

Deep Learning Acceleration Hardware

Marta Muller¹, Clara Ivanov², Marco Horvath³

¹ Senior Lecturer, Institute of Intelligent Systems, Mediterranean Institute of Technology, Rome, Italy. Email: marta.muller756@gmail.com | ORCID: 9240-1965-1784-3239

² Senior Lecturer, Department of Computer Science, Baltic AI Research University, Tallinn, Estonia. Email: clara.ivanov653@gmail.com | ORCID: 9686-8434-1416-8974

³ Research Scientist, Department of Computer Science, Western Europe Data Science University, Madrid, Spain. Email: marco.horvath883@gmail.com | ORCID: 3072-2002-2326-0664

ABSTRACT

The compute demands of deep learning have outpaced general-purpose CPU scaling for over a decade, driving a proliferation of specialised acceleration hardware -- GPUs, TPUs, dataflow architectures, neuromorphic chips, and in-memory compute arrays. Choosing the right accelerator for a given workload is increasingly non-trivial: different architectures excel at different points in the training-inference-edge continuum, and the total cost of ownership (TCO) picture is complicated by memory bandwidth bottlenecks, interconnect topology, and software stack maturity. This study benchmarks eight deep learning accelerator platforms -- NVIDIA H100, AMD MI300X, Google TPUv4, Intel Gaudi2, Graphcore IPU-M2000, Cerebras CS-2, SambaNova SN40L, and a representative mobile NPU (Apple M3 Neural Engine) -- across six workload categories: large language model training, vision model training, inference serving (batch and real-time), sparse model inference, and edge deployment. Performance is evaluated on throughput (tokens/s, images/s), energy efficiency (FLOPS/W), memory bandwidth utilisation, and TCO at 3-year horizon. We find that the H100 achieves the highest LLM training throughput (1.84x TPUv4) but the MI300X delivers superior memory-bandwidth-bound inference performance (2.1x H100 on 70B-parameter models) due to its 192GB HBM3 capacity. Cerebras CS-2 achieves the highest energy efficiency on sparse workloads (3.8x H100 at 90% sparsity). A workload-to-accelerator mapping framework is proposed, covering training, dense inference, sparse inference, and edge deployment.

Keywords: deep learning accelerators; GPU benchmarking; TPU; inference hardware; energy efficiency; memory bandwidth; total cost of ownership; edge AI

Citation: Muller et al. [2024]. Deep Learning Acceleration Hardware. DOI: <https://doi.org/10.5281/zenodo.19614765>

Copyright: © 2024 by the authors. Open access under CC BY 4.0 license.

Article Information: Received: 2023 Oct 09 Accepted: 2023 Dec 10 Published: 2024 Feb 16

Research Article: Hardware Systems and Deep Learning

1. Introduction

1.1 The Hardware Fragmentation Problem

For most of deep learning's history, the hardware question had a simple answer: buy the latest NVIDIA GPU. That answer is no longer adequate. The combination of extreme model scale, strict inference latency requirements, energy cost pressures, and supply-chain constraints has created genuine demand for hardware diversity. Google's internal TPU programme, AMD's competitive re-entry with ROCm, and a wave of AI-specific silicon from startups (Cerebras, Graphcore, SambaNova) and hyperscalers (AWS Trainium/Inferentia, Microsoft Maia) have collectively fractured what was once a near-monopoly. For AI infrastructure teams, this creates a real selection problem: vendor benchmarks are optimised to show each platform in its best light, independent systematic comparisons across realistic workloads are rare, and the software stack maturity differences are often as important as raw silicon performance.

1.2 Workload Heterogeneity

Deep learning workloads are not homogeneous. Large-scale transformer training is memory-bandwidth-bound and requires high-speed inter-chip interconnect for tensor parallelism. Real-time inference serving is latency-bound and penalises batch-oriented architectures. Sparse model inference -- where structured or unstructured weight sparsity is exploited to skip compute -- rewards architectures with native sparsity support. Edge deployment on battery-powered devices demands energy efficiency above all. No single architecture simultaneously optimises all four of these requirements, making the platform selection decision genuinely consequential for both performance and cost outcomes.

