Towards Sustainable Large Language Model Serving

Sophia Nguyen*
University of Waterloo
Waterloo, ON, CAN
s62nguye@uwaterloo.ca

Beihao Zhou*
University of Waterloo
Waterloo, ON, CAN
b72zhou@uwaterloo.ca

Yi Ding
Purdue University
West Lafayette, IN, USA
yiding@purdue.edu

Sihang Liu
University of Waterloo
Waterloo, ON, CAN
sihangliu@uwaterloo.ca

Abstract

In this work, we study LLMs from a carbon emission perspective, addressing both operational and embodied emissions, and paving the way for sustainable LLM serving. We characterize the performance and energy of LLaMA with 1B, 3B, and 7B parameters using two Nvidia GPU types, a latest-generation RTX6000 Ada and an older-generation T4. We analytically model operational carbon emissions based on energy consumption and carbon intensities from three grid regions — each representing a different energy source mix, and embodied carbon emissions based on chip area and memory size. Our characterization and modeling provide us with an in-depth understanding of the performance, energy, and carbon emissions of LLM serving. Our findings highlight the potential for optimizing sustainable LLM serving systems by considering both operational and embodied carbon emissions simultaneously.

CCS Concepts

• **Social and professional topics** → **Sustainability**; • **Computing methodologies** → *Machine learning*.

Keywords

Sustainability, Carbon Emissions, Machine Learning, Large Language Model, GPU

ACM Reference Format:
Sophia Nguyen, Beihao Zhou, Yi Ding, and Sihang Liu. 2024. Towards Sustainable Large Language Model Serving. In *Proceedings of 3rd Workshop on Sustainable Computer Systems (HotCarbon'24).* ACM, New York, NY, USA, 7 pages.

1 Introduction

Large language models (LLMs) have revolutionized numerous industries, including search engines, natural language processing, and programming [14, 26, 32]. Their widespread adoption has significantly increased energy demands. The cycle of LLM deployment includes training and inference/serving. Recent studies reveal that LLM serving now exceeds training in energy consumption, leading to substantial environmental impacts, notably in carbon emissions [2]. The carbon emissions stemming from the energy consumption of these applications are referred to as *operational carbon emissions* – typically quantified as carbon dioxide equivalent (CO_2eq). For instance, serving one prompt in ChatGPT generates more than 4 grams of CO_2eq [34] – over 20 times the carbon emission of a web search query [9].

The development of LLMs not only demands significant energy, but also requires substantial computing hardware resources. Training and serving LLMs require powerful GPUs and machine learning (ML) accelerators, such as NVIDIA HGX [21] and Google TPU [8]. These devices are typically equipped with large processor chips manufactured with advanced feature sizes (e.g., 5 nm CMOS) and high-bandwidth, high-capacity onboard memory, enabling efficient execution of LLMs. Nevertheless, the manufacturing process of the hardware entails substantial carbon emissions, known as *embodied carbon emissions*. These emissions are particularly prominent in the case of the latest high-performance devices, as demonstrated in prior studies [10, 13, 39].

To mitigate the environmental impact of LLM serving, a thorough understanding of its carbon emissions at a granular level, such as per-token level across various hardware platforms, is essential. However, this aspect has been under-explored. Some studies have only focused on performance metrics. For example, SplitWise [24] divides LLM serving into two phases—prefill and decode—and executes them on different GPUs; PowerInfer reduces LLMs' need for GPU onboard memory to better adapt LLMs to consumer-grade GPUs [27]. Another line of work examines the carbon emissions of traditional ML and web service applications [15, 25, 33]. However, LLMs differ significantly from these applications due to their highly compute and memory-intensive nature. Most recent work on carbon modeling, such as ACT [10], LLMCarbon [6], and carbon accounting for BLOOM [16], focus on end-to-end, high-level counting without profiling LLMs on real-world hardware platforms. Therefore, there is a gap in understanding the environmental impacts of LLM serving at a granular level.

To bridge this gap, we characterize LLMs through low-level profiling and modeling to understand the environmental impact of LLM serving. We examine the carbon emissions of three parameter sizes of Meta's LLaMA model [31] – 1B, 3B, and 7B – on two different types of GPUs: a latest-generation Nvidia RTX6000 Ada from 2023 and an older-generation Nvidia T4 released back in 2018. We first characterize the performance and energy consumption

*Both authors contributed equally and are listed in alphabetical order by last name.

HotCarbon'24, July 9, 2024, Santa Cruz, CA

Table 1: Specifications of GPUs in this work.

