Unveiling Environmental Impacts of Large Language Model Serving: A Functional Unit View

Anonymous ACL submission

Abstract

Large language models (LLMs) offer powerful capabilities but come with significant environmental costs, particularly in carbon emissions. Existing studies benchmark these emissions but lack a standardized basis for comparison across models. To address this, we introduce the concept of a *functional unit* (FU) and develop FUEL, the first FU-based framework for evaluating LLM serving's environmental impact. Through case studies on model size, quantization, and hardware, we uncover key trade-offs in sustainability. Our findings highlight the potential for reducing carbon emissions by optimizing model selection, deployment strategies, and hardware choices, paving the way for more sustainable AI infrastructure.

1 Introduction

Large language models (LLMs) have been widely adopted in various industries due to their ability to perform complex language tasks (Vu et al., 2024; Shen et al., 2024; Liu et al., 2024c). However, LLM serving comes with significant environmental impacts, particularly in terms of carbon emissions. For instance, processing a single prompt on ChatGPT produces over 4 grams of CO_2eq (Wong, 2023), which is over 20x the carbon emissions generated by a web search query (Griffiths, 2020).

Recent studies have benchmarked the carbon emissions of LLM serving by analyzing performance (e.g., throughput, latency) and energy consumption, then modeling carbon emissions under varying conditions such as request rate, and input/output length (Nguyen et al., 2024; Li et al., 2024d; Shi et al., 2024; Li et al., 2024b). However, these efforts have two limitations: **(1)** they focus on individual LLMs rather than cross-model comparisons, and **(2)** they lack a standardized basis for fair carbon emission comparisons. These gaps limit the broader applicability and fairness of their analyses.

Building on principles from life cycle assessment in environmental sustainability (Klöpffer and Grahl, 2014), we address these two limitations by introducing the concept of a *functional unit* (FU) as a standardized basis for comparing LLMs. In LLM serving, an FU represents a token generation defined by workload intensity, performance, and quality constraints. Using this, we develop FUEL, a **F**unctional **U**nit-based **E**valuation framework for evaluating the environment impact of **L**LMs. To demonstrate its effectiveness and generalizability, we conduct three case studies exploring model size, quantization, and hardware. Our key insights for building sustainable LLM serving systems include:

- *Model size:* Larger models are greener in high output quality and low request rate, while smaller models excel as the request rate increases.
- *Quantization:* Quantization significantly lowers carbon emissions, especially for larger models.
- *Hardware:* Newer hardware offers better performance but is not always greener due to higher embodied carbon. Older hardware can lower carbon emissions while meeting quality and performance constraints.

The contributions of this paper are:

- Introducing and defining FU for LLM serving from environmental sustainability.
- Developing FUEL, the first FU-based framework for assessing the environmental impact of LLM serving.
- Conducting case studies on model size, quantization, and hardware impact on carbon emissions.

2 Related Work

Environmental impact of LLM serving. Researchers have recognized the environmental impact of LLM serving and explored it through modeling and profiling (Ding and Shi, 2024). Modeling efforts include LLMCarbon (Faiz et al., 2024) and LLMCO2 (Fu et al., 2024), which provide end-

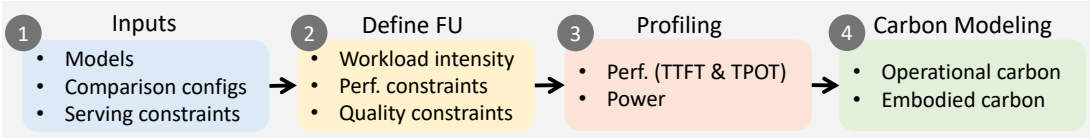

Figure 1: Overview of FUEL framework.

to-end carbon modeling frameworks, while LLM-Campass (Zhang et al., 2024) focuses on hardware evaluation for LLM workloads. Profiling studies have run various LLM serving models across different hardware and QPS settings (Nguyen et al., 2024; Li et al., 2024c; Patel et al., 2024a), with GreenLLM (Shi et al., 2024) and Sprout (Li et al., 2024b) optimizing carbon emissions based on their profiling. However, none of these studies take a functional unit perspective as we do in this work.

LLM serving optimization. Prior work on LLM serving has primarily focused on optimizing performance and energy efficiency. Performance improvements can be categorized into model-level and system-level techniques. Model-side optimizations include quantization (Lin et al., 2024; Frantar et al., 2022), sparsification (Frantar and Alistarh, 2023), and speculative decoding (Leviathan et al., 2023). System-side approaches involve memory management (Kwon et al., 2023), batching (Agrawal et al., 2024; Yu et al., 2022), and kernel optimizations (Dao et al., 2022). Additionally, efforts to enhance energy efficiency include solutions like Splitwise (Patel et al., 2024b) and DynamoLLM (Stojkovic et al., 2024). However, they have largely overlooked quality constraints.

3 The Framework FUEL

We present FUEL, a **F**unctional **U**nit-based **E**valuation framework for evaluating the environment impact of **L**LMs. FUEL enables a systematic and comprehensive analysis across various comparison configurations (e.g., model size, quantization, and hardware). Inspired by life cycle assessment in environmental sustainability (Klöpffer and Grahl, 2014), the key insight is to establish a functional unit as a standardized basis for comparison. In LLM serving, a *functional unit* (FU) represents a token characterized by its serving constraints during generation. In the FUEL framework, we compare the environmental impact of tokens generated by different model configurations with the same performance and quality constraints.

Figure 1 illustrates the four key steps of FUEL. First, FUEL identifies the inputs, including models, comparison configurations, and serving constraints. Next, it defines the FU based on these inputs. Then, experiments are conducted to profile performance and energy consumption. Finally, FUEL quantifies the environmental impact — focusing on carbon emissions in this work — using the collected data. Next, we will introduce each step in detail.

3.1 Step 1: Inputs

The inputs to FUEL include three key components:

- *Models:* The LLMs being compared, which can be different versions within the same model family or models from different families.
- *Comparison configurations:* The primary parameter that varies across comparisons. This paper focuses on three configurations: model size, quantization, and hardware.
- *Serving constraints:* The standardized basis for comparison, including workload intensity, performance constraint, and quality constraint. These constraints are critical in defining the FU.

3.2 Step 2: Define Functional Unit

In LLM serving, a *functional unit* represents a token characterized by its workload intensity, performance, and quality constraints during generation.

Workload intensity. FUEL defines workload intensity as the request rate (QPS), measuring incoming user requests per second (req/s).

Performance constraint. FUEL evaluates performance using two widely adopted metrics: Time-to-First-Token (TTFT) and Time-Per-Output-Token (TPOT). TTFT reflects how quickly the system responds to a new request by generating the first token, while TPOT quantifies the time per output token during decoding. Following prior work Liu et al. (2024b), FUEL sets a TTFT requirement of 1 second and a TPOT threshold of 200 ms, aligning with average human reading speed to ensure a smooth user experience.

Quality constraint. Quantitatively assessing output quality is challenging. While prior works (Zhong et al. (2022); Yuan et al. (2021); Jiang et al. (2023)) have introduced various methods,

they depend on either specific datasets or the need for reference answers. After evaluating multiple quality metrics, we adopt the reward model (Liu et al., 2024a), a common approach in reinforcement learning from human feedback training (Ouyang et al., 2022). Our experiments show that the reward model's scores align most closely with human preferences and effectively differentiate outputs across models. Using the reward model's score, we define *Qscore* as a measure of output quality, where a higher Qscore reflects better quality and indicates that the output meets a certain quality threshold.

An example of FU definition. Based on these serving constraints, we define an example FU below:

> A token generated by an LLM at a request rate of 5 req/s, with a Qscore of 10, and performance constraints of 1s TTFT and 200ms TPOT.

3.3 Step 3: Profiling

FUEL profiles performance (TTFT and TPOT) and energy consumption by running LLMs under different configurations, based on the inputs given to FUEL and the specified workload intensity. During profiling, Qscore is collected using an off-the-shelf reward model to evaluate output quality. For NVIDIA GPUs and Intel CPUs, power is measured every 200ms using NVIDIA (`pynvml`) and Intel (`psutil`) APIs for energy modeling, respectively.

3.4 Step 4: Carbon Modeling

Unlike prior work that profiles performance and energy without considering serving constraints, FUEL defines and calculates *carbon emission per FU* (CFU), measuring the emissions of FUs that meet certain serving constraints. Formally,

$$\text{CFU} = \frac{\text{Total carbon emissions for all tokens}}{\text{N}_\text{f}},$$

$$N_f = \sum_{i=1}^{N} \mathbb{I}(\text{Q}_i \geq \alpha) \cdot \mathbb{I}(\text{TTFT}_i \leq \beta) \cdot \mathbb{I}(\text{TPOT}_i \leq \gamma),$$

where N is the total number of output tokens, N_f is the total number of tokens considered FUs, Q is the Qscore, α, β, and γ are the constraints for Qscore, TTFT, and TPOT, respectively. Note that we consider a token to meet the Qscore requirement if its corresponding response does, as Qscore is defined at the response level. Next, we describe how to calculate carbon emissions.

