Offline Energy-Optimal LLM Serving: Workload-Based Energy Models for LLM Inference on Heterogeneous Systems

Grant Wilkins
gfw27@cam.ac.uk
University of Cambridge
Cambridge, UK

Srinivasan Keshav
sk818@cam.ac.uk
University of Cambridge
Cambridge, UK

Richard Mortier
rmm1002@cam.ac.uk
University of Cambridge
Cambridge, UK

ABSTRACT

The rapid adoption of large language models (LLMs) has led to significant advances in natural language processing and text generation. However, the energy consumed through LLM model inference remains a major challenge for sustainable AI deployment. To address this problem, we model the workload-dependent energy consumption and runtime of LLM inference tasks on heterogeneous GPU-CPU systems. By conducting an extensive characterization study of several state-of-the-art LLMs and analyzing their energy and runtime behavior across different magnitudes of input prompts and output text, we develop accurate ($R^2 > 0.96$) energy and runtime models for each LLM. We employ these models to explore an offline, energy-optimal LLM workload scheduling framework. Through a case study, we demonstrate the advantages of energy and accuracy aware scheduling compared to existing best practices.

CCS CONCEPTS

• **Computer systems organization** → **Heterogeneous (hybrid) systems**; • **Hardware** → *Impact on the environment*.

KEYWORDS

Sustainable computing, Heterogeneous computing, Large Language Models, Artificial Intelligence

1 INTRODUCTION

Rapid advancements in large language models (LLMs) have revolutionized natural language processing, enabling AI systems to achieve human-level performance on a wide range of language tasks [3, 26, 40]. However, the computational resources and energy consumption associated with deploying these models present significant challenges to not only energy systems but also sustainability goals [20, 21, 29]. As LLMs become increasingly integrated into

real-world applications, optimizing their energy efficiency during inference is crucial for sustainable AI development [45].

Inference, the process of using a trained model to make predictions on new data, is a critical phase in LLM deployment as it is the point at which AI capabilities become accessible to users. Unlike the one-time training process, inference is an ongoing, real-time process that directly impacts end-user experience. Inference in LLMs can be computationally expensive due to model size [45] and quality of service/latency expectations [42]. Scaling LLMs up across large data centers is challenging due to power [27] and communication overheads [28].

The energy intensity of LLM inference can be substantial even when compared to training [4]. Decarbonizing the energy sources for data centers can be challenging due to both sporadic demand and regional inefficiencies in adopting renewables. Higher energy consumption of an application approximately correlates with greater carbon intensity [32]. It is thus crucial to find energy-efficient methods to mitigate the environmental costs of LLM inference [1, 18].

To address this issue, we propose a workload-based model of energy consumption for LLM inference to let system operators navigate the trade-off between accuracy and energy usage.

Our contributions are as follows:

(1) We characterize the energy consumption and runtime behavior of several state-of-the-art LLMs on a heterogeneous GPU-CPU system (§4).

(2) We develop workload-based energy and runtime models that accurately capture the relationship between the number of input and output tokens and the energy and runtime characteristics of each LLM (§5).

(3) We demonstrate the effectiveness of our approach through a case study, showcasing a tunable trade-off between energy and accuracy (§6).

Our profiling framework and datasets are openly available.[1]

2 RELATED WORK

2.1 Energy Consumption in AI Systems

Recent reports have found that the computation required by state-of-the-art AI systems entail massive energy consumption and carbon emissions [4, 22, 29, 35, 45]. The energy intensity of AI systems can be broadly split between the energy required for training and that required for inference after models are deployed [10]. Training complex models on massive datasets is an energy-intensive process, with estimates finding that training GPT-3 required 1,287 megawatt-hours of energy [29]. Even with this huge amount of energy, a year of inference by an LLM on cloud infrastructure can consume over

---

[1]https://github.com/grantwilkins/energy-inference.git

25× more energy than training that same model [4]. Some of these issues and emissions of course depend on the deployment scale and hardware efficiency [35], however, the trend remains that energy consumption in inference is a large issue. Optimizing software and hardware specifically for AI workloads is thus essential [1].

Desislavov et al. [5] provide an examination of trends in AI inference energy consumption, arguing that while performance has increased dramatically, energy consumption has not escalated at the same pace, thanks to hardware optimizations and algorithmic innovations. Chien et al. [4] discuss larger trends in LLM inference energy consumption and do not focus on device-level energy modeling benefits. Samsi et al. [35] explore the energy consumption of Meta's Llama LLMs for different batch sizes and numbers of GPUs, showing the potential energy reductions obtainable by tuning these parameters. Stojcovik et al. [37] discuss the impacts of GPU frequency scaling on the energy efficiency of serving LLMs; however, at this point, this work is only a characterization and not an applied analysis.

Our work extends these studies with a thorough CPU+GPU energy measurements across multiple model families and sizes, producing one of the most comprehensive datasets of its kind.

2.2 Energy-Aware Data Center Scheduling

A large body of work that focuses on energy-aware scheduling [6, 13, 19, 24, 34, 38], but none of these focus on the unique challenge of developing workload-aware models for LLM inference towards this goal. Hu et al. [12] analyze deep learning workloads in GPU data centers, offering insights into energy conservation strategies through workload scheduling. This research aligns with our objectives by confirming the critical role of scheduling in reducing energy footprints.