GPU Type	RTX6000 Ada	T4
Architecture	Ada Lovelace	Tesla
Chip Size	608.4 mm^2	545 mm^2
Technology Node	5 nm	12 nm
Memory	48 GB	16 GB
Thermal Design Power	300 W	70 W
Year	2023	2018
Embodied Carbon	26.6 kg	10.3 kg

of different *configurations* (i.e., model parameter size and batch size), including both per-query and per-token measurement in the prefill and decode phases of LLM serving execution (Section 2). Then, we model their operational and embodied carbon emissions, respectively (Section 3). For operational carbon, we analyze the emissions by considering three regions with different carbon intensities (CIs) [4], expressed as CO_2eq emissions per kWh of electricity consumption. For embodied carbon, we analytically model emissions based on chip area and memory size using existing modeling tools [10]. We conclude with a discussion of open questions and future directions based on our characterization and modeling findings (Section 4). We summarize our key findings as follows:

- The older and slower T4 has higher energy efficiency compared to the newer and faster RTX6000 Ada when processing less compute-intensive requests (e.g., batch size of 1), especially in the decode phase of LLM serving. This finding indicates that older hardware can be viable in specific configurations.
- A configuration (i.e., model parameter size and batch size) that achieves the highest throughput on a particular GPU type does not necessarily yield the highest energy efficiency, highlighting the complex tradeoffs in LLM serving design space exploration.
- With incorporation of both embodied and operational carbon, an energy-efficient configuration may not necessarily minimize carbon emissions. Instead, factors such as model parameters, batch size, and GPU platform collectively influence carbon emissions.
- Strategically using older GPUs like T4 could effectively reduce total carbon emissions by amortizing the embodied carbon emissions of GPUs over time.

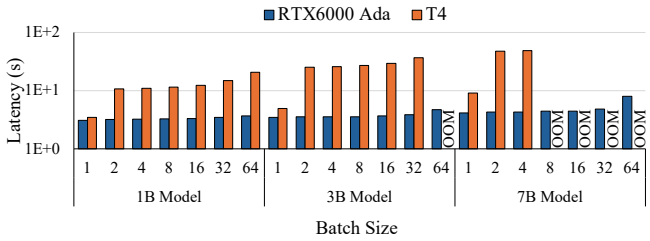
2 LLM Characterization

In this section, we characterize the performance and energy consumption of LLM serving.

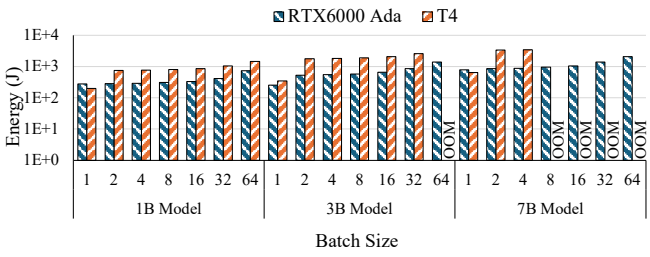
2.1 Methodology

This section describes the methodology of the characterizations.

Model and dataset. We characterize LLM serving using the widely-used LLaMA model [31] with 1B, 3B, and 7B parameters. For all experiments, we evaluate prompts from the Alpaca dataset [28]. While large-scale LLMs (e.g., those with hundreds of billions of parameters) are powerful, recent work has shown smaller models (e.g., 1B parameters) can achieve high accuracy and are especially valuable in many scenarios, e.g., speculative decoding [12], fine-tuned for specific tasks [11], working in cooperation with large models [35], and retrieval-augmented generation (RAG) [37].



(a) Latency.



(b) Energy.

Figure 1: Latency and energy consumption of RTX6000 Ada and T4 under different parameter sizes and batch sizes. "OOM"=out of memory.

GPU platforms. We evaluate LLMs on two GPU platforms: RTX6000 Ada and T4, as listed in Table 1. RTX6000 Ada is based on Nvidia's newer Ada Lovelace architecture that became available in 2023. It has a large 48 GB onboard memory and is manufactured with the advanced 5 nm node [30]. T4 is based on an older Nvidia Tesla architecture, equipped with a smaller 16 GB onboard memory that was released in 2018 and manufactured with an older 12 nm node [29]. RTX6000 Ada has a much higher 300 W thermal design power (TDP) as compared to T4's 70 W.

Measurement. As the output length varies by prompts and model sizes, we time LLM execution for 150 tokens and consider only prompts generating more than 150 tokens when comparing end-to-end latency and energy consumption. As LLMs typically run on GPUs and mainly utilize GPU resources, this study focuses on GPU power consumption. We use NVML [22] to measure the GPU's power every 100 ms. The energy consumption (E_{prompt}) of a prompt is the product of the average GPU power (P_{prompt}) and the execution time of the prompt (t_{prompt}):

$$E_{\text{prompt}} = P_{\text{prompt}} \cdot t_{\text{prompt}} \quad (1)$$

2.2 Latency vs. Energy Consumption

We first compare the latency and energy consumption across LLaMA models with different parameter sizes and GPU platforms. We report the average power consumption and median latency values for all prompts, during the execution of a specific model/batch size.

Latency. Figure 1a shows the median prompt processing latency (s) (log-scaled y-axis) across different model sizes from 1B to 3B and batch sizes from 1 to 64 (x-axis). Across all parameter sizes, the latency is higher in T4 compared to RTX6000 Ada. When the batch size is 1, T4 is 1.1×, 1.4×, and 2.2× slower than RTX6000 Ada when running 1B, 3B, and 7B-parameter models, respectively. T4

is more prone to latency increase than RTX6000 Ada as the batch size and parameter size of the model increases, as T4 is an older, lower-tier GPU compared to RTX6000 Ada. T4 is up to 11.4× slower than RTX6000 Ada when running the 7B model with a batch size of 4. For sufficiently large model and batch sizes, the 16 GB memory on T4 renders it unsuitable for outputting results, leading to "OOM" (out of memory) indications.

