Energy Efficient or Exhaustive? Benchmarking Power Consumption of LLM Inference Engines

CHENXU NIU, Texas Tech University, USA
WEI ZHANG, Lawrence Berkeley National Laboratory, USA
YONGJIAN ZHAO, Texas Tech University, USA
YONG CHEN, Texas Tech University, USA

Large Language Models (LLMs) have remarkable advancements in recent years and have revolutionized the field of natural language processing. To reduce latency and improve inference throughput, many inference engines have been proposed such as vLLM, TensorRT-LLM, and DeepSpeed. However, there is no comprehensive analysis on the power consumption and energy efficiency of these inference engines. In this paper, we benchmark the power consumption of LLM inference engines on one single GPU node with 2 H100 GPUs and provide a fine-grained analysis by decomposing the inference lifecycle into two stages: the setup stage including engine initialization and model loading; and the token generation stage. For each stage, we further measure power consumption across key system components, including GPU, CPU, and DRAM. This breakdown analysis allows us to identify energy bottlenecks of inference lifecycle and gain deeper insights into the energy efficiency of modern inference engines.

CCS Concepts: • **Hardware** → **Power and energy**; • **Computing methodologies** → **Natural language processing**; • **Computer systems organization** → *Parallel architectures.*

Additional Key Words and Phrases: LLM Inference Engine, Energy Efficiency, Power Profiling

1 INTRODUCTION

Large Language Models (LLMs) such as GPT-series (GPT-1 [21], GPT-2 [22], GPT-3 [3]), OPT [31], BERT [9], LLaMA-series [26], Mixtral [13], and Falcon [1] have exploded in popularity due to their new generative capabilities on reasoning, summarization [17, 29], and text generation tasks in natural language processing domain [16, 18, 28**?**]. However, as these LLMs grow in parameter size and complexity, the performance of inference becomes a key bottleneck in real-world deployments of cloud environments or data centers. For instance, the parameters of LLaMA series have been scaled from 7 billions to 405 billion in only 2 years (from 2023 to 2025). LLM inference is both computationally intensive and latency-sensitive, especially in applications that require real-time or high-throughput responses.

To improve throughput and hardware utilization of inference, many inference engines have been proposed in recent years. Typical examples are vLLM [15], TensorRT-LLM [27] and DeepSpeed [2]. These engines applied different techniques such as dynamic batching, model parallelism, quantization, and memory-efficient caching to optimize LLM inference. For example, the "PagedAttention" technique of vLLM improves memory efficiency for high-throughput batch inference; TensorRT-LLM integrates many optimization techniques, like kernel fusion and quantization and Triton support to cooperate latest GPU architecture; DeepSpeed improves scalability and applies distributed optimizations on the most powerful and largest language models.

While prior work primarily focused on the energy demands of training LLMs, since it requires massive GPU resources and GPU hours. But recent evidence [4, 14] shows that inference process now dominates the energy footprint in large-scale deployments. According to a report on the operational lifecycle of LLMs from Amazon Web Services (AWS) [12], inference consumes nearly 90% of the energy consumption. Inference is a continuous process that runs every time a user interacts with a system powered by an LLM, while training only occurs during the development or fine-tuning phases of a LLM. This constant demand makes inference the primary driver of computational expense, latency, and energy use. Measuring and optimizing inference efficiency have become essential for researchers and companies to reduce both costs and environmental footprint of AI.

Although inference now dominates the energy footprint of LLMs, power consumption and energy efficiency of inference still receive less attention than the energy costs of training and fine-tuning LLMs. At the same time, there is no comprehensive evaluations of power consumption across inference engines during the lifecycle of inference. This gap is particularly concerning for cloud environments and data centers, since power consumption directly impacts operational costs and environmental sustainability. For instance, a single high-end GPU, such as the NVIDIA A100 and H100, can consume hundreds of watts during inference, and scaling this across clusters amplifies the energy demand exponentially.