Carbon emission calculation. Following prior work (Nguyen et al., 2024; Li et al., 2024d; Shi et al., 2024; Ding and Shi, 2024), total carbon emissions in LLM serving include operational carbon emission C_{op} and embodied carbon emissions C_{em}. We now describe how to calculate each.

- *Operational carbon* is calculated as the product of the energy consumed, E_{op}, and the carbon intensity of the energy source (`CI`). Carbon intensity is defined as the amount of $CO_{2\text{eq}}$ emitted per kilowatt-hour (kWh) of electricity used (Maji et al., 2022; Li et al., 2024a). The operational carbon emission is thus given by:

$$C_{\text{op}} = E_{\text{op}} \cdot \texttt{CI} \tag{1}$$

- *Embodied carbon* of a hardware device is determined by factors such as processor chip area and memory capacity (Gupta et al., 2022; Faiz et al., 2024). The detail of modeling the total embodied carbon of a hardware device is in Appendix B. The embodied carbon emission of an LLM execution over time t is calculated by amortizing the hardware's total embodied carbon $C_{\text{em,total}}$ over its lifetime (`LT`), typically 5 to 7 years (Ostrouchov et al., 2020). Thus, the embodied carbon for a time period t is given by:

$$C_{\text{em}} = \frac{t}{\texttt{LT}} \cdot C_{\text{em,total}} \tag{2}$$

- *Total carbon* is thus given by:

$$C_{\text{total}} = E_{\text{op}} \cdot \texttt{CI} + \frac{t}{\texttt{LT}} \cdot C_{\text{em,total}} \tag{3}$$

3.5 Summary and Implementation

FUEL provides a systematic framework for evaluating the environmental impact of LLM serving, using FU as a comparison basis. To demonstrate its effectiveness and generalizability, we will present three case studies exploring different comparison configurations: model size (§4), quantization (§5), and hardware (§6). For broadly applicable insights, we focus on two widely used model families, Qwen2.5 (Qwen et al., 2025) and Llama2 (Touvron et al., 2023), and conduct experiments using the open-source LLM serving platform vLLM (Kwon et al., 2023). We use a carbon intensity of 518 gCO_2eq, the 12-month average of our server's region, to calculate operational carbon emissions. All experiments were conducted in a single run with the LLM temperature set to 0 to minimize output randomness. We use the NewsQA (Trischler et al., 2016) summarization dataset for main results, as it tests language understanding without extra context. Results on other datasets are in the Appendix.

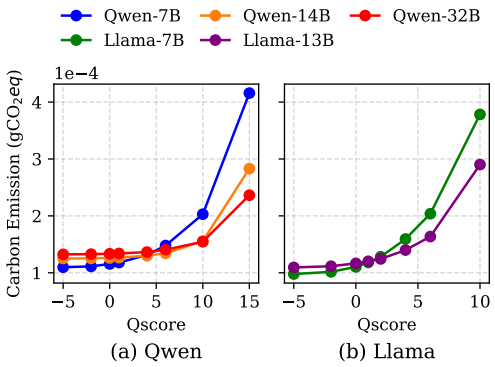

Figure 2: Carbon emission per FU for different model sizes across Qscores at QPS=1 req/s.

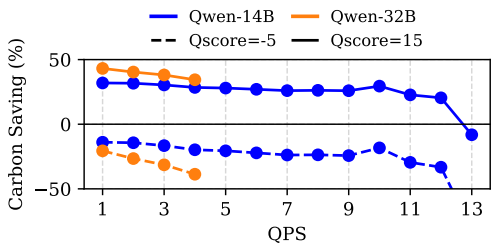

Figure 3: Carbon savings of Qwen 14B and 32B compared to 7B with Qscore low (-5) and high (15). Data for Qwen 32B are missing at QPS > 4 req/s, as larger models cannot serve intensive workloads.

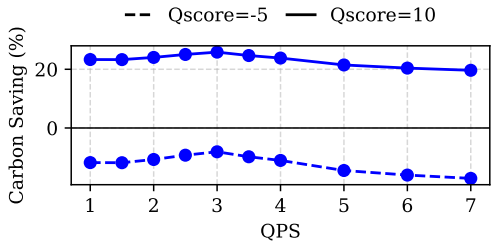

Figure 4: Carbon savings of Llama 13B compared to 7B with Qscore low (-5) and high (10).

4 Case Study: Model Size

In this section, we use FUEL to examine the environmental impact of model size on LLM serving.

4.1 Evaluation Methodology

Setup. We evaluate various model sizes from two LLM families—Qwen2.5 (7B, 14B, 32B) and Llama2 (7B, 13B)—on an NVIDIA H100 GPU paired with an Intel Xeon 8480+ CPU.

Benchmarking configurations. To assess how model sizes affect the environmental impact—or how "green" 🌿 each model is in terms of carbon efficiency—we evaluate a range of FUs by adjusting serving constraints. QPS is from 1 to 20 req/s. The Qscore ranges are set to [-5, 15] for Qwen and [-5, 10] for Llama, based on the Qscore distribution of each model family (Figure 21 in Appendix C.1). These ranges ensure broad coverage while providing sufficient outputs across model sizes that meet quality requirements. TTFT is at 1s and TPOT is at 200ms to align with human reading speed.

4.2 Evaluation Results

Question 1: *Are smaller models always greener?* We first investigate whether smaller models are always greener. Figure 2 shows carbon emissions per FU across model sizes under different Qscore settings at QPS = 1 req/s. We choose a relatively low QPS to ensure all models generate enough tokens without violating performance constraints. The results indicate that the answer is **no**.

For Qwen, at a low Qscore of -5, smaller models emit less carbon. However, as Qscore increases, carbon emissions increase for all model sizes, with smaller models increasing at a faster rate. When Qscore exceeds 5, the smallest 7B model becomes the highest emitter. At Qscore 15, the 32B model has the lowest emissions, while the 7B model emits over 1.8× more. A similar trend is seen in Llama, where larger models become greener as quality requirement rise. We confirm that larger models produce higher-quality outputs with higher Qscores in Figure 21 in Appendix C.1. This underscores the need to balance model size and output quality for lower environmental impact.

Question 2: *When are larger models greener?* To examine when larger models become greener, we set FUs with a broader QPS range and two quality requirements: low (Qscore = -5) and high (Qscore = 15 for Qwen, 10 for Llama). Figure 3 shows that for Qwen, larger models (14B and 32B) save more carbon compared to the 7B model under high Qscore, with the 32B saving over 40%. However, under a low-quality requirement (Qscore = -5), larger models offer no advantage. A similar trend is seen for Llama, where the 13B model saves over 20% carbon compared to the 7B model at high quality. Thus, **larger models become greener when output quality requirements are high**.

To explain the carbon savings shift with varying QPS, we analyze its impact on *service level objective (SLO) attainment*, which refers to meeting TTFT and TPOT constraints. In Figure 5, we observe that once QPS exceeds a certain threshold, SLO attainment drops, as the system becomes saturated. This explains why larger models can

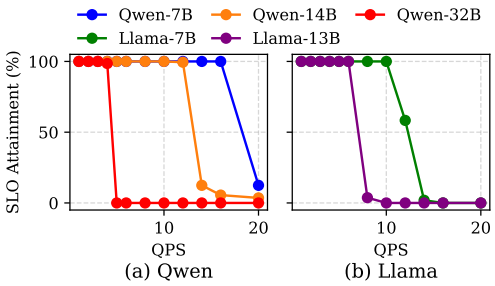

Figure 5: SLO attainment of Qwen and Llama families across QPS range.

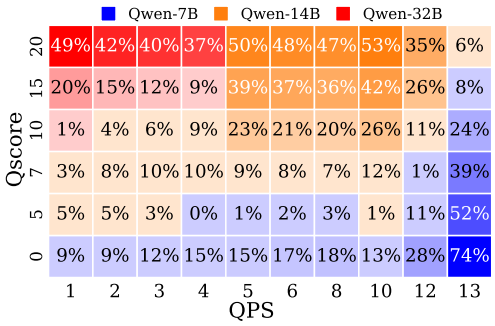

Figure 6: Comparison of Qwen 7B, 14B, and 32B in FUEL. Tile colors indicate the model size with the lowest carbon per FU. Tile values are carbon savings (%) of greenest model size compared to the second greenest.

be greener at lower QPS: they meet performance constraints while producing higher-quality output.

Question 3: *Does a universal greenest model size exist?*
The answer is **no**. Figure 6 shows the relative carbon savings of Qwen 7B, 14B, and 32B across various QPS and Qscore values. No model size consistently has the lowest carbon emissions. At low QPS (1-4 req/s) with high Qscore, Qwen 32B can save up to 49% in carbon emissions compared to the second greenest. However, as QPS increases, the 32B fails to meet the performance constraints, making the 14B the greenest. When the quality requirement is low (Qscore = 0), the 7B model is always the greenest, especially at high QPS.

Takeaway 1: Larger models are greener under high-quality, low-QPS conditions. Smaller models become greener as QPS increases. No single model size is the greenest across all scenarios.