Energy consumption. Figure 1b shows the energy consumption (J) (log-scaled y-axis), which is calculated from the average power consumption when executing a specific model/batch size (x-axis) and the previous median latency, using Equation 1. For both RTX6000 Ada and T4, the energy consumption increases as the parameter and batch size increase, due to the higher compute intensity and longer execution time.

Surprisingly, despite RTX6000 Ada being a recently released GPU that is better optimized for LLMs, the energy consumption of T4 is 28 % and 20 % lower than RTX6000 Ada when executing a batch size of 1 in the 1B and 7B parameter models, respectively. And, the 3B model shows the energy consumption of T4 is not much higher than RTX6000 Ada (1.4×). The main reasons are two-fold. First, a batch size of 1 has a lower computational load. Even T4 is capable of executing such a load efficiently. Second, RTX6000 Ada's TDP is more than 4× higher than T4. Therefore, under such a light load, RTX6000 Ada is less power-efficient than T4.

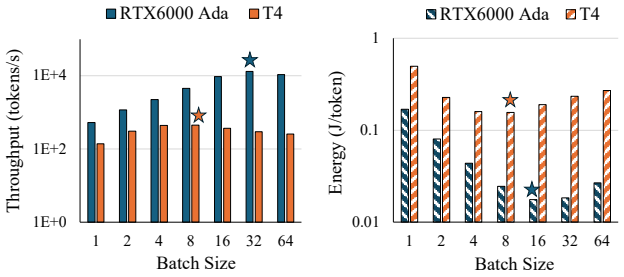
When the batch size increases, T4 consumes more energy than RTX6000 Ada (up to 2.9×) – in these more compute-intensive scenarios, the newer RTX6000 Ada has a significant advantage over the older and less performant T4. Despite the higher TDP, RTX6000 Ada executes the prompts an order of magnitude faster than T4 and thus reduces the per-prompt energy consumption.

Takeaway 1: RTX6000 Ada is faster than T4, regardless of the parameter size or batch size. However, when executing a batch size of 1, T4 may have an advantage over RTX6000 Ada, because T4 runs at a much lower power without major degradation in execution time. RTX6000 Ada becomes more energy-efficient under larger batch sizes, attributed to its faster processing speed.

2.3 Prefill and Decode Phases

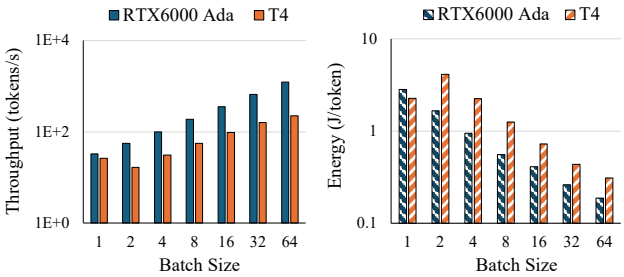
LLM execution consists of *prefill* and *decode* phases. The prefill phase involves processing the prompt, initializing the model state, and then generating the first token. Afterward, the decode phase generates the rest of the tokens in an auto-regressive fashion, until the termination token (or output token limit) is reached. Prior work has shown that these two phases have different performance characteristics [7, 24, 40], where the prefill phase is compute-bounded and more compute-intensive, and the decode phase is memory-bounded and less compute-intensive. In this experiment, we study not only the performance of both phases but also their energy consumption. Because T4 experiences OOM when running large batch and parameter sizes, we evaluate the LLaMA with 1B parameters.

Prefill Phase. We evaluate the prefill phase by measuring its throughput in processing input prompts (tokens/s) and the corresponding per-token energy consumption (J/token). Figure 2a shows the throughput and Figure 2b energy consumption (log-scaled y-axis) for different batch sizes (x-axis) when running on



(a) Throughput. **(b) Per-token energy.**

Figure 2: Latency and throughput in the *prefill* phase (1B-parameter LLaMA).



(a) Throughput. **(b) Per-token energy.**

Figure 3: Latency and throughput in the *decode* phase (1B-parameter LLaMA).

RTX6000 Ada and T4. We observe that RTX6000 Ada is more efficient from both a throughput and energy perspective during the prefill phase. This is because the prefill phase is compute-bounded, which prefers more powerful GPUs like RTX6000 Ada.

We also notice that the throughput reaches the peak when the batch size is 8 on T4 and 32 on RTX6000 Ada. Energy-wise, the per-token energy is lowest when batch size is 8 on T4 and 16 on RTX6000 Ada. RTX6000 Ada achieves higher throughput and lower per-token energy in a larger batch size than T4 because it can better handle compute-intensive scenarios. However, the batch size that achieves the highest throughput is not necessarily the same as which achieves the highest energy efficiency.

