Towards Sustainable NLP: Insights from Benchmarking Inference Energy in Large Language Models

Soham Poddar[1]*, Paramita Koley[2]*, Janardan Misra[3], Niloy Ganguly[1], Saptarshi Ghosh[1]
[1] Indian Institute of Technology, Kharagpur, India.
[2] Indian Statistical Institute, Kolkata, India.
[3] Accenture Labs, Bangalore, India.

Abstract

Large language models (LLMs) are increasingly recognized for their exceptional generative capabilities and versatility across various tasks. However, the high inference costs associated with these models have not received adequate attention, particularly when compared to the focus on training costs in existing research. In response to this gap, our study conducts a comprehensive benchmarking of LLM inference energy across a wide range of NLP tasks, where we analyze the impact of different models, tasks, prompts, and system-related factors on inference energy. Specifically, our experiments reveal several interesting insights, including strong correlation of inference energy with output token length and response time. Also, we find that quantization and optimal batch sizes, along with targeted prompt phrases, can significantly reduce energy usage. This study is the first to thoroughly benchmark LLM inference across such a diverse range of aspects, providing insights and offering several recommendations for improving energy efficiency in model deployment.

1 Introduction

Recent discussions on the energy and carbon impact of machine learning (ML) algorithms have mainly concentrated on quantifying energy usage during the training phase of these models (Dodge et al., 2022; Luccioni et al., 2023; Patterson et al., 2021; Raffel et al., 2020). Studies on inference energy are much rarer because a single inference operation consumes considerably less energy and resources. However, under deployment, inference is performed many more times, making its energy impact significant and warranting separate investigation (Wu et al., 2022; Patterson et al., 2022). For example, 90% of total cloud computing demand for AWS, the largest global cloud provider, were for model inference purpose (Barr, 2019). Moreover, a key motivation for training large language models is that a single model can achieve state-of-the-art performance across diverse NLP tasks due to its impressive zero-shot and few-shot capabilities, making it energy-efficient from a training perspective. However, when we consider the total carbon footprint of the entire lifetime of the model, the energy requirement for model inference plays a significant role, considering the number of inferences that are carried out during the model's lifetime. Thus it is crucial to conduct a systematic study to quantify the energy requirements and carbon emissions for model inference across various models and tasks.

1.1 Literature survey

Existing works have attempted to study the inference energy of LLMs from various perspectives. Everman et al. (2023) evaluated energy usage of GPT models on question-answering tasks, employing Software Carbon Intensity (SCI) released by green software. Samsi et al. (2023) provides a detailed account of energy usage during inference for question-answering tasks for Llama models on various GPU architectures. Liu et al. (2022) study the energy consumption trade-off for fine-tuning vs few-shot learning for both encoder-decoder and decoder-only models on SuperGLUE and RAFT tasks, measuring energy in FLOPS. These preliminary works mostly ignore the performance-energy tradeoff.

In recent times, Li et al. (2024) compare the effect of prompt directives on inference energy and performance for Llama-2 models for 3 question-answering tasks. Luccioni et al. (2024) offers a comparatively detailed account of inference energy usage while choosing LLMs and tasks from a diverse range, showing the dependency of energy with model size and architecture, task type, and output token length. While these works provide an initial account of inference energy usage of LLMs in different configurations, they are often limited by the number and diversity of the models and tasks (Check Appendix A for details).

	Fine-grained I/O	Top-level I/O	Task Complexity	Response time	Model size & family	Batch size	Quanti-zation	Targeted phrase
Our Work	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Luccioni et al. (2024)	No	Yes	No	No	Yes	No	No	No
Li et al. (2024)	No	No	No	No	No	No	No	Yes
Everman et al. (2023)	No	No	No	No	Yes	No	No	No
Desislavov et al. (2021)	No	No	No	No	Yes	No	No	No
Samsi et al. (2023)	No	No	No	No	Yes	No	No	No

Table 1: Comparing our approach with existing literature on energy for specific experimental settings

1.2 Our contributions & differences with prior works

In this work, we present a comprehensive study of LLMs, running models from both encoder-decoder and decoder-only models on both discriminative and generative NLP tasks, while analyzing the impact of different models, tasks, prompts, and system-related factors on inference energy. Specifically, (i) We start with a detailed analysis of various model-related factors that affect the inference energy of LLMs, where we systematically study the correlation between inference energy and influencing factors like input and output token length, response time, model size and complexity. (ii) We conduct various experiments to study the connection between inference energy and batch size, level of quantization, and prompt editing. (iii) We then complement our analysis by introducing the Normalized Accuracy metric, providing an accuracy-energy usage tradeoff analysis across tasks and models. (iv) Finally, we present a list of efficiency guidelines in Section 4.

In particular, our contributions include an energy analysis from eight aspects, as given in Table 1. Out of these aspects, several were primarily unexplored in the literature, as per our knowledge - these are indicated in the table. We vary the input and output token length at a finer granular level while reporting the variation in energy to have a better understanding of the correlation between energy and these factors, revealing several valuable insights (Section 3.2). A more thorough analysis of energy with variation of input or output while keeping the other fixed provides a more accurate insight into their comparative influence in inference energy, implying immediate solutions for energy-efficient inference approaches. The correlation between energy and response time validates response time as a good proxy of inference energy in black-box models (Section 3.1). Our work first explores task complexity's effect with inference energy (Section 3.3). We also provide the performance change for quantization (Section 3.5) and use of targeted phrases (Section 3.6) along with improvement in energy. While the effects of quantization and increasing batch size have been studied individually, we show that using them together under fixed memory constraints can help speed up the computations. Finally, we explore each of the above aspects for a diverse range of LLMs from both architecture families along with a diverse range of tasks, which was mostly lacking in previous attempts.

2 Experimental setup

In this section, we describe the models, datasets and various settings we use for our experiments.

2.1 Models

We select 6 popular and recent GPT-style models from the decoder-only family and 4 Flan-T5 models from the encoder-decoder family, adding to 10 models in total (details in Appendix B).

Decoder-only Models generate output in an autoregressive manner by predicting the next token in the sequence based on the context (key-value-query) vectors corresponding to the input and previously generated tokens. We consider the following models from decoder family in our study. (D1) **Tiny-LLama** (1.1B params); (D2) **Phi-3-mini** (3.8B params); (D3) **Mistral-7B** (7.2B params); (D4) **Llama-2-7B** (6.7B params); (D5) **Llama-3-8B** (8.0B params); (D6) **Llama-2-13B** (13B params);

Encoder-Decoder models process the input data and convert it into context (key-value) vectors. Then the decoder takes these vectors and generates output autoregressively. Models from this family, considered in our study, include:

(ED1) **Flan-T5-base** (248M params); (ED2) **Flan-T5-large** (783M params); (ED3) **Flan-T5-xl** (2.8B params); (ED4) **Flan-T5-xxl** (11B params);

2.2 Tasks and Datasets

In this work, we select a diverse range of NLP tasks, from generative to question-answering, classification, and single-sentence tasks. This includes both general GLUE / SuperGLUE benchmarks, as well as domain specific VAX-STANCE and CAVES (for studying effect of task complexity). We describe the tasks and their corresponding datasets in Table 2. For each dataset, we selected 1024 data samples randomly and performed all experiments on the same set for comparable results. Performance metrics are chosen depending on the tasks. For summarization tasks, average of ROUGE1, ROUGE2, and ROUGE-L are reported, whereas some form of F1 score are reported for the other tasks. Description/prompts of the datasets and the individual metrics have been given in Appendix C.

Task	Dataset
Linguistic acceptability check	COLA (GLUE)
Logical entailment	Mnli (GLUE)
Sentiment classification	SST2 (GLUE)
Contextual question answering	Boolq (SuperGLUE)
Causal reasoning	COPA (SuperGLUE)
Entity Question answering	ReCoRD (SuperGLUE)
Extractive question answering	SQuAD v2 (Rajpurkar et al., 2016)
Document summary generation	CNN-DM (Nallapati et al., 2016)
Dialogue summary generation	SAMSum (Gliwa et al., 2019)
3 class vaccine-stance classification	VAX-Stance (Poddar et al., 2022a)
12 class multi-label anti-vaccine concerns classification	CAVES (Poddar et al., 2022b)

Table 2: List of tasks/datasets we experimented on. Most tasks are taken from the GLUE (Wang et al., 2019b) and SuperGLUE(Wang et al., 2019a) benchmarks. Description/prompts are been given in Appendix C.

