2023 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW) | 979-8-3503-1199-0/23/$31.00 ©2023 IEEE | DOI: 10.1109/IPDPSW59300.2023.00019

Evaluating Energy Efficiency of GPUs using Machine Learning Benchmarks

Brett Foster, Shubbhi Taneja
Department of Computer Science
Worcester Polytechnic Institute
{befoster, staneja}@wpi.edu

Joseph Manzano, Kevin Barker
High Performance Computing Group
Pacific Northwest National Laboratory
{joseph.manzano, kevin.barker}@pnnl.gov

***Abstract*—As we enter the exascale era, the energy efficiency and performance of High-Performance Computing (HPC) systems, especially running Machine Learning (ML) applications, are becoming increasingly important. Nvidia recently released its 9th-generation HPC-grade Graphics Processing Unit (GPU) microarchitecture, Ampere, claiming significant improvements over the previous generation's Volta architecture. In this paper, we perform fine-grained power collection and assess the performance of these two HPC architectures' performance by profiling ML benchmarks. In addition, we analyze various hyperparameters, primarily the batch size and the number of GPUs, to determine their impact on these systems' performance and power efficiency. While Ampere is 3.16x more energy-efficient than Volta in isolation, this is counteracted by the PCIe interconnects of the A100s as the ML tasks are parallelized to run on more GPUs.**

***Index Terms*—high-performance computing, benchmarking, machine learning, GPU, Ampere, NVLink, nvprof, memory footprint, data movement, hugging face**

I. Introduction

The world of High-Performance Computing (HPC) has seen significant advances during the effort to reach the exascale era. Over the past decade, the world's most powerful HPC system has increased in performance 42x, from 10,510.0 TFlops in November 2011 to 442,010.0 TFlops in November 2021 [1]. As computation power increases, so does power usage. To continue pushing the performance of supercomputers, we must continue to improve the energy efficiency of these power-hungry devices.

A crucial factor in the advancements of HPC is the usage of Graphics Processing Units (GPUs), which have recently become integral to many HPC systems. Out of the world's ten most powerful supercomputers in 2021, seven of them used GPUs as their primary computing resource [1]. GPUs also lead the way in energy efficiency, relied on by nine out of the ten most energy-efficient supercomputing systems in the world [1]. In particular, these systems often use Nvidia's HPC-grade GPU microarchitectures. Nvidia released its ninth-generation HPC-grade Ampere architecture in May 2020 [2], claiming significant improvements in both performance and power usage over its predecessor, the Volta architecture, released three years prior [3].

One of the driving forces behind this progress is the evolution of artificial intelligence (AI), specifically Machine Learning (ML), which is particularly well-suited to run on GPUs and HPC systems. Machine learning has transformed several fields, such as language processing, computer vision, and speech recognition. The performance of training deep neural networks (DNN) and other ML applications on these HPC systems is thus exciting to the computing industry.

Our major contributions in this research are as follows. We perform fine-grained power collection to evaluate the performance and energy efficiency of the Ampere and Volta GPU architectures when running ML benchmarks. Specifically, we modify input data, batch size, hyperparameters, and number of GPUs to train transformer models for natural language processing to find the most energy-efficient configuration. In addition, we analyze how changing hyperparameters of these transformers' training benchmarks, primarily batch size and GPU quantity, impacts these metrics.

These measurements show that the Ampere architecture is 3.16x more energy-efficient in isolation than Volta. However, the interconnects of the Ampere GPUs—a central PCIe hub rather than a DGX-1 NVLink mesh—cause the energy efficiency to rapidly decrease rather than stay approximately constant as expected, as the ML tasks are parallelized to run on more GPUs. For example, as we increased from one to four Volta GPUs with 300 GB/s of interconnect bandwidth, their energy efficiency decreased by only 14.6%. In contrast, while the same increase for Ampere GPUs with 64 GB/s of interconnect bandwidth caused a decrease of 56.2%. We theorize that as interconnect bandwidth decreases, the GPUs spend a higher percentage of time waiting for GPUs in the cluster to communicate, which results in lower utilization and thus a lower energy efficiency.

The rest of this paper is organized as follows. In Section II, we discuss related work. Then, we present some background on the technologies used for our research in Section III. Section IV presents the methodology of our experimentation. After that, in Section V, we discuss the results. Finally, in Section VI, we conclude with a summary and describe our future work.

II. Related Work

Our work is related to three areas of interest: the need for energy-efficient computing, analyzing modern GPUs for state-

979-8-3503-1199-0/23/$31.00 ©2023 IEEE
DOI 10.1109/IPDPSW59300.2023.00019

of-the-art Machine Learning (ML) applications, benchmarking of HPC systems, and evaluating modern GPU interconnects. The following subsections will discuss some of the recent studies conducted in these three areas.

A. Energy Efficient computing

The increasing interest in HPC systems for solving complex problems in multiple domains has motivated researchers to investigate improving the energy efficiency of these power-hungry systems. As a result, energy efficiency has become a first-class constraint in designing data centers housing HPC systems [4]. Dynamic voltage and frequency scaling (DVFS) energy management technologies are one of the most popular power-management strategies that reduce energy consumption by dynamically adjusting the voltage and frequency parameters [5]. We discuss a select research that employs DVFS and other energy-efficient computing strategies in the remainder of this subsection.

