Open Compass: Accelerating the Adoption of AI in Open Research

Paola A. Buitrago
Pittsburgh Supercomputing Center
Carnegie Mellon University
Pittsburgh, PA, USA
paola@psc.edu

Nicholas A. Nystrom
Pittsburgh Supercomputing Center
Carnegie Mellon University
Pittsburgh, PA, USA
nystrom@psc.edu

ABSTRACT

Artificial intelligence (AI) has immense potential spanning research and industry. AI applications abound and are expanding rapidly, yet the methods, performance, and understanding of AI are in their infancy. Researchers face vexing issues such as how to improve performance, transferability, reliability, comprehensibility, and how better to train AI models with only limited data. Future progress depends on advances in hardware accelerators, software frameworks, system and architectures, and creating cross-cutting expertise between scientific and AI domains. Open Compass is an exploratory research project to conduct academic pilot studies on an advanced engineering testbed for artificial intelligence, the Compass Lab, culminating in the development and publication of best practices for the benefit of the broad scientific community. Open Compass includes the development of an ontology to describe the complex range of existing and emerging AI hardware technologies and the identification of benchmark problems that represent different challenges in training deep learning models. These benchmarks are then used to execute experiments in alternative advanced hardware solution architectures. Here we present the methodology of Open Compass and some preliminary results on analyzing the effects of different GPU types, memory, and topologies for popular deep learning models applicable to image processing.

CCS CONCEPTS

• **Computing methodologies** → **Artificial intelligence** • Computing methodologies → Knowledge representation and reasoning • **Hardware** → **Analysis and design of emerging devices and systems**

KEYWORDS

Evaluation of emerging technologies, Artificial Intelligence, High Performance Computing, Training, Education

 ACM ISBN 978-1-4503-7227-5/19/07. https://doi.org/10.1145/3332186.3332253

1 Introduction

Artificial intelligence (AI) has immense potential spanning research and industry. Research initiatives, conferences, investment, and products based on AI abound and are expanding rapidly [1, 2], yet the methods, performance, and understanding of AI are actually in their infancy. The progress that has been made is impressive [3–6], for example the breakthrough of *Libratus* for decision-making with imperfect information [7, 8] enabled by converged high performance computing (HPC) and AI on the *Bridges* supercomputer [9, 10], but greater challenges lie ahead. Researchers face vexing issues such as how to improve performance, transferability, reliability, comprehensibility, and how better to train AI models when available data is quite limited [11]. Future progress depends on advances in hardware accelerators, software frameworks, system architectures, and creating cross-cutting expertise between scientific and AI domains.

Recently, the combination of graphics processing units (GPUs) and AI have emerged as a disruptive technology, changing the way that we obtain insights and create value from data. Ongoing advances and forthcoming, qualitatively different technologies will open the door to faster, deeper, and more productive ways of analyzing and working with data. Advances in technology are outpacing traditional system acquisition cycles, creating an urgent need to deeply understand the potential of new kinds of accelerators, interconnects, volatile and nonvolatile memory, and other hardware components. The community needs to develop experience with new technologies as early as possible to build applications, develop software ecosystems around the most promising technologies, and inform future large-scale investment. Infrastructure alone is insufficient to develop this experience, and many existing benchmarks (e.g., training ImageNet [12, 13]) do not reflect the different requirements of research. A concerted effort between domain experts and AI experts is needed to create workloads representative of research and to understand the value of new technologies and to make the most of emerging hardware and new and platform-optimized software frameworks. Hands-on experience with new technologies is needed to establish and disseminate best practices and to evaluate the potential of diverse technologies.

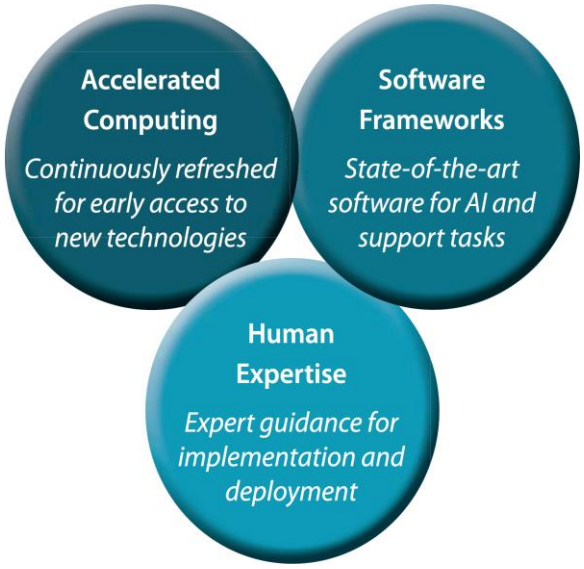

Figure 1. PSC's Compass Lab integrates emerging hardware accelerators including processors, memory, and interconnects with advanced software frameworks and human expertise to let users make informed decisions for investment and implementation.

The Pittsburgh Supercomputing Center (PSC), a research center of Carnegie Mellon University and the University of Pittsburgh, launched the *Compass Lab,* an advanced engineering testbed for artificial intelligence research and pilot projects. The Compass Lab makes available emerging, continuously-refreshed hardware and software technologies, backed by human expertise and training, to help users "find their direction" in the complex AI landscape (Figure 1). Compass Lab is a unique collaboration between a national HPC center, academia, hardware and software vendors, and private-sector startups, accelerators, and incubators.

Open Compass is an exploratory research project, focusing on academia, to conduct pilot projects on this advanced engineering testbed for artificial intelligence, culminating in the development and dissemination of best practices. Open Compass builds on the Compass Lab to make those advanced technologies available also to the open research community.