Normalized Accuracy (NA) Metric: Since different tasks use different metrics on different scales, it is difficult to compare the accuracy performance of models across the tasks. To gauge the overall performance of the models across multiple tasks, we introduce the *NA* metric that is obtained as follows. For each dataset, we first perform Z-score normalization across all the models, followed by a sigmoid operation to scale models between 0 and 1. We then average the scores for each model across all datasets and multiply by 100. Note that this metric depends on the set of models used and will vary if models are added/removed. However, it allows us to quantify how well a model performs compared to others in the set.

2.3 Hardware and Energy metrics

We perform our experiments on a single NVIDIA A6000 GPU with 48GB VRAM hosted in a local server with Intel Xeon Silver 4210R processor and 128GB RAM, running Ubuntu 20.04-LTS. The server also hosted an NVIDIA A5000 GPU (24 GB), which was used for only one experiment, but otherwise was unused. We also performed some experiments on two other systems to verify the generalizability of our findings, as detailed in Appendix G. We use Pytorch version 2.3 (with CUDA 12.1) and huggingface transformers version 4.41.

We use the popular Code Carbon (Schmidt et al., 2021) and Carbon Tracker (Anthony et al., 2020) packages to measure the energy consumed in different experiments. Jay et al. (2023) demonstrated the suitability and accuracy of CarbonTracker, CodeCarbon, Energy Scope, and Experiment Impact Tracker across various software-based power meter setups, while Bouza et al. (2023) further established the superiority of CodeCarbon and CarbonTracker among these tools. CodeCarbon is especially the most user-friendly and works out of the box, provided appropriate NVIDIA libraries and permissions to Intel RAPL files.

These two packages measure the GPU-power usage using pynvml and CPU-power using Intel RAPL files every $\mathcal{X}$ seconds, and integrates it over time. Carbon-tracker reports sum of these as the total energy. Code-carbon also adds an estimate of the RAM-power being used depending on the RAM size. We use $\mathcal{X} = 10secs$ for a balance between overhead costs and tracking accuracy. We keep the Power Usage Effectiveness (PUE) to the default 1.0 since we run all experiments on the same server, but this implies that the actual energy usage is higher than reported.

During inference, we provide test samples in batches to the LLM, and measure the total energy required for 1024 samples per dataset using these tools. This includes both the input tokenization process by each model's tokenizer and the output generation from the model. We keep the batch size to 8 for most experiments, except on the CNN-DM and SAMSUM dataset for which we use a batch size of 4. While reporting results, we average the energy usage and report the **energy per sample** in mWh (milli-Watt-hour). Unless otherwise stated, these are the default settings used for experiments.

3 Factors Affecting Energy / Accuracy

In this section, we discuss how different task, model, and setup-related factors contribute to the inference energy and accuracy metrics.

3.1 Response time

Response time is an important indicator for actual inference energy, as given a (somewhat) fixed amount of power draw, the energy consumed is proportional to inference response time. To this end, we track the energy for each batch where the tracking interval is set to $1sec$ for the energy-measuring libraries. The batches were formed after sorting the inputs (prompt + query) by length (so that similar-length queries end up together, allowing optimal padding and energy usage).

Figure 1 (left column) compares per-sample average response time and inference energy. They report the comparison for Mistral (rest of the models in Appendix D). Points in the plot correspond to the average scores per query for the individual batches, with distinct color for each dataset.

We find a strong correlation between response time and the inference energy (pearson $r = 0.996$, spearman $rs = 0.968$), indicating a strong possibility of using the response time as a reliable proxy for the energy consumed if demographic factors like location, energy grid, model, etc, are fixed. However, for different datasets, the slope of the dependency is different, which may be because of slightly different power draws due to datasets having different-sized inputs. We also compare the energy measures returned by CarbonTracker and CodeCarbon package and find a good correlation (pearson $r = 0.610$, spearman $rs = 0.912$), indicating reliable tracking.

3.2 Input and Output token length

The complexity of each attention block in a transformer decoder model is given by the following equation (Vaswani et al., 2017), where n is the #input tokens, d is the hidden dimension, and t is the #output tokens.

$$O(n, d, t) = (n.d^2 + n^2.d).t \tag{1}$$

This equation suggests input and output length play a major role in deciding the computational complexity of large language models, which consist of several consecutive layers of multiple such attention blocks, and thereby, the required inference energy. In this section, we attempt to explore the influence of the aforementioned factors in a more systematic manner. Toward that, we first explore a similar setup explained in Section 3.1 to plot the batch-wise energy. Sorting the inputs by their length before batching is especially important because the batches with random input lengths can all get averaged out to have similar values otherwise, making it difficult to visualize the effect of energy with input/output sizes.

Input Length: Figure 1 (middle column) compares per-sample average inference energy with per-sample average input token length. The bigger, spread out clusters primarily belong to the generative tasks, namely, CNN-DM and SAMSUM datasets, due to their larger outputs than the other discriminative tasks, which lay in the bottom clusters. Even though the input size appears as a quadratic term in Eq. 1, we see a linear variation of energy usage with input size. This discrepancy can be attributed to various factors – hardware-related optimizations such as parallel processing, caching mechanisms, and memory hierarchies – that significantly influence the relationship between computational complexity and energy usage. For instance, Transformer models (including LLMs) leverage GPU parallelism to process the input, which effectively mitigates the quadratic scaling predicted by theoretical calculations (Vaswani et al., 2017). Additionally, mechanisms like key-value (KV) caching ensure that input tokens are processed only once, further reducing energy demands compared to the theoretical worst-case scenario (Pope et al., 2022). These optimizations are likely the primary contributors to the observed linear correlation between input token length and energy consumption.

Output Length: Figure 1 (right column) compares per-sample average output token length with per-sample average inference energy for individual batches. Here we observe a similar trend indicating linear increment of energy with output size, in accordance with Eq. 1. Here most tasks except CNN-DM and SAMSUM cluster around the bottom left because of their short outputs, whereas the widespread clusters of CNN-DM and SAMSUM towards the top provide a better visualization of the linear dependency. Note that, the slope of

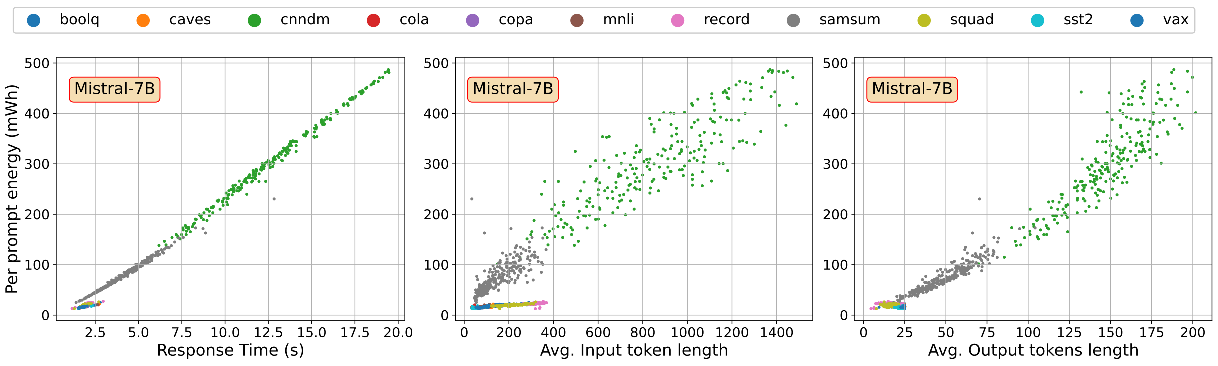


Figure 1: Inference energy vs response time, input and output-token length averaged across samples in a batch plotted across all datasets for **Mistral-7B**. Dots correspond to distinct batches of different datasets.