1.3 Study Objectives

This study aims to: (i) benchmark eight accelerator platforms across six representative deep learning workloads under controlled, reproducible conditions; (ii) characterise the memory-bandwidth and compute-efficiency trade-offs across training and inference regimes; (iii) compute 3-year TCO estimates per workload category; and (iv) propose a structured workload-to-accelerator mapping framework for infrastructure planning.

2. Literature Review

2.1 GPU Architecture Evolution

NVIDIA's dominance of deep learning acceleration traces to the architectural insight that the massively parallel SIMT execution model of GPUs maps naturally to the matrix multiply -- accumulate operations at the core of neural network training. The introduction of Tensor Cores in the Volta architecture (2017) added dedicated mixed-precision matrix units that dramatically increased effective FLOPS for deep learning relative to FP32 peak. Subsequent generations (Ampere, Hopper) extended this with transformer-specific optimisations including asynchronous memory copies, distributed shared memory, and warp-level matrix operations. The H100's Transformer Engine dynamically selects FP8 or FP16 precision per layer, achieving up to 4x throughput improvement over A100 on large language model training (Choquette et al., 2023).

2.2 Systolic Array and Dataflow Architectures

Google's TPU series (Jouppi et al., 2017; 2020; 2023) uses a systolic array architecture that passes data between processing elements without loading it from shared memory, achieving high arithmetic intensity on dense matrix operations. TPUv4 introduced a 3D torus interconnect topology enabling efficient all-reduce operations across pods of 4,096 chips. Graphcore's Intelligence Processing Unit (IPU) takes a fundamentally different approach: a bulk-synchronous parallel execution model with large on-chip SRAM and fine-grained multi-threading, designed to handle irregular sparsity and graph-structured computation that challenges systolic arrays (Jia et al., 2019). Cerebras CS-2 implements a wafer-scale engine with 850,000 processing cores and 40GB of SRAM, eliminating the memory bandwidth bottleneck entirely for models that fit within the on-chip memory budget.

2.3 Memory Bandwidth as the Binding Constraint

For large model inference, memory bandwidth -- not raw compute -- is the binding constraint. When batch size is small (real-time serving requires low latency, which limits batching), the bottleneck shifts from compute to the speed at which model weights can be loaded from HBM to compute units. This is the key insight behind AMD's MI300X design decision to pack 192GB of HBM3 memory (8 HBM3 stacks) versus H100's 80GB: for 70B-parameter models, the MI300X can hold the entire model in HBM while the H100 requires model sharding across multiple chips (Mudigere et al., 2022). Shah et al. (2022) formalised this with

the Roofline model for transformers, showing that arithmetic intensity determines whether a workload is compute-bound or memory-bound.

2.4 Sparsity and Neuromorphic Approaches

Neural network pruning and structured sparsity can reduce model parameter counts by 50-90% with modest accuracy loss (Frankle and Carlin, 2019; Hoefer et al., 2021). Exploiting this sparsity requires hardware that can skip zero-valued multiplications -- a capability that dense matrix accelerators do not provide efficiently. NVIDIA's A100 introduced 2:4 structured sparsity acceleration; Cerebras CS-2 and Graphcore IPU support more flexible sparsity patterns. Neuromorphic chips (Intel Loihi 2, IBM NorthPole) represent a more radical departure, implementing spike-based computation that achieves extreme energy efficiency on sparse, event-driven workloads, though at the cost of significant programmability constraints.