1.2 Related Work

Several other significant efforts are underway to correlate the performance of AI frameworks and models with various hardware. These initiatives fall into two categories: organized efforts to understand the relationship of performance to architecture in a more systematic way, and published benchmark specifications with periodic submissions by the community.

Fathom [14] is a set of eight workloads (i.e., model plus data) consisting of specific seminal works in deep learning from 2012 to 2015. Fathom's coverage addresses speech, image classification, and game learning, using different model types (convolutional, recurrent, fully-connected, memory), learning types (supervised, unsupervised, reinforcement), and datasets. However, Fathom's workloads are now dated, and they do not address many important learning tasks in research.

MLPerf [15], driven by a broad industry consortium, aims to accelerate progress in machine learning by providing a set of benchmarks for image classification, object detection, speech-to-text, translation, recommendation, sentiment analysis, and reinforcement learning. Like the Top500 list in high-performance computing (HPC), MLPerf quickly came to be a competition, with the top published results reflecting runs on impressively large-scale AI infrastructure. DAWNBench [16] is similar to MLPerf, also providing a benchmark and competition around industry-standard datasets (e.g., ImageNet) and models.

The HPE Deep Learning Cookbook [17] takes a different approach, providing a benchmarking suite, performance guide, and reference designs to characterize workloads and recommend optimal hardware and software platforms for executing them.

Of the projects summarized above, Fathom, MLPerf, and DAWNBench are narrowly focused on providing benchmarks for the community to execute, in some cases competitively. The HPE Deep Learning Cookbook is closest in spirit to Open Compass in its focus on providing insights to AI users. Their emphasis is on datasets relevant to widespread commercial applications, for example, classification of low-resolution images and speech recognition and translation. Their approach has great value, but it does not address complementary requirements of research data that pose qualitatively different challenges.

For organizing knowledge of AI technologies, existing computer ontologies do not address computer hardware at the right level of detail, nor do they address AI software. Hwaitat *et al.* [18] focus on the formal defensible logic for a basic ontology that is not well-suited to characterizing hardware for AI workloads. They address a few aspects of CPU, memory, and I/O devices, but they do not address accelerators, interconnects, or object or data properties. Faheem *et al.* [19] propose an ontology for mapping applications onto heterogeneous architectures. Their focus is primarily on application mapping, so some ideas from their approach may eventually interface well with the AI hardware and software ontology developed in Open Compass.

Open Compass differs and adds value in two vital ways. First, Open Compass focuses on developing and disseminating *best practices* for application of AI to research, which goes well beyond just measuring performance. Second, Open Compass explicitly addresses AI for research data, for example, volumetric medical imaging, genomic, and medical time-series data, as described in section 2.2.

2 Methodology

Open Compass leverages the PSC Compass Lab to execute pilot studies and address research questions relevant across scientific domains. Open Compass methodology includes the preliminary evaluation of alternative hardware solutions in terms of the produced training performance of different neural networks. The "Cambrian explosion" of architectural innovation makes Open Compass very timely, and it also requires careful attention to diversity of computer architectures, what they have in common, and how they differ.

2.1 Some Relevant AI Concepts

Artificial intelligence, particularly deep learning [20], is advancing very rapidly. It is beyond the scope of this paper to provide an in-depth introduction (the interested reader is referred to, for example, [21] and recent NeurIPS, ICML, and related conference proceedings); however, it is helpful to survey some directly relevant concepts.

Deep neural networks (DNNs) have demonstrated great capacity for representation learning, which is the ability to learn features from data. Representation learning allowed DNNs to surpass prior machine learning techniques for many domains such as image classification [3], speech recognition and translation, and games of strategy [22].

DNNs are built on many (typically tens to hundreds) two-dimensional layers of artificial "neurons" of several basic types, with different properties and connectivity between them. The connections are "weights", which are optimized through an iterative process known as "training". The specification of layers and how they are connected is a "network" or a "model", and the model plus a set of weights optimized for a particular application (e.g., recognizing melanoma in photos of skin) is a "trained model". The trained model may then be applied to new data, a process called "inferencing", to classify, predict, or otherwise generate information.

Different kinds of networks are suited to different kinds of learning. Examples of networks that are currently in widespread use are Convolutional Neural Networks (CNNs) [23] for image classification and segmentation; Recurrent Neural Networks (RNNs) [24] and Long Short-Term Memory (LSTM) [6] for time-series data such as speech recognition, generation, and translation; Generative Adversarial Networks (GANs) [11] for combining other DNNs into generator and discriminator pairs that can augment limited raw data. Other networks are being devised, such as Capsule Networks [25] to model hierarchical relationships and 3D CNNs for three-dimensional image data and also video data (2D images plus time). The performance of different DNNs depends strongly and in different ways on computation, memory capacity, bandwidth and latency to memory and storage, and other factors.

Understanding of which hardware would be best – as defined by fastest, most cost-effective, lowest-power, or other some other measure – for a specific training or inference application is generally quite limited. Graphics processing units (GPUs) are generally regarded as a good choice for training, but often without understanding of which GPU would be optimal. Similarly, CPUs, differently-configured GPUs, and FPGA (field-programmable gate arrays) are often used for inferencing. But the real choice is far more complicated. For example, for a given application, is high-bandwidth memory (HBM2) beneficial, *how much does it help*, and *what capacity is needed*? *Can the application effectively scale to multiple GPUs within a node*, and if so, *what internal topology is best*? Need it be a full crossbar, or is a hybrid mesh cube or even a ring good enough? Can the application scale to multiple nodes, and if it can, how much internode bandwidth is needed to avoid a bottleneck? These are already hard questions for which researchers need answers. Many new architectural options are being developed, to be introduced over coming years. Researchers need understanding of their AI applications' requirements relative to hardware architecture to make informed decisions about which hardware technologies to invest in or gain access to, as well as in which software frameworks they should invest their time. Open Compass is addressing that need for the research community.