Decode Phase. We evaluate the decode phase by measuring the throughput (tokens/s) and per-token energy consumption (J/token). Figure 3a and Figure 3b show the throughput and per-token energy (log-scaled y-axis), respectively, in different batch sizes (x-axis). Comparing RTX6000 Ada and T4 in the decode phase, RTX6000 Ada always outperforms T4 in throughput but the difference is not as significant as the prefill phase, as the decode phase is memory-bound and less compute-intensive than the prefill phase. From an energy perspective, the per-token energy consumption of T4 is lower than RTX6000 Ada when the batch size is 1 — T4 consumes 20% less energy than RTX6000 Ada but only 24% lower throughput. As the batch size increases, RTX6000 Ada becomes more energy-efficient — up to 5.5× higher throughput and 2.4× lower energy consumption than T4. Although older GPUs like T4 are less powerful, their performance is not much worse than RTX6000 Ada while consuming less energy in a batch size of 1, similar to the observations in

Section 2.2. We further observe that the throughput and per-token energy consumption generally improve as the batch size increases, unlike the prefill phase.

Takeaway 2: Dividing LLM serving into prefill and decode phases reveals more energy optimization opportunities, including distributing them across different GPU platforms. In the prefill phase, the batch size with the highest throughput may not yield the best energy efficiency, indicating an intricate interplay between batch size, performance, and energy consumption. In the decode phase, the optimal tradeoff between energy efficiency and batch size also varies across GPU platforms.

3 Carbon Emission Analysis

In this section, we analyze the carbon emissions of LLM serving.

3.1 Methodology

We model and analyze the total carbon emissions for LLM serving, which include operational and embodied carbon emissions.

Operational Carbon. We model the operational carbon C_{op} based on the energy consumption and carbon intensity (CI) of the grid. The energy consumption of a prompt, E_{prompt}, is the energy consumed during the execution time, as we have shown in Section 2.2. Therefore, the operational carbon of a prompt is:

$$C_{\text{prompt,op}} = E_{\text{prompt}} \cdot CI \tag{2}$$

In this work, we study CIs of grids in three regions [4]: Québec (QC), California Independent System Operator (CISO), and PacifiCorp East (PACE). Table 2 lists their average CIs in 2023, which will be used in the rest of this paper. We select these regions due to their distinct energy mixes: QC relies heavily on renewable hydro and wind energy, CISO incorporates renewable solar energy alongside non-renewable natural gas, while PACE depends on non-renewable sources like coal and natural gas. Overall, a higher fraction of renewable energy sources leads to a lower CI.

Embodied Carbon. We model the embodied carbon (C_{em}) using the approach in ACT [10], by considering the processor chip areas and memory capacities from the specifications of RTX6000 Ada [30] and T4 [29]. The total embodied carbon emissions of both GPU types are listed in Table 1, which are close to a prior study on GPU embodied carbon emissions [13]. The ACT approach discounts the embodied carbon emissions by the ratio between total execution time (T) and GPU's lifetime (LT), i.e., $T/LT \cdot C_{\text{em}}$. Therefore, a prompt executed by the GPU that lasts t_{prompt} generates the embodied carbon emission of:

$$C_{\text{prompt,em}} = \frac{t_{\text{prompt}}}{LT} \cdot C_{\text{em}} \tag{3}$$

In this work, we assume a total GPU lifetime of 5 years – a typical lifetime of datacenter components [10, 23].

Total Carbon Emissions. The total carbon emission consists of embodied and operational carbon emissions. For a prompt that executes t_{prompt} time, its total carbon emission is:

$$\begin{aligned} C_{\text{prompt}} &= C_{\text{prompt,op}} + C_{\text{prompt,em}} \\ &= E_{\text{prompt}} \cdot CI + \frac{t_{\text{prompt}}}{LT} C_{\text{em}} \end{aligned} \tag{4}$$

Table 2: Carbon intensities (CIs) of three regions in 2023 [4].

Region	QC	CISO	PACE
State/Province	QC (Canada)	CA (USA)	WY, UT, AZ, NM, ID (USA)
Main Sources	Hydro, Wind	Gas, Solar	Coal, Gas
CI (g/kWh)	**31**	**262**	**647**

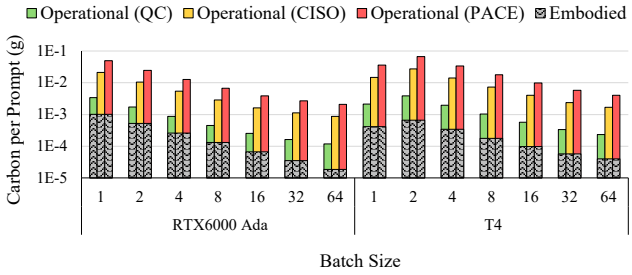


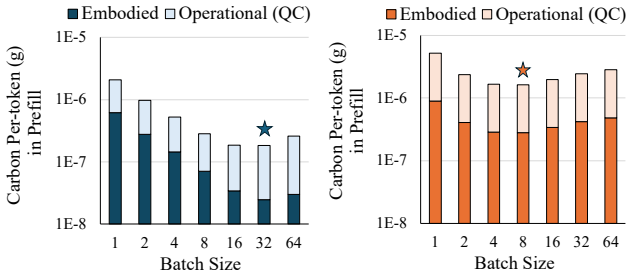
Figure 4: Per-prompt carbon emission under the QC, CISO, and PACE grids (1B-parameter LLaMA).