Li et al. [17] introduce Clover, which promises to minimize carbon emissions for serving AI inference. Unlike our study, this work does not explicitly consider LLMs or a per-model function to capture energy and runtime, instead focusing on carbon-emission patterns for a data center.

Gu et al. [8] presents PowerFlow, a tool that uses clock-frequency data from GPUs to minimize energy consumption as a scheduling decision. However, their study does not consider LLMs and is not necessarily workload-aware.

Patel et al. introduce POLCA [27], which can provide a way to automatically power-cap based on existing workload traces. Li et al. [18] focuses on delivering a geographic load balancing perspective for AI inference, optimizing environmental equity. However, their model considers large-scale workload traces, not device-level energy and runtime data.

Our work aims to fill the niche with energy-aware LLM inference scheduling using measurements from state-of-the-art open-source LLMs leading to an applied analysis using offline optimization. The results of our findings can be used by system operators to accurately predict and schedule based on the amount of energy and runtime for inference.

3 METHODS

For our LLM inference engine we use Hugging Face's Accelerate [9]. This library uses all available GPUs, and divides a model among the available GPUs in a tensor parallel fashion to minimize intermediate communication and maximize the distributed capabilities for computation across the devices. We disable KV-caching [41] to ensure that our measurements are consistent between runs and do not require a warm-start phase.

3.1 LLM Selection

We study several open-source LLMs, summarized in Table 1. By profiling different LLMs we are able to explore the effects of diverse architectures and parameter values on runtime, energy consumption, and accuracy. For each model, we conduct a series of standardized text generation prompts to evaluate their energy consumption during inference.

Numerous benchmarks have sought to quantify LLM accuracy, e.g., the MMLU [11] and HellaSwag [46]. To avoid the inadequacies introduced by individual tests for accuracy [23], we use the Hugging Face Leaderboard's [2] *average accuracy*, denoted A_K, that averages the performance of a model, K, on a large repository of datasets and tests.

Table 1: LLM Energy Consumption and Runtime

LLM (# Params)	vRAM Size (GB)	# A100s	A_K (%) [2]
Falcon (7B)	14.48	1	44.17
Falcon (40B)	83.66	3	58.07
Llama-2 (7B)	13.48	1	50.97
Llama-2 (13B)	26.03	1	55.69
Llama-2 (70B)	137.98	4	64.52
Mistral (7B)	15.00	1	60.97
Mixtral (8x7B)	93.37	3	68.47

3.2 Energy Profiling of Our Cluster

We perform all experiments the Swing cluster at Argonne National Lab using a single node with 8×Nvidia A100 (40GB) GPUs, 2×AMD Epyc 7742 64-core processors, and 1TB of DDR4 RAM. We use only the minimum number of GPUs as shown in Table 1. We profile the system's energy consumption during inference using tools that capture Nvidia GPU energy and AMD CPU power while timing the operation. Our methods utilize the known relationship that $E = Pt$ where E represents energy, P is average power, and t is runtime.

3.2.1 NVIDIA GPUs. We use PyJoules [31], a Python-based energy measurement library, to quantify the energy consumption associated with inference on NVIDIA GPUs. PyJoules provides an interface to NVML [25], providing a software-defined energy usage assessment for targeted NVIDIA devices. This tool offers GPUs real-time energy consumption for a given tracked process.

3.2.2 AMD CPUs. We adopt a different strategy for AMD CPUs due to the absence of a Python API. Instead, we utilize AMDμProf's `timechart` feature, which provides detailed power draw metrics for every core on the chip at fine-grained intervals. By polling AMDμProf at 100ms intervals, we can capture the power draw of each physical CPU core throughout the model inference process.

To ensure we accurately attribute the energy consumption to our inference task, we monitor the CPU core residency through `psutil`.

This information allows us to identify and record the specific cores actively engaged in the inference process at each time step. The total energy consumption for the inference task is then calculated by summing the power usage across all active cores and summing over the product of the power usage and time of inference, as follows:

$$E_{Total,CPU} = \sum_{core} \left(\sum_i P_{core,i} \Delta t_i \right)$$

where $P_{core,i}$ represents the power draw of an individual core at each time step, i, with Δt_i being the time step size.

4 PROBLEM FORMULATION

The purpose of developing workload-based models of LLM inference is to create a framework that allows a data center operator to navigate the trade-off between model accuracy and energy consumption. To do so, we formalize an optimization problem below.

Consider a data center that hosts $\mathcal{K} = \{1, \ldots, K\}$ distinct LLM models. Assume that a fraction γ_K of the inference workload is assigned to each model K, where $\gamma_K \in [0, 1], \forall K$ and $\sum_{K \in \mathcal{K}} \gamma_K = 1$.