2.2 PSC Compass Program

PSC's Compass Program focuses on the application of AI technologies to address open challenges in research. It consists of four main components: the *Compass Center*, *Compass Lab*, *Compass Consortium*, and *Open Compass*.

The *Compass Center* brings together faculty, staff, and members from industry, government, academic and nonprofit organizations to advance AI research and development and accelerate the adoption of AI across various fields, industries, and sectors of society. The *Compass Lab* (Figure 1) provides unique access through PSC to new hardware and software technologies for AI, backed by human expertise and partnering with faculty thought leaders to develop solutions. The *Compass Consortium* provides, depending on level of membership, access to seminars, meetings, presentations, publications, reports, case studies, technical briefings, and benchmark results; access to consortium projects; member-directed projects emphasizing the application of new AI technologies to derive value from data; networking with faculty, students, vendors, and other members; connections to domain experts; training; input through an advisory board to define consortium projects; and other benefits.

Open Compass brings the benefits of the Compass Lab to the academic research community. This is important because quite often, research data differs significantly from the kinds of data that dominate commercial applications to date and that drove the development of today's most popular deep learning frameworks and models. Examples of data that are not covered by standard benchmarks are segmentation of volumetric (3D) image data to indicate and identify structures in human organs (e.g., neurons in brains and nephrons in kidneys), analysis of large (e.g., 100,000×100,000 pixels) whole-slide images that are routine in digital pathology, classification of anomalous patterns in medical time-series data (e.g., classification of different kinds of arrhythmia in electrocardiogram data), analysis of 4k and 8k video data, and imputation of extended genomic information from microarray data.

Through a suite of pilot applications and working with leading domain researchers, Open Compass is expanding the community's understanding of algorithmic requirements on hardware and software resources. This paper establishes the methodology, kinds of performance experiments, and performance measurement protocol, initially applying them to a set of well-known workloads to establish a strong foundation.

The Open Compass project also provides expert assistance in applying Compass Lab resources and collaborative development of algorithms, models, and software; develops and disseminates

best practices; and provides training tailored to the open research community.

2.3 Compass AI Hardware Ontology

The proliferation of hardware technologies being developed presents a daunting combinatoric challenge if viewed from the perspective of planning to benchmark everything. Researchers are presented with a bewildering array of processors and accelerators, tiered memory and data management systems built on a mix of new and existing technologies, different interconnect technologies both between processors and between nodes, different multiples of the preceding components for scaling, different topologies, and different software layers optimized for particular hardware. Traditional factorial design on even one node can be prohibitive in the time that would be required for the resulting runs. To understand the impacts of emerging technologies on AI performance and how to apply specific technologies most effectively, a more carefully planned approach is needed.

We therefore developed an ontology of AI architectures that has value in framing experiments and assessing performance in a useful, interpretable way. Over time, the Compass ontology will be expanded to factor in additional hardware architectures as they emerge. Software aspects are being added, and performance results are being integrated, thereby creating a knowledge base that can be queried, added to, and expanded.

The Compass Ontology is developed in Protégé [26] and is available as an OWL specification [27]. Figure 2 illustrates the hardware class hierarchy, with (sub)classes of the most relevance to this discussion expanded. The hierarchy allows intuitive specification of object properties, each with specified ranges and domains to allow checking of axioms. For example, class *Processor* has object property (among others) of *hasMemoryType* to represent the type of memory that it supports. That property is inherited by subclasses *CPU* and *GPU* and instances (called "individuals" in Protégé) thereof. This hierarchical representation is helpful when thinking about how architectural components relate to each other and, for Open Compass, to AI tasks and

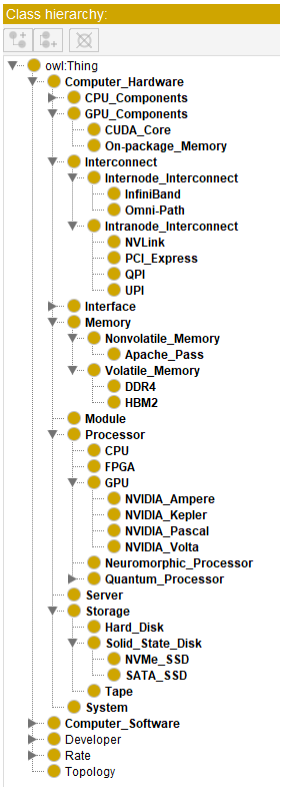

Figure 4. Class hierarchy for the Compass Ontology, with hardware classes selectively expanded to illustrate key concepts. The relatively high level of detail is required to accurately represent and reason on the differences between architectures. Complementing the class hierarchy are object properties that describe attributes of class members, individuals that are instances of the classes, data properties that are attributes of individuals, and data types.