In this paper, we conduct the first comprehensive evaluation of power consumption and benchmark the energy efficiency across several widely used LLM inference engines, including vLLM, TensorRT-LLM, DeepSpeed, and Transformers. Our study provides a fine-grained analysis by decomposing the inference lifecycle into two stages: (1) the setup stage, which includes engine initialization and model loading steps; and (2) the token generation stage, where actual inference takes place. For each stage, we further measure real-time power consumption across key hardware components, including the total power input, GPU, CPU, and DRAM. Contributions of this paper include benchmarking several inference engines and presenting a breakdown analysis to share key insights about energy efficiency, and answering the following questions:

Authors' addresses: Chenxu Niu, chenxu.niu@ttu.edu, Texas Tech University, Lubbock, Texas, USA, 79409; Wei Zhang, Lawrence Berkeley National Laboratory, Berkeley, California, USA, zhangwei217245@lbl.gov; Yongjian Zhao, Texas Tech University, Lubbock, Texas, USA, yongjian.zhao@ttu.edu; Yong Chen, Texas Tech University, Lubbock, Texas, USA, yong.chen@ttu.edu.

- During the setup stage, what is the power consumption of initializing inference engines and loading LLMs, until the first token is generated?
- During the token generation stage, how does power consumption and energy efficiency vary across inference engines, considering GPU, CPU, and memory components?
- What is the relationship between energy efficiency and throughput? We investigate the hypothesis that higher throughput can improve energy efficiency by reducing per-token energy cost.
- Is there a single inference engine that optimizes energy efficiency across all scenarios?

2 BACKGROUND AND RELATED WORK

This section provides the background and reviews prior work relevant to our study, including the inference engines we evaluate and the related research on energy consumption during LLM inference.

2.1 Inference Engines

To evaluate energy efficiency, we focus on four representative and widely-used LLM inference engines: Transformers, vLLM, DeepSpeed and TensorRT-LLM.

The Hugging Face Transformers [10] is a widely used and highly flexible framework to deploy LLMs in all kinds of environments. It is designed to support a wide range of models and hardware. But it is not a dedicated and optimized inference engine, that's why we use it as a baseline for comparison. vLLM [15] is an open-source inference engine designed for high-throughput LLM serving. Its key innovation is PagedAttention technique, which optimizes memory management by partitioning key-value caches to enable efficient batch inference with minimal memory overhead. Microsoft's DeepSpeed [2] is a distributed training and inference engine optimized to improve the scalability of models. It utilizes techniques like Zero Redundancy Optimizer (ZeRO) and model parallelism to reduce memory and computational overhead. The unique ability of DeepSpeed is to handle the most powerful and largest-scale LLMs across multiple devices. TensorRT-LLM [27] is NVIDIA's specialized LLM inference engine optimized for execution on Nvidia GPUs. It integrates Triton and targets low-latency and high-performance inference for real-time applications.

2.2 Energy Consumption in Inference

While significant research has addressed energy consumption in LLM training, energy efficiency research on LLM inference engines remains underexplored.

Several studies [11, 20, 24] quantified the carbon footprint of training and fine-tuning NLP models. Desislavov et al. [8] analyzed energy cost in small scale NLP models during inference, but their findings do not extend to modern LLMs or LLM inference engines. Recently, some work [23, 30] studied the relationship between energy consumption and the parameters of LLMs. In their paper, they use only one engine to serve LLM or even use naive ways to deploy LLMs. They still ignore the role of engines and the comparison across LLM inference engines.

In summary, these studies do not systematically compare inference engines or measure energy efficiency across diverse workloads.

3 PROBLEM FORMULATION AND MEASUREMENT APPROACH

This section presents a mathematical model to quantify the power consumption of LLM inference lifecycle on inference engines and provides the measurement methodology we used for breakdown analysis.

3.1 Problem Formulation

In general, inference is the process of using the deployed LLM to generate responses to user queries. If we apply inference engines to serve LLM inference, we can breakdown the whole inference process into two stages: the setup stage and token generation stage.

The total energy consumption of an inference engine during the inference lifecycle can be modeled into two stages as follows:

$$\begin{aligned} E_{\text{total}} &= E_{\text{Setup}} + E_{\text{TG}} \quad (1)\\ &= E_{\text{IE}} + E_{\text{LM}} + T \cdot E_{\text{PT}} \quad (2) \end{aligned}$$

where:

- E_{Setup}: energy consumed during the setup stage, which includes the initialization process of the inference engine and the loading process of the LLMs.
- E_{TG}: energy consumed during the token generation process.
- E_{IE}: energy consumed during the initialization process of inference engines.
- E_{LM}: energy consumed during the loading process of LLMs.
- T: number of tokens generated during token generation process.
- E_{PT}: energy consumed per token during token generation process.