A popular technique to lower the energy consumption of the HPC systems is to use nodes that support both frequency and voltage scaling [6]. In their research, Rajachandrasekar et al. address the challenge of energy-efficient checkpointing in HPCs by proposing a framework that performs data-funneling mechanisms and selective power capping to reduce the CPU utilization when checkpoint-restart happens, thus increasing the energy savings [7]. With their framework in place, they demonstrated a significant energy consumption reduction while improving performance. Another study, conducted by Devich et al., showed that a decrease in energy consumption could be achieved by a combination of (1) localizing rollbacks to the last checkpoint - *a technique to recover from failures that lead to unexpected failures in applications* and (2) frequency scaling for idle nodes [8]. Huang et al. presented a survey on GPU DVFS characterizations explicitly designed for GPUs. Through a series of experiments, they concluded that the effect of scaling core voltage/frequency and memory voltage/frequency is based on multiple factors like GPU architecture as well as GPU applications [9]. Data replication is a common strategy employed in data centers to ensure data availability and prevent data loss. When users request a copy of their data, the system responds by fetching a copy from various geographically distributed locations. In their research, Li et al. developed a replica selection system that considers bandwidth and latency and the total system energy cost [10]. Their algorithm showed a 12% savings in energy costs for data-intensive applications.

In our research, we aim to understand the energy efficiency of two popular GPU architectures, Ampere and Volta, by benchmarking them using ML applications to find avenues for saving energy.

B. Benchmarking

In order to compare the fastest computers or state-of-the art hardware, benchmarks evaluating their performance are required. The study conducted in this paper aims to assess the performance and understand characteristics of state-of-the-art HPC systems, primarily GPUs, using popular ML applications.

For over a decade, studies have underpinned the need for benchmarking on real applications. Keeping neural networks at the center, Blott designed a benchmarking methodology, namely QuTiBench, that aids system-level designers gain an understanding of the benefits and limitations of novel compute architectures [11]. Marjanović et al. emphasized the importance of benchmarking in their research and presented the performance results of the multiple problem sizes obtained on six different hardware architectures using three benchmarks, namely, HPL, HPCG, and HPGMG [12]. They discussed the impacts of changing data sizes on the system performance using the aforementioned benchmarks. Another similar study, by Ibrahim et al. stressed on understanding the performance characteristics of applications on hardware in order to leverage applications' characteristics [13]. In their research, they primarily focus on memory hierarchies and data movement across both single- and multiple-GPU.

Our work is similar to the above studies as we try to assess the performance of two hardware architectures, GPUs, by using ML benchmarking application but none of these studies pay attention to the energy consumption of the hardware. We evaluate and discuss the energy-efficiency of the two GPU architectures executing ML workload.

C. Modern GPU interconnects

Due to their high processing power and memory bandwidth, the use of GPUs as accelerators, especially for deep learning and big data processing, is not unheard of anymore. However, GPU-based systems run into transfer bottlenecks, limiting them from processing large data sets. Today, multiple modern GPU interconnects are available, including PCIe, NVLink, InfiniBand, and NVSwitch. Several studies have looked into the impacts of GPU interconnects on various state-of-the-art applications.

Li et al. also underscore the need for understanding the role of interconnect technologies in multi-GPU and HPC platforms [14]. They analyzed six modern GPUs on the Tartan benchmarking suite and also addressed four new types of NUMA effects for intra-node GPU communication. With the memory wall challenge in mind [15], Nabavinejad et al. surveyed developments in efficient on-chip interconnection and design methodology of the Deep Neural Network (DNN) accelerator design [16]. They evaluated the DNN performance under different interconnections and applications. For ASIC-based DNNs, they describe the trade-offs for array-based, mesh-based, and reconfigurable ones and the interconnection for the non-ASIC DNN computing platforms, including FPGAs, GPGPUs, Manycores, and embedded processors. They concluded that interconnections would continue to play an essential role in the designing of the DNNs. Even from the perspective of databases holding large volumes of data, the data transfer bottleneck is a primary reason for the slow adoption of GPUs for accessing databases [17], [18], [19].

In our study, we pay attention to two available interconnects - *PCIe* and *NVLink*. In Section V, we analyze the impacts these two interconnect technologies when training our two ML models - RoBERTa and GPT-2.

III. Background

In this section, we provide some background on the GPU microarchitectures used in this research, as well as the ML models used to benchmark these GPUs.

A. GPU Architectures

Volta is the eighth generation of Nvidia's HPC-grade GPU architectures, released in December 2017 [3]. It features a major new redesign of the Streaming Multiprocessor (SM) processor architecture which is optimized for deep learning. According to Nvidia, the Volta SM is 50% more energy efficient than the SM design of the previous generation, Pascal [20]. It is also the first of Nvidia's architectures to use tensor cores, which Nvidia claims achieves 12x higher peak TFlops when training neural networks and 6x higher peak TFLops for neural network inference.

Ampere is the ninth generation of Nvidia's HPC-grade GPU architectures, released in May 2020 [2]. Ampere's SM uses third-generation tensor cores, which, in addition to a 2x increase in computational horsepower per SM compared to Volta, add support for many new precisions, including bfloat16, TensorFloat-32, and double-precision floating-point (FP64). This generation of tensor cores also adds the ability to accelerate operations on sparse matrices for an additional 2x increase in speed. Ampere also features improvements to shared memory, increasing the size of it and the L1 data cache by 50% and introducing a new asynchronous copy instruction to lead data directly from global memory into shared memory, optionally bypassing the L1 cache.

B. Interconnects

PCIe, or the Peripheral Component Interconnect Express, is a high-speed serial computer expansion bus standard. It is generally used to connect one or more devices, such as GPUs, to the motherboard and the CPUs, as shown in Fig. 1-(A). Although it is flexible with what it can connect, PCIe is limited in speed by its central hub topology.