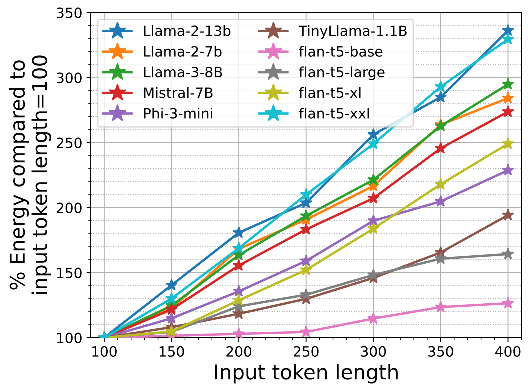


Figure 2: Inference energy on CNN-DM where we vary input token lengths fixing #output tokens to 1.

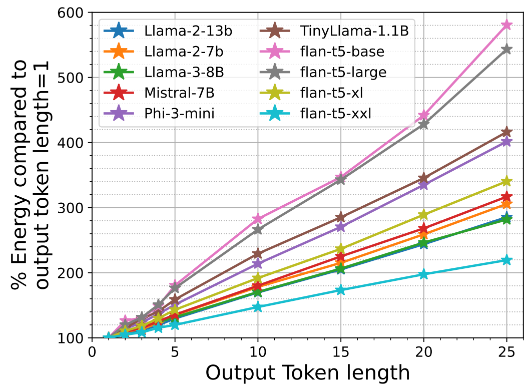


Figure 3: Inference energy on CNN-DM dataset when the output length is varied, keeping input length fixed.

variation of energy consumption is steeper for the output length, in comparison to the slope for the input length despite the input being larger than the output (input Pearson $r = 0.697$, output Pearson $r = 0.952$). These observations indicate *a stronger role played by output length (than the input length) in deciding the inference energy*, which can be explained as follows. During inference, LLMs leverage key-value (KV) caching, which allows tokens to be processed only once, avoiding repeated prior token processing. Furthermore, input processing can be parallelized on GPUs, leading to significant speedups and lower energy consumption (Vaswani et al., 2017). In contrast, generating output tokens is inherently a sequential process, as each token depends on the previous one. This lack of parallelism for output token generation leads to the steeper slope of output token length.

Controlled setup: For better and more exclusive insights into the relation between inference time and input and output length, we perform the following controlled experiment where we fix either input or output length and vary the other.

***Effect of varying input length*:** For CNN-DM dataset, we truncated each input text into N tokens and ask the model to summarize the input, where we vary N from 100 to 400 at fixed intervals of 50 tokens, by means of truncation/padding. If input has more than N token, the query text is truncated, and if the input contains less than N tokens, then padding tokens are added to the left of the input. To eliminate the influence of generated output on inference energy, we stop generation after the first token, which allows us to monitor the influence of input length on model inference energy in an exclusive manner.

Figure 2 plots the inference energy (in terms of %) relative to the energy required for the input with 100 tokens. The results indicate a linear increase in energy with longer input lengths, with a steeper slope observed for decoder-only models. This is likely due to the longer decoder modules present in these models.

***Effect of varying output length*:** Similarly, to study the effect of output length on inference energy exclusively, we take the CNN-DM dataset, fix the input text length and allow the generation length to vary from 1 to 25 tokens. Specifically, we instruct the model to summarize the input text but force the model to stop after it generates the required number of tokens.

Figure 3 plots the energy (in %) relative to the energy required for generating a single token, confirming linear energy increment with generation length. However, the energy of generating the 1^{st} token is much more than additional tokens, e.g. generating 2 tokens takes only about 12% more energy than generating 1 token. This is because the model processes the entire input in the first time step, but only 1 token for subsequent steps (by means of caching the K-V computations for prior tokens). Also, note that, the increment of energy is larger with increasing output length, compared to input length. Contrary to Figure 2, the relative increase is higher for the encoder-decoder family here, attributed to the fact that initial energy requirement is smaller for these families, along with higher jumps in energy with output length.

3.3 Task complexity

Next, we explore whether task 'complexity' has a significant impact on inference energy. Toward that, we conduct a series of controlled experiments where inference energy of two tasks with identical input and output length and distinctly different levels of complexity are compared. Here, we interpret "complexity" based on human cognition.

We compare the inference energy between the VAX-STANCE and CAVES datasets, where the input texts are similar—tweets related to vaccines—but the tasks differ: a 3-class single-label classification for VAX-STANCE versus a 12-class multi-label classification for CAVES. We ensure consistent input length via padding and fix the output to a single token. We find the difference in energy consumption between the two tasks to be very small ($< 1\%$), as shown in Table 3.

Similarly, we compare the average inference energy for the summarization (hard) vs returning the first three sentences of the input (trivial) over the CNN-DM dataset. We find the energy difference between the two tasks as less than 1.3% (Table 3), again indicating that task complexity has hardly any impact on inference energy if input and output lengths are kept fixed. This observation follows from the fact that the computational steps per token are fixed by the model's architecture, with LLMs processing the inputs uniformly, without additional branches or conditional logic that would increase the load for more complex tasks.

	CAVES vs VAX-Stance	**Summarization vs first 3 sent extraction**
fT5-large	−0.2%	1.3%
fT5-xl	0.5%	−0.6%
Phi3-mini	0.4%	−0.1%
Mistral-7B	0.4%	−0.3%
Llama3-8B	0.5%	0.8%

Table 3: Percentage difference in energy consumption for two tasks with very different complexities, keeping input and output lengths same. The differences are very small.

Although we explore task complexity as perceived by humans, results could be different in more complex Chain of Thought (Wei et al., 2022; Wang et al., 2022), Tree of Thought (Yao et al., 2023) and other reasoning frameworks. These frameworks have different ways of processing a task, which could impact the energy consumption based on the settings used. Comparing energy consumption for these different settings would need a study of its own, and is left as future work.

3.4 Model family and size

We now compare the energy usage and normalized accuracy of different models with respect to their size (number of parameters). The model sizes and family have been listed in Section 2.1. Figure 4 compares the size of models with per-sample inference energy, averaged across all samples, showing a linear increase with the size of the model (note that only Y-axis is in log scale in Fig 4), that are individually visible for both the encoder-decoder and the decoder-only models.

Encoder-decoder models typically consume less energy than decoder-only models with a comparable number of parameters. For instance, Flan-T5-xl and Phi-3-mini have a similar parameter count but use significantly less energy. The same pattern holds true for Flan-T5-large versus tinyLlama, and Flan-T5-xxl versus Llama-2-13B. This is because the decoder part in the encoder-decoder models is half the size, which reduces computational demands during the autoregressive decoding phase.

Accuracy metrics: The top row of Table 4 lists the Normalized accuracy (NA) scores for each model across all datasets (performance on each dataset given in Appendix E). Here, we observe that performance depends on both model size and family. Smaller models perform poorly, with TinyLlama

	flan-t5 base	flan-t5 large	flan-t5 xl	flan-t5 xxl	TinyLlama 1.1B	Phi-3 mini	Mistral 7B	Llama-2 7B	Llama-3 8B	Llama-2 13B
***I.** Average **Normalized Accuracy** across all datasets with original settings*										
default	42.85	69.77	55.45	58.12	23.5	41.82	59.91	42.83	55.09	52.31
***II.** Average **change in performance** (%) on Quantization*										
8-bit	0.47	0.03	-1.16	1.43	0.9	-2.92	-0.55	-0.46	-2.43	-4.66
4-bit	1.78	-1.83	-0.63	-0.61	9.66	-4.19	4.27	-1.57	-1.95	-2.76
***III.** Average **change in performance** (%) on introducing targeted phrases in prompts*										
fix-output	-1.7	-2.42	-1.22	0.39	4.71	-8.21	11.64	-12.54	-8.53	3.24
energy-eff	-1.28	-4.2	-0.89	0.84	1.44	9.55	2.52	0.33	-5.19	5.2
+ fix-output	-1.33	-2.68	-1.78	0.74	4.07	4.11	12.4	-15.73	-10.07	5.38
quick	1.29	-3.88	-0.41	-1.71	2.63	8.14	5.82	-4.24	-14.88	3.69
+ fix-output	-0.72	-5.48	-0.27	0.2	4.64	-2.55	12.45	-13.23	-18.88	2.64

Table 4: Accuracy metrics for LLM inferences averaged across all datasets. **I.** Encoder-Decoder models perform better or close to Decoder only models. **II.** Quantization does not decrease performance by much ($< 5\%$) **III.** Performance degrades with most phrases, more so where energy / output token length had also reduced.