3.2 Measurement Methodology and Breakdown Analysis of Energy Consumption

For each stage, we measure power consumption across key system components, including GPU, CPU, and DRAM. The energy consumption of each component can be expressed as the following:

$$E_{\text{PowerInput}} = E_{\text{GPU}} + E_{\text{CPU}} + E_{\text{DRAM}} + E_{\text{Others}} \quad (3)$$

To collect and measure the power draw information of key component, we utilize multiple hardware-specific tools. Specially, we use IPMI [5] to monitor total system power draw; for GPU partition, we use NVIDIA Management Library [7] to collect real-time GPU power usage during the inference process; then we use Intel RAPL [6] to record CPU and DRAM subsystem power consumption.

In the quation, E_{Others} denotes for the power consumed by other system components not explicitly monitored, such as motherboard controllers, storage devices, fans, and power conversion losses. While we do not isolate these components individually, their contribution is included in the total system power reported by IPMI (usually accounts for 20%-25%).

Table 1. Energy Consumption and Latency of Loading Inference Engines and Model Loading for Different Model Sizes

Phase	Engine/Model	Metrics				
		Latency (s)	Total Energy (J)	GPU Energy (J)	CPU Energy (J)	DRAM Energy (J)
E_{IE}	**vLLM**	48.39	27209.42	7298.76	11985.02	982.83
	Transformers	2.89	1632.91	413.02	518.35	70.81
	DeepSpeed	2.92	1691.98	419.71	538.71	71.98
	TensorRT-LLM	30.21	18722.34	4566.90	8627.28	627.02
E_{LM}	**vLLM – 1B**	3.81	2302.98	691.22	1028.89	80.73
	vLLM – 3B	9.11	5502.72	1792.73	2328.41	194.54
	vLLM – 8B	11.64	7184.06	2480.81	2659.39	251.56
	Transformers – 1B	1.29	748.43	229.19	323.24	27.85
	Transformers – 3B	1.75	1020.65	311.28	469.02	38.37
	Transformers – 8B	3.31	1963.59	586.35	726.50	84.18
	DeepSpeed – 1B	1.23	718.59	218.47	316.14	28.38
	DeepSpeed – 3B	1.77	1024.17	313.37	474.21	39.11
	DeepSpeed – 8B	3.23	1951.83	574.56	712.26	82.07
	TensorRT-LLM – 1B	2.62	1734.79	522.29	762.74	56.21
	TensorRT-LLM – 3B	4.27	3022.91	917.63	1492.41	102.67
	TensorRT-LLM – 8B	7.92	4892.89	1492.7	2088.12	162.83

To avoid variability of the system power information, we perform multiple runs (100 times in our experiments) and compute average values for each component.

4 EVALUATION

In our evaluation section, we benchmark the power consumption of vLLM, DeepSpeed, TensorRT-LLM and Transformers and conduct a breakdown analysis.

4.1 Experimental Setup

The experiments were performed on a dedicated server with two NVIDIA H100 GPUs (94GB memory each), paired with two Intel Xeon Gold 6426Y CPUs (16 cores, 32 threads each) and 503GB of RAM.

Since we only have 2 GPUs in this experiments, we chose to evaluate small-scale Llama series: Llama 3.1-8B, Llama 3.2-1B and Llama 3.2-3B. In the following sections, we use 1B, 3B and 8B to denote these LLMs. For a fair comparison, we set the decoder temperature as 0.8 and the top-p value as 0.95 across all engines. We chose Alpaca [25] as our dataset, which contains 52,002 prompts generated by OpenAI's text-davinci-003 engine [19].

4.2 Evaluation Metrics

To comprehensively evaluate the energy efficiency of LLM inference frameworks, we define the following metrics:

- During the setup stage, we measured latency and energy consumption of initialing engines and loading LLMs until first token.
- During the token generation stage, we measured three types of energy consumption during inference: energy per token, energy per response and energy per second.
- Energy/throughput ratio: we evaluated the relationship between energy efficiency and throughput of each engine.

