Energy Efficiency of LLM Inference Across Various AI Accelerators

Giacomo Brunetta, Zhiling Lan, Xingfu Wu, Valerie Taylor

Abstract

In recent years, **numerous hardware accelerators** have been developed to meet the rising demand for **machine learning (ML) workloads**, and **Large Language Models inference** in particular. **GPUs are currently the standard** for ML training and inference. However, they require substantial data movements that hurt performance and increase power consumption, making systems extremely energy-intensive. In response, many companies, including **Intel, AMD, and Google**, as well as numerous startups such as **Groq, SambaNova, Cerebras, and Graphcore**, have introduced **specialized accelerators for ML workloads that leverage a dataflow design**, which aims to reduce data movement and thus improve both performance and power consumption.
This article presents a **comparative analysis of the performance and energy efficiency** of various AI accelerators and GPUs for large language model (LLM) inference, using popular open-source models evaluated on both synthetic and real-world datasets.

Background

Time Complexity

LLMs are a type of DNN based on the **Transformer** architecture. Their distinctive feature is the presence of Self-Attention Layers.

$$Attention(Q, K, V) = \text{softmax}(\frac{QK^\top}{\sqrt{d_k}})V$$

The computational complexity of attention is $O(nd^2) + O(n^2d)$
Where n is the sequence length and d is the model's hidden dimension. Generating multiple tokens requires multiple forward passes, each quadratic in complexity in its current sequence length, making the **complexity of the sequence generation cubic**.

A popular optimization known as **KV caching** reduces **sequence generation complexity to quadratic in sequence length** by storing K and V matrices and updating them incrementally.

Hardware Accelerators

Current AI accelerators can be grouped into three main categories:

- **GPUs:** use the **Single Instruction, Multiple Threads** model and feature a hierarchical memory system that combines a large, DRAM-based global memory with a fast, low-latency shared memory scoped to each thread block.
- **TPUs:** feature VLIW cores that operate on tensors of data. In particular, TPUs feature **matrix multiplication functional units**. TPUs feature a complex memory hierarchy comprising a global DRAM and local scratchpad memories.
- **Dataflow architectures** interconnect many cores over an **on-chip network**. This allows each core to use its own local scratchpad memory only, giving predictable access latency and enabling fine-grained, compiler-driven data scheduling.

Methodology

To **test the performance** of accelerators on LLM Inference, we use different **open-source models** from the **Llama, Qwen, Mistral,** and **DeepSeek** families. In particular, we focus on models of size **7B**, **14B**, **32B**, and **70B** parameters. Using random batches of tokens, we model the performance as a function of **input, output tokens, and batch size**. For all runs, we collect both **performance metrics** (latency, throughput, TTFT) and **power consumption** data (average power, peak power, energy).
As our baseline machine, we use a **Polaris node** at Argonne National Laboratory that features an AMD Milan CPU and **four Nvidia SXM A100** GPUs with 40GB of HBM2 memory. The inference framework adopted is **vLLM**, which offers a state-of-the-art KV cache system and leverages the FlashAttention library to ensure efficient computation.

Results

Given a machine, the factors that impact performance are sequence **length, batch size**, and **tensor-parallel size**.

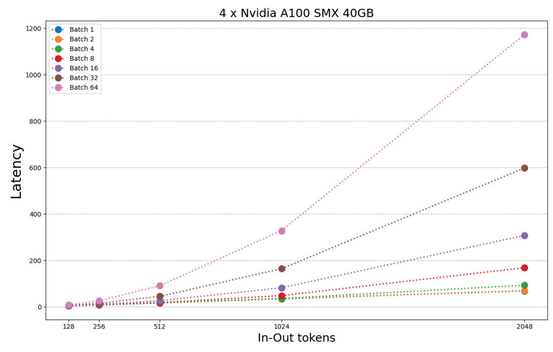

Fig. 1: **Lantecy** (s) of DeepSeekR1 Distill-Llama70B on Polaris.

