# Beyond Throughput: Performance and Energy Insights of LLM Inference Across AI Accelerators

Giacomo Brunetta*, Varuny Sastry†, Xinfu Wu†, Valerie Taylor†, Michael E. Papka†*, Zhiling Lan*†
*University of Illinois Chicago, Chicago, IL, USA †Argonne National Laboratory, Lemont, IL, USA

***Abstract*—Serving large language models (LLMs) is highly resource-intensive and requires specialized hardware acceleration. While GPUs remain the dominant solution, dataflow accelerators are emerging as a compelling alternative. In this study, we present an extensive empirical performance analysis of six datacenter-class GPUs from Nvidia, AMD, and Intel, alongside two dataflow AI accelerators from Cerebras and SambaNova, using fourteen different open-source LLMs. Our evaluation focuses on investigating the main factors that impact LLM inference, including model size, batch size, quantization, and multi-GPU scaling under various parallelism strategies. Importantly, we analyze both performance and energy efficiency, providing an energy-aware comparison across accelerator types. Our experimental results show that dataflow AI accelerators achieve an order-of-magnitude improvement in throughput and latency for small ($\leq 8$) batch sizes compared to GPUs. On the other hand, GPUs offer larger HBM memories and benefit from a simpler programming model, allowing for more flexibility in terms of batch size, which is particularly beneficial in offline inference. These insights provide practical guidance for LLM inference in diverse deployment scenarios.**

***Index Terms*—LLM, Serving, Inference, Energy Efficiency, GPU, Dataflow.**

## I. Introduction

In recent years, Large Language Models (LLMs) have demonstrated remarkable capabilities across a wide range of natural language processing tasks, including translation, question answering, chain-of-thought reasoning, and code generation. Their success has led to a wide range of popular applications that rely on LLMs in their backend, including search engines, chatbots, and code agents. LLMs are notoriously demanding in terms of computing power, memory, and energy consumption, often requiring dedicated acceleration. The International Energy Agency (IEA) projects that the energy consumption of accelerated servers will grow 30% annually, primarily due to AI-related demand [1]. In particular, companies like NVIDIA and AWS report that 80-90% of their AI-related energy consumption comes from model serving (inference), rather than training [2]. Consequently, minimizing the energy consumption of LLM inference is crucial for the economic and environmental sustainability of these tools.

In this paper, we evaluate the energy efficiency of LLM inference workloads across a range of AI accelerators. Two classes of AI accelerators are widely used for LLM inference: general-purpose GPUs and specialized dataflow accelerators [3]–[5]. The key difference between them lies in their architectural design and in how they handle computation and data movement for deep learning workloads [6]. GPUs use the single-instruction, multiple-thread (SIMT) model to execute thousands of threads in parallel, accelerating the matrix and vector operations. Dataflow accelerators stream data through a network of compute units in a highly pipelined, parallel, and often asynchronously scheduled manner. GPUs are the dominant platform for LLM acceleration, while specialized dataflow accelerators are emerging as a promising alternative.

Recent studies have examined LLM inference across different hardware accelerators. Most prior work focuses on NVIDIA GPUs [7]–[13], while others also include AMD and Intel GPUs [14], [15] and dataflow accelerators [15]–[17]. However, key gaps remain: *there is limited cross-vendor evaluation directly comparing modern GPUs, with most studies emphasizing performance while overlooking energy efficiency; and there is little investigation into dataflow accelerators.*

Bridging these gaps is challenging, as LLM inference depends on many interacting factors beyond the choice of accelerators. We identify three *main challenges*. First, model choice matters: the size and architecture of an LLM strongly influence computation and memory footprint. Second, modern GPUs often employ quantization, which can reduce memory requirements and enhance computational efficiency. Importantly, the impact of quantization varies considerably across architectures, underscoring the need for explicit, architecture-aware evaluations. Third, batch size complicates the problem, as larger batches improve throughput but increase latency and memory pressure, making tradeoffs highly dependent on both model and hardware configurations. Together, these factors highlight the inherent complexity of deriving systematic insights into the performance and energy efficiency of LLM inference across diverse accelerators.

In this study, we present a systematic performance analysis of LLM inference across a range of AI accelerators, including six datacenter-class GPUs from NVIDIA, AMD, and Intel, alongside two dataflow accelerators from Cerebras and SambaNova. We evaluate fourteen open-source LLMs with different architectures and scales, and analyze the effects of model size, batch size, quantization, and deployment configurations ranging from single- and multi-GPU setups to dataflow-accelerator deployments. This comprehensive analysis reveals trade-offs among these factors, performance, and energy efficiency, offering deeper insight into how models and hardware interact under realistic inference workloads.

TABLE I: Hardware specifications of the GPUs and dataflow accelerators, along with the software stack, used in this study.

| Name | Cores | Memory | | Power (TDP) | Count | Software | | |
|---|---|---|---|---|---|---|---|---|
| | | Cache | Off-chip | | | Driver | Torch | vLLM |
| **NVIDIA A100** | 108 SMs | L1: 164 KB<br>L2: 40 MB | HBM2e: 80 GB | 400 W / GPU | 8 / Node | CUDA 12.6 | 2.7.0 | 0.9.0 |
| **NVIDIA H100** | 132 SMs | L1: 256 KB<br>L2: 50 MB | HBM3: 80 GB | 700 W / GPU | 4 / Node | CUDA 12.6 | 2.7.0 | 0.9.0 |
| **NVIDIA GH200** | 132 SMs | L1: 256 KB<br>L2: 50 MB | HBM3: 96 GB | 450 W / GPU | 1 / Node | CUDA 12.9 | 2.7.0 | 0.9.0 |
| **AMD MI250** | 220 CUs | L1: 16 KB<br>L2: 8 MB | HBM2e: 64 GB | 560 W / GPU | 4 / Node | ROCm 5.3 | 2.7.0 | 0.9.1 |
| **AMD MI300X** | 304 CUs | L1: 32 KB<br>L2: 4 MB<br>L3: 256 MB | HBM3: 192 GB | 750 W / GPU | 8 / Node | ROCm 5.3 | 2.7.0 | 0.9.1 |
| **Intel MAX 1550** | 128 Xe-cores | L1: 64 KB<br>L2: 408 MB | HBM2e: 128 GB | 600 W / GPU | 6 / Node | oneAPI<br>2025.2.0 | 2.8.0 | 0.10.1 |
| **SambaNova SN40L** | 1040 PCUs<br>1040 PMUs | 512 KB<br>520 MB / chip | HBM: 64 GB<br>DDR: 786 GB | 10 kW / Node | 16 / Node | – | – | – |
| **Cerebras CS3** | 900k cores | 512 KB<br>44 GB / chip | N/A | 24 kW / Node | 1 / Node | – | – | – |

## II. Hardware Accelerators

LLM inference is a complex task that typically requires dedicated hardware acceleration because of its high computational demands and the need for large, high-speed memory. GPUs remain the most widely adopted solution, but dedicated dataflow accelerators have recently emerged as a promising alternative [18].

In this study, we investigate 6 datacenter-level GPUs and 3 dataflow AI accelerators. Table I summarizes the architectural characteristics of the AI accelerators under analysis, along with details of the software stack used with each machine.

### *A. GPUs*

Our GPU analysis comprises six modern GPUs released between 2020 and 2024 by Intel, NVIDIA, and AMD. All the GPU nodes used in our experiments are hosted at the Argonne Leadership Computing Facility [19].

NVIDIA Ampere A100 [20] is the most widely used datacenter GPU for AI workloads. In its DGX configuration, it provides 80GB of HBM2e memory, dedicated Tensor Cores for mixed-precision matrix operations, and operates at a 400W TDP. NVIDIA Hopper H100 [21], builds on this design with enhanced Tensor Cores supporting FP8 precision, a redesigned memory hierarchy featuring 80GB of HBM3, and a higher 700W TDP. The NVIDIA Grace-Hopper GH200 [22] further integrates compute and memory by coupling a Grace CPU with an H100 GPU through a cache-coherent link, enabling unified access to 96GB of HBM3 and 512GB of DRAM.

Alongside NVIDIA GPUs, we also use datacenter GPUs from other major vendors. The AMD Instinct MI250 combines two Graphics Compute Dies (GCDs) with 128GB of HBM2e memory and a 560W TDP, while the AMD MI300X integrates eight XCDs into a single GPU, adds FP8 support, improves the memory hierarchy, and scales up to 192GB of HBM3 at 700W. Similarly, the Intel Data Center Max 1550 [23] features two GPU tiles, each with 64 Xe cores, Intel Xe Matrix Extensions, and 64GB of HBM2e memory, for a combined 600W TDP.

### *B. Dataflow Accelerators*

In our study, we use two dataflow hardware accelerators from SambaNova and Cerebras.