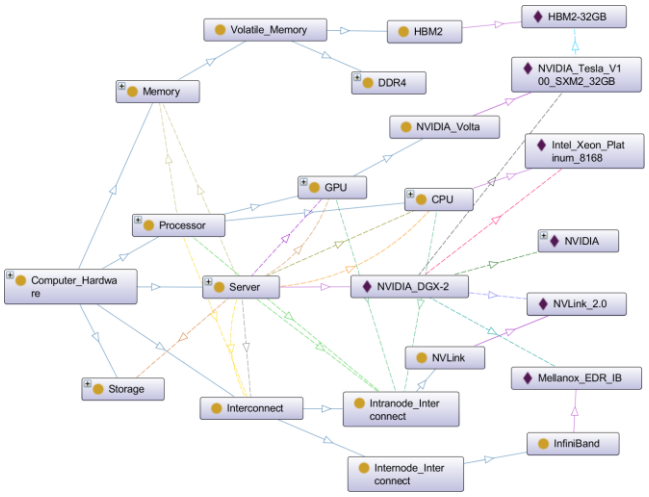

Figure 3. Example visualization of the Compass Ontology, focusing on individual *NVIDIA_DGX-2* of class *Server*. Solid arcs indicate *has subclass* relationships, and dashed arcs indicate object properties such as *hasGPUType* and *hasInternalInterconnectType*.

NVIDIA_DGX-2
URI: http://www.psc.edu/aibd/ontologies/Compass#NVIDIA_DGX-2
Different individuals:
NVIDIA_DGX-2
NVIDIA_DGX_Station
HPE_Apollo_6500_Gen10
Object property assertions:
NVIDIA_DGX-2 hasGPUType NVIDIA_Tesla_V100_SXM2_32GB
NVIDIA_DGX-2 hasDeveloper NVIDIA
NVIDIA_DGX-2 hasCPUType Intel_Xeon_Platinum_8168
NVIDIA_DGX-2 hasInternalInterconnectType NVLink_2.0
NVIDIA_DGX-2 hasExternalInterconnectType Mellanox_EDR_IB
Data property assertions:
NVIDIA_DGX-2 hasPeakSpeedFP64 "7800"^^Gf/s
NVIDIA_DGX-2 hasGPUCount "16"^^units
NVIDIA_DGX-2 hasCPUMemoryCapacity "1536"^^GB
NVIDIA_DGX-2 hasTopologyIntranode "Switch (Crossbar)"^^xsd:string
NVIDIA_DGX-2 hasCPUCount "2"^^units
NVIDIA_DGX-2 hasStorageCapacityUser "29920"^^GB
NVIDIA_DGX-2 hasBisectionBandwidth "2400.0"^^GB/s
NVIDIA_DGX-2 hasPeakSpeedTensor "2000000"^^Gf/s
NVIDIA_DGX-2 hasExternalInterconnectLinkCount "8"^^units
NVIDIA_DGX-2 hasStorageCapacitySystem "1920"^^GB
NVIDIA_DGX-2 hasPeakSpeedFP32 "251200"^^Gf/s

Figure 3. Example of instance *NVIDIA_DGX-2* of class *Server*. The URI is simply the Uniform Resource Identifier, used in OWL to build and integrate ontologies. "Different individuals" are mutually exclusive members of class *Server* (many more are in the full ontology). "Object property assertions" indicate attributes of classes, and "Data property assertions" indicate properties of individuals.

models such as training, inferencing, CNNs, and LSTMs. It also enables validating instantiated data, for example, confirming that *NVIDIA_Tesla_V100_SXM2_32GB*, an instance of class and domain *GPU*, *hasMemoryType HBM2*, an instance of class and range *Memory*.

Figure 2 illustrates a sample of the hardware class hierarchy, Figure 3 shows an example of the graph associated with individual *NVIDIA_DGX-2* and class *Server*, and Figure 4 shows some of the object and data properties of *NVIDIA_DGX-2*. In each node of the graph (Figure 3), maize circle indicates a class, and a dark purple diamond indicates an individual. Solid arcs (edges) indicate subclasses, and dashed arcs indicate object properties. Data properties, which represent instance-specific attributes, are not shown in this view. In Figure 4, an example of a data property is *NVIDIA_DGX-2* (instance) *hasGPUCount* (data property) *"16"^^units* (value). Here, ^^ indicates that "16" has datatype *units*.

The Compass Ontology defines data types such as *GB*, *GB/s*, *Gf/s*, *W*, and *units* for readability and to strongly encourage consistency, both of which reduce the potential for errors. At this time, capacities, data rates, and computational rates are expressed in gigabytes (GB), gigabytes per second (GB/s), and gigaflops per second (Gf/s) rather than mixing giga, tera, and peta. This tradeoff results in occasionally large numbers that can be converted, if desired, by tools returning the results of queries. These units greatly improve understandability of data properties, as illustrated by a display of the *NVIDIA_DGX-2* individual (Figure 4). For example, consistent use of datatype *GB/s* for data rates is shown by *NVIDIA_DGX-2 hasBisectionBandwidth "2400"^^GB/s*, ensuring that there is no confusion for subsequent inferencing and calculations.

Examples of questions the Compass Ontology is designed to support are as follows: "What are the differences between two architecture classes, for example, systems, servers, or GPUs?" "What minimal set of performance experiments will provide a reasonably complete set of insights?" "How do differences in performance correlate with differences in architecture?"

2.4 Technologies Addressed

The Compass Lab roadmap addresses a wide range of devices, including many of types that will only become available in coming years. Technology classes include processors (e.g., CPUs, GPUs, FPGAs, and neuromorphic and quantum processors), memory (e.g., new nonvolatile and volatile technologies, new form factors, and integration with interconnect fabrics and processors), and interconnects.

These preliminary results address several important, current technologies to illustrate the approach and serve as a baseline. These technologies include two GPU architectures, (Volta and Pascal); two generations of high-bandwidth memory; Tensor Cores; different GPU interconnects (NVLink 2.0, PCI Express

Table 1. Technologies included in this study, including types of servers in which they are implemented and relevant details regarding GPUs, GPU interconnects, CPUs, and local storage for training data.

