations of relying on a single method.

F. Comparative Analysis

For each model and prompt set, the following were recorded:

- Average CPU utilization, GPU utilization, and memory usage.
- Energy consumption in kWh (from CodeCarbon and NVML-based tracking).
- Estimated CO_2 emissions in kg CO_2eq.

Comparisons were made across:

1) CPU vs. GPU inference for the same model.
2) Model size categories (7B/8B, 14B, 30B) to assess scaling effects.
3) Prompt categories (warm-up vs. evaluation) to evaluate the relationship between task complexity and energy footprint.

This methodology enables a comprehensive analysis of the trade-offs between performance, hardware utilization, and carbon efficiency in Ollama LLM inference.

IV. Results

Fig. 1 presents the results obtained using the NVML library, while Fig. 2 shows the results from the CodeCarbon library. In all histograms, the y-axis shows the evaluated LLM models, and the x-axis shows the metrics being assessed. The NVML metrics include Total Time (s), Energy (kWh), and CO_2 (kg). The CodeCarbon metrics include Duration (s), Emissions (kg), Emission Rate, Energy Consumed, GPU Energy, GPU Power, CPU Energy, and RAM Energy. A consistent color-coding scheme is applied to all models across all histograms, with each model represented by the same color for both the y-axis label and the corresponding histogram bars.

V. Discussion

A brief query like ”thanks” to an LLM typically uses only 0.005–0.015 Wh of server energy for models above 30B, roughly the energy needed to charge a small LED light for a few seconds, making a single interaction negligible. However, when scaled to billions of users (assuming 1 billion people each send one ”thanks” per day at an average of 0.01 Wh per query), the daily energy consumption reaches 10,000 kWh, enough to power approximately 350 average homes for a day. Over a month, this totals close to 300,000 kWh, and over a year, roughly 3,650,000 kWh. With an average grid CO_2 emission of 0.4 kg per kWh, this results in approximately 1.46 million kg (1,460 tons) of CO_2 annually, demonstrating that billions of tiny interactions can cumulatively have a significant environmental impact. This underscores why large-scale LLM use is tracked with tools like CodeCarbon and NVML to monitor energy use and emissions. A short query like ”thanks” also promotes good conversational norms and fosters a greater sense of connection to the technology. In all cases, whether measured with CodeCarbon or NVML, as model parameters increase, so do total execution time, energy consumption, and CO_2 emissions.

VI. Conclusion and Future Work

Benchmarking 31 Ollama models across 60 CPU and GPU scenarios with NVML and CodeCarbon shows that a short query to an LLM model above 30B consumes only 0.005–0.015 Wh of energy. Scaling to 1 billion daily queries at 0.01 Wh each results in approximately 10,000 kWh per day, 300,000 kWh per month, and 3,650,000 kWh per year, producing roughly 1.46 million kg (1,460 tons) of CO_2 annually. This demonstrates how billions of small interactions can add up to a significant environmental impact and why monitoring tools like CodeCarbon and NVML are vital.

We plan to benchmark energy consumption and carbon footprint during inference to guide hardware allocation and support more sustainable AI deployments, especially in power-limited edge and cloud environments.

References

[1] E. Strubell, A. Ganesh, and A. McCallum, “Energy and policy considerations for deep learning in nlp,” in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. ACL, 2019, pp. 3645–3650.

[2] Ollama, “Ollama: Run large language models locally,” 2024, [Online]. [Online]. Available: https://ollama.com

[3] CodeCarbon contributors, “Codecarbon: Track the carbon emissions of your code,” https://codecarbon.io, 2022, [Online].

[4] T. Miller *et al.*, “Melodi: Multi-dimensional evaluation of llms for energy efficiency,” *arXiv preprint arXiv:2407.16893*, 2024.

[5] L. Wang *et al.*, “Measuring and improving the energy efficiency of large language models inference,” *ResearchGate Preprint*, 2024. [Online]. Available: https://www.researchgate.net/publication/381199506

[6] R. Kumar *et al.*, “Carbontrak: Real-time tracking of llm inference emissions,” *arXiv preprint arXiv:2504.17674*, 2025.

[7] M. Gonzalez *et al.*, “Measuring the environmental impact of generative ai models,” *arXiv preprint arXiv:2505.09598*, 2025.

[8] W. Zhang, X. Liu, and Y. Chen, “Energy-efficient ai: Measurement, optimization, and sustainability,” *Energies*, vol. 18, no. 11, p. 2810, 2025.

[9] A. Smith *et al.*, “Carbon-aware large language model inference: Measurement and mitigation,” *arXiv preprint arXiv:2502.05610*, 2025.

[10] J. Anderson *et al.*, “Benchmarking ai energy efficiency with hybrid cpu/gpu workloads,” *arXiv preprint arXiv:2504.03360*, 2025.

[11] J. Doe *et al.*, “Melodi: Benchmarking energy and carbon efficiency of llm inference,” *arXiv preprint arXiv:2507.11417*, 2025.