5 Case Study: Quantization

In this section, we explore how quantization affects the environmental impact of LLM serving. By reducing model weight and activation precision, quantization significantly decreases model size. For example, 4-bit quantization cuts model size by 4× compared to FP16. This reduction lowers memory usage and computational costs while maintaining accuracy. Using FUEL, we investigate whether quantization, especially weight-only (Lin et al., 2024) and activation (Frantar et al., 2022) quantization techniques, can improve carbon efficiency while maintaining output quality.

5.1 Evaluation Methodology

Setup. We evaluate two widely used quantization methods: 4-bit AWQ (Lin et al., 2024) (weight-only) and W8A8 (Frantar et al., 2022) (INT8 quantization for both weights and activations). We evaluate Qwen2.5 (7B, 14B, 32B) and Llama2 (7B, 13B) on an NVIDIA H100 GPU with an Intel Xeon 8480+ CPU. Qwen provides an official AWQ version, while Llama's AWQ is from Hugging Face (TheBloke, 2023b,a). For W8A8, we quantize the models using LLM Compressor (vLLM Project, 2023), an open-source library designed for vLLM.

Benchmarking configurations. Same as in §4.

5.2 Evaluation Results

Question 1: *Is weight-only quantization always greener?*
The answer is **no**. Figure 7 shows the relative carbon emission savings per FU for AWQ compared to the FP16 version of Qwen under high (10) and low (-5) Qscores. Overall, AWQ's carbon savings decline as QPS increases. For the 7B model, AWQ consistently reduces emissions, even under high Qscore. At QPS = 1 req/s and Qscore = 10, AWQ cuts emissions by over 20% compared to FP16. This is because AWQ slightly increases the output quality of 7B (Table 3 in Appendix D), resulting in an increased number of FUs. On the other hand, the 14B model shows positive carbon savings at low Qscore (-5) but negative savings at high Qscore (10). The 32B model never achieves positive carbon savings, regardless of Qscore. We observe a similar trend for Llama in Figure 8. As QPS increases, the carbon savings of AWQ over FP16 decline and can even become negative at high QPS.

To understand why AWQ does not always outperform FP16 in carbon savings, we analyze its impact on TTFT and TPOT speedup. Figures 9 and 10 show that TPOT sees some speedup at low QPS but slows down at high QPS, while TTFT is always slower than FP16. This is because quantization reduces weight size, but weights are dequantized

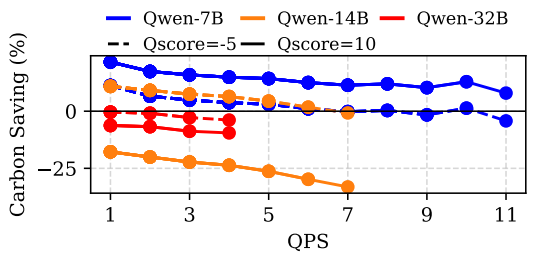

Figure 7: Carbon savings of AWQ Qwen compared to the FP16 version with Qscore low (-5) and high (10). Data are missing at higher QPS for 14B and 32B, as larger models cannot serve intensive workloads.

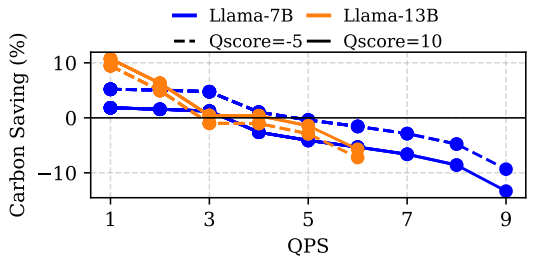

Figure 8: Carbon savings of AWQ Llama compared to the FP16 version with Qscore low (-5) and high (10). Data are missing at higher QPS for 13B, as larger models cannot serve intensive workloads.

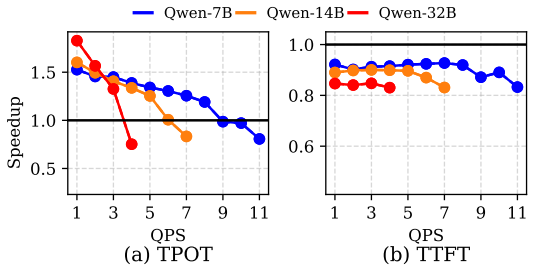

Figure 9: Latency speedup of AWQ Qwen compared to the FP16 version.

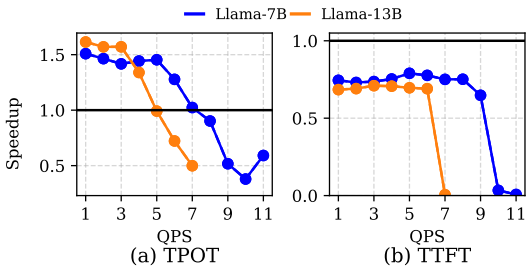

Figure 10: Latency speedup of AWQ Llama compared to the FP16 version.

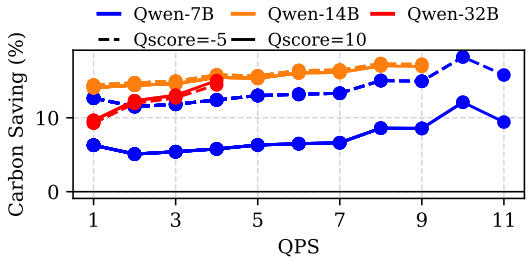

Figure 11: Carbon savings of W8A8 Qwen compared to the FP16 version with Qscore low (-5) and high (10). Data are missing at higher QPS for 14B and 32B, as larger models cannot serve intensive workloads.

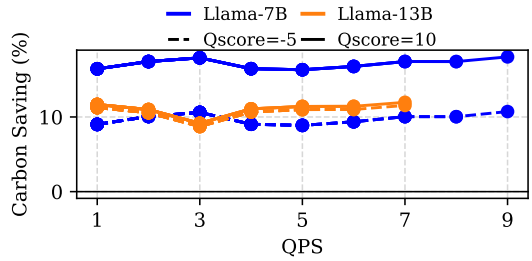

Figure 12: Carbon savings of W8A8 Llama compared to the FP16 version with Qscore low (-5) and high (10).

back to 16-bit during inference, adding overhead. AWQ improves TPOT in memory-bound cases at low QPS by reducing memory transfer, but this advantage diminishes as QPS increases and computation grows. Since TTFT is compute-intensive, AWQ provides no speedup.

Takeaway 2: Weight-only quantization reduces carbon emissions at low QPS but loses its advantage as QPS increases.

Question 2: *Is activation quantization always greener?*

Unlike weight-only quantization, activation quantization applies to both weights and activations. We compared the relative carbon savings of W8A8 compared to the FP16 version under different Qscores and QPS, and the results show that the answer is **yes**. As shown in Figure 11, W8A8 consistently reduces carbon emissions for Qwen models, regardless of quality requirements. Despite some accuracy loss in the 7B model (Table 3 in Appendix D), it still achieves a 5% carbon reduction at Qscore = 10. Unlike AWQ, W8A8 maintains stable savings even as QPS increases.

We observe a similar trend for Llama in Figure 12. Notably, Llama 7B improved in output quality after quantization (Table 3 in Appendix D), saving over 15% of carbon at Qscore = 10. This shows activation quantization can break the trade-off between FP16 and AWQ, ensuring consistent carbon savings across different FUs.

To understand why W8A8 always outperforms FP16 in carbon savings, we analyze its impact on TTFT and TPOT speedup. Figures 13 and 14 show

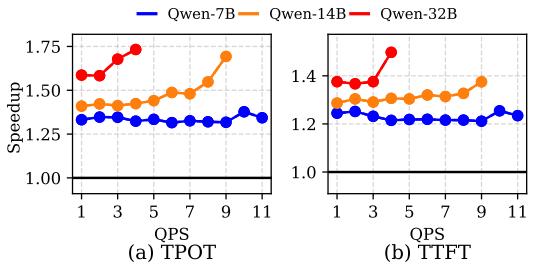

Figure 13: Latency speedup of W8A8 Qwen compared to the FP16 version.

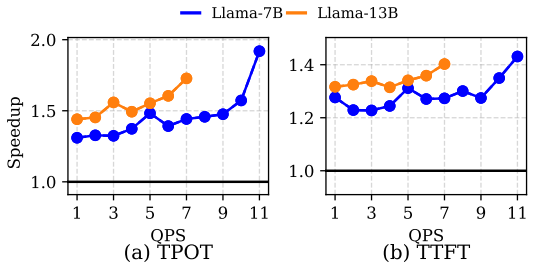

Figure 14: Latency speedup of W8A8 Llama compared to the FP16 version.

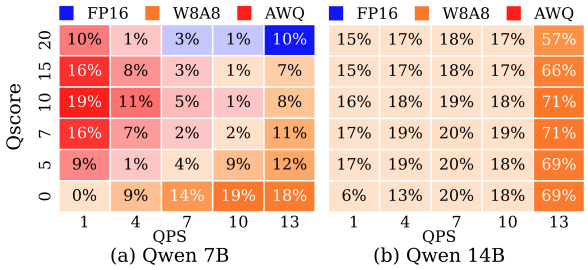

Figure 15: Comparison of FP16, AWQ and W8A8 versions of Qwen 7B/14B in FUEL. Tile colors indicate the model with the lowest carbon per FU. Tile values are carbon savings (%) of greenest quantization version-compared to the second greenest.

that W8A8 consistently speeds up TPOT and TTFT across all QPS ranges. This improvement comes from reducing both weight and activation precision, which decreases the amount of data movement and computation during inference. This makes W8A8 a more sustainable choice for LLM serving, as it strikes a balance between quality and performance.