NVLink is a wire-based serial multi-lane communications link for near-range devices developed by Nvidia. It uses Nvidia's High-Speed Signaling interconnect (NVHS). Each NVLink consists of a sublink for each direction, and each one has of eight differential NVHS lanes. In addition to enabling both CPU-GPU and GPU-GPU links with direct reads and writes on remote CPU's main memory and peer GPU's device-memory, NVLink also allows multiple links per device. As a result, devices can use a mesh network to communicate, as shown in Fig. 1-(B), rather than a central hub like PCIe.

C. Transformers

In our experiments, we benchmark the GPUs with transformers [21]. Transformer models are an ML architecture introduced in 2017 that transform a sequence of elements, such as words in a sentence, into another sequence. The feature that sets transformers apart from previous sequence-to-sequence models is the self-attention mechanism, which weighs the relevance of a token in a sequence to each token in the same sequence. For example, when an input sequence is a sentence, the attention mechanism determines the relevant context for each word in the sentence. As a result, transformers perform better and are faster than previous sequence-to-sequence models, such as recurrent neural networks and long short-term memory. In addition, unlike these earlier models, transformers do not necessarily process their data in order, allowing for better parallelization.

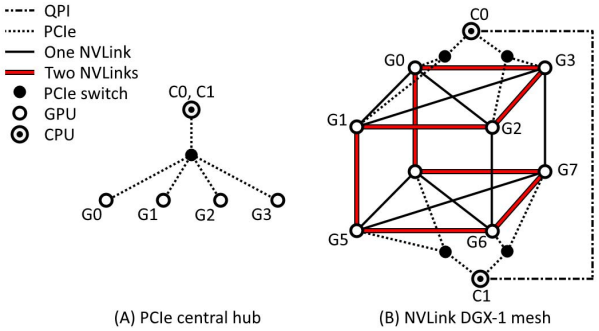


Fig. 1. PCIe vs NVLink interconnect topology

IV. Methodology

In this section, we describe the procedure used to generate the data used in our analysis and the metrics we collected.

A. Hardware Testbeds

We use two different systems in this research for experimentation purposes. The first system consists of a single node of eight Tesla V100-32GB SXM2 (Volta) GPUs on the Bridges-2 system at the Pittsburgh Supercomputing Center. The GPUs are connected with NVLink 2.0 in a mesh as seen in Fig. 1-(B). Each link supports data transfer rates up to 25 GB/s in each direction, and each V100 GPU supports up to six NVLinks. In Fig. 1-(B), each solid black line has a bandwidth of 25 GB/s in each direction, and each red line has a bandwidth of 50 GB/s in each direction. The Bridges-2 node also contains two Intel Xeon Gold 6248 "Cascade Lake" CPUs with 20 cores per CPU and clock speeds of 2.50 - 3.90 GHz and 512 GB of DDR4-2933 RAM.

The second system is a single node of four A100-40GB PCIe (Ampere) GPUs on the Rockfish system at Johns Hopkins University. These GPUs are connected via PCIe 4.0 with a central hub, as shown in Fig. 1-(A). The PCIe switch supports 2 GB/s per lane with 16 lanes, for a total bandwidth of 32 GB/s. The Rockfish node also contains two Intel Xeon Gold 6248R "Cascade Lake" CPUs with 24 cores per CPU, a clock speed of 3.0 GHz, and 192 GB of DDR4 RAM.

TABLE I
GPU ARCHITECTURES COMPARISON

Architecture	Volta	Ampere
Transistor Count	21.1 Billion	54.2 Billion
Tensor Cores	640 1st-generation	432 3rd generation
CUDA Cores	5120	6912
Boost Clock	1530 MHz	1410 MHz
Memory	32 GB	40 GB
Memory Bandwidth	900 GB/s	1555 GB/s
CUDA Compute Capability	7.0	8.0
Double-Precision Performance	7.8 TFlops	9.7 TFlops
Single-Precision Performance	15.7 TFlops	19.5 TFlops
Tensor Performance	125 TFlops	312 TFlops

B. Software Testbeds

To get a consistent software configuration on each system, we used Singularity [22] to build Hugging Face's [23] transformers-pytorch-gpu Docker container. The versions of Singularity, Python, PyTorch, CUDA, and Nvidia-SMI used are shown in Table III.

TABLE III
SOFTWARE TESTBED

Software	Version
Singularity	3.8.0
Python	3.6.9
CUDA	11.4
Nvidia-SMI	470.57.02
PyTorch	1.10.2

C. Experimentation

For each experiment, we ran Nvidia-SMI on every GPU to collect data on their utilization and power draws over the training period. We trained two different transformers models in our experiments: RoBERTa, a masked language model, and GPT-2, a causal language model, both of which are found in the transformers Python library [24] provided by Hugging Face [23]. The training period consisted of 200 training steps on a 10k slice of OpenWebText [25], the open-source replication of OpenAI's WebText dataset. Our methodology for experimentation is as follows. First, we collected the logs generated by Nvidia-SMI and the Hugging Face transformers library (See Fig. 2). These logs were then used to calculate four different metrics, which are described in Section IV-D. We performed three executions for every combination of hyperparameter values shown in Table IV.