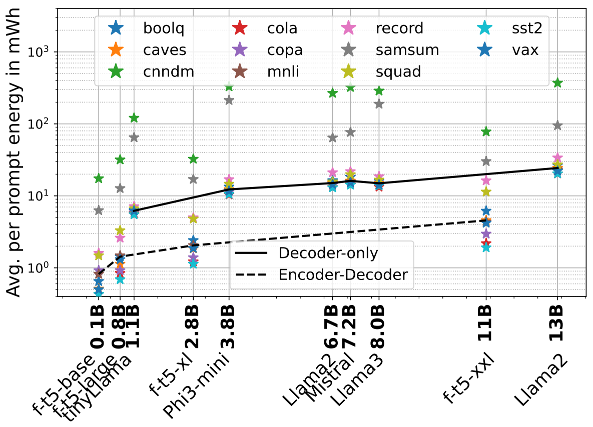


Figure 4: Average per-prompt inference energy vs model size for all models and datasets. The black lines join the median energy for each model family.

giving the worst performance, followed by Phi-3-mini and flan-t5-base. Llama models tend to perform inferior to flan-t5 models of similar size (flan-t5-large, -xl, and -XXL). Mistral-7B is the only exception among the decoder-only models that performs comparably with the flan-t5 family.

As a general statement, it can be commented that selecting models from the encoder-decoder family for NLP tasks is recommended from an energy-efficiency perspective, as well as their performance which is improved by sequence to sequence instruction tuning. In contrast, decoder-only models trained on vast amounts of general data is more suited as an informational chatbot (though instruction tuned versions of Llama3, Mistral and Phi-3 try to bridge the gap).

3.5 Batch size and Quantization

We try to understand the effect of batch size on the energy usage during inference. Intuitively, increasing the batch size should require more energy per batch but we show that it requires less energy per individual sample.

We have used the *bitsandbytes* package to load the transformers model weights in 8-bit and 4-bit quantized format. These quantized versions take up much less GPU memory to load (and thus can be run with larger batch sizes), though the computations still get executed in 16-bit single precision format. We run 4-bit quantized models on all the tasks under different batch sizes and plotted the average energy consumption per sample across all tasks in Figure 5. We observe that increasing the batch size leads to a decrease in per-sample inference energy.

However, the maximum batch size possible is limited by size of the GPU VRAM (48GB for A6000). For certain datasets and larger model combinations, higher batch sizes can result in out-of-memory errors, suggesting that there is an optimal batch size for each dataset and model size combination. To achieve energy-efficient inferences, it is advisable to perform inferences close to this optimal batch size.

A5000 GPU: We repeated this experiment on an NVIDIA A5000 GPU instead of the A6000, reported in Appendix F; however, we did not find a significant difference in the inference time. This also signifies that GPUs with similar usable power require similar energy. Instead, the GPU VRAM size plays a more important role, allowing larger batches.

Quantization Energy: Figure 5 shows the average change in energy for the 4-bit quantized model, while the energy required by original model is

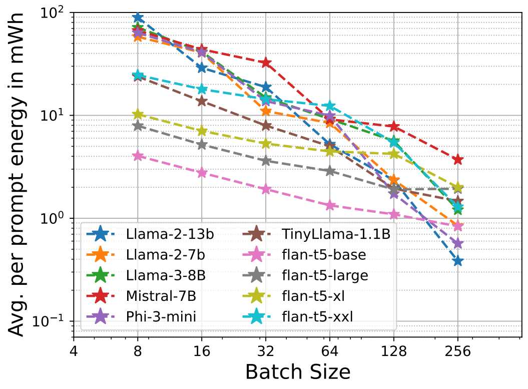


Figure 5: Per-sample inference energy with 4-bit quantized models when the batch size is varied, averaged across all datasets.

given in the Appendix F. Interestingly, keeping all factors same, quantization increases the energy used to almost $2\times$, because of the overhead of additional data format conversions to 16-bit. However, using the 4-bit quantized model with larger batch size of 256 reduces the energy to about $0.33\times$ of the original 16-bit model with batch size of 64. We noticed very similar results with 8-bit quantization and thus, its energy plot is given in Appendix F.

Quantization Accuracy metrics: the change in performance of the quantized models compared to the original is given in the middle set of rows of Table 4. Quantization seems to reduce the performance by less than 5% for some models (mostly decoder-only models in 8-bit quantization and most of the models for 4-bit quantization) and even increases performance slightly for some smaller models, which may have been overfitting earlier. Thus, quantization does not seem to degrade performance too much and should be used to speed up inference time by increasing batch size.

3.6 Effect of targeted phrases in prompts on inference energy

Finally, we attempt to find whether addition of phrases targeted towards energy optimization can affect the inference energy. Specifically, we wanted to see if adding a few more input tokens can lead to a larger decrease in energy by reducing the output token length. In this experiment, for each dataset, we append certain targeted phrases after the default prompt, as shown in Table 5.

Figure 6 reports the % inference energy usage using modified prompts compared to default prompts, averaged across the two generative and rest discriminative tasks. Significant energy reduction is

Directive	Targeted Phrases
default	Read the passage and answer the question with True or False.
quick	<*default*> Answer as quickly as possible.
fix-output	<*default*> Do not output anything else.
energy-eff	<*default*> Answer in energy-efficient way.

Table 5: List of targeted phrases that are used to instruct the LLM for energy-efficient inference.

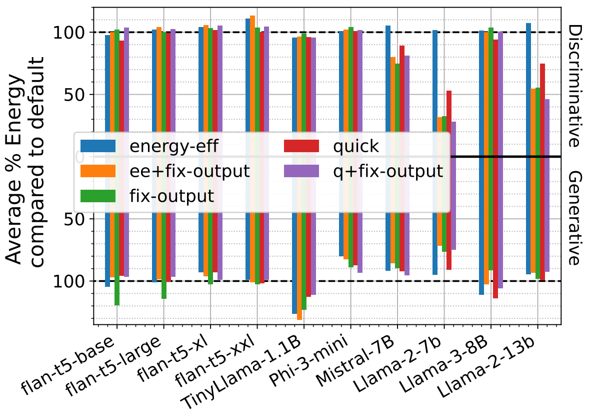


Figure 6: Effect of inserting targeted phrases in prompt on inference energy, as a percentage of default prompt. 'ee': energy-efficient, 'q': quick (see Table 5).

observed for Mistral and Llama-2 models. The reduction is less pronounced in generative tasks, where it mainly results from slightly shorter outputs. However, for discriminative tasks, the reductions are much more significant. This difference arises because these models typically include explanations with their answers, leading to longer outputs by default. By instructing the model to be concise we can limit the output length and, thus, reduce inference energy. However, the change in energy is negligible for the encoder-decoder models, Llama-3 and Phi-3-mini, as they typically generate short, brief answers, leaving little scope for reducing the output. Thereby, additional phrases in the prompt increase the input without reducing the generation, resulting in higher inference energy. TinyLlama always generates long outputs, often stopping only at the generation limit, rendering the targeted phrases useless.

Accuracy metrics: The performance metrics for modified prompts are given in bottom rows of Table 4, we observe that the introduction of such phrases in prompts results in diverse behavior depending on model size and architecture. Performance degrades in most of the cases, especially where output token length had also reduced, taking lesser energy. In summary, we can see that the introduction of such phrases turns out to be useful only for Mistral-7B and Llama-2-13B, considering energy efficiency without affecting performance.

4 Concluding discussions

In this study, we benchmarked the power consumption of various large language models (LLMs) during inference across diverse NLP tasks.