4.3 Latency and Energy Consumption of Initialing Engines and Loading Models

The table 1 provides a detailed comparison of TTFT (Time to First Token) latency and energy consumption during the setup stage. We list the latency, total energy, GPU energy, CPU energy and DRAM energy consumed during the engine initialing and models loading process. In the table, red color denotes the most suboptimal energy efficiency, and green color denotes the highest energy efficiency.

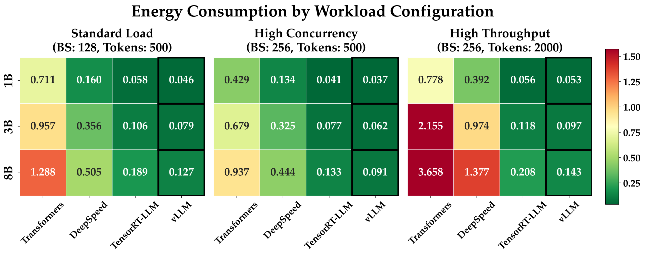
As shown in the table, Transformers and DeepSpeed take only 2 to 3 seconds for engine loading, while TensorRT-LLM and vLLM consume over 30 seconds. In loading LLMs step, vLLM still takes the longest time and consumes the most energy across all LLMs. Transformers and DeepSpeed are the most efficient engine to load LLMs, and they only consume less than 1/3 energy compared to vLLM. The table demonstrates that Transformers and DeepSpeed have better energy efficiency compared to TensorRT-LLM and vLLM during the setup stage.

The experimental results show that the deployment of vLLM and TensorRT-LLM is much slower than that of Transformers and DeepSpeed. In general, vLLM's initialization involves setting up "PagedAttention" for efficient memory management and configuring distributed inference for token generation process; TensorRT-LLM requires extensive model compilation, including layer fusion, and hardware-specific CUDA kernel generation. These optimization techniques are designed to accelerate the token generation step, however, it will increase the workload of initializing engines and loading models. In contrast, Transformers uses dynamic computation graphs with limited hardware-specific optimizations for inference. These features will help it to have faster deployment.

4.4 Energy Consumption during Token Generation Stage

Three workload configurations were used during the token generation stage to simulate the workloads of real-world scenarios:

- **Standard Load**: Batch size of 128, output tokens of 500
- **High Concurrency**: Batch size of 256, output tokens of 500
- **High Throughput**: Batch size of 256, output tokens of 2000



Standard Load (BS: 128, Tokens: 500)	Transformers	DeepSpeed	TensorRT-LLM	vLLM
1B	0.711	0.160	0.058	0.046
3B	0.957	0.356	0.106	0.079
8B	1.288	0.505	0.189	0.127

High Concurrency (BS: 256, Tokens: 500)	Transformers	DeepSpeed	TensorRT-LLM	vLLM
1B	0.429	0.134	0.041	0.037
3B	0.679	0.325	0.077	0.062
8B	0.937	0.444	0.133	0.091

High Throughput (BS: 256, Tokens: 2000)	Transformers	DeepSpeed	TensorRT-LLM	vLLM
1B	0.778	0.392	0.056	0.053
3B	2.155	0.974	0.118	0.097
8B	3.658	1.377	0.208	0.143

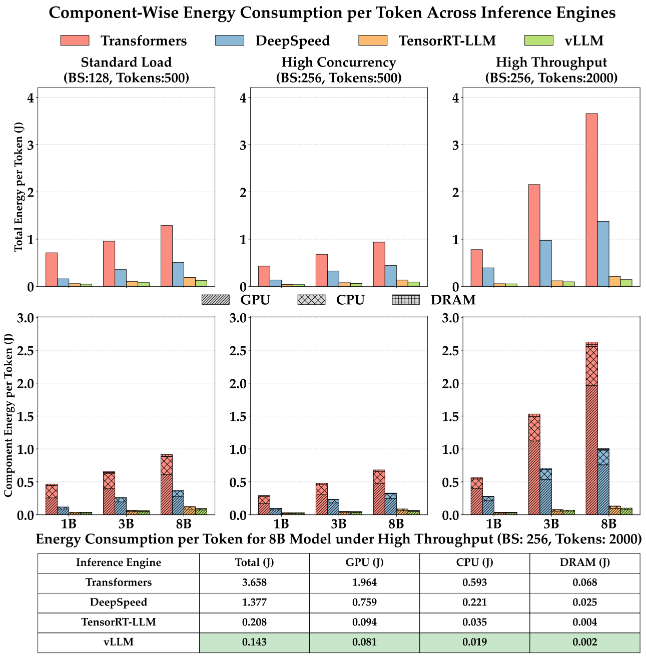
Fig. 1. Heatmap of Energy per Token across All Engines