3.2 Carbon Emissions in Different Regions

Figure 4 shows the per-prompt embodied and operational carbon emissions (log-scaled y-axis) with different batch sizes (x-axis) in the three regions of Table 2: QC, CISO, and PACE.

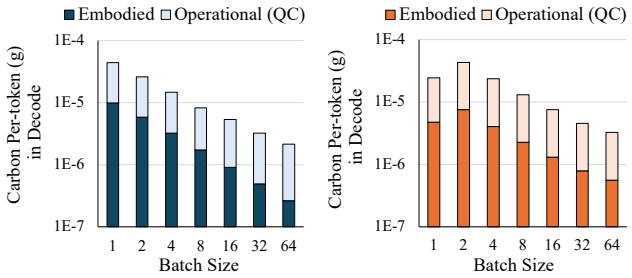
Section 2.2 demonstrates that the newer RTX6000 Ada GPU is more energy-efficient than the older T4 except when the batch size is 1. When considering CI, the same conclusion remains true within the same region. However, when comparing among regions, the older T4 may yield a lower operational carbon emission if it is located in a low-CI region. For instance, T4 in QC leads to lower operational carbon emissions compared to RTX6000 Ada in CISO or PACE, even when the batch size is 64. Especially after combining the embodied carbon of T4, the total carbon emission of T4 is lower than RTX6000 Ada in low-CI regions.

We observe that the embodied carbon, which is the bottom stack in the log-scaled bar chart, is a relatively small fraction of total carbon emissions across all regions. According to Equation 3, the embodied carbon is independent of CI because it is determined at manufacturing time. In contrast, the operational carbon varies and depends on the grid carbon intensity, according to Equation 2. Therefore, the operational carbon is lower when the same prompt is executed in a region with a lower grid carbon intensity. For T4, the embodied carbon comprises up to 19.7%, 2.8%, and 1.2% of the total carbon emissions in QC, CISO, and PACE, respectively. For RTX6000 Ada, the embodied carbon comprises up to 30.7%, 5.0%, and 2.1% in the same regions. This disparity arises from RTX6000 Ada's newer CMOS technology, larger chip area, and higher onboard memory capacity, resulting in significantly higher embodied carbon. These results indicate that embodied carbon emissions carry greater weight when GPUs are powered by grids with lower CIs.



(a) RTX6000 Ada (b) T4

Figure 5: Per-token carbon emission in the *prefill* phase under the CISO grid (1B parameters).



(a) RTX6000 Ada (b) T4

Figure 6: Per-token carbon emission in the *decode* phase under the CISO grid (1B-parameter LLaMA).

Takeaway 3: The balance between operational and embodied carbon emissions varies, depending on the CI. In high-CI regions, operational carbon makes up a significant portion of per-prompt carbon emissions, while in low-CI regions, embodied carbon becomes more prominent. Consequently, older and less powerful GPUs may be preferable in low-CI regions, whereas newer and more powerful GPUs are preferable in high-CI regions.

3.3 Carbon Emissions in Prefill/Decode Phases

We further study the operational and embodied carbon emissions in the prefill and decode phases of the 1B parameter version. We use the QC's CI value to calculate the operational carbon emissions.

Figure 5 depicts the per-token carbon emissions (g) in the prefill phase that consists of embodied (bottom stack) and operational (upper stack) carbon emissions, respectively. Overall, the result is similar to the per-token energy numbers in Figure 2b. However, when considering embodied carbon, the optimal batch size for minimizing carbon emissions on RTX6000 Ada varies. A batch size of 32 leads to the lowest per-token carbon emissions (0.18 µg/token), while a batch size of 16 consumes the lowest energy but does not achieve the lowest per-token carbon emission (0.19 µg/token).

Similarly, Figure 6 shows the per-token carbon emissions (g) in the decode phase. The result follows the same trend as the per-token energy numbers in Figure 3b with a smaller difference between T4 and RTX6000 Ada due to T4's lower embodied carbon emissions. We expect that for regions with lower CIs than QC (e.g., those

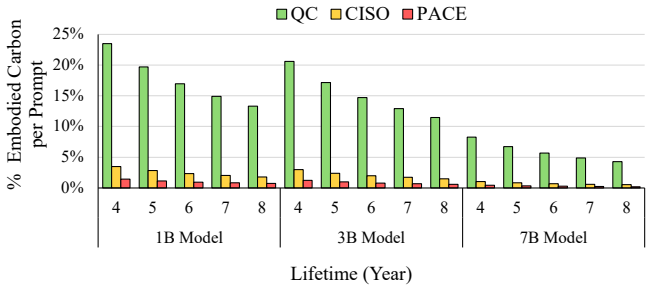


Figure 7: Embodied carbon emissions of T4 GPU under different expected lifetimes (batch size of 1).

powered by 100% renewable energy), older GPUs like T4 could potentially yield even lower total carbon emissions, provided latency requirements are met, owing to their lower embodied carbon.