Question 3: *Does a universal greenest quantization method exist?*
The answer is **no**. Figure 15 shows the relative carbon savings of FP16, AWQ, and W8A8 models across various QPS and Qscores for Qwen 7B and 14B. For Qwen 14B, W8A8 outperforms in all scenarios, with carbon savings increasing as QPS rises. However, for Qwen 7B, AWQ maintains slightly better quality at low QPS, while W8A8 lags behind at high QPS and high-quality requirements due to its slight accuracy loss (Table 3 in Appendix D).

Table 1: Hardware platform specifications in this paper.

Specification	L40 server	H100 server
GPU	4 × L40	8× H100
TDP	300W	350W
Process size	5nm	5nm
Die size	609 mm^2	814 mm^2
GPU memory	40GB	80GB
Release Year	2022	2023
CPU	AMD EPYC 7443	Intel Xeon 8480+
TDP	200W	350W
Process size	7nm	10nm
Die size	4×81 mm^2	4×477 mm^2
CPU memory	504GB	1031GB
Release Year	2021	2023

Takeaway 3: Weight and activation quantization methods, like W8A8, hold significant potential for reducing carbon emissions in LLM serving, particularly for larger models.

6 Case Study: Hardware

In this section, we examine how hardware platform affects the environmental impact of LLM serving. Using FUEL, we investigate whether more advanced hardware can enhance carbon efficiency while maintaining output quality.

6.1 Evaluation Methodology

Setup. We conduct experiments on two GPU servers with different hardware configurations, one older and one newer, as detailed in Table 1. For fair comparisons, we use a single GPU per server for all experiments. We evaluate the Qwen2.5 (7B, 14B) and Llama2 (7B, 13B).

Benchmarking configurations. Same as in §4.

6.2 Evaluation Results

Question 1: *How does different hardware contribute to total carbon emissions?*
Figure 16 shows the breakdown of carbon emissions per FU for Qwen and Llama 7B models on different hardware platforms, separating operational and embodied carbon. It is worth noting that different hardware contributes to different embodied carbon per FU, due to differences in the total embodied carbon for each hardware. The L40 platform has lower total embodied carbon than the H100, with values of 26.6 and 29.92 $kgCO_2eq$ repectively. These differences are based on calculations using the ACT modeling tool (Gupta et al.,

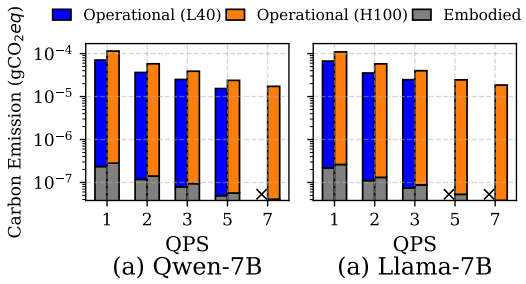

Figure 16: Breakdown of Carbon emission per FU for Qwen and Llama 7B models on different hardware platforms, evaluated in FUEL with Qscore=0.

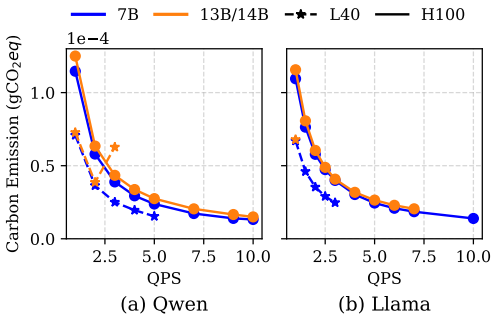

Figure 17: Carbon emission per FU of Qwen and Llama model families on different hardware platforms, evaluated in FUEL with Qscore=0.

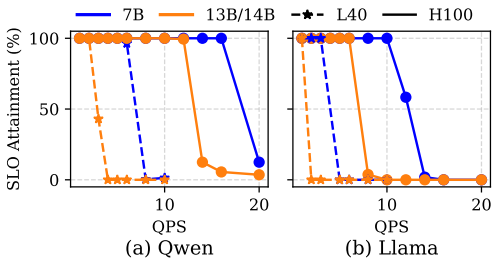

Figure 18: SLO attainment of Qwen and Llama model families on different hardware platforms.

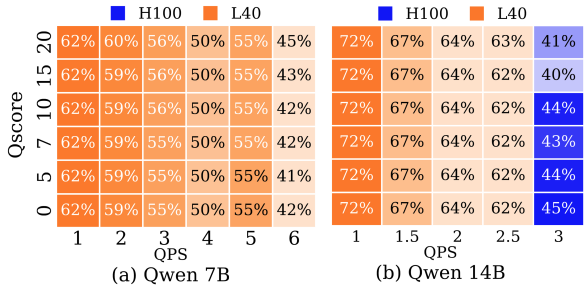

Figure 19: Comparison of Qwen 7B and 14B on different hardware platforms in FUEL. Tile colors indicate the hardware with the lowest carbon emission per FU. Tile values are carbon savings (%) of the greenest hardware compared to the second greenest.

2022) and are due to hardware factors such as process and die size. The difference is even more pronounced between the AMD EPYC 7443 and Intel Xeon 8480+ CPUs, with the AMD CPU having 9.98 $kgCO_2eq$, compared to the Intel's 42.81 $kgCO_2eq$, over 4x higher.

Advanced hardware like the H100 offers better performance but higher embodied carbon. Extending hardware lifetime can yield more carbon savings, especially considering the large difference in embodied carbon between older and newer devices.

Question 2: *Is LLM serving on advanced hardware greener?*

Figure 17 shows the carbon emissions per FU for the Qwen and Llama model families on two hardware platforms. At low QPS, the L40 server consistently has lower carbon emissions than the H100. This means that the answer is **no**: advanced hardware is not necessarily greener.

The main advantage of advanced hardware like the H100 is its ability to produce higher-quality outputs and meet performance constraints, as shown in Figure 18. Although advanced hardware may not be greener, it provides better performance and supports higher-quality LLM serving at scale.

Question 3: *How to choose greener hardware?*

To answer this question, we run experiments across different FUs with varying QPS and Qscores. Figure 19 shows the relative carbon savings of L40 and H100 servers for Qwen 7B and 14B. Hardware carbon efficiency depends mainly on model size and QPS, with a minor influence from Qscore. Newer hardware is more carbon efficient at high QPS, while older hardware is better at low QPS. These findings underscore the sustainability benefits of reusing older hardware to cut carbon emissions while maintaining performance and quality.

Takeaway 4: Advanced hardware offers higher performance but is not always greener due to higher embodied carbon. Older hardware can achieve lower carbon emissions while still meeting quality and performance constraints.

7 Conclusion

We introduce FUEL, the first evaluation framework for unveiling LLM serving's environmental impact by leveraging functional units as the basis for comparison. We explore how model size, quantization, and hardware affect carbon emissions. Our findings highlight opportunities for greener LLM deployment, paving the way for sustainable AI systems.

Limitations

We discuss the limitations of this work as follows.

Model families. Our case studies examine two widely used open-source LLM families, Qwen2.5 and Llama2, which we believe are representative of general LLM serving behaviors. However, we have not yet explored other model families, such as Mistral, or task-specific models like multimodal, vision-language, and code-focused LLMs. We leave these investigations for future work.

Hardware. All our experiments are conducted on a single GPU to ensure fair comparisons, limiting us to models up to 32B. We have yet to explore the performance and power dynamics in a multi-GPU distributed environment, which would allow us to run larger models like Llama 70B. This setup introduces additional overhead, particularly from communication, making the results even more insightful. We leave this exploration for future work.

Quality metrics. Quantitatively evaluating LLM output quality remains a challenging and open research question. We experimented with various metrics before selecting the reward model, a common approach in reinforcement learning from human feedback. While we believe our key findings remain robust regardless of the specific quality metric used, access to more advanced evaluation methods in the future could further enhance the accuracy and rigor of our work.

Ethical Statement

This research aims to contribute to the development of sustainable and carbon efficient LLM serving systems. We are committed to conducting our work in a responsible manner, adhering to ethical guidelines and best practices. Our focus is on minimizing the environmental impact of LLM deployments while ensuring that the quality of the models and the performance of the systems meet the necessary standards for practical use.

We recognize the potential environmental consequences of the widespread use of LLMs, including energy consumption, electronic waste, and the environmental impact of hardware manufacturing. Therefore, we emphasize the importance of optimizing LLMs for lower energy and carbon emissions, not only in terms of performance but also through hardware reuse and longevity, as part of a more sustainable approach to AI infrastructure.

We strive to be transparent in our research methodologies and encourage further exploration of green AI practices. As we explore new avenues for improving LLM efficiency, we remain mindful of the broader social, economic, and environmental implications of deploying large-scale AI systems and aim to promote solutions that benefit both the technology and society at large.