4.4.1 Energy per Token. The most important metrics to measure energy efficiency is "energy per token". Figure 1 presents a heatmap of the metrics across the three workload configurations on these engines. Green blocks indicate the better energy efficiency. Under Standard Load, all engines have low energy per token. In High Concurrency, vLLM achieves the lowest energy per token by optimizing GPU utilization for large batch sizes. For High Throughput workload, vLLM and TensorRT-LLM are better engines compared to the other two engines. Transformers consumes the largest energy to generate tokens since it struggles with large scale parameters of LLMs.

In summary, vLLM and TensorRT-LLM are energy efficient on all kinds of scenarios, especially on high-concurrency and high-throughput tasks.

Figure 2 provides a breakdown component-wise analysis on energy consumption per token. GPU consumes more than 50% of the total energy. vLLM still achieves the best energy efficiency across all seperate components. Under High Throughput workloads, vLLM achieved the lowest GPU energy consumption at only 0.081 J/token, which is only 4% of GPU energy consumption of Transformers. TensorRT-LLM followed at 0.094 J/token and DeepSpeed recorded 0.759 J/token. A similar trend is observed for CPU and DRAM energy consumption. vLLM also maintains the lowest CPU and DRAM energy usage, significantly outperforming the other engines.

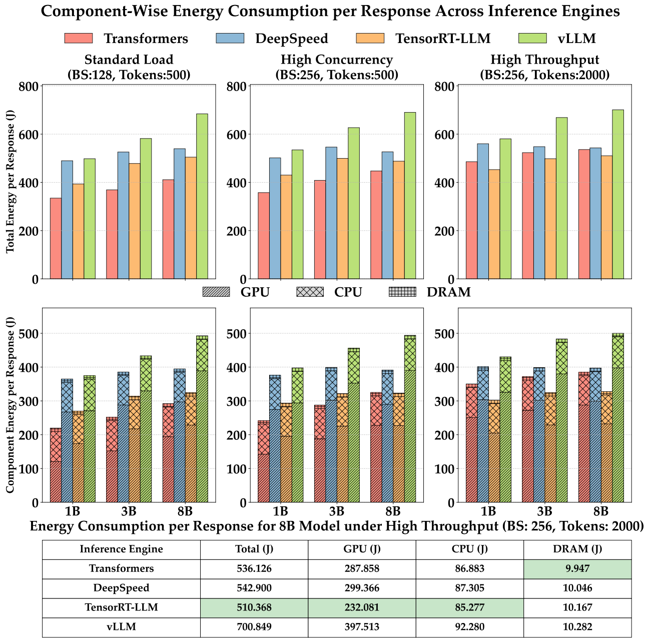
As mentioned above, vLLM and TensorRT-LLM achieve much better energy efficiency per token during inference due to their optimization techniques tailored for LLM inference processing. The advantages are even bigger when they have larger batch sizes, which means they have better scalability in HPC environments.