General implications: Our primary high-level takeaways can be summarized as follows: **(1)** Inference energy displays a strong correlation with response time for locally run open-source LLMs, making it a reliable proxy for estimating energy consumption, with significantly less overhead compared to using specialized energy measurement tools. In the case of models accessed through online APIs (such as closed-source models), in the absence of energy-related metrics from the API providers, response time can play as a proxy for energy estimation (see Appendix H). **(2)** While input size shows a linear relationship with energy use, output length has a stronger influence on inference energy. **(3)** Task complexity has little impact on inference time independent of input and output lengths. **(4)** Selecting models from the encoder-decoder family for NLP tasks is recommended from an energy-efficiency perspective, as well as their performance. **(5)** Increasing batch size reduces inference energy. However, it is constrained by the GPU memory availability, recommending an optimal batch size for a particular model, task pair. **(6)** Quantization allows larger batches, resulting in lower energy use without degrading the inference accuracy much. **(7)** Introducing targeted phrases achieves energy reduction for older decoder-only models by restricting their output for discriminative tasks.

Implications in energy-constrained environments: In situations where LLMs are to be deployed in resource/energy constrained settings, our experiments lead to the following insights: **(i)** The response time can be used as a proxy for energy consumed. This is useful not only since measuring power/energy may be difficult on most hardware but also because measuring response time requires much less overhead (in terms of energy/latency) than actually measuring the energy consumption. In low resource settings, this overhead can be large in terms of percentage. For example, we found that removing the energy-tracking libraries and just recording the system time can reduce inference time between 15%–50% under different scenarios. **(ii)** Where possible, input compression and output optimization should be employed. At least, some targeted phrases should be incorporated so that the model generates only what is needed. **(iii)** Fine-tuned encoder-decoder models (especially Flan-T5 models), are better suited for low-resource settings, particularly for NLP tasks such as classification, summarization, etc. **(iv)** Quantized models should be employed, allowing larger batches and larger models on such settings, from the point of memory constraint.

Limitations

Despite the comprehensive analysis and valuable insights provided by this study, the following limitations should be considered. First, the benchmarking experiments were conducted under controlled conditions, which may not fully capture the variability and complexity of real-world deployment environments. The results might differ when models are deployed on different hardware, infrastructure or in varying operational contexts. Also, the study focuses primarily on specific NLP tasks and may not generalize fully to other domains like vision or time series analysis. Additionally, while the study explores a range of system-related factors, it does not account for all possible variables that could influence inference energy, such as network latency or hardware-specific optimizations.

Ethical Considerations

One of the main ethical issues in our experimentation was the substantial energy consumption and carbon emissions it produced. We perform 1024 inferences for 11 datasets over 10 models in several configurations, necessitating multiple repetitions of the inferences, along with several pilot experiments to finalize the experimental setup. This led to an approx total energy use of 3000 kWh. To reduce our environmental impact, we limited our experiments to only 1024 test examples sampled from the datasets. We hope that the insights from this study will lead the community to a much larger reduction of the energy consuption of LLMs.

Acknowledgments

The research is partially supported by a research grant from Accenture Corporation. Paramita Koley is supported by SERB NPDF Fellowship.

References

Lasse F. Wolff Anthony, Benjamin Kanding, and Raghavendra Selvan. 2020. Carbontracker: Tracking and predicting the carbon footprint of training deep learning models. ICML Workshop on Challenges in Deploying and monitoring Machine Learning Systems. ArXiv:2007.03051.

Haoli Bai, Lu Hou, Lifeng Shang, Xin Jiang, Irwin King, and Michael R Lyu. 2022. Towards efficient post-training quantization of pre-trained language models. *Advances in neural information processing systems*, 35:1405–1418.

Jeff Barr. 2019. Amazon ec2 update–inf1 instances with aws inferentia chips for high performance cost-effective inferencing.

Lucía Bouza, Aurélie Bugeau, and Loïc Lannelongue. 2023. How to estimate carbon footprint when training deep learning models? a guide and review. *Environmental Research Communications*, 5(11):115014.

Qingqing Cao, Aruna Balasubramanian, and Niranjan Balasubramanian. 2020. Towards accurate and reliable energy measurement of nlp models. *arXiv preprint arXiv:2010.05248*.

Charlie Chen, Sebastian Borgeaud, Geoffrey Irving, Jean-Baptiste Lespiau, Laurent Sifre, and John Jumper. 2023a. Accelerating large language model decoding with speculative sampling. *arXiv preprint arXiv:2302.01318*.

Lingjiao Chen, Matei Zaharia, and James Zou. 2023b. Frugalgpt: How to use large language models while reducing cost and improving performance. *arXiv preprint arXiv:2305.05176*.

Benoit Courty, Victor Schmidt, Sasha Luccioni, Goyal-Kamal, MarionCoutarel, Boris Feld, Jérémy Lecourt, LiamConnell, Amine Saboni, Inimaz, supatomic, Mathilde Léval, Luis Blanche, Alexis Cruveiller, ouminasara, Franklin Zhao, Aditya Joshi, Alexis Bogroff, Hugues de Lavoreille, Niko Laskaris, Edoardo Abati, Douglas Blank, Ziyao Wang, Armin Catovic, Marc Alencon, Michał Stęchły, Christian Bauer, Lucas-Otavio, JPW, and MinervaBooks. 2024. mlco2/codecarbon: v2.4.1.

Radosvet Desislavov, Fernando Martínez-Plumed, and José Hernández-Orallo. 2021. Compute and energy consumption trends in deep learning inference. *arXiv preprint arXiv:2109.05472*.

Jesse Dodge, Taylor Prewitt, Remi Tachet des Combes, Erika Odmark, Roy Schwartz, Emma Strubell, Alexandra Sasha Luccioni, Noah A Smith, Nicole DeCario, and Will Buchanan. 2022. Measuring the carbon intensity of ai in cloud instances. In *Proceedings of the 2022 ACM conference on fairness, accountability, and transparency*, pages 1877–1894.

Brad Everman, Trevor Villwock, Dayuan Chen, Noe Soto, Oliver Zhang, and Ziliang Zong. 2023. Evaluating the carbon impact of large language models at the inference stage. In *2023 IEEE international performance, computing, and communications conference (IPCCC)*, pages 150–157. IEEE.

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*.

Bogdan Gliwa, Iwona Mochol, Maciej Biesek, and Aleksander Wawer. 2019. Samsum corpus: A human-annotated dialogue dataset for abstractive summarization. In *Proceedings of the 2nd Workshop on New Frontiers in Summarization*, pages 70–79.

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. *Journal of Machine Learning Research*, 21(248):1–43.

Mathilde Jay, Vladimir Ostapenco, Laurent Lefèvre, Denis Trystram, Anne-Cécile Orgerie, and Benjamin Fichel. 2023. An experimental comparison of software-based power meters: focus on cpu and gpu. In *2023 IEEE/ACM 23rd International Symposium on Cluster, Cloud and Internet Computing (CCGrid)*, pages 106–118. IEEE.

Eldar Kurtić, Elias Frantar, and Dan Alistarh. 2024. Ziplm: Inference-aware structured pruning of language models. *Advances in Neural Information Processing Systems*, 36.

Alexandre Lacoste, Alexandra Luccioni, Victor Schmidt, and Thomas Dandres. 2019. Quantifying the carbon emissions of machine learning. *arXiv preprint arXiv:1910.09700*.

Loïc Lannelongue, Jason Grealey, and Michael Inouye. 2021. Green algorithms: quantifying the carbon footprint of computation. *Advanced science*, 8(12):2100707.

Yaniv Leviathan, Matan Kalman, and Yossi Matias. 2023. Fast inference from transformers via speculative decoding. In *International Conference on Machine Learning*, pages 19274–19286. PMLR.

Baolin Li, Yankai Jiang, Vijay Gadepally, and Devesh Tiwari. 2024. Toward sustainable genai using generation directives for carbon-friendly large language model inference. *arXiv preprint arXiv:2403.12900*.

Gauthier Limpens, Stefano Moret, Hervé Jeanmart, and Francois Maréchal. 2019. Energyscope td: A novel open-source model for regional energy systems. *Applied Energy*, 255:113729.

Haokun Liu, Derek Tam, Mohammed Muqeeth, Jay Mohta, Tenghao Huang, Mohit Bansal, and Colin A Raffel. 2022. Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning. *Advances in Neural Information Processing Systems*, 35:1950–1965.

Alexandra Sasha Luccioni et al. 2023. Counting carbon: A survey of factors influencing the emissions of machine learning. *arXiv preprint arXiv:2302.08476*.