The DataScale SN40L [24] is a dataflow accelerator based on SambaNova's Reconfigurable Dataflow Architecture. It features 16 Reconfigurable Dataflow Units (RDUs), composed of Pattern Compute Units (PCUs) and Pattern Memory Units (PMUs), that manage the distribution of on-chip memory across the on-chip network. In total, each RDU features 520 MB of on-chip SRAM and can access 64GB of HBM and 768GB of DDR off-chip memory. The power consumption for inference workloads is estimated at around 10kW.

Cerebras CS3 hosts the third generation of Cerebras's Wafer Scale Engine (WSE-3) featuring 900.000 cores equipped with 8-wide FP16 SIMD units [25]. The WSE's memory is fully distributed to maximize throughput via data locality: each core has a dedicated SRAM scratchpad, totaling 44GB per chip. All data sharing is performed through the on-chip network via the fabric router, allowing the streaming of weights among cores and between cores and the MemoryX unit (off-chip memory).

## III. Methodology

In this section, we discuss the methodology used to characterize LLM inference across AI accelerators, including the LLM models used, the inference dataset, the experimental setup, and the evaluation metrics.

### *A. LLMs*

Large Language Models (LLMs) are commonly based on the decoder-only transformer architecture, which comprises a stack of layers containing multiple attention heads [26], [27]

TABLE II: LLMs Used in This Study: Fourteen Models and Their Features.

| Model Name | Parameters Total | Parameters Active | Layers | Attention | FFN Type | Context |
|---|---|---|---|---|---|---|
| **Llama 3.1 8B** | 8B | = | 32 | GQA (32/8) | Dense | 128K |
| **Llama 3.3 70B** | 70B | = | 80 | GQA (64/8) | Dense | 128K |
| **Nemotron Super 49B** | 49B | = | 80 | GQA (64/8) | Dense | 128K |
| **Llama 4 Scout 17B 16E** | 109B | 17B | 48 | GQA (40/8) | MoE (16E) | 10M |
| **Qwen3 8B** | 8.2B | = | 36 | GQA (32/8) | Dense | 128K |
| **Qwen3 14B** | 14.8B | = | 40 | GQA (40/8) | Dense | 128K |
| **Qwen3 32B** | 32.8B | = | 64 | GQA (64/8) | Dense | 128K |
| **Qwen2.5 72B** | 72B | = | 80 | GQA (64/8) | Dense | 128K |
| **Qwen3 235B A22B** | 235B | 22B | 94 | GQA (64/4) | MoE (128E) | 128K |
| **Ministral 8B** | 8B | = | 36 | GQA (32/8) | Dense | 128K |
| **Mistral Small 3.1 24B** | 24B | = | 40 | GQA (32/8) | Dense | 128K |
| **Mixtral 8×7B** | 47B | 13B | 32 | GQA (32/8) | MoE (8E) | 32K |
| **Mixtral 8×22B** | 141B | 39B | 56 | GQA (32/8) | MoE (8E) | 32K |
| **DeepSeeK R1** | 671B | 37B | 61 | MLA | MoE (257E) | 23K |

and feed-forward networks. Although this structure is widely shared, LLMs differ substantially in the number of layers and attention heads, leading to a wide range of model sizes—from hundreds of millions to trillions of parameters. Another key distinction arises between dense models and sparse Mixture-of-Experts (MoE) architectures, where the latter replace the standard feed-forward layers with sets of selectively activated expert networks [28].

In our analysis, we concentrate on the model size $N$ and sparsity $s$, as both directly determine inference-time computational cost. For every forward pass, the memory traffic is

$$\mathrm{Mem} = P\,(N + B\, n_{\mathrm{layer}}\, n_{\mathrm{ctx}}\, d_{\mathrm{model}}) \;\text{ bytes} \tag{1}$$

while the number of floating-point operations (FLOPs) is

$$\mathrm{FLOPs} = 2B\,(N + n_{\mathrm{layer}}\, n_{\mathrm{ctx}}\, d_{\mathrm{model}}) \tag{2}$$

Here, $P$ denotes the floating-point precision in bytes, $n_{\mathrm{layer}}$ is the number of layers, $n_{\mathrm{ctx}}$ is the average context length per batch, $d_{\mathrm{model}}$ is the attention head output dimension, and $B$ is the batch size. Both memory and the number of operations are constituted of two portions, a constant factor related to model size $N$ and a dynamic portion related to the context length $n_{\mathrm{ctx}}$. As discussed in the literature, the constant portion is dominant, allowing one to ignore the dynamic portion for simplicity [29]. Under this simplification, we can linearly relate operational intensity to batch size $OI = BP/2$.

In contrast, sparse MoE models reduce the number of FLOPs by a factor of $(1 - s)$. However, the memory required to load expert parameters depends on the degree of overlap among active experts across sequences in a batch; the effective memory reduction can range from $(1 - s)$ to 0, depending on how much expert reuse occurs.

Moreover, the model size $N$ affects the available memory for the KV cache and could necessitate multi-GPU inference, adding new dimensions to the analysis.

To account for these factors, we evaluate *fourteen open-source large language models* from the Llama [30], [31], Qwen [32], [33], Mistral [34], and DeepSeek R1 [35] families, ranging from 8B to 670B parameters and covering both dense and MoE architectures. Table II summarizes their main characteristics.

## *B. Dataset*

Prompt and response lengths strongly affect generation latency in large language models. Each generated token requires a forward pass and, as discussed in III-A, the context length $n_{\mathrm{ctx}}$ influences both computation and memory usage.

To simulate a realistic chatbot scenario, we employ the MLSysChat1M dataset [36], a collection of one million real-world, multi-turn conversations with chatbots. From this dataset, we selected the first 1,000 conversations for our analysis, filtering to include only those with a total character count between 64 and 20,000. This range excludes trivial or empty samples while staying within the models' maximum context length, ensuring that the resulting distribution of prompt and answer lengths remains plausible.

Conversations follow the standard OpenAI chat format with "user," "system," and "assistant" roles. For evaluation, we remove the final assistant reply and ask the model to regenerate it. To control output length, we tokenize the ground-truth response and set both the minimum and maximum generation length to its exact token count, forcing outputs of identical length.

## *C. Experimental Configurations*

*1) Single-GPU Analysis:* We examine single-GPU inference performance across different GPU architectures, including three NVIDIA GPUs (A100, H100, and GH200), two AMD GPUs (MI250 and MI300X), and one Intel GPU (Max 1550). The distinctive features of these accelerators are summarized in Table I.

To achieve a uniform software stack across a diverse set of machines, we leveraged vLLM [37], a high-performance, open-source LLM inference framework compatible with NVIDIA, AMD, and Intel GPUs. vLLM offers some key optimizations at the level of kernels [37], [38], and at the level of scheduling [39] that make it substantially faster and more energy efficient than the base Transformers implementation [7]. Table I reports the specific software stack used on each machine, including the driver, PyTorch, and vLLM versions.

For each experiment, we launch a vLLM instance, specifying the model to serve, numeric precision, and the maximum batch size. We then process the entire inference dataset as a batch workload, recording key performance and power metrics for each generated sequence. While vLLM runs, a parallel process polls the vendor system monitoring interfaces, i.e., *py3nvml* for NVIDIA, *amdsmi* for AMD, and *xpu-smi* for Intel, to capture power consumption, memory usage, and GPU utilization. For NVIDIA and AMD devices, we sample every 0.5 seconds, whereas for Intel devices, we average 1.9 seconds due to the longer response time of *xpu-smi*.

We evaluate four LLMs—Llama 3.1 8B, Qwen3 14B, Mistral Small 3.1 24B, and Qwen3 32B—on each GPU, with batch sizes ranging from 16 to 128. We also test multiple quantization settings. For 16-bit precision, we use the *bfloat16* format, except on Intel, where *FP16* is better optimized. For 8-bit experiments, we use *W8A8* on FP8-native GPUs and *W8A16* on the NVIDIA A100.

*2) Multi-GPU Analysis:* We further evaluate LLM inference by scaling it across three multi-GPU nodes featuring NVIDIA A100, NVIDIA H100, and AMD MI300X GPUs (see Table I). In this less memory-constrained setup, we evaluate all 14 LLM models listed in Table II, ranging from 8B to 670B parameters, using both *bfloat16* and *FP8* quantizations.

Multi-GPU inference can be achieved through various parallelization techniques, introducing a further dimension to the analysis: *Data parallelism (DP)* replicates the LLM instance on multiple GPUs, improving performance and requiring no inter-GPU communication; however, it cannot increase the memory capacity. *Tensor parallelism (TP)* insteads shards model weights and attention heads across GPUs, allowing larger models to fit in memory and keeping all GPUs active, at the cost of frequent all-gather operations. Optimal node-level performance on an $N$-GPU node is therefore achieved by partitioning the GPUs across $D$ data-parallel instances, with Tensor Parallel size $T = N/D$.