Takeaway 4: Energy efficiency is not equivalent to carbon efficiency. To quantify environmental impacts, carbon-based metrics are more suitable than energy-based metrics. Factoring in embodied carbon emissions yields different optimal configurations compared to those that do not. Simultaneously accounting for both embodied and operational carbon reveals more optimization opportunities than considering either alone.

3.4 Impact of Extending GPU Lifetime

The previous analysis assumes a GPU lifespan of 5 years, which is typical for datacenter server components, as discussed in Section 3.1. Figure 7 depicts the anticipated percentage of embodied carbon emissions over the total per-token carbon emissions (y-axis) by sweeping the lifetime of the older T4 GPU from 4 to 8 years (x-axis) in different regions. We observe that a reduced lifetime increases the percentage of embodied carbon emissions, while an increased lifetime reduces this percentage, as it is amortized over a longer time span. In regions with lower CIs, this increase is more prominent as embodied carbon emissions occupy a large fraction of the total carbon emissions. The same trend holds true for the 1B, 3B, and 7B parameter models, except that the embodied carbon emissions take up a lower percentage in larger models, as they are more compute-intensive and lead to more operational carbon emissions.

Takeaway 5: Extending the lifetime of GPUs results in lower embodied carbon emissions, as they are amortized over a longer duration. This decrease in embodied carbon emissions from using older GPUs is particularly prominent in regions with lower CIs.

4 Future Directions

In this section, we discuss future directions based on the findings and insights from this work.

Designing ML software-hardware infrastructure with long-term sustainability. Latest-generation GPUs and ML accelerators are often in shortage. Our study has shown that old GPUs can be usable in many scenarios and may lead to even lower total carbon emissions, making old hardware reuse a viable approach to bridge the gap between the demand and limited supplies of new GPUs.

When designing hardware infrastructure for LLMs, it is critical to keep the whole lifetime in consideration, rather than optimizing for the current, specific applications. For example, the memory capacity can be a deal-breaker – latest LLMs do not fit into older GPUs with small onboard memory; an awareness of interconnect and extensibility can help improve the lifespan of these ML accelerators. On the other hand, to better leverage old GPUs or ML accelerators, we envision optimizations for old platforms throughout the LLM software stack, from LLM algorithms, to runtimes, to the ML compiler level. Together, we expect efforts in these directions to pave the way to achieve longevity in ML infrastructure.

Characterization of diverse LLM hardware platforms. This work focuses on GPUs, the dominant processors for LLM. As demand for ML infrastructure rises, particularly with the widespread adoption of LLMs, datacenters are increasingly integrating specialized ML accelerators, such as TPUs in Google cloud [8], MTIA accelerators in Meta cloud [20], and Trainum in AWS [1]. These ML accelerators possess distinct performance, energy consumption, and embodied carbon compositions compared to GPUs. Therefore, fully understanding the performance and carbon emission tradeoffs of specialized ML accelerators is critical to achieving the sustainability of LLM serving in the clouds. Moving forward, we anticipate the development of profiling frameworks tailored to assess the carbon emissions of diverse GPUs and ML accelerators.

CI-directed LLM serving. Our study has shown that the operational carbon emissions of LLM serving largely depend on the CI. When the CI is high, the operational carbon emissions dominate the total carbon emissions. Conversely, in cases of low CI, the embodied carbon emissions have a higher weight, making the use of older GPUs more beneficial. Prior work has demonstrated significant variability in carbon intensity across both temporal and spatial scales, as renewable energy sources are intermittent [17, 18, 38]. The flexibility in workload execution and carbon intensities presents opportunities for datacenter providers and users to schedule LLM workloads on GPUs that best match the current carbon intensity levels. Further, CI predictions [18, 19] can work collaboratively with the CI-directed scheduling strategy to make early scheduling decisions for LLM workloads and infrastructure.

Sustainable LLM training. While our focus lies on LLM serving, there is considerable potential for sustainable LLM training. Training LLMs is highly compute-demanding. For instance, Google trained their 540B-parameter PaLM model using 6144 TPU v4's[3]. Hence, there's a pressing need to reduce the massive operational carbon emissions of LLM training. Two directions can be explored. First, unlike latency-critical LLM serving, LLM training offers flexibility as it lacks strict deadlines, allowing for easy workload shifting to periods and regions with lower carbon intensity. Second, in fine-tuning scenarios that are less compute-intensive [5, 36], older and lower-end GPUs/accelerators can be better utilized for such tasks.

5 Conclusions

Serving LLMs on a large scale results in significant environmental impacts. This work studies both operational and embodied carbon emissions of LLaMA across various model parameter sizes and batch sizes using two Nvidia GPU types: RTX6000 Ada and T4. Our findings indicate several optimization opportunities for reducing both operational and embodied carbon emissions for LLM serving. We hope this work inspires system researchers and developers to consider environmental impacts for building next-generation sustainable LLM serving systems.