	Server Type 1	Server Type 2	Server Type 3	Server Type 4
Server type	NVIDIA DGX-2	HPE Apollo 2000 Gen10	NVIDIA DGX Station	HPE Apollo 2000 Gen9
GPU type	NVIDIA Tesla V100	NVIDIA Tesla V100	NVIDIA Tesla V100	NVIDIA Tesla P100
GPU architecture	Volta	Volta	Volta	Pascal
GPU count	16	8	4	2
CUDA cores	5120	5120	5120	3584
Tensor cores	640	640	640	0
GPU memory type	HBM2	HBM2	HBM2	CoWoS HBM2
GPU memory capacity, per GPU	32 GB	16 GB	32 GB	16 GB
GPU memory bandwidth, per GPU	900 GB/s	900 GB/s	900 GB/s	732 GB/s
GPU interconnect type	NVLink 2.0 + NVSwitch	NVLink 2.0	NVLink 2.0	PCIe3 ×16
GPU interconnect bandwidth, per Link	50 GB/s	50 GB/s	50 GB/s	16 GB/s
GPU interconnect bandwidth, aggregate	300 GB/s	300 GB/s	300 GB/s	16 GB/s
GPU interconnect bisectional Bandwidth	2.4 TB/s	300 GB/s	200 GB/s	16 GB/s
GPU interconnect topology	Fully Connected	Hybrid Cube-Mesh	Fully Connected+	Link, through PCIe
CPU type	Xeon Platinum 8168	Intel Xeon Gold 6148	Intel Xeon E5-2698 v4	Intel E5-2683 v4
CPU microarchitecture	Skylake	Skylake	Broadwell	Broadwell
CPU cores	24	20	20	16
CPU clock, base	2.7 GHz	2.4 GHz	2.2 GHz	2.1 GHz
CPU count	2	2	2	2
CPU memory type	DDR4-2666	DDR4-2666	DDR4-2400	DDR4-2400
CPU memory capacity	1.5 TB	192 GB	256 GB	128 GB
CPU memory maximum bandwidth	128 GB/s	128 GB/s	76.8 GB/s	76.8 GB/s
Persistent local storage type	NVMe SSD	NVMe SSD	NVMe SSD	HDD
Persistent local storage capacity	8× 3.84 TB	4× 1.92 TB	3× 1.92 TB	4 TB
Native operating system (OS)	Ubuntu 18	CentOS 7.4	Ubuntu 18	CentOS 7.4

Gen 3 (PCIe3)); different topologies connecting GPUs (fully connected, hybrid cube-mesh, indirect through PCIe3); different storage technologies to hold training data (NVM Express solid state disk (NVMe SSD) and hard disk drive (HDD)); different operating systems (Ubuntu and CentOS); and native vs. containerized execution. These technologies are implemented in four types of servers: 1) NVIDIA DGX-2, 2) NVIDIA DGX Station, 3) HPE Apollo 6500 Gen10 server with NVIDIA Tesla V100 SXM2 GPUs, and 4) HPE Apollo 2000 Gen9 server with NVIDIA Tesla P100 PCIe GPUs. Table 1 summarizes their technical characteristics that are relevant to performance measurements reported here.

These technologies differ in several important ways. Larger GPU memory capacities allow larger batch sizes for deep learning training and potentially higher performance. Different GPU interconnects and topologies affect the rate at which information can be exchanged between GPUs. The effect on performance varies with algorithm, ranging from weak, for algorithms requiring low communication (e.g., communicating in a ring), to strong, for algorithms requiring intense communications (e.g., all-to-all communications). Different storage technologies affect performance of training from large datasets, where the performance of the storage subsystem is expected to be more important for training involving non-sequential reads. Other factors are also salient. It is important to note that performance cannot be measured independently for all variables because servers cannot be configured with arbitrary combinations of technology. We therefore focus on carefully-selected experiments, for which the Compass Ontology will be increasingly important as the technology ecosystem becomes increasingly complex, and combine information from complementary experiments to infer relationships that cannot be directly measured.

2.5 Performance Experiments

Open Compass addresses both training and inferencing, initially focusing on training neural networks for image processing for its familiarity to the community and to document our methodology. Subsequent publications will address research networks for which there is no baseline and little or no community experience.

Section 3 discusses performance results for training five neural networks: InceptionV3 [28], ResNet-50 [5], ResNet-152 [5], and VGG16 [29], and AlexNet [3], all using synthetic data. These networks differ significantly in their layers and topologies and therefore have different memory and computational requirements. In practice, each has proven valuable for image classification. Initially, we focus on training because training speed is of great importance to developing deep learning approaches to research data.