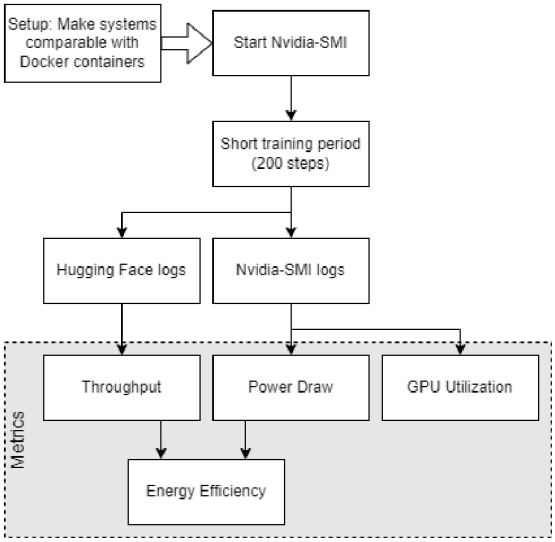


Fig. 2. Analysis pipeline

TABLE II
HARDWARE TESTBEDS

System	Bridges-2	Rockfish
GPU Architecture	Volta	Ampere
GPU Model	Tesla V100-32GB SXM2	A100-40GB PCIe
GPU Count	8	4
Max Power per GPU	300 W	250 W
Interconnect	NVLink 2.0	PCIe 4.0
Max Interconnect Bandwidth	300 GB/s	32 GB/s
CPUs	2 Intel Xeon Gold 6248	Intel Xeon Gold 6248R
CPU Cores	20 per CPU	24 per CPU
CPU Clock Speed	2.50 - 3.90 GHz	3.0 GHz
RAM	512 GB DDR4-2933	192 GB DDR4

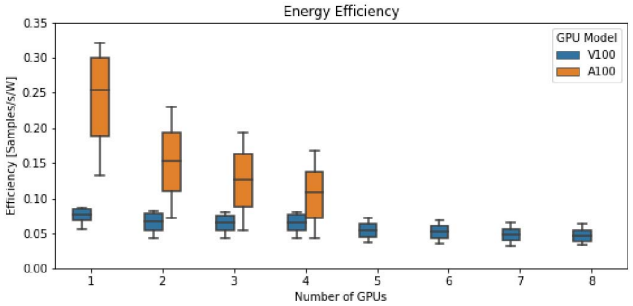


Fig. 3. The figure represents the energy efficiency of the two GPU architectures, A100 and V100, as the number of GPU varies from one to eight.

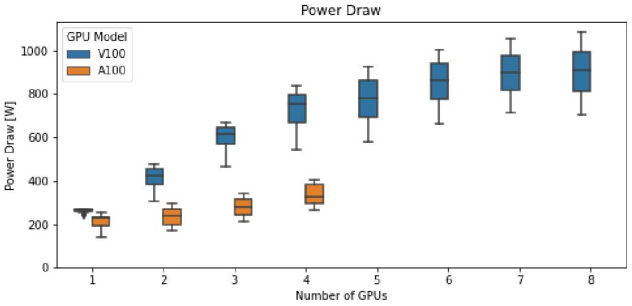


Fig. 4. The figure shows the correlation between the power drawn by the two GPU architectures, A100 and V100 as the number of GPUs are varied from one to eight(8).

D. Metrics

To analyze the performance and energy efficiency of the GPUs during our experiments, we used the following evaluation metrics:

Throughput: The mean number of samples processed per second by the GPU cluster during training. Throughput is a crucial metric for the performance of DNN training since the total number of samples processed generally stays constant regardless of other hyperparameters. The Hugging Face transformers library provided facilities to measure the throughput.

Power Draw: The mean power draw of the GPU cluster during training, measured in watts. The power draw of each GPU is measured every 100 ms by Nvidia-SMI.

Energy Efficiency: The throughput divided by the total power draw, measured in samples per second per watt, which is equivalent to samples per Joule or throughput per watt. Energy efficiency is especially relevant as a measurement of DNN training efficiency since, with a fixed number of total samples, higher energy efficiency means that the system uses less total energy.

GPU Utilization: The mean percentage of time during which one or more kernels were executing on each GPU. The GPU is responsible for performing critical operations for DNN training, so for optimal throughput, the GPU should always be busy. However, a GPU with high utilization also tends to have a higher power draw. Therefore, Nvidia-SMI measures each GPU's utilization every 100 ms. Then the average utilization of all GPUs is calculated as the mean of all utilization measurements collected during training.

V. Results and Discussion

In this section, we use the procedures described in the previous section (Sec. IV) to collect data on the Volta and Ampere GPU architectures. We analyze the energy efficiency, power draw, throughput, and utilization of the GPUs during ML training and discuss the potential bottleneck in the systems.

A. Energy Efficiency

Fig. 3 shows the energy efficiency of the A100 and V100 GPUs as the number of GPUs increases from one to eight. When the ML training is only run on a single GPU, the A100s significantly outperform the V100s, ranging from 2.58x to 3.66x more energy efficiency, with an average efficiency increase of 3.16x. As the number of GPUs increases, we expect to see each architecture's energy efficiency stay nearly constant. The V100 GPUs satisfy this, their efficiency only decreasing by 14.6% as the number of GPUs increases from one to four. However, increasing GPUs causes the A100s to fall in efficiency by 56.2%, indicating the existence of a bottleneck somewhere in the system. Although we were limited to only four A100s, the V100s continue their trend of slowly dropping in energy efficiency, only decreasing by another 25.8% as they increase further to eight GPUs.