Energy Consumption per Token for 8B Model under High Throughput (BS: 256, Tokens: 2000)

Inference Engine	Total (J)	GPU (J)	CPU (J)	DRAM (J)
Transformers	3.658	1.964	0.593	0.068
DeepSpeed	1.377	0.759	0.221	0.025
TensorRT-LLM	0.208	0.094	0.035	0.004
vLLM	0.143	0.081	0.019	0.002

Fig. 2. Component-Wise Energy Consumption per Token Across Inference Engines



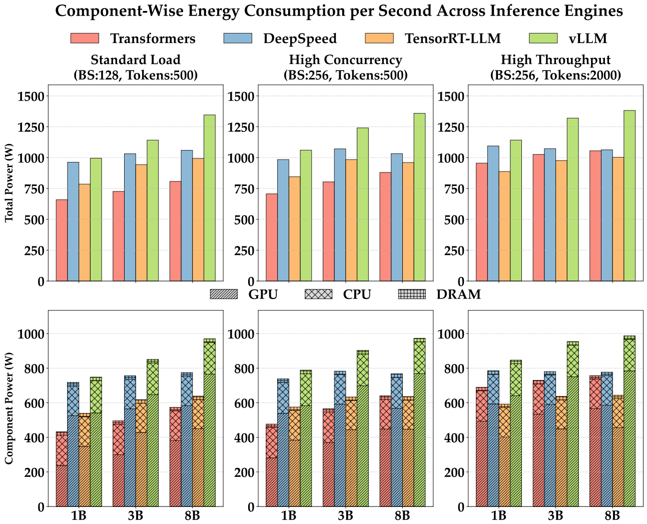
Energy Consumption per Response for 8B Model under High Throughput (BS: 256, Tokens: 2000)

Inference Engine	Total (J)	GPU (J)	CPU (J)	DRAM (J)
Transformers	536.126	287.858	86.883	9.947
DeepSpeed	542.900	299.366	87.305	10.046
TensorRT-LLM	510.368	232.081	85.277	10.167
vLLM	700.849	397.513	92.280	10.282

Fig. 3. Component-Wise Energy Consumption per Response Across Inference Engines

4.4.2 Power Consumption Per Response. Figure 3 shows the results of "Energy per Response" across all workloads. For High Throughput, TensorRT-LLM achieved the lowest total energy at 510.4 J, while its efficient GPU energy of 232.1 J accounts for 45% of total energy per response. vLLM consumed the highest energy at 700.8 J per response. Transformers and DeepSpeed recorded similar total energies and component energies. Across all components, CPU and DRAM contributions were consistent across engines, with 12–13% and 1–2% of total, respectively. Both figures indicate that GPU optimizations primarily drive energy differences.

While vLLM demonstrates the best energy efficiency per token, it shows the worst energy consumption per response. This contrast arises because different engines applies different ending policy of inference, the actual number of output tokens are different. vLLM generates the largest number of tokens per response on Alpaca dataset. So the total energy per response is highest compared to other engines.



Power Consumption for 8B Model under High Throughput (BS: 256, Tokens: 2000)

Inference Engine	Total (W)	GPU (W)	CPU (W)	DRAM (W)
Transformers	1054.955	566.429	170.964	19.572
DeepSpeed	1063.147	586.240	170.968	19.674
TensorRT-LLM	1003.051	456.120	167.599	19.981
vLLM	1382.202	783.968	181.993	20.277

Fig. 4. Component-Wise Energy Consumption per Second Across Inference Engines

4.4.3 Power Consumption Per Second. As shown in Figure 4, Transformers and DeepSpeed have similar energy numbers on all seperate components. TensorRT-LLM is the most energy efficient engine on the metrics of "energy per second". vLLM still has the largest Watts number across all the engines on each component. When they serve 1B model, the difference of total energy across all engines is small. DeepSpeed have similar energy per second with vLLM, but it generated less tokens than vLLM. Transformers consumes the least DRAM energy due to its memory utilization technique. The trend of this figure is similar to the figure of "energy per response".

4.5 Energy Efficiency/Throughput Ratio

Figure 5 illustrates the relationship between "energy per token" and "throughput" across all engines. The plot validates the hypothesis that higher throughput can improve energy efficiency by reducing per-token energy cost. As shown in the following equation, if we multiply the value of energy per token and throughput together, the results would be the energy per second. From the previous figures, we know GPU and CPU dominates the power consumption.

$$\begin{aligned} \text{Watts} &= \text{Watts} \cdot \text{s /token} \cdot \text{token/s} && (4) \\ &= \text{J/token} \cdot \text{token/s} && (5) \\ &= \text{Energy per token} \cdot \text{Throughput} && (6) \end{aligned}$$

When a system is at idle states, GPUs and CPUs consume substantial baseline power at about 120W. As throughput increases during High Concurrency or High Throughput workload, GPU and CPU power draw rises to about 1000W. But energy per Token decreases at higher throughput, as the fixed idle power is amortized over more tokens generated per unit time.