We also recognize the importance of fairness and inclusivity, ensuring that our research does not disproportionately harm any community or group and aligns with the goal of creating AI systems that are accessible and beneficial to all.

References

Amey Agrawal, Nitin Kedia, Ashish Panwar, Jayashree Mohan, Nipun Kwatra, Bhargav Gulavani, Alexey Tumanov, and Ramachandran Ramjee. 2024. Taming throughput-latency tradeoff in llm inference with sarathi-serve. In *18th USENIX Symposium on Operating Systems Design and Implementation (OSDI 24)*.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. 2021. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.

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 preprint arXiv:1803.05457*.

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*.

Tri Dao, Dan Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. 2022. Flashattention: Fast and memory-efficient exact attention with io-awareness. *Advances in Neural Information Processing Systems (NeurIPS)*.

Yi Ding and Tianyao Shi. 2024. Sustainable llm serving: Environmental implications, challenges, and opportunities. In *2024 IEEE 15th International Green and Sustainable Computing Conference (IGSC)*.

Ahmad Faiz, Sotaro Kaneda, Ruhan Wang, Rita Chukwunyere Osi, Prateek Sharma, Fan Chen, and Lei Jiang. 2024. LLMCarbon: Modeling the end-to-end carbon footprint of large language models. In *The Twelfth International Conference on Learning Representations*.

Elias Frantar and Dan Alistarh. 2023. Sparsegpt: Massive language models can be accurately pruned in one-shot. In *International Conference on Machine Learning*, pages 10323–10337. PMLR.

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. 2022. Gptq: Accurate post-training quantization for generative pre-trained transformers. *arXiv preprint arXiv:2210.17323*.

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*.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. 2024. A framework for few-shot language model evaluation.

Sarah Griffiths. 2020. 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.

Udit Gupta, Mariam Elgamal, Gage Hills, Gu-Yeon Wei, Hsien-Hsin S. Lee, David Brooks, and Carole-Jean Wu. 2022. ACT: Designing sustainable computer systems with an architectural carbon modeling tool. In *ISCA*.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2020. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*.

Dongfu Jiang, Yishan Li, Ge Zhang, Wenhao Huang, Bill Yuchen Lin, and Wenhu Chen. 2023. Tigerscore: Towards building explainable metric for all text generation tasks. *Transactions on Machine Learning Research*.

Walter Klöpffer and Birgit Grahl. 2014. *Life cycle assessment (LCA): a guide to best practice*. John Wiley & Sons.

Sven Köhler, Benedict Herzog, Henriette Hofmeier, Manuel Vögele, Lukas Wenzel, Andreas Polze, and Timo Hönig. 2023. Carbon-aware memory placement. In *Proceedings of the 2nd Workshop on Sustainable Computer Systems*, HotCarbon '23, New York, NY, USA. Association for Computing Machinery.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *SOSP*.

Yaniv Leviathan, Matan Kalman, and Yossi Matias. 2023. Fast inference from transformers via speculative decoding. In *International Conference on Machine Learning*.

Amy Li, Sihang Liu, and Yi Ding. 2024a. Uncertainty-aware decarbonization for datacenters. In *Proceedings of the 3rd Workshop on Sustainable Computer Systems (HotCarbon)*.

Baolin Li, Rohan Basu Roy, Daniel Wang, Siddharth Samsi, Vijay Gadepally, and Devesh Tiwari. 2023. 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*, pages 1–15.

Baolin Li, Yankai Jiang, Vijay Gadepally, and Devesh Tiwari. 2024b. Sprout: Green generative ai with carbon-efficient llm inference. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*.

Tianle Li, Wei-Lin Chiang, Evan Frick, Lisa Dunlap, Tianhao Wu, Banghua Zhu, Joseph E Gonzalez, and Ion Stoica. 2024c. From crowdsourced data to high-quality benchmarks: Arena-hard and benchbuilder pipeline. *arXiv preprint arXiv:2406.11939*.

Yueying Lisa Li, Omer Graif, and Udit Gupta. 2024d. Towards carbon-efficient llm life cycle. In *Proceedings of the 3rd Workshop on Sustainable Computer Systems (HotCarbon)*.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. 2024. Awq: Activation-aware weight quantization for on-device llm compression and acceleration. *Proceedings of Machine Learning and Systems*, 6:87–100.

Stephanie Lin, Jacob Hilton, and Owain Evans. 2021. Truthfulqa: Measuring how models mimic human falsehoods. *arXiv preprint arXiv:2109.07958*.

Chris Yuhao Liu, Liang Zeng, Jiacai Liu, Rui Yan, Jujie He, Chaojie Wang, Shuicheng Yan, Yang Liu, and Yahui Zhou. 2024a. Skywork-reward: Bag of tricks for reward modeling in llms. *arXiv preprint arXiv:2410.18451*.

Jiachen Liu, Zhiyu Wu, Jae-Won Chung, Fan Lai, Myungjin Lee, and Mosharaf Chowdhury. 2024b. Andes: Defining and enhancing quality-of-experience in llm-based text streaming services. *arXiv preprint arXiv:2404.16283*.

Jiawei Liu, Chunqiu Steven Xia, Yuyao Wang, and Lingming Zhang. 2024c. Is your code generated by ChatGPT really correct? rigorous evaluation of large language models for code generation. *Advances in Neural Information Processing Systems (NeurIPS)*.

Diptyaroop Maji, Prashant Shenoy, and Ramesh K Sitaraman. 2022. 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)*.

Sophia Nguyen, Beihao Zhou, Yi Ding, and Sihang Liu. 2024. Towards sustainable large language model serving. In *Proceedings of the 3rd Workshop on Sustainable Computer Systems (HotCarbon)*.

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

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.

Pratyush Patel, Esha Choukse, Chaojie Zhang, Íñigo Goiri, Brijesh Warrier, Nithish Mahalingam, and Ricardo Bianchini. 2024a. Characterizing power management opportunities for llms in the cloud. In *Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 3*.

Pratyush Patel, Esha Choukse, Chaojie Zhang, Aashaka Shah, Íñigo Goiri, Saeed Maleki, and Ricardo Bianchini. 2024b. Splitwise: Efficient generative LLM inference using phase splitting. In *ISCA*.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. 2025. Qwen2.5 technical report. *Preprint*, arXiv:2412.15115.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2021. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106.

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

Tianyao Shi, Yanran Wu, Sihang Liu, and Yi Ding. 2024. Greenllm: Disaggregating large language model serving on heterogeneous gpus for lower carbon emissions. *Preprint*, arXiv:2412.20322.

Jovan Stojkovic, Chaojie Zhang, Íñigo Goiri, Josep Torrellas, and Esha Choukse. 2024. Dynamollm: Designing llm inference clusters for performance and energy efficiency. *Preprint*, arXiv:2408.00741.

TheBloke. 2023a. Llama-2-13b-chat-awq. https://huggingface.co/TheBloke/Llama-2-13B-chat-AWQ. Accessed: 2025-02-15.

TheBloke. 2023b. Llama-2-7b-chat-awq. https://huggingface.co/TheBloke/Llama-2-7B-Chat-AWQ. Accessed: 2025-02-15.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models. *Preprint*, arXiv:2307.09288.

Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. 2016. Newsqa: A machine comprehension dataset. *arXiv preprint arXiv:1611.09830*.

vLLM Project. 2023. Llm compressor. https://github.com/vllm-project/llm-compressor. Accessed: 2025-02-15.

Tu Vu, Mohit Iyyer, Xuezhi Wang, Noah Constant, Jerry Wei, Jason Wei, Chris Tar, Yun-Hsuan Sung, Denny Zhou, Quoc Le, and Thang Luong. 2024. Freshllms: Refreshing large language models with search engine augmentation. *Findings of the Association for Computational Linguistics ACL 2024*.

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

Gyeong-In Yu, Joo Seong Jeong, Geon-Woo Kim, Soojeong Kim, and Byung-Gon Chun. 2022. Orca: A distributed serving system for transformer-based generative models. In *16th USENIX Symposium on Operating Systems Design and Implementation (OSDI)*.

Weizhe Yuan, Graham Neubig, and Pengfei Liu. 2021. Bartscore: Evaluating generated text as text generation. *Advances in Neural Information Processing Systems*, 34:27263–27277.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? *arXiv preprint arXiv:1905.07830*.

Hengrui Zhang, August Ning, Rohan Baskar Prabhakar, and David Wentzlaff. 2024. Llmcompass: Enabling efficient hardware design for large language model inference. In *2024 ACM/IEEE 51st Annual International Symposium on Computer Architecture (ISCA)*.

Ming Zhong, Yang Liu, Da Yin, Yuning Mao, Yizhu Jiao, Pengfei Liu, Chenguang Zhu, Heng Ji, and Jiawei Han. 2022. Towards a unified multi-dimensional evaluator for text generation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 2023–2038, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

A Code Availability

Please find our code repository at `https://anonymous.4open.science/r/FUEL-76EF`. We provide a README file that offers instructions on how to get started and utilize FUEL.