TABLE IV
Experiment Parameters

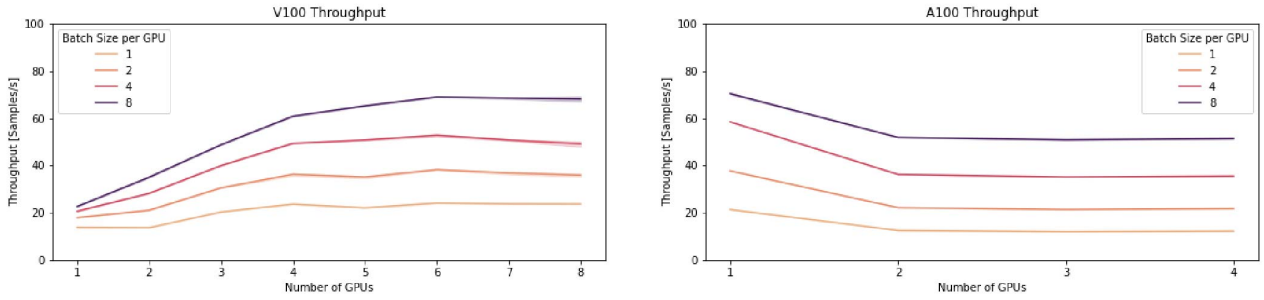
Parameter	Possible Values
GPU Quantity	1, 2, 3, 4 (and 5, 6, 7, 8 on the V100s)
Hugging Face Model	roberta-base, gpt2
Batch Size per GPU	1, 2, 4, 8, 16, 32
Learning Rate	2e-5, 5e-5, 1e-4

B. Power Draw

Fig. 4 shows the power draw of the A100 and V100 GPUs as the number of GPUs increases. Initially, the V100s and A100s use 262.8 W and 212.2 W of power, close to their maximum power draws of 300 W and 250 W, respectively. Increasing to four GPUs, the power draw of the V100s increases by 2.74x while the A100s only increase by 1.58x. Although the V100 GPUs increase in power usage faster than the A100 GPUs at first, their power usage starts to plateau after four GPUs, only increasing by an additional 1.25x with eight GPUs. Although the energy efficiency of the Volta GPUs appears to scale better than that of the Ampere GPUs, the opposite seems true for power draw, implying that the bottleneck cause primarily affects the throughput instead.

C. Throughput

Figure 5 shows the throughputs of the V100s and A100s, respectively, as the number of GPUs increases. We expect these graphs to trend upward since, with a constant batch size



(a) (b)

Fig. 5. The figures represent the system throughput (samples/sec) when batch size is changed along with the number of GPUs on (a) V100 (b) A100

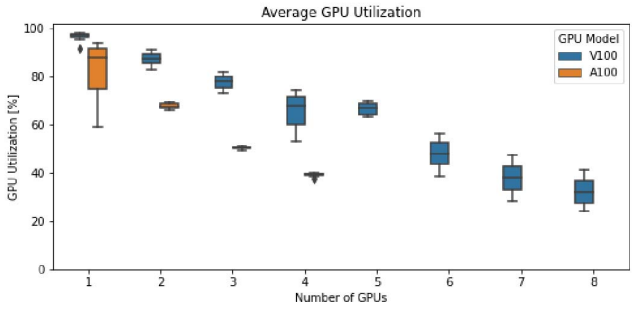


Fig. 6. The figure represents the correlation between GPU utilization and the number of GPUs.

per GPU, the total batch size increases with more GPUs. As expected, the V100s consistently speed up as more GPUs are added, increasing average speed by 2.24x as the system goes from one to eight GPUs. In contrast, the A100s slow down by 29.0% when they start to parallelize on a second GPU, and then the throughput stays nearly constant, decreasing by 1.0% as the number of GPUs increases to four. This behavior strongly points toward the A100 bottleneck only being applicable when the ML training is parallelized and not when it runs on only a single GPU.

D. GPU Utilization

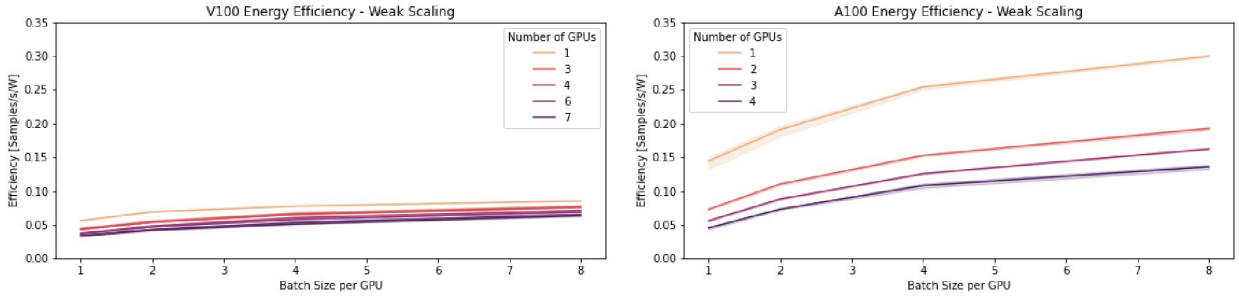
One potential cause for the differences in both power draw and throughput is the average GPU utilization of each system, shown in Fig. 6. Although the utilization decreases on both systems as the number of GPUs increases, the GPU utilization of the A100s decreases more quickly. Specifically, as the number of GPUs increases from one to four, the V100s and A100s decrease in utilization by 32.4% and 51.8%, respectively. The utilization of each GPU affects its power draw, with higher utilization resulting in slightly higher power draws. Partially because their utilization decreases more quickly, the power draw of the A100 GPUs grows more slowly as their quantity increases compared to the V100 GPUs, as seen in Fig. 4. In contrast, the GPUs should always be busy for optimal throughput, so a higher utilization results in higher throughput. We see this effect in Figure 5: both GPU models decrease in utilization as their number increases, and their throughput per GPU decreases as well. However, it is only with the A100s, which have a larger bottleneck limiting their utilization, that the decreasing throughput per GPU outweighs the increase in GPUs.