We denote a query q by its count of input and output tokens, $q = (\tau_{in}, \tau_{out})$. A workload with m queries is then a multiset $Q \in (\mathbb{N}^2)^m$. As our goal is to perform scheduling of each query, we must create a disjoint partition of our set Q. We say that each $Q_K \in (\mathbb{N}^2)^{m_K}$ has m_K prompts and is composed of a set of lengths of input and output tokens $Q_K = \{(\tau_{in,1}, \tau_{out,1}), \ldots, (\tau_{in,i}, \tau_{out,i})\}$.

Since this is an offline setting we assume we have perfect knowledge of our system, including the number of output tokens that a given input prompt will produce. In reality, this is not known *ab initio* though work by Zheng et al. [47] has shown that the number of output tokens can be reasonably well estimated by analyzing past input-output pairs.

For optimization purposes, we must define a function based on the constant A_K from Table 1. We propose $a_K : \mathbb{N}^2 \to [0, \infty)$, a monotonically increasing function based on the number of input and output tokens that a model K ingests and produces. Therefore, for a model K processing tokens (τ_{in}, τ_{out}) we have

$$a_K(\tau_{in}, \tau_{out}) = A_K \tau_{in} + A_K \tau_{out}. \tag{1}$$

As we will later derive, we denote a model for energy consumption for a given number of input and output tokens as $e_K(\tau_{in,i}, \tau_{out,i}) : \mathbb{N}^2 \to [0, \infty)$.

Both of these functions have a normalized counterpart $\widehat{e_K}, \widehat{a_K} : \mathbb{N}^2 \to [0, 1]$ that scales the cost associated with these values $[0, 1]$ to make these different metrics comparable. We normalize by dividing by the largest known value of energy and accuracy prior to optimization.

Finally, let $\zeta \in [0, 1]$ denote a tuning parameter that lets a data center operator trade off energy for accuracy. Let $|Q|$ represent the total number of queries in our workload, and $|Q_K|$ represent the total number of queries each model K processes.

We now formulate our workload assignment problem as:

$$\min_{Q_K \in Q} \sum_{K \in \mathcal{K}} \sum_{(\tau_{in}, \tau_{out}) \in Q_K} \zeta \widehat{e_K}(\tau_{in}, \tau_{out}) - (1 - \zeta) \widehat{a_K}(\tau_{in}, \tau_{out}) \tag{2}$$

$$\text{s.t., } 0 < \frac{|Q_K|}{|Q|} < 1 \tag{3}$$

$$Q = \bigcup_{K \in \mathcal{K}} Q_K \tag{4}$$

$$Q_I \cap Q_J = \emptyset, I \neq J, \forall I, J \in \mathcal{K}, \tag{5}$$

where Equations 4 and 5 define the partition coverage of the workload, and Equation 3 ensures we give each LLM some queries. In our implementation, we dynamically normalize our energy and accuracy measures across all the queries to allow us to adjust the scale of costs across different models and query combinations. This problem is computationally intensive to solve as it is an example of a general assignment problem which are known to be NP-hard [7].

5 LLM INFERENCE PERFORMANCE

All hardware information we state in Section 3.2. We use Ubuntu 20.04 with Python 3.12.0, PyTorch v2.0.1, Torchvision v0.15.2, Numpy v1.26.0, Hugging Face v0.20.2, and Accelerate v0.26.1.

5.1 Experimental Strategy

We conduct an experimental campaign to evaluate the performance of differing workloads across various models. We systematically varied the number of input and output tokens to measure their effects on runtime and energy consumption under two main experimental conditions. In each experiment we do not allow for key-value caches [41] to be re-used to ensure our measurements are standard between iterations. We fix the batch size at 32.

5.1.1 Vary Input Tokens. For the first experimental condition, we executed inference requests with increasing the number of input tokens, ranging from 8 to 2048 tokens, while maintaining a fixed output token size of 32. This setup allowed us to isolate the impact of input size on the system's performance and energy efficiency.

5.1.2 Vary Output Tokens. In the second set of experiments, we varied the output token limit from 8 to 4096 tokens, keeping the number of input tokens constant at 32. This approach helped us understand how increasing output demands affect the runtime and energy consumption of the systems tested.

5.1.3 Randomization and Stopping Criteria. Each experiment was conducted in a randomized order to mitigate any potential bias introduced by the sequence of tests. Also, we repeated trials until either of two conditions was met: (*i*) the measured runtime was within 0.5 seconds of the actual mean runtime with 95% confidence; and (*ii*) a maximum of 25 trials were conducted for each setting if the first condition could not be met.

5.2 Input Token Effects

Figure 1 presents the impact of varying numbers of input tokens on the runtime, throughput, and energy per token for various LLMs. The results depict a clear trend: as the number of input tokens increases, the runtime tends to increase, while the throughput
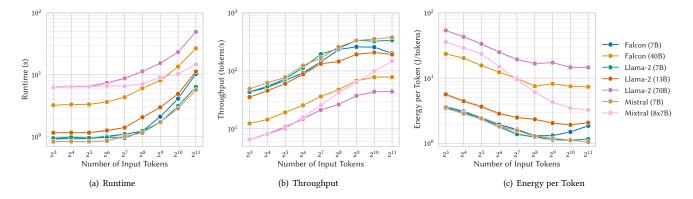
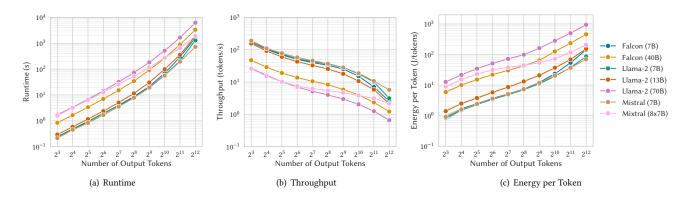Figure 1: Model performance against number of input tokens. Low variance renders error bars invisible.