Sasha Luccioni, Yacine Jernite, and Emma Strubell. 2024. Power hungry processing: Watts driving the cost of ai deployment? In *The 2024 ACM Conference on Fairness, Accountability, and Transparency*, pages 85–99.

Joseph McDonald, Baolin Li, Nathan Frey, Devesh Tiwari, Vijay Gadepally, and Siddharth Samsi. 2022. Great power, great responsibility: Recommendations for reducing energy for training language models. *arXiv preprint arXiv:2205.09646*.

Gianluca Moro, Luca Ragazzi, and Lorenzo Valgimigli. 2023. Carburacy: summarization models tuning and comparison in eco-sustainable regimes with a novel carbon-aware accuracy. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 14417–14425.

Rakshit Naidu, Harshita Diddee, Ajinkya Mulay, Aleti Vardhan, Krithika Ramesh, and Ahmed Zamzam. 2021. Towards quantifying the carbon emissions of differentially private machine learning. *arXiv preprint arXiv:2107.06946*.

Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, Çağlar Gulçehre, and Bing Xiang. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond. In *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning*, pages 280–290.

David Patterson, Joseph Gonzalez, Urs Hölzle, Quoc Le, Chen Liang, Lluis-Miquel Munguia, Daniel Rothchild, David R So, Maud Texier, and Jeff Dean. 2022. The carbon footprint of machine learning training will plateau, then shrink. *Computer*, 55(7):18–28.

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 preprint arXiv:2104.10350*.

OA Plosskaya, VS Akhripkin, IV Pavlov, et al. 2022. Eco2ai: carbon emissions tracking of machine learning models as the first step towards sustainable ai. In *Doklady Mathematics*, volume 106, pages S118–S128. Springer.

Soham Poddar, Mainack Mondal, Janardan Misra, Niloy Ganguly, and Saptarshi Ghosh. 2022a. Winds of change: Impact of covid-19 on vaccine-related opinions of twitter users. In *Proceedings of the international aaai conference on web and social media*, volume 16, pages 782–793.

Soham Poddar, Azlaan Mustafa Samad, Rajdeep Mukherjee, Niloy Ganguly, and Saptarshi Ghosh. 2022b. Caves: A dataset to facilitate explainable classification and summarization of concerns towards covid vaccines. In *Proceedings of the 45th international ACM SIGIR conference on research and development in information retrieval*, pages 3154–3164.

Reiner Pope, Sholto Douglas, Aakanksha Chowdhery, Jacob Devlin, James Bradbury, Anselm Levskaya, Jonathan Heek, Kefan Xiao, Shivani Agrawal, and Jeff Dean. 2022. Efficiently scaling transformer inference. corr, abs/2211.05102 (2022).

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, Austin, Texas. Association for Computational Linguistics.

Jon Saad-Falcon, Amanpreet Singh, Luca Soldaini, Mike D'Arcy, Arman Cohan, and Doug Downey. 2022. Embedding recycling for language models. *arXiv preprint arXiv:2207.04993*.

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)*, pages 1–9. IEEE.

Victor Schmidt, Kamal Goyal, Aditya Joshi, Boris Feld, Liam Conell, Nikolas Laskaris, Doug Blank, Jonathan Wilson, Sorelle Friedler, and Sasha Luccioni. 2021. Codecarbon: estimate and track carbon emissions from machine learning computing. *Cited on*, 20.

Roy Schwartz, Jesse Dodge, Noah A Smith, and Oren Etzioni. 2020. Green ai. *Communications of the ACM*, 63(12):54–63.

Raghavendra Selvan, Nikhil Bhagwat, Lasse F Wolff Anthony, Benjamin Kanding, and Erik B Dam. 2022. Carbon footprint of selecting and training deep learning models for medical image analysis. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 506–516. Springer.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2019a. Superglue: A stickier benchmark for general-purpose language understanding systems. *Advances in neural information processing systems*, 32.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2019b. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *International Conference on Learning Representations*.

Boshi Wang, Sewon Min, Xiang Deng, Jiaming Shen, You Wu, Luke Zettlemoyer, and Huan Sun. 2022. Towards understanding chain-of-thought prompting: An empirical study of what matters. *arXiv preprint arXiv:2212.10001*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Carole-Jean Wu, Ramya Raghavendra, Udit Gupta, Bilge Acun, Newsha Ardalani, Kiwan Maeng, Gloria Chang, Fiona Aga, Jinshi Huang, Charles Bai, et al. 2022. Sustainable ai: Environmental implications, challenges and opportunities. *Proceedings of Machine Learning and Systems*, 4:795–813.

Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. 2023. Smoothquant: Accurate and efficient post-training quantization for large language models. In *International Conference on Machine Learning*, pages 38087–38099. PMLR.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. Tree of thoughts: Deliberate problem solving with large language models, 2023. *URL https://arxiv. org/pdf/2305.10601. pdf*.

A Additional Literature Survey

Sustainable Large Language Models: Schwartz et al. (Schwartz et al., 2020) discuss the growing compute cost of deep learning research and advocate for making efficiency an evaluation criterion alongside accuracy and related measures with a focus on making AI both greener and more inclusive. Lacoste et al. (Lacoste et al., 2019) consider various factors like energy grid, energy draw of server, make and model of training hardware to assess the environmental impact of machine learning algorithms. Following that trend, recent literature focuses on various alternatives to reduce the inference energy of large language models. Among the black-box approaches, Li et al (Li et al., 2024) append generation directives to user prompts for carbon-friendly LLM inferences. (McDonald et al., 2022) focus on techniques to measure energy usage and propose various hardware and datacenter-oriented settings that can be tuned to reduce energy consumption for training and inference for language models. Frugal GPT (Chen et al., 2023b) explores strategies like prompt adaptation, LLM cascade, and LLM approximation for reducing the inference cost for a large set of queries. On the contrary, white-box approaches include speculative decoding (Leviathan et al., 2023), speculative sampling (Chen et al., 2023a), prunning (Kurtić et al., 2024), embedding recycling (Saad-Falcon et al., 2022), quantization (Bai et al., 2022; Frantar et al., 2022; Xiao et al., 2023), and many more.

Tools for measuring energy impact: Researchers propose various tools for tracking the realtime energy consumption and carbon emissions during model training and inferences. These tools include CodeCarbon (Courty et al., 2024), CarbonTracker (Anthony et al., 2020), Experiment impact tracker (Henderson et al., 2020), EnergyScope (Limpens et al., 2019), etc. Green Algorithms (Lannelongue et al., 2021) is another online tool, enabling a user to estimate and report the carbon footprint of their computation. Eco2AI is another open-source package to help data scientists and researchers to track energy consumption and equivalent CO_2 emissions of their models in a straightforward way (Plosskaya et al., 2022). Carburacy (Moro et al., 2023) proposes the first carbon-aware accuracy measure that captures both model effectiveness and eco-sustainability for generative transformer-based models (Moro et al., 2023). Researchers explore energy impact analysis in terms of carbon footprints of ML algorithms in various domains, namely differential privacy (Naidu et al., 2021), medical image analysis (Selvan et al., 2022), etc.

Benchmarking energy tools: Researchers benchmark the tools for measuring carbon footprints in various configurations for various deep learning based ML models. Cao et al (Cao et al., 2020) compare energy returned by software-based energy measurements with hardware power meter (WhattsUPMeter) on various NLP models and report experiment impact tracker as not so accurate. Jay et al (Jay et al., 2023) qualitatively and experimentally compare several software-based power meters against high-precision physical power meters while executing various intensive workloads, where they conclude that for measuring energy, Carbon Tracker, Code Carbon, Energy Scope, and Experiment Impact Tracker are suitable fits. However, Bouza et al (Bouza et al., 2023) establish that the energy value reported by CodeCarbon is closest to Wattmeter, followed by CarbonTracker, with more variability between infrastructures.

Benchmarking LLMs: Existing works have attempted to study the inference energy of LLMs from various perspectives. Everman et al. (2023) evaluated energy usage of GPT models on question-answering tasks, employing Software Carbon Intensity (SCI) released by green software. Samsi et al. (2023) provides a detailed account of energy usage during inference for question-answering tasks for Llama models on various GPU architectures. Liu et al. (2022) study the energy consumption trade-off for fine-tuning vs few-shot learning for both encoder-decoder and decoder-only models on SuperGLUE and RAFT tasks, measuring energy in FLOPS. These preliminary works mostly ignore the performance-energy tradeoff.