E. Batch Size

The Ampere and Volta GPUs exhibit both *strong scaling*, where the total batch size is fixed, and *weak scaling*, where the batch size per GPU is fixed. With weak scaling, we expect the throughput to increase approximately linearly with the number of GPUs, since the amount of time to process a batch remains roughly constant, and the time spent on back-propagation also remains unchanged, while the number of samples processed is proportional to the number of GPUs. Meanwhile, the total power draw of the system also increases approximately proportionally to the number of GPUs, since the utilization of each GPU stays the same. As a result, the energy efficiency of the system should be roughly the same for any number of GPUs.

With strong scaling, we expect the throughput to increase more slowly as more GPUs are added until the throughput reaches a limit. The throughput increases because the time spent processing each batch decreases as it's split between more GPUs. However, it is limited by the time spent on back-propagation and the total number of samples processed, both of which remain constant. Meanwhile, the power draw per GPU decreases as more GPUs are added, since their utilizations decrease. The total power draw of the system still increases, since this decrease is outweighed by power draw of an additional GPU. The energy efficiency could then either increase or decrease depending on whether the throughput or power draw increases faster.

Figures 7(a) and 7(b) show the weak scaling of the energy efficiencies of the V100s and A100s, respectively. As the batch size per GPU increases, the A100 GPUs scale much better than the V100s, increasing energy efficiency by 2.08x and 2.99x on one and four GPUs, respectively, while the V100s increase by 1.52x and 1.77x. The number of GPUs barely affects the energy efficiency of the V100s, as expected, only causing it



(a) (b)

Fig. 7. The figures represent energy efficiency (Samples/sec/Watts) of (a) V100 (b) A100 as the batch size per GPU as well as the total number of GPUs increase(a.k.a, weak scaling).

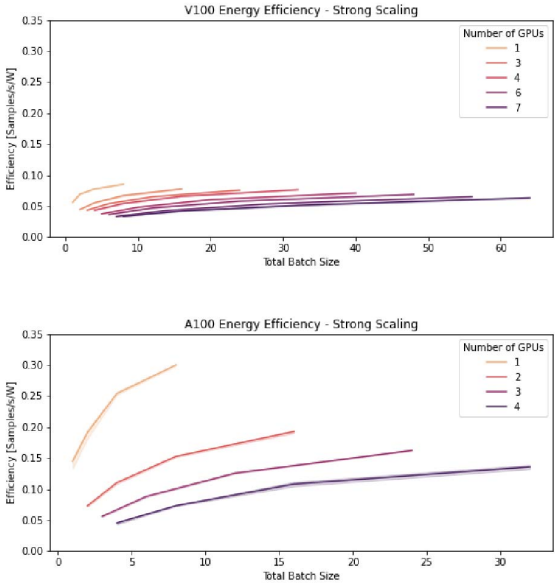


Fig. 8. The figures represent energy efficiency (Samples/sec/Watts) of (a) V100 (b) A100 as the number of GPUs increase while keeping the batch size constant (a.k.a, strong scaling).

to decrease by an average of 34.1% when going from one to eight GPUs. However, the energy efficiency of the A100s decreases by 59.3% when going from one to four GPUs.

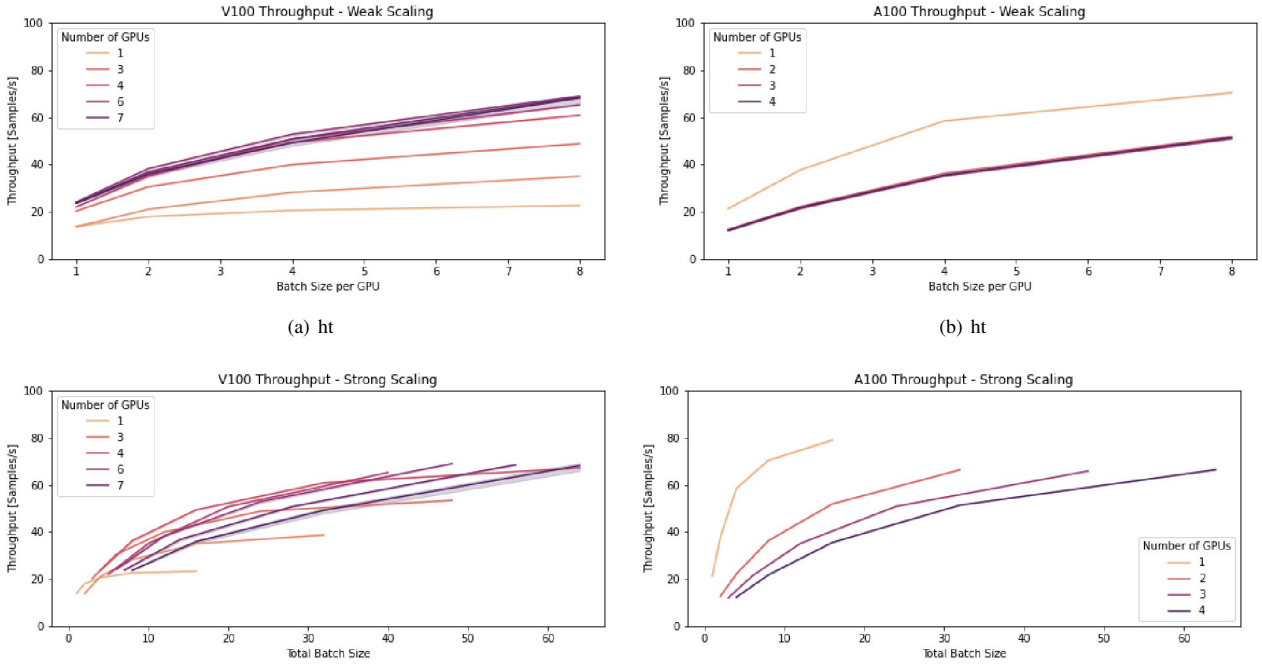
Fig. 8 show the strong scaling of the energy efficiencies of the V100s and A100s, respectively. As total batch size increases, the A100s scale better than the V100s. For example, with four GPUs, the energy efficiency of the A100s increases by 2.99x as the total batch size increases from 4 to 32, while the V100s only increase by 1.76x. Both GPU architectures scale better with total batch size when there are more GPUs.