Figure 2: Model performance against number of output tokens. Low variance renders error bars invisible.

plateaus, in accordance with a roofline model [44]. Specifically, the runtime increase is most pronounced for larger models like Llama-2 (70B) and Falcon (40B), likely due to the higher computational burden these models sustain as they process more extensive input sequences. The energy consumption per token demonstrates similar trends, with smaller models exhibiting lower energy per token compared to larger models.

An outlier to all of these cases is Mixtral (8x7B), which has a higher throughput and energy efficiency compared to other large models at larger token input sizes. This LLM's sparse mixture-of-experts architecture (SMoE) [14, 33] allows it to activate just 12B parameters on average by selecting two expert sub-models. This classification phase comes with an added runtime and energy overhead, however, on larger prompts it regains its performance capabilities. Therefore, for SMoE one gets the accuracy advantages of a large model for less energy and lower runtime than its denser counterparts.

5.3 Output Token Effects

Figure 2 illustrates how changes in the number of output tokens affect runtime, throughput, and energy consumption per token across different LLMs. Notably, the runtime exhibits a steep increase with larger output token sizes, which is consistent across all

models but is especially significant for the high-parameter models such as Falcon (40B) and Llama-2 (70B). Throughput, decreases as the number of output tokens increases. This inverse relationship highlights the additional time required to generate each additional token, which involves more extensive interaction between model layers and successive passes through the LLM to generate each token [41]. Energy per token also increases with the number of output tokens and number of parameters. This increase is particularly sharp in higher-parameter models like Falcon (40B).

Again, Mixtral (8x7B) demonstrates greater energy efficiency compared to its large parameter counterparts. Even in cases of high output token generation, an SMoE architecture can yield improvements in energy efficiency.

6 WORKLOAD-BASED MODEL FITTING

From the experimental results in Section 5 we can see that each LLM has a unique runtime and energy consumption characteristic that is a function of the given workload. In this section we develop and apply these models to optimizing energy and runtime of serving LLMs.

6.1 Independence of Input and Output Tokens

From observing our results, we explored whether the number of input and output tokens are independent in their effect on the energy consumption and runtime. The following table presents the ANOVA results for assessing the effects of the number of input tokens, the number of output tokens, and their interaction on the total energy consumption and runtime for LLM inference. To collect this data we perform a grid search from 8 to 2048, in increments of powers of two, for the space of input and output tokens to eliminate the bias of holding the input or output size constant. This analysis includes data aggregated across all models in Table 1.

Table 2: ANOVA Results for LLM Energy Consumption and Runtime

Metric	Variable	Sum of Squares	F-statistic	p-value
Energy (J)	Input Tokens	5.17×10^{10}	15.86	3.79×10^{-17}
	Output Tokens	4.13×10^{11}	126.63	1.22×10^{-65}
	Interaction	1.18×10^{11}	4.53	4.67×10^{-15}
Runtime (s)	Input Tokens	3.43×10^{5}	12.97	2.34×10^{-14}
	Output Tokens	2.78×10^{6}	104.98	4.56×10^{-60}
	Interaction	8.21×10^{5}	3.88	1.92×10^{-12}

The *number of input tokens* and *number of output tokens* both individually have a substantial impact on energy consumption and runtime, with output tokens having a larger effect size as indicated by the higher F statistic. Also, the *interaction* term shows that the input and output tokens depend on each other while impacting energy consumption and runtime. The high F-statistics and extremely low p-values for these effects confirm their significance. Therefore, we conclude that there is dependence between the number of input and output tokens for energy consumption and runtime.

6.2 Modeling Energy and Runtime

We use the results in Table 2 to guide the creation of models to predict the energy consumption and runtime of LLMs for use in optimization problems such as those discussed in Section 4.

For accurate models based on the number of input and output tokens there needs to be an interaction term that combines them. We therefore propose a model to describe the total energy consumption for a model K as a function of input and output tokens, τ_{in} and τ_{out}, respectively:

$$e_K(\tau_{in}, \tau_{out}) = \alpha_{K,0}\tau_{in} + \alpha_{K,1}\tau_{out} + \alpha_{K,2}\tau_{in}\tau_{out}, \quad (6)$$

where $\alpha_{K,0}, \alpha_{K,1}, \alpha_{K,2}$ are parameters determined through ordinary least squares (OLS) regression for each model and system combination.