We summarize the appendix as follows:

- Section B provides a detailed description of embodied carbon modeling.
- Section C presents additional results for the model size case study, including experiments on NewsQA and two additional datasets (Arena Hard and HumanEval).
- Section D provides supplementary results for the quantization case study on the same datasets.
- Section E offers more detailed comparisons of model selections in the hardware case study.

B Embodied Carbon Modeling

We utilize the ACT (Gupta et al., 2022) embodied carbon modeling tool. The embodied carbon footprint can be divided into manufacturing and packaging carbon emissions. Manufacturing carbon arises from producing electronic components like transistors and resistors from raw materials, while packaging carbon is associated with assembling these components into chips and circuit boards:

$$C_{\text{em}} = C_{\text{manufacturing}} + C_{\text{packaging}} \tag{4}$$

The manufacturing embodied footprint C_m of processors and SoCs like CPUs and GPUs depends on several factors: die area (A_{die}), carbon intensity of the energy consumed by the fab ($\texttt{CI}_{\text{fab}}$), energy consumed per unit area manufactured (EPA), the GHG emissions from gases and chemicals per unit area (GPA), the footprint of procuring raw materials per unit area (MPA), and fabrication yield (Yield, set to 0.875 as in Gupta et al. (2022)). The information is sourced from product data sheets and sustainability reports. The manufacturing embodied carbon of a processor can be calculated as:

$$C_{\text{m}} = \frac{(\texttt{CI}_{\text{fab}} \times \text{EPA} + \text{GPA} + \text{MPA}) \times A_{die}}{\text{Yield}} \tag{5}$$

The packaging carbon emission C_{p} is calculated by the number of integrated circuits (N_{IC}) with a packaging footprint. Following ACT, we use an average packaging overhead of 150 gCO_2 per IC.

$$C_{\text{p}} = N_{\text{IC}} \times 150 \tag{6}$$

In cloud environments or HPC clusters, it is often challenging to obtain details of DRAM specifications. Previous studies (Li et al., 2023; Köhler et al., 2023) generally assume that the embodied carbon of DRAM is proportional to its capacity. Following prior work, we adopt a fixed rate of $65gCO_2$/GB to estimate the embodied carbon of DRAM.

C Additional Results for Model Size Case Study

C.1 Results on NewsQA Summarization

Figure 20 illustrates the naive carbon emission per token for various model sizes across a range of QPS. This figure represents carbon per token without the use of FUEL. Without considering server constraints, smaller models consistently exhibit lower carbon emissions per token, which does not reflect real-world serving requirements where larger models may be preferred for higher quality outputs.

Figure 22 shows the cumulative percentage of quality scores $\geq$ a given threshold for different models on the NewsQA summarization task. This figure highlights significant differences between models, particularly between Llama 7B and 13B, and between Qwen 7B and 32B. This discrepancy demonstrates why smaller models may not be as advantageous when higher quality is required, as larger models provide better outputs.

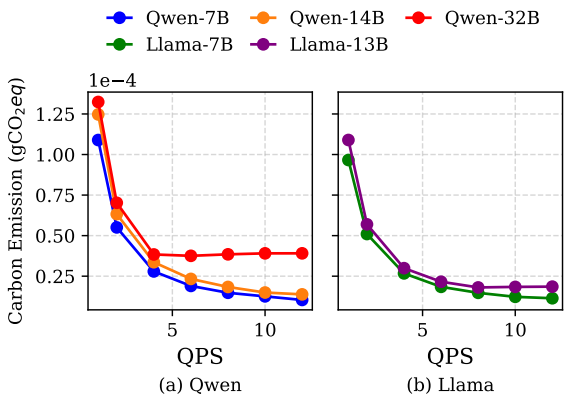

Figure 20: Naive carbon emission per token for different model sizes across QPS range on NewsQA dataset.

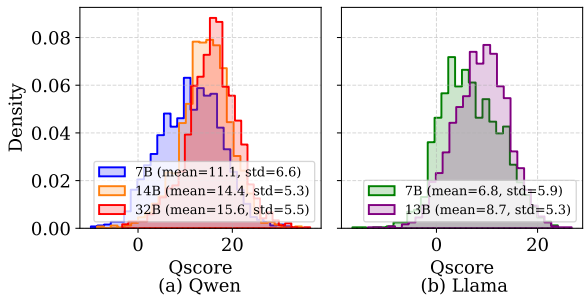

Figure 21: Qscore distribution of outputs across different model sizes on the NewsQA dataset.

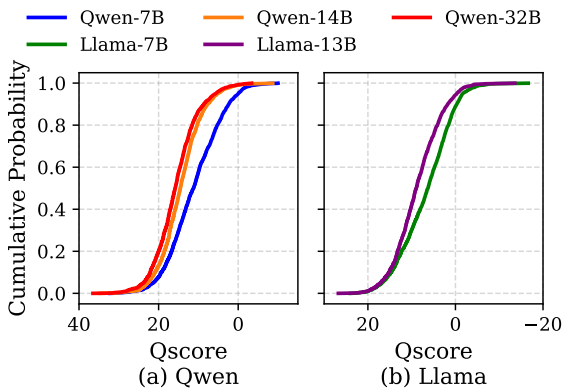

Figure 22: Cumulative percentage of Qscore $\geq$ threshold on NewsQA dataset.

C.2 Results on Arena Hard

The Arena Hard dataset (Li et al., 2024c) is a challenging benchmark designed to evaluate the instruction following capabilities of LLMs, which is derived from real user interactions on Chatbot Arena.

Figure 23 shows the Qscore distribution for different model sizes on the Arena Hard dataset. As shown, larger models tend to achieve higher quality scores. However, compared to the quality distribution on the NewsQA summarization task, while

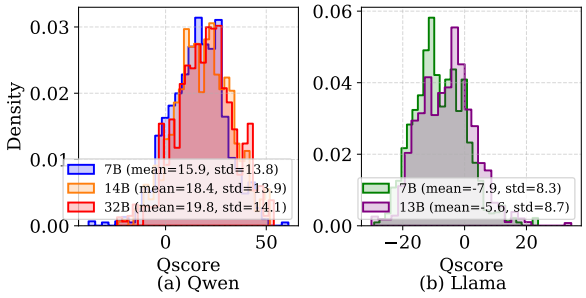

Figure 23: Qscore distribution of outputs across different model sizes on the Arena Hard dataset.

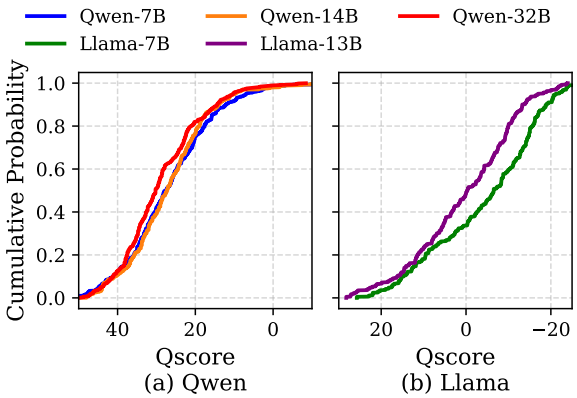

Figure 24: Cumulative percentage of Qscore $\geq$ threshold for different model sizes on Arena Hard dataset.

larger models still perform better, the differences between model sizes on Arena Hard are less pronounced than on the NewsQA. However, the gap between Llama 7B and 13B remains significant. Figure 24 also confirms this trend by showing the cumulative percentage of quality scores $\geq$ a given threshold for various model sizes on Arena Hard.

Figure 25 shows carbon emissions per FU across model sizes under different Qscore settings at QPS = 1 req/s. For the Qwen model family, since the Qscore distribution gap has narrowed, we only observe the 32B model producing less carbon than the 7B model when the quality requirement becomes very high (Qscore > 15). On the other hand, due to the significant quality distribution gap between the Llama models, a slight increase in the quality requirement makes the Llama 13B model greener than the 7B model. Moreover, when the quality requirements become stricter, the carbon emission gap between Llama 13B and 7B becomes larger.

The result aligns well with our findings on the NewsQA dataset: if the quality requirement is high, larger models become a greener choice, especially when there are large differences in quality distri-

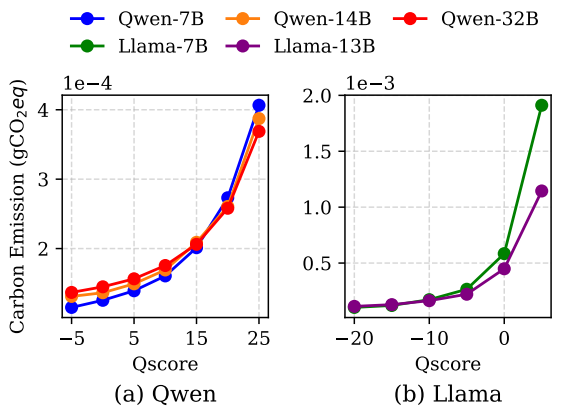

Figure 25: Carbon emission per FU for different model sizes on Arena Hard across Qscores at QPS=1 req/s.