Acknowledgments

We thank the anonymous reviewers for their valuable feedback, and Tianyao Shi and Jiashu Zhang for proofreading. We acknowledge GCP and AWS for providing generous cloud research credits. This work is supported by the Natural Sciences and Engineering Research Council of Canada (NSERC) and the Undergraduate Research Assistantship (URA) program of the Cheriton School of Computer Science at the University of Waterloo.

References

[1] AWS. AWS Trainium. https://aws.amazon.com/machine-learning/trainium/.
[2] Andrew A Chien, Liuzixuan Lin, Hai Nguyen, Varsha Rao, Tristan Sharma, and Rajini Wijayawardana. Reducing the carbon impact of generative AI inference (today and in 2035). In *Proceedings of the 2nd Workshop on Sustainable Computer Systems (HotCarbon)*, pages 1–7, 2023.
[3] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. PaLM: Scaling language modeling with pathways. arXiv preprint arXiv:2204.02311, 2022.
[4] Electricity Maps ApS. Electricity maps. https://app.electricitymaps.com/, 2024.
[5] Saar Eliad, Ido Hakimi, Alon De Jagger, Mark Silberstein, and Assaf Schuster. Fine-tuning giant neural networks on commodity hardware with automatic pipeline model parallelism. In *USENIX Annual Technical Conference (ATC)*, 2021.
[6] Ahmad Faiz, Sotaro Kaneda, Ruhan Wang, Rita Chukwunyere Osi, Prateek Sharma, Fan Chen, and Lei Jiang. LLMCarbon: Modeling the end-to-end carbon footprint of large language models. In *The Twelfth International Conference on Learning Representations (ICLR)*, 2024.
[7] Bin Gao, Zhuomin He, Puru Sharma, Qingxuan Kang, Djordje Jevdjic, Junbo Deng, Xingkun Yang, Zhou Yu, and Pengfei Zuo. AttentionStore: Cost-effective attention reuse across multi-turn conversations in large language model serving. arXiv preprint arXiv:2403.19708, 2024.
[8] Google. Accelerate AI development with Google cloud TPUs. https://cloud.google.com/tpu.
[9] Sarah Griffiths. Why your internet habits are not as clean as you think. https://www.bbc.com/future/article/20200305-why-your-internet-habits-are-not-as-clean-as-you-think, 2020.
[10] Udit Gupta, Mariam Elgamal, Gage Hills, Gu-Yeon Wei, Hsien-Hsin S. Lee, David Brooks, and Carole-Jean Wu. ACT: Designing sustainable computer systems with an architectural carbon modeling tool. In *Proceedings of the 49th Annual International Symposium on Computer Architecture (ISCA)*, 2022.
[11] Gurusha Juneja, Subhabrata Dutta, Soumen Chakrabarti, Sunny Manchanda, and Tanmoy Chakraborty. Small language models fine-tuned to coordinate larger language models improve complex reasoning. arXiv preprint arXiv:2310.18338, 2024.
[12] Yaniv Leviathan, Matan Kalman, and Yossi Matias. Fast inference from transformers via speculative decoding. In *Proceedings of the 40th International Conference on Machine Learning (ICML)*, 2023.
[13] Baolin Li, Rohan Basu Roy, Daniel Wang, Siddharth Samsi, Vijay Gadepally, and Devesh Tiwari. Toward sustainable HPC: Carbon footprint estimation and environmental implications of HPC systems. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis (SC)*, 2023.
[14] Jiawei Liu, Chunqiu Steven Xia, Yuyao Wang, and Lingming Zhang. Is your code generated by ChatGPT really correct? rigorous evaluation of large language models for code generation. *Advances in Neural Information Processing Systems (NeurIPS)*, 36, 2024.

[15] Alexandra Sasha Luccioni and Alex Hernandez-Garcia. Counting carbon: A survey of factors influencing the emissions of machine learning. arXiv preprint arXiv:2302.08476, 2023.

[16] Alexandra Sasha Luccioni, Sylvain Viguier, and Anne-Laure Ligozat. Estimating the carbon footprint of BLOOM, A 176B parameter language model. *J. Mach. Learn. Res.*, 24(1), mar 2024.

[17] Diptyaroop Maji, Ben Pfaff, Vipin P R, Rajagopal Sreenivasan, Victor Firoiu, Sreeram Iyer, Colleen Josephson, Zhelong Pan, and Ramesh K Sitaraman. Bringing carbon awareness to multi-cloud application delivery. In *Proceedings of the 2nd Workshop on Sustainable Computer Systems (HotCarbon)*, 2023.

[18] Diptyaroop Maji, Prashant Shenoy, and Ramesh K. Sitaraman. CarbonCast: Multi-day forecasting of grid carbon intensity. In *Proceedings of the 9th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation (BuildSys)*, 2022.