Similarly, we propose the following model to describe the total runtime for a model K as a function of input and output tokens, τ_{in} and τ_{out}, respectively:

$$r_K(\tau_{in}, \tau_{out}) = \beta_{K,0}\tau_{in} + \beta_{K,1}\tau_{out} + \beta_{K,2}\tau_{in}\tau_{out}, \quad (7)$$

where $\beta_{K,0}, \beta_{K,1}, \beta_{K,2}$ are also unique to each model K.

Using the statsmodel (v0.14.2) Python package and its OLS API, we can determine the values of $\alpha_{K,i}$ and $\beta_{K,j}$ that best fit Equations 6 and 7 for each LLM, K. A summary of the quality of these fits are included in Table 3. As we can see, this model has high

explainability for the effect of input and output tokens on energy and runtime for inference of these different LLMs.

Table 3: Summary of OLS Regression Results Across Models

LLM (# Params)	Energy Model (e_K)			Runtime Model (r_K)		
	R^2	F-statistic	p-value	R^2	F-statistic	p-value
Falcon (7B)	0.964	681.2	2.53e-55	0.962	651.1	1.35e-54
Falcon (40B)	0.972	904.5	1.78e-60	0.976	1073.0	2.74e-63
Llama-2 (7B)	0.973	942.3	3.76e-61	0.972	1032.0	1.19e-62
Llama-2 (13B)	0.972	887.8	3.60e-60	0.972	907.0	1.60e-60
Llama-2 (70B)	0.976	1022.0	6.66e-62	0.980	1230.0	6.23e-65
Mistral (7B)	0.975	997.0	1.70e-61	0.976	1039.0	3.62e-62
Mixtral (8x7B)	0.980	1238.0	4.97e-65	0.992	3139.0	2.23e-80

6.3 Applying Our Models to Workload Routing

We can now use our runtime and energy consumption models to solve the workload-aware routing problem outlined in Section 4. Using PuLP (v.2.8.0), a Python package designed for solving optimization problems like that we formulate in Equation 2, we can encode a workload of input and output tokens with a set of binary variables that indicate which model will process that pair of tokens. Then, we convert the given constraints in Equations 3–5 using this format and effectively route our workload to different models.

As we show in Table 1 and Figures 1 and 2, an LLM with a larger parameter count has greater accuracy but also greater runtime and energy consumption for each input and output token. It is reasonable to host differently sized models to allow us to serve inference requests more runtime and energy efficiently with a trade-off of slightly lower accuracy.

For this example, we consider a data center serving the three Llama-2 models of 7B, 13B, and 70B parameters. Assume that our set $\mathcal{K} = \{1, 2, 3\}$ enumerates those models, respectively. A tunable parameter that affects our optimization problem is the data center partition γ_i. In our evaluation, we choose $\gamma_1 = 0.05$, $\gamma_2 = 0.2$, and $\gamma_3 = 0.75$.

With this, we can use the model for energy consumption of each LLM, K, in Equation 6 and our function to capture accuracy from Equation 1 to calculate the costs associated with each query and model as shown in Equation 2. For our sample workload, we use a subset of 500 queries from the Alpaca dataset [39], as it is a collection of 52002 queries with answers from GPT-4 [26].

Figure 3 shows the trade-offs in energy consumption, runtime, and accuracy by varying the operational parameter ζ while routing queries to different models. We represent as constants (straight lines) methods that do not use ζ, preferring to pick either a single LLM or to use a simple query-independent mechanism to route a query to an LLM. The remaining non-constant line represents the trade-off our offline scheduler makes as it adjusts to changes in ζ.

In Figure 3(a), we see that energy consumption is high when ζ is low because the system prioritizes accuracy over energy efficiency. Higher ζ values lead to more energy-efficient routing decisions, sacrificing accuracy for energy savings. Similarly, Figure 3(b) shows that the mean runtime per query decreases with increasing ζ. A low ζ value results in longer runtimes as the system routes queries to models that provide higher accuracy but are less efficient in time and energy. Conversely, higher ζ values result in shorter runtimes,
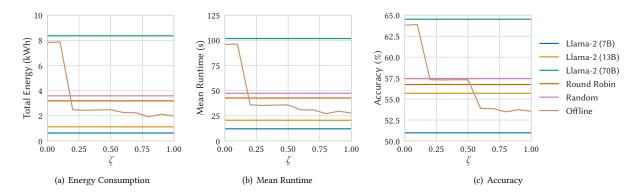
Figure 3: Behavior under offline simulation as ζ varies. Round-robin and Random query assignment are indistinguishable.

as the system favors more energy and time-efficient models over the most accurate ones. Figure 3(c) demonstrates the accuracy-cost trade-off, with small increases in accuracy requiring significant increases in runtime and energy consumption.

Our solution allows data center operators use ζ to navigate the trade-off space by, e.g., providing higher accuracy when energy prices are lower, or delivering lower latency and lower energy responses during times of peak load albeit with slightly reduced accuracy. This flexibility is important for adapting to different operational scenarios.

7 CONCLUSIONS

In this paper, we have examined the significant energy expenditure of LLM inference. We show that modeling and optimizing the energy consumption of LLM inference for a system is straightforward. We also showed that SMoE LLMs exhibit very promising energy efficiency characteristics. Through our models of energy and runtime we contribute to the ongoing efforts towards sustainable AI by providing a tunable optimization framework that allows for system operators to trade-off energy and accuracy. We confirm our hypothesis that there is potential for energy optimization using models of energy and accuracy.