In our study, we investigate different TP/DP configurations to empirically identify near-optimal settings. In particular, we compare performance for $T \in \{1, 2, 4\}$ in a weak scaling regime, tasking each $DP$ instance to process the whole dataset. To maintain a constant sequences-per-GPU ratio, we set the batch size to $T \times 128$.

We further investigate parallelization strategies for Mixture-of-Experts (MoE) models by comparing Tensor Parallelism with vLLM's Expert Parallelism (EP), which distributes experts across multiple GPUs, while replicating attention heads.

Specifically, we evaluate a single vLLM instance ($DP = 1$) using 4 GPUs with either Tensor Parallelism ($TP = 4$) or Expert Parallelism ($EP = 4$). We measure both performance and energy consumption across four large-scale MoE models—Llama4 Scout, Mixtral 8×22B, Qwen 235B A22B, and DeepSeek R1—using FP8 precision on NVIDIA H100 and AMD MI300X GPUs.

*3) Dataflow Accelerator Analysis:* The final part of our analysis examines the performance and energy efficiency of dataflow accelerators, specifically the Cerebras CS3 and the SambaNova SN40L (see Table I).

We evaluate the Cerebras CS3 using an early-prototype inference engine accessible via an OpenAI-style API. We run Llama 3.1 8B—fully hosted in on-chip memory for optimal performance—across batch sizes from 1 to 8. For each batch size, we process the inference dataset by issuing one request per conversation and evenly distributing requests across a number of clients equal to the batch size.

For the SambaNova SN40L, we use the SambaStudio stack to create an inference engine and evaluate batch sizes ranging from 1 to 32. We evaluate both Llama 3.1 8B and Llama 3.3 70B, creating a number of clients to match the engine's effective batch size. For the energy efficiency analysis, in the absence of more granular measurements, we use PDU measurements for the entire rack.

### D. Performance Metrics

To provide a comprehensive evaluation of LLM inference, we consider both performance and energy efficiency using well-established metrics:

- **Latency**: The total time required to generate an output sequence. Latency is a primary quality-of-service (QoS) metric, as it reflects the waiting time experienced by end users. A related metric is **inter-token latency (ITL)**, which is calculated as the average time between the generation of two subsequent tokens.
- **Time To First Token (TTFT)**: The time taken to produce the first output token. Another important QoS metric.
- **Throughput**: The number of tokens processed per second (tokens/s). It is computed as the total number of tokens divided by the overall latency. This is a key metric in a batch-inference scenario, where the goal is to process multiple requests as quickly as possible. Following prior work [15], we consider both input and output tokens.
- **Tokens per Joule**: the total number of input and output tokens processed per joule (tokens/J). Energy consumption is measured as the integral of the power over time.

## IV. Experimental Results

### A. Single-GPU Results

In this section, we discuss the performance of LLM inference on a single-GPU, comparing six different GPUs from Nvidia, AMD and Intel.

Figure 1 shows the change in performance and in energy efficiency as the model size increases. For the dense models considered, we can observe a clear trend:

Latency and TTFT increase linearly with model size, while throughput and energy efficiency decrease inversely.

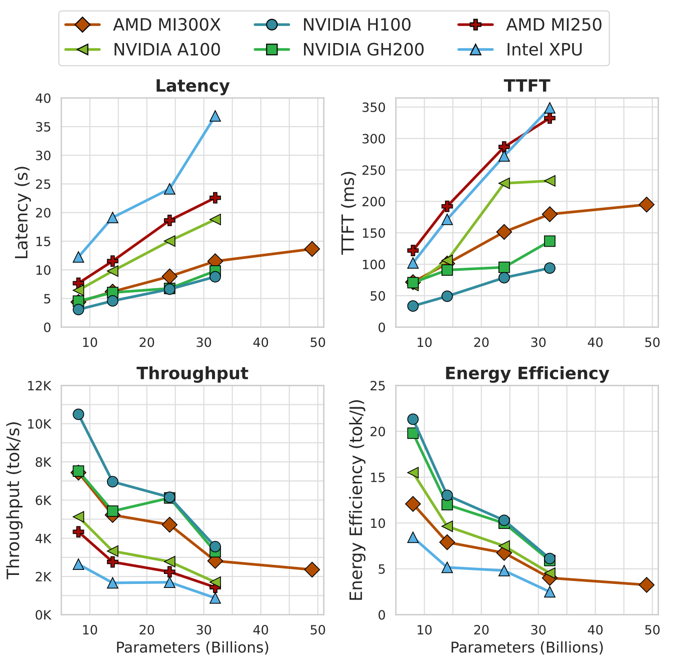


Fig. 1: Performance and energy efficiency across six GPUs for models of varying size (Llama3 8B, Qwen3 14B and 32B, Mistral Small 24B, and Nemotron Super 49B). Multi-Tile GPUs (AMD MI250 and Intel Max 1550) use tensor parallelism to fully utilize their resources. All experiments used a batch size of 64.

This trend is consistent with the observation that the dominant context-independent component of both FLOPs and memory read volume scales linearly with the model size $N$. In Section IV-B, we further examine this behavior in a multi-GPU setting.

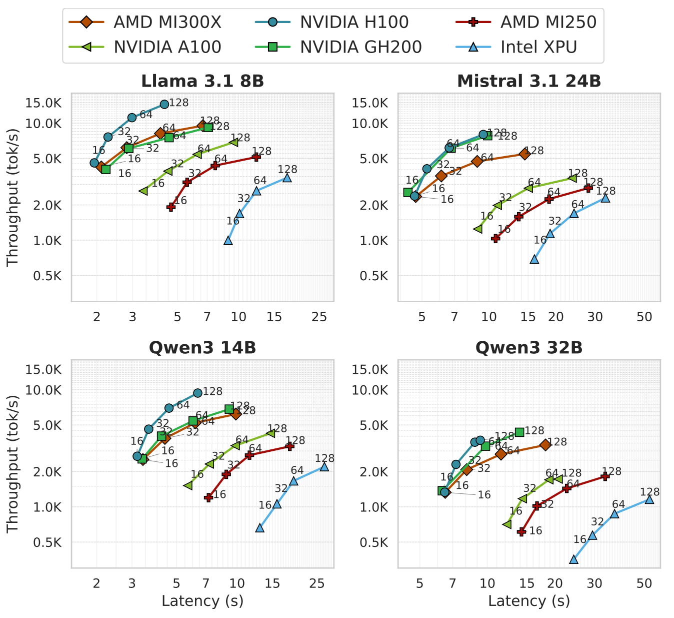


Fig. 2: Trade-off between throughput and latency across batch sizes (16-128), using four LLM models and six different GPUs. The numbers in the plots indicate the batch sizes.

Figure 2 shows the impact of batch size on throughput and latency across four LLM models. We can observe that consistently across GPUs, smaller batches achieve lower latency, while larger batch sizes achieve better throughput. Recalling the FLOP count and memory formulas discussed in III-A, we know that small batch sizes require fewer operations per forward pass, while large batch sizes have better operational intensity, as the loading of model weights is shared by all the sequences in the batch. So, reduced batch sizes lower the latency and TTFT of individual sequences; however, the increased operational intensity achieved with larger batch sizes is beneficial for throughput and energy efficiency.

Larger batch sizes improve the performance in an offline setting while achieving better energy efficiency. In contrast, metrics related to online serving, such as TTFT or individual sequence latency, improve as the batch size decreases.

We make some interesting observations across different GPU architectures. NVIDIA A100 remains the most widely adopted AI GPU, serving as a natural baseline for comparison. NVIDIA H100, NVIDIA GH200, and AMD MI300x all surpass it, delivering superior performance across models and batch sizes in *bfloat16* and *FP8* precision. Among the evaluated GPUs, the NVIDIA H100 stands out as both the most powerful and the most energy-efficient. Although it consumes more power (478 W compared to 337 W), the H100 delivers substantially higher performance and greater efficiency than the NVIDIA A100. The AMD MI300x also consistently surpasses the A100 in raw performance; however, its overall energy efficiency is lower, primarily due to its higher power consumption (625 W).

AMD MI250 and Intel Max 1550 underperform relative to the NVIDIA A100 despite their higher power consumption. This gap is partly due to their multi-tile designs, which require tensor parallelism to fully utilize the hardware, as each tile is presented to the operating system as a separate device. Additionally, the AMD and Intel implementations of vLLM currently lag in software maturity, lacking support for key optimizations. In terms of energy efficiency, the Intel Max 1550 is less efficient than the A100, while no data is available for the AMD MI250.

Memory capacity is another critical factor. NVIDIA GPUs rank lowest in VRAM, which limits the size of models they can host compared to the other GPUs under analysis. The AMD MI300x provides the largest memory capacity at 192GB. It is also important to note that the current vLLM implementation does not utilize all the unified memory of the GH200 SoC, limiting VRAM capacity to 96 GB, resulting in results comparable to the base NVIDIA H100.