Experiments reported here were executed on the four server types detailed in Table 1, allowing analysis of how training speed depends on GPU type, amount of GPU (HBM2) memory, batch size, interconnect topology, bandwidth to data storage, and containerized versus bare-metal execution. Due to space

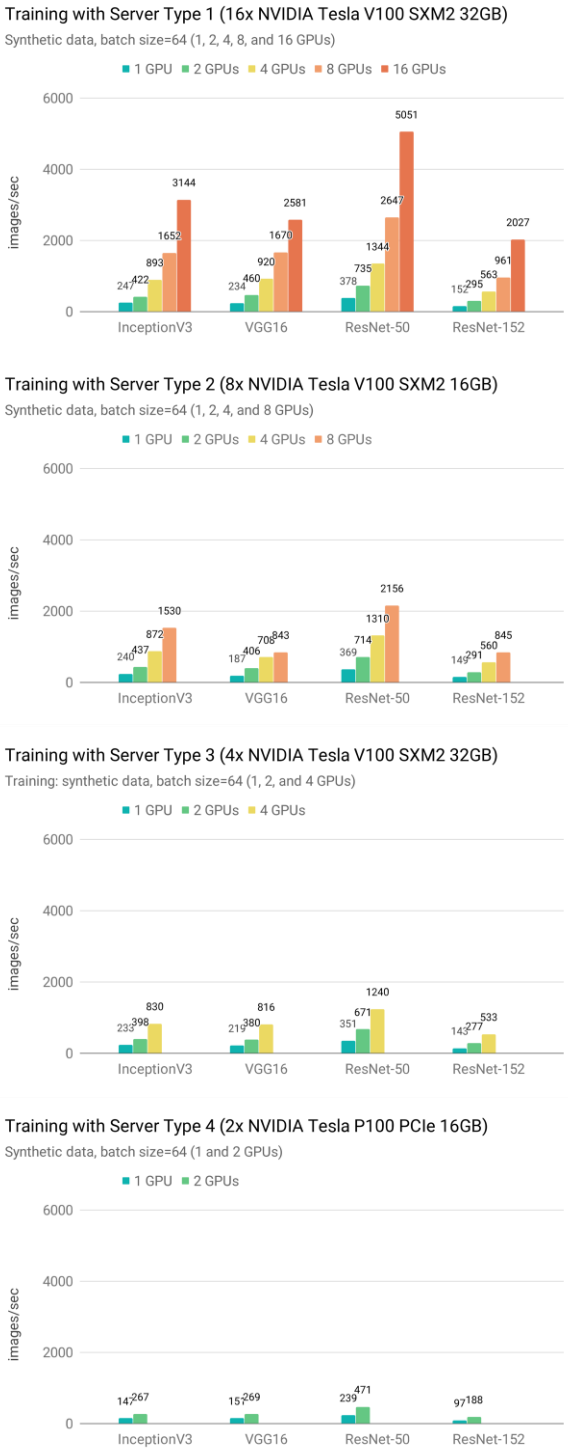

Figure 5. Training rates, measured in images per second, for TensorFlow 1.10 running networks InceptionV3, VGG16, ResNet-50, and ResNet-151 using NGC containers on Server Types 1-4 (as detailed in Table 1). Not shown are results for AlexNet, which are much higher and difficult to show on the same axes (e.g, 24,751 images per second for 8 GPUs running on Server Type 1).

limitations, here we report only a subset of the results, deferring others to a subsequent publication.

2.6 Performance Measurement Protocol

Open Compass performance measurements use a carefully-designed workflow that systematically orchestrates the large volume of runs required for performance experiments, including repetition of runs to determine statistical variation. Up to five runs were conducted for each measurement, and then the times were averaged. For each test, the first 10 steps are used as warm-up and then the next 100 steps are sampled and averaged to produce the test result. Ensembles of runs are scripted to record all salient metadata together with the actual performance metrics. A batch system (e.g., Slurm) is used when possible to maximize use of resources. All performance experiments are conducted on otherwise quiet servers, i.e., without any other jobs running on them.

For the image classification tasks, we measure performance in terms of the number of data items processed per second.

3 Results and Discussion

We start by considering two specific sets of correlations: training speeds for different architectures and numbers of GPUs (Figure 5), and training speeds as a function of batch size and GPU memory capacity (Figure 6). Figure 6 shows results for the network AlexNet.

The results depicted in Figure 5 for P100 GPUs are in agreement with the numbers published by TensorFlow in their official site [30]. The results of this paper augment the TensorFlow benchmarks by providing speed values for NVIDIA Tesla V100 SXM2 GPUs with 16GB and 32GB of HBM2 memory. We can observe that the ratio at which the speed increases is in some cases almost linearly proportional to the increase in number of GPUs. The ratio for the speed increase is influenced by the specific network architecture, the hardware architecture, and the value of hyperparameters such as the batch size.

In Figure 6, it can be seen that parameters such as batch size are directly proportional to the training speed up to a certain point. For certain networks, like AlexNet, there is an optimal number for the batch size after which further increases in batch size result in decreases in training speed. Similar results were found for Server Type 1 and Server Type 3. It was not possible to evidence this effect in Server Type 2 given that its 16GB GPU memory does support a batch size larger than 2048.

Another interesting insight observed in Figure 6 is that at the optimal batch size (in this case, 2048), doubling the number of GPUs increases training speed by almost a factor of two. This clearly shows the importance of determining the optimal batch size when planning to scale up training by augmenting the number of GPUs.

The results in Figure 6 also evidence the effect of the GPU interconnect topology in the increase (or decrease) of training speed when doubling the number of GPUs assigned to a specific