Figures 9(a) and 9(b) show the weak scaling of the throughputs of the V100s and A100s, respectively. Regardless of the number of GPUs, as the batch size per GPU increases, the throughput scales similarly well on the A100s, with 3.28x and 4.16x increases in throughput for one and four GPUs, respectively. In contrast, the V100s scale better with more GPUs, with 1.63x and 2.57x increases in throughput for one and eight GPUs, respectively. As expected, when the batch size per GPU is fixed, the V100s increase in throughput with the number of GPUs. unlike the A100s which decrease in throughput.

Figures 9(c) and 9(d) show the strong scaling of the throughputs of the V100s and A100s, repectively. The V100s have better strong scaling for throughput than the A100s, since for a fixed total batch size, the V100 throughput is similar for any number of GPUs, while the A100 throughput is much higher with fewer GPUs. On the other hand, when the number of GPUs is fixed and the total batch size increases, A100s scale better, with an increase of 3.66x when going from a total batch size of 4 to 64 on four GPUs, while the V100s only increase by 1.86x.

F. Interconnects

We hypothesize that the bottlenecks present in the A100s are because of the PCIe interconnect. The reasons for this hypothesis are as follows. First, out of all the differences between the two systems used for experimentation, the change between the GPU interconnects is the only one where the V100s are significantly superior to the A100s. While the V100 cluster's CPUs have multiple paths to communicate with different GPUs, all the CPU-GPU communication in the A100 cluster must go through a central hub whose bandwidth is only 32 GB/s. This bandwidth is about the same as or worse than the links in the V100 mesh, which are either 25 GB/s or 50 GB/s. Next, a bottleneck with the interconnects in a GPU cluster would cause the cluster to spend more time communicating, decreasing the percentage of time spent actually executing kernels. This results in a lower utilization when more GPUs are added as can be observed with the A100s in Fig. 6. Finally, we use data-parallel training to scale with more GPUs, so that the samples in each batch are split between the GPUs in the cluster. However, this depends on communication requirements being relatively small compared to computation requirements to be maximally effective. As the communication requirements increase, such as when PCIe



(a) ht

(b) ht

(c) ht

(d) ht

Fig. 9. The figures represent the system throughput (Samples/sec) on V100s and A100s when weak ((a) & (b) and strong ((c) & (d)) scaling is implemented.

is used as an interconnect rather than an NVLink mesh, the scalability of data-parallel training decreases, as seen with the A100 GPUs.

VI. Conclusion

In isolation, Ampere is significantly faster than Volta for ML training, with an average speedup of 2.70x on a single GPU. Similarly, Ampere is 3.16x more energy efficient than Volta on a single GPU. However, once the ML training is parallelized to more GPUs, the A100 GPUs become bottlenecked by their PCIe central hub interconnect. With Volta GPUs using an DGX-1 mesh, energy efficiency stays nearly constant. At the same time, Ampere GPUs connected via PCIe quickly decline down to being 21.0% slower and only 1.65x more energy efficient than Volta when running on four GPUs.

Acknowledgment

This work was partially supported by the U.S. DOE Office of Science, Office of Advanced Scientific Computing Research, under award 66150: "CENATE - Center for Advanced Architecture Evaluation" project. The Pacific Northwest National Laboratory is operated by Battelle for the U.S. Department of Energy under contract DE-AC05-76RL01830. For experimentation purposes, this work used the Extreme Science and Engineering Discovery Environment (XSEDE), now known as ACCESS, which is supported by National Science Foundation grant number ACI-1548562. Specifically, it used the Bridges-2 system, which is supported by NSF award number ACI-1928147, at the Pittsburgh Supercomputing Center (PSC), as well as the Rockfish system, which is supported by NSF award number ACI-1920103, at Johns Hopkins University (JHU).

References

[1] E. Strohmaier, J. Dongarra, H. Simon, M. Meuer, and H. Meuer. The top500 list. [Online]. Available: https://www.top500.org/

[2] Nvidia. (2020) Nvidia a100 tensor core gpu architecture. [Online]. Available: https://images.nvidia.com/aem-dam/en-zz/Solutions/data-center/nvidia-ampere-architecture-whitepaper.pdf

[3] ——. (2017) Nvidia tesla v100 gpu architecture. [Online]. Available: https://images.nvidia.com/content/volta-architecture/pdf/volta-architecture-whitepaper.pdf

[4] R. Buyya, A. Beloglazov, and J. Abawajy, "Energy-efficient management of data center resources for cloud computing: A vision, architectural elements, and open challenges," *arXiv preprint arXiv:1006.0308*, 2010.