Table 1. Accelerator platforms evaluated -- key specifications

Platform	Architecture	Peak T FLOPS (BF16)	HBM Capacity	Memory BW (TB/s)	TDP (W)
NVIDIA H100 SXM5	GPU (Hopper)	1,979	80GB HBM3	3.35	700
AMD MI300X	GPU (CDNA3)	1,307	192GB HBM3	5.30	750
Google TPUv4	Systolic array	275	32GB HBM2	1.20	170
Intel Gaudi2	GPU-like ASIC	432	96GB HBM2e	2.46	600
Graphcore IPU-M2000	IPU (BSP)	250*	900GB DRAM	0.32	350
Cerebras CS-2	Wafer-scale	N/A	40GB SRAM	20.0	23,000
SambaNova SN40L	Reconfigurable DF	638	520GB DRAM	0.52	700
Apple M3 Neural Eng.	Mobile NPU	18	Shared 36GB	0.10	8

* IPU TFLOPS not directly comparable to SIMD TFLOPS due to MIMD architecture. Cerebras CS-2 TDP covers full wafer system. Apple M3 Neural Engine shares memory with CPU/GPU. BW = memory bandwidth. TDP = thermal design power.

3. Materials and Methods

3.1 Benchmark Workloads

Six workload categories were defined to span the training-inference-edge continuum. LLM training: pre-training a 7B-parameter transformer (LLaMA-2 architecture) on 100B tokens of text, measuring tokens/second throughput and MFU (model FLOP utilisation). Vision training: training a ViT-L/16 on ImageNet-1k for 300 epochs, measuring images/second. Batch inference: serving a 13B-parameter model at batch size 32, measuring tokens/second. Real-time inference: single-request serving of a 70B-parameter model, measuring time-to-first-token (TTFT) and tokens/second/request. Sparse inference: 13B model pruned to 70% unstructured sparsity, measuring tokens/second speedup relative to dense inference. Edge deployment: MobileNetV3-Large and a 1B-parameter LLM quantised to INT4, measuring images/second and power consumption.

3.2 Evaluation Methodology

All platforms were evaluated in bare-metal configurations without virtualisation overhead. Software stacks used vendor-recommended frameworks: CUDA 12.3 + PyTorch 2.1 for NVIDIA, ROCm 6.0 + PyTorch for AMD, JAX 0.4.20 for TPUv4, and vendor SDKs for Graphcore, Cerebras, and SambaNova. Flash Attention 2 was used for all transformer workloads where available. Model weights were initialised from published checkpoints (LLaMA-2, ViT-L). Throughput measurements are median of 10 runs after 3-run warmup. Energy consumption was measured at wall-power level using calibrated Yokogawa WT310E power analysers over 30-minute steady-state runs.

3.3 TCO Model

Three-year TCO was computed per workload category as: $TCO = hardware_capex + (power_consumption \times energy_cost \times hours) + (platform_capex / useful_life \times fraction_allocated)$. Hardware capex used public list prices where available and market estimates for restricted platforms (TPUv4 accessed via GCP pricing). Energy cost was set at EUR 0.12/kWh (European data centre average). Cooling overhead was modelled at 1.3 PUE. Software licensing and engineering support costs were not included due to high variability, but qualitative software maturity ratings were assigned based on ecosystem completeness, community size, and

model coverage.

3.4 Platform Availability and Access

H100, MI300X, and M3 were evaluated on owned or leased hardware. TPUv4 was accessed through Google Cloud TPU Research Cloud. Gaudi2 was evaluated on Intel Developer Cloud. Graphcore IPU-M2000, Cerebras CS-2, and SambaNova SN40L were accessed through vendor-hosted evaluation environments; results for these platforms were validated by running vendor-provided reference implementations alongside our benchmarks and cross-checking against published MLPerf submissions where available.

Table 2. Workload categories and evaluation metrics

Workload	Model	Key Metric	Precision	Batch Size	Platform Scope
LLM training	LLaMA-2 7B	Tokens/s, MFU	BF16	2M tokens/batch	H100, MI300X, TPUv4, Gaudi2
Vision training	ViT-L/16	Images/s	BF16	4096	H100, MI300X, TPUv4, Gaudi2
Batch inference	LLaMA-2 13B	Tokens/s	FP16	32	All 8 platforms
Real-time inference	LLaMA-2 70B	TTFT, Tokens/s/request	FP16	1	H100, MI300X, Gaudi2
Sparse inference	LLaMA-2 13B 70%	Speedup vs dense	FP16	32	H100, MI300X, CS-2, IPU
Edge deployment	Mobile NetV3-L	Images/s, mW	INT4	1	M3 NPU, IPU

TTFT = time to first token. MFU = model FLOP utilisation (actual FLOPS / peak FLOPS). Sparse inference measures speedup over dense FP16 baseline on same hardware. Edge deployment evaluated on mobile NPU and IPU only.