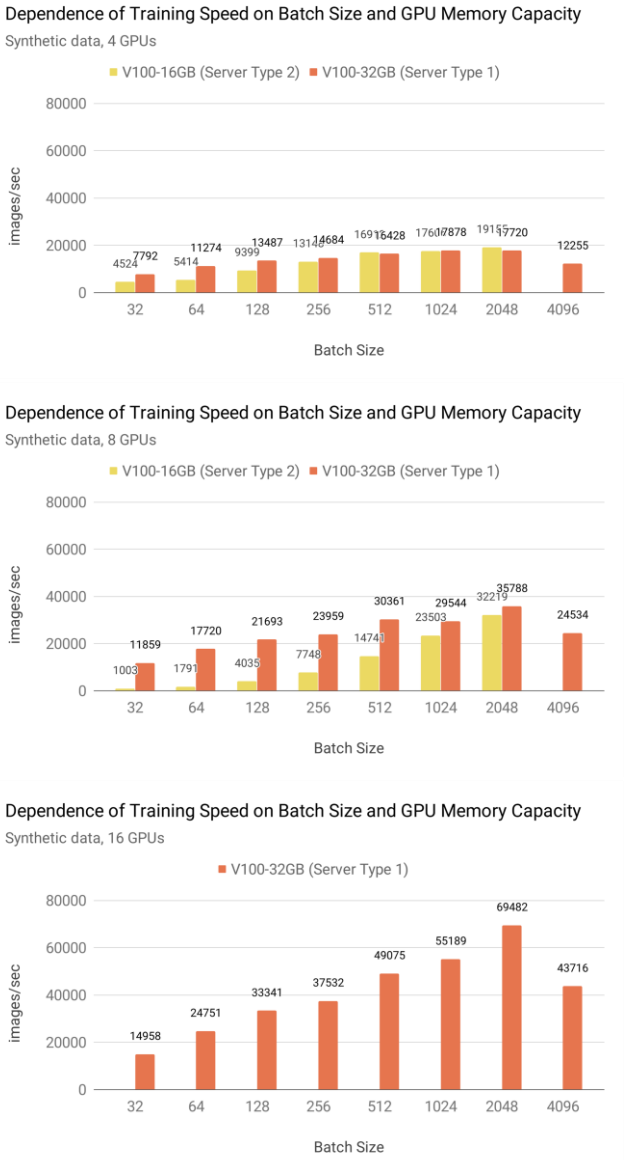

Figure 6. Training speed increases significantly with batch size, reaching a maximum at batch size of 2048, as seen for 4, 8, and 16 GPUs (top, middle, and bottom). Training speed is also generally enhanced by larger HBM2 memory on the GPUs, reflected by performance shown in the top two graphs for 16GB (yellow) and 32GB (orange) HBM2 memory.

task. It can be seen when comparing the upper and middle graphs for 4 and 8 GPUs that, for batch sizes of 512 or less, an increase from 4 to 8 GPUs in the Server Type 2 results in a decrease of training speed. This is an example of communication overhead decreasing the performance of a system, a subtle effect that reinforces the need for deep learning practitioners to understand how performance can depend on computer architecture.

We explored the influence of the batch size and the number of GPUs on training performance. It was found that different hardware topologies and the specific batch size influence the

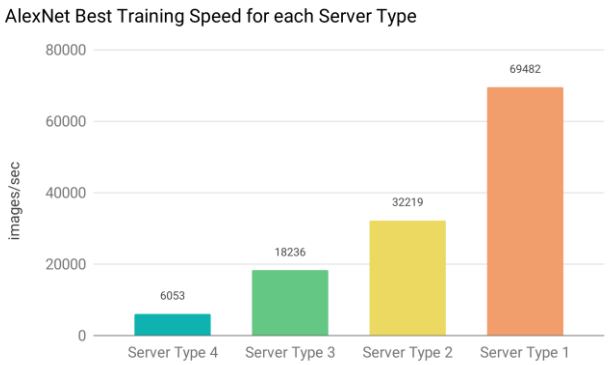

Figure 7. AlexNet best training speed for each of the four server types. The batch size is optimized to use each system in full.

effect of adding more GPUs to a training task, in agreement with observations of Canziani e*t al.* [31].

We also examined the four server types in terms of the best performance each can produce when training the neural network AlexNet. We considered the optimal batch size for the network and the optimal number of GPUs. The results, shown in Figure 7, provide an estimate of the best performance in terms of training speed that each of these server types can provide when training AlexNet. As expected, Server Type 1 (16 V100 GPUs) leads, with a speed over two-fold greater than Server Type 2 (8 V100 GPUs). Server Type 3, with 4 V100 GPUS, is third and exhibits a training speed of over half that of the Server Type 2. Server Type 3 has a speed around three times that of Server Type 4 (2 P100 GPUs), clearly demonstrating the performance gains introduced in Volta through Tensor Cores and other architectural advances.

4 Conclusion

Open Compass is enabling exploratory research leveraging the Compass Lab, an advanced engineering testbed for artificial intelligence, to develop understanding of the benefits of emerging AI technologies for the open science community. The Compass Ontology provides a detailed knowledge base of AI hardware and will become increasingly important as many new architectures that are currently under development become available. Early results of Open Compass are encouraging, revealing important insights into the relationships between deep learning training speed, memory capacity, and batch size.

ACKNOWLEDGMENTS

Open Compass is supported by National Science Foundation award OAC-1833317. We are thankful to Hewlett Packard Enterprise and NVIDIA for their support of the Compass Lab.

REFERENCES

[1] Y. Shoham, R. Perrault, E. Brynjolfsson and J. Clark 2017. *AI Index 2017 Annual Report*. Stanford University.

[2] Y. Shoham, R. Perrault, E. Brynjolfsson, J.C. Openai, J. Manyika, J.C. Niebles, T. Lyons, J. Etchemendy, B. Grosz and Z. Bauer 2018. *AI Index 2018 Annual Report*.

[3] A. Krizhevsky, I. Sutskever and G.E. Hinton 2012. ImageNet Classification with Deep Convolutional Neural Networks. *Advances in Neural Information Processing Systems 25* (2012), 1097–1105.

[4] C. Szegedy, L. Wei, J. Yangqing, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke and A. Rabinovich 2015. Going deeper with convolutions. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2015), 1–9.