NVIDIA Hopper GPUs emerge as the most performant and energy-efficient GPU among those tested. The AMD MI300X also delivers competitive performance, outpacing the NVIDIA A100, but not in energy efficiency. Notably, this architecture benefits from the largest VRAM among all the devices tested. Finally, while Intel Max 1550 and AMD MI250 offer lower performance and energy efficiency than NVIDIA A100.

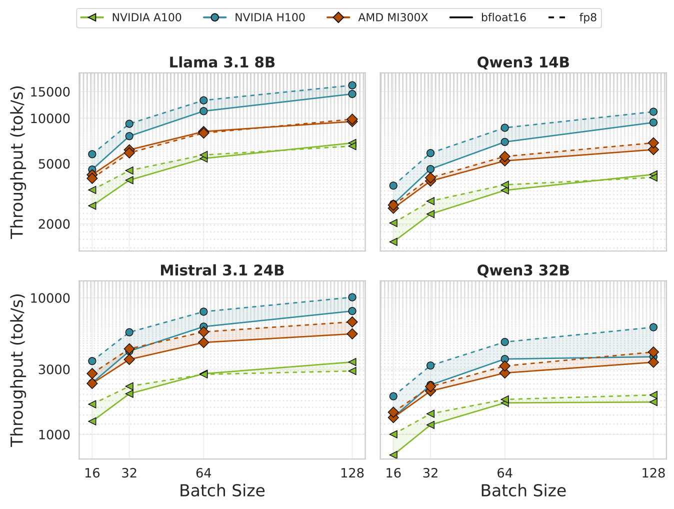


Fig. 3: Throughput comparison of FP8 (dashed) and bfloat16 (solid) precision across three GPU architectures and four LLM models.

Finally, we consider the effect of FP8 quantization on performance and energy efficiency. Figure 3 compares the throughput obtained through FP8 quantization with that of bfloat16 precision across different GPUs, LLM models, and batch sizes. As discussed in III-C1, NVIDIA H100, GH200, and AMD MI300x support W8A8 quantization, while NVIDIA A100 only supports W8A16.

As expected, W8A8 quantization consistently improves both performance and efficiency over bfloat16. In an ideal scenario, latency would halve: memory traffic is reduced by 2× and peak FP8 throughput is 2× that of bfloat16. In practice, however, factors such as kernel launch overhead and other runtime inefficiencies limit the achievable speedup.

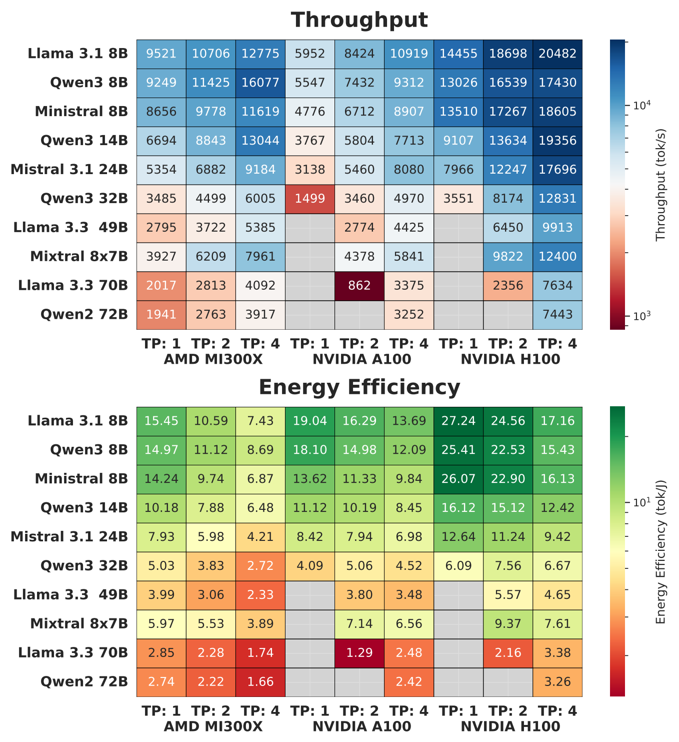
From our experiments, it emerges that on NVIDIA Hopper GPUs, W8A8 delivers substantial gains, with throughput increasing by 20–38% and energy efficiency by 34–46%. Improvements on the AMD MI300X are smaller and more model-dependent: throughput shows no gain on Llama 8B but reaches up to 20% on Mistral Small 24B. Energy-efficiency gains, while still below Hopper's, are more pronounced than throughput improvements, ranging from 16–35%.

In contrast, W8A16 reduces only the memory traffic associated with model weights and does not change the throughput in terms of $FLOPs$. As a result, performance gains depend on batch size: they are largest at small batch sizes, where execution is more memory-bound, and diminish as batch size increases. The same trend is observed for energy efficiency.

We do not explicitly measure the accuracy loss due to quantization. However, previous studies [40] show that FP8-quantized models retain over 99% of the accuracy of their bfloat16 counterparts on standard benchmarks, concluding that the model quality degradation is negligible.

W8A8 quantization delivers significant performance and energy efficiency gains while incurring only minimal accuracy loss. NVIDIA Hopper GPUs gain the most from FP8 quantization (up to 44% throughput and 47% energy efficiency), while AMD MI300X offers smaller improvements, mainly at larger scales. On NVIDIA A100, W8A16 quantization is beneficial only at smaller batch sizes.

## B. Multi-GPU Results



Throughput

| | TP: 1 AMD MI300X | TP: 2 AMD MI300X | TP: 4 AMD MI300X | TP: 1 NVIDIA A100 | TP: 2 NVIDIA A100 | TP: 4 NVIDIA A100 | TP: 1 NVIDIA H100 | TP: 2 NVIDIA H100 | TP: 4 NVIDIA H100 |
|---|---|---|---|---|---|---|---|---|---|
| Llama 3.1 8B | 9521 | 10706 | 12775 | 5952 | 8424 | 10919 | 14455 | 18698 | 20482 |
| Qwen3 8B | 9249 | 11425 | 16077 | 5547 | 7432 | 9312 | 13026 | 16539 | 17430 |
| Ministral 8B | 8656 | 9778 | 11619 | 4776 | 6712 | 8907 | 13510 | 17267 | 18605 |
| Qwen3 14B | 6694 | 8843 | 13044 | 3767 | 5804 | 7713 | 9107 | 13634 | 19356 |
| Mistral 3.1 24B | 5354 | 6882 | 9184 | 3138 | 5460 | 8080 | 7966 | 12247 | 17696 |
| Qwen3 32B | 3485 | 4499 | 6005 | 1499 | 3460 | 4970 | 3551 | 8174 | 12831 |
| Llama 3.3 49B | 2795 | 3722 | 5385 | | 2774 | 4425 | | 6450 | 9913 |
| Mixtral 8x7B | 3927 | 6209 | 7961 | | 4378 | 5841 | | 9822 | 12400 |
| Llama 3.3 70B | 2017 | 2813 | 4092 | | 862 | 3375 | | 2356 | 7634 |
| Qwen2 72B | 1941 | 2763 | 3917 | | | 3252 | | | 7443 |

Energy Efficiency

| | TP: 1 AMD MI300X | TP: 2 AMD MI300X | TP: 4 AMD MI300X | TP: 1 NVIDIA A100 | TP: 2 NVIDIA A100 | TP: 4 NVIDIA A100 | TP: 1 NVIDIA H100 | TP: 2 NVIDIA H100 | TP: 4 NVIDIA H100 |
|---|---|---|---|---|---|---|---|---|---|
| Llama 3.1 8B | 15.45 | 10.59 | 7.43 | 19.04 | 16.29 | 13.69 | 27.24 | 24.56 | 17.16 |
| Qwen3 8B | 14.97 | 11.12 | 8.69 | 18.10 | 14.98 | 12.09 | 25.41 | 22.53 | 15.43 |
| Ministral 8B | 14.24 | 9.74 | 6.87 | 13.62 | 11.33 | 9.84 | 26.07 | 22.90 | 16.13 |
| Qwen3 14B | 10.18 | 7.88 | 6.48 | 11.12 | 10.19 | 8.45 | 16.12 | 15.12 | 12.42 |
| Mistral 3.1 24B | 7.93 | 5.98 | 4.21 | 8.42 | 7.94 | 6.98 | 12.64 | 11.24 | 9.42 |
| Qwen3 32B | 5.03 | 3.83 | 2.72 | 4.09 | 5.06 | 4.52 | 6.09 | 7.56 | 6.67 |
| Llama 3.3 49B | 3.99 | 3.06 | 2.33 | | 3.80 | 3.48 | | 5.57 | 4.65 |
| Mixtral 8x7B | 5.97 | 5.53 | 3.89 | | 7.14 | 6.56 | | 9.37 | 7.61 |
| Llama 3.3 70B | 2.85 | 2.28 | 1.74 | | 1.29 | 2.48 | | 2.16 | 3.38 |
| Qwen2 72B | 2.74 | 2.22 | 1.66 | | | 2.42 | | | 3.26 |

Fig. 4: Throughput and energy efficiency heatmap across three multi-GPU nodes in bfloat16 precision. For each node, results are reported for tensor parallel sizes $T = 1, 2, 4$. The batch size is set to $128 \times T$.