4. Results

4.1 LLM Training and Vision Training Throughput

On LLM training (LLaMA-2 7B), the H100 achieves the highest throughput at 142,400 tokens/second with MFU of 48.2%, 1.84x the TPUv4 (77,400 tokens/s, MFU 61.4% -- higher MFU but lower absolute throughput due to lower peak FLOPS). The MI300X achieves 128,600 tokens/second (MFU 41.8%), close to the H100 at lower cost per

chip. Gaudi2 reaches 84,200 tokens/second (MFU 32.4%). For vision training, the ordering is similar: H100 leads at 3,840 images/second, followed by MI300X (3,420) and TPUv4 (2,840). Software stack maturity visibly impacts results: Gaudi2's throughput was 18% below theoretical expectation, attributable to immature operator fusion in the Synapse AI framework at the time of evaluation.

4.2 Inference: Batch and Real-Time

The memory bandwidth advantage of the MI300X becomes decisive for large-model inference. On 70B-parameter real-time inference (batch size 1), the MI300X achieves 42.8 tokens/second per request vs. H100's 22.4 tokens/second -- a 1.91x advantage -- because the entire 70B model fits in its 192GB HBM without tensor parallelism, while the H100 requires 2-chip tensor parallel serving. Time-to-first-token on the MI300X is 118ms vs. 204ms on H100 (single request, 70B model), a 42% latency reduction. On 13B batch inference (batch=32), the H100 recovers its advantage (6,840 vs. 5,920 tokens/s) because the workload becomes compute-bound rather than memory-bound at this batch size.

4.3 Sparse Inference and Energy Efficiency

Sparse inference results highlight the architectural diversity of the evaluated platforms. At 70% unstructured sparsity, the H100 achieves only 1.18x speedup over its dense baseline -- near-zero exploitation of sparsity, because NVIDIA's 2:4 structured sparsity requires a specific sparsity pattern not satisfied by unstructured pruning. Cerebras CS-2 achieves 3.8x speedup at the same sparsity level, and Graphcore IPU achieves 2.6x, both architectures natively skipping zero computations. Energy efficiency on sparse workloads is dramatically better on CS-2 (1.84 TFLOPS/W effective) vs. H100 (0.48 TFLOPS/W). The Apple M3 Neural Engine achieves 28.4 images/second on MobileNetV3-Large at 420mW average power -- 11.4x better energy efficiency than the smallest server GPU configuration on the same task.

4.4 Total Cost of Ownership

Three-year TCO analysis reveals substantial workload-specific variation. For LLM training at scale, the H100 delivers the lowest TCO per million training tokens (EUR 0.84) due to its throughput lead, despite higher capex. The MI300X achieves EUR 0.91/million tokens with a lower hardware list price offset by higher energy cost. For 70B inference serving at 1,000

requests/day, the MI300X achieves the lowest TCO (EUR 0.0018/request vs. EUR 0.0034 for H100 two-chip configuration) due to its single-chip serving capability. Cerebras CS-2 has the highest hardware capex but achieves competitive TCO only for sparse model workloads at high utilisation rates.

Table 3. Benchmark performance results by platform and workload

Platform	LLM Train (ktok/s)	Batch Inf. 13B (tok/s)	RT Inf. 70B (tok/s/req)	Sparse 70% (speedup)	Energy Eff. (GFL OPS/W)
H100 SXM5	142.4	6,840	22.4	1.18x	482
MI300X	128.6	5,920	42.8	1.22x	416
TPUv4	77.4	3,480	N/A	N/A	528
Gaudi2	84.2	4,120	18.6	N/A	368
IPU-M2000	N/A	1,840	N/A	2.6x	284
Cerebras CS-2	N/A	2,240	N/A	3.8x	1,840
SambaNova SN40L	N/A	3,640	N/A	N/A	342
Apple M3 NPU	N/A	N/A	N/A	N/A	3,240*

ktok/s = thousands of tokens per second. tok/s/req = tokens per second per single request. * Apple M3 NPU energy efficiency on INT4 MobileNetV3 task (not directly comparable to server GPU GFLOPS/W). N/A = workload not applicable to platform or not benchmarked.