In recent times, Li et al. (2024) compare the effect of prompt directives on inference energy and performance for Llama-2 models for 3 question-answering tasks. Luccioni et al. (2024) offers a comparatively detailed account of inference energy usage while choosing LLMs and tasks from a diverse range, showing the dependency of energy with model size and architecture, task type, and output token length. While these works provide an initial account of inference energy usage of LLMs in different configurations, they are often limited by the number and diversity of the models and tasks. Our work is the first to provide a comprehensive benchmarking of energy consumption of LLMs during inference for a diverse range of configurations, i.e., variation of input and output token length (for both coarse and granular level), correlation with response time, model size and family, task complexity, batch size, quantization level, presence of targeted phrases, and energy directives in prompt for a more complete set of LLM and tasks. Table 6 presents an overview of existing approaches and their limitations.

B Model Descriptions

Check Table 7

C Dataset Examples and Metrics

Check Table 8

D Scatter Plots of batch-wise energy tracking

Check Figure 7

E Original Accuracy Metrics of individual datasets

Check Table 9

F Batch Size & Quantization experiments

Check Figure 9

G Testing on other systems

To verify the generalizability of our findings, we additionally ran limited experiments on two different systems.

System Descriptions: The first system contains an NVIDIA Tesla P100 GPU with 16GB VRAM paired with Intel Xeon Gold 6126 CPU and 128GB RAM. For the second system we used Kaggle [1] which contains an NVIDIA Tesla T4 GPU with 16GB VRAM, Intel Xeon CPU and 30GB RAM.

Experiments: Figure 8 shows the energy consumptions of 2 representative models from each family – Mistral-7B and flan-t5-large on the different systems. The results are averaged over 4 representative datasets – BoolQ, Mnli, CNN-DM and SQuaD-v2. We observe very similar trends for each model across different systems, even though the magnitude is different. We found that the system with P100 required less energy than our original setup, though with a limited GPU VRAM, which does not allow very high batch sizes. Interestingly, the

[1] www.kaggle.com/code

Reference	LLM	Tasks/Datasets	Observations
(Everman et al., 2023)	GPT-styled models (4)	10 manual prompts	Study on the energy-performance tradeoff of LLMs.
(Samsi et al., 2023)	LlaMA models (3)	QA tasks (2)	Study inference cost on diverse GPUs.
(Desislavov et al., 2021)	DNN-based NLP models (7)	GLUE (9)	Study model complexity vs inference cost.
(Liu et al., 2022)	T5 models (3), GPT-styled models (3)	NLP tasks (9), RAFT	Study energy consumption of few-shot PEFT vs in-context learning.
(Li et al., 2024)	Llama models (2)	QA tasks (3)	Study effect of prompt directives on inference cost.
(Luccioni et al., 2024)	Flan-T5 models (4), BLOOMz models (4)	NLP + vision tasks (10)	Study inference energy vs model complexity, task type, output, etc.
Our work	GPT styled models (6), Flan-T5 models (4)	NLP tasks (11)	Study inference cost vs input, output, response time, model size & family, task complexity, quantization, batch size, targeted phrases

Table 6: Comparison of our approach with existing literature on benchmarking inference cost of LLMs

Model	Model description link
Tiny-LLama (1.1B params)	https://huggingface.co/TinyLlama/TinyLlama-1.1B-Chat-v1.0
Phi-3-mini (3.8B params)	https://huggingface.co/microsoft/Phi-3-mini-4k-instruct
Mistral-7B (7.2B params)	https://huggingface.co/mistralai/Mistral-7B-Instruct-v0.2
Llama-2-7B (6.7B params)	https://huggingface.co/meta-llama/Llama-2-7b-chat-hf
Llama-3-8B (8.0B params)	https://huggingface.co/meta-llama/Meta-Llama-3-8B-Instruct
Llama-2-13B (13B params)	https://huggingface.co/meta-llama/Llama-2-13b-chat-hf
Flan-T5-base (248M params)	https://huggingface.co/google/flan-t5-base
Flan-T5-large (783M params)	https://huggingface.co/google/flan-t5-large
Flan-T5-xl (2.8B params)	https://huggingface.co/google/flan-t5-xl
Flan-T5-xxl (11B params)	https://huggingface.co/google/flan-t5-xxl

Table 7: Links to specific models versions we used in our experiments

system with T4 GPU consumes a lot more power even though it is a weaker system.

We further hypothesize that modern improvements like flash-attention-2 will benefit newer GPU architectures and newer models like Phi-3 and Llama-3 more. However, the overall trends should still be similar.

H Energy proxy for black-box models

Our work demonstrates that for locally run open-source LLMs, inference time is a reliable proxy for estimating energy consumption, with significantly less overhead compared to using specialized energy measurement tools. However, in the case of models accessed through online APIs (such as closed-source models), it is difficult to estimate the energy from just the response time due to a variety of factors, including network latency, number of concurrent user requests, type of batch scheduling/processing, etc. However, to date, the API providers do not provide any energy-related metrics; hence, there exists no way yet to estimate the energy for such black-box models. For the purpose of improving sustainability research, API providers should provide some energy metrics with the outputs. In the absence of such estimates, inference time can be used to compare energy use of different black-box strategies. A workaround for variations in network latency could be to run experiments multiple times and averaging the time taken across all experiments. This will hopefully even out the effects of the varying factors such as network latency, when comparing different settings under the same hardware. Though this would not completely eliminate the effect of network latency, as it is not a mean-zero random variable, and is influenced by a variety of complex, hardware-related factors, this is the most straightforward workaround for now. In future work, it would be beneficial to develop a model that accounts for network latency variations to improve energy consumption estimation.

Task	**Dataset**	**Metric**	**Input Prompt with query**
Linguistic acceptability check	COLA (GLUE)	Macro-F1	Answer in binary whether the given sentence is grammatically, semantically, and logically acceptable.
Logical entailment	Mnli (GLUE)	Macro-F1	Select the stance of the premise towards the hypothesis: Entailment (0), Neutral (1) or Contradiction (2).
Sentiment classification	SST2 (GLUE)	Macro-F1	Classify the sentiment of the sentence as positive (1) or negative (0).
Contextual question answering	Boolq (SuperGLUE)	Macro-F1	Read the passage and answer the question with True (1) or False (0).
Causal reasoning	COPA (SuperGLUE)	Macro-F1	Select Choice1 (0) or Choice2 (1) that is a cause/effect of a given premise.
Entity Question answering	ReCoRD (SuperGLUE)	F1	Read the passage and find the entity that replaces "@placeholder" inside the query.
Extractive question answering	SQuAD v2	F1	Read the context and answer the question with a phrase from the context.
Document summary generation	CNN-DM	avgROUGE-1,2,L	Summarize a given news article.
Dialogue summary generation	SAMSum	avgROUGE-1,2,L	Summarize a given dialogue sequence.
3 class vaccine-stance classification	VAX-Stance	Macro-F1	Classify into one of the following three vaccine stances: Pro-Vaccine, Anti-Vaccine or Neutral.
12 class multi-label anti-vaccine concerns classification	CAVES	Macro-F1	Classify into one or more of these anti-vax classes: 0: ineffective, 1: ingredients, 2: rushed **...** 11: side-effect.

Table 8: List of tasks/datasets we experimented on along with input prompts/descriptions

Dataset	flan-t5 base	flan-t5 large	flan-t5 xl	flan-t5 xxl	TinyLlama 1.1B	Phi-3 mini	Mistral 7B	Llama-2 7B	Llama-3 8B	Llama-2 13B
cola	23.5	68.8	31.2	24.9	22.1	44.1	54.1	22.1	55.6	30.3
mnli	54.2	88.0	79.4	87.6	22.6	24.6	46.1	28.5	50.6	41.3
sst2	33.0	74.5	32.7	40.2	47.4	51.8	75.1	48.6	71.1	57.7
boolq	71.3	86.4	91.6	88.5	45.5	46.4	74.7	61.4	65.2	64.5
copa	33.3	41.9	26.0	42.7	36.7	64.9	60.3	53.2	73.7	56.1
squad	57.2	59.5	59.5	58.4	18.3	20.2	31.0	47.6	16.9	44.1
cnndm	21.4	20.8	16.5	16.2	12.7	18.4	21.7	18.9	19.6	22.9
samsum	40.0	44.6	46.1	45.3	21.9	16.0	25.8	28.4	22.1	29.3
caves	11.9	30.0	37.0	38.9	4.8	24.3	34.5	12.5	28.2	20.8
vax	20.3	53.0	52.1	54.6	23.0	47.9	52.7	50.2	52.5	54.2

Table 9: Original average ROUGE/F1 metrics for LLM inferences averaged across all datasets.