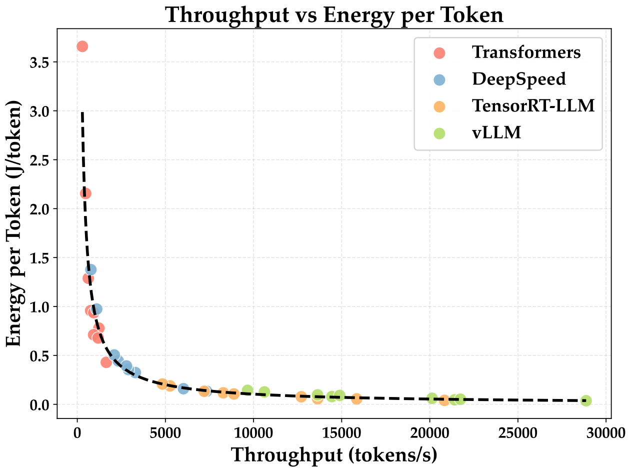


Fig. 5. The Relationship between Energy per Token and Throughput

If researchers can improve the performance and throughput of a inference engine, the energy efficiency will also be improved.

4.6 Is There a Single Overall Best Solution on Energy Efficiency?

Our evaluation shows that no single inference engine universally optimizes energy efficiency during the lifecycle of inference. As shown in Table 1, during the setup stage, Transformers and DeepSpeed are the most efficient in both latency and energy consumption on each component. But during the token generation stage, vLLM and TensorRT-LLM dominate in energy efficiency per token, especially under High Concurrency or High Throughput workload.

These findings provide insights for engine selection. For latency-sensitive or on-demand environments, DeepSpeed and Transformers may be better choices. For an intensive inference environment, vLLM or TensorRT-LLM are the preferred choices.

5 CONCLUSION AND FUTURE WORK

In this paper, we benchmarked the power consumption of inference engines during the whole LLM inference lifecycle, including the setup stage and token generation stage. Our experimental results also provide a fine-grained breakdown analysis across key system components, including GPU, CPU, and DRAM for each stage. By profiling the power consumption experiments, we offer a detailed understanding of where and how energy is consumed during LLM inference lifecycle. These insights reveal critical inefficiencies and guide the development of more energy-aware inference engines and frameworks.

In the near future, we plan to expand this study to further advance the development of energy-efficient LLM inference systems. First, we will evaluate more inference engines on more larger-scale LLM models to assess the scalability of energy efficiency across model sizes and complexities. We will extend our experiments to large-scale GPU clusters, analyzing the impact of distributed inference and inter-GPU communication on energy consumption, particularly for large-scale, real-world workloads. Then, we will propose to design and develop a novel energy-efficient inference engine or framework that integrates the strengths of existing systems. Our findings serve as a foundation for future research on sustainable AI deployment and system-level optimization.

ACKNOWLEDGMENTS

We are thankful to the anonymous reviewers for their valuable feedback. This research is supported in part by the National Science Foundation under grant OAC-2404438 (REPACSS: Empowering Scientific Discovery through Renewable Energy Powered Advanced Computing Systems and Services), and CNS-1939140 (A U.S. National Science Foundation Industry-University Cooperative Research Center on Cloud and Autonomic Computing). We are also very grateful to the REPACSS and High Performance Computing Center of Texas Tech University for providing HPC resources for this project.

REFERENCES

[1] Ebtesam Almazrouei, Hamza Alobeidli, Abdulaziz Alshamsi, Alessandro Cappelli, Ruxandra Cojocaru, Mérouane Debbah, Étienne Goffinet, Daniel Hesslow, Julien Launay, Quentin Malartic, et al. 2023. The falcon series of open language models. *arXiv preprint arXiv:2311.16867* (2023).