Of course, as many others have done [4, 10, 16, 20, 21, 36] we have used energy consumption as a proxy for carbon footprint. As pointed out by Kannan and Kremer [15], improving carbon efficiency and energy efficiency are distinct goals, yet they are related and energy metrics can assist in understanding the magnitude of emissions for a given application [1]. Our measurements of energy consumption are also based on a single node in an HPC setting and so we cannot capture the runtime and energy overheads introduced by faults, networking, and communications that would pertain at data center scale. We also disabled key-value caching [30] to establish a performance baseline; future work should explore the impact of this and other optimizations. Finally, our workload-models are specific primarily to an NVIDIA A100 (40GB), as pointed out in other studies there are large variations for the same inference task across hardware [35, 43].

We hope that our energy models can be used in real-time systems to reduce energy consumption dynamically. By integrating these models into online scheduling algorithms, data centers can make energy-aware decisions based on the current workload and system state. This real-time optimization approach has the potential to significantly improve the energy efficiency of LLM inference in production environments. Similarly, including externalities like energy pricing and availability of sustainable energy into our model would bring systems closer to meeting sustainability goals.

REFERENCES

[1] Thomas Anderson, Adam Belay, Mosharaf Chowdhury, Asaf Cidon, and Irene Zhang. 2023. Treehouse: A Case For Carbon-Aware Datacenter Software. *SIGENERGY Energy Inform. Rev.* 3, 3 (oct 2023), 64–70. https://doi.org/10.1145/3630614.3630626

[2] Edward Beeching, Clémentine Fourrier, Nathan Habib, Sheon Han, Nathan Lambert, Nazneen Rajani, Omar Sanseviero, Lewis Tunstall, and Thomas Wolf. 2023. Open LLM Leaderboard. https://huggingface.co/spaces/HuggingFaceH4/open_llm_leaderboard.

[3] Rishi Bommasani, Drew A. Hudson, Ehsan Adeli, and et al. 2022. On the Opportunities and Risks of Foundation Models. arXiv:2108.07258 [cs.LG]

[4] Andrew A Chien, Liuzixuan Lin, Hai Nguyen, Varsha Rao, Tristan Sharma, and Rajini Wijayawardana. 2023. Reducing the Carbon Impact of Generative AI Inference (Today and in 2035). In *Proceedings of the 2nd Workshop on Sustainable Computer Systems* (Boston, MA, USA) *(HotCarbon '23).* Association for Computing Machinery, New York, NY, USA, Article 11, 7 pages. https://doi.org/10.1145/3604930.3605705

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

[6] Kaijie Fan, Marco D'Antonio, Lorenzo Carpentieri, Biagio Cosenza, Federico Ficarelli, and Daniele Cesarini. 2023. SYnergy: Fine-grained Energy-Efficient Heterogeneous Computing for Scalable Energy Saving. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis.* 1–13.

[7] Marshall L Fisher, Ramchandran Jaikumar, and Luk N Van Wassenhove. 1986. A multiplier adjustment method for the generalized assignment problem. *Management science* 32, 9 (1986), 1095–1103.

[8] Diandian Gu, Xintong Xie, Gang Huang, Xin Jin, and Xuanzhe Liu. 2023. Energy-Efficient GPU Clusters Scheduling for Deep Learning. arXiv:2304.06381 [cs.DC]

[9] Sylvain Gugger, Lysandre Debut, Thomas Wolf, et al. 2022. Accelerate: Training and inference at scale made simple, efficient and adaptable. https://github.com/huggingface/accelerate.

[10] Peter Henderson, Jieru Hu, Joshua Romoff, Emma Brunskill, Dan Jurafsky, and Joelle Pineau. 2020. Towards the Systematic Reporting of the Energy and Carbon Footprints of Machine Learning. *J. Mach. Learn. Res.* 21, 1, Article 248 (jan 2020), 43 pages.

[11] Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring Massive Multitask Language Understanding. In *International Conference on Learning Representations.* https://openreview.net/forum?id=d7KBjmI3GmQ

[12] Qinghao Hu, Peng Sun, Shengen Yan, Yonggang Wen, and Tianwei Zhang. 2021. Characterization and prediction of deep learning workloads in large-scale GPU datacenters. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis (SC '21).* Association for Computing Machinery, New York, NY, USA, Article 104, 15 pages. https://doi.org/10.1145/3458817.3476223

[13] Hongpeng Huo, Chongchong Sheng, Xinming Hu, and Baifeng Wu. 2012. An energy efficient task scheduling scheme for heterogeneous GPU-enhanced clusters. In *2012 International Conference on Systems and Informatics (ICSAI2012).* IEEE, 623–627.