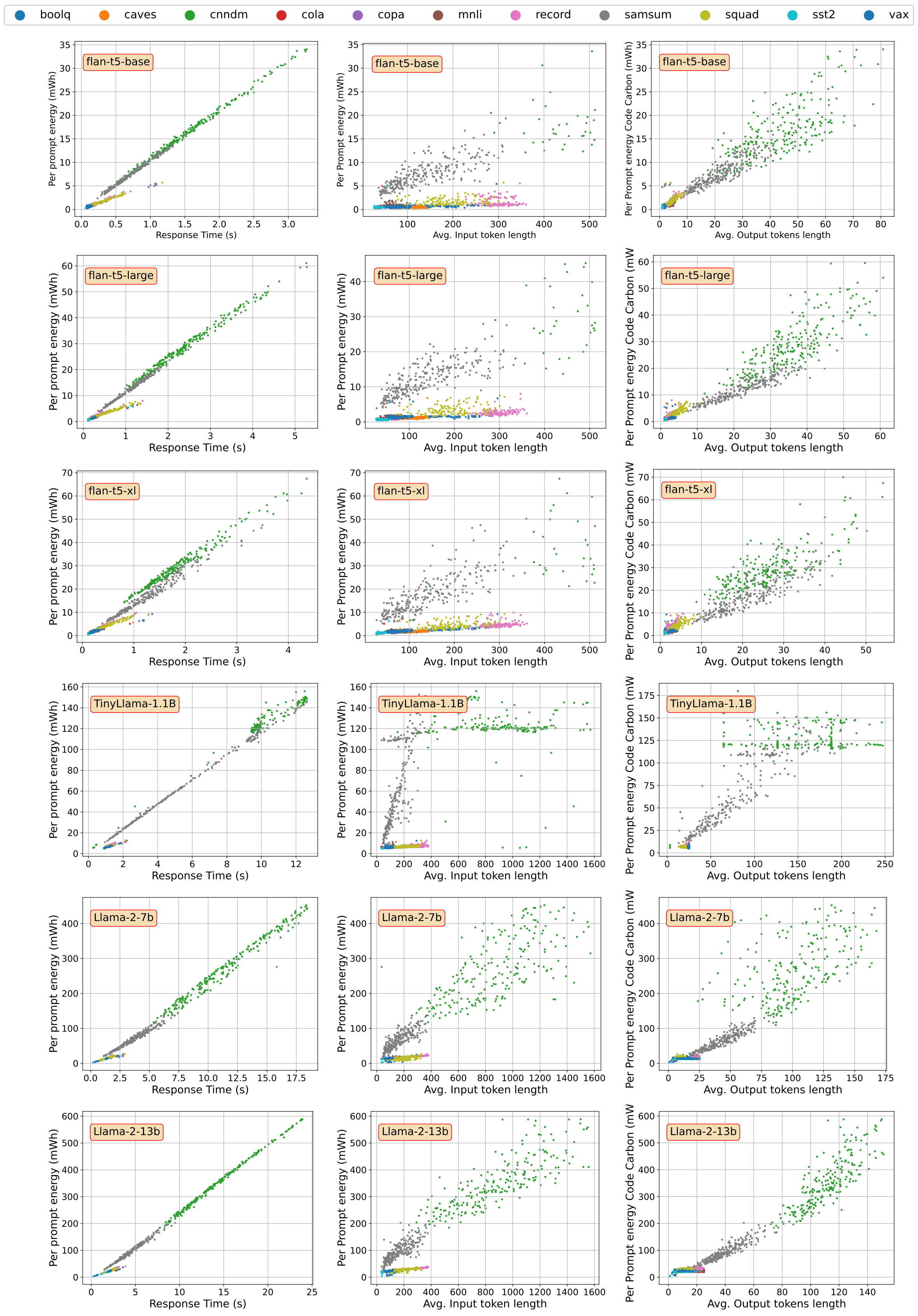


Figure 7: Average per-sample inference energy vs average per-sample response time, input and output-token length across all datasets for different models where points in the image correspond to individual batches of different datasets.

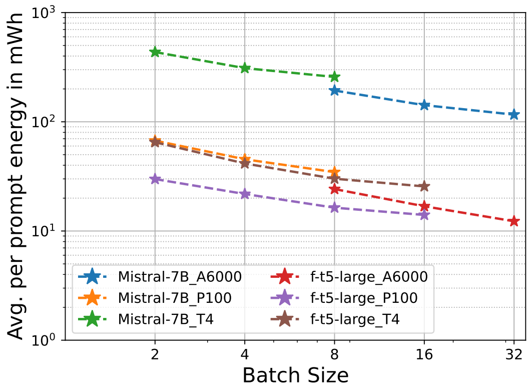
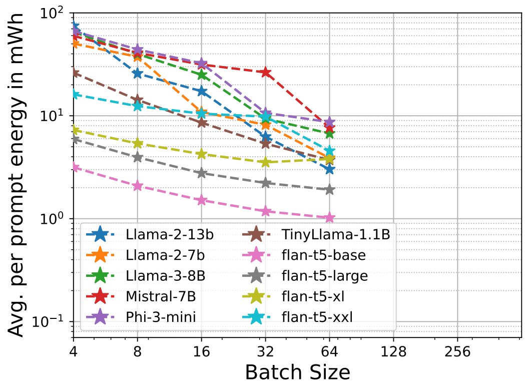
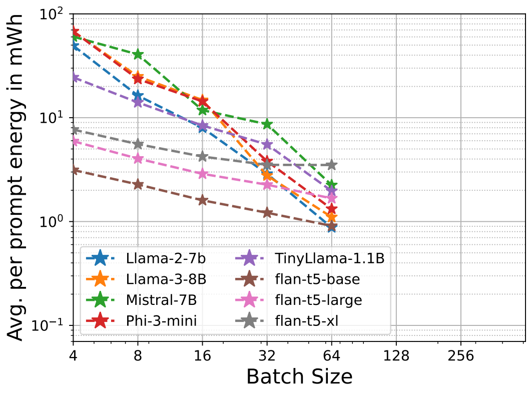


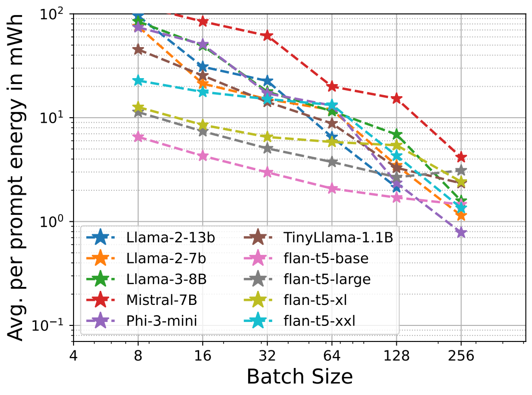
Figure 8: Energy consumption of running Mistral-7B and flan-T5-large on different systems (identified by their GPUs), averaged across 4 datasets. A6000 refers to the original setup.



(a) Per-sample inference energy averaged across all datasets when the batch size is varied. Overhead of using 4-bit precision can increase energy to almost $2\times$ for the same batch size. However, a 4-bit model in BS=256 takes only about $0.33\times$ the energy as default model in BS=64



(b) Per-sample inference energy averaged across all datasets when the batch size is varied, on the A5000 GPU instead of A6000.



(c) Per-sample inference energy averaged across all datasets with 8-bit quantized models.

Figure 9: Additional Batch size experiments on the A5000 GPU, and using 8-bit quantization.