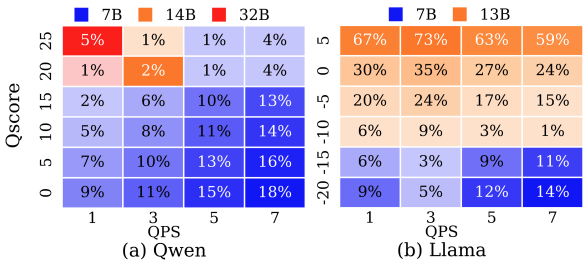

Figure 26: Comparison of different model sizes on Arena Hard in FUEL. Tile colors indicate the model with the lowest carbon per FU. Tile values are carbon savings (%) of the greenest size compared to the second greenest.

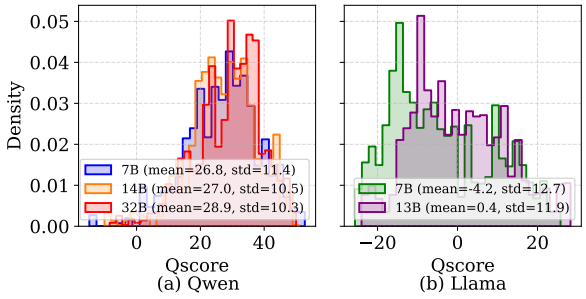

Figure 27: Qscore distribution for different model sizes on HumanEval dataset.

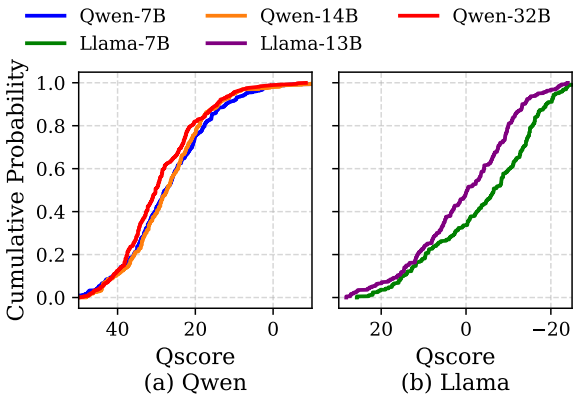

Figure 28: Cumulative percentage of Qscore $\geq$ threshold for different model sizes on HumanEval dataset.

bution across models of different sizes. Figure 26 shows the optimal model size choice across various Qscore and QPS conditions for the Qwen and Llama families. Due to the close quality distribution within the Qwen family, the advantage of larger models is constrained to the top-left corner (high Qscore, low QPS). In contrast, in the bottom right corner, as the quality requirement decreases and QPS increases, the 7B model becomes the greenest one.

C.3 Results on HumanEval

The HumanEval dataset (Chen et al., 2021) is a benchmark designed to evaluate the code generation ability of LLMs. It consists of Python coding problems and requires LLMs to implement the specific functions.

Figure 27 shows the Qscore distribution for different model sizes on the HumanEval dataset. Qscore distribution for the Qwen models is much closer on this dataset. This is consistent with their technical report (Qwen et al., 2025), which also highlights similar performance across models in the HumanEval evaluation. However, for the Llama models, the gap between the 7B and 13B model remains large, as expected.

Figure 29 shows carbon emissions per FU across model sizes on HumanEval dataset under different Qscore settings at QPS = 1 req/s. For the Qwen family, since the quality difference between the three model sizes on this task is not significant, increasing the Qscore does not lead to larger models demonstrating carbon emission saving over the 7B model. The carbon emissions per FU remain similar across model sizes even with higher Qscore requirements. In contrast, for the Llama model family, due to the large quality gap between the 7B and 13B models, we observe that even at very low quality requirements (e.g., Qscore = -15), the 13B model exhibits lower carbon emissions than the 7B model.

If we extend the Qscore requirement and QPS into two dimensions, as demonstrated in Figure 30, we observe that on HumanEval, Qwen 14B only shows an incremental carbon saving of 1-2% at QPS = 1 req/s, while in most other cases, Qwen 7B remains the greenest model. This is because the

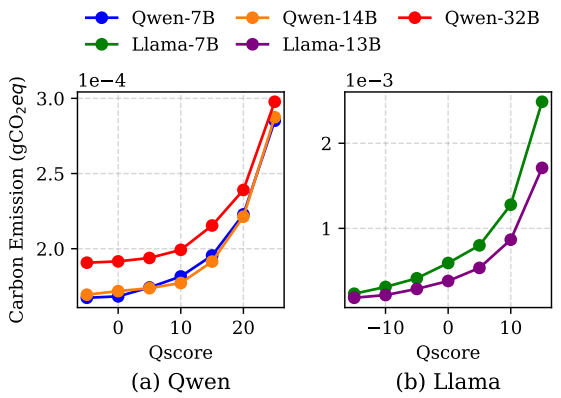

Figure 29: Carbon emission per FU for different model sizes on HumanEval across Qscores at QPS=1 req/s.

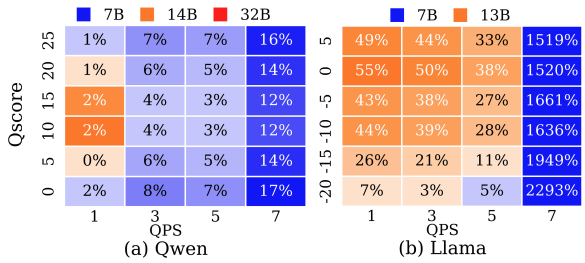

Figure 30: Comparison of different model sizes on HumanEval in FUEL. Tile colors indicate the model with the lowest carbon per FU. Tile values are carbon savings (%) of the greenest size compared to the second greenest.

output quality of Qwen 7B is very close to that of Qwen 14B and 32B. For the Llama model family, the results align with the previous observations: at lower QPS and higher quality requirements, the 13B model becomes the greenest option, as it can produce higher-quality responses compared to the 7B model.

This experiment on the HumanEval dataset further highlights our previous conclusion that selecting the greenest model size requires a comprehensive consideration of both model output quality and workload intensity.

D Additional Results for Quantization Case Study

As shown in Table 2, we used LM Eval (Gao et al., 2024), an open-source LLM evaluation tool, to assess the LLMs used in our experiments and their quantized versions. The evaluations were conducted on tasks from the Open LLM Leaderboard, including ARC-c (Clark et al., 2018), GSM8k (Cobbe et al., 2021), HellaSwag (Zellers et al., 2019), MMLU (Hendrycks et al., 2020), TruthfulQA (Lin et al., 2021), and Winogrande (Sakaguchi et al., 2021).

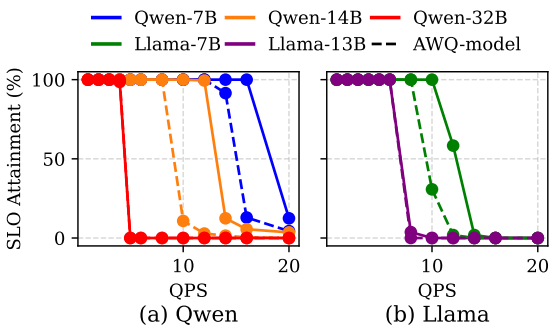

Figure 31: SLO attainment of Qwen and Llama families with AWQ version across QPS range.

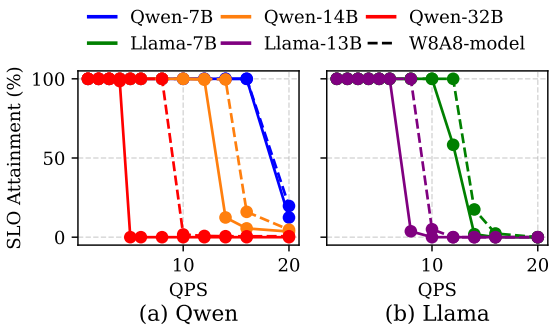

Figure 32: SLO attainment of Qwen and Llama families with W8A8 version across QPS range.

We also present the Qscore for the LLMs and their quantized versions across three datasets, as shown in Table 3.

D.1 Results on NewsQA Summarization

Figure 31 and Figure 32 illustrate the impact of the AWQ and W8A8 quantized versions on SLO attainment across the QPS range for the Qwen and Llama families. As shown in Figure 31, the AWQ version of the models fails to meet the SLO at lower QPS values. In contrast, the W8A8 quantized version improves efficiency, enabling models to serve a higher QPS.

Figure 33 shows the comparison results among FP16, AWQ and W8A8 versions in Llama family. W8A8 has almost the lowest carbon emission under all conditions.

D.2 Results on Arena Hard

Figure 34 shows the results of different quantization methods for Qwen 7B/14B models on Arena Hard dataset. This aligns well with the results on the previous NewsQA summarization dataset,

Table 2: Evaluation on different benchmarks for Qwen and Llama families with their quantized versions.