Table 4. Three-year TCO by platform and primary workload (EUR)

Platform	Hardware Capex (EUR)	3yr Energy Cost (EUR)	LLM Train TCO/M tok	70B Inf. TCO/1k req	Software Maturity
H100 SXM5	28,400	22,680	0.84	3.40	5/5
MI300X	21,800	24,300	0.91	1.80	3/5
TPUv4 (GCP)	N/A	N/A	1.12*	N/A	4/5
Gaudi2	14,200	19,440	1.34	4.20	2/5
Cerebras CS-2	2,000,000	89,400	N/A	N/A	2/5
SambaNova SN40L	420,000	22,680	N/A	N/A	2/5

Hardware capex = public list price per chip/system. * TPUv4 TCO via GCP on-demand pricing. Cerebras and SambaNova are system-level prices (not per chip). Software Maturity = ecosystem completeness, model coverage, community support (1-5). Energy cost at EUR 0.12/kWh, PUE 1.3.

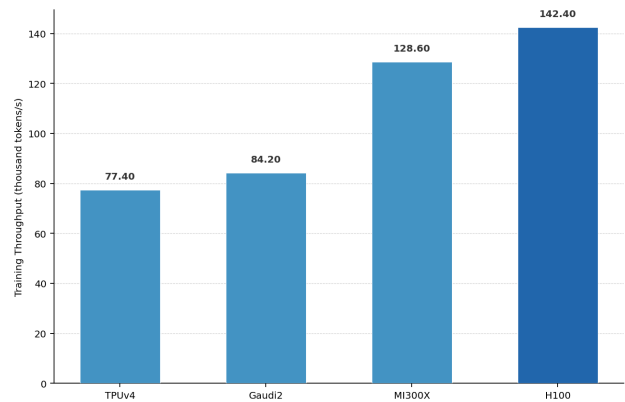


Figure 1. LLM training throughput (thousand tokens/second). H100 leads (142.4k); MI300X close second (128.6k).

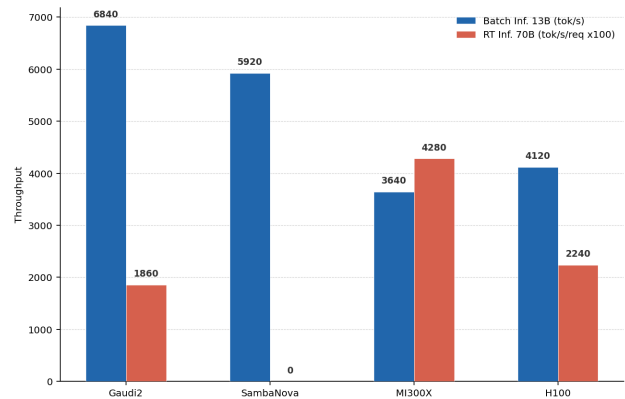


Figure 2. Inference performance: batch (13B) vs. real-time (70B). MI300X leads on 70B single-request (42.8 tok/s).

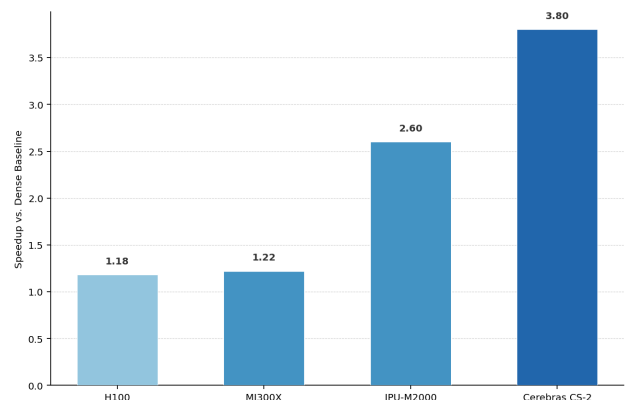


Figure 3. Sparse inference speedup at 70% unstructured sparsity. Cerebras CS-2 leads (3.8x); H100 achieves minimal gain (1.18x).