We now examine the scaling behavior of LLM inference across multiple GPUs. Figure 4 presents the throughput and energy efficiency of ten LLMs, ordered by parameter count. For each node, results are reported for Tensor Parallel sizes $T = 1, 2, 4$, with Data Parallelism $D$ scaled to fully utilize all available GPUs. In Figure 4, throughput values are normalized by $D$.

The figure highlights a key trade-off in multi-GPU inference: balancing memory capacity and computational efficiency when choosing between Tensor and Data p-Parallelism. As discussed in Section III-C, Tensor Parallelism partitions model weights across GPUs, enabling larger models to run and potentially improving throughput. However, it introduces frequent all-gather operations, resulting in synchronization overhead and idle GPU time.

Configurations with tensor parallel size 2 achieve throughput improvements from 12% to 34% compared to single-GPU baselines, while tensor parallel size 4 yields gains of 32% to 140%. These results correspond to scaling efficiencies of at most 67% for 2 GPUs and 60% for 4 GPUs. In contrast, data parallelism consistently achieves close to linear scaling, with $N$ GPUs providing approximately $N$ times the performance of a single GPU, and thus outperforming tensor parallel configurations. Consistently, data-parallel configurations also achieve superior energy efficiency, despite tensor-parallel configurations, on average, drawing less power per GPU.

From this observation, we establish the following guideline:

In offline inference scenarios, maximizing the number of data-parallel instances generally provides the highest throughput and energy efficiency, rather than increasing the tensor parallel size.

However, there are important exceptions in which tensor parallelism outperforms data parallelism, occasionally exceeding 100% efficiency. This occurs when VRAM limitations in data-parallel configurations force costly recomputation, as memory is insufficient to store both the model weights and the full key–value (KV) cache. For instance, Qwen3 32B achieves better scaling efficiency with tensor parallel size 2 on both NVIDIA A100 and H100 nodes. Similarly, the large memory demands of Llama 3.3 70B make a tensor parallel size of 4 preferable to a size of 2 on the same devices.

In those scenarios, the performance increase is also reflected by superior energy efficiency.

From these observations, we propose a second guideline:

When model weights exceed 80% of GPU memory capacity, performance and energy efficiency degrade due to insufficient space for the KV cache. To mitigate this issue, it is advisable to either quantize the model to a lower precision or increase the tensor parallel size.

Once these general principles for multi-GPU performance are established, we can compare performance across different hardware nodes. Consistent with Section IV-A, the NVIDIA H100 delivers the highest throughput and energy efficiency at bfloat16 and FP8 precisions. The AMD MI300X follows, outperforming the NVIDIA A100 in most scenarios. In FP8 precision, the MI300X is the second most energy-efficient device, while in bfloat16 precision, the A100 surpasses it for models smaller than 32B parameters and for Mixtral 8x7B.

However, when performance is compared under a fixed GPU budget (e.g., 4 GPUs), the AMD MI300X often outperforms due to its larger memory capacity. This enables it to operate in pure data-parallel mode, thereby avoiding the communication overheads typically required by NVIDIA cards at high tensor parallel sizes. For example, with Llama 3.3 70B, the MI300X achieves around 2017 tokens/s with a tensor parallel size of 1, while the NVIDIA H100 achieves strong performance (7634 tokens/s) with a tensor parallel size of 4. In a 4-GPU setup, the MI300X reaches 8068 tokens/s with pure data parallelism, surpassing the H100. A similar trend is observed for Qwen2 72B in bfloat16 precision. This leads to our final takeaway:

A larger memory capacity allows for a reduction in the tensor parallel size required to host a model. This provides an indirect benefit to performance, as data parallelism has better scaling efficiency.

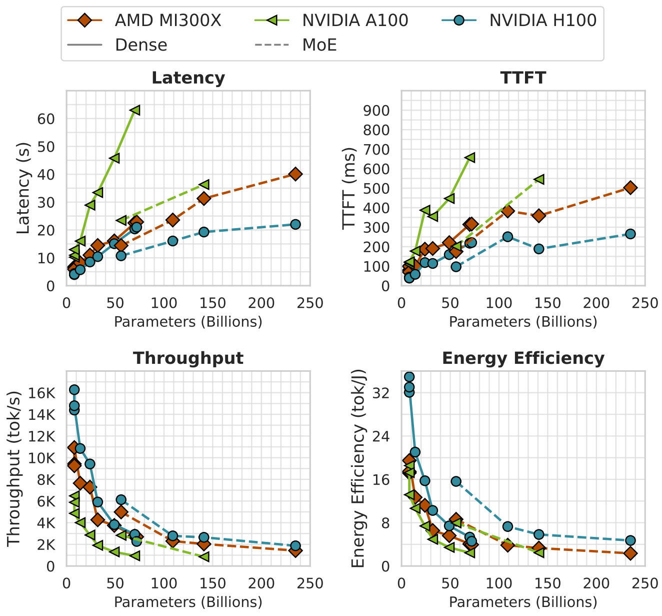


Fig. 5: Key performance and energy efficiency metrics across various LLMs of different sizes. All the results refer to the best achievable across parallel configurations in FP8 precision. Throughput is divided by GPU count to allow for comparison among nodes.

The second aspect of our analysis investigates how energy efficiency and performance evolve with increasing model size and sparsity. Figure 5 presents the trends of four key metrics as the number of parameters increases. The trends across dense models are shown with solid lines, while dashed lines represent the Mixture-of-Experts (MoE) model.

We notice that in dense models, latency and TTFT increase linearly with the number of parameters, while throughput and energy efficiency exhibit an inverse correlation with model scale, consistently with the growth in floating-point operations per forward pass. In a multi-GPU scenario, we also need to consider that larger models typically require distribution across multiple GPUs, which introduces the communication overhead of tensor parallelism, reducing the throughput and energy efficiency even more pronouncedly and reinforcing the relative advantage of smaller models.

We extend our analysis to include sparse MoE models. As discussed in III-A, sparsity implies a linear reduction of the number of FLOPs and a reduction in the memory volume to read that depends on expert overlap across sequences.

Figure 5 clearly shows that MoE models (dashed line) achieve better performance and energy efficiency than dense models (full line) of equal size. For example, Llama 4 Scout, despite its 109B total parameters, delivers 17% higher throughput and 52% better energy efficiency than the smaller Llama 3 70B on NVIDIA H100. However, the benefits are not proportional to the degree of sparsity. On both the NVIDIA H100 and AMD MI300x, Llama 4 Scout underperforms compared to the 49B Llama 3 Nemotron Super, despite using only 22B active parameters.

MoE models achieve improved computational efficiency and performance relative to the dense counterparts, while achieving similar or better accuracy on established evaluation benchmarks. However, the improvement is substantially lower than $1/s$, where $s$ is the sparsity.

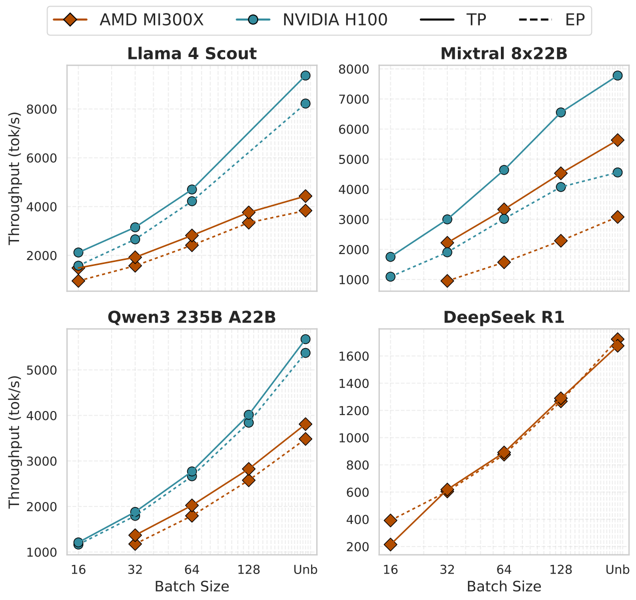


Fig. 6: Throughput of four Mixture-of-Expert models in two multi-GPU nodes using 4-way Tensor and Expert Parallelism.