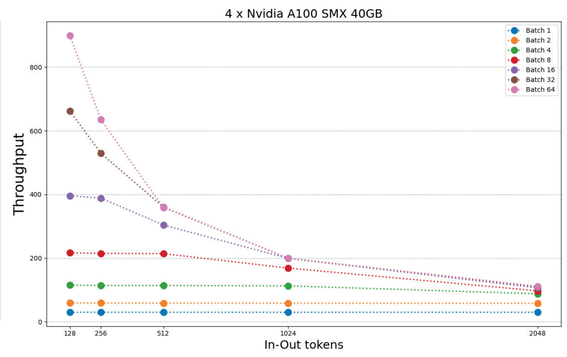

Fig. 2: **Throughput** (token/s) of DeepSeekR1 Distill-Llama70B on Polaris.

As shown in Figs. 1 and 2, longer sequences increase the latency and reduce the throughput, while larger batch sizes cause the latency to increase, but improve the overall system throughput. As shown in Fig. 3 it also reduced the energy consumption per token.

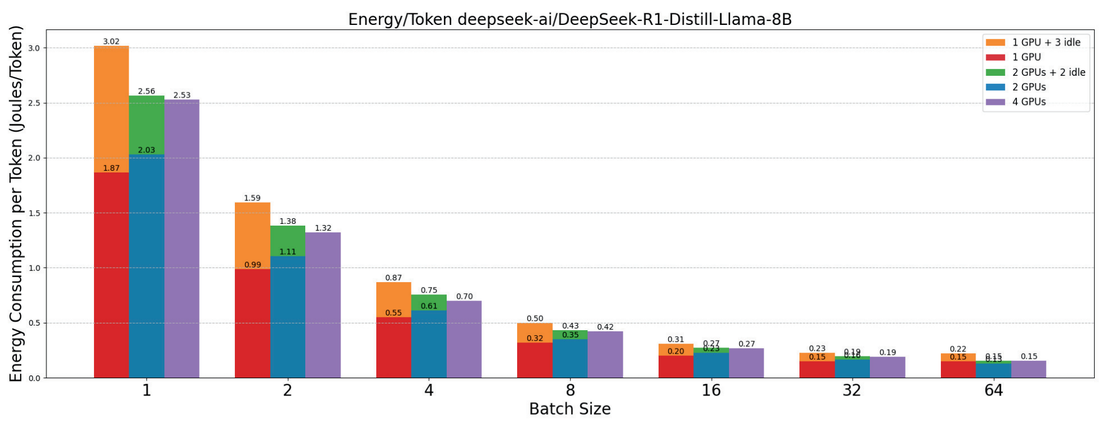

Fig. 3: **Energy / Token** (input and output) for DeepSeekR1 Distill-Llama8B on Polaris.

Increasing the number of GPUs to enable **tensor-parallelism** allows **fitting larger models in memory** and **increases the throughput** significantly for smaller models. This increase in throughput comes at the cost of **increased energy consumption** (as shown in Fig. 3), but never enough to justify letting GPUs idle.