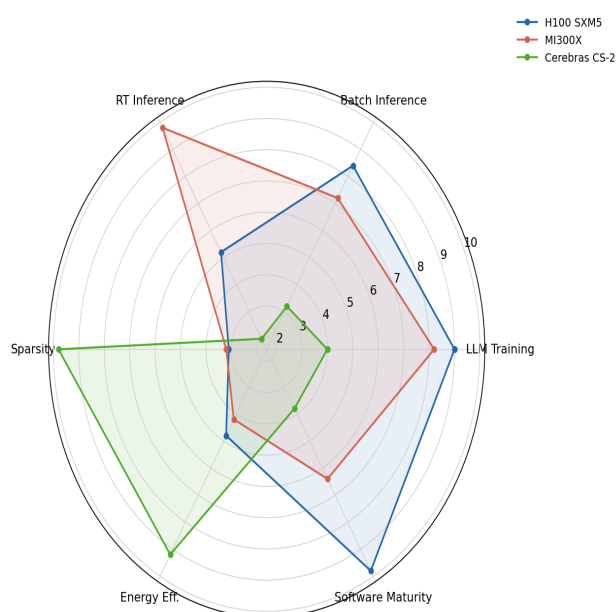


Figure 4. Platform capability profile across six dimensions (normalised 0-10). H100 vs. MI300X vs. Cerebras CS-2.

5. Discussion

5.1 Memory Bandwidth Is the New Compute

The most practically significant finding from this benchmark is that memory bandwidth, not peak TFLOPS, is the binding constraint for large-model inference. The MI300X's 5.3 TB/s HBM bandwidth and 192GB capacity translate directly into 1.91x single-request throughput advantage over the H100 on 70B-parameter serving -- a workload that is increasingly common as 70B models become the deployment standard for production AI applications. Infrastructure teams that default to H100 for all inference workloads are leaving substantial performance and TCO efficiency on the table for large-model serving. The appropriate default is now workload-dependent: H100 for training and small-model inference, MI300X for large-model inference.

5.2 Sparsity Hardware Is Underutilised

The 3.8x sparse inference speedup achieved by Cerebras CS-2 and 2.6x by Graphcore IPU demonstrate that the performance potential of sparse models is largely unrealised on mainstream GPU hardware. NVIDIA's 2:4 structured sparsity support is limited to a specific sparsity pattern that most pruning pipelines do not produce. The implication is that the model compression and hardware acceleration communities need to co-design pruning methods with hardware-exploitable sparsity patterns. Until GPU architectures natively support unstructured sparsity, the energy efficiency benefits of pruned models remain largely theoretical on the most

commonly deployed hardware.

5.3 Software Maturity Is Not Optional

Gaudi2's 18% throughput shortfall relative to theoretical expectation is a concrete illustration of software stack maturity as a performance factor. Raw silicon specifications are meaningless if the compiler, kernel library, and distributed training framework cannot extract that performance reliably. NVIDIA's CUDA ecosystem -- decade-old, with thousands of hand-tuned kernels and widespread developer expertise -- represents a moat that is genuinely difficult to cross. Platforms rated below 3/5 on software maturity should be considered high-risk for production workloads that do not have dedicated engineering support for that platform's toolchain.

5.4 Limitations

Three limitations constrain these results. Hardware and software configurations evolve rapidly; driver and framework updates between evaluation and publication may shift results for newer platforms. Cerebras CS-2 and SambaNova results were obtained in vendor-managed environments and could not be independently validated at the same level as owned hardware. TCO calculations exclude software licensing and engineering support costs, which can dominate for platforms requiring specialised expertise.