[19] Diptyaroop Maji, Ramesh K. Sitaraman, and Prashant Shenoy. DACF: Day-ahead carbon intensity forecasting of power grids using machine learning. In *Proceedings of the Thirteenth ACM International Conference on Future Energy Systems (e-Energy)*, e-Energy '22, 2022.

[20] Meta. MTIA v1: Meta's first-generation AI inference accelerator. https://ai.meta.com/blog/meta-training-inference-accelerator-AI-MTIA/, 2023.

[21] Nvidia. NVIDIA HGX AI Supercomputer. https://www.nvidia.com/en-us/data-center/hgx/, 2024.

[22] Nvidia. NVIDIA management library (NVML). https://developer.nvidia.com/management-library-nvml, 2024.

[23] George Ostrouchov, Don Maxwell, Rizwan A. Ashraf, Christian Engelmann, Mallikarjun Shankar, and James H. Rogers. GPU lifetimes on titan supercomputer: Survival analysis and reliability. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis (SC)*, 2020.

[24] Pratyush Patel, Esha Choukse, Chaojie Zhang, Íñigo Goiri, Aashaka Shah, Saeed Maleki, and Ricardo Bianchini. Splitwise improves GPU usage by splitting LLM inference phases. In *International Symposium on Computer Architecture (ISCA)*, 2024.

[25] David Patterson, Joseph Gonzalez, Urs Hölzle, Quoc Le, Chen Liang, Lluis-Miquel Munguia, Daniel Rothchild, David R. So, Maud Texier, and Jeff Dean. The carbon footprint of machine learning training will plateau, then shrink. *Computer*, 2022.

[26] Yongliang Shen, Kaitao Song, Xu Tan, Dongsheng Li, Weiming Lu, and Yueting Zhuang. HuggingGPT: Solving AI tasks with ChatGPT and its friends in hugging face. *Advances in Neural Information Processing Systems (NeurIPS)*, 36, 2024.

[27] Yixin Song, Zeyu Mi, Haotong Xie, and Haibo Chen. PowerInfer: Fast large language model serving with a consumer-grade GPU. arXiv preprint arXiv:2312.12456, 2023.

[28] Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. Stanford Alpaca: An instruction-following LLaMA model. https://github.com/tatsu-lab/stanford_alpaca, 2023.

[29] TechPowerUP. NVIDIA Tesla T4. https://www.techpowerup.com/gpu-specs/tesla-t4.c3316/, 2018.

[30] TechPowerUP. NVIDIA RTX 6000 Ada Generation. https://www.techpowerup.com/gpu-specs/rtx-6000-ada-generation.c3933/, 2023.

[31] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. LLaMA: Open and efficient foundation language models. arXiv preprint arXiv:2302.13971, 2023.

[32] Tu Vu, Mohit Iyyer, Xuezhi Wang, Noah Constant, Jerry Wei, Jason Wei, Chris Tar, Yun-Hsuan Sung, Denny Zhou, Quoc Le, et al. FreshLLMs: Refreshing large language models with search engine augmentation. *arXiv preprint arXiv:2310.03214*, 2023.

[33] Jaylen Wang, Udit Gupta, and Akshitha Sriraman. Peeling back the carbon curtain: Carbon optimization challenges in cloud computing. In *Workshop on Sustainable Computer Systems (HotCarbon)*, 2023.

[34] Vinnie Wong. Gen AI's environmental ledger: A closer look at the carbon footprint of ChatGPT. https://piktochart.com/blog/carbon-footprint-of-chatgpt/, 2023.

[35] Canwen Xu, Yichong Xu, Shuohang Wang, Yang Liu, Chenguang Zhu, and Julian McAuley. Small models are valuable plug-ins for large language models. arXiv preprint arXiv:2305.08848, 2023.

[36] Zhengmao Ye, Dengchun Li, Jingqi Tian, Tingfeng Lan, Jie Zuo, Lei Duan, Hui Lu, Yexi Jiang, Jian Sha, Ke Zhang, and Mingjie Tang. ASPEN: High-throughput LoRA fine-tuning of large language models with a single GPU. arXiv preprint arXiv:2312.02515, 2023.

[37] Tianjun Zhang, Shishir G. Patil, Naman Jain, Sheng Shen, Matei Zaharia, Ion Stoica, and Joseph E. Gonzalez. RAFT: Adapting language model to domain specific RAG. arXiv preprint arXiv:2403.10131, 2024.

[38] Xiaoyang Zhang and Dan Wang. A GNN-based day ahead carbon intensity forecasting model for cross-border power grids. In *Proceedings of the 14th ACM International Conference on Future Energy Systems (e-Energy)*, e-Energy '23, 2023.

[39] Xiaoyang Zhang, Yijie Yang, and Dan Wang. Embodied carbon accounting through spatial-temporal embodied carbon models. arXiv preprint arXiv:2312.06364, 2023.

[40] Yinmin Zhong, Shengyu Liu, Junda Chen, Jianbo Hu, Yibo Zhu, Xuanzhe Liu, Xin Jin, and Hao Zhang. DistServe: Disaggregating prefill and decoding for goodput-optimized large language model serving. arXiv preprint arXiv:2401.09670, 2024.