In Section III-C, we introduced Expert Parallelism as an alternative strategy for parallelizing Mixture-of-Experts (MoE) inference. Figure 6 compares Expert and Tensor Parallelism across four MoE models on 4 NVIDIA H100 and 4 AMD MI300x GPUs in FP8 precision over multiple batch sizes. Overall, Tensor Parallelism achieves higher throughput and energy efficiency across most models and platforms, with the performance gap largely driven by the number of experts. Qwen3 (128 experts) shows only modest gains under Tensor Parallelism (16% on MI300x and 4% on H100 for throughput, with 18% and 2% efficiency improvements), whereas Llama4 Scout (16 experts) benefits more substantially, reaching up to 6% higher throughput on MI300x and 34% on H100. Mixtral 8x22B (8 experts) exhibits the largest improvements, with Tensor Parallelism outperforming Expert Parallelism by 61% on H100 and up to 2.7× on MI300x in throughput, alongside roughly two-fold efficiency gains on MI300x and 60% on H100. DeepSeek R1 deviates from this trend due to its architecture with 256 rooted experts plus one shared expert: Expert Parallelism achieves competitive or superior throughput and efficiency at batch sizes of 16 or smaller, while Tensor Parallelism becomes superior at larger batch sizes.

Tensor Parallelism outperforms Expert Parallelism in both performance and energy efficiency across models and hardware platforms, with DeepSeek R1 being the sole exception. The performance gap narrows as the number of experts increases.

## C. Dataflow Accelerator Results

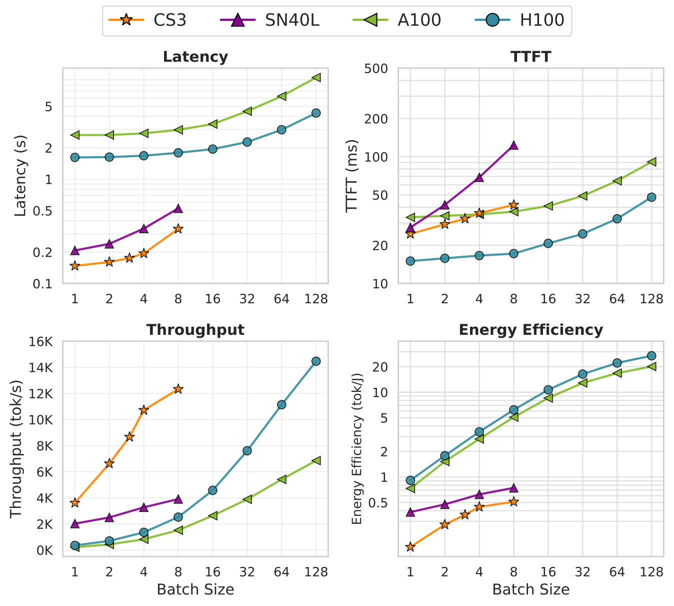


Fig. 7: Performance and energy efficiency of Llama 3.1 8B inference across AI accelerators, measured at several batch sizes (1-128).

Next, we analyze LLM inference efficiency on two state-of-the-art dataflow AI accelerators (Cerebras CS3 and Sambanova SN40L), comparing their results with NVIDIA A100 and NVIDIA H100 GPUs. Figure 7 displays how the LLM inference performance and the energy efficiency change across batch sizes on GPUs and dataflow accelerators. All the experiments are done using Llama 3.1 8B with bfloat16 precision.

As discussed in Section IV-A, increasing the batch size improves throughput but also raises TTFT and latency. A similar trend appears on dataflow machines, though with notable distinctions: (1) Cerebras and SambaNova provide precompiled artifacts supporting batch sizes between 1 and 8, while vLLM supports dynamic batch size. (2) Their performance degrades more sharply as batch size increases compared to GPUs.

Within their supported batch sizes, dataflow machines achieve markedly lower latency and substantially higher throughput than GPUs. The most pronounced improvement occurs at batch size 1, where the NVIDIA A100 reaches 215 tokens/s, compared to 3609 tokens/s (16.8×) on the Cerebras CS3 and 2024 tokens/s (9.41×) on the SambaNova SN40L. SamabaNova SN40L also outpaces GPUs with Llama 3.3 70B, achieving 353tokens/s at batch size 1, compared to 88 tokens/s (4.01×) obtained with 4 NVIDIA A100s working with tensor parallelism. From this, we derive our first key takeaway:

Dataflow accelerators deliver up to an order-of-magnitude throughput improvement relative to GPUs at small batch sizes (1 – 8), making them particularly effective for online scenarios with limited user requests per node.

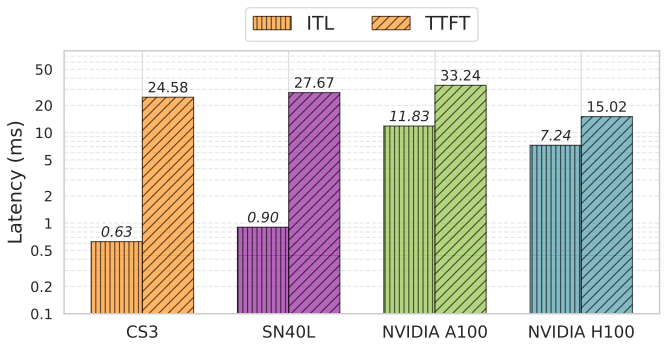


Fig. 8: Time-to-first-token (TTFT) and Inter Token Latency (ITL) of GPUs and dataflow accelerators for Llama 3.1 8B at batch size 1.

Observing the TTFT subplot in Figure 7, we note that although the SambaNova SN40L and Cerebras CS3 achieve substantially lower latency, a similar improvement is not reflected in the TTFT. Figure 8 further illustrates this trend by showing the TTFT—representing the latency of the prefill phase—and the ITL—representing the latency of the decode phase—for Llama 3.1 8B. We observe that the Cerebras CS3 achieves an 18.8× lower ITL compared to the NVIDIA A100, while SambaNova shows a 13.1× improvement. However, the improvement in TTFT is only 35% for Cerebras and 20% for SambaNova. A similar pattern appears when comparing Llama 3.3 70B performance: the SambaNova SN40L achieves slower TTFT compared to 4 NVIDIA A100 (99 $ms$ compared to 91 $ms$), while having a 5.18× shorter ITL (5.6 $ms$ compared to 29 $ms$). From these observations, we can draw an important takeaway:

The performance gains of dataflow machines primarily stem from their advantage in the memory-bound decode phase, rather than the compute-bound prefill phase, highlighting the effectiveness of spatial architectures in alleviating memory bottlenecks.

Another observation regards peak throughput. Dataflow machines provide pre-compiled artifacts that allow running a model at a fixed batch size. Both Cerebras and SambaNova provide images for batch sizes up to 8, which are ideal for online inference scenarios. vLLM, instead, handles batch sizes dynamically and allows us to test batch sizes up to 128. In an offline scenario, where the maximum achievable throughput using the optimal batch size is the only relevant performance metric, we observe that the capacity of GPUs to scale to larger batch sizes allows them to achieve higher peak throughput. Cerebras at a batch size of 8 produces a competitive throughput (12327 tokens/s) that outperforms both the NVIDIA A100 and the AMD MI300x, but is still lower than the NVIDIA H100 at batch size 128. The SambaNova SN40L instead peaks at 3907 tokens/s.

In batch offline inference, GPUs benefit from larger batch sizes, narrowing the gap with dataflow accelerators. The Cerebras CS3 is still able to provide competitive offline performance, running at batch size 8.

Finally, we analyze energy efficiency. Even at small batch sizes—where the performance gain is largest—this improvement does not translate into better energy efficiency, because dataflow systems consume more power.

The Cerebras CS-3 system consumes about 24,200 W and achieves 0.15 tokens/J, while the SambaNova SN40L consumes on average 5,230 W and achieves 0.38 tokens/J. For comparison, the NVIDIA A100 achieves 0.73 tokens/s and the NVIDIA H100 achieves 0.91 tokens/s. With Llama 3.3 70B, the SambaNova SN40L significantly narrows the gap with the NVIDIA A100, reaching 0.067 tokens/J at batch size 1, compared with 0.074 tokens/J obtained using four NVIDIA A100 GPUs.

This analysis is constrained by differences in host configurations and in the available power-measurement tools. To ensure a fair comparison, we use the framework described in III-C for GPUs, RDU-level power measurements for SambaNova, and chassis-level readings for the CS-3 system. It is also worth noting that the idle power consumption of the Cerebras CS-3 is only about 4,500 W lower than during inference, indicating that a large fraction of the system's power is independent of the workload.

The energy efficiency, measured in tokens per Joule, of the Cerebras CS3 and SambaNova SN40L systems is lower than that of GPUs, creating a trade-off between performance and energy efficiency.

## V. Discussion

The results in Section IV highlight the critical role of memory bandwidth and capacity in LLM inference.

Regarding bandwidth, we observe that during the decode phase, GPU inference is largely memory-bound, especially at small batch sizes. As discussed in Section III-A, the operational intensity is approximately equal to the batch size, while GPU compute throughput (in FLOPs) exceeds HBM memory bandwidth by roughly three orders of magnitude. This imbalance primarily limits performance to memory movement rather than computation.