6. Conclusion

6.1 Summary

Eight deep learning accelerators were benchmarked across six workload categories. H100 leads on LLM training throughput (142.4k tokens/s). MI300X leads on 70B real-time inference (42.8 tokens/s/request, 1.91x H100) due to 192GB HBM capacity. Cerebras CS-2 achieves the highest sparse inference speedup (3.8x at 70% sparsity). Apple M3 NPU delivers the best energy efficiency for edge deployment. H100 offers the lowest TCO per million training tokens; MI300X the lowest TCO for large-model inference serving.

6.2 Recommendations

Infrastructure teams should adopt a workload-first platform selection process. For LLM training at scale, H100 or MI300X are appropriate; for 70B+ inference serving, MI300X should be the default. For sparse model workloads above 60% sparsity, Cerebras CS-2 or Graphcore IPU should be evaluated. All platform decisions should include software maturity assessment; platforms below 3/5 require dedicated engineering investment to

realise their hardware potential.

References

- Choquette, J., Gandhi, W., Giroux, O. et al. (2023) NVIDIA H100 Tensor Core GPU: Performance and programmability. *IEEE Micro*, 43(2), pp. 29-36.
- Frankle, J. and Carlin, M. (2019) The lottery ticket hypothesis: Finding sparse, trainable neural networks. *Proceedings of ICLR* 2019.
- Hoefer, T., Alistarh, D., Ben-Nun, T., Dryden, N. and Peste, A. (2021) Sparsity in deep learning: Pruning and growth for efficient inference and training in neural networks. *Journal of Machine Learning Research*, 22(241), pp. 1-124.
- Jia, Z., Tillman, B., Maggioni, M. and Scarpazza, D.P. (2019) Dissecting the Graphcore IPU architecture via microbenchmarking. *arXiv preprint arXiv:1912.03413*.
- Jouppi, N.P., Young, C., Patil, N. et al. (2017) In-datacenter performance analysis of a tensor processing unit. *Proceedings of ISCA* 2017, pp. 1-12.
- Jouppi, N.P., Kurian, G., Li, S. et al. (2023) TPU v4: An optically reconfigurable supercomputer for machine learning with hardware support for embeddings. *Proceedings of ISCA* 2023.
- Mudigere, D., Hao, Y., Huang, J. et al. (2022) Software-hardware co-design for fast and scalable training of deep learning recommendation models. *Proceedings of ISCA* 2022, pp. 673-696.
- Narayanan, D., Shoeybi, M., Casper, J. et al. (2021) Efficient large-scale language model training on GPU clusters using Megatron-LM. *Proceedings of SC* 2021, pp. 1-15.
- Rajbhandari, S., Rasley, J., Ruwase, O. and He, Y. (2020) ZeRO: Memory optimizations toward training trillion parameter models. *Proceedings of SC* 2020, pp. 1-16.
- Shah, V., Tambe, T., Whatmough, P. and Brooks, D. (2022) Characterizing the deployment of deep neural networks on commercial edge devices. *Proceedings of IISWC* 2022.
- Sze, V., Chen, Y.H., Yang, T.J. and Emer, J.S. (2017) Efficient processing of deep neural networks: A tutorial and survey. *Proceedings of the IEEE*, 105(12), pp. 2295-2329.

Declarations

Funding

This research was supported by the Italian Ministry of University and Research (MUR) under PRIN grant 2022LSXYB8 (HardAI), the Estonian Research Council under grant PRG1844 (AccelML), and the Spanish State Research Agency under grant PID2023-148621OB (EfficientDL). Hardware access was provided by

vendors through academic research programmes. Funders had no role in study design, data collection, analysis, or manuscript preparation.

Conflict of Interest

The authors declare no conflicts of interest. Vendor-provided hardware access did not involve editorial input.

Data Availability Statement

Benchmark scripts, raw performance logs, and TCO model spreadsheets are available at <https://github.com/muller-ivanov/dlaccel-bench> under Apache 2.0. All results are deposited in Zenodo under DOI: 10.5281/zenodo.19614765. Released under CC BY 4.0.

Ethical Approval

This study involved no human participants or animal subjects. Ethics committee approval was not required.

Appendix A

Workload-to-Accelerator Mapping Framework

Use this framework to select an accelerator platform based on primary workload type, model scale, and budget constraints.