[14] Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, Guillaume Lample, Lélio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, Szymon Antoniak, Teven Le Scao, Théophile Gervet, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2024. Mixtral of Experts. arXiv:2401.04088 [cs.LG]

[15] Sudarsun Kannan and Ulrich Kremer. 2023. Towards Application Centric Carbon Emission Management. In *Proceedings of the 2nd Workshop on Sustainable Computer Systems* (Boston, MA, USA) *(HotCarbon '23).* Association for Computing Machinery, New York, NY, USA, Article 5, 7 pages. https://doi.org/10.1145/3604930.3605725

[16] Baolin Li, Yankai Jiang, Vijay Gadepally, and Devesh Tiwari. 2024. Toward Sustainable GenAI using Generation Directives for Carbon-Friendly Large Language Model Inference. arXiv:2403.12900 [cs.DC] https://arxiv.org/abs/2403.12900

[17] Baolin Li, Siddharth Samsi, Vijay Gadepally, and Devesh Tiwari. 2023. Clover: Toward Sustainable AI with Carbon-Aware Machine Learning Inference Service. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis (SC '23).* Association for Computing Machinery, New York, NY, USA, Article 20, 15 pages. https://doi.org/10.1145/3581784.3607034

[18] Pengfei Li, Jianyi Yang, Adam Wierman, and Shaolei Ren. 2024. Towards Environmentally Equitable AI via Geographical Load Balancing. In *Proceedings of the 15th ACM International Conference on Future and Sustainable Energy Systems* (Singapore, Singapore) *(e-Energy '24).* Association for Computing Machinery, New York, NY, USA, 291–307. https://doi.org/10.1145/3632775.3661938

[19] Qianlin Liang, Walid A Hanafy, Ahmed Ali-Eldin, and Prashant Shenoy. 2023. Model-driven cluster resource management for ai workloads in edge clouds. *ACM Transactions on Autonomous and Adaptive Systems* 18, 1 (2023), 1–26.

[20] Liuzixuan Lin and Andrew A Chien. 2023. Adapting Datacenter Capacity for Greener Datacenters and Grid. In *Proceedings of the 14th ACM International Conference on Future Energy Systems* (Orlando, FL, USA) *(e-Energy '23).* Association for Computing Machinery, New York, NY, USA, 200–213. https://doi.org/10.1145/3575813.3595197

[21] Liuzixuan Lin, Rajini Wijayawardana, Varsha Rao, Hai Nguyen, Emmanuel Wedan GNIBGA, and Andrew A. Chien. 2024. Exploding AI Power Use: an Opportunity to Rethink Grid Planning and Management. In *Proceedings of the 15th ACM International Conference on Future and Sustainable Energy Systems* (Singapore, Singapore) *(e-Energy '24).* Association for Computing Machinery, New York, NY, USA, 434–441. https://doi.org/10.1145/3632775.3661959

[22] Alexandra Sasha Luccioni, Sylvain Viguier, and Anne-Laure Ligozat. 2023. Estimating the Carbon Footprint of BLOOM, a 176B Parameter Language Model. *Journal of Machine Learning Research* 24, 253 (2023), 1–15. http://jmlr.org/papers/v24/23-0069.html

[23] Timothy R. McIntosh, Teo Susnjak, Tong Liu, Paul Watters, and Malka N. Halgamuge. 2024. Inadequacies of Large Language Model Benchmarks in the Era of Generative Artificial Intelligence. arXiv:2402.09880 [cs.AI]

[24] Xinxin Mei, Xiaowen Chu, Hai Liu, Yiu-Wing Leung, and Zongpeng Li. 2017. Energy efficient real-time task scheduling on CPU-GPU hybrid clusters. In *IEEE INFOCOM 2017-IEEE Conference on Computer Communications.* IEEE, 1–9.

[25] NVIDIA. Accessed 2024. NVIDIA-NVML. https://docs.nvidia.com/deploy/nvml-api/index.html. Available online.

[26] OpenAI, :, Josh Achiam, Steven Adler, Sandhini Agarwal, et al. 2023. GPT-4 Technical Report. arXiv:2303.08774 [cs.CL]

[27] Pratyush Patel, Esha Choukse, Chaojie Zhang, Íñigo Goiri, Brijesh Warrier, Nithish Mahalingam, and Ricardo Bianchini. 2024. 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 (ASPLOS '24).* Association for Computing Machinery, New York, NY, USA, 207–222. https://doi.org/10.1145/3620666.3651329

[28] Pratyush Patel, Esha Choukse, Chaojie Zhang, Aashaka Shah, Íñigo Goiri, Saeed Maleki, and Ricardo Bianchini. 2024. Splitwise: Efficient generative LLM inference using phase splitting. In *ISCA.* https://www.microsoft.com/en-us/research/publication/splitwise-efficient-generative-llm-inference-using-phase-splitting/

[29] David Patterson, Joseph Gonzalez, Quoc Le, Chen Liang, Lluis-Miquel Munguia, Daniel Rothchild, David So, Maud Texier, and Jeff Dean. 2021. Carbon Emissions and Large Neural Network Training. arXiv:2104.10350 [cs.LG]

[30] Reiner Pope, Sholto Douglas, Aakanksha Chowdhery, Jacob Devlin, James Bradbury, Jonathan Heek, Kefan Xiao, Shivani Agrawal, and Jeff Dean. 2023. Efficiently scaling transformer inference. *Proceedings of Machine Learning and Systems* 5 (2023), 606–624.