Dataflow accelerators mitigate this limitation by improving data locality through on-chip SRAM, reducing ITL by up to 16.8×. The Cerebras CS-3, which achieves the best small-batch performance, pushes this approach to the extreme by relying entirely on SRAM scratchpads. However, because each core provides only 512 KB, hundreds of thousands of cores are required to store the model, resulting in very high power consumption. The SambaNova SN40L adopts a hybrid design, combining large on-chip memories (512 KB per PMU) with off-chip DRAM and HBM, but its overall system scale still leads to higher power draw than the tested GPU systems.

Memory capacity also plays a major role in GPU-based systems. GPUs with limited memory require multi-GPU deployments to host large models, introducing significant communication overheads that further impact performance.

## VI. Related Work

As large language model (LLM) inference becomes an increasingly significant contributor to datacenter energy consumption, research efforts focusing on the performance and energy efficiency of these workloads have grown. This section surveys key efforts in the related areas.

MLPerf Inference [14] is the primary benchmark for evaluating machine learning inference performance. It encompasses multiple scenarios and performance metrics, while ensuring target accuracy requirements are met. Within the domain of language models, MLPerf Inference benchmarks six models (340M–670B parameters) spanning dense and MoE architectures. Complementing this, MLPerf Power [41] provides guidelines for measuring system-level power consumption during MLPerf evaluations. Although our work shares the broader goal of characterizing energy efficiency, *we do not follow the MLPerf Power guidelines because capturing all required data (e.g., CPU, memory, and network metrics) across different AI accelerators is infeasible.*

LLM-Inference Bench [15] systematically analyzes LLM inference performance across various dimensions, including model size, sparsity, sequence length, batch size, and FP8 quantization, on multiple hardware platforms and frameworks. *Our work is inspired by this study, especially in terms of model and hardware selection, but places greater emphasis on energy efficiency and scaling across different accelerators.*

Several studies assess LLM inference energy efficiency on NVIDIA GPUs. Fernandez et al. [7] evaluate optimizations such as continuous batching, kernel compilation, and PagedAttention. Wilhelm et al. [8] evaluate the efficiency of test-time compute strategies, introducing the Energy-per-Token metric. Wilkins et al. [9] analyze energy and performance across seven LLMs on A100 GPUs, while Dynamo-LLM [10] benchmarks six models on H100s under varied tensor-parallel settings. Other works [11]–[13] explore GPU frequency scaling for performance–efficiency trade-offs. *Unlike these studies, our analysis jointly evaluates performance and energy efficiency across modern GPUs from NVIDIA, AMD, and Intel and three dataflow AI accelerators, providing new insights regarding LLM inference across diverse deployment scenarios.*

Emani et al. [17] analyzed the LLM inference and training performance of five dataflow accelerators and compared it with that of NVIDIA A100 and AMD MI250 GPUs. This study used two LLM models, GPT-2 XL and GenSLM. Ferdaus et al. [42] evaluated BERT's training and inference performance, as well as energy efficiency, across four dataflow accelerators and compared it with that of the NVIDIA A100. *While these studies form the basis of our work, they use older LLMs that are no longer representative of the current state of the art.*

## VII. Conclusion and Future Work

With the emergence of AI accelerators, there is a need to identify the conditions, with respect to LLM inference, for which the different accelerators provide performance gains, with a focus on energy efficiency. It is recognized that power demand is one of several limiting factors when considering the deployment of data centers and the cost to maintain operations. In this study, we presented an extensive empirical study of eight AI accelerators for LLM inference and to extract actionable insights for practical deployment. The study includes six GPUs from three vendors in addition to two dataflow accelerators. Our analysis examined the primary factors that affect inference performance and efficiency, including model size, batch size, quantization, and multi-GPU scaling. The study yielded several key insights into the joint performance–energy-efficiency trade-offs among accelerator types. Overall, our results demonstrate that dataflow accelerators outperform GPUs by an order of magnitude at the low ($\leq$ 8) batch sizes typically encountered in online inference. GPUs instead benefit from larger HBM memories, FP8 functional units, and better software maturity that allows for some key optimizations that benefit them in offline inference scenarios.

Several evolving techniques continue to reshape the inference landscape. In particular, 4-bit quantization, prefill–decode disaggregation, hybrid Mamba–attention models, and multimodal LLMs represent promising avenues for further analysis of performance and efficiency []– [].

## Acknowledgments

This work was supported in part by the U.S. National Science Foundation under Grants OAC-2402901 and CCF-2515009; in part by the U.S. Department of Energy under Contract DE-SC0024271, DE-AC02-06CH11357, the ASCR Microelectronics project BIA, and the DOE ASCR SciDAC RAPIDS project. We acknowledge the Argonne Leadership Computing Facility for the use of AI testbeds under project EE-ECP.

## References

[1] International Energy Agency (IEA), "Energy and ai: Energy demand from ai," https://www.iea.org/reports/energy-and-ai/energy-demand-from-ai, 2025, accessed: 2025-09-16.

[2] D. Patterson, J. Gonzalez, Q. Le, C. Liang, L.-M. Munguia, D. Rothchild, D. So, M. Texier, and J. Dean, "Carbon emissions and large neural network training," *arXiv preprint arXiv:2104.10350*, 2021.

[3] Y. Chen, Y. Xie, L. Song, F. Chen, and T. Tang, "A survey of accelerator architectures for deep neural networks," *Engineering*, vol. 6, no. 3, pp. 264–274, 2020.

[4] N. Koilia and C. Kachris, "Hardware acceleration of llms: A comprehensive survey and comparison," *arXiv preprint arXiv:2409.03384*, 2024.

[5] C. Silvano, D. Ielmini, F. Ferrandi, L. Fiorin, S. Curzel, L. Benini, F. Conti, A. Garofalo, C. Zambelli, E. Calore *et al.*, "A survey on deep learning hardware accelerators for heterogeneous hpc platforms," *ACM Computing Surveys*, vol. 57, no. 11, pp. 1–39, 2025.

[6] Y. Kundu, M. Kaur, T. Wig, K. Kumar, P. Kumari, V. Puri, and M. Arora, "A comparison of the cerebras wafer-scale integration technology with nvidia gpu-based systems for artificial intelligence," *arXiv preprint arXiv:2503.11698*, 2025.

[7] J. Fernandez, C. Na, V. Tiwari, Y. Bisk, S. Luccioni, and E. Strubell, "Energy considerations of large language model inference and efficiency optimizations," *arXiv preprint arXiv:2504.17674*, 2025.

[8] P. Wilhelm, T. Wittkopp, and O. Kao, "Beyond test-time compute strategies: Advocating energy-per-token in llm inference," in *Proceedings of the 5th Workshop on Machine Learning and Systems*, 2025, pp. 208–215.

[9] G. Wilkins, S. Keshav, and R. Mortier, "Offline energy-optimal llm serving: Workload-based energy models for llm inference on heterogeneous systems," *ACM SIGENERGY Energy Informatics Review*, vol. 4, no. 5, pp. 113–119, 2024.

[10] J. Stojkovic, C. Zhang, Í. Goiri, J. Torrellas, and E. Choukse, "Dynamollm: Designing llm inference clusters for performance and energy efficiency," in *2025 IEEE International Symposium on High Performance Computer Architecture (HPCA)*. IEEE, 2025, pp. 1348–1362.

[11] P. J. Maliakel, S. Ilager, and I. Brandic, "Investigating energy efficiency and performance trade-offs in llm inference across tasks and dvfs settings," *arXiv preprint arXiv:2501.08219*, 2025.

[12] A. K. Kakolyris, D. Masouros, S. Xydis, and D. Soudris, "Slo-aware gpu dvfs for energy-efficient llm inference serving," *IEEE Computer Architecture Letters*, vol. 23, no. 2, pp. 150–153, 2024.

[13] J. Stojkovic, E. Choukse, C. Zhang, I. Goiri, and J. Torrellas, "Towards greener llms: Bringing energy-efficiency to the forefront of llm inference," *arXiv preprint arXiv:2403.20306*, 2024.

[14] V. J. Reddi, C. Cheng, D. Kanter, P. Mattson, G. Schmuelling, C.-J. Wu, B. Anderson, M. Breughe, M. Charlebois, W. Chou, R. Chukka, C. Coleman, S. Davis, P. Deng, G. Diamos, J. Duke, D. Fick, J. S. Gardner, I. Hubara, S. Idgunji, T. B. Jablin, J. Jiao, T. S. John, P. Kanwar, D. Lee, J. Liao, A. Lokhmotov, F. Massa, P. Meng, P. Micikevicius, C. Osborne, G. Pekhimenko, A. T. R. Rajan, D. Sequeira, A. Sirasao, F. Sun, H. Tang, M. Thomson, F. Wei, E. Wu, L. Xu, K. Yamada, B. Yu, G. Yuan, A. Zhong, P. Zhang, and Y. Zhou, "Mlperf inference benchmark," 2019.