[2] Reza Yazdani Aminabadi, Samyam Rajbhandari, Ammar Ahmad Awan, Cheng Li, Du Li, Elton Zheng, Olatunji Ruwase, Shaden Smith, Minjia Zhang, Jeff Rasley, et al. 2022. Deepspeed-inference: enabling efficient inference of transformer models at unprecedented scale. In *SC22: International Conference for High Performance Computing, Networking, Storage and Analysis*. IEEE, 1–15.

[3] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems* 33 (2020), 1877–1901.

[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*. 1–7.

[5] Intel Corporation. 2006. Intelligent Platform Management Interface Specification, v2.0. https://www.intel.com/content/www/us/en/products/docs/servers/ipmi/ipmi-specifications.html.

[6] Intel Corporation. 2023. *Intel 64 and IA-32 Architectures Software Developer's Manual, Volume 3B: System Programming Guide, Part 2.*

[7] NVIDIA Corporation. 2023. pynvml: Python bindings for the NVIDIA Management Library (NVML). https://github.com/NVIDIA/pynvml.

[8] 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.

[9] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*. 4171–4186.

[10] Hugging Face. 2024. Transformers: State-of-the-art machine learning for PyTorch TensorFlow and JAX. *GitHub* (2024).

[11] 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 (2020), 1–43.

[12] Gadi Hutt, Vibhav Viswanathan, and Adam Nadolski. 2019. Deliver high performance ML inference with AWS Inferentia. (2019).

[13] 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, et al. 2024. Mixtral of experts. *arXiv preprint arXiv:2401.04088* (2024).

[14] Peng Jiang, Christian Sonne, Wangliang Li, Fengqi You, and Siming You. 2024. Preventing the immense increase in the life-cycle energy and carbon footprints of llm-powered intelligent chatbots. *Engineering* 40 (2024), 202–210.

[15] Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th Symposium on Operating Systems Principles*. 611–626.

[16] Tianyang Liu, Qi Tian, Jianmin Ye, LikTung Fu, Shengchu Su, Junyan Li, Gwok-Waa Wan, Layton Zhang, Sam-Zaak Wong, Xi Wang, et al. 2024. ChatChisel: Enabling Agile Hardware Design with Large Language Models. In *2024 2nd International Symposium of Electronics Design Automation (ISEDA)*. IEEE, 710–716.

[17] Chenxu Niu, Wei Zhang, Suren Byna, and Yong Chen. 2023. PSQS: Parallel Semantic Querying Service for Self-describing File Formats. In *2023 IEEE International Conference on Big Data (BigData)*. IEEE, 536–541.

[18] Chenxu Niu, Wei Zhang, Mert Side, and Yong Chen. 2025. ICEAGE: Intelligent Contextual Exploration and Answer Generation Engine for Scientific Data Discovery. In *Proceedings of the 37th International Conference on Scalable Scientific Data Management*. 1–10.

[19] OpenAI. 2022. text-davinci-003 [Large language model]. https://platform.openai.com/docs/models/gpt-3. Accessed: YYYY-MM-DD.

[20] 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* (2021).

[21] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. 2018. Improving language understanding by generative pre-training. (2018).

[22] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog* 1, 8 (2019), 9.

[23] 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)*. IEEE, 1–9.

[24] Emma Strubell, Ananya Ganesh, and Andrew McCallum. 2020. Energy and policy considerations for modern deep learning research. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 34. 13693–13696.

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

[26] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971* (2023).

[27] N Vaidya, F Oh, and N Comly. 2023. Optimizing inference on large language models with nvidia tensorrt-llm, now publicly available.

[28] 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 (2017).

[29] Xi Wang, Gwok-Waa Wan, Sam-Zaak Wong, Layton Zhang, Tianyang Liu, Qi Tian, and Jianmin Ye. 2024. ChatCPU: An Agile CPU Design and Verification Platform with LLM. In *Proceedings of the 61st ACM/IEEE Design Automation Conference*. 1–6.

[30] Grant Wilkins, Srinivasan Keshav, and Richard Mortier. 2024. Offline energy-optimal llm serving: Workload-based energy models for llm inference on heterogeneous systems. *ACM SIGENERGY Energy Informatics Review* 4, 5 (2024), 113–119.

[31] Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. 2022. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068* (2022).