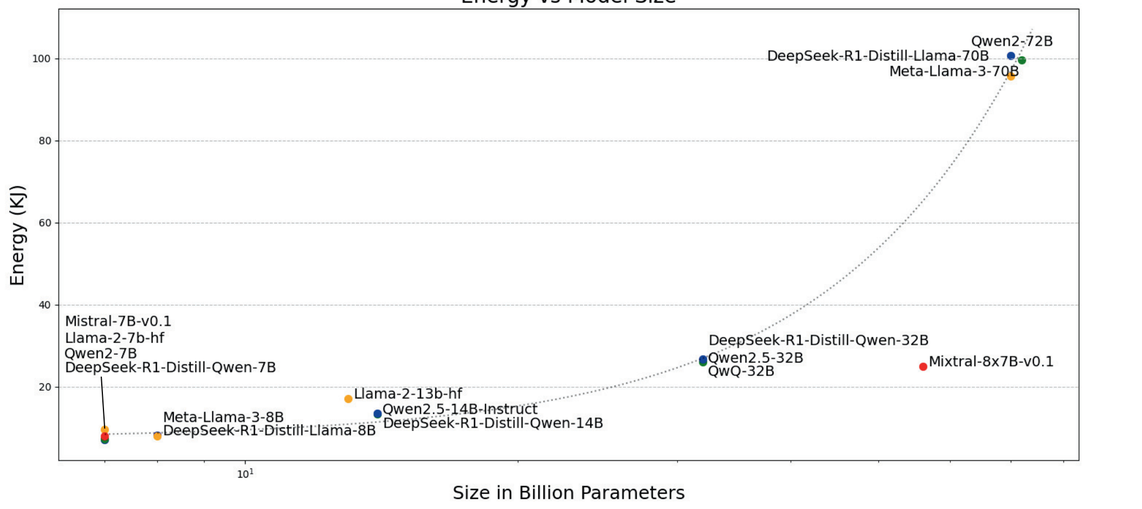

Fig. 4: **Energy vs Model Size** on Polaris (1024 input-output tokens, batch size 16)

As Fig. 4 shows, **energy usage grows with the square of model size**, but architecture choices can significantly lower that cost. For instance, the **Mixture-of-Experts** model Mixtral uses about 62% less energy than a similarly sized dense model. Likewise, adding **Grouped Query Attention** cuts consumption of around 35% for models in the 7-8B range.

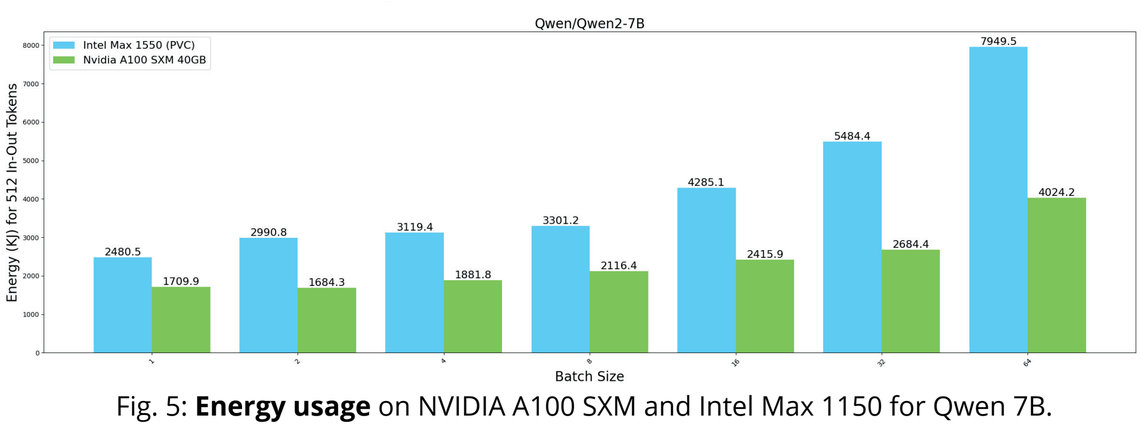

Fig. 5: **Energy usage** on NVIDIA A100 SXM and Intel Max 1150 for Qwen 7B.

Fig. 5 shows an example of an **energy consumption comparison** between **Nvidia** (A100 SXM 40GB) and **Intel** (Max 1550) **GPUs**.
The results show that the Nvidia A100 consistently outperforms the Intel GPU in terms of energy consumption.
We are currently working on replicating similar tests on other models of **Nvidia and AMD GPUs** and several **Dataflow machines** available at Argonne National Laboratory to gain further insights on how the architecture impacts the performance and the energy consumption.