[15] K. T. Chitty-Venkata, S. Raskar, B. Kale, F. Ferdaus, A. Tanikanti, K. Raffenetti, V. Taylor, M. Emani, V. Vishwanath, and V. Gov, "Llm-inference-bench: Inference benchmarking of large language models on ai accelerators." [Online]. Available: https://github.com/argonne-lcf/LLM-Inference-Bench

[16] M. Emani, S. Foreman, V. Sastry, Z. Xie, S. Raskar, W. Arnold, R. Thakur, V. Vishwanath, M. E. Papka, S. Shanmugavelu *et al.*, "Toward a holistic performance evaluation of large language models across diverse ai accelerators," in *2024 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*. IEEE, 2024, pp. 1–10.

[17] M. Emani, Z. Xie, S. Raskar, V. Sastry, W. Arnold, B. Wilson, R. Thakur, V. Vishwanath, Z. Liu, M. E. Papka, C. O. Bohorquez, R. Weisner, K. Li, Y. Sheng, Y. Du, J. Zhang, A. Tsyplikhin, G. Khaira, J. Fowers, R. Sivakumar, V. Godsoe, A. MacIas, C. Tekur, and M. Boyd, "A comprehensive evaluation of novel ai accelerators for deep learning workloads," in *Proceedings of PMBS 2022: Performance Modeling, Benchmarking and Simulation of High Performance Computer Systems, Held in conjunction with SC 2022: The International Conference for High Performance Computing, Networking, Storage and Analysis*. Institute of Electrical and Electronics Engineers Inc., 2022, pp. 13–25.

[18] C. Kachris, "A survey on hardware accelerators for large language models," *Applied Sciences*, vol. 15, no. 2, p. 586, 2025.

[19] "Argonne leadership computing facility (alcf)," U.S. DOE Office of Science User Facility, this research used resources of the Argonne Leadership Computing Facility, which is a DOE Office of Science User Facility supported under Contract DE-AC02-06CH11357.

[20] J. Choquette, E. Lee, R. Krashinsky, V. Balan, and B. Khailany, "3.2 the a100 datacenter gpu and ampere architecture," in *Digest of Technical Papers - IEEE International Solid-State Circuits Conference*, vol. 64. Institute of Electrical and Electronics Engineers Inc., 2 2021, pp. 48–50.

[21] J. Choquette, "Nvidia hopper h100 gpu: Scaling performance," *IEEE Micro*, vol. 43, pp. 9–17, 5 2023.

[22] G. Schieffer, J. Wahlgren, J. Ren, J. Faj, and I. Peng, "Harnessing integrated cpu-gpu system memory for hpc: A first look into grace hopper," in *ACM International Conference Proceeding Series*. Association for Computing Machinery, 8 2024, pp. 199–209.

[23] W. Gomes, A. Koker, P. Stover, D. Ingerly, S. Siers, S. Venkataraman, C. Pelto, T. Shah, A. Rao, F. O'Mahony *et al.*, "Ponte vecchio: A multi-tile 3d stacked processor for exascale computing," in *2022 IEEE International Solid-State Circuits Conference (ISSCC)*, vol. 65. IEEE, 2022, pp. 42–44.

[24] R. Prabhakar, R. Sivaramakrishnan, D. Gandhi, Y. Du, M. Wang, X. Song, K. Zhang, T. Gao, A. Wang, K. Li, Y. Sheng, J. Brot, D. Sokolov, A. Vivek, C. Leung, A. Sabnis, J. Bai, T. Zhao, M. Gottscho, D. Jackson, M. Luttrell, M. K. Shah, E. Chen, K. Liang, S. Jain, U. Thakker, D. Huang, S. Jairath, K. J. Brown, and K. Olukotun, "Sambanova sn40l: Scaling the ai memory wall with dataflow and composition of experts," 5 2024. [Online]. Available: http://arxiv.org/abs/2405.07518 http://dx.doi.org/10.1109/MICRO61859.2024.00100

[25] S. Lie, "Cerebras architecture deep dive: First look inside the hardware/software co-design for deep learning," *IEEE Micro*, vol. 43, pp. 18–30, 5 2023.

[26] A. Radford, K. Narasimhan, T. Salimans, I. Sutskever *et al.*, "Improving language understanding by generative pre-training," 2018.

[27] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates, Inc., 2017.

[28] M. Artetxe, S. Bhosale, N. Goyal, T. Mihaylov, M. Ott, S. Shleifer, X. V. Lin, J. Du, S. Iyer, R. Pasunuru *et al.*, "Efficient large scale language modeling with mixtures of experts," *arXiv preprint arXiv:2112.10684*, 2021.

[29] J. Kaplan, S. McCandlish, T. Henighan, T. B. Brown, B. Chess, R. Child, S. Gray, A. Radford, J. Wu, and D. Amodei, "Scaling laws for neural language models," *arXiv preprint arXiv:2001.08361*, 2020.

[30] A. Grattafiori, A. Dubey, A. Jauhri, A. Pandey, A. Kadian, A. Al-Dahle, A. Letman, A. Mathur, A. Schelten, A. Vaughan *et al.*, "The llama 3 herd of models," *arXiv preprint arXiv:2407.21783*, 2024.

[31] Meta AI, "Llama 4: Multimodal intelligence," https://ai.meta.com/blog/llama-4-multimodal-intelligence/, Apr. 2025, accessed: 2025-08-01.

[32] A. Yang, B. Yang, B. Zhang, B. Hui, B. Zheng, B. Yu, C. Li, D. Liu, F. Huang, H. Wei *et al.*, "Qwen2. 5 technical report," *arXiv preprint arXiv:2412.15115*, 2024.

[33] A. Yang, A. Li, B. Yang, B. Zhang, B. Hui, B. Zheng, B. Yu, C. Gao, C. Huang, C. Lv *et al.*, "Qwen3 technical report," *arXiv preprint arXiv:2505.09388*, 2025.

[34] A. Jiang, A. Sablayrolles, A. Mensch, C. Bamford, D. Chaplot, D. Casas, F. Bressand, G. Lengyel, G. Lample, L. Saulnier *et al.*, "Mistral 7b. arxiv 2023," *arXiv preprint arXiv:2310.06825*, 2024.

[35] D. Guo, D. Yang, H. Zhang, J. Song, R. Zhang, R. Xu, Q. Zhu, S. Ma, P. Wang, X. Bi *et al.*, "Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning," *arXiv preprint arXiv:2501.12948*, 2025.

[36] L. Zheng, W.-L. Chiang, Y. Sheng, T. Li, S. Zhuang, Z. Wu, Y. Zhuang, Z. Li, Z. Lin, E. P. Xing *et al.*, "Lmsys-chat-1m: A large-scale real-world llm conversation dataset," *arXiv preprint arXiv:2309.11998*, 2023.

[37] W. Kwon, Z. Li, S. Zhuang, Y. Sheng, L. Zheng, C. H. Yu, J. Gonzalez, H. Zhang, and I. Stoica, "Efficient memory management for large language model serving with pagedattention," in *Proceedings of the 29th Symposium on Operating Systems Principles*, 2023, pp. 611–626.

[38] T. Dao, D. Fu, S. Ermon, A. Rudra, and C. Ré, "Flashattention: Fast and memory-efficient exact attention with io-awareness," *Advances in neural information processing systems*, vol. 35, pp. 16 344–16 359, 2022.

[39] A. Agrawal, A. Panwar, J. Mohan, N. Kwatra, B. S. Gulavani, and R. Ramjee, "Sarathi: Efficient llm inference by piggybacking decodes with chunked prefills," *arXiv preprint arXiv:2308.16369*, 2023.

[40] E. Kurtic, A. Marques, S. Pandit, M. Kurtz, and D. Alistarh, “” give me bf16 or give me death”? accuracy-performance trade-offs in llm quantization,” *arXiv preprint arXiv:2411.02355*, 2024.

[41] A. Tschand, A. T. R. Rajan, S. Idgunji, A. Ghosh, J. Holleman, C. Kiraly, P. Ambalkar, R. Borkar, R. Chukka, T. Cockrell, O. Curtis, G. Fursin, M. Hodak, H. Kassa, A. Lokhmotov, D. Miskovic, Y. Pan, M. P. Manmathan, L. Raymond, T. S. John, A. Suresh, R. Taubitz, S. Zhan, S. Wasson, D. Kanter, and V. J. Reddi, “Mlperf power: Benchmarking the energy efficiency of machine learning systems from microwatts to megawatts for sustainable ai,” 10 2024. [Online]. Available: http://arxiv.org/abs/2410.12032

[42] F. Ferdaus, X. Wu, V. Taylor, Z. Lan, S. Shanmugavelu, V. Vishwanath, and M. E. Papka, “Evaluating energy efficiency of ai accelerators using two mlperf benchmarks,” in *2025 IEEE 25th International Symposium on Cluster, Cloud and Internet Computing (CCGrid)*. IEEE, 2025, pp. 549–558.

[43] “Ai in the data center : Harnessing the power of fpgas ebook series.” [Online]. Available: www.xilinx.com