[31] PowerAPI. 2024. PyJoules: Python-based energy measurement library for various domains including NVIDIA GPUs. https://github.com/powerapi-ng/pyJoules. Accessed: 2024-01-10.

[32] Ana Radovanović, Ross Koningstein, Ian Schneider, Bokan Chen, Alexandre Duarte, Binz Roy, Diyue Xiao, Maya Haridasan, Patrick Hung, Nick Care, et al. 2022. Carbon-aware computing for datacenters. *IEEE Transactions on Power Systems* 38, 2 (2022), 1270–1280.

[33] Samyam Rajbhandari, Conglong Li, Zhewei Yao, Minjia Zhang, Reza Yazdani Aminabadi, Ammar Ahmad Awan, Jeff Rasley, and Yuxiong He. 2022. DeepSpeed-MoE: Advancing Mixture-of-Experts Inference and Training to Power Next-Generation AI Scale. In *Proceedings of the 39th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 162),* Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (Eds.). PMLR, 18332–18346. https://proceedings.mlr.press/v162/rajbhandari22a.html

[34] Lavanya Ramapantulu, Bogdan Marius Tudor, Dumitrel Loghin, Trang Vu, and Yong Meng Teo. 2014. Modeling the energy efficiency of heterogeneous clusters. In *2014 43rd International Conference on Parallel Processing.* IEEE, 321–330.

[35] Siddharth Samsi, Dan Zhao, Joseph McDonald, Baolin Li, Adam Michaleas, Michael Jones, William Bergeron, Jeremy Kepner, Devesh Tiwari, and Vijay Gadepally. 2023. From Words to Watts: Benchmarking the Energy Costs of Large Language Model Inference. In *2023 IEEE High Performance Extreme Computing Conference (HPEC).* 1–9. https://doi.org/10.1109/HPEC58863.2023.10363447

[36] Satveer and Mahendra Singh Aswal. 2016. A comparative study of resource allocation strategies for a green cloud. In *2016 2nd International Conference on Next Generation Computing Technologies (NGCT).* 621–625. https://doi.org/10.1109/NGCT.2016.7877487

[37] Jovan Stojkovic, Esha Choukse, Chaojie Zhang, Inigo Goiri, and Josep Torrellas. 2024. Towards Greener LLMs: Bringing Energy-Efficiency to the Forefront of LLM Inference. arXiv:2403.20306 [cs.AI]

[38] Xiaoyong Tang and Zhuojun Fu. 2020. CPU–GPU utilization aware energy-efficient scheduling algorithm on heterogeneous computing systems. *IEEE Access* 8 (2020), 58948–58958.

[39] R. Taori, I. Gulrajani, T. Zhang, and et al. 2024. Stanford alpaca: An instruction following llama model. https://github.com/tatsu-lab/stanford_alpaca. Accessed: 2024-01-15.

[40] Google Gemini Team. 2024. Gemini: A Family of Highly Capable Multimodal Models. arXiv:2312.11805 [cs.CL]

[41] Ashish Vaswani, Noam Shazeer, Niki Parmar, et al. 2017. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems* (Long Beach, California, USA) *(NIPS'17).* Curran Associates Inc., Red Hook, NY, USA, 6000–6010.

[42] Yuxin Wang, Yuhan Chen, Zeyu Li, Zhenheng Tang, Rui Guo, Xin Wang, Qiang Wang, Amelie Chi Zhou, and Xiaowen Chu. 2024. Towards Efficient and Reliable LLM Serving: A Real-World Workload Study. arXiv:2401.17644 [cs.DC]

[43] Grant Wilkins, Srinivasan Keshav, and Richard Mortier. 2024. Hybrid Heterogeneous Clusters Can Lower the Energy Consumption of LLM Inference Workloads. In *Proceedings of the 15th ACM International Conference on Future and Sustainable Energy Systems (e-Energy '24).* Association for Computing Machinery, New York, NY, USA, 506–513. https://doi.org/10.1145/3632775.3662830

[44] Samuel Williams, Andrew Waterman, and David Patterson. 2009. Roofline: an insightful visual performance model for multicore architectures. *Commun. ACM* 52, 4 (apr 2009), 65–76. https://doi.org/10.1145/1498765.1498785

[45] Carole-Jean Wu, Ramya Raghavendra, Udit Gupta, and et al. 2022. Sustainable ai: Environmental implications, challenges and opportunities. *Proceedings of Machine Learning and Systems* 4 (2022), 795–813.

[46] Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. HellaSwag: Can a Machine Really Finish Your Sentence? arXiv:1905.07830 [cs.CL]

[47] Zangwei Zheng, Xiaozhe Ren, Fuzhao Xue, Yang Luo, Xin Jiang, and Yang You. 2023. Response Length Perception and Sequence Scheduling: An LLM-Empowered LLM Inference Pipeline. In *Thirty-seventh Conference on Neural Information Processing Systems.* https://openreview.net/forum?id=eW233GDOpm