[5] K. He, X. Zhang, S. Ren and J. Sun 2016. Deep Residual Learning for Image Recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016), 770–778.

[6] H. Sak, A. Senior and F. Beaufays 2014. Long Short-Term Memory Based Recurrent Neural Network Architectures for Large Vocabulary Speech Recognition. *arXiv:1402.1128*. (2014).

[7] N. Brown and T. Sandholm 2017. Safe and Nested Subgame Solving for Imperfect-Information Games. *Advances in Neural Information Processing Systems 30*. I. Guyon, U. V Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, eds. Curran Associates, Inc. 689–699.

[8] N. Brown and T. Sandholm 2017. Superhuman AI for heads-up no-limit poker: Libratus beats top professionals. *Science*. (2017). DOI:https://doi.org/10.1126/science.aao1733.

[9] N.A. Nystrom, P.A. Buitrago and P.D. Blood 2019. Bridges: Converging HPC, AI, and Big Data for Enabling Discovery. *Contemporary High Performance Computing: From Petascale toward Exascale, Vol. 3*. J.S. Vetter, ed. CRC Press.

[10] N.A. Nystrom, M.J. Levine, R.Z. Roskies and J.R. Scott 2015. Bridges: a uniquely flexible HPC resource for new communities and data analytics. *Proceedings of the 2015 XSEDE Conference: Scientific Advancements Enabled by Enhanced Cyberinfrastructure*. ACM.

[11] I.J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville and Y. Bengio 2014. Generative Adversarial Nets. *Advances in Neural Information Processing Systems 27* (2014), 2672–2680.

[12] J. Deng, W. Dong, R. Socher, L. Li, K. Li and L. Fei-Fei 2009. ImageNet: A Large-Scale Hierarchical Image Database. *2009 IEEE Conference on Computer Vision and Pattern Recognition* (2009), 248–255.

[13] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A.C. Berg and L. Fei-Fei 2015. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*. 115, 3 (2015), 211–252. DOI:https://doi.org/10.1007/s11263-015-0816-y.

[14] R. Adolf, S. Rama, B. Reagen, G. Wei and D. Brooks 2016. Fathom: reference workloads for modern deep learning methods. *2016 IEEE International Symposium on Workload Characterization (IISWC)* (2016), 1–10.

[15] MLPerf: A broad ML benchmark suite for measuring performance of ML software frameworks, ML hardware accelerators, and ML cloud platforms: 2018. *https://mlperf.org/*. Accessed: 2019-04-06.

[16] C. Coleman, D. Narayanan, D. Kang, T. Zhao, J. Zhang, L. Nardi, P. Bailis, K. Olukotun, C. Ré and M. Zaharia 2017. DAWNBench: An End-to-End Deep Learning Benchmark and Competition. *Thirty-first Annual Conference on Neural Information Processing Systems (NIPS)*. Nips (2017).

[17] HPE Deep Learning Cookbook: 2018. *https://developer.hpe.com/platform/hpe-deep-learning-cookbook/home*. Accessed: 2019-04-06.

[18] A.K. AL Hwaitat, A. Shaheen, K. Adhim, E.N. Arkebat and A.A. AL Hwiatat 2018. Computer Hardware Components Ontology. *Modern Applied Science*. 12, 3 (2018), 35–40. DOI:https://doi.org/10.5539/mas.v12n3p35.

[19] M.H. Faheem, B. König-Ries, A.M. Aslam, R.N. Aljohani and I. Katib 2018. Ontology Design for Solving Computationally-Intensive Problems on Heterogeneous Architectures. *Sustainability*.

[20] Y. LeCun, Y. Bengio and G. Hinton 2015. Deep learning. *Nature*. 521, 7553 (2015), 436–444.

[21] I. Goodfellow, Y. Bengio and A. Courville 2016. *Deep Learning*. The MIT Press.

[22] D. Silver et al. 2016. Mastering the game of Go with deep neural networks and tree search. *Nature*. 529, 7587 (Oct. 2016), 484–489. DOI:https://doi.org/10.1038/nature16961.

[23] B. Le Cun, Y Boser, J.S. Denker, R.E. Howard, W. Habbard, L.D. Jackel and D. Henderson 1990. Handwritten Digit Recognition with a Back-propagation Network. D.S. Touretzky, ed. Morgan Kaufmann Publishers Inc. 396–404.

[24] M.I. Jordan 1986. *Serial Order: A Parallel Distributed Processing Approach*.

[25] S. Sabour, N. Frosst and G.E. Hinton 2017. Dynamic Routing Between Capsules. *Adv. in Neural Information Processing Systems 30 (NIPS 2017)* (2017).

[26] M.A. Musen 2015. The Protégé Project: A Look Back and a Look Forward. *{AI} Matters*. 1, 4 (2015), 4–12. DOI:https://doi.org/10.1145/2757001.2757003.

[27] OWL Web Ontology Language Overview: 2004. *https://www.w3.org/TR/2004/REC-owl-features-20040210/*.

[28] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens and Z. Wojna 2016. Rethinking the Inception Architecture for Computer Vision. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016), 2818–2826.

[29] S. Liu and W. Deng 2015. Very deep convolutional neural network based image classification using small training sample size. *2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR)* (2015), 730–734.
[30] TensorFlow Benchmarks: 2018. *https://www.tensorflow.org/guide/performance/benchmarkshttps://www.tensorflow.org/guide/performance/benchmarks*.
[31] A. Canziani, A. Paszke and E. Culurciello 2016. An Analysis of Deep Neural Network Models for Practical Applications. (2016), 1–7.