Model	Method	ARC-c	GSM8k	HellaSwag	MMLU	TruthfulQA	Winogrande
Qwen-7B	FP16	63.57	81.96	62.24	74.23	49.82	73.64
	AWQ	62.03 (-1.54)	79.61 (-2.35)	61.52 (-0.72)	73.33 (-0.9)	50.43 (+0.61)	74.11 (+0.47)
	W8A8	63.65 (+0.08)	82.11 (+0.15)	62.15 (-0.09)	74.18 (-0.05)	49.45 (-0.37)	74.35 (-0.71)
Qwen-14B	FP16	69.54	79.23	65.73	79.87	52.26	80.66
	AWQ	68.00 (-1.54)	80.89(+1.66)	64.78 (-0.95)	78.88 (-0.99)	48.84 (-3.42)	79.48 (-1.18)
	W8A8	69.71 (+0.71)	79.83 (-0.6)	65.74 (+0.01)	79.93 (+0.06)	51.04 (-1.22)	81.14 (+0.48)
Qwen-32B	FP16	71.42	75.89	67.11	83.28	51.16	80.03
	AWQ	69.88 (-1.54)	76.72(+0.83)	66.47 (-0.64)	82.40 (-0.88)	52.14 (-0.98)	79.72 (-0.31)
	W8A8	71.08 (-0.34)	75.82 (-0.07)	67.14 (+0.03)	83.15 (-0.13)	50.55 (-0.61)	80.43 (+0.4)
Llama-7B	FP16	49.83	23.2	59.34	47.22	45.04	72.93
	AWQ	48.98 (-0.85)	21.23(-1.97)	58.61 (-0.73)	45.34 (-1.88)	43.57 (-1.47)	72.53 (-0.4)
	W8A8	50.34 (+0.51)	22.67 (-0.53)	59.3 (-0.04)	47.24 (+0.02)	44.80 (-0.24)	73.32 (+0.39)
Llama-13B	FP16	55.63	35.56	63.1	53.55	40.88	75.06
	AWQ	54.95 (-0.68)	31.69(-3.87)	62.13 (-0.97)	53.77 (+0.22)	41.37 (+0.49)	76.09 (+1.03)
	W8A8	55.29 (-0.34)	35.18 (-0.38)	63.08 (-0.02)	53.65 (+0.1)	41.49 (+0.61)	75.22 (+0.16)

Table 3: Mean Qscore on three datasets for Qwen and Llama families with their quantized versions.

Model	Method	Qscore NewsQA	Qscore ArenaHard	Qscore HumanEval
Qwen-7B	FP16	11.11	15.90	26.82
	AWQ	11.77 (+0.66)	14.17 (-1.73)	26.52 (-0.3)
	W8A8	10.46 (-0.65)	15.76 (-0.14)	26.84 (+0.02)
Qwen-14B	FP16	14.37	18.41	27.03
	AWQ	12.01 (-2.36)	15.33 (-3.08)	26.41 (-0.62)
	W8A8	14.40 (+0.03)	18.49 (+0.08)	27.24 (+0.21)
Qwen-32B	FP16	15.62	19.82	28.86
	AWQ	14.87 (-0.75)	18.89 (-0.93)	27.99 (-0.87)
	W8A8	15.36 (-0.26)	19.73 (-0.09)	28.38 (-0.48)
Llama-7B	FP16	6.81	-7.93	-4.19
	AWQ	6.49 (-0.32)	-10.19 (-2.26)	-8.23 (-4.04)
	W8A8	6.87 (+0.06)	-8.58 (-0.65)	-4.23 (-0.04)
Llama-13B	FP16	8.73	-5.63	0.41
	AWQ	8.63 (-0.11)	-6.34 (-0.71)	-1.65 (-2.06)
	W8A8	8.64 (-0.09)	-5.69 (-0.06)	0.24 (-0.17)

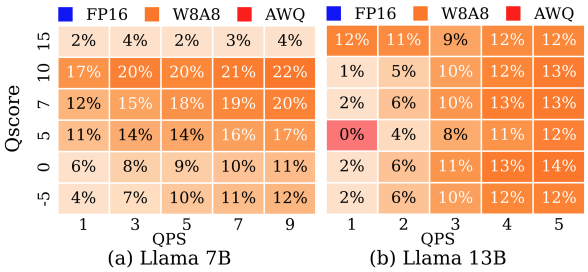

Figure 33: Comparison of FP16, AWQ and W8A8 versions of Llama 7B/13B in FUEL on NewsQA dataset. Tile colors indicate the model with the lowest carbon per FU. Tile values are carbon savings (%) of greenest version compared to the second greenest.

as AWQ shows an advantage at low QPS on the smaller 7B model. When we use 14B model, W8A8 illustrates the great potential to save up to 50% carbon emission under each scenario. We can see a similar trend in Figure 35 on Llama models.

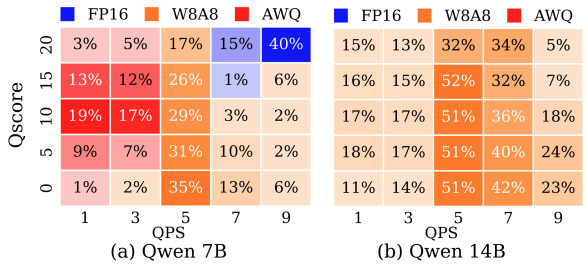

Figure 34: Comparison of FP16, AWQ and W8A8 versions of Qwen 7B/14B in FUEL on Arena Hard dataset. Tile colors indicate the model with the lowest carbon per FU. Tile values are carbon savings (%) of the greenest version compared to the second greenest.

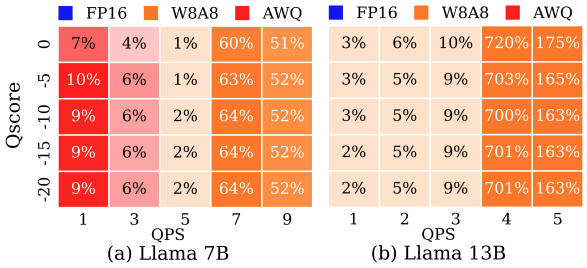

Figure 35: Comparison of FP16, AWQ and W8A8 versions of Llama 7B/13B in FUEL on Areana Hard dataset. Tile colors indicate the model with the lowest carbon per FU. Tile values are carbon savings (%) of greenest version compared to the second greenest.

D.3 Results on HumanEval

As shown in Figure 36, AWQ still becomes the greenest method when QPS is low, but W8A8 dominates in more conditions on the Qwen 7B model. This is because, after W8A8 quantization, the Qscore of Qwen 7B improves on the HumanEval dataset. For the Qwen 14B model, W8A8 is no

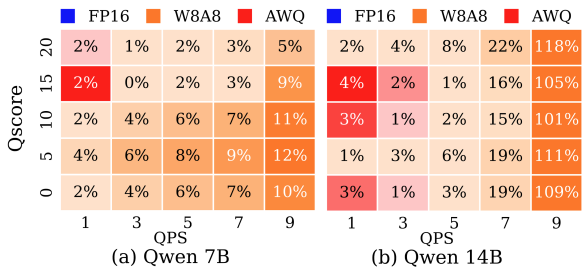

Figure 36: Comparison of FP16, AWQ and W8A8 versions of Qwen 7B/14B in FUEL on HumanEval dataset. Tile colors indicate the model with the lowest carbon per FU. Tile values are carbon savings (%) of greenest quantization version compared to the second greenest.

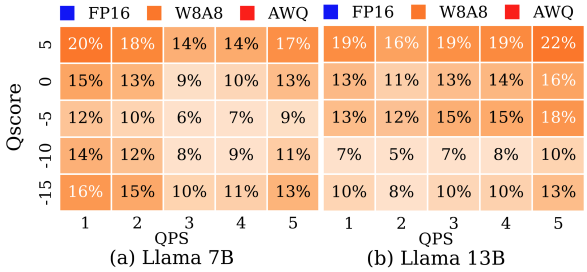

Figure 37: Comparison of FP16, AWQ and W8A8 versions of Llama 7B/13B in FUEL on HumanEval dataset. Tile colors indicate the model with the lowest carbon per FU. Tile values are carbon savings (%) of greenest quantization version compared to the second greenest.

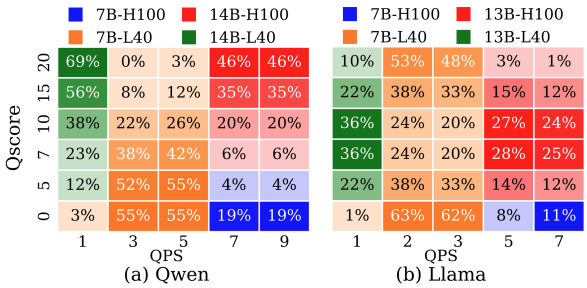

Figure 38: Comparison of model-hardware combinations for Qwen and Llama in FUEL. Tile colors indicate the model-hardware with the lowest carbon per FU. Tile values are carbon savings (%) of the greenest choice compared to the second greenest.

longer the greenest method under all conditions. This is due to AWQ experiencing minimal accuracy degradation on this dataset, allowing it to retain its advantage at low QPS.

E Additional Results for Hardware Case Study

Figure 38 compares model and hardware combinations, further confirming our previous conclusion: older hardware can achieve lower carbon emissions. As shown in the figure, whether for the Qwen or Llama model families, the greenest choice at low QPS is consistently the L40 server. Once the hardware is fixed, we can apply insights from the model size case study to select the model size based on the quality requirement. This pattern reaffirms that choosing the optimal model and hardware combination requires a balance between performance needs and carbon efficiency.