[5] E. Le Sueur and G. Heiser, "Dynamic voltage and frequency scaling: The laws of diminishing returns," in *Proceedings of the 2010 international conference on Power aware computing and systems*, 2010, pp. 1–8.

[6] V. W. Freeh, D. K. Lowenthal, F. Pan, N. Kappiah, R. Springer, B. L. Rountree, and M. E. Femal, "Analyzing the energy-time tradeoff in high-performance computing applications," *IEEE Transactions on Parallel and Distributed Systems*, vol. 18, no. 6, pp. 835–848, 2007.

[7] R. R. Chandrasekar, A. Venkatesh, K. Hamidouche, and D. K. Panda, "Power-check: An energy-efficient checkpointing framework for hpc clusters," in *2015 15th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*, 2015, pp. 261–270.

[8] K. Dichev, K. Cameron, and D. S. Nikolopoulos, "Energy-efficient localised rollback via data flow analysis and frequency scaling," in *Proceedings of the 25th European MPI Users' Group Meeting*, ser. EuroMPI'18. New York, NY, USA: Association for Computing Machinery, 2018. [Online]. Available: https://doi.org/10.1145/3236367.3236379

[9] Y. Huang, B. Guo, and Y. Shen, "Gpu energy consumption optimization with a global-based neural network method," *IEEE Access*, vol. 7, pp. 64 303–64 314, 2019.
[10] B. Li, S. L. Song, I. Bezakova, and K. W. Cameron, "Edr: An energy-aware runtime load distribution system for data-intensive applications in the cloud," in *2013 IEEE International Conference on Cluster Computing (CLUSTER)*, 2013, pp. 1–8.
[11] M. Blott, L. Halder, M. Leeser, and L. Doyle, "Qutibench: Benchmarking neural networks on heterogeneous hardware," *ACM Journal on Emerging Technologies in Computing Systems (JETC)*, vol. 15, no. 4, pp. 1–38, 2019.
[12] V. Marjanović, J. Gracia, and C. W. Glass, "Hpc benchmarking: Problem size matters," in *2016 7th International Workshop on Performance Modeling, Benchmarking and Simulation of High Performance Computer Systems (PMBS)*, 2016, pp. 1–10.
[13] K. Z. Ibrahim, T. Nguyen, H. A. Nam, W. Bhimji, S. Farrell, L. Oliker, M. Rowan, N. J. Wright, and S. Williams, "Architectural requirements for deep learning workloads in hpc environments," in *2021 International Workshop on Performance Modeling, Benchmarking and Simulation of High Performance Computer Systems (PMBS)*. IEEE, 2021, pp. 7–17.
[14] A. Li, S. L. Song, J. Chen, J. Li, X. Liu, N. R. Tallent, and K. J. Barker, "Evaluating modern GPU interconnect: PCIe, NVLink, NV-SLI, NVSwitch and GPUDirect," *IEEE Transactions on Parallel and Distributed Systems*, vol. 31, no. 1, pp. 94–110, jan 2020. [Online]. Available: https://doi.org/10.1109%2Ftpds.2019.2928289
[15] S. A. McKee, "Reflections on the memory wall," in *Proceedings of the 1st conference on Computing frontiers*, 2004, p. 162.
[16] S. M. Nabavinejad, M. Baharloo, K.-C. Chen, M. Palesi, T. Kogel, and M. Ebrahimi, "An overview of efficient interconnection networks for deep neural network accelerators," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 10, no. 3, pp. 268–282, 2020.
[17] C. Lutz, S. Breß, S. Zeuch, T. Rabl, and V. Markl, "Pump up the volume: Processing large data on gpus with fast interconnects," in *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 1633–1649. [Online]. Available: https://doi.org/10.1145/3318464.3389705
[18] C. Gregg and K. Hazelwood, "Where is the data? why you cannot debate cpu vs. gpu performance without the answer," in *(IEEE ISPASS) IEEE International Symposium on Performance Analysis of Systems and Software*, 2011, pp. 134–144.
[19] Y. Yuan, R. Lee, and X. Zhang, "The yin and yang of processing data warehousing queries on gpu devices," *Proc. VLDB Endow.*, vol. 6, no. 10, p. 817–828, aug 2013. [Online]. Available: https://doi.org/10.14778/2536206.2536210
[20] Nvidia, "Nvidia tesla p100," 2016. [Online]. Available: https://images.nvidia.com/content/pdf/tesla/whitepaper/pascal-architecture-whitepaper.pdf
[21] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," 2017. [Online]. Available: https://arxiv.org/abs/1706.03762
[22] SingularityCE Developers, "SingularityCE 3.8.3," Sep 2021, doi:10.5281/zenodo.5564915.
[23] H. Face. Hugging face. [Online]. Available: https://huggingface.co/
[24] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. L. Scao, S. Gugger, M. Drame, Q. Lhoest, and A. M. Rush, "Huggingface's transformers: State-of-the-art natural language processing," 2019. [Online]. Available: https://arxiv.org/abs/1910.03771
[25] A. Gokaslan and V. Cohen. (2019) Openwebtext corpus. [Online]. Available: https://skylion007.github.io/OpenWebTextCorpus/